# Enhancing Physical Modeling with Interpretable Physics-Aware Machine Learning

by

Christian S. Jacobsen

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in the University of Michigan
2024

Doctoral Committee:

Professor Karthik Duraisamy, Chair
Professor Krzysztof Fidkowski
Assistant Professor Alex Gorodetsky
Assistant Professor Xun Huan

Christian S. Jacobsen
csjacobs@umich.edu
ORCID iD: 0000-0002-1394-5893

# DEDICATION

*Dedicated to my parents.*

# ACKNOWLEDGEMENTS

I would like to first extend my deepest gratitude to my advisor, Professor Karthik Duraisamy, for his invaluable help, guidance, motivation, and encouragement throughout my journey at the University of Michigan. His unwavering support has been instrumental in shaping my research path and guiding the trajectory of my future career. We have spent countless hours over the past few year discussing research, interests outside of research, and variety of other topics. He has always been very supportive and understanding, not only encouraging my research interests but also accommodating my desire to travel to destinations interesting to me. My time under Professor Duraisamy's mentorship has been an incredibly rewarding experience, and for that, I will always be thankful.

Second, I would like to thank the other members of my committee, Professors Krzysztof Fidkowski, Alex Gorodetsky, and Xun Huan for all of their time, support, and assistance in planning and completing the thesis. All of their comments and insights regarding the work itself are extremely valuable in the development of the final work.

An additional thank you goes out to Professor Marco Panesi and Ivan Zanardi at the University of Illinois at Urbana Champaign for the extensive research collaborations that we have undertaken in the past two years. Without their expertise in modeling non-equilibrium flows, a portion of this dissertation would not have been possible. During my visit to UIUC and theirs to UM, we have accomplished much, and I am grateful for their help and insight during our time of collaboration.

During the initial years of my research, I was drawn towards a career in the industry, with a growing interest in AI/ML research. It was during this period that I had the privilege of undertaking two internships at NVIDIA, experiences that were not only immensely educational but also immensely enjoyable. These internships were pivotal in clarifying the direction of my career and have left a significant mark on the work presented in this dissertation. I am grateful to Alice and Xunlei for their guidance and collaborative spirit during my time at NVIDIA. A special word of thanks goes to Shalini, whose guidance, insight, encouragement, and advice over the past few years have been invaluable.

The memories created with the friends I have made in Ann Arbor and the members of the Computational Aerosciences Laboratory (CASLAB) will always hold a special place in

my heart. A special thank you to Prit, Kim, Simon, Andrew, Rakesh, Jordan, Ayoub, Chris, Nick, Vishal, Sahil, and Jasmin for the friendship and unforgettable moments we have shared in the past five years as we look forward to many more to come.

Most of all, I owe a debt of gratitude to my parents for their steadfast support and belief in me, especially during the most challenging times of my studies. Their love and encouragement have been a primary source of motivation for me throughout my time in Ann Arbor.

Finally, I would like to extend a thank you to all of those that have helped me along the way and made my time truly enjoyable towards the completion of my studies at the University of Michigan.

# TABLE OF CONTENTS

# LIST OF FIGURES

FIGURE

# LIST OF TABLES

TABLE

# LIST OF APPENDICES

# ABSTRACT

The burgeoning intersection of machine learning (ML) with physics has catalyzed a transformative approach to physical modeling marked by an enhanced capacity for innovation and discovery. Traditional applications of ML in physics often grapple with a critical challenge: predictions frequently lack transparency and interpretability, ultimately compromising generalizability and hindering the diagnosis of limitations. This opacity reduces the ability to develop generalized physical models and extract deeper insights into the underlying physical processes. This work aims to harness the capabilities of ML to learn from data while enhancing the interpretability of its models by creating physics-aware models. In doing so, it seeks to facilitate deeper analyses and enable more profound insights into physical phenomena, developing models which improve accuracy and generalizability over baseline methods. Through a series of studies, this work delves into the development and application of physically interpretable models in the realm of physics-aware machine learning. Each of the works, while distinct in their focus and application, collectively contributes to the advancement of interpretable physics-informed modeling with machine learning, addressing both theoretical and practical aspects.

The first research application harnesses Variational Autoencoders (VAEs), a generative modeling technique, for non-linear dimensionality reduction in computational physics. This approach applies VAEs to compress complex, high-dimensional data sets into more interpretable low-dimensional latent variables which are correlated with physically-relevant quantities governing the generation of data itself. The primary objective is to isolate independent physical parameters that govern the generation of these data sets in an unsupervised manner. Achieving a balance between high reconstruction accuracy and meaningful disentangled and physically-correlated representations, the study demonstrates the efficacy of using hierarchical priors in enhancing the interpretability of learned representations within physics-based contexts. This work exemplifies the merging of deep learning techniques with the principles of physical interpretability, effectively extracting physically-relevant parameters from data without any knowledge of the physics. However, it is observed that incorporating prior knowledge of the physical system into the learning process significantly enhances the robustness of learning such representations. This insight has guided the design of models in

subsequent studies within this work, where a foundational understanding of the physical system is integrated from the outset to improve model performance and interpretability.

The second study expands the horizon of ML applications in physics by introducing a host of data-driven methods aimed at enhancing the modeling of physical systems, specifically focusing on the process of cathodic electrophoretic deposition, or e-coat. Here, the method systematically identifies and addresses limitations inherent in traditional physical models through a combination of variational inference techniques and ML enhancements. The underlying physical phenomena responsible for the onset of film deposition are not well understood, and this study aims to augment the currently modeled mechanisms with additional behavior observed in experimental data. The incorporation of neural networks trained as Neural Ordinary Differential Equations (Neural ODEs) into the modeling process is performed by augmenting a baseline model with flexible yet physically interpretable forms based on physical insight. This integrated approach demonstrates how ML-augmented models can more accurately capture observed behaviors in physical systems, thus enhancing both their predictive power and generalizability while maintaining interpretability. The study showcases how careful integration of ML with physics can lead to advancements in the accuracy and generalizability of physical models, particularly when the underlying mechanisms are not well understood.

In a further extension of this interdisciplinary approach, the third study focuses on the development of reduced order models for non equilibrium gas dynamics. Challenging existing methods of cluster assignment in reduced order models (ROMs) for maximum entropy-based coarse-graining approaches, the method introduces a novel ML framework to learn cluster assignments where the optimization is informed by rate-distortion theory to improve robustness. The methodology transforms the discrete optimization problem of cluster assignments into a probabilistic and continuous form, leading to the development of a fully differentiable, adjoint-driven dynamics solver to facilitate the use of gradient-based optimization techniques. The end-to-end differentiability of this framework allows efficient backward pass gradient computation, enhancing the training of a classifier to predict cluster assignments based on information pertinent to each of the internal states. This training is then performed using a loss function informed by rate-distortion theory to aid in finding the global minimum solution. The application of this method to the evolution of particle quantum states under non-equilibrium conditions highlights the framework's effectiveness in managing high-dimensional equations and improving the coarse-graining process. This study exemplifies how machine learning can be leveraged to optimize and refine traditional modeling processes in physics, contributing to a more nuanced and accurate understanding of physical systems.

The final study of this dissertation is the application of generative artificial intelligence

in physical problem domains, specifically focusing on the enforcement of physical laws in the form of partial differential equations (PDEs) with diffusion and score-based generative models. The study introduces a novel approach to promote consistency of generated samples with underlying PDEs in various applications of forward and inverse problems. Score-based generative models, rooted in stochastic differential equations (SDEs), are illustrated to be a flexible and robust method for several scientific machine learning tasks, ranging from surrogate modeling to probabilistic field reconstruction and inversion from sparse measurements. The ability of these models to generate high-fidelity samples that align closely with ground truth data distributions underscores their potential in advancing physics-based problem-solving. This research not only demonstrates the practical applicability of generative models in physics but also highlights the versatility and adaptability of these models in addressing a range of physical modeling needs.

Together, these studies develop a narrative that underscores the importance of interpretability in machine learning for enhancing the generalizability of models. Additionally, it focuses on the development of physics-aware machine learning models to achieve this objective. Each piece of research, distinct in its application, contributes to a unified theme of enhancing the interpretability, generalizability, and practical utility of machine learning in the realm of physics. This body of work advances our understanding of how machine learning can be effectively applied to physics problems and offers new perspectives and methodologies that can aid in many aspects of scientific exploration and discovery.

# CHAPTER 1

# Introduction

The field of computational science has undergone a significant transformation with the convergence of machine learning (ML) and physical modeling [2]. This union heralds a new era of progress, marked by enhanced computational and data-driven modeling techniques, specifically tailored for applications deeply rooted in physics. This synthesis is particularly transformative in many physics and engineering applications, where computational modeling and simulation offer unique advantages over real-world experimentation. These advantages include the ability to simulate scenarios that are impractical, expensive, dangerous, or impossible to create in physical experiments, such as extreme conditions. Computational models also allow for the exploration of vast parameter spaces at a lower cost and higher speed than traditional experimental methods. In particular, the development of computational models has significantly benefited areas such as designing and analyzing combustion systems [3], optimizing aircraft profiles [4], and predicting aerodynamic properties of airfoils [5], offering a more efficient and feasible alternative to relying solely on real-world experimentation.

Machine learning, renowned for its powerful flexibility to learn from data, can significantly enhance the efficiency and precision of computational processes. Data-driven discovery is often regarded as the *fourth paradigm* [6] of scientific innovation, and it complements the existing domains of theoretical, experimental, and numerical advancements. The application of ML techniques has been pivotal in harnessing this fourth paradigm, propelling substantial progress in scientific discovery and advancement. For instance, this approach enables the detailed analysis and optimization of aircraft designs and flight conditions [7, 8] which would be impractically time-consuming and expensive to test physically. Turbulent flow simulations [9], detection of cyber attacks on power grid controllers [10], and prediction of flight departure delays [11] have all recently benefited from the predictive power provided by ML techniques. The synergy of ML and physics thus paves the way for more sophisticated, efficient, and accurate models, potentially capable of addressing some of the most intricate and longstanding challenges in the field.

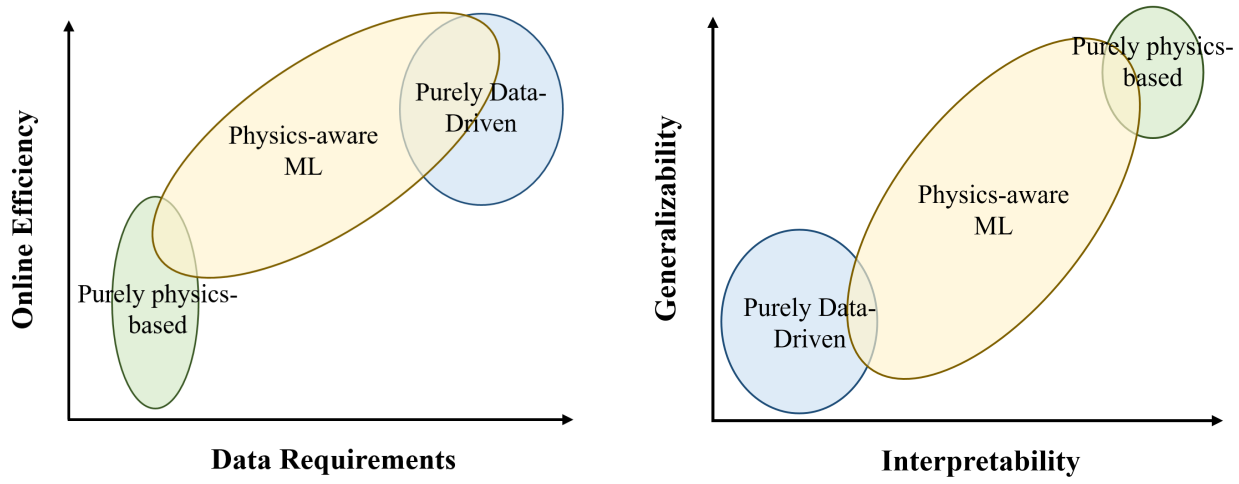



Figure 1.1: *(a)* Data-driven approaches can provide models with improved computational efficiency at the cost of requiring data *(b)* Purely data-driven models often lack generalizability and interpretability in their predictions. Physics-aware ML models aim to provide the best of both worlds - improved computational efficiency while maintaining generalizability and interpretability.

The introduction of data-driven machine learning marks a significant shift from traditional standalone physics models. This thesis explores how combining data-driven machine learning with physics has unlocked new potential in well-established areas such as reduced order modeling and surrogate modeling. These methodologies, integral to computational physics and aerospace engineering, have been enhanced by machine learning to improve their efficiency and expand their capabilities, addressing more complex and computationally demanding problems.

A focal point here is the concept of physics-aware machine learning. This approach aims at integrating physical laws and principles into ML models, ensuring that the predictions and insights gleaned are not only data-driven but also grounded in the empirical, theory-guided realm of physics [12]. This is particularly crucial in aerospace applications where accurate and interpretable models are necessary for safe and efficient system operation [2]. Figure 1.1 illustrates the trade offs that exist when modeling physical systems with purely physical insight, purely data-driven approaches, and physics-aware ML. Data-driven approaches can provide more computationally efficient simulations, but purely data-driven approaches often lack generalizability and interpretability, as further discussed and defined in Section 1.7.

This work attempts to provide a comprehensive examination of the intersection between machine learning and physical modeling, showcasing diverse areas of application. It be-

gins by establishing the background material necessary for understanding the benefits that physics-aware ML provides before detailing the unique contributions of this research through four separate studies, highlighting its significance within the broader landscape of physical modeling and machine learning.

## 1.1 Overview and Objectives

While ML offers powerful tools for data analysis and pattern recognition such as neural networks, its applications in physics are often hindered by a critical gap: the lack of transparency and interpretability in ML models. This gap presents a substantial problem, as the ability to understand and interpret the outcomes of these models is crucial in physics, a field fundamentally grounded in empirical evidence and theoretical consistency. While machine learning models have enormous capacity to learn from data, they are generally not constrained by the same set of laws which govern the true generation of data.

Physics-based applications of ML typically involve complex datasets where the underlying physical processes are deeply intertwined with data patterns themselves. Traditional ML approaches, while adept at discerning correlations through vast datasets to make accurate predictions, frequently fall short in providing insights into the 'why' behind such predictions. The models tend to operate as 'black boxes' offering little insight into the physical processes they model or the nature of their predictions. Although ML-based models greatly improve efficiency and sometimes accuracy over traditional physical models by learning from data, their opacity limits their utility in physics, where understanding the causal relationships and underlying principles is often as important as the predictive accuracy. In aerospace, for instance, when analyzing aerodynamic performance, it may not be sufficient for an ML model to predict airflow patterns; engineers need to understand the physical principles driving these patterns to safely understand improvements in aircraft design and comprehensively validate predictions. Additionally, when a model fails in some scenarios, a lack of interpretability hinders insights into potential solutions.

Another significant challenge is integrating the established laws, principles, and ideas of physics into ML models. Traditional ML models, primarily purely data-driven, often overlook the necessity to adhere to physical laws, leading to results that, while statistically sound, may be physically implausible and difficult to generalize. Such machine learning models can therefore suffer from observational bias. This bias arises due to limitations in data collection, leading to skewed, unrepresentative, or limited scope in observed data. As a result, the model's accuracy and generalizability are hampered. Observational bias can lead to models that do not truly reflect the underlying physical phenomena they are intended to represent,

thus reducing the reliability of their predictions or analyses in real-world applications. This is particularly relevant in aerospace engineering, a field where adherence to fundamental principles is paramount for safety and functionality. In the realms of fluid dynamics and material science, for instance, overlooking key principles can have serious implications. In fluid dynamics, disregarding or failure to enforce the known principles governing airflow can lead to incorrect predictions about lift and drag forces on an aircraft, which are crucial for its stable and efficient flight. Similarly, in material science, failure to properly account for physical properties such as stress and fatigue of materials can result in structural designs that are prone to failure under the extreme conditions experienced during flight. Such inaccuracies and oversights, stemming from a lack of physical grounding in machine learning models, can compromise the integrity and safety of aerospace engineering solutions, leading to potentially unsafe conclusions which could have real-world ramifications.

These ideas combined are frequently referred to *interpretability* in this work - the ability of a model to provide insights into its decision-making process in a way that is understandable to humans, specifically by aligning its operations and outputs with known physical laws and principles. This concept goes beyond mere prediction accuracy, aiming to make the internal workings of the model transparent, so that its predictions can be directly related to physical phenomena and understood in terms of physical concepts. Thus the overarching problem this thesis addresses is twofold:

- **Firstly**, it aims to illustrate methods for enhancing the interpretability of ML models in physics applications, facilitating the observance of meaningful insights into the physical phenomena being modeled.

- **Second**, it aims to develop and implement physics-aware ML models that respect and incorporate fundamental physical principles but also advance the accuracy and/or efficiency of state of the art models.

These two ideas are often coupled together and are demonstrated in tandem through much of the work presented.

In the context of aerospace engineering, adopting an approach that enhances the interpretability and physical consistency of machine learning models could lead to significant advancements in various critical areas. For instance, in the design of flight control systems, this approach could enable the development of more sophisticated algorithms that not only respond effectively to real-time flight data but also provide insights into the underlying aerodynamic principles affecting flight dynamics. This could result in control systems that are not only more reliable and responsive but also easier to diagnose and maintain, ultimately

improving flight safety and performance. Furthermore, in the realm of fuel efficiency optimization, such machine learning models could offer a more nuanced understanding of the complex interplay between aircraft design, operational parameters, and environmental conditions. By accurately modeling these factors and understanding their physical basis, it's possible to identify optimal flight paths, speeds, and configurations that minimize fuel consumption without compromising safety or performance. This could lead to more fuel-efficient flight operations, significantly reducing operational costs and environmental impact.

This work is thus aimed towards devising and elucidating methodologies that bridge the gap between the inherently empirical, theory-informed domain of physics and the rapidly evolving field of purely data-driven machine learning. By strategically incorporating ML into physical models and harnessing the power of data, we aim to significantly advance the predictive accuracy and efficiency of physics-based research and applications. Addressing these pivotal challenges is essential not only for advancing the fields of physics and machine learning, but also for ensuring that advancements in ML are optimally utilized to enhance the outcomes in applications of physical modeling. Central to this goal is the enhancement of interpretability and physical consistency within ML models, which is critical for establishing a framework for more effective, dependable, and scientifically robust applications of machine learning within physical modeling. This approach not only fortifies the theoretical foundations of physics-aware machine learning but also may inspire future innovations across a spectrum of physics-driven applications, thereby advancing the goals of achieving superior predictive accuracy and operational efficiency through the integration of ML into physical modeling.

## 1.2 Forward and Inverse Problems

The concepts of forward and inverse problems form a cornerstone of understanding, analyzing, and modeling complex systems in physics and engineering. Many practical problems can be framed in the context of either a forward or an inverse problem. These problems, inherently different in their scope, approach, and objectives, are fundamental to a range of applied engineering solutions.

Forward problems in physics involve predicting the outcomes or effects of a given set of causes or conditions. Essentially, they start with a known set of parameters and governing equations of a system and seek to compute the consequent state or behavior. These problems are direct in nature; given the initial conditions, boundary conditions, and the governing laws (such as in the form of differential equations), the goal is to determine the future state of the system. Forward problems are prevalent in scenarios such as predicting the trajectory

of a celestial body, simulating climate patterns, or modeling the behavior of materials under stress.

We consider a general function $\mathbf{f}$ which models and encapsulates physical laws or dynamics of a system of interest. Forward problems defined in this context are constructed to predict

$$\mathbf{y} = \mathbf{f}(\mathbf{x}; \boldsymbol{\eta}) \,, \tag{1.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ represents the known inputs or conditions of the system, $\boldsymbol{\eta} \in \mathbb{R}^p$ denotes known parameters governing the system, and $\mathbf{y} \in \mathbb{R}^m$ is the output or state to be predicted.

Conversely, inverse problems address deducing the unknown causes, conditions, or parameters from observed outcomes. They are often inherently more complex and sometimes even ill-posed, as they involve working backward from the effects to infer the underlying causes. Inverse problems are fundamental in situations where direct measurement of a system's parameters is impossible or impractical. Examples include reconstructing the internal structure of the Earth from seismic data, determining the properties of a star from its emitted light, or medical imaging techniques like MRI and CT scans, where the internal structure of the body is inferred from external observations.

In this case, the parameters $\boldsymbol{\eta}$ are unknown, and the goal is to invert the model to find the parameters of a system given observed inputs $\mathbf{x}$ and outputs $\mathbf{y}$ in the form of data.

While forward problems allow us to predict and prepare for future scenarios based on our current understanding, inverse problems enable us to uncover hidden information and deepen that understanding. However, inverse problems are often more challenging due to issues such as non-uniqueness and being ill-posed. Successfully addressing these challenges not only enhances our predictive capabilities but also enriches our comprehension of the underlying principles governing various physical phenomena. In the following sections, methodologies and tools used to tackle these problems are explored, in particular with an emphasis on machine learning methods. These discussions set the stage for understanding the pitfalls of machine learning techniques used for solving forward and inverse problems in physics, allowing insights into avoiding such pitfalls.

## 1.3 Surrogate Modeling

Another primary concept in the domain of computational physics and engineering is that of surrogate modeling. Surrogate models are approximate models used to emulate the behavior of more complex, computationally expensive simulations or unknown real-world processes. In essence, surrogate modeling serves as a bridge between high-fidelity models, which are

accurate but computationally intensive, and the need for quicker, more efficient computational methods. This is particularly useful in scenarios where repeated simulations are required. For instance, many inverse problems such as optimization, parameter estimation, and uncertainty quantification can require numerous forward evaluations - which may not be feasible to perform without an efficient surrogate model. By creating such an approximating surrogate, computational burdens can be significantly reduced, enabling faster predictions and unlocking additional applications for analysis. For instance, surrogate models have been employed in the development of digital twins due to the natural abundance of data available for developing such frameworks [13]. Even in cases where experimental data may be available without rigorous theoretic understanding of the particular physical processes at hand, surrogate models may be created to model the process itself.

Surrogate models are typically constructed by employing various machine learning and/or probabilistic techniques to learn the underlying relationship between inputs and outputs of the actual model or process, where learning such relationships requires information about the system in the form of data. Techniques ranging from polynomial chaos expansion [14] to more complex machine learning algorithms, including Gaussian processes [15, 16] and neural networks [17, 18], are often utilized to build these models. However, this work focuses primarily on the use of neural networks due to the flexible, expressive, and scalable nature of their forms.

Surrogate modeling aims to construct an approximate model, $\hat{\mathbf{f}}(\mathbf{x};\boldsymbol{\eta})$, that emulates the behavior of a more complex, computationally more expensive, or unmodeled process $\mathbf{f}(\mathbf{x};\boldsymbol{\eta})$. Let $\mathbf{x} \in \mathbb{R}^n$ represent the input parameters of the model, $\boldsymbol{\eta} \in \mathbb{R}^p$ parameters governing the system, and $\mathbf{y} \in \mathbb{R}^m$ the outputs. The process $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ thus maps the inputs to the outputs, i.e., $\mathbf{y} = \mathbf{f}(\mathbf{x};\boldsymbol{\eta})$. The surrogate model $\hat{\mathbf{f}} : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^m$ is then constructed such that it approximates $\mathbf{f}$ as closely as possible, i.e. $\hat{\mathbf{f}}(\mathbf{x};\boldsymbol{\eta}) \approx \mathbf{f}(\mathbf{x};\boldsymbol{\eta})$. The approximation quality of $\hat{\mathbf{f}}$ for a given input is often measured in terms of some error metric $d : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ which quantifies the difference between the outputs of the surrogate model and the original model.

The construction of $\hat{\mathbf{f}}$ often involves selecting a parametric functional form $\hat{\mathbf{f}}_\phi$ for the surrogate with parameters $\phi$ to approximate the true process. This parametric form can be selected as a complex neural network-based architectures with potentially millions of parameters, for example. These parameters are then determined through training via optimization algorithms such as stochastic gradient descent (SGD). The process employs a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, \boldsymbol{\eta}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$ to learn the parameters of the surrogate, where $N$ is the number of data points, and each $\mathbf{y}^{(i)}$ is obtained by sampling from the original model either through real-world processes or computing the true process $\mathbf{y}^{(i)} = \mathbf{f}(\mathbf{x}^{(i)};\boldsymbol{\eta}^{(i)})$. The goal is

to find the parameters $\theta$ which minimize average error metric $d$ over the dataset $\mathcal{D}$.

Creating a surrogate model can benefit in both forward and inverse applications. In the former, the surrogate approximates the forward process as $\hat{\mathbf{f}}_\phi(\mathbf{x};\boldsymbol{\eta}) \approx \mathbf{f}(\mathbf{x};\boldsymbol{\eta})$. An example use case for surrogate modeling in forward applications is to model system dynamics using a lightweight, computationally efficient model. This may be useful in real time prediction and control applications in which the forward model is too expensive to evaluate in real time. Suppose a physical system is modeled by some partial differential equation (PDE) describing the dynamics of the system in time

$$\frac{\partial \mathbf{u}}{\partial t} = \mathcal{F}(\mathbf{u}(t), t; \boldsymbol{\eta}) \ , \tag{1.2}$$

where $\mathbf{u} \in \mathbb{R}^n$ is the physical state and $\mathcal{F}$ is an operator typically involving gradients of the physical state variable. Supposing that operator $\mathcal{F}$ is computationally demanding to evaluate, a surrogate model may aim to approximate it with a computationally efficient parametric function $\hat{\mathbf{f}}_\phi(\mathbf{u}(t), t; \boldsymbol{\eta}) \approx \mathcal{F}(\mathbf{u}(t), t; \boldsymbol{\eta})$ to approximate the dynamics as

$$\frac{\partial \mathbf{u}}{\partial t} = \hat{\mathbf{f}}_\phi(\mathbf{u}(t), t; \eta) \ . \tag{1.3}$$

A particular method of training this type of surrogate model is discussed in Section 2.4.

Inverse problems can also benefit greatly from surrogate models. In this case, such a surrogate may approximate a forward model as $\hat{\mathbf{f}}_\phi(\mathbf{x}, \boldsymbol{\eta}) \approx \mathbf{f}(\mathbf{x}, \boldsymbol{\eta})$ to more efficiently determine system parameters. For example, consider the case in which a system is modeled by some forward process $\mathbf{y} = \mathbf{f}(\mathbf{x}; \boldsymbol{\eta})$, and noisy real world data is obtained in the form of a set of input-output pairs $\mathcal{D}_r = \{(\mathbf{x}_r^{(i)}, \mathbf{y}_r^{(i)})\}_{i=1}^{N_r}$. It is assumed that the real world process adheres to the model with constant but unknown model parameters $\boldsymbol{\eta}_r$. Inferring the model parameters from data may require a large number of forward model evaluations (Section 1.5.2). If the forward model $\mathbf{f}(\mathbf{x}; \boldsymbol{\eta})$ is computational expensive, a surrogate model can be employed to potentially improve the efficiency of the parameter inference process. Suppose that a dataset is created by computing the forward model at a variety of input conditions to create a dataset $\mathcal{D}_s = \{(\mathbf{x}_s^{(i)}, \boldsymbol{\eta}_s^{(i)}, \mathbf{y}_s^{(i)})\}_{i=1}^{N_s}$. This dataset can then be leveraged to create a surrogate model of the expensive forward process. An example of how to train such a surrogate model's parameters is to solve the optimization problem

$$\min_\phi \frac{1}{N_s} \sum_{i=1}^{N_s} \left\| \mathbf{y}_s^{(i)} - \hat{\mathbf{f}}_\phi \left( \mathbf{x}_s^{(i)}; \boldsymbol{\eta}_s^{(i)} \right) \right\|_2^2 \ . \tag{1.4}$$

After successfully training the surrogate model, it can be used to infer the parameters of

the model from the real-world dataset $\mathcal{D}_r$. However, constructing such surrogates may often require more time than the savings they provide. For instance, if more function evaluations are required to create the dataset $\mathcal{D}_s$ than to simply infer the model parameters $\mathcal{D}_r$ from the forward model, creating the surrogate will provide no cost-saving benefits. Additionally, selecting an appropriate parametric surrogate form is critical to the success of the surrogate.

Machine learning-based surrogate models have enormous expressive power and flexibility, but generalizability is often poor outside the range of training data due to observational biases. An effective surrogate model may only require accurate predictions on available training data. However, it is often more desirable to accurately predict the system's behavior in untested conditions not included in the data which it has learned from. This is referred to as *generalizability*, and without special care towards incorporating information on the known physical principles underpinning a particular process, generalizability often suffers. Additionally, general modeling based on ML techniques often provides little, if any, physical interpretability into the surrogate. Consideration for physical principles should therefore be incorporated into the design of surrogate models to facilitate interpretability and generalizability of the model. Without such consideration, it is often not possible to explain the inner workings of the model, preventing insights into potential solutions when issues inevitably arise during the modeling process.

## 1.4 Dimensionality Reduction and Reduced Order Modeling

### 1.4.1 Dimensionality Reduction

Dimensionality reduction is a critical process in data analysis and machine learning, particularly useful for simplifying high-dimensional datasets. By reducing the number of variables under consideration, it becomes easier to visualize, process, and interpret data.

Linear dimensionality reduction techniques are grounded in the assumption that the data lies approximately on a linear manifold within the higher-dimensional space. Proper Orthogonal Decomposition (POD) is a quintessential example, where the data is projected onto a lower-dimensional linear space which minimizes the projection error in a Frobenius norm sense. Mathematically, a matrix $\mathbf{Y} \in \mathbb{R}^{m \times N}$ is constructed such that each of the $N$ columns represents a data sample with $m$ elements, and POD seeks to find a transformation $\mathbf{\Psi} \in \mathbb{R}^{m \times c}$ of rank $c \leq N$ by solving optimization problem

$$\min_{\mathbf{\Psi} \in \mathbb{R}^{m \times c}} \|\mathbf{Y} - \mathbf{\Psi}\mathbf{\Psi}^T\mathbf{Y}\|_F, \text{ subject to } \mathbf{\Psi}^T\mathbf{\Psi} = \mathbf{I}_{c \times c} \, . \tag{1.5}$$

According to the Schmidt-Mirsky-Eckart-Young theorem [19], the solution is given by the first $c$ left singular vectors of the data matrix $\mathbf{Y}$, where the singular vectors $\mathbf{L}$ are computed by singular value decomposition (SVD) such that $\mathbf{Y} = \mathbf{L}\mathbf{\Sigma}\mathbf{R}^T$.

Data samples can then be projected onto the basis to compute a lower dimensional representation of the data sample. Assuming that $c < m$ (to achieve dimensionality reduction), then the solution $\hat{\mathbf{a}} \in \mathbb{R}^c$ to the optimization problem

$$\min_{\mathbf{a}} \|\mathbf{y} - \mathbf{\Psi}\mathbf{a}\| \tag{1.6}$$

gives the compressed representation of a data sample $\mathbf{y}$. This solution is found by leveraging the left inverse of the basis: $\hat{\mathbf{a}} = (\mathbf{\Psi}^T\mathbf{\Psi})^{-1}\mathbf{\Psi}^T\mathbf{y}$ .

However, noting that $\mathbf{\Psi}^T\mathbf{\Psi} = \mathbf{I}_{c\times c}$, the basis coefficient vector (compressed representation) is given by

$$\hat{\mathbf{a}} = \mathbf{\Psi}^T\mathbf{y} \ .$$

Nonlinear dimensionality reduction techniques, on the other hand, can be used when the data resides on a highly nonlinear yet still lower-dimensional manifold. Techniques such as t-Distributed Stochastic Neighbor Embedding (t-SNE) [20] and autoencoders [21] are popular in such scenarios. The t-SNE method, for instance, converts similarities between data points to joint probabilities and tries to minimize the Kullback–Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data. Autoencoders utilize a pair of functions, typically implemented as neural networks, for encoding and decoding data. The encoder function $\mathbf{E} : \mathbb{R}^m \to \mathbb{R}^n$ compresses data samples into a lower-dimensional space, while the decoder function $\mathbf{D} : \mathbb{R}^n \to \mathbb{R}^m$ aims to reconstruct the data back to its original space from the lower dimensional representation. As the encoding and decoding functions can have general form, this is a flexible nonlinear approach to dimensionality reduction. The neural networks are trained to reconstruct the data samples by minimizing an error metric $d(\mathbf{x}, \mathbf{D}(\mathbf{E}(\mathbf{x})))$ during training. A compressed representation $\mathbf{z} \in \mathbb{R}^n$ of the data can then be computed after appropriate encoding and decoding functions are found by $\mathbf{z} = \mathbf{E}(\mathbf{x})$.

### 1.4.2 Reduced Order Modeling

In the context of physics-based problems, reduced order modeling (ROM) techniques are often employed to reduce the computational complexity of numerical simulations by modeling the system in a lower dimensional space. These models operate on the assumption that the dynamics of high-dimensional systems are often constrained to a lower-dimensional

manifold. This reduction in complexity is particularly beneficial when solving the full high dimensional form of a physical model is computationally expensive, such as in large scale computational fluid dynamic (CFD) simulations. ROM techniques typically leverage the previously mentioned methods of dimensionality reduction, requiring data to compute and perform compression operations, and solve a system of equations in the lower dimensional space rather than in the high dimensional space of the data. While surrogate modeling focuses on approximating the full order system with a more efficient form, reduced order modeling addresses a slightly different, yet related challenge. Surrogate modeling and reduced order modeling each exhibit a similar goal in common: the reduction of computational complexity. While the former primarily deals with creating an external model that predicts a full system's outputs, however, reduced order modeling aims towards simplifying the system itself. This distinction is important as it implies a different yet related approach to handling the trade-off between model accuracy and computational efficiency.

We consider the case of simulating dynamical systems of the form

$$\frac{d\mathbf{u}(t)}{dt} = \mathcal{F}(\mathbf{u}(t); \boldsymbol{\eta});\ \mathbf{u}(0) = \mathbf{u}_0\ , \tag{1.7}$$

where $\mathbf{u} \in \mathbb{R}^m$ is the solution, $\boldsymbol{\eta} \in \mathbb{R}^p$ are parameters defining the dynamical system, $\mathbf{u}_0$ is the initial condition, and $\mathcal{F}$ defines the right hand side operator of the dynamical system. The aim in reduced order modeling is to develop an approximation to the full order model of Eq. 1.7 with a system of lower dimension

$$\frac{d\mathbf{u}_r(t)}{dt} = \mathbf{f}_r(\mathbf{u}_r(t); \boldsymbol{\eta});\ \mathbf{u}_r(0) = \mathbf{u}_{r0}\ , \tag{1.8}$$

where $\mathbf{u}_r \in \mathbb{R}^c$. To develop data-driven ROMs, we define the dataset by a matrix $\mathbf{U} = [\mathbf{u}(t_0), \mathbf{u}(t_1), \ldots, \mathbf{u}(t_N)]$, where $\mathbf{u}(t_i)$ is the full order solution at discrete time index $i$ and defines a column of the overall data matrix, and there are $d$ discrete time steps in the data.

#### 1.4.2.1 Linear Projection-Based Approaches

Projection-based ROMs develop the reduced order model by a projection of the full order solution to a lower dimensional linear subspace. It is assumed in such methods that the projection $\tilde{\mathbf{u}} = \mathbf{V}\mathbf{u}_r$ is a good approximation of the full order $\mathbf{u}$, and that the dynamics of the model are approximately constrained to a lower dimensional linear manifold. The orthonormal matrix $\mathbf{V} \in \mathbb{R}^{m \times c}$ is known in this setting as the *trial basis*. It is therefore

assumed that the following equation holds (although this is not a strictly true statement):

$$\frac{d\mathbf{V}\mathbf{u}_r(t)}{dt} = \mathcal{F}(\mathbf{V}\mathbf{u}_r(t);\boldsymbol{\eta});\ \mathbf{V}\mathbf{u}_r(0) = \mathbf{u}_0\ . \tag{1.9}$$

Next, an orthonormal *test-basis* $\mathbf{W} \in \mathbb{R}^{c\times m}$ is defined, and left multiplied with Eq. 1.9. The resulting ROM equations are then given by

$$\frac{\mathbf{u}_r(t)}{dt} = [\mathbf{W}^T\mathbf{V}]^{-1}\mathbf{W}^T\mathcal{F}(\mathbf{V}\mathbf{u}_r(t);\boldsymbol{\eta});\ \mathbf{u}_r(0) = [\mathbf{W}^T\mathbf{V}]^{-1}\mathbf{W}^T\mathbf{u}_0\ . \tag{1.10}$$

In the special case of an equivalent test and trial basis $\mathbf{W} = \mathbf{V}$, known as Galerkin projection [22], the ROM equations become

$$\frac{\mathbf{u}_r(t)}{dt} = \mathbf{V}^T\mathcal{F}(\mathbf{V}\mathbf{u}_r(t);\boldsymbol{\eta});\ \mathbf{u}_r(0) = \mathbf{V}^T\mathbf{u}_0\ . \tag{1.11}$$

The basis $\mathbf{V}$ is often taken as the POD-obtained basis (defined in Sec. 1.4.1) of the data matrix $\mathbf{U}$. Although this effectively reduces the dimensionality of the model, it is noted that evaluating the full order operator is still required. In the particular case of the Galerkin ROM, $\mathbf{f}_r(\mathbf{u}_r(t);\boldsymbol{\eta}) = \mathbf{V}^T\mathcal{F}(\mathbf{V}\mathbf{u}_r(t);\boldsymbol{\eta})$, requiring the evaluation of the full operator $\mathcal{F}$. One method of reducing this computational complexity is to evaluate the full order operator at only specified locations given by a measurement matrix $\mathbf{P} \in \mathbb{R}^{s\times m}$, and reconstruct the entire full order function. If data is obtained by computing the full order operator $\mathcal{F}$ to form a matrix $\mathbf{F} = [\mathcal{F}(\mathbf{u}(t_0);\boldsymbol{\eta}), \mathcal{F}(\mathbf{u}(t_1);\boldsymbol{\eta}), \ldots, \mathcal{F}(\mathbf{u}(t_d);\boldsymbol{\eta})]$, the data can be used to computed a POD basis $\boldsymbol{\Psi} \in \mathbb{R}^{m\times s}$ such that samples are represented via basis coefficients $\mathbf{a} \in \mathbb{R}^s$ as $\mathcal{F} = \boldsymbol{\Psi}\mathbf{a}$. Taking specific measurements defined by the measurement matrix results in $\mathbf{P}\mathcal{F} = \mathbf{P}\boldsymbol{\Psi}\mathbf{a}$, giving approximate basis coefficients of $\hat{\mathbf{a}} = [\mathbf{P}\boldsymbol{\Psi}]^\dagger\mathbf{f}_s$ where $\mathbf{f}_s$ is the full order operator evaluated only at the locations specified by $\mathbf{P}$. The full order operator can thus be approximately reconstructed by $\hat{\mathbf{f}} = \boldsymbol{\Psi}[\mathbf{P}\boldsymbol{\Psi}]^\dagger\mathbf{f}_s$. Finally, this leads to a reduced order model (in the general case) in which evaluating the full order operator is restricted to evaluation at only $s < m$ locations by

$$\frac{d\mathbf{u}_r(t)}{dt} = [\mathbf{W}^T\mathbf{V}]^{-1}\mathbf{W}^T\boldsymbol{\Psi}[\mathbf{P}\boldsymbol{\Psi}]^\dagger\mathbf{f}_s(t) \tag{1.12}$$

An advantage of these methods is that they are not only data-driven, but also take into account some knowledge of the full-order equations, potentially providing additional information which could aid in prediction and maintaining some degree of interpretability. However, as the methods are data driven, prediction performance is heavily dependent on the available data, limiting generalizability to outside of the training regime - one of the

primary difficulties to overcome in data-driven methods. An additional shortcoming of linear projection-based ROMs is just that - they are linear. Projecting the data in such a manner will therefore inevitably lead to large truncation errors in nonlinear systems where reduced order predictions are compared to simulating the full order system. Adaptive ROMs in which the basis is updated in time can partially alleviate these shortcomings at the cost of increased computational complexity [23], but are outside the scope of this work.

#### 1.4.2.2 Autoencoder-Based Reduced Order Models

Reduced order models that utilize autoencoders [24] and machine learning represent a notable departure from traditional projection-based ROMs. As outlined in Section Sec. 1.4.1, autoencoders employ an encoder-decoder architecture to learn a nonlinear manifold, known as the latent space, where data is compactly represented. Typically, these functions are built using neural networks and are trained using established machine learning techniques. Of primary interest in autoencoder-based ROMs is the trajectory that the full order dynamical system traces within this latent space, which is essentially what is being modeled in a ROM. The non-linearity of autoencoders combined with the flexibility which neural networks and deep learning can provide facilitates the capturing of more complex patters and relationships compared to linear projection methods. However, with the general nonlinear nature of the encoding and decoding functions, integrating specific insights from the original, high-dimensional form of the full order model into the latent space representation can be challenging. As a result, such models often rely primarily on data-driven approaches to capture the dynamics within the latent space, leading to the loss of physical interpretability.

Assuming that the encoding and decoding functions have already been trained, data samples $\mathbf{u}$ can be encoded to a latent vector $\mathbf{u}_r$ (reduced order representation) by leveraging the encoder using the relationship $\mathbf{z} \equiv \mathbf{u}_r = \mathbf{E}(\mathbf{u})$. Assuming that the data $\mathbf{U}$ used to train the encoder and decoder is encoded to the latent space, a matrix $\mathbf{Z} = [\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_N]$ corresponding to the latent representations is constructed where $\mathbf{z}_i = \mathbf{E}(\mathbf{u}(t_i))$.

After obtaining the reduced order description of all data samples, the goal is to create an approximation to the true latent dynamics model

$$\frac{d\mathbf{z}}{dt} = \mathbf{g}(\mathbf{z}(t)) \tag{1.13}$$

by learning an approximate parametric functional form $\mathbf{g}_\phi(\mathbf{z}(t)) \approx \mathbf{g}(\mathbf{z}(t))$ or otherwise modeling the dynamics in discrete time with a parametric functional form by [25]

$$\mathbf{z}_{i+1} = \hat{\mathbf{g}}_\phi(\mathbf{z}_i, \mathbf{z}_{i-1}, \ldots, \mathbf{z}_0) \; . \tag{1.14}$$

The modeling of such functions is often performed through the use of neural networks [25, 1] to create parametric model forms in which parameters $\phi$ are learned using data $\mathbf{Z}$. Learning an approximation to the continuous right hand side of Eq. 1.13 can be achieved with the use of Neural Ordinary Differential Equations (NeuralODE) [26]. The discrete time approximation given in Eq. 1.14 can be trained through simpler means of minimizing some error measure such as the square of the $L_2$-norm between data and prediction by

$$\min_{\phi} \sum_{i=1}^{N-1} \|\mathbf{z}_{i+1} - \hat{\mathbf{g}}_{\phi}(\mathbf{z}_i, \mathbf{z}_{i-1}, \ldots, \mathbf{z}_0)\|_2^2 \,. \tag{1.15}$$

These methods, due to their ability to incorporate flexible nonlinear forms in both the encoding-decoding functions and the dynamics approximation functions, can be more expressive and yield more accurate ROMs compared to linear projection-based approaches. By embracing this flexibility, they are adept at capturing complex patterns and relationships within the data. However, a significant limitation of such purely data-driven approaches is their often limited generalizability and interpretability, particularly during prediction phases. Models frequently struggle to make accurate predictions outside the range of the training data. Additionally, the absence of physical interpretability in these models hampers efforts to gain deeper insights into the underlying reasons behind inaccurate predictions.

Reduced order modeling and surrogate modeling offer a lens through which we can appreciate the intricacies and nuances of capturing physical phenomena. However, to fully grasp the capabilities and limitations of machine learning in this context, it's crucial to delve into the underlying principles that these algorithms leverage. Central to many machine learning algorithms are the concepts and methodologies derived from probability theory, information theory, and various inference techniques. These concepts not only provide some of the theoretical backbone for machine learning methods but also illuminate the pathways through which these algorithms process and learn from data. Thus, before exploring the core mechanisms of machine learning algorithms, we will comprehensively examine these probabilistic frameworks.

## 1.5 Uncertainty Quantification, Inference, and Probabilistic Modeling

### 1.5.1 Uncertainty Quantification and Inference

Uncertainty Quantification (UQ) is an essential aspect of physics modeling, particularly for managing and understanding the inherent uncertainties present in physical systems and measurement processes. Until now in this work, modeling in physics has been approached from a deterministic standpoint, where systems are presumed to behave in predictable ways under given conditions. However, real-world systems are often subject to various uncertainties that can significantly affect predictions. Recognizing and quantifying these uncertainties can become critical, especially in scenarios requiring substantial decision-making.

In contexts where decisions have considerable safety implications or involve substantial financial or time investments, accurately assessing the uncertainty in model predictions is vital. This assessment is especially crucial in the operation of autonomous systems, where decisions must be made dynamically and with a high degree of reliability. The uncertainties in model predictions typically fall into two broad categories: aleatoric and epistemic.

1. **Aleatoric uncertainty:** This type of uncertainty arises from the inherent randomness or variability in the system. It is an intrinsic part of the system and remains even with complete knowledge of the system. For instance, the unpredictability in quantum mechanical systems, the variability in material properties under different environmental conditions, and other types of noise in observed measurements are all examples of aleatoric uncertainty.

2. **Epistemic uncertainty:** Epistemic uncertainty stems from incomplete knowledge or information about the system. This could be due to limitations in our understanding of the system, errors in data measurement, or inadequacies in the model itself. Unlike aleatoric uncertainty, epistemic uncertainty can be reduced as we gain more information or develop more sophisticated models. An example would be the uncertainty in predicting climate change due to limitations in our current climate models. Epistemic uncertainty includes both model form uncertainty and parameter uncertainty.

Understanding these types of uncertainties allows for more informed and cautious decision-making, particularly in high-stakes scenarios. It enables us to evaluate the reliability of the predictions made by models and to consider the potential risks associated with decisions based on these predictions. Techniques include probabilistic modeling, sensitivity analy-

sis, and error propagation, which help in quantifying the impact of uncertainties on model outputs.

Even within probabilistic modeling, there are two primary perspectives, and each differs philosophically in their respective approach:

1. **Frequentist perspective**

    - Treats probability as a long-run frequency of events, focusing on the relative frequency of events in repeated trials
    - Considers parameters as fixed but unknown quantities
    - Typically involves hypothesis testing and confidence intervals based on sample data without incorporating prior knowledge

2. **Bayesian perspective**

    - Views probability as a measure of belief or certainty in an event, incorporating prior knowledge and evidence
    - Treats parameters as random variables with probability distributions, reflecting uncertainty or prior beliefs about these parameters
    - Uses prior information to represent beliefs and update these beliefs in light of new information

This thesis predominantly adopts a Bayesian perspective in presenting probabilistic modeling and inference, despite some commonalities with the frequentist viewpoint. Central to this approach is the utilization of variational inference techniques, which are deeply rooted in Bayesian principles. These techniques stand out for their ability to systematically incorporate prior knowledge, thereby enriching the model-building and analysis process.

However, it is important to note that while UQ and inference are valuable for enhancing the robustness and reliability of predictions, this thesis primarily focuses on leveraging physics-aware machine learning to achieve these objectives. This approach is particularly chosen for its effectiveness in integrating physical laws and principles, which inherently improve the robustness and interpretability of predictive models while being computationally more efficient than Bayesian methods. Nevertheless, probabilistic concepts and Bayesian inference techniques are still pivotal to some of the work presented here. These aspects are not only integrated into the framework of physics-aware machine learning but are also elaborated in detail, highlighting their significance in enhancing model accuracy and reliability.

#### 1.5.1.1 Bayesian Inference

Bayesian inference provides a framework for updating the probability distribution of a random variable based on observed data, encapsulating uncertainties in a probabilistic manner and facilitating rational decision making. For example, in the context of a physical model, suppose that Eq. 1.2 is assumed as model $M$ describing the dynamics of a physical system. Data $\mathcal{D}$ obtained through observing the real world system can be used to infer a conditional distribution on the model parameters $\boldsymbol{\eta}$ as

$$p(\boldsymbol{\eta}|\mathcal{D}, M) = \frac{p(\mathcal{D}|\boldsymbol{\eta}, M)p(\boldsymbol{\eta}|M)}{p(\mathcal{D}|M)} \ , \tag{1.16}$$

known as Bayes' theorem [27, 28, 29, 30]. Bayes' theorem contains four important distributions:

- **Likelihood** $p(\mathcal{D}|\boldsymbol{\eta}, M)$ - represents the probability of observing the data given the assumed model $M$ and parameters $\boldsymbol{\eta}$
- **Prior** $p(\boldsymbol{\eta}|M)$ - encapsulates the initial beliefs about the parameters before considering the current data
- **Evidence** $p(\mathcal{D}|M) = \int p(\mathcal{D}|\boldsymbol{\eta}, M)p(\boldsymbol{\eta}|M)d\boldsymbol{\eta}$ - the probability of the observed data under all possible model parameters. The evidence can be considered a regularization constant.
- **Posterior** $p(\boldsymbol{\eta}|\mathcal{D}, M)$ - the updated belief about the parameters after taking into account both the prior and the likelihood of the observed data

The likelihood model is typically formed based on theoretical knowledge or assumptions about the underlying data-generating process. It involves specifying a probability distribution that models how the observed data is generated given a model and its parameters. The choice of distribution and its parameters depend on the nature of the data (e.g., continuous, discrete, binary) and the specific context of the application. Selecting the likelihood model is discussed in more detail in Section 1.5.2

The prior model in the Bayesian framework is typically formed based on subjective expert experience, empirical insights from data, selected as non-informative or weak, or selected as a matter of computational convenience. Substantial expertise from past experience may provide prior information alone or in combination with empirical information based on data. Non-informative or weak priors may be selected when little to no prior knowledge is available to prevent bias. Additionally, it is often computationally convenient to select prior

forms which may have easily computable or standard forms such as the Gaussian family of distributions.

The evidence serves as a normalization constant in Bayes' theorem, and computing it directly (marginalization) can be challenging, particularly in high dimensional problems. Many Bayesian methods therefore avoid computing or approximating the evidence altogether. For example, Markov chain Monte Carlo (MCMC) [27] methods aim to sample from the posterior distribution without ever computing the evidence. Additionally, variational inference techniques avoid computing the evidence by approximating the posterior with a parameterized form for which normalization is guaranteed.

While a variety of algorithms are available for conducting Bayesian inference, variational inference stands out for its particular relevance in the field of machine learning. This method is notable for its efficiency in approximating complex probability distributions, making it highly suitable for large-scale and computationally intensive machine learning applications.

### 1.5.2 Probabilistic Modeling

Probabilistic modeling defines the use of mathematical tools for representing uncertain statements and making predictions based on these uncertainties. It is grounded in the principles of probability theory and is instrumental in a wide range of fields, from machine learning and statistics to finance and engineering. The core idea of probabilistic modeling is to describe the uncertainty inherent in complex systems and observations through probability distributions. This approach contrasts with deterministic models, which assume exact outcomes given specific inputs.

The utility of probabilistic modeling lies in its ability to capture the randomness and variability inherent in real-world phenomena. By accounting for uncertainty, these models can provide more robust and realistic representations of complex systems with the downside of higher computational costs. They enable the estimation of not just the most likely outcomes, but also the variability and risk associated with different scenarios. This aspect is particularly crucial in fields where decision-making under uncertainty is common, such as finance [31], weather and climate modeling [31, 32], and energy production [33].

In the context of Bayesian inference, probabilistic modeling plays a pivotal role in the selection of the likelihood model. The likelihood model, a fundamental component of Bayesian analysis, represents the probability of the observed data given certain model parameters. Developing and selecting an appropriate likelihood model is critical as it directly influences the inference process. The chosen model should be reflective of the underlying data generation process and capture the essential features of the data. For instance,

a normal distribution might be used to model continuous data with a symmetric distribution around the mean. Assuming a deterministic physical model $\mathbf{u}(t) = \mathbf{f}(t, \mathbf{u}(0), \boldsymbol{\eta})$ may correspond to a data measurement model with additive Gaussian noise given by $\mathbf{u}(t) = \mathbf{f}(t, \mathbf{u}(0), \boldsymbol{\eta}) + \boldsymbol{\xi} \;\; \boldsymbol{\xi} \sim \mathcal{N}(0, \boldsymbol{\sigma}^2)$, where the variance $\boldsymbol{\sigma}^2$ is estimated from data. Assuming an i.i.d dataset $\mathcal{D} = \{t^{(i)}, \mathbf{u}^{(i)}(0), \mathbf{u}^{(i)}(t^{(i)})\}_{i=1}^{N}$, the likelihood assuming this measurement model is computed by

$$p(\mathcal{D}|\boldsymbol{\eta}) = \prod_{i=1}^{N} \mathcal{N}(\mathbf{u}^{(i)}(t^{(i)}); \mathbf{f}(t^{(i)}, \mathbf{u}^{(i)}(0), \boldsymbol{\eta}), \boldsymbol{\sigma}^2) \; . \tag{1.17}$$

The process of selecting a likelihood model in a Bayesian framework also involves balancing simplicity with accuracy. A model that is too simple might not capture the nuances of the data, while an overly complex model can lead to overfitting and computational challenges. For instance, the likelihood model in Eq. 1.17 may be too simple in some applications. However, a more general measurement model which is a nonlinear function of noise, given by $\mathbf{u}(t) = \mathbf{f}(t, \mathbf{u}(0), \boldsymbol{\eta}, \boldsymbol{\xi}) \;\; \boldsymbol{\xi} \sim \mathcal{N}(0, \mathbf{I})$, may result in a far more complex and computationally expensive likelihood model. Therefore, probabilistic modeling in Bayesian inference is as much an art as it is a science, requiring careful consideration of both the data and the underlying physical or theoretical principles governing the system being modeled.

Probabilistic modeling is a powerful tool for understanding and predicting behavior in complex systems where uncertainty is a key factor. Embracing the inherent uncertainty in data and systems, it offers a nuanced and realistic lens for understanding complex phenomena. However, importantly traditional probabilistic modeling methods can be computationally intensive, particularly when dealing with large datasets. The integration of probabilistic modeling with machine learning techniques becomes invaluable in such cases, as it can significantly reduce computational costs.

For instance, consider the computational demands of calculating the likelihood in Eq. 1.17, which might involve numerous evaluations of a complex forward model. If this model is computationally intensive, the development of a surrogate model could substantially decrease the resources needed to evaluate the likelihood. This approach not only streamlines the process but also retains the model's effectiveness.

Moreover, the principles of probability theory and probabilistic modeling are foundational in many machine learning algorithms. Understanding the role of these probabilistic techniques is crucial, especially in the context of machine learning. By comprehending key concepts in machine learning, we can better identify and leverage the opportunities for integrating probabilistic approaches. This integration does more than just enhance computational efficiency - it also enriches machine learning models with the ability to handle

uncertainty and variability inherent in real-world data and models.

## 1.6 Data Driven Machine Learning

Advances in data-driven machine learning have represented a paradigm shift in computational modeling and analysis. At their core, such approaches leverage large datasets to train algorithms to make predictions or decisions without being explicitly programmed for specific tasks [34]. This methodology contrasts sharply with traditional physics model-driven approaches, where rules and relationships are predefined based on theoretic understanding of the processes under investigation. Data-driven ML has proved immensely useful in various domains, from image and speech recognition to complex decision-making processes in autonomous systems [35]. Its ability to extract patterns and insights from vast and complex datasets has surpassed traditional methods in terms of efficiency, scalability, and often, accuracy.

One of the most significant advancements brought about by data-driven ML is in the realm of predictive analytics. For instance, in healthcare, ML models can predict patient outcomes based on historical data, a task that was previously reliant on less precise statistical methods [36]. In finance, ML algorithms have transformed risk assessment and fraud detection processes [37]. Countless other examples exists in the literature, demonstrating the enormously diverse areas of application for such algorithms when datasets are available. In this thesis, a variety of ML techniques are demonstrated in each of the works presented. Although numerous areas of machine learning exist, this work focuses mainly on the application of the following topics:

1. **Supervised Learning**: This involves training ML models on labeled data, where the desired input and output of the model are known on some dataset. The model learns a function mapping inputs to desired outputs, which is crucial for tasks such as classification and regression [38], both of which are prevalent in this work.

2. **Unsupervised Learning**: In scenarios where the input-output relationships within data are unlabeled, unsupervised learning aims to descern inherent structures or patterns present in the data. This form of learning is useful for identifying underlying features or structures of the data, lending it useful for tasks such as representation learning, transfer learning, and clustering [39], all of which are discussed here.

3. **Neural Networks and Deep Learning**: Neural networks, a fundamental concept in machine learning, are computational models designed to mimic the human brain's

interconnected neuron structure. They process input data through multiple layers of nodes, each layer transforming the input in a way that helps the network learn complex patterns and relationships. This design enables neural networks to perform a wide range of tasks, from recognizing images and speech to making predictions and decisions, by adjusting internal parameters based on data [24].

Despite its advancements, data-driven ML is not without challenges. The quality and quantity of data significantly influences the performance of ML models. Training ML models, particularly without the introduction of inductive biases, often requires enormous amounts of data to be effective. Issues such as data bias can additionally lead to skewed or unfair outcomes [40]. Furthermore, the black box nature of many ML models, particularly deep learning, poses challenges in interpretability and trustworthiness [41]. While this work includes smaller explorations into the former issue, its primary focus is on addressing the latter, particularly the challenge of enhancing the interpretability and reliability of ML models.

At the heart of machine learning lies the process of creating and solving optimization problems. This core principle is integral to the development and functionality of various machine learning models. Essentially, a machine learning algorithm involves constructing an optimization problem where the objective is to find the best possible model parameters that minimize a loss or cost function. This function quantitatively measures how far off a model's predictions are from the data according to some objective. The optimization problem, therefore, revolves around adjusting the model parameters in a way that the loss is minimized, indicating that the model's predictions are as close as possible to achieving the particular object enforced by the selected loss function. Assuming a dataset $\mathcal{D}$, a loss function $\mathcal{L}$, and a prediction model $f_\phi$ parameterized by parameters $\phi$, most machine learning algorithms can be framed as solving the optimization problem

$$\min_{\phi} \mathcal{L}(\mathcal{D}, f_\phi(\mathcal{D})) \ . \tag{1.18}$$

In supervised learning, for instance, this might involve adjusting weights in a neural network or coefficients in a regression model to best fit the observed data. Similarly, in a classification task, the loss may quantify some comparison between predicted class probability distributions and the true class. Loss functions may also be augmented with information from physical principles to softly guide model performance with learning biases towards consistency with such principles. The optimization techniques employed in minimizing loss functions can range from simple gradient descent in linear models to more complex algorithms in deep learning, although this thesis will primarily rely on stochastic gradient descent (SGD) techniques common in many machine learning packages. The efficacy of a machine learning model is

largely determined by how well the optimization problem is formulated and solved, balancing the trade-offs between accuracy, computational efficiency, and the ability to generalize from the training data to unseen data.

Along with selecting an appropriate loss function, model performance is also greatly dependent on the particular parametric form $f_\phi$ of the model. This work will often leverage *neural networks* in constructing the parameterized model, but often in combination with inductive biases, changing the model form to improve adherence to physical principles which the standard form does not take into account.

### 1.6.1 Neural Networks

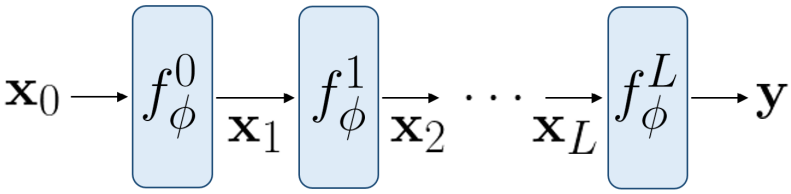


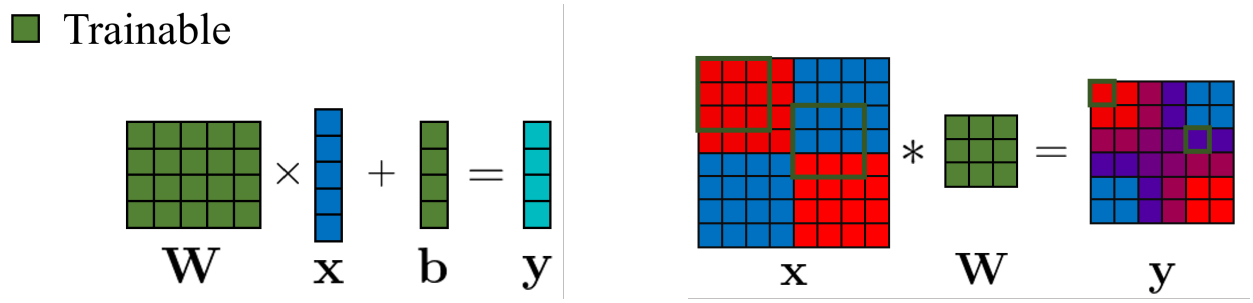

Figure 1.2: Neural networks employ a composition of simple functions to transform an input $\mathbf{x}_0$ to a prediction $\mathbf{y}$.

Neural networks (NNs) are a cornerstone of modern machine learning, inspired by the structure and function of the human brain. They originally consisted of layers of interconnected nodes or 'neurons', known as a *dense layer* or *fully connected layer*, each performing simple computations. The output of these computations is passed through the network to achieve complex tasks such as regression and classification. At its core, an NN transforms an input by passing it through multiple layers of non-linear functions. This process is illustrated in Fig. 1.2. Mathematically, consider an input vector $\mathbf{x}_0$ and an NN consisting of $L$ layers. If each layer $l$ is defined by a parametric function $f_\phi^l$ with output $\mathbf{x}_l$, the output of the network $\mathbf{y}$ is given by the composition of functions

$$\mathbf{y} = f_\phi^L(f_\phi^{L-1}(\dots f_\phi^1(f_\phi^0(\mathbf{x}_0)))) \,. \tag{1.19}$$

The parameterization of each $f_\phi^l$ can take on many forms, and the particular forms chosen are usually application dependent. The parametric form of each $f_\phi^l$ along with the number of functions defines the network's *architecture*. In more advanced architectures, even the form of Eq. 1.19 can be significantly altered to include alternate functional compositions. However, for the purposes of the discussion presented here, we will adhere to this structure. There are

various types of layers which can define the functions $f_\phi^l$ which are heavily prevalent in this work, the first of which has already been mentioned as the fully connected layer. Two of the simplest but primary layer types prevalent in NNs will be briefly discussed, but many other functions exist in the literature.




(a) Dense layers (fully connected layers)

(b) Convolutional layers

Figure 1.3: Two neural network layer functions $f$ common in deep learning.

**Fully connected layers** These layers, also known as dense layers, are one of the simplest types of layer used in the creation of NNs. Each value in the input $\mathbf{x} \in \mathbb{R}^n$ is connected to each value in the output $\mathbf{y} \in \mathbb{R}^m$ through a matrix of trainable weights $\mathbf{W} \in \mathbb{R}^{m \times n}$ with the addition of a trainable bias term $\mathbf{b} \in \mathbb{R}^m$. The mathematical form of this function is given by the linear function

$$f_{dense}(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b} \,, \tag{1.20}$$

and is exemplified in Figure 1.3a. Such layers are often used in processing single dimensional vector inputs and contain trainable parameters $\phi = \{\mathbf{W}, \mathbf{b}\}$. However, they can become computationally intensive in much higher dimensions, particularly in data such as images which can contain 2 or 3 spatial dimensions with many degrees of freedom each.

**Convolutional Layers** Convolutional layers are more conducive to learning and processing local spatial relationships, particularly in image-like inputs. These layers apply a convolution operation to the input $\mathbf{x} \in \mathbb{R}^{n \times d}$, using a set of learnable filters. Each filter $\mathbf{W} \in \mathbb{R}^{k_1 \times k_2}$ is convolved across the width and height of the input volume, computing the dot product between the entries of the filter and the input, producing a two-dimensional activation map for each filter. As a result, the network learns filters that activate when they see specific types of features at given spatial positions in the input. An example of the convolutional operation with a single filter is illustrated in Figure 1.3b. Convolutional layers can also be

extended to a third dimension by considering additional channels, such as in natural image data with red, green, and blue channels.

**Activation Functions** Activation functions are central to the performance of NNs. The nonlinear composition which facilitates NNs to be flexible approximators is provided primarily through the use of such functions. An activation function is a nonlinear function which usually operates element-wise on an input. The simplest of these functions is the Rectified Linear Unit (ReLU) which transforms any negative values to 0. However, many common nonlinear activation functions are used in NN architectures, and a review of most common ones is presented in recent work by Szandala et al [42].

Activation functions are typically applied at the output of a layer in the NN. For example, for a nonlinear element-wise activation function $\sigma$, the overall output of an NN layer leveraging the fully connected layer may be

$$\mathbf{x}_{l+1} = \sigma(f^l_{phi}(\mathbf{x}_l)) \ . \tag{1.21}$$

The commonly used NN layers and activation functions which are prevalent in ML no doubt produce powerful and flexible models with advanced capabilities. However, they do not inherently take any aspect of modeling physics into account in the standard purely data-driven formulation. This can lead to inaccurate, non-generalizable, or non-interpretable predictions from the model. Purely data driven approaches certainly have seen success in certain application, but a more robust approach is to combine NNs and ML principles with that of physical modeling.

## 1.7 Modeling Physics with Machine Learning

Modeling physical phenomena has traditionally been anchored in empirical and theoretical foundations, but the emergence of machine learning has heralded a paradigm shift, reshaping physical modeling with data-driven insights. This section explores two distinct data-driven methodologies, purely ML-driven modeling and physics-aware ML, emphasizing their applications, challenges, and advancements. In aerospace engineering, as highlighted by Brunton et al. [2], the role of data is pivotal. The industry, rich in data due to its complex systems and rigorous testing regimes, faces the challenge of leveraging this wealth of information effectively. However, the availability and quality of data, often constrained by practical and safety considerations, play a crucial role in the applicability and success of ML models. This data-centric approach, while offering transformative potential in optimization and predictive

modeling, also necessitates models that are interpretable, generalizable, and certifiable, especially in safety-critical applications [2]. Two of the primary benefits which physics-aware ML provides over purely data-driven approaches are:

- **Improved model interpretability:** *Interpretability* refers to the ability to understand and articulate the mechanism by which a model arrives at its predictions, in terms that align with physical laws and principles. Improving interpretability renders the model's decision-making process more transparent, allowing researchers to grasp how and why specific predictions are made, and ensuring that these explanations are grounded in established physical theories.

- **Improved model generalizability:** *Generalizability* is the capacity of a model to perform accurately across a wide range of conditions and datasets, beyond the specific scenarios on which it was trained. In physics-based machine learning, a model with a high degree of generalizability implies that the model can apply its learned principles to solve problems or make predictions in new physical contexts, demonstrating the model's robustness and its ability to capture and apply physical laws.

Machine learning can significantly enhance the modeling of physical systems, offering a versatile toolkit that differs from traditional methods. In this work, the focus is on employing these tools to model dynamical systems driven by PDEs as well as steady-state conditions. The utility of machine learning spans a broad spectrum, capable of applications such as predicting stochastic events, refining control systems, simulating quantum phenomena, and elucidating the intricacies of complex networks. While the applications are varied, they converge on a common objective: to utilize machine learning for accurate predictions, pattern discovery, and efficient resolution of complex problems in physics. Notably, modeling the complete dynamics of a system, rather than directly forecasting specific quantities or outcomes, provides a comprehensive approach. This robust methodology allows for the computation of various quantities of interest and offers a deeper analysis by understanding the full state of the system.

In the proceeding discussions, the focus of machine learning in physics will be narrowed to dynamical systems and PDEs. Such systems are governed by a system of equations

$$\frac{\partial \mathbf{u}}{\partial t} = \mathcal{F}(\mathbf{u}(t), t; \boldsymbol{\eta}) \,, \tag{1.22}$$

where $\mathcal{F}$ is a nonlinear differential operator defining the dynamics of the system, $\mathbf{u} \in \mathbb{R}^n$ is the physical state of interest, $t$ indicates time, and $\boldsymbol{\eta} \in \mathbb{R}^p$ are parameters which govern the system. The aim of modeling physics with machine learning presented here is often

to approximate the operator $\mathcal{F}$ through the use of surrogate modeling or reduced-order modeling to solve a variety of problems.

The solution to Eq. 1.22 can also be expressed as

$$\mathbf{u}(t) = \mathbf{u}(0) + \int_0^t \mathcal{F}(\mathbf{u}(\tau), \tau; \boldsymbol{\eta}) d\tau \ . \tag{1.23}$$

Therefore, sometimes rather than modeling the right hand side of Eq. 1.22 directly, a model may be sought to approximate the right hand side of Eq. 1.23. For example, given the initial condition, system parameters, and time, a model may be sought to compute an output $\mathbf{u}(t) = \mathbf{f}(t, \mathbf{u}(0), \boldsymbol{\eta})$, where the function $\mathbf{f}$ attempts to approximate the right hand side of Eq. 1.23.

### 1.7.1 Purely ML-Driven Modeling

Purely ML-Driven Modeling in physics embodies an approach where data is the primary driver for machine learning models, without direct integration of physical laws. These models have shown remarkable success in discerning complex patterns and offering predictions, especially in scenarios where there is an abundance of data. However, the reliance on data brings forth certain limitations, particularly observational biases that stem from the nature and scope of the collected data. This issue becomes increasingly significant when distinguishing between experimental datasets, which are often limited in size but rich in real-world applicability, and simulated datasets, which are larger and more controlled but may lack certain real-world complexities.

Purely data-driven machine learning approaches strive to model the operator $\mathcal{F}$ by relying solely on data. Although this strategy has achieved notable success in certain fields, its effectiveness can be quite restricted in other areas due to limited applicability and narrow scope. This section will discuss some prevalent methods of purely data-driven modeling in the context of physical sciences, highlighting their strengths and limitations.

#### 1.7.1.1 Deep Operator Networks

The development of Deep Operator Networks (DeepONets) by Lu et al. illustrates the efficacy of ML in approximating complex operators [43]. Utilizing the universal approximation theorem, DeepONets have showcased their proficiency in addressing a diverse spectrum of problems, encompassing areas from fluid dynamics to material sciences. A more detailed description of the DeepONet framework is provided in Section 2.5. However, their standard formulation does not inherently integrate physical principles, leading to potential challenges

in ensuring the physical plausibility of their predictions. They are employed to provide accurate and efficient approximations of operators; however, these are only approximations with no enforcement of prediction consistency with the operator itself. As discussed by Lu et al. [43], even with well-trained DeepONets, prediction for some inputs can 'explode', resulting in physically implausible results.

#### 1.7.1.2 General Probablistic Modeling Framework

The work of Yang and Perdikaris [44] represents a significant stride in employing data-driven approaches for handling high-dimensional and stochastic systems using a formulation combining variational inference and discriminator training, a bit like combining VAEs and GANs. Their aim is to model probabilistic relationships $p(\mathbf{y}|\mathbf{x})$ in which a measurement model is assumed to contain general non-additive noise $\mathbf{y} = f(\mathbf{x}, \mathbf{z})$, where $\mathbf{x}$ are model inputs, $\mathbf{y}$ are model outputs, and $\mathbf{z}$ are latent variables. This model suggests that any output is a function of the input as well as a randomly distributed latent vector. Their approach is to create a parametric model $f_\theta(\mathbf{x}, \mathbf{z})$ which is trained by leveraging a variational approximation $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})$ to approximate the latent posterior $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$. Essentially, the variational posterior along with the parametric forward model create a generative model which can be trained such that $f_\theta(\mathbf{x}, \mathbf{z})$, $\mathbf{z} \sim p(\mathbf{z})$ generates samples from $p(\mathbf{y}|\mathbf{z})$ with no variational approximations. In other words, the model is capable of predicting general probabilistic relationships with non-parametric density estimation. The models are trained by minimizing the reverse KL between the generative distribution $p_\theta(\mathbf{x}, \mathbf{y})$ and the observed data distribution $q(\mathbf{x}, \mathbf{y})$ with entropy and prediction accuracy regularization. The loss function that is used to train the variational latent posterior and the prediction model is given by Eq. 1.24.

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{x},\mathbf{y})p(\mathbf{z})}[D_{KL}(p_\theta(\mathbf{x}, \mathbf{y})\|q(\mathbf{x}, \mathbf{y})) + (1 - \lambda) \log q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y}) + \beta\|f_\theta(\mathbf{x}, \mathbf{z}) - \mathbf{y}\|^2] \quad (1.24)$$

Additionally, the KL term is approximated by leveraging a discriminative classification model $T_\psi(\mathbf{x}, \mathbf{y})$ trained alongside the prediction model to predict whether input-output pairs correspond to the data distribution or from the prediction model. This discriminative model is what allows for no variational approximation in the measurement model and allows for an unconstrained nonlinear prediction model $f_\theta$. Typically the prediction model is parameterized by a variational approximation to allow a closed form solution to computing the KL divergence term. The second term on the right hand side of Eq. 1.24 is an entropy regularization term with parameter $\lambda$ controlling how much the variational latent posterior is encouraged to spread out. The final term on the right hand side encourages the prediction model to minimize the $L^2$ norm between noisy predictions and the data.

#### 1.7.1.3 Autoencoder-based ROMs

Although such models illustrate success in modeling complex dynamical systems with accurate uncertainty quantification, the prediction model $f_\theta$ is purely data driven. That is, no inductive or learning biases are incorporated into the creation of such models, indicating that model predictions (along with uncertainties) may ultimately prove non physical, uninterpretable, and not generalizable. However, such models may benefit from augmentation with physics-aware methods such as PINNs or the introduction of inductive biases to improve physical consistency.

Significant improvements in computational efficiency for dynamics modeling are brought about by leveraging ROMs. Xu and Duraisamy [1] leverage ML techniques to develop a nonlinear encoding-decoding system and learn reduced-order dynamics in the resulting latent space, employing autoencoders and temporal convolutional architectures to efficiently process and predict dynamics.

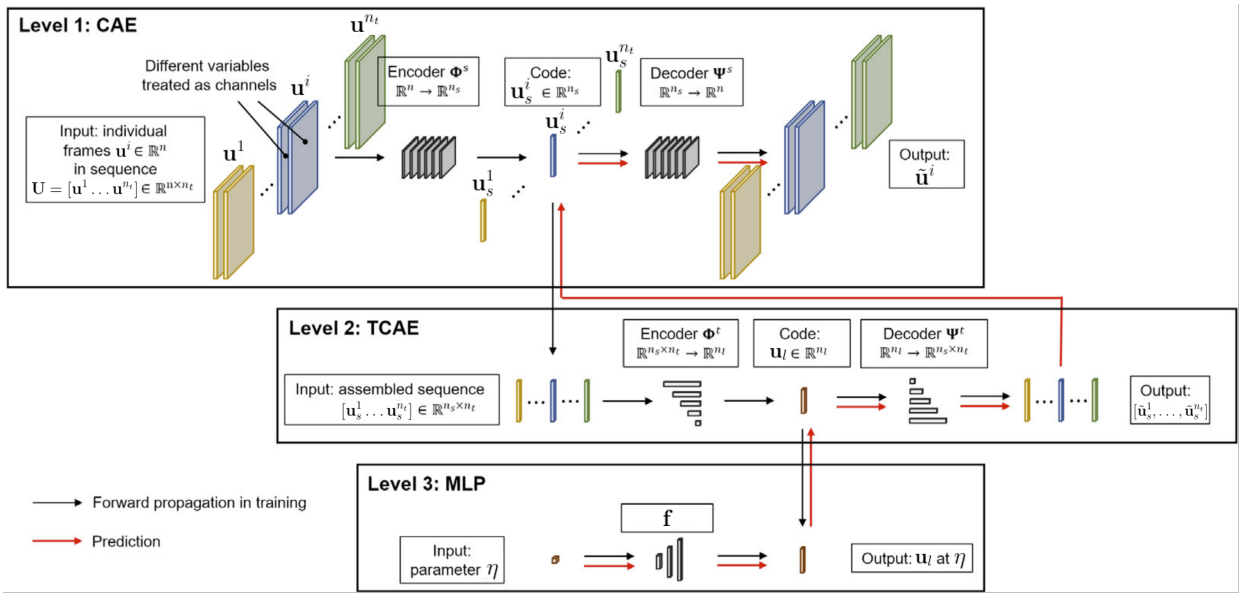



Figure 1.4: Schematic of data-driven reduced order modeling for new parameter prediction. Figure adapted from [1] with permission.

Data in the form of state observations is obtained from a dynamical system with parameters $\boldsymbol{\eta}$ (Eq. 1.22) at discrete times to form a dataset $\mathbf{U}(\boldsymbol{\eta}) = \{\mathbf{u}(t_1;\boldsymbol{\eta}), \mathbf{u}(t_2;\boldsymbol{\eta}), \dots, \mathbf{u}(t_{n_t};\boldsymbol{\eta})\} = \{\mathbf{u}^1, \mathbf{u}^2, \dots, \mathbf{u}^{n_t}\}$. The snapshots in this dataset are used to train a convolutional autoencoder (CVAE), reducing the dimensionality of the data by encoding it through the learned encoding function $\mathbf{\Phi}^s : \mathbb{R}^n \to \mathbb{R}^{n_s}$ to compute the lower dimensional representations as $\mathbf{u}_s^i = \mathbf{\Phi}^s(\mathbf{u}^i)$. The learned decoder $\mathbf{\Psi}^s : \mathbb{R}^{n_s} \to \mathbb{R}^n$ is used to reconstruct the original data snapshots as $\tilde{\mathbf{u}}^i = \mathbf{\Psi}^s(\mathbf{u}_s^i)$. The work addresses two related yet

distinct tasks by leveraging the encoded representations $\mathbf{u}_s$: the first is to predict system dynamics on unseen system parameters $\boldsymbol{\eta}$, and the second is to perform future state prediction in time for a single system.

In the first case, a temporal convolutional autoencoder (TCAE) is used to further compress many snapshots of the lower dimensional representation in the temporal dimension. The encoding function $\boldsymbol{\Phi}^t : \mathbb{R}^{n_s \times n_t} \rightarrow \mathbb{R}^{n_l}$ encodes the entire sequence in time to a single vector as $\mathbf{u}_l = \boldsymbol{\Phi}^t(\mathbf{u}_s^1, \mathbf{u}_s^2, \ldots, \mathbf{u}_s^{n_t})$. This single latent code can then be decoded by the temporal decoder $\boldsymbol{\Psi}^t : \mathbb{R}^{n_l} \rightarrow \mathbb{R}^{n_s \times n_t}$ to obtain a reconstruction of the entire time series of compressed representations by $\tilde{\mathbf{u}}_s^1, \tilde{\mathbf{u}}_s^2, \ldots, \tilde{\mathbf{u}}_s^{n_t} = \boldsymbol{\Psi}^t(\mathbf{u}_l)$. The vector $\mathbf{u}_l$ thus represents a compressed description of the dynamical system corresponding to a particular value of $\boldsymbol{\eta}$. Finally, a relationship between the compressed representation and the system parameters is learned by a neural network as $\mathbf{u}_l = \mathbf{f}(\boldsymbol{\eta})$. After the spatial and temporal autoencoders are learned along with the function $\mathbf{f}$, system dynamics can be predicted for new values of $\boldsymbol{\eta}$. This process is depicted in Figure 1.4, where the figure is adapted with permission from Xu et al [1].

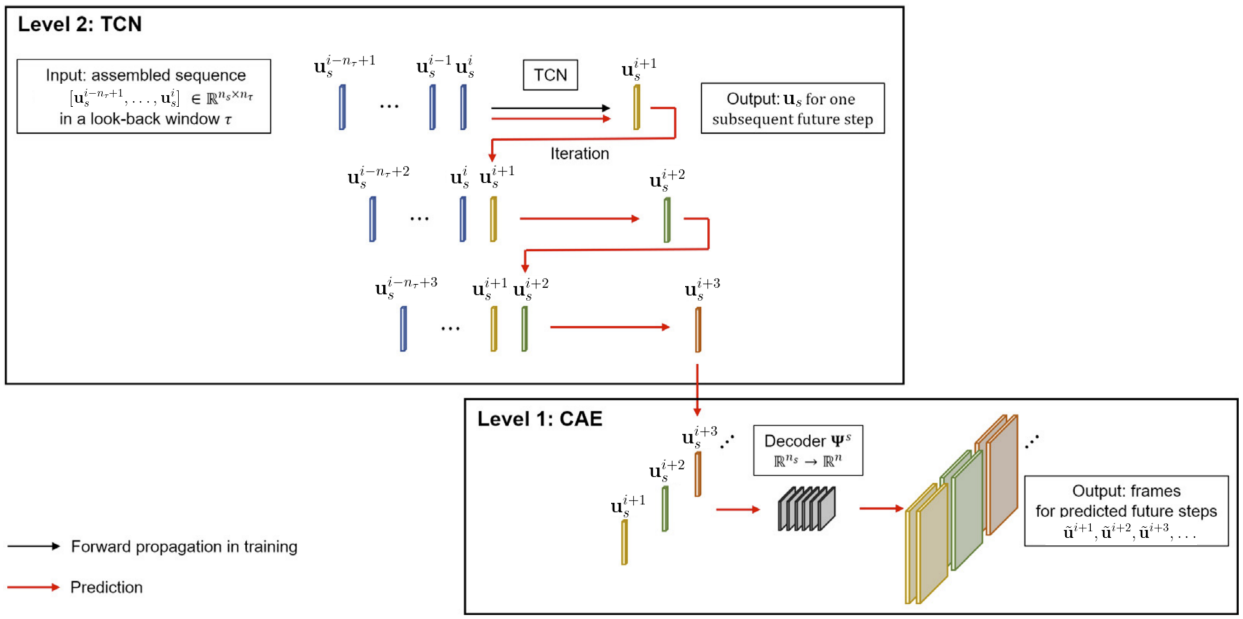



Figure 1.5: Schematic of data-driven reduced order modeling for future state prediction. Figure adapted from [1] with permission.

In the second case, future state prediction leverages a sequence of $n_\tau$ snapshots of the spatially compressed representations to predict the next state in the sequence. A temporal convolutional network (TCN) maps a sequence of states $[\mathbf{u}_s^{i-n_\tau+1} \ldots \mathbf{u}_s^i]$ to the next state in the sequence $\mathbf{u}_s^{i+1}$. These predicted states are then decoded using the spatial decoder

$\mathbf{\Psi}^s$ to obtain reconstructions of the predicted dynamics. The entire process is illustrated in Figure 1.5.

This methodology demonstrates impressive capabilities for purely ML-driven modeling and is a significant contribution to the field; however, from a physics-aware ML perspective, the approach has limitations. While efficient in data-driven contexts, it lacks an explicit incorporation of underlying physical laws and information, which can be crucial for ensuring physical plausibility, interpretability of the predictions, and generalizability. In particular, the spatial decoder may predict non physical reconstructions which do not obey the physical principles being modeled. This omission might limit the model's applicability in scenarios where adherence to physical principles is paramount.

#### 1.7.1.4 Other Limitations

The reliance on large datasets in purely ML-driven models also brings to the foreground the challenge of data quality and representativeness. In many real-world physics applications, acquiring sufficiently large and diverse datasets that accurately capture the underlying phenomena can be difficult. This limitation often leads to models that are highly specialized to the datasets they are trained on, lacking generalizability to new or slightly different conditions. The divide between experimental and simulated data further complicates this issue. Experimental data, while highly realistic, is often limited in scope and may contain noise and measurement errors. Simulated data, although more abundant and controllable, might not fully capture the intricacies of real-world conditions, leading to a gap between model predictions and actual phenomena.

Moreover, the 'black box' nature of many purely ML-driven models poses significant challenges in interpretability. In physics-based applications, and particularly aerospace engineering, where understanding causal relationships, underlying principles, and model predictions is crucial, the inability of these models to provide insight into the 'why' behind their predictions limits their applicability in certain contexts. This lack of interpretability can be particularly problematic in areas requiring rigorous validation and explanation, such as in safety-critical applications [2].

While purely ML-driven models have opened new avenues in physics modeling, their application requires careful consideration of their limitations, particularly in terms of data dependency, lack of physical law integration, and interpretability. Future advancements in this field may focus on developing techniques to mitigate these challenges through approaches which combine data-driven methodologies with physics-informed principles, as many works have aimed towards already.

### 1.7.2 Physics-Aware Machine Learning

Physics-aware machine learning is a transformative approach in computational physics that bridges the gap between traditional data-driven models and the foundational principles of physics. This integration is realized through two complementary methodologies which we call physics-informed machine learning and physics-augmented machine learning. Physics-informed ML primarily focuses on incorporating physical laws into the learning process, ensuring that models not only learn from data but also conform to established physical principles. Physics-augmented ML, on the other hand, extends this paradigm by enhancing the generative properties of the models, incorporating components of traditional physical modeling into the modeling process thus, particularly via inductive biases, allowing them to predict and simulate physical phenomena more accurately.

The significance of physics-aware ML lies in its ability to tackle complex problems that traditional ML struggles with. This approach is crucial in scenarios where data is scarce, noisy, or expensive to obtain. Additionally, traditional methods often lack inductive biases to incorporate knowledge on known physical principles, which limits their generalizability and applicability. Physics-aware ML models can generate reliable and robust predictions by adhering to physical laws, offering insights into areas where empirical data alone may lead to incomplete or erroneous conclusions. The work of Karniadakis et al. [12] is a testament to this, highlighting the potential of physics-aware ML in harmonizing the empirical richness of machine learning with the theoretical rigor of physics. Furthermore, Sinz et al. [45] argue the importance of inductive biases in enhancing the generalization and reliability of these models, pointing out that these biases can lead to more plausible model forms that better mimic the adaptability and robustness seen in natural systems.

#### 1.7.2.1 Learning Biases

In the realm of physics-aware ML, two types of biases are particularly prevalent: learning and inductive biases. Learning biases involve softly constraining models through penalizing loss functions to align predictions with physical principles. Physics-Informed Neural Networks (PINNs) [46] are a prime example of this approach. PINNs leverage prior physical information in the form of PDEs to approximate solutions to the PDE using data along with a set of collocation points at which to evaluate the PDE on.

Considering the dynamical system defined in Eq. 1.22, PINNs aim to model the solution $\mathbf{u}(t)$ at some time $t$. It is also assumed that $\mathbf{u}(t)$ is a function of spatial coordinates as $\mathbf{u}(\mathbf{x}, t)$. Modeling this solution is accomplished by leveraging a dataset consisting of the set $\{\mathbf{x}_u^{(i)}, t_u^{(i)}, \mathbf{u}^{(i)}\}_{i=1}^{N_u}$ along with a set of collocation points $\{\mathbf{x}_{\mathcal{F}}^{(i)}, t_{\mathcal{F}}^{(i)}\}_{i=1}^{N_{\mathcal{F}}}$ at which to compute to

operator $\mathcal{F}$. The PINN is constructed as a parameterized model $\mathbf{u}_\phi(\mathbf{x}, t)$ which is trained by minimizing the loss given in Eq. 1.25.

$$\mathcal{L} = \frac{1}{N_u}\sum_{i=1}^{N_u} |\mathbf{u}_\phi(\mathbf{x}_u^{(i)}, t_u^{(i)}) - \mathbf{u}^{(i)}|^2 + \frac{1}{N_\mathcal{F}}\sum_{i=1}^{N_\mathcal{F}} \left| \frac{\partial \mathbf{u}_\phi(\mathbf{x}, t)}{\partial t} - \mathcal{F}(\mathbf{u}_\phi(\mathbf{x}_\mathcal{F}^{(i)}, t_\mathcal{F}^{(i)}), t_\mathcal{F}^{(i)}; \boldsymbol{\eta}) \right|^2 \tag{1.25}$$

The first term on the right hand side encourages the PINN to match the dataset. However, training a model with only this loss will limit the applicability and generalizability of the model, particularly with limited data samples. The second term on the right hand side defines the introduction of a learning bias into the model. It encourages the model to adhere to the dynamical system at spatial and temporal locations outside of the training data, leading to improved generalization of the model predictions. It is noted that this particular formulation assumes constant dynamical system parameters $\boldsymbol{\eta}$, but the idea can easily be extended to include such parameters in model predictions.

By embedding physical equations directly into the learning process, PINNs offer a powerful tool for solving complex forward and inverse problems in physics. This approach is particularly effective in environments with limited or noisy data, as it leverages the underlying physical laws to guide the learning process and ensure the reliability of predictions. However, PINNs typically predict the solution only at a single spatial and temporal location per function evaluation. Many other types of models often predict entire spatio-temporal fields with a single forward pass, rending them far cheaper than PINNs during online prediction. Additionally, PINNs require many evaluations of the potentially expensive full order operator $\mathcal{F}$ during training, albeit at a subset of collocation points. This renders the training often significantly slower than approaches leveraging data alone. Although the offline and online modeling costs are greater than purely data-driven modeling approaches, generalization and accuracy with respect to the physical equations are greatly improved. An extension of this idea to generative models is also leveraged to encourage generated samples from VAEs to adhere to the underlying PDE [47].

#### 1.7.2.2 Inductive Biases

Rather than softly constraining models representing physical systems with learning biases, inductive biases are introduced directly into the architecture of the machine learning models. Specially designing the model forms allows them to naturally adhere to physical laws, or a portion therein, by construction. Such biases introduce hard constrains which guarantee the model will adhere to some physical principle. This is exemplified well in the development of port-metriplectic neural networks [48], which incorporate thermodynamic principles into

the learning of complex physical systems by ensuring compliance with the first and second laws of thermodynamics through network architectural design.

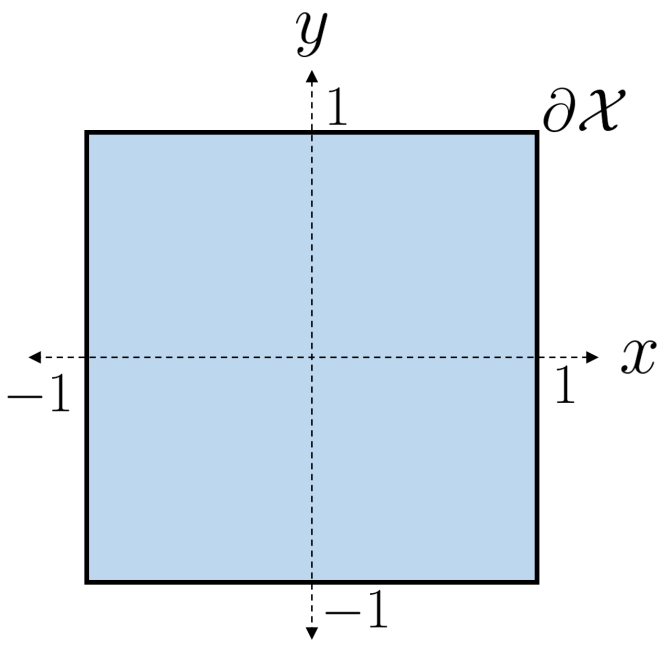



Figure 1.6: Physical domain of 2D heat sink model. The temperature is specified on the boundaries while the interior temperature is modeled by Eq. 1.26.

A simpler yet illustrative example of incorporating inductive bias for boundary conditions is evident in the context of modeling the solution of the diffusion equation (Eq. 1.26). A thermal model for a conductive heat sink is shown in Fig. 1.6 in which temperature on the physical domain $\mathcal{X} = [-1, 1]^2$ satisfies Eq. 1.26, and a Dirichlet boundary of $g(x, y, t)$ sets the temperature on the boundaries $\partial\mathcal{X}$ as a function of time. Additionally, a function $f(x, y, t)$ models a heat source in the computational domain as a function of time.

$$\frac{\partial u(x, y, t)}{\partial t} = \eta \nabla^2 u(x, y, t) + f(x, y, t) \ , \ u(t = 0) = u_0 \tag{1.26}$$

When designing a model $u_\phi(u_0, t, x, y, \eta, f(x, y, t))$ to predict temperatures $u(x, y, t)$ from the initial condition $u_0$, time $t$, spatial coordinates $x$ and $y$, conductivity $\eta$, and source function $f(x, y, t)$, enforcing boundary conditions can be seamlessly integrated into the model's architecture, even while employing neural networks as flexible parametric models. This approach guarantees that the model adheres to at least some of the physical constraints intrinsic to system. By structurally embedding these boundary conditions, the model can more accurately and reliably predict the behavior of the system under various scenarios.

By carefully selecting the form of the model $u_\phi(u_0, t, x, y, \eta, f(x, t))$, the boundary condi-

tions can be satisfied for any input conditions, regardless of the output of the neural network. A particular choice of model may therefore be given by

$$u_\phi(u_0, t, x, y, \eta, f(x,t)) = (1-y^2)(1-x^2)v_\phi(u_0, t, x, t, \eta, f(x,t)) + g(x,y,t)\ , \tag{1.27}$$

where $v_\phi(u_0, t, x, t, \eta, h(x,t))$ is a flexible parametric model such as a neural network. In this case the specified boundary conditions are always satisfied since $u_\phi(x = \pm 1) = g(x = \pm 1, y, t)$ and $u_\phi(y = \pm 1) = g(x, y = \pm 1, t)$. This simple example is intended to clearly illustrate the incorporation of inductive biases into models while retaining the flexibility that machine learning provides. However, it is noted that this simple example of an inductive bias will ensure that the boundary conditions are satisfied, but it will not ensure that predictions adhere to Eq. 1.26.

More complex methods of enforcing physical constraints are present in the literature, although usually they require customization to a particular application. For exaple, constrained sparse Galerkin regression (CSGR) by Loiseau and Brunton [49] presents a method which introduces inductive biases by combining sparse identification techniques with hard physical constraints such as conservation laws. This methodology exemplifies the potential of embedding physical laws directly into machine learning models, leading to more robust and physically consistent predictions. However, designing the constraints is an arduous process which must be performed for each application.

Srivastava and Duraisamy's LIFE framework [50] takes a slightly different approach, emphasizing the design of physics-informed feature spaces and constructing robust maps for feature augmentation in aerodynamic modeling. However, the method still necessitates designing augmentation functions which are application-specific. This framework showcases the importance of carefully selected features that are informed by the underlying physics in combination with a carefully designed method to maintain generalizability. Such careful consideration enhances the model's predictive capability and generalizability through the use of inductive biases.

As we delve deeper into the realm of physics-aware ML, it becomes apparent that these approaches are often highly application-specific. The choice between employing inductive or learning biases—or a blend of both—hinges on the problem at hand, the availability of data, and the specific physical equations involved. For example, in fluid dynamics, as discussed in the review by Duraisamy et al. [51], the complexity and computational demands of modeling turbulence necessitate a tailored approach that integrates machine learning with foundational physical knowledge.

In summary, physics-aware machine learning represents a significant advancement in com-

putational physics. By leveraging inductive and learning biases, these models transcend the limitations of traditional data-driven approaches, offering enhanced predictive capabilities and a deeper understanding of complex physical phenomena. This integration of machine learning with physical principles is not just an academic exercise; it is a crucial step towards developing innovative solutions across various domains of physics, where the empirical power of machine learning is seamlessly combined with the foundational rigor of physical laws.

## 1.8 Contributions

In this thesis, we explore physics-aware machine learning for improving the efficiency and accuracy of physical models, showcasing how specific applications can influence method development while also providing generalizable frameworks. The initial studies focus on tailored applications while the final work introduces a more general framework which is broadly applicable for solving a range of forward and inverse problems with physically-consistent machine learning. The primary contributions and insights from the four studies presented are summarized as follows:

- **Variational Autoencoders in Computational Physics:** VAEs are used to extract disentangled representations correlating with physically-relevant parameters from complex data. The significant insight from this study is that incorporating even minimal prior physical knowledge greatly enhances the learning process's robustness and accuracy, advocating for the integration of physical principles in machine learning models for improved physics modeling and generalization.

- **Modeling Dynamical Systems through Machine Learning:** Focusing on cathodic electrophoretic deposition, this contribution utilizes variational inference to address and refine the limitations of a baseline physical model, integrating ML augmentations to preserve physical interpretability while boosting predictive accuracy and model generalizability. This showcases the potential of combining machine learning with traditional physics models to deepen our understanding and manipulation of complex systems. Additionally, this results in a model which may advance the state of the art in e-coat modeling by providing insights into physical principles missing from current models.

- **Machine Learning Framework for Reduced Order Modeling of Non-Equilibrium Gas Dynamics:** Employing a novel machine learning framework informed by rate-distortion theory, this study further develops a well known reduced-order model which upholds physical constraints such as mass and energy conservation,

focusing on an innovative clustering strategy for pseudo-states. This approach significantly improves ROM accuracy over the baseline methods presented. It also illustrates the inherent trade off between interpretability and accuracy in physical modeling. This improved accuracy ROM learned in zero spatial dimensions can be implemented in more complex hypersonic simulations in higher spatial dimensions to render such simulations computationally feasible with state of the art accuracy relative to other ROM strategies.

- **Physical Consistency in Score-Based Generative Models:** The final study of this work introduces a framework for ensuring physical consistency in score-based generative models, combining advanced generative modeling with physical laws to address both forward and inverse problems efficiently. This method reduces computational demands while maintaining prediction accuracy and consistency with PDEs describing the system, embodying the ideal of physics-aware machine learning in which models are constrained to physical principles. Although this study leverages basic sampling strategies, the computational cost of solving forward and inverse problems is similar to a second order finite difference solver in generating solutions to the Darcy flow equations. With the adoption of more efficient sampling methods and fine tuning model architectures, it is likely that samples can be generated orders of magnitude faster than state of the art methods for solving PDEs numerically.

The research presented in this thesis represents a step towards the integration of machine learning with physical sciences, demonstrating nuanced and deeply synergistic approaches to tackling both theoretical and practical problems in physics. Each study, while distinct in its application and methodology, collectively contributes to a cohesive vision of physics-aware machine learning and improving the efficiency and accuracy of physical models over traditional methods. Each also emphasizes the critical importance of incorporating physical principles into machine learning models to enhance their interpretability, reliability, and generalizability.

A central theme that emerges across these studies is the transformative potential of embedding physical knowledge into machine learning frameworks. From the disentangled representations in computational physics to the refinement of physical models in cathodic electrophoretic deposition and the development of reduced-order models for non-equilibrium gas dynamics, each contribution underscores the value of a physics-informed perspective. This approach also highlights machine learning's role as a complementary tool rather than a replacement for traditional physics-based methods. The incorporation of machine learning into physical models is additionally shown to improve efficiency and accuracy over state of

the art methods in reduced order modeling for non equilibrium gas dynamics and solving forward and inverse problems using score-based generative models. This emphasizes that ML in physical modeling can provide benefits over traditional physical modeling when implemented correctly to ensure physical principles are still upheld.

Furthermore, the progression from specific applications to the development of a generalizable framework illustrates the scalability of such methods. The initial focus on tailored applications provides a solid foundation, demonstrating some of the practical benefits and feasibility of incorporating machine learning into physics. The subsequent transition to a more general framework showcases the potential adaptability of these principles across different domains, suggesting a broad potential impact on various fields within physical sciences and beyond.

The integration of differential equations in score-based generative models represents a pinnacle of this thesis, bridging advanced computational techniques with fundamental physics laws. This approach not only ensures physical consistency but also opens new avenues for solving forward and inverse problems with greater flexibility, efficiency, and potentially accuracy. By reducing the computational burden traditionally associated with computing solutions to PDEs, this method exemplifies the practical benefits of physics-aware machine learning, offering a scalable solution that can be adapted to diverse applications. The framework has the potential to greatly improve accuracy, efficiency, and flexibility in solving a variety of physical modeling challenges.

The works presented here illustrate that the fusion of machine learning with physics is not merely a matter of applying new tools to old problems but involves a fundamental rethinking of how we model and understand physical systems. The shared principles across the studies—such as the importance of physical knowledge integration, the pursuit of interpretability and generalizability, the development of scalable, efficient frameworks, and the improvement in both accuracy and efficiency over traditional modeling techniques—form a solid foundation for future research in this interdisciplinary field. Continuing to explore the frontiers of physics-aware machine learning, the insights and methodologies developed in this work will hopefully serve as guideposts, driving towards more robust, accurate, and versatile modeling capabilities that harness the best of both worlds.

# CHAPTER 2

# Probability and Machine Learning Background

This chapter presents essential background material in probability and machine learning methodologies useful for understanding some of the methods presented and developed in the later chapters. It aims to provide a concise yet comprehensive overview of the theoretical foundations necessary for understanding the advanced computational techniques employed in subsequent chapters. By introducing essential principles from these domains, this chapter sets the stage for the exploration of sophisticated models and algorithms which define the approaches that follow.

## 2.1 Information Theory

Information theory, pioneered by Claude Shannon [52], is a branch of applied mathematics and engineering involving the quantification, storage, and communication of information. It provides a quantifiable measure of information in the form of entropy, mutual information, and other quantities, and has wide applications in telecommunications, cryptography, and, more recently, machine learning.

In information theory, understanding the distinction between discrete and continuous settings is essential. Discrete settings deal with variables that have a specific and countable set of outcomes, typical in digital communication. In contrast, continuous settings involve variables with a continuous range of values, common in analog systems. The fundamental concepts of information theory are defined differently in these two settings.

*Entropy*, denoted by $H(X)$ for a discrete random variable $X$ with probability mass function (PMF) $P(x)$, measures the uncertainty or unpredictability of a variable's state. It is defined as

$$H(X) = -\sum_{x \in X} P(x) \log P(x) \, , \tag{2.1}$$

and provides a clear interpretation as the average number of bits required to represent the state of the random variable $X$ (a non-negative quantity). In continuous settings, differential entropy $h(X)$ is used, defined for a continuous random variable $X$ with probability density function (PDF) $p(x)$ as

$$h(X) = -\int_{-\infty}^{\infty} p(x) \log p(x)\, dx \ . \tag{2.2}$$

While differential entropy shares similarities with discrete entropy, it has different properties and interpretations. Differential entropy can be negative, unlike its discrete counterpart, rendering its interpretation more difficult to define. However, it can still be considered a measure of the uncertainty associated with a continuous random variable.

*Conditional entropy* measures the average uncertainty in a variable $X$ given that another variable $Y$ is known. For discrete variables,

$$H(X|Y) = -\sum_{y \in Y} P(y) \sum_{x \in X} P(x|y) \log P(x|y) \ . \tag{2.3}$$

Conditional entropy is always less than or equal to entropy: $H(X|Y) \leq H(X)$. This is because additional information in the form of conditioning $Y$ cannot increase uncertainty. Extra knowledge will typically reduce the uncertainty about a variable. However, in the limiting case of $Y$ providing no additional information about $X$, the conditional entropy is equal to the entropy: $H(X|Y) = H(X)$.

For continuous variables, conditional differential entropy is defined as

$$h(X|Y) = -\int_{-\infty}^{\infty} p(y) \left( \int_{-\infty}^{\infty} p(x|y) \log p(x|y)\, dx \right) dy \ , \tag{2.4}$$

and has a similar interpretation as discrete conditional entropy.

Another critical concept is the *Kullback-Leibler Divergence* (KL Divergence) [53], a measure of how one PMF $P(x)$ differs from a second PMF $Q(x)$ on the same random variable. For discrete variables, it is expressed as

$$D_{KL}(P(x)||Q(x)) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)} \ . \tag{2.5}$$

For a continuous variable given two PDFs $p(x)$ and $q(x)$, KL divergence is given by

$$D_{KL}(p(x)||q(x)) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)}\, dx \ . \tag{2.6}$$

It is particularly useful in quantifying the difference between two probabilistic models or

distributions. This divergence is a fundamental tool in identifying the "distance" between distributions, a concept widely used in machine learning for tasks such as model comparison and anomaly detection. Notably, it is always non-negative in both the discrete and continuous cases.

*Mutual Information* is a measure of the amount of information that one random variable contains about another. It is defined for discrete variables as

$$I(X;Y) = \sum_{x \in X, y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)} \,, \tag{2.7}$$

and in continuous settings by

$$I(X;Y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \, dy \, dx \,. \tag{2.8}$$

Mutual information can be interpreted as the reduction in uncertainty of one variable given that the other is observed, and it is symmetric: $I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$. For instance, if the observation of $Y$ entirely determines the value of $X$, the mutual information $I(X;Y)$ is equal to the entropy of $X$ as the conditional entropy is zero ($H(X|Y) = 0$) since there is no uncertainty. Mutual information can also be expressed as the KL divergence between the joint distribution and the product of the marginal distributions: $I(X;Y) = D_{KL}(P(x,y)||P(x)P(y))$, again providing an interpretation as a quantification of the correlation between two random variables. These definitions are equivalent in the continuous case, and mutual information is non-negative in both cases.

An important and fundamental concept in information theory is the data processing inequality (DPI). It states that processing data cannot increase the information it contains [54]. Formally, for a Markov chain of random variables $X - Y - Z$ where $Y$ is a function of $X$ and $Z$ is a function of $Y$, the DPI asserts that $I(X;Y) \geq I(X;Z)$. This indicates that the mutual information between $X$ and $Z$ is less than or equal to the mutual information between $X$ and $Y$, meaning that it is impossible for $Z$ to contain more information about $X$ than $Y$ contains about $X$. In any data processing chain, information can only be preserved or lost, but never gained.

Information theory is profoundly relevant in machine learning. Entropy and differential entropy can be utilized to understand the uncertainty and information content in datasets and model predictions. Conditional entropy is crucial in feature selection and understanding dependencies in datasets. KL Divergence is extensively used in model comparison and anomaly detection. Mutual information is a powerful tool for feature selection and understanding complex dependencies in data.

In physics-informed machine learning, these concepts help in integrating physical laws with data-driven models. For instance, mutual information can quantify the alignment between a model's predictions and physical laws. KL Divergence is critical in analyzing how well one distribution aligns with another. The DPI can be leveraged to create more robust optimization methods in developing ROMs, as illustrated in Chapter 5.

In this work, information theory serves as a cornerstone, offering mathematically rigorous methodologies for both the development and analysis of various models and techniques. Its principles, ranging from entropy to mutual information, provide a framework for quantifying and understanding the informational dynamics within many models. This theoretical backbone not only enhances the robustness of our methodological approaches but also ensures a deeper and more nuanced understanding of the underlying processes. In essence, the application of information theory aids in ensuring that methods and models are grounded in a solid mathematical foundation, which is critical for advancing the field of machine learning, particularly for applications in which interpretability and robustness are paramount.

### 2.1.1 Rate Distortion Theory

Rate distortion theory [54, 55, 56, 57, 58] serves as a valuable tool for analyzing encoding-decoding systems and establishing the theoretically optimal performance of such systems. It serves to analyze the reconstruction performance of encoding-decoding systems or algorithms, and comparing their relative performance. Consider a random variable $Y$ distributed according to a *source* distribution $p(y)$. In transmitting samples $y \in \mathbb{R}^n$ of this random variable from the source to a *destination* or *receiver*, reducing the amount of information required to represent the sample improves the efficiency of transmission. Therefore, an encoder is sought such that samples from the source are compressed to a random variable $X$ which can be stored and transmitted with fewer bits. This encoder is defined as a function $f : \mathbb{R}^n \to \mathbb{R}^m$ such that the source and encoded representation are related by $X = f(Y)$.

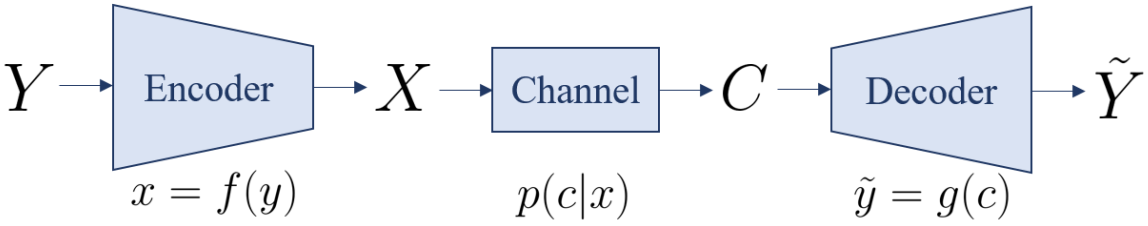



Figure 2.1: Encoding-decoding system with noisy channel transmission.

After samples of the source have been encoded, they can be transmitted to the destination

at a reduced storage cost over transmitting them directly, and such transmission is modeled by a *channel*. In the context of information theory, a channel refers to the medium through which information is transmitted from a source to a destination. It is characterized by the properties that define how information is conveyed and the potential distortions or noise that may affect the transmission. Mathematically, a channel can be described by a conditional probability distribution $p(c|x)$ that relates the input signal $X$ to the output signal $C$, taking into account the channel's capacity to faithfully transmit information as well as any errors or noise introduced during the process. If there exists a nonzero probability of transmission error in the channel, the channel is said to be *noisy*. In other words, a noisy channel is one in which $p(c = x|x) < 1$.

After the transmission arrives at the destination, it is decoded by a function $g : \mathbb{R}^m \to \mathbb{R}^n$ to a form a *reconstruction* $\tilde{Y}$ of the original source $Y$. This entire process of encoding, transmission, and decoding defines an encoding-decoding system, illustrated in Figure 2.1. Rate distortion theory defines a framework for analyzing such systems by defining the trade-off between the average number of bits of the compressed representation, known as the *rate*, and the average error of the reconstruction compared to the original source, known as the *distortion*. For a given source encoded at rate $R$, rate distortion theory defines the minimum achievable distortion $D$ across all possible encoding-decoding systems.

The primary objective in RD is to reconstruct the source accurately by closely approximating it with the reconstructed output. However, the specific definition of accuracy can vary depending on the problem at hand. In other words, the criteria for evaluating the fidelity of reconstruction may vary based on the specific context and requirements of the system being analyzed. The *distortion function* is used to determine the fidelity of the reconstruction when compared to the source and is given by a function

$$d : \mathcal{Y} \times \tilde{\mathcal{Y}} \to \mathbb{R}^+ .$$

For example, the well known $L^2$-norm $d(\mathbf{y}, \tilde{\mathbf{y}}) = \|\mathbf{y} - \tilde{\mathbf{y}}\|_2$ is a common distortion function, but many others exist. The distortion for an encoding-decoding system is then defined as

$$D = \mathbb{E}_{Y,\tilde{Y} \sim p(\mathbf{y})p(\tilde{\mathbf{y}}|\mathbf{y})}[d(Y, \tilde{Y})] ,$$

which is the expected value of the distortion function over the source and reconstruction.

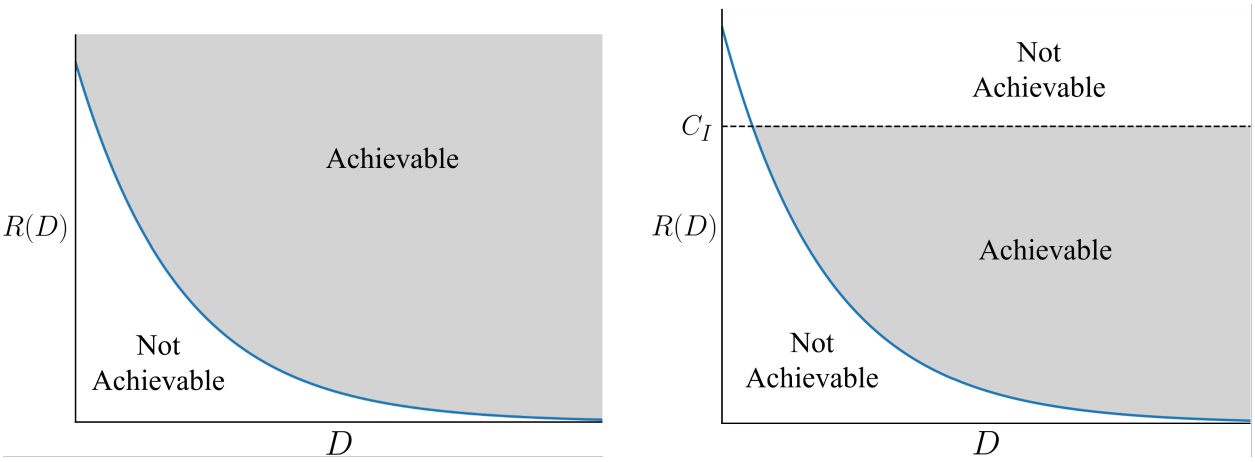
First, a noiseless channel in which $p(c = x|x) = 1$ is considered prior to the discussion of RD with noisy channels. RD seeks to determine the extent of compression that can be achieved given a specified acceptable level of distortion. Conversely but equivalently, it investigates the minimum achievable distortion for a given amount of compression.

We refer the reader to Ref. [54] for a more in-depth discussion and understanding of rate distortion theory. However, some of the primary results useful for understanding our methods are presented here. The central concept in RD is the *rate distortion function* $R(D)$, which quantifies the minimum amount of data compression needed $R$ for a given source while keeping the distortion (or loss of information) below a specified level $D$. It provides a theoretical limit for how much a signal can be compressed without exceeding a certain level of information loss. The rate distortion function is defined in Theorem 1, and readers are referred to Ref. [54] for a mathematical proof.

**Theorem 1.** *The rate distortion function for an i.i.d. source $Y$ with distribution $p(y)$ and bounded distortion function $d(y, \tilde{y})$ is given by the minimization of mutual information between source and reconstruction with respect to all possible encoding and decoding functions subject to a bounded distortion. Thus,*

$$R(D) = \min_{p(\tilde{y}|y):\mathbb{E}_{Y,\tilde{Y}\sim p(y)p(\tilde{y}|y)}[d(Y,\tilde{Y})]\leq D} I(Y;\tilde{Y})$$

*is the minimum achievable rate for distortion $D$.*




(a) A noiseless channel does not limit the achievable rate with infinite channel capacity, allowing a lower distortion.

(b) A noisy channel with limited channel capacity further limits minimum achievable distortion.

Figure 2.2: Illustration of the rate distortion plane showing the region of achievability. An $R(D)$ curve exists for each source distribution, and each encoding-decoding system can be plotted as a point in the achievable RD plane.

Theorem 1 says the minimum achievable rate (maximum achievable compression) is given by the minimum mutual information between the source and reconstruction subject to a

bounded distortion. This indicates that an encoding-decoding system exists which can compress the source to rate $R(D)$ while maintaining distortion $D$, but it does not provide a method for finding such functions. However, it importantly provides us with the concept of the *RD plane*. Notably, the rate distortion function is a monotonically decreasing function of the distortion [54], indicating that $R(D_1) \geq R(D_2)$ if $D_1 < D_2$, as illustrated in Figure 2.2.

Theoretically, the rate distortion function can be computed for any source at a given distortion value. For instance, in the case of ROMs, a particular ROM contains some rate of compression often related to the size of the compressed representation. Additionally, the average distortion in the reconstructed ROM predictions and the full order predictions can be compared to compute the distortion. The ROM can therefore be plotted as a point in the RD plane, providing a framework for directly comparing ROMs of different types.

Although comparing encoding-decoding systems in the RD plane provides a relative measure of performance, the optimal performance in terms of computing the RD curve itself is often prohibitively unstable or impractical to accurately compute. Additionally, Theorem 1 assumes a noiseless channel, but in some applications, limitations of a noisy channel can make it infeasible to achieve a particular distortion level due to limits on the capacity of the channel to transmit information. This issue can be formalized by leveraging another concept from information theory known as channel coding.

Suppose samples from some distribution $X \sim p(x)$ are transmitted through a noisy channel denoted by $p(c|x)$. The *channel capacity* in information theory is defined as

$$C_I = \max_{p(x)} I(X; C) \ ,$$

where $C_I$ signifies the maximum rate at which information can be transmitted through the channel with an arbitrarily low probability of error.

Consider the encoding-decoding system defined in Figure 2.1 in which the source is encoded through a deterministic function, transmitted through a noisy channel, and decoded by a deterministic function. This is equivalent to having a probabilistic encoder $p(c|y)$ rather than separating it into an encoding function $f(y)$ and noisy channel $p(c|x)$. In Chapter 5, this interpretation is employed; however, leaving the encoder and channel separate aids in illustrating a useful result.

The *source-channel separation theorem with distortion* [54] tells us that distortion $D$ is achievable if and only if $C_I > R(D)$. The channel capacity must be greater than the rate distortion function of the source to have any hope of achieving distortion D. This is an important concept which is critical to the presented method of encouraging a more robust optimization process informed by rate distortion theory (Chapter 5).

In this work, RD is used as a tool of analysis and development of encoding-decoding systems; however, computing the true RD curve is not addressed. Some recent works in the literature [59, 60, 61] have addressed the difficulties in computing the RD function, but RD in this work is leveraged as a tool to develop and compare the relative performance of compression algorithms primarily through the idea of the RD plane.

## 2.2 Variational Inference

Variational inference (VI) methods [62, 63, 64, 65] represent a powerful and versatile approach in probabilistic and Bayesian modeling, particularly notable for its efficiency in handling high-dimensional problems. In contrast to Markov Chain Monte Carlo (MCMC) methods [66, 67, 68], variational inference formulates posterior inference as an optimization problem. This approach involves approximating the Bayesian posterior with a parametric distribution, focusing on learning the parameters of this distribution rather than the typically intractable Bayesian posterior itself. The optimization objective in VI is to minimize the KL divergence between the variational approximation and the true posterior.

This optimization-based framework of VI offers several advantages over MCMC and other methods. Firstly, VI tends to be faster and more scalable, especially for large datasets, making it more practical in various applications. However, it may sometimes provide less accurate estimations of the posterior, particularly in terms of underestimating variance [69]. On the other hand, while MCMC methods often yield more accurate and consistent estimations of the posterior distribution and are more flexible with distribution types, they are computationally intensive and slower, particularly for large datasets.

The optimization nature of VI aligns well with the principles underlying many machine learning techniques, facilitating its integration into ML frameworks. This natural coupling allows for more scalable models, especially in the development of models that are both robust and computationally efficient. Another consideration is the adaptability of VI in the context of different model complexities and data structures, making it a flexible choice in a variety of scenarios.

Despite these advantages, it's important to recognize the limitations of VI. The approximation nature of VI can lead to inaccuracies, particularly in cases where the true posterior is complex or multi-modal [69]. This is a key area where MCMC methods, despite their computational intensity, may offer more reliable results. Furthermore, the choice of the variational family can significantly impact the performance of VI, requiring careful consideration and expertise.

The alignment of Variational Inference (VI) with machine learning frameworks, primarily

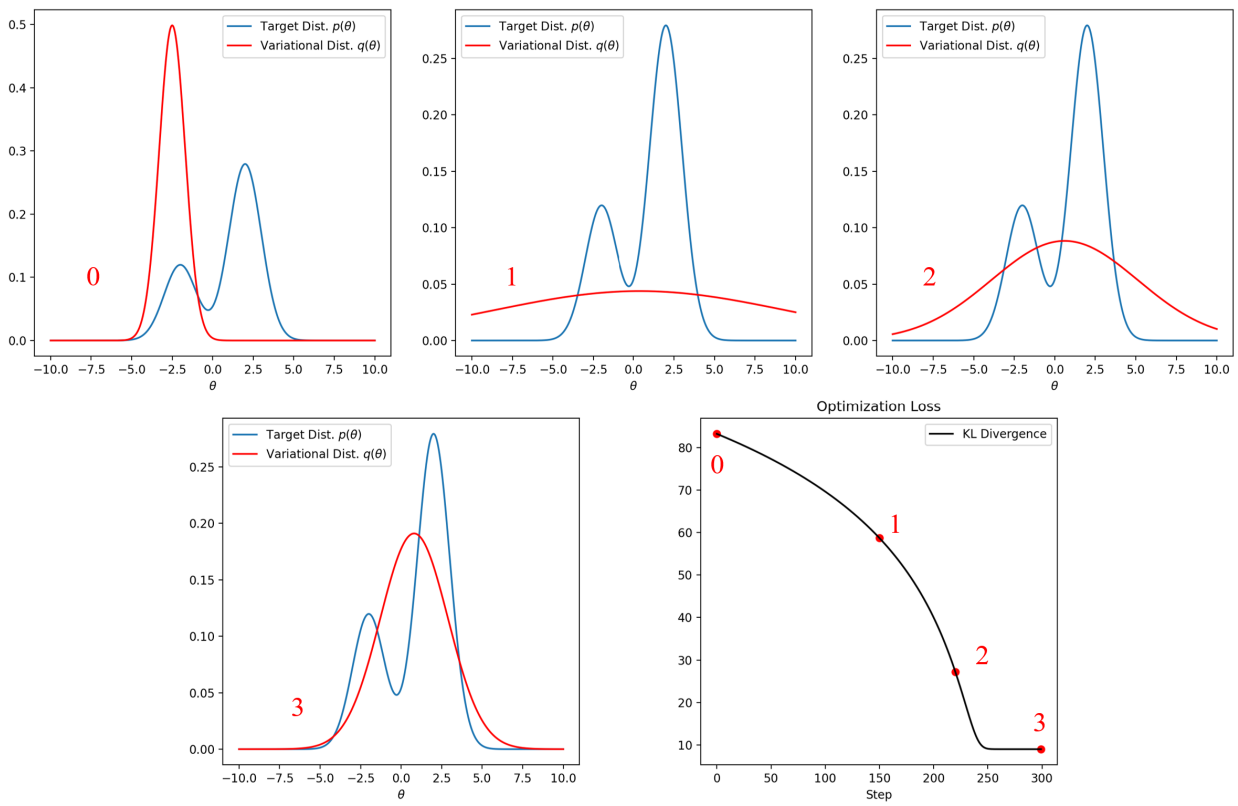



Figure 2.3: Variational inference minimizes KL divergence between a parametric variational approximation and a target distribution. The variational approximation is initialized (0), and the optimization proceeds until the KL divergence is minimized (3).

due to its optimization-based approach, enhances its scalability and efficiency, making it a particularly attractive option in modern probabilistic modeling. These characteristics of VI, especially its computational efficiency and ability to handle large datasets effectively, are crucial in the context of physics-informed machine learning. In this work, VI techniques are predominantly employed, as they seamlessly integrate with machine learning methodologies, which are foundational to the presented approaches. The decision to use VI over other methods like MCMC is driven by the specific demands of physics-informed problems where computational efficiency and the ability to handle complex, high-dimensional data are paramount. This choice ensures that our models are not only robust and precise but also practically feasible for large-scale applications in physics.

Variational inference assumes a parametric distribution $q_\phi(\theta|\mathcal{D})$ which is used to approximate the true posterior distribution $p(\theta|\mathcal{D})$. The goal of variational inference is to minimize the discrepancy between the true and approximate distributions by optimizing the distribution parameters. To achieve this, a measure of discrepancy or distance between two distributions, called the Kullback-Leibler divergence (KL divergence), is minimized between

the two distributions. The KL divergence is given by

$$\mathrm{KL}\left[\mathrm{q}_\phi(\theta|\mathcal{D})\|\mathrm{p}(\theta|\mathcal{D})\right] = \mathbb{E}_{\theta\sim\mathrm{q}_\phi(\theta|\mathcal{D})}\left[\log\left(\frac{\mathrm{q}_\phi(\theta|\mathcal{D})}{\mathrm{p}(\theta|\mathcal{D})}\right)\right] . \tag{2.9}$$

Variational inference seeks to minimize this by solving the optimization problem

$$\phi^* = \arg\min_\phi \mathrm{KL}\left[\mathrm{q}_\phi(\theta|\mathcal{D})\|\mathrm{p}(\theta|\mathcal{D})\right] . \tag{2.10}$$

Using Bayes' rule and defining the evidence lower bound (ELBO) $\mathcal{L}(\theta,\phi)$ as

$$\mathcal{L}(\theta,\phi) = \mathbb{E}_q\left[\log\left(\frac{p(\theta,\mathcal{D})}{q_\phi(\theta|\mathcal{D})}\right)\right] = \mathbb{E}_q\left[\log p(\theta,\mathcal{D})\right] - \mathbb{E}_q\left[\log q_\phi(\theta|\mathcal{D})\right] , \tag{2.11}$$

the KL divergence can be written as a function of the ELBO by

$$\mathrm{KL}\left[\mathrm{q}_\phi(\theta|\mathcal{D})\|\mathrm{p}(\theta|\mathcal{D})\right] = -\mathcal{L}(\theta,\phi) + \log\left(\mathrm{p}(\mathcal{D})\right) . \tag{2.12}$$

As the *evidence* $p(\mathcal{D})$ depends only on the distribution of the data and is constant with changes in the parametric distribution $q_\phi(\theta|\mathcal{D})$, the optimization problem of Eq. 2.10 is equivalent to minimizing the negative ELBO as

$$\phi^* = \arg\min_\phi -\mathcal{L}(\theta,\phi) .$$

An example of solving this optimization problem to approximate a multimodal target distribution $p(\theta)$ with a Gaussian distribution $q(\theta)$ is visualized in Fig. 2.3. The variational approximation does not perfectly represent the target distribution, but it approximates it as closely as possible given the distribution class. Using a Gaussian approximation as the variational density is a common and inexpensive choice.

### 2.2.1 Gaussian variational inference

Fixed-form variational inference uses simple parametric distributions as the variational posterior. An independent Gaussian distribution can be chosen such that the variational posterior does not depend on the data by $q_\phi(\theta|\mathcal{D}) = \mathcal{N}(\mu_\phi, \Sigma_\phi)$. The parameters which define the distribution $\phi = [\mu_\phi, \Sigma_\phi]$ to be optimized consist of the mean $\mu_\phi \in \mathbb{R}^m$ and covariance $\Sigma_\phi \in \mathbb{R}^{m\times m}$ matrix. Note that the assumption of independence limits the covariance matrix to $m$ trainable parameters and $2m$ total trainable parameters.

However, the distribution can also be constructed to depend on individual data samples

to model a conditional distribution by parameterizing the mean and covariance as functions of $y$. This parameterized form is given by $q_\phi(\theta|y) = \mathcal{N}(\mu_\phi(y), \Sigma_\phi(y))$ where $\mu_\phi(y) : \mathbb{R}^n \to \mathbb{R}^m$ and $\Sigma_\phi(y) : \mathbb{R}^n \to \mathbb{R}^{m\times m}$ are parametric functions defining the mean and covariance matrices as a function of a single input $y$.

Gradient-free optimization methods like the Nelder-Mead algorithm [70] tend to be less efficient and scale poorly with high-dimensional problems compared to gradient-based algorithms. Consequently, the effectiveness of these techniques is significantly enhanced by utilizing gradients. Therefore, designing parametric forms that are differentiable is advantageous to leverage the full potential of gradient-based optimization.

### 2.2.2 Variational Inference with Normalizing Flows

Adopting a Gaussian posterior in variational inference, as discussed in Sec. 2.2.1, is a common practice but may not always be ideal. This is particularly problematic when dealing with multimodal posterior distributions, where minimizing the KL divergence can lead to a variational posterior that fails to accurately capture any of the modes. This situation necessitates a more expressive and adaptable distribution model. Normalizing flows [62], which consist of a series of invertible transformations applied to samples from a simple probability distribution (like the standard normal distribution), offer a solution by enabling the creation of complex distributions. These distributions can be efficiently sampled, and their probability density function can be calculated, provided that the transformations are invertible [62]. This approach integrates well with variational inference, offering enhanced flexibility in modeling complex posteriors.

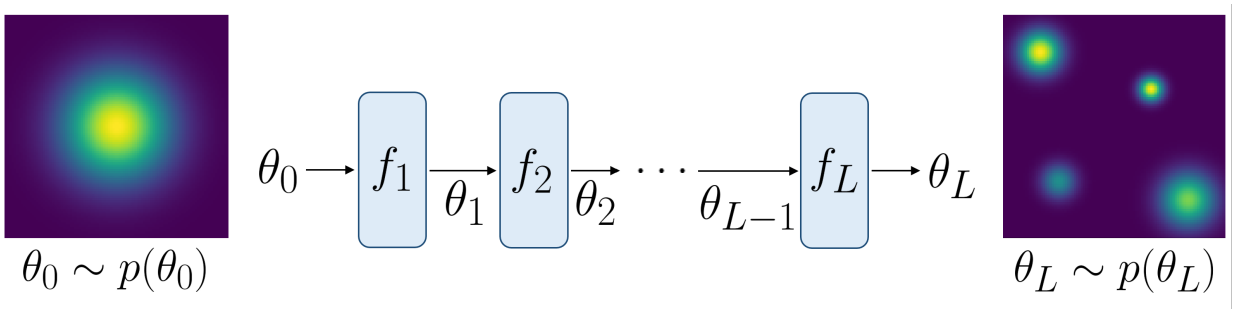



Figure 2.4: Normalizing flow consisting of $L$ invertible transformations to map samples from $p(\theta_0)$ to samples from $p(\theta_L)$ (2D example).

Considering smooth, invertible mappings $f : \mathbb{R}^d \to \mathbb{R}^d$ and random variable $\theta_0$ with PDF

$p(\theta_0)$, the PDF of the transformed random variable $\theta_1 = f_1(\theta_0)$ is given by

$$p(\theta_1) = p(\theta_0) \left| \det \frac{\partial f_1}{\partial \theta_0} \right|^{-1} .$$

An arbitrarily long sequence of flow functions of length $L$ can be composed such that

$$\theta_L = f_L \circ \cdots \circ f_2 \circ f_1(\theta_0)$$

and

$$\log p(\theta_L) = \log p(\theta_0) - \sum_{i=1}^{L} \log \left| \det \frac{\partial f_i}{\partial \theta_{i-1}} \right| \tag{2.13}$$

Many choices of flow function $f$ exist in the literature [71]. One simple and regularly employed flow function is that of planar flow [62], which will be used as a demonstrative example in this discussion. The planar flow function is given by

$$f_\phi(\theta) = \theta + \mathbf{u}\sigma(\mathbf{w}^T\theta + b)$$

where $\phi = \{\mathbf{u} \in \mathbb{R}^d, \mathbf{w} \in \mathbb{R}^d, \text{ and } b \in \mathbb{R}\}$ are the learnable flow parameters and $\sigma$ is a smooth element-wise non linear function.

Normalizing flows assume the fixed-form on the variational posterior as $q_\phi(\theta_L)$ where $\theta_L = f_L \circ \cdots \circ f_2 \circ f_1(\theta_0)$ and solve the minimization problem

$$q^*_\phi(\theta_L) = \min_\phi D_{KL}[q_\phi(\theta_L)||p(\theta|\mathcal{D})], \tag{2.14}$$

as in standard variational inference. The only difference is the choice of parameterization of the variational posterior. Figure 2.4 illustrates the transformation of samples through invertible functions to form an expressive and parametric distribution. The particular choice of flow function along with the quantity of transformation chosen significantly influences the expressiveness of such a distribution. However, employing normalizing flows can result in variational posterior distributions which are multimodal with complex shapes which are not possible with simple parametric distributions [**?** ][This citation will be filled in during revisions. Citation not yet available].

## 2.3 Generative Modeling

The overarching goal of generative modeling is to sample from an unknown data distribution given only samples in the form of data from such a distribution. Unconditional generative

modeling aims to sample from $p(\mathbf{y})$ given $N$ independent identically distributed (i.i.d.) samples $\{\mathbf{y}^{(i)}\}_{i=1}^{N}$ from the distribution.

Generative modeling techniques can be largely classified into three primary categories: likelihood-based models, implicit generative models, and the more recent diffusion and score-based generative models. A large array of likelihood-based techniques exist in the literature such as variational autoencoders (VAEs) [72, 73, 74], autoregressive models [75, 76, 77], energy-based models (EBMs) [78, 79], and normalizing flow models [62, 80, 81]. Likelihood-based approaches have been shown to be useful, yet impose restrictions on the model or distribution form to ensure a regularized distribution and facilitate likelihood computation during training. Implicit generative models such as generative adversarial networks (GANs) [82] are challenging to train due to the adversarial nature of training. Score-based approaches, on the other hand, model the score function (gradient of the log density) directly, bypassing restrictions on model or distribution form to represent a valid probability density [83].

In this work, our attention will be primarily centered on two distinct classes of generative models: variational autoencoders (VAEs) and score-based generative models. While each of these models possesses unique characteristics and are applied to different specific applications in the examples presented, they are intrinsically connected by a shared overarching objective of generative modeling. Variational autoencoders leverage the principles of variational inference to generate new data points that are statistically similar to the input data. On the other hand, score-based generative models, a relatively newer class of generative models, excel in generating high-quality samples by iteratively refining them following a score function derived from the data distribution. Despite their differing mechanisms and application contexts, both VAEs and score-based models play a pivotal role in our explorations of generative modeling. They exemplify the innovative ways in which machine learning can be harnessed to create data-driven, physics-aware models that not only generate new data samples but also may offer deeper insights into the underlying physical processes.

### 2.3.1 Variational Autoencoders

Variational autoencoders (VAEs) [72] and their variants [84, 85, 86, 87] aim to approximate an underlying distribution $p(\mathbf{y})$ of high-dimensional data through a two-step process. Compressed representations $\mathbf{z}$ are sampled from a low-dimensional - yet unknown - distribution $p(\mathbf{z})$. An encoding distribution $p(\mathbf{z}|\mathbf{y})$ and a decoding distribution are learned simultaneously by maximizing a bound on the likelihood of the data (i.e. the evidence lower bound (ELBO) [72]). Thus, a mapping from the high-dimensional space to a low-dimensional space and the corresponding inverse mapping is learned simultaneously, as shown in Figure 2.5,

allowing approximations of both $p(\mathbf{y})$ and $p(\mathbf{z})$. Learning the lower-dimensional representation, or *latent space*, can facilitate computationally-efficient data generation and extract only the information necessary to reconstruct the data [88]. VAEs have been successfully implemented in many physics-based applications including inverse problems [89], extracting physical parameters from spatio-temporal data [90], and constructing probabilistic reduced order models [91, 92], among others.

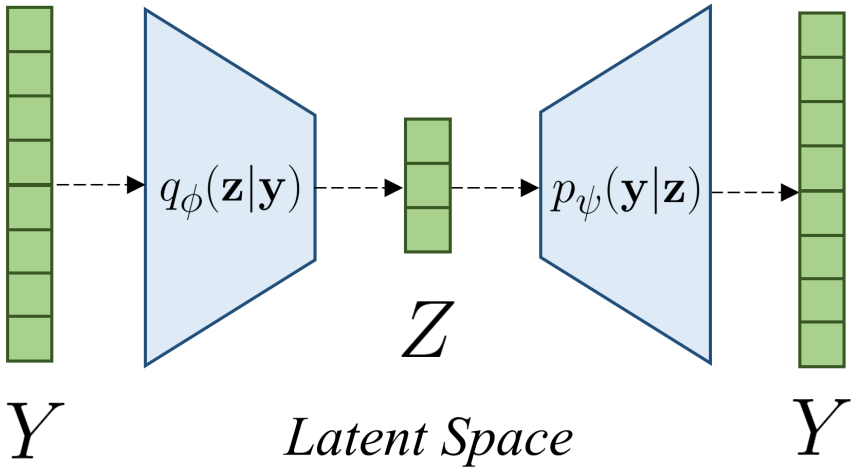



Figure 2.5: VAEs encode data samples to a distribution in the latent space with the aim of reconstructing the original input.

The VAE framework infers a latent-variable model by replacing the posterior $p(\mathbf{z}|\mathbf{y})$ with a parameterized approximating posterior $q_\phi(\mathbf{z}|\mathbf{y})$ [72], known as the encoding distribution. A parameterized decoding distribution $p_\psi(\mathbf{y}|\mathbf{z})$ is also constructed to predict data samples given samples from the latent space. Only the encoding distribution and the decoding distribution are learned in the VAE framework, but the *aggregated posterior* $q_\phi(\mathbf{z})$ (to the best of our knowledge, first referred to in this way by [93]), is of particular importance (discussed further in Chapter 3). It is defined as the marginal latent distribution induced by the encoder

$$q_\phi(\mathbf{z}) \triangleq \int_{\mathcal{Y}} p(\mathbf{y}) q_\phi(\mathbf{z}|\mathbf{y}) d\mathbf{y} \ , \tag{2.15}$$

where the true data distribution is denoted $p(\mathbf{y})$. The induced data distribution is the marginal output distribution induced by the decoder

$$p_\psi(\mathbf{y}) \triangleq \int_{\mathbb{R}^n} p(\mathbf{z}) p_\psi(\mathbf{y}|\mathbf{z}) d\mathbf{z} \ . \tag{2.16}$$

It is noted that the true data distribution is typically unknown; only samples of data $\{\mathbf{y}^{(i)}\}_{i=1}^N$

are available. The empirical data distribution is thus denoted $\hat{p}(\mathbf{y})$, and any expectation with respect to the empirical distribution is simply computed as an empirical average $\mathbb{E}_{\hat{p}(\mathbf{y})}[f(\mathbf{y})] \triangleq \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{y}^{(i)})$.

Learning the latent model is accomplished by simultaneously learning the encoding and decoding distributions through maximizing the evidence lower bound (ELBO), which is a lower bound on the log-likelihood [94]. To derive the ELBO loss, we begin by expanding the relative entropy between the data distribution and the *induced* data distribution

$$D_{KL}[p(\mathbf{y})||p_\psi(\mathbf{y})] = \mathbb{E}_{Y\sim p(\mathbf{y})}[\log p(\mathbf{y})] - \mathbb{E}_{Y\sim p(\mathbf{y})}[\log p_\psi(\mathbf{y})]$$

where the first term on the right hand side is the negative differential entropy $-H(Y)$. Noting that relative entropy $D_{KL}$ is always greater than or equal to zero and introducing Bayes' rule, we arrive at the following inequality

$$H(Y)+\mathbb{E}_{Y\sim p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z}|\mathbf{y})]] \leq \mathbb{E}_{p(\mathbf{y})}[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{y})}[\log p_\psi(\mathbf{y}|\mathbf{z})]] - \mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})]] \ .$$

Thus,

$$\mathbb{E}_{p(\mathbf{y})}[\log(p(\mathbf{y}))] \geq \mathbb{E}_{p(\mathbf{y})}[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{y})}[\log p_\psi(\mathbf{y}|\mathbf{z})]] - \mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})], \tag{2.17}$$

where $p(\mathbf{z})$ is a prior distribution. The prior is specified by the user in the classic VAE framework. The right-hand side in Eq. (2.17) is the well-known ELBO. Maximizing this lower bound on the log-likelihood of the data is done by minimizing the negative ELBO. The optimization is performed by learning the encoder and decoder parameterized as neural networks. The negative ELBO is defined as

$$-ELBO = \mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})]] + \mathbb{E}_{p(\mathbf{y})}[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{y})}[-\log p_\psi(\mathbf{y}|\mathbf{z})]], \tag{2.18}$$

and we assume $\mathcal{L}_{VAE} \approx -ELBO$, where the difference results in the expectation being evaluated over the empirical data distribution in $\mathcal{L}_{VAE}$. The VAE loss function is defined as

$$\mathcal{L}_{VAE} = \mathbb{E}_{\hat{p}(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})] + \mathbb{E}_{\hat{p}(\mathbf{y})}[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{y})}[-\log p_\psi(\mathbf{y}|\mathbf{z})]], \tag{2.19}$$

where the first term on the right-hand side is the regularization loss $\mathcal{L}_{REG}$ and drives the encoding distribution closer (in the sense of minimizing KL divergence) to the prior distribution. The second term on the right-hand side is the reconstruction error $\mathcal{L}_{REC}$ and encourages accurate reconstruction of the data.

Selecting the prior distribution as well as the parametric form of the encoding and decod-

ing distribution can allow closed form solutions to compute $\mathcal{L}_{VAE}$. The prior distribution is often conveniently chosen as a standard normal distribution $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, \mathbf{I}_{n\times n})$. The encoding and decoding distributions are also often chosen as factorized normal distributions $q_\phi(\mathbf{z}|\mathbf{y}) = \mathcal{N}(\mathbf{z}; \mu_\phi(\mathbf{y}), \text{diag}(\sigma_\phi(\mathbf{y})))$ and $p_\psi(\mathbf{y}|\mathbf{z}) = \mathcal{N}(\mathbf{y}; \mu_\psi(\mathbf{z}), \text{diag}(\sigma_\psi(\mathbf{y})))$, where the mean and log-variance of each distribution are functions parameterized by neural networks. Selecting the parameterized form of these distributions facilitates the reparameterization trick [72], allowing backpropagation through sampling operations during training. This selection of the prior, encoding, and decoding distributions allows a closed form solution to compute $\mathcal{L}_{VAE}$.

#### 2.3.1.1 $\beta$-VAE

The $\beta$-VAE objective gives greater weighting to the regularization loss,

$$\mathcal{L}_{\beta-VAE} = \beta\mathbb{E}_{\hat{p}(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})] + \mathbb{E}_{\hat{p}(\mathbf{y})}[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{y})}[-\log p_\psi(\mathbf{y}|\mathbf{z})]] \ .$$

This encourages greater regularization, often leading to improved disentanglement over the standard VAE loss [84]. It is worth noting that when $\beta = 1$, with a perfect encoder and decoder, the VAE loss reduces to the Bayes rule [95, 96]. More details on the $\beta$-VAE are provided in Section 2.3.1.2.

#### 2.3.1.2 Connections to RD

Rate distortion theory (Section 2.1.1) aids in a deeper understanding in the trade off and balance between the regularization and reconstruction losses of VAEs. Minimizing the $\beta$-VAE loss is closely tied to a solving a rate-distortion problem. Rearranging the VAE regularization loss ($\mathcal{L}_{REG}$) results in

$$\mathcal{L}_{REG} = \mathbb{E}_{\hat{p}(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})] = I_\phi(Y; Z) \ ,$$

which is equal to the mutual information between $Y$ and $Z$ according to the data and encoding distributions. Minimizing the $\beta$-VAE loss gives the optimization problem

$$\min_{\phi,\psi} \mathcal{L}_{\beta-VAE} = \min_{\phi,\psi} I_\phi(Y; Z) + \beta\mathbb{E}_{\hat{p}(\mathbf{y})p_\psi(\mathbf{z}|\mathbf{y})}[-\log p_\phi(\mathbf{y}|\mathbf{z})] \ .$$

This optimization problem is similar to minimizing the rate-distortion Lagrangian (Eq. 2.20) with $d(\mathbf{y}, \tilde{\mathbf{y}}) = -\log p_\phi(\mathbf{y}|\mathbf{z})$ and the mutual information $I_\phi(Y; Z)$ just an approximation to

the true mutual information $I(Y;Z)$.

$$\min \mathcal{J}(\beta) = \min_{p(\mathbf{z}|\mathbf{y})} I(Y;Z) + \beta(\mathbb{E}_{Y,Z}[d(\mathbf{y},\tilde{\mathbf{y}}(\mathbf{z})] - D). \tag{2.20}$$

Depending on $\beta$, solutions can be found at any location along the RD curve with each containing differing properties. RD curve for VAEs is simply an analogy: $\mathcal{L}_{REG}$ is considered the rate $R$ and $\mathcal{L}_{REC}$ is considered the distortion $D$.

With increased $\beta$, the $\beta$-VAE minimizes the mutual information between the data and the latent parameters, limiting reconstruction accuracy. In Ref. [97], disentanglement is illustrated to be caused inadvertently through the assumed factored form of the encoding distribution even though rotations of the latent space have no effect on the ELBO. However, their proof relies on training in the 'polarized' regime characterized by loss of information or 'posterior collapse' [98]. Training in this regime often requires increasing the weight of the regularization loss, necessarily decreasing reconstruction performance in the process. In our work, we illustrate disentanglement through training VAEs with the ELBO loss ($\beta = 1$), keeping reconstruction accuracy high. Ref. [97] additionally provides many useful insights into disentanglement for curious readers.

### 2.3.2 Score-based Generative Models

The pioneering work of Song et al. [99] ties the earliest diffusion models together under a common framework of score-based modeling, a framework which some of the work in Chapter 6 is based on. Some of the earliest score-based generative models include de-noising diffusion probabilistic models (DDPM) [100] and de-noising score matching with Langevin dynamics (SMLD) [101]. These early models initially showed great potential in advancing the state of the art in generative modeling. However, more recent approaches such as score-based generative modeling with SDEs [99], Poisson flow generative models (PFGM) [102, 103], and consistency trajectory models (CTM) [104] suggest that score-based generative models may be a superior choice among generative modeling techniques.

Diffusion models and score-based generative models have demonstrated state-of-the-art results in a variety of fields including natural language processing [105, 106, 107, 108], computer vision [109, 110, 111, 112], multi-model learning [113, 114, 115, 116, 117], drug design [118, 119], and medical imaging [120, 121], also opening new avenues of exploration in their respective domains. Models based on diffusion and score-based generation overwhelmingly dominate the current state-of-the-art [104] in generative tasks such as the Frechet inception distance (FID) score on the popular CIFAR-10 dataset [122]. We therefore do not address comparisons to other classes of generative models in this work, but rather focus on

a new breadth of areas to advance and apply these models to. A comprehensive study of the available methods and applications of diffusion models and score-based generative models is included in Ref. [123].

Score-based generative models are a class of generative modeling technique in which data is used to learn the Stein score function [83] (gradient of the log-density $\nabla_{\mathbf{y}} \log p(\mathbf{y})$). Once the score function is approximated, sampling from the distribution $p(\mathbf{y})$ can be performed using an iterative process known as Langevin dynamics [124, 125]. However, in practice it is difficult to accurately estimate the score function in all regions of the data space due to data sparsity. Recent works [101, 100] have thus suggested a key process of iteratively adding varying levels of noise to data samples and estimating the score function at each noise level. Adding noise effectively changes the underlying distribution, reducing sparsity in the data space and allowing for accurate score estimation in broader regions of the data space when compared to the original data distribution. Noise is sequentially added until the data is approximately distributed according to a prior distribution which is easy to sample from. Sample generation is then performed as the reverse of this process: sampling from the prior distribution and iteratively removing the noise until the data sample approximately belongs to the original data distribution. We refer readers to [126] for an intuitive and thorough explanation of score-based generative models.

This iterative process of sequentially adding noise to data can be viewed in a continuous sense as the solution to a stochastic differential equation (SDE) [99]. We consider a data distribution at $t = 0$ given by $p(\mathbf{y}(t = 0))$ where $t$ does not represent physical time, but is rather a modeling quantity used to simulate the SDE. This is the distribution from which data samples $\{\mathbf{y}^{(i)}\}_{i=1}^{N}$ are drawn, where $\mathbf{y}^{(i)} \sim p(\mathbf{y}(t = 0))$ and $\mathbf{y} \in \mathcal{X}$ (the data space). A stochastic differential equation (SDE) of the form

$$d\mathbf{y} = \mathbf{f}(\mathbf{y}, t)dt + g(t)d\mathbf{w} \tag{2.21}$$

describes the 'dynamics' of adding noise to data samples, where $\mathbf{f} : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ is a function known as the *drift coefficient*, $g : \mathbb{R} \to \mathbb{R}$ is a function known as the *diffusion coefficient*, and $\mathbf{w}$ denotes standard Brownian motion. The system 'dynamics' are constructed such that the distribution $p(\mathbf{y}(t = T))$ at final time $T$ approximately follows a prior distribution $\pi(\mathbf{y})$ which is easy to sample from, such as the standard normal distribution. The particular form of $\mathbf{f}(\mathbf{y}, t)$ and $g(t)$ are chosen such that $p(\mathbf{y}(t = T)) \approx \pi(\mathbf{y})$, or vice-versa. The SDE describes dynamics of adding random noise to samples from $p(\mathbf{y}(t = 0))$ until $\pi(\mathbf{y})$ is achieved. Suppose a sample $\mathbf{y}(t = 0)$ is drawn from $p(\mathbf{y}(t = 0))$, and the dynamics of the SDE are simulated. Viewing $\mathbf{y}(t)$ at any intermediate time step $t \in (0, T]$ is essentially a noisy version of the

initial sample. In this work, we assume a final time $T = 1$ in all experiments, though this is simply a convention.

In this continuous setting, the noising process can be reversed by using a well-known result [127] to solve the *reverse SDE* backward in time. This reverse SDE is given by

$$d\mathbf{y} = [\mathbf{f}(\mathbf{y}, t) - g^2(t)\nabla_{\mathbf{y}} \log p(\mathbf{y}(t))]dt + g(t)d\mathbf{w} \ , \tag{2.22}$$

where the distribution at any time $p(\mathbf{y}(t))$ is identical when solving the forward or reverse SDE. As $\mathbf{f}$ and $g$ are chosen particularly such that $p(\mathbf{y}(t = T)) \approx \pi(\mathbf{y})$, a sample $\mathbf{y}(t = T) \sim \pi(\mathbf{y})$ is drawn and the reverse SDE (Eq. 2.22) is simulated backward in time to obtain a sample $\mathbf{y}(t = 0)$ which is approximately drawn from $p(\mathbf{y}(t = 0))$.

To solve the reverse SDE, the score function $\nabla_{\mathbf{y}} \log p(\mathbf{y}(t))$ must be known for all times $t \in [0, T]$. Therefore, score-based generative modeling aims to model the score function using a parameterized model such as CNN-based or transformer-based machine learning models. This model $s_\phi(\mathbf{y}(t), t)$ should approximate the score function for all times $t \in [0, T]$ such that $s_\phi(\mathbf{y}(t), t) \approx \nabla_{\mathbf{y}} \log p(\mathbf{y}(t))$ and the dynamics of Eq. 2.22 are approximated by

$$d\mathbf{y} = [\mathbf{f}(\mathbf{y}, t) - g^2(t)s_\phi(\mathbf{y}(t), t)]dt + g(t)d\mathbf{w} \ . \tag{2.23}$$

The functions $\mathbf{f}$ and $g$ are often selected such that they induce a Gaussian transition kernel

$$p(\mathbf{y}(t)|\mathbf{y}(0)) = \mathcal{N}(\mathbf{y}(t); \boldsymbol{\mu}(\mathbf{y}(0), t), \boldsymbol{\sigma}^2(\mathbf{y}(0), t)I) \ , \tag{2.24}$$

where $\mathbf{y}(0)$ denotes a sample $\mathbf{y}(t = 0)$ from the data distribution. This means that the score of the transition kernel can be computed for any time $t$ without solving the forward SDE as $\nabla_{\mathbf{y}} \log p(\mathbf{y}(t)|\mathbf{y}(0)) = (\mu(\mathbf{y}(0), t) - \mathbf{y}(t))(\mu(\mathbf{y}(0), t) - \mathbf{y}(t))^T/\sigma^2(\mathbf{y}(0), t)$.

The model $s_\phi(\mathbf{y}(t), t)$ is trained using a score-matching objective [99]. In our work, we use denoising score-matching (DSM) [128, 99] in which the loss function is given by

$$\min_\phi \mathbb{E}_{t\sim\mathcal{U}[0,T]} \left[\lambda(t)\mathbb{E}_{p(\mathbf{y}(0))p(\mathbf{y}(t)|\mathbf{y}(0))}[||s_\phi(\mathbf{y}(t), t) - \nabla_{\mathbf{y}(t)} \log p(\mathbf{y}(t)|\mathbf{y}(0))||_2^2]\right] \tag{2.25}$$

where $\lambda(t) : [0, 1] \to \mathbb{R}_{>0}$ is a positive weighting function and $p(t) \sim \mathcal{U}[0, T]$. A closed form solution for the quantity $\nabla_{\mathbf{y}(t)} \log p(\mathbf{y}(t)|\mathbf{y}(0))$ can be easily obtained from the transition kernel in Eq. 2.24. We note that when $\lambda(t) = 1/2$, the optimal solution to Eq. 2.25 corresponds to $s_\phi(\mathbf{y}(t), t) = \nabla_{\mathbf{y}} \log p(\mathbf{y}(t))$ almost surely [128], which is sufficient to solve the reverse SDE in Eq. 2.22.

The probability flow ordinary differential equation (PF ODE) is similar to the reverse

SDE in that it can be solved in reverse time to approximately sample from $p(\mathbf{y}(t=0))$ given the distribution at $p(\mathbf{y}(t=T))$. Although this equation is an ODE rather than an SDE, the marginals at each intermediate time $t$ of the reverse SDE and PF ODE are identical [99]. The PF ODE is reversible in time and given by

$$d\mathbf{y} = \left[\mathbf{f}(\mathbf{y}, t) - \frac{1}{2} g^2(t) \nabla_x \log p(\mathbf{y}(t))\right] dt \, . \tag{2.26}$$

The use and benefits of sampling from both the reverse SDE and PF ODE backwards in time are thoroughly discussed in Chapter 6.

## 2.4 Neural Ordinary Differential Equations

Neural Ordinary Differential Equations (Neural ODEs) represent a novel intersection of machine learning and differential equations, offering a unique framework for modeling continuous-time dynamics. Introduced by Chen et al. [26], Neural ODEs are a type of deep learning model which transforms traditional neural network architectures by treating the depth (or layers) of the network as a continuous dimension. This is achieved by framing the network's evolution as the solution to an ODE problem, where the derivative of the hidden state with respect to depth (or time) is parameterized using a neural network.

The NeuralODE framework is informed by the observation that models such as residual networks often compose a sequence of transformation layers of the form

$$\mathbf{u}_{t+1} = \mathbf{u}_t + f_\phi(\mathbf{u}_t), \tag{2.27}$$

where $f_\phi(\mathbf{x}_t)$ is a parameterized neural network. This structure closely resembles the forward Euler finite discretization of a dynamical system given by

$$\frac{\partial \mathbf{u}}{\partial t} = f_\phi(\mathbf{u}), \tag{2.28}$$

which is the NeuralODE formulation. Predictions are created by solving the dynamical system in Eq. 2.28 with initial condition $\mathbf{u}(t=0) = \mathbf{u}_0$ using any black box ODE solver. This formulation can be far more efficient in terms of number of parameters required than the traditional additive residual approach of Eq. 2.27. However, they can also be computationally more expensive to train and predict with due to the need for solving the ODE. Training is performed by computing gradients via the adjoint method [26, 129] and leverage automatic differentiation where possible.

The modeling and prediction of dynamical systems is a natural application for NeuralODE in which the r.h.s. of a dynamical system is approximated directly with a neural network $f_\phi(\mathbf{u})$. Trained models for a variety of dynamical systems have have seen accurate generalization outside of training times and for prediction of dynamical systems trained with noisy data. Additionally, inductive biases can be easily incorporated into NeuralODE models [130].

An extension of NeuralODE to include the modeling of *event functions* which can instantaneously change the state of a system allows for even more flexibility [131]. This is particularly beneficial in scenarios where abrupt changes occur that are not accounted for in the model's dynamics. For example, it applies to a ball that suddenly changes direction upon bouncing on the ground, or the sudden and immediate activation of a heat source in thermal modeling. These instances require the ability to swiftly adapt to sudden shifts in conditions. We demonstrate an extension to these ideas in Chapter 4 in which gradients are passed through an instantaneous change in model form.

Although the NeuralODE framework provides very flexible and powerful models for a variety of applications, one of the primary drawbacks in its modeling of dynamical systems is the decrease in efficiency over additive residual or other standard approaches. In contrast, other advanced methods such as operator learning techniques can improve the efficiency of dynamical system modeling at the cost of increased difficulty in the integration of inductive biases.

## 2.5 Operator Learning

Operator learning, an area of ML recently receiving much attention, focuses on learning mappings between infinite-dimensional spaces, such as functions, operators, or differential equations, which is a significant departure from standard ML's focus on finite-dimensional vector spaces. Unlike traditional ML models that predict scalar or vector quantities, operator learning aims to predict entire functions or operators, offering powerful frameworks for tackling complex problems in physics, engineering, and beyond.

While standard ML techniques typically aim to predict specific outputs from given inputs, often treating the data as isolated points without considering the underlying physics, operator learning methods seek to learn the mapping between function spaces that describe the physical laws governing the system. This allows operator learning models to generalize across different scenarios and initial conditions, offering a principled way to incorporate physical knowledge directly into the model. Consequently, operator learning can provide more accurate and physically consistent predictions, especially for complex systems described by differential equations, at the cost of potentially higher computational complexity and the

need for careful design with specialized knowledge to construct and train these models effectively. Notable works in the field include the development of Deep Operator Networks (DeepONets) by Lu et al. [43] which learn operators mapping between function spaces, Fourier neural operators [132], and the aforementioned PINNs [46]. These pioneering studies highlight operator learning's potential to improve approaches to dynamical system and PDE modeling by learning operators which can greatly improve model generalization. Although operator learning methods are not directly utilized directly in this work, they are relevant to physical modeling with machine learning and may aid in understanding some of the results presented.

### 2.5.1 Deep Operator Netorks

The introduction of Deep Operator Networks (DeepONets) by Lu et. al [43] develops a deep learning framework designed to learn continuous nonlinear operators from data. DeepONets extend the universal approximation theorem to deep neural networks for operator approximation. It comprises two sub-networks: the branch network, which encodes input functions, and the trunk network, which encodes the domain of the output functions. This architecture allows DeepONet to efficiently learn mappings from input functions to output functions, applicable to both explicit operators (e.g., integrals, fractional Laplacians) and implicit operators (e.g., solutions to differential equations).

DeepONets aim to approximate an operator $G$ which takes a function $s$ as input. The output $G(s)(\mathbf{y})$ is then predicted at locations $\mathbf{y}$ using a deep learning-based approximation leveraging $p$ branch nets $b_k(s(\mathbf{x}_1), s(\mathbf{x}_2), \ldots, s(\mathbf{x}_m))$ and trunk nets $t_k(\mathbf{y})$, where $\mathbf{x}_1, \ldots, \mathbf{x}_m$ are a set of $m$ collocation points in the input space of function $s(\mathbf{x})$, and $\mathbf{y}$ is the desired output prediction locations. The branch and trunk nets are both constructed as deep neural networks with parameters trained by minimizing the difference between approximate operator predictions and data. The operator is approximated by

$$G(s)(\mathbf{y}) \approx \sum_{k=1}^{p} b_k(s(\mathbf{x}_1), s(\mathbf{x}_2), \ldots, s(\mathbf{x}_m)) t_k(\mathbf{y}) \tag{2.29}$$

such that predictions require two inputs: a set of function values $u(\mathbf{x})$ evaluated at various input locations and the output location $\mathbf{y}$. This particular form for the approximation of operator $G$ is informed by the universal approximation theorem for operators [43].

For example, given the dynamical system

$$\frac{\partial \mathbf{u}(\mathbf{x})}{\partial t} = F(\mathbf{u}(\mathbf{x}), s(\mathbf{x}), \mathbf{x}, t), \tag{2.30}$$

the operator $G$ maps the input function $s(\mathbf{x})$ to the solution of the dynamical system $\mathbf{u}(\mathbf{x}, t)$. A particular example [43] is given by learning the implicit operator which maps a source term $s(\mathbf{x})$ to the solution to a nonlinear diffusion-reaction PDE given by

$$\frac{\partial \mathbf{u}(\mathbf{x})}{\partial t} = D\frac{\partial^2 \mathbf{u}(\mathbf{x})}{\partial \mathbf{x}^2} + k\mathbf{u}(\mathbf{x})^2 + s(\mathbf{x}). \tag{2.31}$$

Key advantages of DeepONet include its ability to generalize well from limited data and its flexibility to handle various types of operators, including those representing complex physical systems. Compared to traditional methods, DeepONet offers efficient real-time predictions and can be trained with scattered data without requiring explicit forms of the operators.

The effectiveness of DeepONets hinge on accurately representing the input functions and judiciously selecting the network's structure, which may necessitate balancing complexity against the ability to generalize. It also requires extensive datasets, comprised of numerous system solutions under a wide range of input functions $s(\mathbf{x})$. Furthermore, incorporating inductive biases—prior knowledge to guide learning—presents challenges due to the distinct roles of the branch and trunk networks in generating predictions, complicating the integration of domain-specific insights into the model's architecture. This difficulty results in models which may not be physically interpretable and may not adhere to the defined physical principles in a particular application.

### 2.5.2 Fourier Neural Operators

The Fourier Neural Operator (FNO) for solving parametric partial differential equations (PDEs), introduced by Li et al. [132], offers a novel approach to learning mappings between infinite-dimensional function spaces. Unlike traditional neural networks which learn specific discretizations, FNO learns an operator mapping any functional parametric dependence to its solution, making it discretization-invariant and efficient. It leverages the Fourier transform for parameterizing the integral kernel in Fourier space, enabling significant computational speed-ups and accuracy improvements over existing methods.

One of the primary motivations for FNO is that it can be trained on data from one mesh, but predict on any other mesh. This is done by leveraging convolutions in the Fourier domain rather than the spatial domain, resulting in global convolutions rather than local ones. Li et al. introduce the Fourier layer, which consists of transforming an input to the Fourier domain, removing the high frequency modes, applying a linear transform on the low frequency modes, and finally performing the inverse Fourier transform. A bias term and a nonlinear activation function are then applied in the spatial domain. This Fourier layer is

specified mathematically by

$$\mathbf{v}_{t+1} = \sigma\left(\mathcal{F}^{-1}(\mathbf{R}_\phi \cdot (\mathcal{F}\mathbf{v}_t)) + \mathbf{W}\mathbf{v}_t\right) \ , \tag{2.32}$$

where $\mathbf{v}_t$ is the input to the later, $\mathcal{F}$ specifies the Fourier transform, $\mathbf{R}_\phi$ is a learnable linear transformation in Fourier space representing the Fourier transform of a continuous periodic function, $\mathbf{W}$ is a learnable bias term, $\sigma$ is an element-wise nonlinear activation function, and $\mathbf{v}_{t+1}$ is the layer output.

The method's efficacy is demonstrated in the original work through numerical experiments on Burgers' equation, Darcy Flow, and Navier-Stokes equations, highlighting its ability for zero-shot super-resolution and its application in solving Bayesian inverse problems. FNO provides impressive computational efficiency and accuracy for modeling complex PDEs. Its resolution-invariance means it can adapt to different discretizations without retraining. However, the method's reliance on extensive training data across varied parametric conditions can be a significant limitation, requiring considerable computational resources for data generation and training. Moreover, incorporating domain-specific knowledge or inductive biases into the FNO framework is challenging due to the transformation to Fourier space, potentially limiting its applicability in scenarios where such knowledge is critical for accurate modeling, particularly in true prediction past the training time in dynamical systems or where physical interpretability is paramount.

# CHAPTER 3

# Extracting Physical Parameters from Data with Variational Autoencoders

In computational physics and engineering, the advent of machine learning technologies has introduced transformative approaches to understanding complex systems. Among these, the application of VAEs stands out by offering the capability to distill meaningful insights from vast datasets, particularly when the underlying physical principles are not fully understood. This chapter explores the application and analysis of VAEs in extracting physical insights from datasets, especially emphasizing their utility for scenarios in which limited information on physical properties of the dataset is known.

The exploration of physical systems through computational models has traditionally hinged on the accuracy and comprehensiveness of the underlying physical laws. However, a number of limitations often render complete descriptions of the physical laws governing datasets elusive, necessitating the development of approaches capable of extracting meaningful physical insights from such datasets. VAEs, through their unsupervised learning paradigm, offer a promising avenue by learning to represent complex data distributions in lower-dimensional latent spaces, thereby uncovering hidden patterns and relationships that might not be immediately apparent.

The significance of extracting physical insights from datasets lies in the learned representation itself. If a learned representation is directly correlated to the underlying physical parameters governing the generation of samples in the dataset, then the learned representation can aid in uncovering physically-relevant structures in the data, leading to a better understanding of the governing physics. A better understanding of the governing physics can in turn aid in developing more accurate and robust models. In fields ranging from fluid dynamics to quantum mechanics, the ability to glean physical insights from data therefore represents an important step towards the development of more accurate and predictive models.

This work on unsupervised representation learning is motivated from a computational-

physics perspective. It is focused on the application of VAEs for use with data generated by partial differential equations (PDEs). The central questions it aims to answer are: a) can we reliably learn a physically-relevant representation from data obtained from PDEs governing physical problems using VAEs, and b) what are the characteristics of such representations? Successfully learning physically-relevant representations can ultimately lead to additional utility in many capacities: developing probabilistic reduced order models, design optimization, parameter extraction, and data interpolation, among others.

Through a study on the modeling of Darcy Flow, we illustrate the practical implementation of representation learning techniques with VAEs and their impact on enhancing our understanding of physical phenomena. The discussion extends to the challenges inherent in training VAEs on physical data, the solutions devised to address these issues, and the concept of incorporating physical learning biases to improve parameter discovery through semi-supervision.

By integrating VAEs into the analysis of physical systems, we leverage the power of unsupervised (and semi-supervised) learning to aid in analyzing data. This chapter underscores the objective of integrating machine learning with physical sciences, aiming to unlock use cases for the former in the pursuit of the latter. The following discussions draw material directly from Ref. [74].

## 3.1 Application to Physical Systems

Representation learning, especially within the framework of VAEs, is an important tool in computational physics and engineering. Its primary utility lies in its ability to automatically learn features and patterns directly from data, bypassing the need for manual feature extraction. Representation learning on its own is useful itself in a few ways:

- **Automatic feature extraction:** It simplifies the analysis process by automatically identifying significant features in large datasets, which might be non-obvious or difficult to extract manually.
- **Dimensionality reduction:** Representation learning condenses information into a more manageable form, facilitating easier manipulation and analysis while retaining the essence of the original data.
- **Transfer learning:** The representations learned from one task can often be reused for other tasks, especially if the tasks are related. This transfer learning capability is particularly valuable in scenarios where labeled data is scarce for the new task, as it

allows the use of pre-trained models that have learned useful representations from a different, data-rich task.

Unsupervised representation learning is also a popular area of research because of the need for low-dimensional representations in unlabeled data. Low-dimensional *latent* representations of high-dimensional data have many applications ranging from facial image generation [133] and music generation [134] to autonomous controls [135] among many others. Generative adversarial networks (GANs) [136], VAEs [72] and their variants [84, 85, 86, 87], among other methods such as diffusion models [100] and score-based generative models [99] (Chapter 6), aim to approximate an underlying distribution $p(y)$ of high-dimensional data by leveraging only samples from that distribution. In the case of VAEs, compressed representations $z$ are sampled from a low-dimensional - yet unknown - distribution $p(z)$. A mapping from the high-dimensional space to a low-dimensional space and the corresponding inverse mapping are learned simultaneously, allowing approximations of both $p(y)$ and $p(z)$. The learned lower dimensional representation or *latent representation* corresponds to the primary goal of representation learning. However, this work focuses on exactly how the latent representation represents the data.

In particular, if learned representations are correlated to physically relevant parameters governing data generation, a host of new use cases are unlocked. The utility of successfully extracting these representations lies primarily in their potential to unveil the hidden physical parameters that govern the generation of data. Especially in cases where the physical laws governing data are not entirely well understood or defined, learning a data representation which is correlated to physically relevant parameters may be useful for:

- **Interpretability:** Successfully mapping data to physically-relevant generative parameters enhances the interpretability of the dataset. It enables understanding of the impact of varying specific parameters on the system's behavior, providing deep insights into the dynamics and mechanics of complex systems.

- **Efficient Experimentation:** Having a model that encodes data to its true generative parameters allows for the systematic alteration of these parameters to observe their effects. This capacity can guide experimental designs, enabling a more focused exploration of physical phenomena and the optimization of conditions for desired outcomes.

- **Data Labelling:** Data samples can be effectively labeled with the latent representation. If the representation corresponds to physically-relevant generative parameters, each data sample is labeled and can then be used for other tasks.

- **Synthetic Data Generation:** The manipulation of identified physically-relevant parameters facilitates the generation of new, synthetic data samples with labels included. This ability is crucial for augmenting datasets, especially in situations where collecting real-world data is challenging or expensive, thereby enhancing the robustness of machine learning models.

As an illustrative example in the context of climate science, representation learning can be leveraged to analyze vast datasets of atmospheric conditions. For instance, a VAE might learn representations correlating with key climatic factors like temperature, humidity, or CO2 levels, even if these relationships are not explicitly labeled in the data. This capability is invaluable for identifying patterns and predicting climate trends, aiding in the development of more accurate and comprehensive climate models. By learning these representations, researchers can gain insights into the complex dynamics of climate systems, facilitating a deeper understanding of weather patterns, seasonal variations, and long-term climate change effects. Without such a physically-relevant representation, the underlying causes of changes in climate are difficult to interpret and therefore difficult to diagnose.

The utility of representation learning in physics extends far beyond climate modeling. It encompasses a broad range of applications from understanding the fundamental properties of materials at the atomic level to understanding dark matter structures in the universe. In this work, the example employed for an in-depth investigation of physically-relevant representation learning is that of Darcy flow, primarily describing the flow of a fluid through a porous medium.

## 3.2 Darcy Flow

The Darcy flow equations are fundamental in fluid mechanics, describing the flow of fluids through porous media [137]. They are vital in various fields, particularly in groundwater hydrology and petroleum engineering, but the equations defining Darcy flow are also relevant in modeling the electric potential of conductive materials according to Ohm's law [138]. These equations enable the calculation of fluid flow based on the properties of the fluid and the medium, helping to predict how water or oil moves through subsurface environments. This understanding is crucial for water resource management, environmental engineering, and the extraction of oil and natural gas. The impact of the Darcy flow equations is significant, as they inform critical decisions in environmental protection [139], agricultural planning, and energy production [140] worldwide.

Darcy flow is used as an illustrative example in this chapter as well as Chapter 6, modeled by the Darcy flow equations or Darcy's law. A permeability field $K(\mathbf{x})$ defined on a physical

domain $\mathcal{X}$ describes the degree to which a fluid can pass through the media at each location $\mathbf{x}$ in the domain while a source function $f_s(\mathbf{x})$ models a source or sink of fluid entering or exiting the system at spatial locations. The pressure $p(\mathbf{x})$ and velocity fields $\mathbf{u}(\mathbf{x})$ of the fluid then satisfy the following equations according to Darcy's law:

$$\begin{aligned} \mathbf{u}(\mathbf{x}) &= -K(\mathbf{x})\nabla p(\mathbf{x}), \quad \mathbf{x} \in \mathcal{X} \\ \nabla \cdot \mathbf{u}(\mathbf{x}) &= f_s(\mathbf{x}), \quad \mathbf{x} \in \mathcal{X} \\ \mathbf{u}(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x}) &= 0, \quad \mathbf{x} \in \partial\mathcal{X} \\ \int_{\mathcal{X}} p(\mathbf{x}) d\mathbf{x} &= 0. \end{aligned} \tag{3.1}$$

Data samples of Darcy flow fields are generated by setting a constant source function given by Eq. 3.2 and sampling a random permeability field $K(\mathbf{x})$. The source function models an injection well and extraction well in opposite corners of the domain, while the permeability field is generated randomly according to a specified distribution.

$$f_s(\mathbf{x}) = \begin{cases} r, & |x_i - \frac{1}{2}w| \leq \frac{1}{2}w, \;\; i = 1,2 \\ -r, & |x_i - 1 + \frac{1}{2}w| \leq \frac{1}{2}w, \;\; i = 1,2 \\ 0, & \text{otherwise} \end{cases}, \tag{3.2}$$

We sample $K(\mathbf{x})$ by modeling the log-permeability field as a Gaussian random field with covariance function $k$

$$K(\mathbf{x}) = \exp(G(\mathbf{x})), \;\; G(\cdot) \sim \mathcal{N}(\bar{\mu}, k(\cdot,\cdot)) \; . \tag{3.3}$$

The covariance function is defined as the exponential kernel by

$$k(\mathbf{x}, \mathbf{x}') = \exp(-||\mathbf{x} - \mathbf{x}'||_2 / l) \tag{3.4}$$

in our experiments, as in Ref. [141].

For dimensionality reduction and to create a parametric representation of the data, the intrinsic dimensionality $s$ of the data is specified by leveraging the Karhunen-Loeve Expansion (KLE), retaining only the first $s$ terms in

$$G(\mathbf{x}) = \bar{\mu} + \sum_{i=1}^{s} \sqrt{\lambda_i} \theta_i \phi_i(\mathbf{x}) \; , \tag{3.5}$$

where $\lambda_i$ and $\phi_i(\mathbf{x})$ are eigenvalues and eigenfunctions of the covariance function (Eq. 3.4)

sorted by decreasing $\lambda_i$, and each $\theta_i$ are sampled according to some distribution $p(\boldsymbol{\theta})$, denoted the *generative parameter distribution*. These generative parameters $\boldsymbol{\theta}$ entirely determine the solution to the governing equations. In other words, the solution is a deterministic function of the generative parameters, and so $p(\boldsymbol{\theta})$ entirely determines the dataset distribution.

### 3.2.1 Discretized Finite Difference Solution

After sampling the permeability field, Eq 3.1 is solved for the pressure fields. Note that once the pressure field is obtained, the velocity fields can be easily approximated using finite difference approximations using the relationship $\mathbf{u}(\mathbf{x}) = -K(\mathbf{x})\nabla p(\mathbf{x})$. We discretize the spatial domain $\mathcal{X} = [0,1]^2$ on an $n \times n$ grid to form a computational grid of $n^2$ nodes. This leads to discrete steps of $\Delta x_1 = \Delta x_2 = 1/(n-1)$ in the spatial coordinate system. Each node in the computational domain is labeled with two indices $i, j$ corresponding to the spatial location of the node such that $\mathbf{x}_{i,j} = [(i-1)/(n-1), (j-1)/(n-1)]^T$. A linear system $\mathbf{Ap} = \mathbf{f}$ is then formed and solved for pressure on the computational domain. To create this linear system, we define the solution $p_{i,j}$ at each point in the discretized spatial domain. The Darcy flow equations for pressure only are given by

$$-\nabla \cdot [K(\mathbf{x})\nabla p(\mathbf{x})] = f_s(\mathbf{x}) \tag{3.6}$$

$$\nabla p(\mathbf{x}) \cdot \hat{\mathbf{n}}(\mathbf{x}) = 0 \tag{3.7}$$

$$\int_{\mathcal{X}} p(\mathbf{x}) d\mathbf{x} = 0 \; . \tag{3.8}$$

Equation 3.6 can be expanded using chain rule to be

$$-K(\mathbf{x})\frac{\partial^2 p(\mathbf{x})}{\partial x_1^2} - \frac{\partial K(\mathbf{x})}{\partial x_1}\frac{\partial p(\mathbf{x})}{\partial x_1} - K(\mathbf{x})\frac{\partial^2 p(\mathbf{x})}{\partial x_2^2} - \frac{\partial K(\mathbf{x})}{\partial x_2}\frac{\partial p(\mathbf{x})}{\partial x_2} = f_s(\mathbf{x}) \tag{3.9}$$

On the discretized domain, second order central finite differences are utilized on interior points to compute first and second order partial derivatives. For example,

$$\left.\frac{\partial p(\mathbf{x})}{\partial x_1}\right|_{\mathbf{x}_{i,j}} \approx \frac{p_{i+1,j} - p_{i-1,j}}{2\Delta x_1} \quad \text{and} \quad \left.\frac{\partial^2 p(\mathbf{x})}{\partial x_1^2}\right|_{\mathbf{x}_{i,j}} \approx \frac{p_{i-1,j} - 2p_{i,j} + p_{i+1,j}}{\Delta x_1^2} \; .$$

The pressure gradient across the boundaries is zero according to Eq. 3.7. Thus on the left ($x_1 = 0$) boundary for example,

$$\left.\frac{\partial p(\mathbf{x})}{\partial x_1}\right|_{\mathbf{x}_{i,j}} \approx \frac{p_{i+1,j} - p_{i-1,j}}{2\Delta x_1} = 0 \; ,$$

and thus $p_{i+1,j} = p_{i-1,j}$. The second order partial derivative on the left ($i = 1$) boundary is therefore given by

$$\left.\frac{\partial^2 p(\mathbf{x})}{\partial x_1^2}\right|_{\mathbf{x}_{1,j}} \approx \frac{2p_{2,j} - 2p_{1,j}}{\Delta x_1^2} \ .$$

Additionally, the integral constraint in Eq. 3.8 can be enforced by adding another row to the matrix $\mathbf{A}$, creating an over-determined system of equations. We thus solve the system $\mathbf{Ap} = \mathbf{f}$ where $\mathbf{A} \in \mathbb{R}^{(n^2+1)\times n^2}$, $\mathbf{p} \in \mathbb{R}^{n^2}$, and $\mathbf{f} \in \mathbb{R}^{n^2+1}$. The vectors $\mathbf{p}$ and $\mathbf{f}$ are given by Eq. 3.10.

$$\mathbf{p} = \begin{bmatrix} p(\mathbf{x}_{1,1}) \\ p(\mathbf{x}_{2,1}) \\ \vdots \\ p(\mathbf{x}_{n,1}) \\ p(\mathbf{x}_{1,2}) \\ \vdots \\ p(\mathbf{x}_{n,n}) \end{bmatrix}, \ \mathbf{f}_s = \begin{bmatrix} f_s(\mathbf{x}_{1,1}) \\ f_s(\mathbf{x}_{2,1}) \\ \vdots \\ f_s(\mathbf{x}_{n,1}) \\ f_s(\mathbf{x}_{1,2}) \\ \vdots \\ f_s(\mathbf{x}_{n,n}) \\ 0 \end{bmatrix}. \tag{3.10}$$

The matrix $\mathbf{A}$ is constructed using the finite difference formulas previously described. The exact form and a description of the discretized equations which the linear system solves are included in Appendix A.1. The velocity components $\mathbf{u}(\mathbf{x})$ are computed using second order finite difference approximations on $\mathbf{p}$ to compute $\mathbf{u}(\mathbf{x}) = -K(\mathbf{x})\nabla p(\mathbf{x})$.

### 3.2.2 Datasets

Each dataset is characterized by an intrinsic dimensionality $s$, and is labeled accordingly. Example samples from datasets of various intrinsic dimension are illustrated in Fig. 3.1. Variations on the $s = 2$ dataset are employed for the explorations present in the work of this Chapter for ease of visualization in the generative parameter space, but datasets with greater intrinsic dimensionality are employed in Chapter 6. The concepts in this Chapter are primarily explored by varying the generative parameter distribution $p(\boldsymbol{\theta})$ in each dataset.

Each snapshot existing in a dataset contains the generative parameters $\boldsymbol{\theta}$ along with the permeability field $K(\mathbf{x})$, the pressure $p(\mathbf{x})$ field, and the velocity fields $\mathbf{u}(\mathbf{x})$ at each node in the computational domain. The pressure and velocity fields are concatenated into a 3-channel input to the VAE; the permeability field and generative parameters are saved, but not used in unsupervised training. The generative parameters and employed in semi-supervised training, while the permeability fields are used for evaluation purposes only.

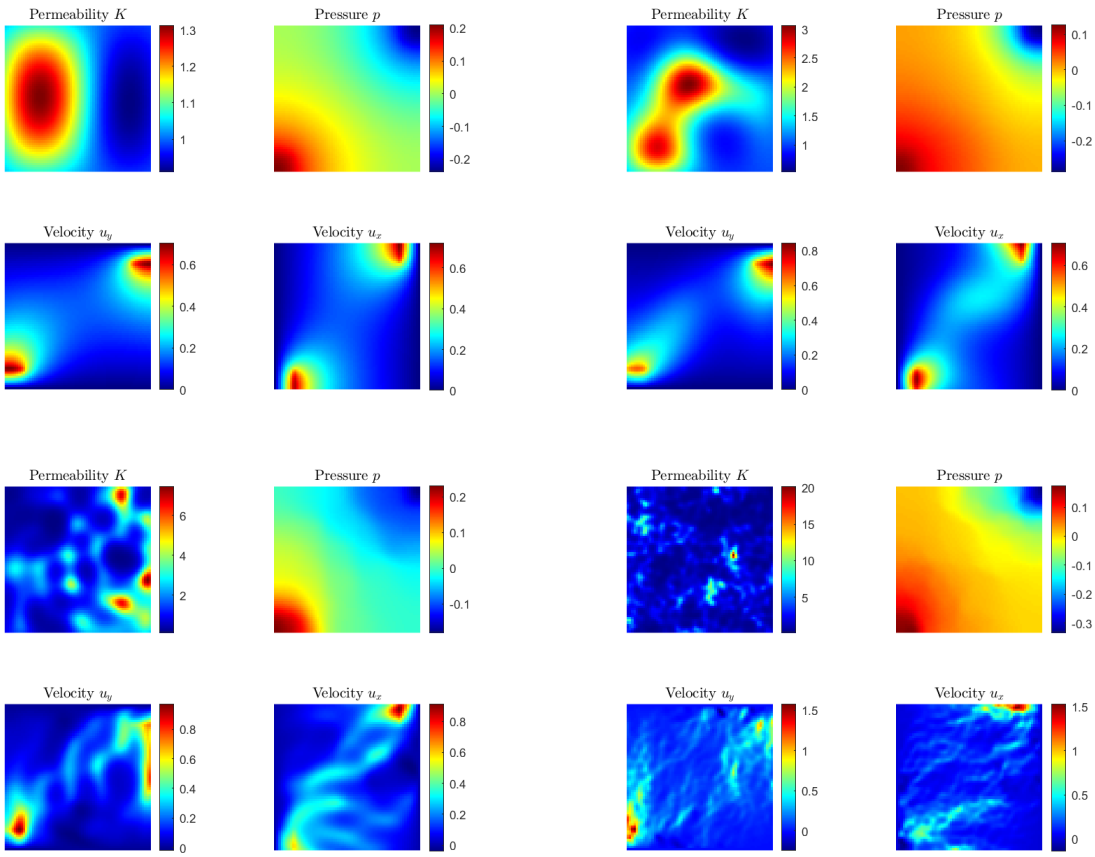



Figure 3.1: Samples from datasets (*top left*) $s = 2$ (*top right*) $s = 10$ (*bottom left*) $s = 100$ (*bottom right*) $s = 1000$.

## 3.3 Disentanglement and Hierarchical Priors

Predicting the encoding distribution $q_\phi(\mathbf{z}|\mathbf{y})$ results in a lower dimensional latent distribution given data sample $\mathbf{y}$, with $\mathbf{y}$ being a function of generative parameters $\boldsymbol{\theta}$. Investigating the exact relationship between $\mathbf{z}$, $\mathbf{y}$, and $\boldsymbol{\theta}$ is of primary importance in this work. Central to understanding these relationships are the concepts of disentanglement and hierarchical priors.

### 3.3.1 Disentanglement

*Disentanglement*, in the context of this work, is realized when variations in a single latent dimension $z_i$ are directly correlated to variations in a single generative parameter $\theta_j$. In other words, the relationship $\theta_j = \alpha z_i$ for some constant $\alpha$ is satisfied for each of the generative parameters. Throughout this chapter, this is frequently referred to as 'disentanglement'

or 'disentangling the generative factors'. It allows the latent space to be interpretable by the user and improves transferability of representations between tasks. Additionally, in Section 3.6, an even more powerful relationship is learning such that the latent mean is shown to be nearly equivalent to the generative parameters, i.e. $\theta_j = z_i$, with the enforcement of a few true labels. With just a few labeled samples, the rest of the dataset can effectively be labeled by the trained VAE.

Disentanglement may not be required for some tasks which may not require knowledge on each parameter individually or perhaps only a subset of the generative parameters. Nevertheless, a disentangled representation can be leveraged across many tasks. Bengio et al. [88] note that a disentangled representation captures each of the relevant features of the data, but downstream applications may only require a subset of these factors. We therefore hypothesize that disentangled representations lead to a more comprehensive range of downstream applications over non disentangled representations.

Enforcing disentanglement using VAEs was first addressed in the literature [84, 86] by modifying the strength of regularization in the ELBO loss, with the penalty of sub-optimal compression and reconstruction. FactorVAEs [84] encourage a factorized representation, which can be useful for disentanglement in the case of independent generative parameters, but undesirable when parameters are correlated. Rolinek et al [97] suggest that the ability of the VAE to learn disentangled representations is not inherent to the framework itself, but an "accidental" byproduct of the typically assumed factorized form of the encoder. The prior distribution is of particular importance as the standard normal prior often assumed allows for rotation of the latent space with no effect on the ELBO loss. Disentangled representations are still often learned due to a factorized form of the encoding distribution with sufficiently large weight on regularization. Additional interpretations and insight into the disentanglement ability of VAEs are found in Ref. [142].

To illustrate the idea of disentanglement and its implications, consider a dataset consisting of images of teapots [143]. Each image is generated from 3 parameters indicating the color of the teapot (RGB) and 2 parameters corresponding to the angle the teapot is viewed from. Thus, even though the RGB image may be very high dimensional, the intrinsic dimensionality is just 5. Representation learning can be used to extract a low-dimensional latent model containing useful and meaningful representations of the high-dimensional images. Learned latent representations need not be disentangled to be useful in some sense, but disentanglement enhances interpretability of the representation. Disentanglement references a structure of the latent distribution in which changes in each parameter in the learned representation are correlated directly to changes in a single yet different generative parameter. Humans tend to naturally and easily identify independent factors of variation, and thus a

disentangled representation often corresponds to one which would be naturally identified by a human. A representation which is more naturally explained by a human observer is therefore one characterized by greater interpretability. In an unsupervised setting, however, our work indicates that one cannot guarantee that a disentangled representation will be learned. On the other hand, introducing a small amount of physically-relevant inductive bias in the form of data labels for just a few samples can greatly improve the robustness of learning such representations.

The requirement for disentanglement depends on the task at hand, but a disentangled representation may be used in many tasks containing different objectives. Indeed, Bengio et. al [88] state that 'the most robust approach to feature learning is to disentangle as many factors as possible, discarding as little information about the data as is practical'. In the teapot example, changes in one of the learned latent dimensions may correspond to changes in the color red and one of the viewing angles, which would indicate an entangled representation. Another example, more relevant to this work, is that of fluid flow over an airfoil. Learning a disentangled representation of the flow conditions along with the shape parameters using VAEs can allow rapid prediction of the flow field with enhanced interpretability of the latent representation, facilitating efficient computation of the task at hand. The disentangled representation can be transferred to a variety of tasks easily such as design optimization, developing reduced order models in the latent space or parameter inference from flow fields. It is the ability of disentangled representations which are correlated to physically relevant factors to transfer across tasks with ease and interpretability which makes them so useful. In many practical physics problems, full knowledge regarding the underlying generative parameters of high-dimensional data may not exist, thus making it challenging to ascertain the quality of the representation learned.

Many metrics of disentanglement exist in the literature [144], few of which take into account the generative parameter data. Often knowledge on the generative parameters is lacking, and these metrics can be used to evaluate disentanglement in that case (although there seems to be no consensus on which metric is appropriate). In controlled experiments, however, knowledge on generative parameters is available, and correlation between the latent space and the physically-relevant generative parameter space can be directly determined. To evaluate disentanglement in a computationally efficient manner, we propose a disentanglement score given by

$$S_D = \frac{1}{n} \sum_i \frac{\max_j \ |\text{cov}(z_i, \theta_j)|}{\sum_j |\text{cov}(z_i, \theta_j)|} \ , \tag{3.11}$$

where $z_i$ indicates the $i^{th}$ component of the latent vector $\forall i \in \{1, \ldots, n\}$ and $\theta_j$ indicates

the $j^{th}$ component of the generative parameter vector $\forall j \in \{1, \ldots, s\}$. Noting that

$$\frac{\max_j |\text{cov}(z_i, \theta_j)|}{\sum_j |\text{cov}(z_i, \theta_j)|} \in [1/s, 1] \ ,$$

it is clear that $S_D \in [1/s, 1]$. It is noted that this score is not used during the training process. This score is created from the intuition that each latent parameter should be correlated to only a single generative parameter. Although this score is computationally efficient to compute and works well in our experiments, issues with the score exist in certain scenarios. For instance, if multiple latent dimensions are correlated to the same generative parameter dimension, the score will be inaccurate. Similarly, if the latent dimension is greater than the generative parameter dimension, some latent dimensions may contain no information about the data and be uncorrelated to all dimensions, inaccurately reducing the score. For the cases presented here (we will use the score only when $n = s$), Eq. 3.11 suffices as a reasonable measure of disentanglement. This score is used as an efficient means of scoring disentanglement when efficiency is important, but we propose another score based on comparisons between disentangled and entangled representations.

We observe empirically that disentanglement is highly correlated to a match in 'shape' between the generative parameter distribution $p(\theta)$ and the aggregated posterior $q_\phi(z)$ (Section 3.5). A match in the scaled-and-translated shapes results in good disentanglement but an aggregated posterior which does not match the shape of the generative parameter distribution or contains incorrect correlations ('rotated') relative to the generative parameter distribution does not. Using this knowledge, another disentanglement metric is postulated to compare these shapes by leveraging the KL Divergence (Eq. (3.12)) where $\circ$ denotes the Hadamard product. The disentanglement score is given by

$$S_{KL} = \min_{a,b} D_{KL}[p(\boldsymbol{\theta})||q_\phi(a \circ (\mathbf{z} - b))] \ . \tag{3.12}$$

This metric compares the shapes of the two distributions by finding the minimum KL divergence between the generative parameter distribution and a scaled and translated version of the aggregated posterior. When $q_\phi(a \circ (\mathbf{z} - b))$ is close to $p(\boldsymbol{\theta})$ for some vectors $a, b \in \mathbb{R}^n$, disentanglement is observed.

It is noted by Rolinek et. al [97] that rotation of the latent space relative to the generative parameter distribution has a significant negative effect on disentanglement, which is precisely what is observed in this work. Additionally, the ELBO loss is unaffected by rotations of the latent space when using rotationally-invariant priors such as the standard normal (Appendix A.2).

### 3.3.2 Hierarchical Priors

Often the prior (in the case of classic VAEs, specified by the user) and generative parameter distributions (data dependent) may not be correlated. Hierarchical priors [85] (HP) can be implemented within the VAE network such that the prior is learned as a function of additional random variables, potentially leading to more expressive priors and aggregated posteriors. Hierarchical random variables $\boldsymbol{\xi}_i$ are introduced such that 'sub-priors' can be assumed on each $\boldsymbol{\xi}_i$ (typically standard normal). In the case of a single hierarchical random variable

$$p(\mathbf{z}) = \int_{\Xi} p(\mathbf{z}|\boldsymbol{\xi})p(\boldsymbol{\xi})d\boldsymbol{\xi} = \int_{\Xi} \frac{p(\boldsymbol{\xi}|\mathbf{z})}{p(\boldsymbol{\xi}|\mathbf{z})} p(\mathbf{z}|\boldsymbol{\xi})p(\boldsymbol{\xi})d\boldsymbol{\xi} = \mathbb{E}_{\Xi\sim p(\boldsymbol{\xi}|\mathbf{z})}\left[\frac{p(\mathbf{z}|\boldsymbol{\xi})p(\boldsymbol{\xi})}{p(\boldsymbol{\xi}|\mathbf{z})}\right] .$$

The conditional distributions $p(\boldsymbol{\xi}|\mathbf{z})$ and $p(\mathbf{z}|\boldsymbol{\xi})$ are the *prior encoder* and *prior decoder*, respectively. These distributions can be approximated by parameterizing them with neural networks. The parameterized distributions are noted as $q_\gamma(\boldsymbol{\xi}|\mathbf{z})$ and $p_\pi(\mathbf{z}|\boldsymbol{\xi})$ where $\gamma$ are the trainable parameters of the approximating prior encoder and $\pi$ are the trainable parameters of the prior decoder. Thus, the VAE prior can be approximated through the prior encoding and decoding distributions

$$p(\mathbf{z}) \approx \mathbb{E}_{\Xi\sim q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\left[\frac{p_\pi(\mathbf{z}|\boldsymbol{\xi})p(\boldsymbol{\xi})}{q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\right]. \tag{3.13}$$

Rearranging the VAE regularization loss

$$\begin{aligned}\mathcal{L}_{REG} =& \int_{Y,Z} \hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y}) \log \frac{q_\phi(\mathbf{z}|\mathbf{y})}{p(\mathbf{z})} d\mathbf{y}d\mathbf{z} \\ =& \mathbb{E}_{Y,Z\sim\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\log q_\phi(\mathbf{z}|\mathbf{y})] - \int_{Y,Z} \hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y}) \log p(\mathbf{z}) d\mathbf{y}d\mathbf{z} ,\end{aligned} \tag{3.14}$$

and substituting the approximating hierarchical prior Eq. (3.13) into Eq. (3.14), the final term on the right-hand side becomes

$$-\int_{Y,Z} \hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y}) \log p(\mathbf{z}) d\mathbf{y}d\mathbf{z} = -\int_{Y,Z} \hat{p}(\mathbf{y}) \quad q_\phi(\mathbf{z}|\mathbf{y}) \log \left[\mathbb{E}_{\Xi\sim q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\left[\frac{p_\pi(\mathbf{z}|\boldsymbol{\xi})p(\boldsymbol{\xi})}{q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\right]\right] d\mathbf{y}d\mathbf{z} .$$

The logarithm function is strictly concave; therefore, by Jensen's inequality the right-hand

side is upper bounded by

$$-\int_{Y,Z}\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})\log\left[\mathbb{E}_{\Xi\sim q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\left[\frac{p_\pi(\mathbf{z}|\boldsymbol{\xi})p(\boldsymbol{\xi})}{q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\right]\right]d\mathbf{y}d\mathbf{z}\leq$$
$$-\int_{Y,Z}\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})\mathbb{E}_{\Xi\sim q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\left[\log\frac{p_\pi(\mathbf{z}|\boldsymbol{\xi})p(\boldsymbol{\xi})}{q_\gamma(\boldsymbol{\xi}|\mathbf{z})}\right]d\mathbf{y}d\mathbf{z}\ .$$

This bound is rearranged to the form

$$\mathbb{E}_{Y,Z\sim\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[D_{KL}[q_\gamma(\boldsymbol{\xi}|\mathbf{z})||p(\boldsymbol{\xi})]-\mathbb{E}_{Y,Z\sim\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\mathbb{E}_{q_\gamma(\boldsymbol{\xi}|\mathbf{z})}[\log p_\pi(\mathbf{z}|\boldsymbol{\xi})]]\ . \tag{3.15}$$

Equation 3.15 takes the same form as the overall VAE loss, but applied to the prior network itself. Thus, the hierarchical prior can be thought of as a system of sub-VAEs within the main VAE. In summary, the VAE loss is upper bounded by

$$\mathcal{L}_{VAE}\leq\mathbb{E}_{Y,Z\sim\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\log q_\phi(\mathbf{z}|\mathbf{y})]+\mathbb{E}_{Y,Z\sim\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[D_{KL}[q_\gamma(\boldsymbol{\xi}|\mathbf{z})||p(\boldsymbol{\xi})]] \tag{3.16}$$
$$-\mathbb{E}_{Y,Z\sim\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\mathbb{E}_{q_\gamma(\boldsymbol{\xi}|\mathbf{z})}[\log p_\pi(\mathbf{z}|\boldsymbol{\xi})]] \tag{3.17}$$
$$-\mathbb{E}_{Y,Z\sim\hat{p}(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\log p_\psi(\mathbf{y}|\mathbf{z})]]\ . \tag{3.18}$$

Implementing hierarchical priors can aid in learning non-rotationally-invariant priors, frequently inducing a learned disentangled representation.

## 3.4 Training Challenges and Solutions

The process of training a VAE involves a number of challenges. For example, convergence of the optimizer to local minima can greatly hinder reconstruction accuracy and failure to converge altogether remains a possibility. A recurrent issue with VAE training in our experiments is that of over-regularization. Over-regularized solutions are characterized by disproportionately small regularization loss ($\mathcal{L}_{REG}\approx 0$). More information on this issue is detailed in Sec. 3.4.2.

To mitigate some of the issues inherent to training VAEs, we employ a training method tailored towards avoiding over-regularization. All experiments are performed using the Adam optimizer in Pytorch using $\mathcal{L}_{\beta-VAE}$ to train the models. The loss function $\beta$ value is varied accross the training epochs, but at the end of training the model is converged with $\beta=1$. The model is trained initially with $\beta_0\ll 1$, typically around $\beta_0=10^{-7}$, for some number of epochs $r_0$ (depending on learning rates) until reconstruction accuracy is well below that of an over-regularized solution (Section 3.3 illustrates this necessity). When $\beta_0$ is too small, the

regularization loss can become too large, preventing convergence altogether. Training is continued by implementing a $\beta$ scheduler [85] to slowly increase the weight of the regularization loss. The learning rate is then decreased to $lr_1 = c(lr_0)$ after some number of epochs $r_1$ to enhance reconstruction accuracy. This training method—in particular the heavily weighted reconstruction phase and the $\beta$ scheduler—result in much more stable training which avoids the local minima characterized by over-regularization and improves convergence consistency. Similar methods have been employed to avoid this issue. In particular, [145] refers to this issue as "KL vanishing" and uses a cyclical $\beta$ schedule to avoid the issue. However, this can take far more training epochs and cycle iterations to converge than the method introduced in this work.

### 3.4.1 Architecture

The primary architecture for the VAE is adapted from Ref. [141] and a more detailed description including architecture optimization is given in Appendix A.3. This architecture consists of a series of encoding blocks to form the encoder, and a series of decoding blocks to form the decoder. Each encoding / decoding block consists of a dense block followed by and encoding / decoding layer. Contrary to the name, dense blocks do not contain any dense layers, but rather a series of skip connections and convolutional layers. Encoding and decoding layers consist of convolutions. The architecture is called DenseVAE and is used for all VAEs trained in this work. The latent and output distributions are assumed to be Gaussian. We use the dense block based architecture to parameterize the encoder mean and log-variance separately, as well as the decoder mean. The decoding distribution log-variance is learned but constant as introducing a learned output log-variance does not aid in reconstruction or improving disentanglement properties in our experiments but increases training time.

### 3.4.2 Over-Regularization

Over-regularization has been identified as a challenge in the training of VAEs [145]. This phenomenon is characterized by the latent space containing no information about the data; i.e. the regularization loss becomes zero or nearly zero. The output of the decoder becomes identical accross all inputs. Thus, the output of the decoder is a constant distribution which does not depend on the latent representation. The constant distribution it learns becomes a normal distribution with mean and variance of the data. With zero regularization loss, the learned decoding distribution becomes $p_\psi(y|z) = \mathcal{N}(y; \hat{\mu}_y, \text{diag}(\hat{\sigma}_y^2)) \ \forall \ z$ where $\hat{\mu}_y = \frac{1}{N}\sum_{i=1}^{N} y^{(i)}$ and $\hat{\sigma}_y^2 = \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - \hat{\mu}_y)^2$. This is proven to minimize $\mathcal{L}_{VAE}$ in Theorem 1. The solution is not shown to be unique, but our experiments indicate that this is the over-

regularized solution found during training. As Theorem 1 illustrates validity for any encoder $q(z|y)$, this is the most robust solution for the VAE to converge to when over-regularization occurs. The decoder learns to predict as accurately as possible given nearly zero mutual information between the latent and data random variables. As the encoder and decoder are trained simultaneously, predicting a constant output regardless of $z$ prevents the necessity of the decoder to adjust as the encoder changes. An empirical comparison between good reconstruction and over-regularization is shown in Figure 3.2.

Theorem 1 requires that the output variance is constant. Parameterizing the output variance with an additional network may aid in avoiding over-regularization.

**Theorem 1:** Given data $\{y^{(i)}\}_{i=1}^N$ and the VAE framework defined in Section 2.3.1, and assuming a decoding distribution of the form $p(y|z) = \mathcal{N}(y; \mu(z), \text{diag}(\sigma^2))$, if $\mathcal{L}_{REG} = 0$, then $\text{argmin}_{\mu(z),\sigma^2} \quad \mathcal{L}_{VAE} = \{\hat{\mu}_y, \hat{\sigma}_y^2\}$ , where $\hat{\mu}_y = \frac{1}{N}\sum_{i=1}^N y^{(i)}$ and $\hat{\sigma}_y^2 = \frac{1}{N}\sum_{i=1}^N (y^{(i)} - \hat{\mu}_y)^2$.

**Proof:** For any $q(z|y)$ s.t. $\mathcal{L}_{REG} = 0$:

$$\begin{aligned}\mathcal{L}_{VAE} = \mathcal{L}_{REC} &= \mathbb{E}_{\hat{p}(y)q(z|y)}[-\log(p(y|z))] \\ &= \mathbb{E}_{q(z|y)}\left[\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{m}\frac{1}{2}\log(2\pi) + \log(\sigma_j) + \frac{1}{2\sigma_j^2}(y_j^{(i)} - \mu_j(z))^2\right].\end{aligned}$$

To minimize$\mathcal{L}_{VAE}$, take derivatives $\frac{\partial \mathcal{L}_{VAE}}{\partial \mu_j(z)}$ and $\frac{\partial \mathcal{L}_{VAE}}{\partial \sigma_j}$ (assuming derivative and expectation can be interchanged), where $j \in \{1, \ldots, m\}$:

$$\frac{\partial \mathcal{L}_{VAE}}{\partial \mu_j(z)} = \mathbb{E}_{q(z|y)}\left[-\frac{1}{N}\sum_{i=1}^{N}\frac{1}{\sigma_j^2}(y_j^{(i)} - \mu_j(z))\right] = 0.$$

Thus, $\mathbb{E}_{q(z|y)}\left[\sum_{i=1}^N y_j^{(i)} - \mu_j(z)\right] = 0$ and

$$\mathbb{E}_{q(z|y)}[\mu_j(z)] = \frac{1}{N}\sum_{i=1}^{N} y_j^{(i)}. \tag{3.19}$$

Eq. (3.19) holds $\forall z, j$ if

$$\mu_j(z) = \hat{\mu}_j = \frac{1}{N}\sum_{i=1}^{N} y_j^{(i)}. \tag{3.20}$$

Taking the derivative w.r.t. variance, we have $\frac{\partial \mathcal{L}_{VAE}}{\partial \sigma_j} =$

$\mathbb{E}_{q(z|y)}\left[\frac{1}{N}\sum_{i=1}^{N}\frac{1}{\sigma_j}-\frac{1}{\sigma_j^3}(y^{(i)}-\mu_j(z))^2\right]=0$, and rearranging, we have

$$\sigma_j^2=\frac{1}{N}\sum_{i=1}^{N}(y_j^{(i)}-\mu_j(z))^2\ \ \forall\ \ z,j. \tag{3.21}$$

Substituting Eq. (3.19) into Eq. (3.21) results in:

$$\hat{\sigma}_j^2=\frac{1}{N}\sum_{i=1}^{N}(y_j^{(i)}-\hat{\mu}_j)^2\ \forall\ z,\ j. \tag{3.22}$$

With Eqs. (3.19) and (3.22) valid for all $z$ and $j$, we can combine them into vector form and note that Eq. (3.23) minimizes $\mathcal{L}_{VAE}$ as required.

$$\hat{\mu}_y=\begin{bmatrix}\hat{\mu}_1\\ \vdots\\ \hat{\mu}_m\end{bmatrix},\quad \hat{\sigma}_y^2=\begin{bmatrix}\hat{\sigma}_1^2\\ \vdots\\ \hat{\sigma}_m^2\end{bmatrix}. \tag{3.23}$$

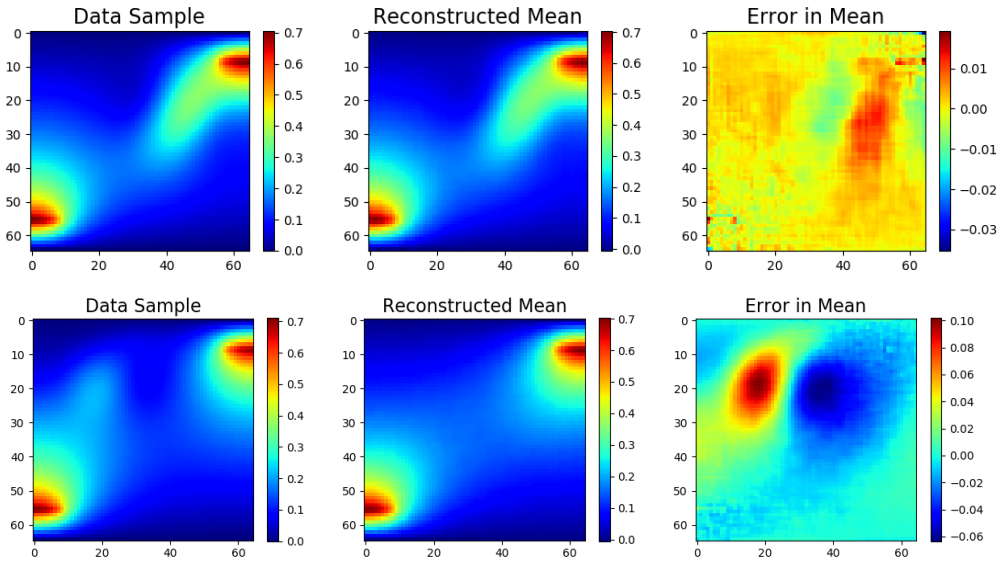



Figure 3.2: (*upper*) Good reconstruction. (*lower*) Over-regularization.

There exists a region in the trainable parameter loss landscape characterized by over-regularized local minimum solutions which partially surrounds the 'desirable' solutions characterized by better reconstruction accuracy and latent properties. This local minima region is often avoided by employing the training method discussed previously, but random initialization of network parameters and changes in hyperparameters between training can render

it difficult to avoid convergence to this region.

We illustrate the problem of over-regularization by training VAEs using the architecture described in Section 3.4.1 on the $s = 2$ Darcy flow dataset with $p(\theta)$ being standard normal. A VAE is trained with 512 training samples (each sample is 65×65×3), converging to a desirable solution with low reconstruction error and nearly perfect disentanglement. The parameters of this trained network are denoted $P_T$. After the VAE is trained and a 'desirable' solution obtained, 10 additional VAEs with identical setup to the desirable solution are initialized randomly using the Xavier uniform weight initialization on all layers. Each of the 10 initializations contain parameters $P_i$. A line in the parameter space is constructed between the converged 'desirable' solution and the initialized solutions as a function of $\alpha$:

$$P(\alpha) = (1 - \alpha)P_i + \alpha P_T. \tag{3.24}$$

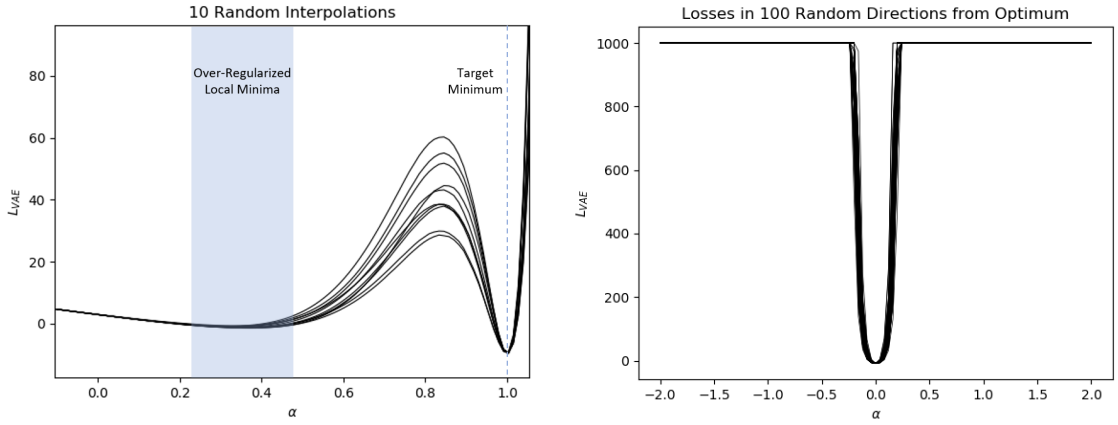



Figure 3.3: (*left*) Loss along interpolated lines between 10 random weight initializations and a desirable converged solution. (*right*) Loss along 100 (of 1,000) random lines emanating from a desirable solution of the DenseVAE architecture. The parameter $\alpha$ indicates the distance along each random direction in parameter space and does not necessarily correspond to the same parameter $\alpha$ in the left figure. Note that the loss is limited to 1,000 for illustration purposes.

Losses are recorded along each of the 10 interpolated lines and plotted in Figure 3.3. Between the random initializations and 'desirable' converged solutions there exists a region of local minima in the loss landscape, and these local minima are characterized by over-regularization. Losses illustrated are computed as an expectation over all training data and a Monte Carlo estimate of the reconstruction loss with 10 latent samples to limit errors due to randomness. We include an illustration of the avoidance of these over-regularized local minima using the described training method in Appendix A.4.

Of particular interest is that this over-regularized local minima region does not fully surround the 'desirable' region. Instead of interpolating in parameter space between random initializations and a converged solution, lines emanating away from the converged solution along 1,000 random directions in parameter space are created and the loss plotted along each. Figure 3.3 illustrates that indeed no local minima are found around the converged solution. We note that there are around 800,000 training parameters in this case, so 1,000 random directions may not completely encapsulate the loss landscape around this solution.

The Xavier uniform weight initialization scheme, and most other initialization schemes, limit the norm of the parameters in parameter space to near the origin. The local minima region seems to exist only between the converged solution region and points in parameter space near the origin. In this case, there may be alternative initialization schemes which can greatly aid in the convergence of VAEs. Similar behavior has been observed in other applications, for example in Ref. [146] where the initialization scheme proposed greatly accelerates the speed of convergence *and* accuracy of reconstruction in prediction tasks.

Over-regularized local minima follow a similar path during training as desirable solutions. A region of attraction exists in the loss landscape, and falling too close to this region will result in an over-regularized solution, illustrated in Figure 3.4. One VAE which obtains a desirable solution shares a similar initial path with an over-regularized solution. Plotted are the VAE losses computed during training. The over-regularized solution breaks from the desired path too early, indicating a necessity for a longer reconstruction-heavy phase during training.

Training many VAEs with various $\beta$ values facilitates a visualization of over-regularization in the RD plane. Each point in Figure 3.4 shows the loss values of converged VAEs trained with different values of $\beta$. The over-regularized region of attraction prevents convergence to desirable solutions for many values of $\beta$. Interpolating in parameter space between each of these points (corresponding to a VAE with its own converged parameters) using the base VAE loss ($\beta = 1$), no other points on the RD curve are local minima of the VAE loss. In Figure 3.4, we observe that during training, the desirable solution reaches the RD curve but continues toward the final solution.

### 3.4.3 Properties of Desirable Solutions

Avoiding over-regularization aids in convergence to solutions characterized by low reconstruction error and higher mutual information between data and latent representation. Among solutions with similar final loss values, inconsistencies remain in latent properties. Two identical VAEs initialized separately often converge to similar loss values, but one may exhibit

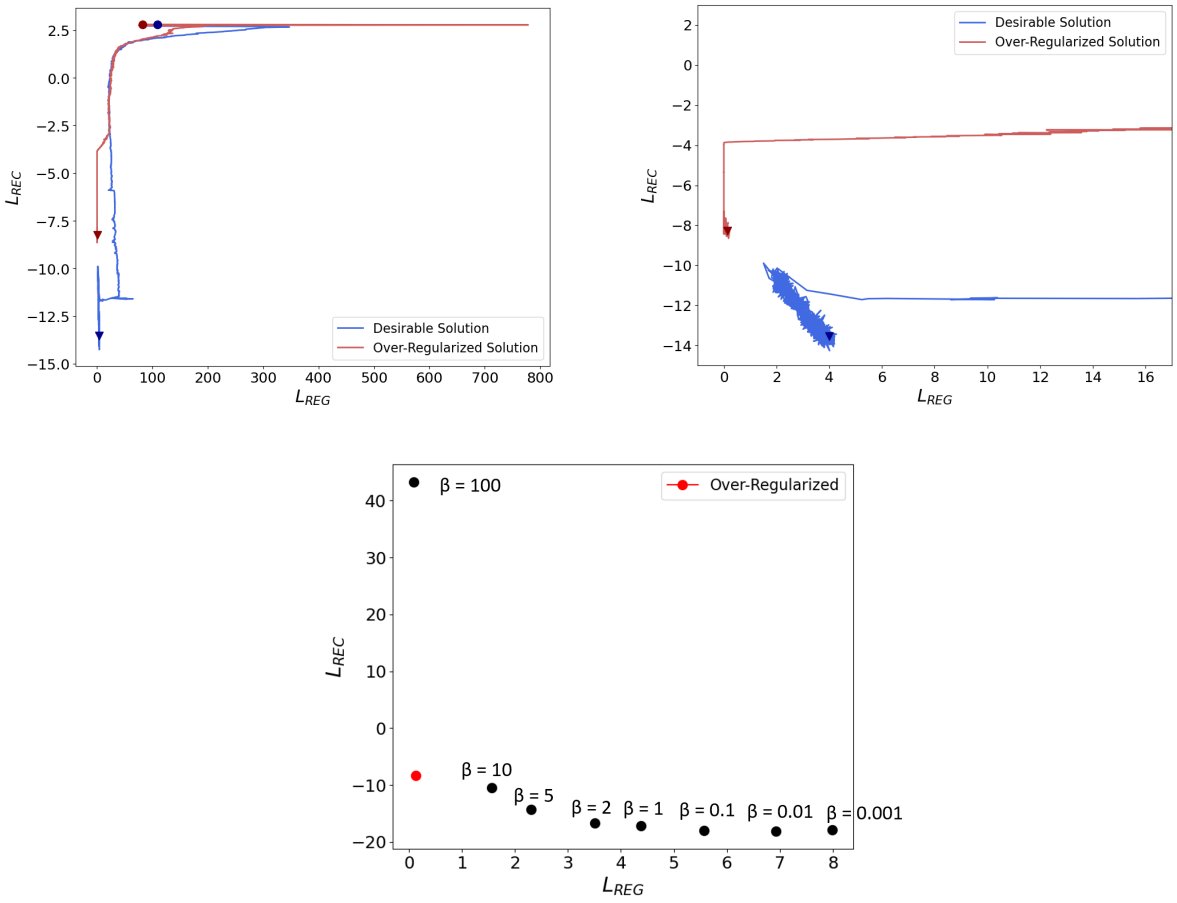



Figure 3.4: RD plane illustrating training convergence of both desirable and over-regularized solutions to the RD curve ($\beta = 1$). (*right*) Scale adjusted. (*lower*) RD plane with points corresponding (from left to right) to $\beta = [100, 10, 5, 2, 1, 0.1, 0.01, 0.001]$. Many values of $\beta$ between 5 and 100 fall into the over-regularized solution.

disentanglement while the other does not. This phenomenon is also explored in Ref. [144] and Ref. [97]. Two VAEs are trained with identical architectures, hyperparameters, and training method; they differ only in the random initialization of network parameters $P$. We denote the optimal network parameters found after training from one initialization as $P_1$ and optimal network parameters found from a separate initialization $P_2$. The losses for each converged solution are quite similar ($L_{VAE_1} \approx -9.50$, $L_{VAE_2} \approx -9.42$); however, disentanglement properties of each are dramatically different. We interpolate between these two solutions in parameter space (Eq. (3.24)) and record losses and disentanglement scores along the line (Figure 3.5). The first network contains a nearly perfectly disentangled latent representation while the second network does not produce a disentangled representation. It is evident that multiple local minima exist in parameter space which converge to similar

values in the loss landscape, but contain very different latent correlations. Local minima exist throughout the loss landscape, and with each initialization, a different local minimum may be found. Many such differing solutions are found throughout our experiments. This phenomenon is partially due to invariance of the ELBO to rotations of the latent space when using rotationally invariant priors. Disentanglement is heavily dependent on a factorized representation of the latent representation. With rotations not affecting the training loss, learning a disentangled representation seems to be somewhat random in this case.

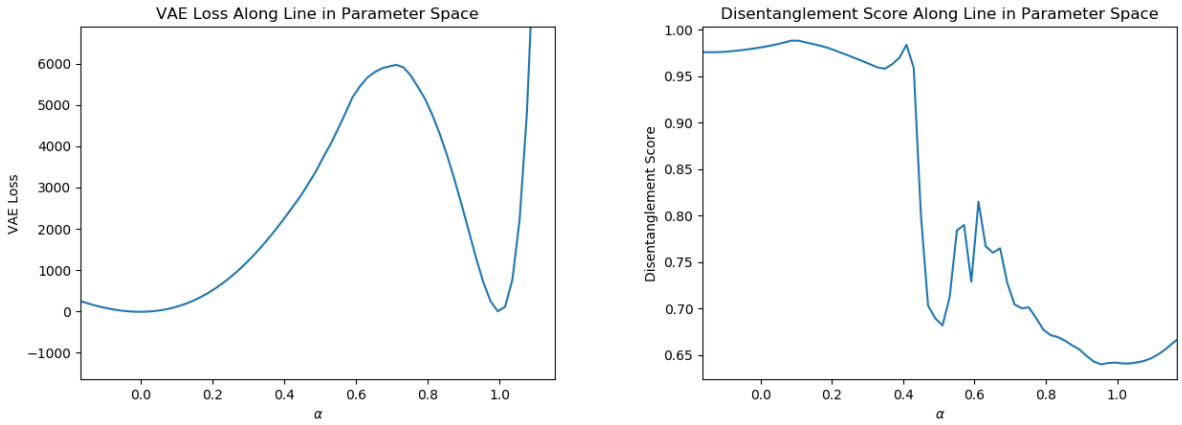



Figure 3.5: (*left*) Loss variation along a line in parameter space between two converged solutions containing identical hyperparameters and training method but different network parameter initializations. (*right*) Disentanglement score along the same line.

This phenomenon exhibits the difficulties in learning a disentangled latent representation correlated to the generative parameters in an unsupervised manner; without prior knowledge of the factors of variation, conclusions cannot be drawn regarding disentanglement by observing loss values alone. In controlled experiments, knowledge of the underlying physically-relevant factors of variation is available, but when only data is available, full knowledge of such factors is often not. It is encouraging that the VAE does have the power to learn physically-relevant latent representations in an unsupervised setting, but the nature of disentanglement must first be understood better to create identifying criterion. It is ultimately realized that without some amount of inductive bias on the applicable physics, the VAE framework cannot consistently learn physically-relevant latent representations.

## 3.5 Unsupervised Disentanglement of Darcy Flow

In this section, we explore the relationship between learning disentangled representations, the aggregated posterior ($q_\phi(\mathbf{z})$), and the generative parameter distribution ($p(\boldsymbol{\theta})$) by incre-

mentally increasing the complexity of $p(\boldsymbol{\theta})$. Disentanglement is illustrated to be achievable but difficult using the classic VAE assumptions and loss due to a lack of enforcement of the rotation of the latent space caused by rotationally-invariant priors. Hierarchical priors are shown to aid greatly in more robustly disentangling the latent space by learning non-rotationally-invariant priors which enforce a particular rotation of the latent space through the regularization loss.

### 3.5.1 Standard Normal Generative Distributions

The intrinsic dimensionality of the data is set to $s = 2$ with a generative parameter distribution $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; 0, \mathbf{I}_{2\times 2})$, the standard normal distribution. Limiting the dimensionality of the generative parameters to two aids greatly in the visualization of the latent space and understanding of the ideas investigated. The standard latent prior is identical to the generative parameter in this case, creating a relatively simple problem for the VAE.

Using the architecture described in Section 3.4.1, the dependence of the regularization loss, reconstruction loss, and disentanglement on the number of training samples is illustrated in Appendix A.5. A similar study is performed in Ref. [144] with a greater sample size. Reconstruction losses continue to fall with the number of training data, indicating improved reconstruction of the data with increased number of samples; however, the regularization loss increases slightly with the number of training data. With too few samples, reconstruction performance is very poor and over-regularization (near zero regularization loss) seems unavoidable. Clear and consistent correlations exist among the loss values and number of training data, but disentanglement properties vary greatly among converged VAEs (Section 3.4.3). The compressed representations range from nearly perfect disentanglement to almost no correlation with the generative parameters.

Although disentanglement properties are inconsistent between experiments, desirable latent representations are often observed. Training is performed using the maximum amount of available data (512 snapshots), and analysis included for 512 testing samples on the dataset (regardless of $p(\boldsymbol{\theta})$). A comparison between a test data sample and the reconstructed mean using the trained VAE is depicted in Figure 3.6, showing little error between the mean $\mu_\psi(\mathbf{z})$ of the decoding distribution and the input data sample. Figure 3.6 also illustrates the aggregated posterior matching the prior distribution in shape. This is unsurprising with a generative parameter and prior distribution match and an expressive network architecture. Finally, Figure 3.7 shows the correlation between the generative parameters of the training and testing data against the latent distribution as a qualitative measure of disentanglement. Each latent dimension is tightly correlated to a single but different generative parameter.

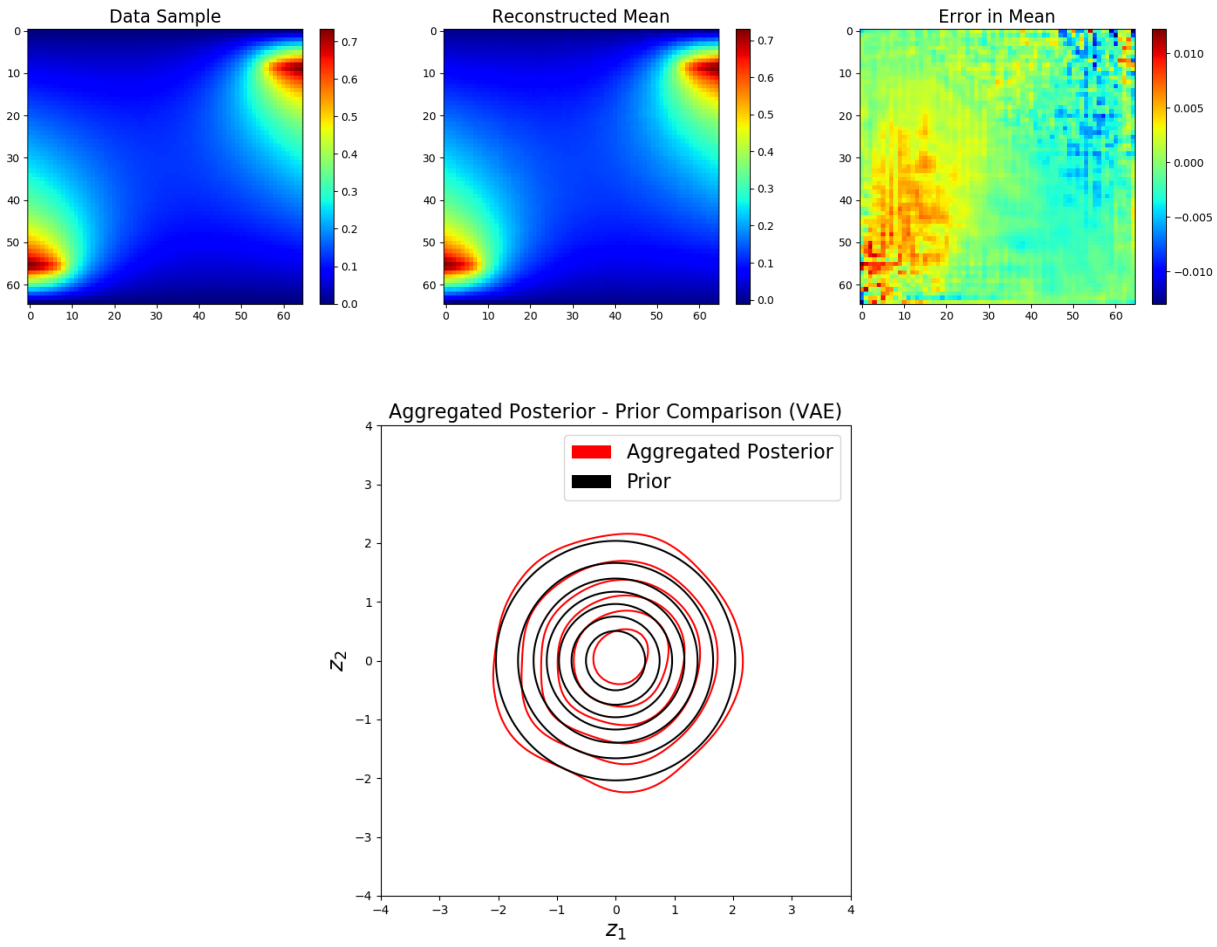



Figure 3.6: (*left*) Data sample from unseen testing dataset. (*center*) Reconstructed data sample from trained VAE. (*right*) Error in the reconstruction mean. (*lower*) Comparison of aggregated posterior ($p_\phi(\mathbf{z})$) and prior ($p(\mathbf{z})$) distributions.

Figure 3.7 also illustrates the uncertainty in the latent parameters, effectively $q_\phi(\mathbf{z}|\boldsymbol{\theta})$. The latent representation is fully disentangled; each latent parameter contains only information about a single generative factor. However, in this case the prior distribution is set to the same as the true generative parameter distribution, providing a relatively easy case to 'discover' the generative parameters.

### 3.5.2 Non Standard Gaussian Generative Distributions

The generative parameter distribution and the prior are identical (independent standard normal) in the previous example. Most often, however, knowledge of the generative parameters is not possessed. The specified prior in this case is unlikely to match the generative

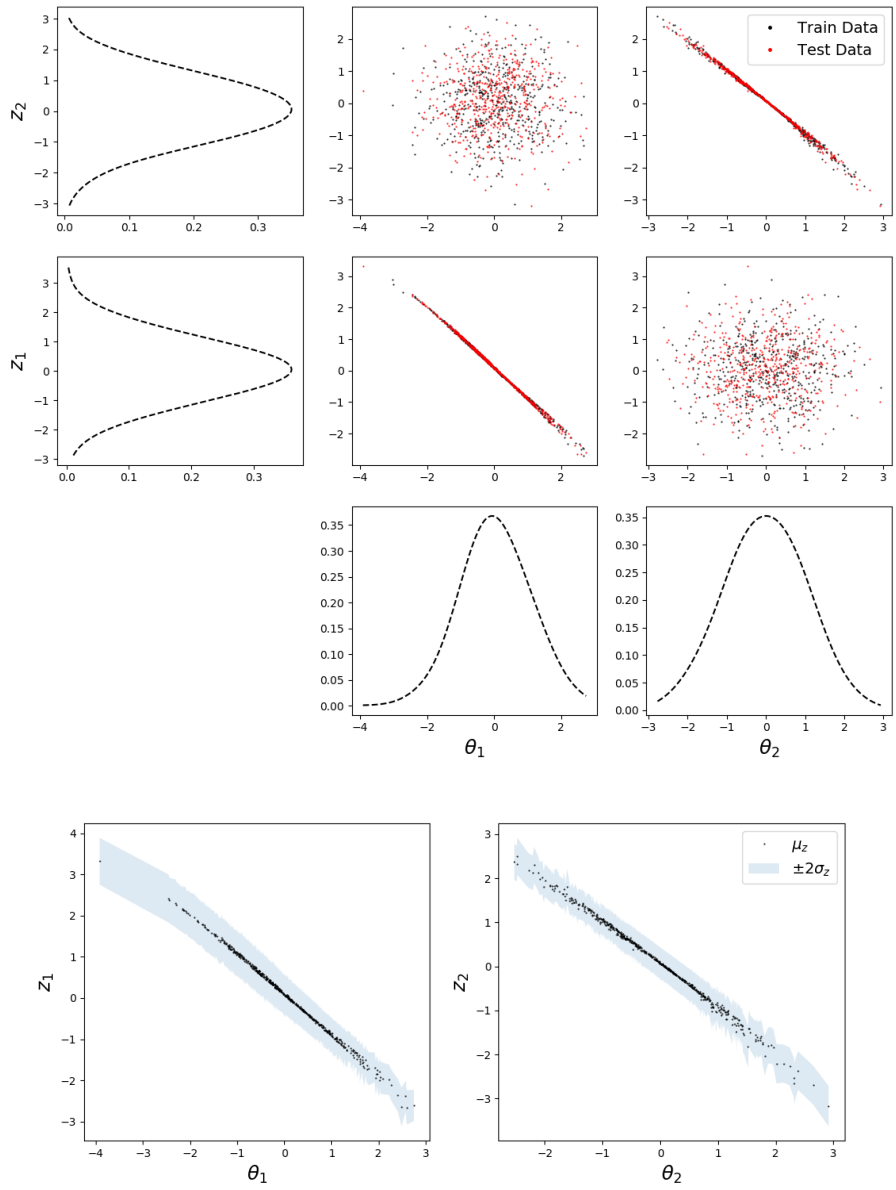



Figure 3.7: (*upper*) Correlations between dimensions of generative parameters and mean of latent parameters. Also shown are the empirical marginal distributions of each parameter. (*lower*) Correlations between generative parameters and latent parameters with uncertainty for test data only.

parameter distribution. The next example illustrates the application of a VAE in which the generative parameter distribution and prior do not match. A additional dataset with $s = 2$ is generated by sampling the generative parameters from a non standard Gaussian distribution. The generative parameter distribution is Gaussian, but scaled and translated relative to the previous example $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; [1; 1], [0.5, 0; 0, 0.5])$.

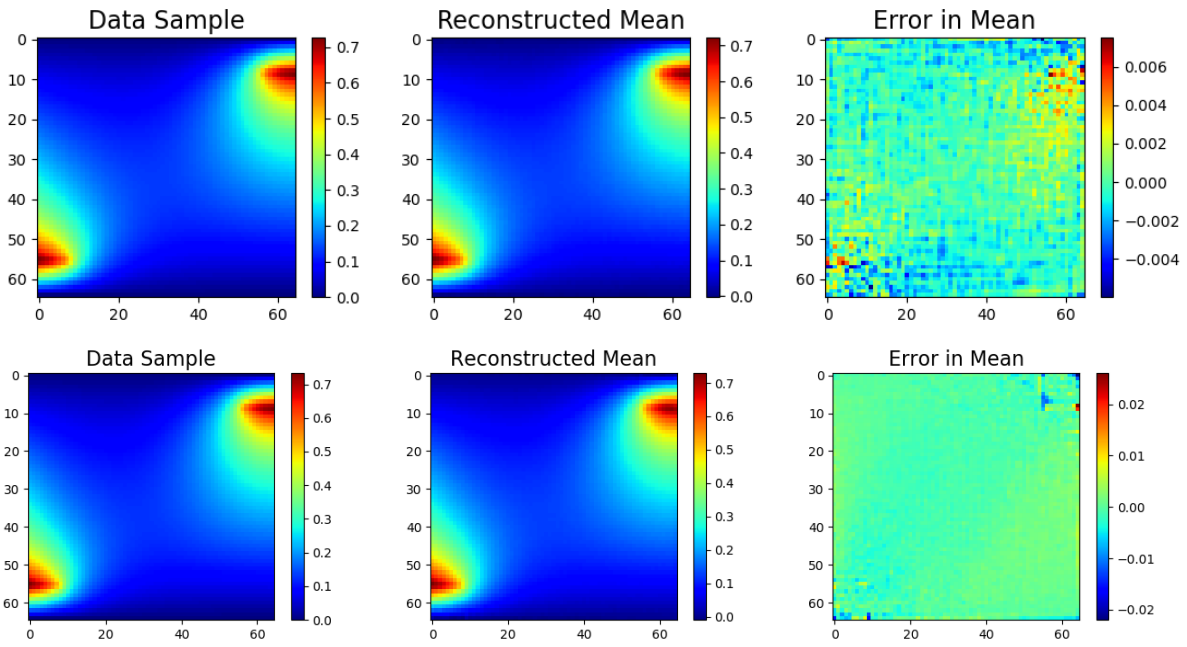



Figure 3.8: (*top*) Reconstruction accuracy of a test sample on trained VAE without hierarchical network, (*bottom*) with hierarchical network.

Training a standard VAE on this dataset results in high reconstruction accuracy, but undesirable disentanglement after many trials. With the use of an additional hierarchical prior network, latent representations which correlated better to the generative parameters can be achieved even with a mismatch in the prior and generative parameter distributions. The sub-prior (Section 3.3.2) is the standard normal distribution, but the hierarchical network learns a non-standard normal prior. Still, the learned prior and generative parameter distributions do not match. Figures 3.8 and 3.9 illustrate comparisons in results obtain from the VAE with and without the hierarchical prior network. When using hierarchical priors, the learned prior and aggregated posterior match reasonably well but do not match the generative parameter distribution. However, this does not matter as long as the latent representation is not *rotated* relative to the generative parameter distribution, as illustrated in the next example. Low reconstruction error and disentanglement are observed using hierarchical priors, but disentanglement was never observed using the standard VAE after many experiments. This may be because $\beta$ is not large enough to enforce a regularization loss large enough to produced an aggregated posterior aligned with the axes of the generative parameter distribution. Therefore, the rotation of the learned latent representation will be random and disentanglement is unlikely to be observed, even in just two dimensions. The hierarchical network consistently enforces a factorized aggregated posterior, which is essential for disentanglement when generative parameters are independent. One potential cause

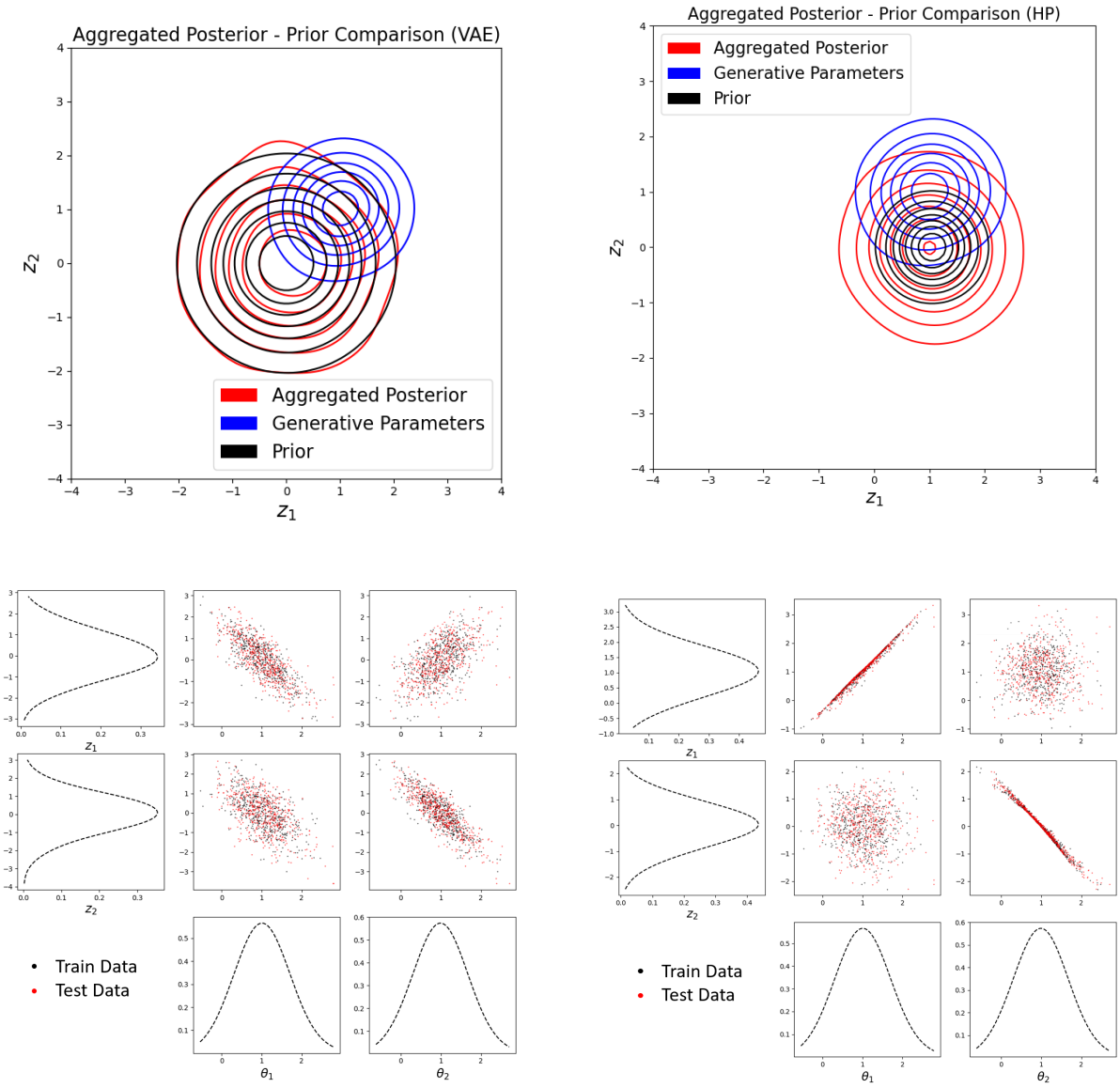



Figure 3.9: (*upper left*) Aggregated poster, prior, and generative parameter distribution comparison on VAE without hierarchical network, (*upper right*) with hierarchical network. (*lower left*) Qualitative disentanglement in VAE trained without hierarchical network, (*lower right*) with hierarchical network.

of this is the learning of non-rotationally-invariant priors, such as a factorized Gaussian with independent scaling in each dimension. The ELBO loss in this case *is* affected by rotations of the latent space, aligning the latent representations to the axes of the generative parameters.

A latent rotation can be introduced such that the reconstruction loss is unaffected, but regularization loss changes with rotation. Introducing a rotation matrix $\mathbf{A}$ with an-

gle of rotation $\omega$ to rotate the latent distribution, the encoding distribution becomes $q_\phi(\mathbf{z}|\mathbf{y}) = \mathcal{N}(\mathbf{z}; \mathbf{A}\boldsymbol{\mu}_\phi(\mathbf{y}), \mathbf{A}\text{diag}(\boldsymbol{\sigma}_\phi(\mathbf{y}))\mathbf{A}^T)$. Reversing this rotation when computing the decoding distribution (i.e. $p_\psi(\mathbf{y}|\mathbf{z}) = \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}_\phi(\mathbf{A}^T\mathbf{z}), \text{diag}(\boldsymbol{\sigma}_\psi(\mathbf{A}^T\mathbf{z})))$) preserves the reconstruction loss. However, the regularization loss can be plotted as a function of the rotation angle (Appendix A.7). When a rotationally-invariant prior is used to train the VAE, regularization loss is unaffected by latent rotation. However, when the prior is non-rotationally-invariant, the regularization loss is affected by latent rotation. Thus, rotation of the latent space is enforced by the prior during training, and without prior knowledge on the physically-relevant generative parameters, it is difficult to robustly enforce the correct rotation.

Although the hierarchical prior adds some trainable parameters to the overall architecture to slightly increase expressiveness, the increase is only 0.048%. This is negligible, and it is assumed that this is not the root cause of improved disentanglement. Rather, it is the ability of the additional hierarchical network to consistently express a factorized aggregated posterior and learn non-rotationally-invariant priors which improves disentanglement. More insights on this behavior are discussed in the next example and Section 3.7.

### 3.5.3 Multimodal Generative Distributions

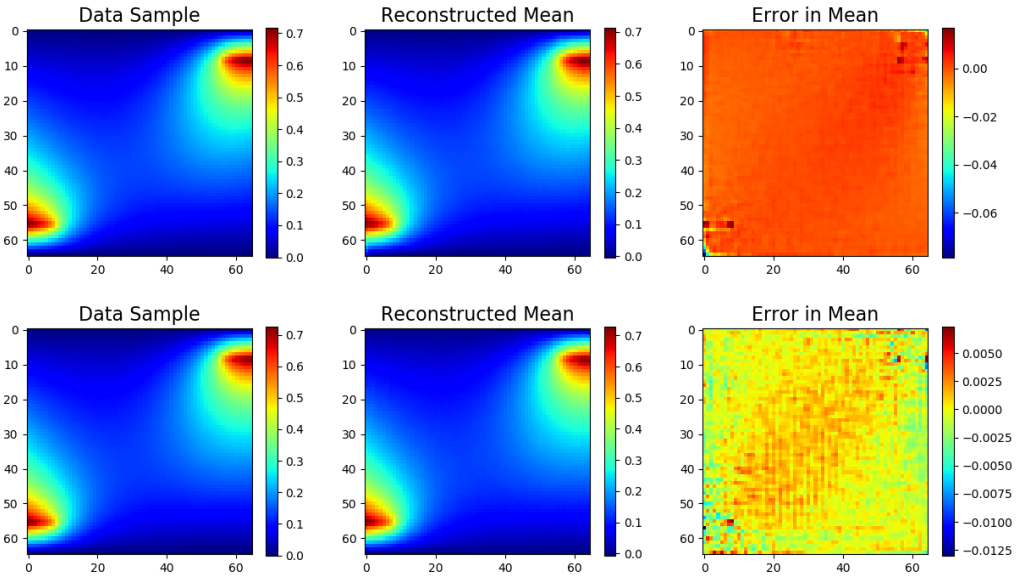



Figure 3.10: (*top*) Reconstruction accuracy of a test sample using VAE trained on multimodal generative parameter distribution without hierarchical network, (*bottom*) with hierarchical network.

In this setup, disentanglement not only depends on a factorized $q_\phi(\mathbf{z})$, but the correlations in $p(\boldsymbol{\theta})$ must be preserved as well, i.e. rotation of the aggregated posterior with

respect to the generative parameter distribution matters. The previous example illustrates a case in which the standard VAE fails in disentanglement but succeeds with the addition of hierarchical priors due to improved enforcement of a factorized $q_\phi(\mathbf{z})$ through learning non-rotationally-invariant priors. The generative parameter distribution is radially symmetric, thus visualization of rotations in $q_\phi(\mathbf{z})$ relative to $p(\boldsymbol{\theta})$ is difficult. To illustrate the benefits of using hierarchical priors for disentanglement, the final example uses data generated from a more complex generative parameter distribution with four lines of symmetry for better visualization. The generative parameter distribution is multimodal (a Gaussian mixture) and is more difficult to capture than a unimodal Gaussian distribution, but allows for better rotational visualization:

$$p(\boldsymbol{\theta}) = \frac{1}{4}\mathcal{N}(\boldsymbol{\theta}; [-1; -1], [0.25, 0; 0, 0.25]) + \frac{1}{4}\mathcal{N}(\boldsymbol{\theta}; [1; 1], [0.25, 0; 0, 0.25]) + \frac{1}{4}\mathcal{N}(\boldsymbol{\theta}; [-1; 1], [0.25, 0; 0, 0.25]) + \frac{1}{4}\mathcal{N}(\boldsymbol{\theta}; [1; -1], [0.25, 0; 0, 0.25]) \ .$$

Training VAEs without hierarchical priors results in over-regularization more often than with the implementation of HP. Out of 50 trials, 10% trained without HP were unable to avoid over-regularization while all trials with HP successfully avoided over-regularization. More epochs are required in the reconstruction-heavy phase (both with and without HP) to avoid over-regularization than in previous examples. Disentanglement was never observed without the use of hierarchical priors. This again is likely due to rotation of the latent space relative to the generative parameter distribution as a result of employing rotationally invariant priors. To illustrate this concept, Figure 3.11 illustrates the effects of rotation of the latent space on disentanglement. Clearly, rotation dramatically impacts disentanglement, and the standard normal prior does not enforce any particular rotation of the latent space.

Implementing hierarchical priors, consistent observation of not only better reconstruction (avoiding over-regularization) but also reasonable disentanglement of the latent space in roughly half of all trained VAEs (out of 50) exemplifies the improved ability of hierarchical priors to produce a disentangled latent representation. Reconstruction of test samples is more accurate when implementing the hierarchical prior network, as illustrated in Figure 3.10. It is hypothesized that disentanglement is observed in roughly half of our experiments due to local minima in the regularization loss corresponding to 45 degree rotations of the latent space, illustrated in Appendix A.6. The learned priors using HP are often non-rotationally-invariant and aligned with the axes. However, the posterior is often rotated 45-degrees relative to this distribution, creating a non-factorized and therefore non-disentangled representation. Thus, it may not be possible to consistently and robustly learn a latent representation which is correctly rotated with respect to the generative parameters without some prior knowledge

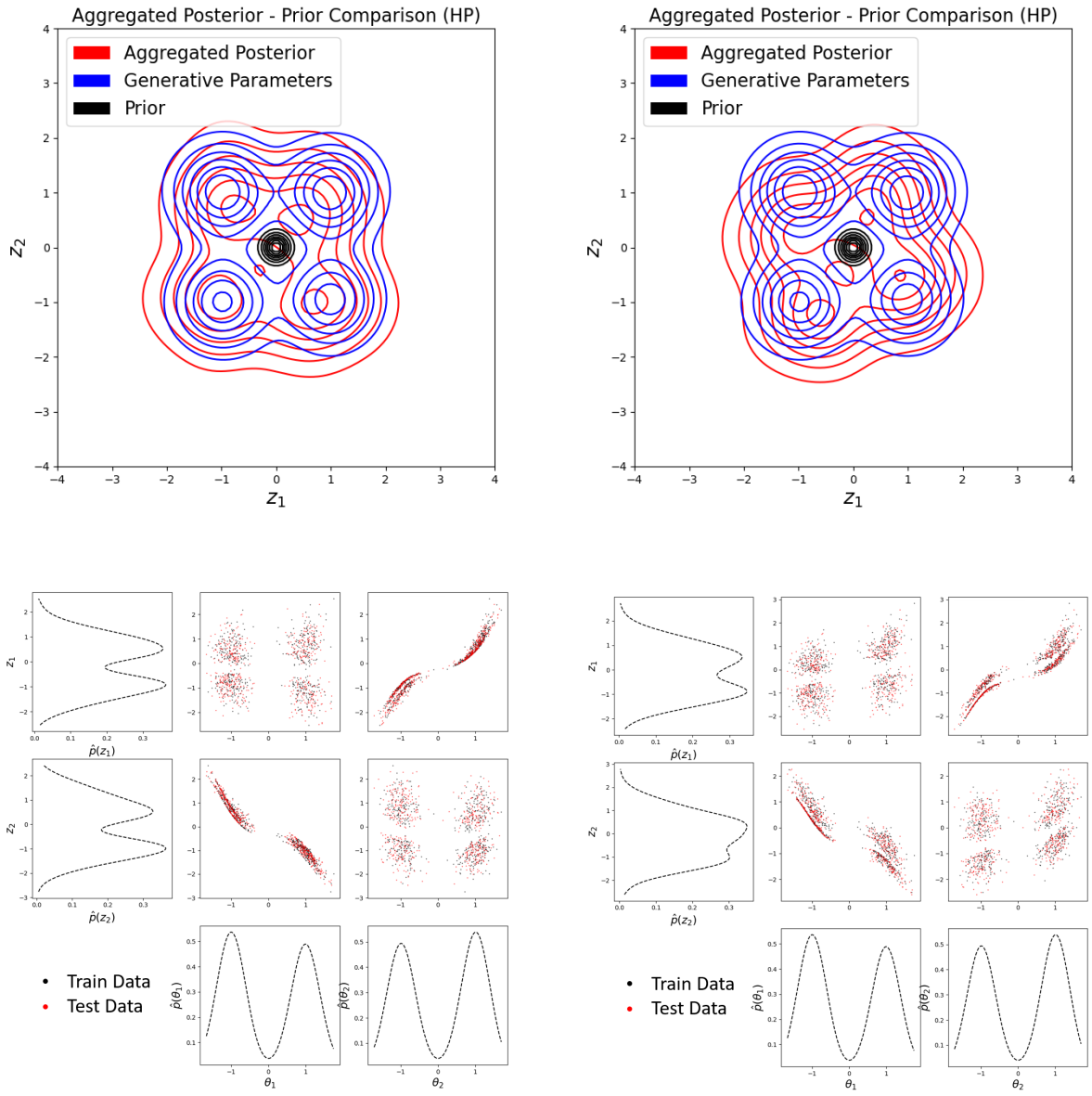



Figure 3.11: (*top*) aggregated posterior comparison showing rotation of the latent space, (*bottom*) worse disentanglement when latent space is slightly rotated.

of the parameters themselves.

Comparing $p(\boldsymbol{\theta})$, $p(\mathbf{z})$, and $q_\phi(\mathbf{z})$ with and without HP (Figure 3.12), stark differences are noticeable. Without the HP network, the aggregated posterior often captures the multimodality of the generative parameter distribution, but it is rotated randomly relative to $p(\boldsymbol{\theta})$, creating a non-factorized $q_\phi(\mathbf{z})$. Training the VAE with hierarchical priors, the learned prior becomes non-rotationally invariant. The rotation of the aggregated posterior is there-

fore controlled by the orientation of the prior through the regularization loss, but mimics the shape of the generative parameter distribution. It is clear that the prior plays a significant role in terms of disentanglement: it controls the rotational orientation of the aggregated posterior.

A qualitative measure of disentanglement is compared in Figure 3.12. Without HP, the latent parameters are entangled; they are each weakly correlated to both of the generative parameters. Adding HP to the VAE results in disentanglement in nearly half of our trials. When disentanglement does occur, each latent factor contains information on mostly a single but different generative factor. Through the course of our experiments, a relationship between disentanglement and the degree to which the aggregated posterior matches the generative parameter distribution is recognized. When disentanglement does not occur with the use of HP, the aggregated posterior is rotated relative to $p(\boldsymbol{\theta})$, or non-factorized (it has always been observed at around a 45-degree rotation). Only when $q_\phi(\mathbf{z})$ can be translated and scaled to better match $p(\boldsymbol{\theta})$, maintaining the correlations, does disentanglement occur. Thus, a quantitative measure of disentanglement (Eq 3.12) is created from this idea. The KL divergence is estimated through sampling using the $k$-nearest neighbors ($k$-NN) approach (version $\epsilon 1$) found in Ref. [147]. The optimization is performed using the gradient-free Nelder-Mead optimization algorithm [70].

In low-dimensional problems, humans are adept at determining disentanglement from qualitative measurements of disentanglement such as Figure 3.12. It is, however, more difficult to obtain quantitative measurements of these properties. Figure 3.13 shows the relationship between Eq. 3.12 and a qualitative measurement of disentanglement. Lower values of $S_{KL}$ indicate better disentanglement. However, it is primarily concluded that without prior knowledge of the generative parameters, it may be impossible to reliably learn a representation corresponding to such parameters. In this following section, it is shown that labeling a few samples with their true generative parameters proves enough of an inductive bias to accurately label the remaining data samples with their correct generative parameters.

## 3.6 Physical Learning Bias with Weak Supervision

Difficulties with consistently disentangling generative parameters have been illustrated up to this point with an unsupervised VAE framework. In some cases, however, generative parameters may be known for some number of samples, suggesting the possibility of a semi-supervised approach. These labeled samples can be leveraged to further improve the consistency of learning a disentangled representation by enforcing their known values in the latent space. Consider data consisting of two partitions: labeled data $\{\mathbf{y}^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{l}$ and unlabeled

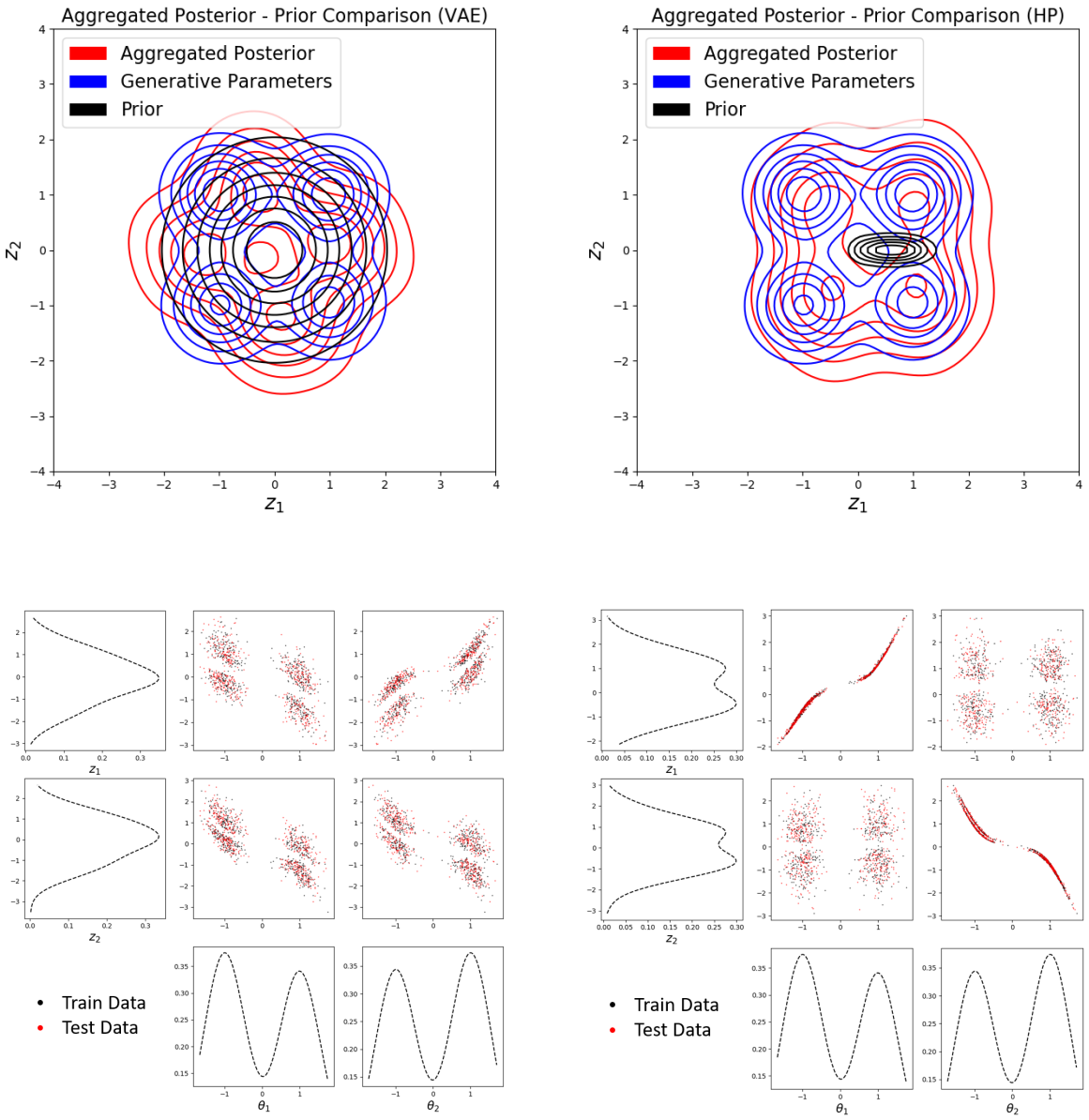



Figure 3.12: (*upper left*) Aggregated posterior, prior, and generative parameter distribution comparison using VAE trained on multimodal generative parameter distribution without hierarchical network, (*upper right*) with hierarchical network. (*lower left*) Qualitative disentanglement using VAE trained on multimodal generative parameter distribution without hierarchical network, (*lower right*) with hierarchical network.

data $\{\mathbf{y}^{(i)}\}_{i=l+1}^{u+l}$. A one-to-one mapping between the generative parameters $\boldsymbol{\theta}$ and the learned latent representation $\mathbf{z}$ is sought when disentanglement is desired. Thus, enforcing the latent representation to match the generative parameters for labeled data in a semi-supervised

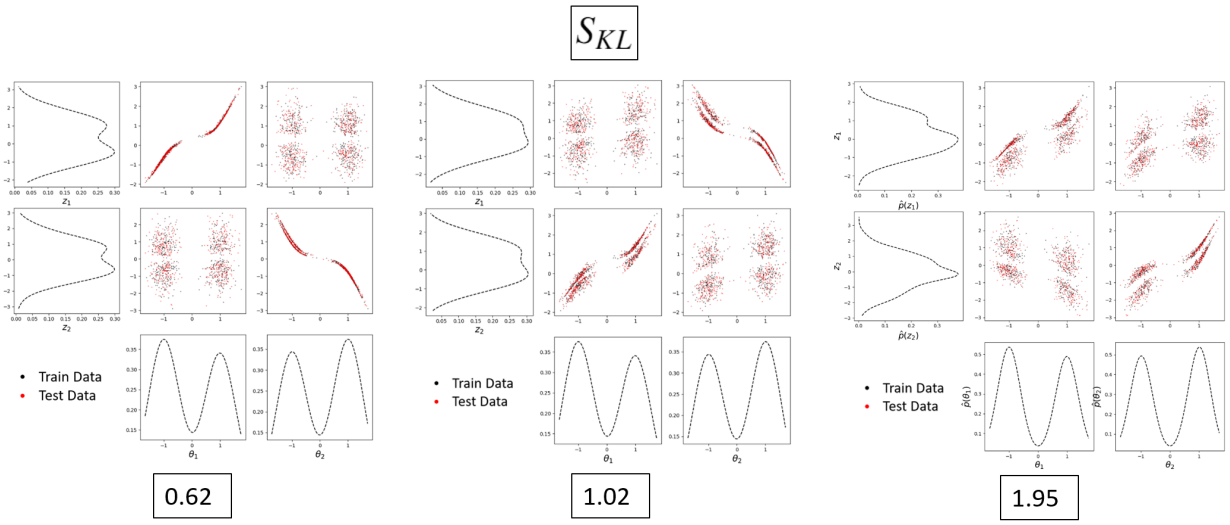



Figure 3.13: A quantitative measure of disentanglement compared to a qualitative measure. As $S_{KL}$ increases, the latent space becomes more entangled.

approach should aid in achieving our desired objective more consistently.

We begin the intuition behind a semi-supervised loss function by illustrating its connection to the standard ELBO VAE loss. One method of deriving the ELBO loss is to first expand the relative entropy between the data distribution and the induced data distribution to obtain

$$\begin{aligned} D_{KL}[p(\mathbf{y})||p_\psi(\mathbf{y})] = &- H(Y) + \mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})]] \\ &- \mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z}|\mathbf{y})]] \\ &- \mathbb{E}_{p(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\log p_\psi(\mathbf{y}|\mathbf{z})] \end{aligned}$$

where $-H(Y)$ is constant and the 'true' encoder $p(\mathbf{z}|\mathbf{y})$ is unknown. Therefore, the term

$$\mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z}|\mathbf{y})]]$$

is usually ignored and we arrive at the ELBO, which upper bounds the left hand side. However, a relationship between $\mathbf{z}$ and $\mathbf{y}$ is known for labeled samples. This relationship can be used to assign $p(\mathbf{z}^{(i)}|\mathbf{y}^{(i)})$ on the labeled partition. For unlabeled data, the standard

ELBO loss is still used for training and the semi-supervised loss to be minimized becomes

$$\begin{aligned}\mathcal{L}_{VAE-SS}(\phi,\psi) =& \mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})]] \\ &-\mathbb{E}_{p_l(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z}|\mathbf{y})]] \\ &-\mathbb{E}_{p(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\log p_\psi(\mathbf{y}|\mathbf{z})] \end{aligned} \tag{3.25}$$

where $p_l(\mathbf{y})$ is the distribution of inputs with corresponding labels, $p(\mathbf{y})$ is the distribution of all inputs (labeled and unlabeled), and $\mathbb{E}_{p_l(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z}|\mathbf{y})]]$ is denoted $\mathcal{L}_{SS}$.

Empirically it is found that this loss is very sensitive to changes in network parameters and unreasonably small learning rates are required for stability. Additionally, there is no obvious way to determine the variance of $p(\mathbf{z}^{(i)}|\mathbf{y}^{(i)})$ for each sample, only the mean is easily identifiable. We therefore propose to train with $\mathcal{L}_{SS} = \mathbb{E}_{p_l(\mathbf{y})}[-\log q_\phi(\mathbf{z}|\mathbf{y})]$ instead such that the loss function becomes

$$\mathcal{L}_{VAE-SS}(\phi,\psi) = \mathbb{E}_{p(\mathbf{y})}[D_{KL}[q_\phi(\mathbf{z}|\mathbf{y})||p(\mathbf{z})]] - \mathbb{E}_{p_l(\mathbf{y})}[\log q_\phi(\mathbf{z}|\mathbf{y})] - \mathbb{E}_{p(\mathbf{y})q_\phi(\mathbf{z}|\mathbf{y})}[\log p_\psi(\mathbf{y}|\mathbf{z})] \ . \tag{3.26}$$

Training with this loss achieves the desired outcome of consistently learning disentangled representations while being simple and efficient to implement.

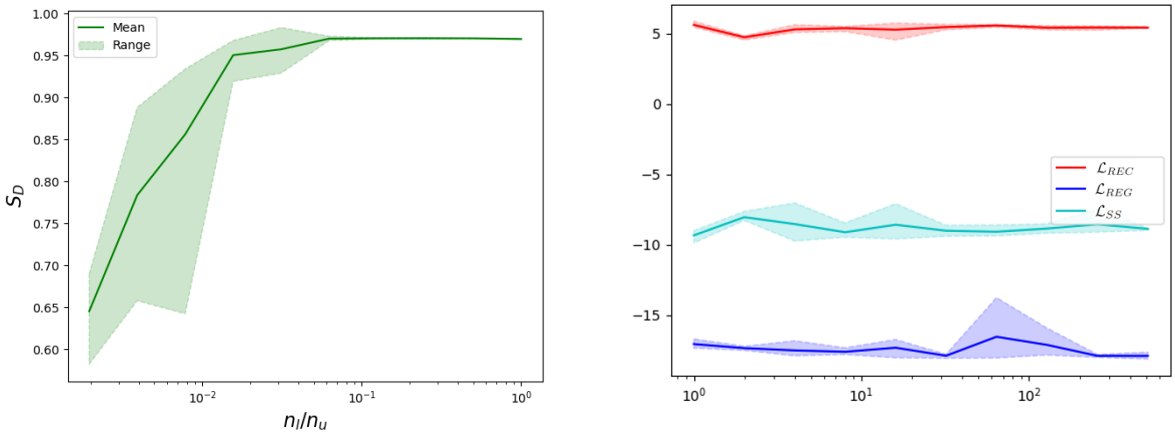



Figure 3.14: *(left)* Disentanglement score mean increases with ratio of labeled to unlabeled samples when training with a semi-supervised loss. Disentanglement also becomes more consistently observed. *(right)* Training losses are unaffected by the number of labeled samples.

Incorporating a small amount of labeled samples into training the VAE, a latent representation corresponding closely to the generative parameters can be consistently learned, even for the unlabeled samples. Figure 3.14 illustrates the relationship between increasing the

number of labeled samples and the disentanglement score of the learned latent representation. As the disentanglement score increases, the latent representation more closely matches the true generative parameter representation. In each case, there are 512 unlabeled samples during training. Each trial varies in the number of labeled samples, and VAEs trained with the same number of labeled samples are trained with a different set of labeled samples. Ten VAEs are trained at each point, and the range illustrated represents the maximum and minimum disentanglement score across the 10 trials.

The training losses do not seem to be effected by the number of labeled samples, only the disentanglement score is effected. With a low number of labeled samples, the semi-supervised VAE trains very similarly to the unsupervised VAE. That is, disentanglement is observed rather randomly, and the learned latent representation varies dramatically between trials. Labeling around 1% of the samples begins to result in consistently good disentanglement. Labeling between 3% and 8% results in learning disentangled latent representations which are nearly identical between trials. It follows from these results that disentangled representations can be consistently learned when training with Eq. 3.26 when using a sufficient number of labeled samples (assuming a sufficiently expressive architecture).

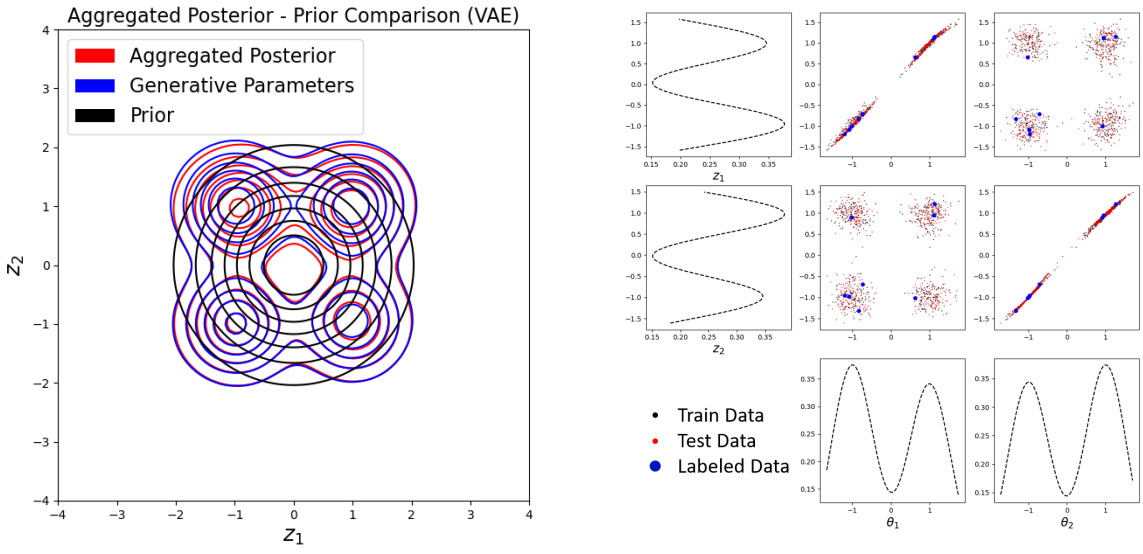



Figure 3.15: *(left)* Aggregated posterior matches the generative parameter distribution with semi-supervised training. *(right)* Multi-modality is well preserved.

Using a semi-supervised method also improves the ability of the VAE to predict data in regions of lower density. In Figure 3.15, it is observed that the aggregated posterior matches the generative parameter distribution much better than the unsupervised case with just over

1% of the samples labeled. Additionally, regions of low density in the generative parameter distribution are better represented in the semi-supervised case over the unsupervised case; in other words, multimodality is better preserved (compare to Figure 3.12). These results demonstrate that introducing a small amount of bias in the form of known physics consistently improves the representation of the data learned, enforcing it to be physically-relevant. It allows for the remaining samples to be labeled with their true generative parameters, leading to a labeled dataset which can be leveraged for a variety of downstream tasks.

## 3.7 Summary

In this chapter, investigate the use of VAEs as an ML technique to automatically extract physical principles in an unsupervised way. The insights gleaned from these explorations underscore the impact that VAEs can have in the realm of computational physics, particularly in terms of enhancing data interpretability and facilitating the discovery of underlying physical principles through data.

The application of VAEs within a physics-aware framework has demonstrated the ability to distill complex physical datasets into more interpretable, low-dimensional representations. This process not only aids in uncovering the latent structures within the data but also in aligning these structures closely with physically relevant parameters, thereby offering a more nuanced understanding of the physics through the lens of machine learning.

A critical takeaway from this chapter is the realization that, without physically-relevant biases—the information that guides the learning process towards more physically-plausible representations—VAEs, and likely other ML models, struggle to provide robust physical insights on their own. The introduction of even a minimal amount of physics information into these models can dramatically enhance their ability to learn and generalize physics concepts, underscoring the indispensable role of physics-aware ML. This insight is particularly relevant in the context of VAEs, where the integration of hierarchical priors and loss-weighting schedules has proven useful in optimizing the training process and overcoming challenges such as over-regularization, limited mutual information, and limited disentanglement capabilities. However, even with these advancements, the representations learned are not consistently correlated to the physically-relevant parameters desired.

These explorations further lead us to infer that while purely ML-driven models exhibit remarkable capabilities in pattern recognition and prediction, their applicability remains constrained by their lack of physical grounding. This limitation highlights the necessity for an approach that incorporates physics directly into the model, as exemplified by physics-informed and physics-augmented machine learning techniques. Such methods not only bol-

ster the interpretability and generalizability of the models but also ensure that predictions remain physically plausible, thereby mitigating the risk of generating non-physical or uninterpretable outcomes.

Looking ahead, the insights and conclusions developed herein serve as a solid foundation for the subsequent chapters, where we will continue to explore the potential and application of physics-aware machine learning in greater depth through a variety of case studies. This chapter provides evidence that including inductive biases in ML models can greatly enhance their robustness in modeling the physical world, and this is further demonstrated in the following works.

# CHAPTER 4

# Enhancing Dynamics Modeling with Data-driven Inference and Interpretable Machine-learning Augmentations

Cathodic electrophoretic deposition (EPD), more commonly known as e-coating, represents a cornerstone technique within industries such as automotive and manufacturing, serving to apply protective coatings to various surfaces. This process is important in preventing corrosion and ensuring the durability of components, marking its significance in achieving both aesthetic and functional quality standards. However, the quest for optimal coating properties and process efficiency necessitates a deep dive into the accurate modeling of EPD dynamics, a challenge compounded by the intricate electrochemical interactions at play.

The journey to refine EPD models faces significant hurdles, primarily due to the uncertainties surrounding the physical properties of the coating film during deposition and a limited grasp of the underlying physics governing the process. Traditional approaches to modeling e-coating have ventured to overcome these obstacles, yet they often fall short, requiring empirical calibration to account for film initiation and growth, or failing to fully capture experimental behaviors [148, 149, 150, 151, 152]. Moreover, the intricacies of modeling the deposition onset of the coating film, characterized by threshold parameters, introduce discontinuities in model outputs, further complicating the parameter inference process. This gap in modeling accuracy and predictability underscores the need for innovative approaches that differ from conventional methodologies. An extensive study on the modeling of the e-coat process and chemistry involved for constant current and constant voltage boundary conditions can be found in Refs. [148, 153].

In response to these challenges, this work aims to refine a baseline EPD model, aiming to bridge the gap between theoretical predictions and experimental realities. At the heart of this endeavor lies the differentiation between two major types of uncertainties: parametric uncertainties, which stem from imprecise knowledge about parameter values within the

model, and model form uncertainties, which arise from the inherent assumptions and simplifications embedded in the computational model itself. Addressing these uncertainties requires a nuanced approach, blending experimental data with advanced computational techniques to achieve a more faithful representation of the e-coating process.

The adoption of variational inference methods to estimate model parameters is employed in tackling parametric uncertainties, albeit challenged by the model's inherent unidentifiability issues. To circumvent these barriers, modifications to the baseline model are proposed, enhancing parameter identifiability and fostering a more generalizable framework across experimental conditions. Yet, the pursuit of an accurate and interpretable model does not end here. The residual gap between the model's predictions and observed experimental data beckons the integration of machine learning tools, particularly Neural Ordinary Differential Equations (NeuralODE) [26], to introduce learnable, physically relevant modifications.

By leveraging NeuralODE, this work demonstrates the incorporation of machine learning augmentations into the e-coat modeling process, aiming to refine the model's performance in alignment with experimentally obtained data. This approach not only enhances the model's fidelity to real-world behaviors but also upholds the principle of physical interpretability and therefore generalizability, a cornerstone in the overarching themes of physics-aware machine learning and this dissertation. As such, this chapter introduces ideas which combine traditional physical modeling with the capabilities of machine learning, illustrating a comprehensive exploration of this synergy within the context of e-coating.

## 4.1 Background

### 4.1.1 Cathodic Electrophoretic Deposition

In the automotive industry, e-coating typically involves using a version of Sand's equation [154] to compute the induction time under constant current conditions. However, the application of a linearly increasing voltage over time is also a common practice in automotive e-coating to ensure good throw-power and prevent defects associated with high voltage coating [155]. Effective models are crucial for predicting the quality of the coating, optimizing the process for energy efficiency, and reducing material waste. In industries where durability and coating quality are paramount, such as in automotive manufacturing, the role of precise EPD modeling is critical. It ensures that the coatings applied are uniform, adhere well, and provide long-lasting protection against environmental factors. Moreover, in sectors focusing on high-precision manufacturing, such as aerospace and electronics, the accurate modeling of EPD processes is essential for meeting stringent quality and performance standards. Over-

all, the development of more robust and accurate models for EPD is not just a theoretical pursuit but a practical consideration. It supports the optimization of industrial processes, enhances product quality, and contributes to sustainable manufacturing practices.

Although many models exist to model e-coat dynamics [148, 149, 150, 151, 152], this work employs a baseline model which is leveraged as a starting point to improve e-coat modeling using data while maintaining physical interpretability.

#### 4.1.1.1 Baseline Model

The baseline model is one-dimensional in space, consisting of a cathode and anode placed length $L$ apart and filled with a solution of suspended colloidal particles in between (see Fig. 4.1). After a voltage is applied and prior to film deposition, an electrochemical reaction takes place at the cathode with $2H^+ + 2e^- \longrightarrow H2$ or $2H_2O + 2e^- \longrightarrow 2OH^- + H_2$ [152]. As the reaction proceeds, the bath solution basicity increases until a critical pH value is reached and film deposition starts. Film deposition is defined by suspended colloidal particles being deposited on the anode, increasing the circuit resistance.


Figure 4.1: Initial setup for the 1D case.

The film deposition process is separated into three components. First, the electric field within the bath is computed using the conservation of current density given by

$$\nabla \cdot \mathbf{j} = 0 \tag{4.1}$$

$$\mathbf{j} = -\sigma_{\text{bath}} \nabla \phi \tag{4.2}$$

$$\phi|_{\Gamma} = R_{\text{film}}\, j_n \quad \text{at the interface film-bath,} \tag{4.3}$$

where $\mathbf{j}$ is the current density, $\sigma_{\text{bath}}$ is the bath conductivity, $j_n = \mathbf{j} \cdot \mathbf{n}$ is the normal component of the current density, $\phi$ is the electrical potential, $R_{\text{film}}$ is the film resistance and $\Gamma$ represents the interface between the film and the bath. Second, once film deposition begins, deposition rate is defined as

$$\frac{dh}{dt} = C_v j_n, \tag{4.4}$$

where $h$ is the film thickness and $C_v$ is the Coulombic efficiency. Third and finally, the film thickness and film resistance are related through

$$\frac{dR_{\text{film}}}{dt} = \rho(j_n)\frac{dh}{dt}, \tag{4.5}$$

where $\rho(\mathbf{j})$ is the film resistivity. In particular, $\rho(\mathbf{j})$ is a decreasing function of the current density, and we adopt an empirical estimate from [156] of

$$\rho = \max(8 \times 10^5 \exp(-0.1j),\ 2 \times 10^6). \tag{4.6}$$

Before deposition begins, the right hand side of Eqs. 4.5 and 4.4 are both equal to 0. The model defines the deposition onset event according to two criteria: the minimum current density and minimum charge conditions. The onset criteria are critical for accurately predicting the film thickness growth in time. If both of the conditions are met, film deposition begins in the baseline model.

The first condition which determines deposition onset is a minimum current density condition. Once the current density at the cathode reaches a threshold value $j_{min}$, the film thickness begins to increase if the other condition is met. The onset condition parameter $j_{min}$ is unknown and estimated or inferred from experimental data. The second onset condition is a minimum charge condition. The minimum charge criterion assumes that the deposition does not start until the accumulated charge on the cathode reaches a threshold value $Q_{min}$, with the electric charge $Q$ defined by

$$Q(t) = \int_t j_n dt. \tag{4.7}$$

The deposition onset threshold $Q_{min}$ is also unknown and estimated or inferred from experimental data. Once both the minimum charge and minimum current conditions are met, film thickness increases as

$$\frac{dh}{dt} = C_v\, j_n \text{ for } Q > Q_{min} \text{ and } j > j_{min}. \tag{4.8}$$

Additional model parameters have been estimated from experimental data. These parameters are therefore fixed to the values presented in Table 4.1.

#### 4.1.1.2 Computational Formulation

Two main types of experiments are considered in the computational setup: voltage ramp (VR) and constant current (CC). In VR, the voltage is increased linearly with time at a

Table 4.1: Summary of baseline model parameters.

| Name | Symbol | Value | Unit |
|---|---|---|---|
| Bath conductivity | $\sigma_{\text{bath}}$ | 0.14 | $S/m$ |
| Initial resistance at cathode | $R_0$ | 0.5 | $\Omega \cdot m^2$ |

rate of $V_R$ such that $\phi(t, x = L) = V(t, x = L) = V_R t$. Electric potential is denoted $\phi$ or $V$ interchangeably in this work. In CC, the current density at the anode is held constant. Experimentally, the current can only be held constant until some maximum voltage $V_{max}$ determined by the available equipment. Numerically, once the maximum voltage is reached in CC, $V_{max}$ is enforced instead of constant current.

Eqs. 4.1 through 4.3 are the Poisson equation with Robin boundary condition on the film / bath interface. This model is imposed for both experiments, but the boundary condition at the anode is different for each. Both experiment types are modeled by

$$\sigma_{\text{bath}} \frac{\partial^2 \phi}{\partial x^2} = 0 \qquad \text{in the bath} \tag{4.9}$$

$$\phi - R_{\text{film}} \sigma_{\text{bath}} \frac{\partial \phi}{\partial x} = 0 \qquad \text{at the interface film-bath,} \tag{4.10}$$

additionally with VR having an anode boundary condition

$$\phi_{\text{anode}}(t) = \phi_{t=0} + \phi_{\text{ramp}}(t), \tag{4.11}$$

and CC having an anode boundary condition

$$\sigma_{\text{bath}} \frac{\partial \phi}{\partial x} = j_0. \tag{4.12}$$

These equations offer an analytic relationship between film resistance and current in 1D for VR:

$$j(t) = \frac{\sigma V(t, x = L)}{\sigma R_{film}(t) + L}. \tag{4.13}$$

As $V(t, x = L)$ is set as the boundary condition, we can solve only the resistance dynamics in Eq. 4.5 and compute $j(t)$ from $V$ and $R_{film}$. For CC, the current is set by the experimental conditions as $j_0$, and the voltage at the anode can be computed by

$$\frac{\partial V}{\partial t} = \frac{\partial R_{film}}{\partial t} j_0 \text{ for } V < V_{max}, \tag{4.14}$$

where $V_{max}$ is the maximum experimental voltage. If $V \geq V_{max}$ in CC, the relationship

between voltage, current, and resistance is given by Eq. 4.13.

To simulate the baseline dynamics, any black box ODE solver can be implemented to solve Eq. 4.5 forward in time, computing the time evolution of film resistance. Current, voltage, and charge are given as boundary conditions, computed analytically from Eq. 4.13, or solved along with resistance dynamics using Eq. 4.14.

#### 4.1.1.3 Experimental Setup and Data

Experimental data is acquired from a laboratory setup that approximates the computational setting. In the experiment, a 16.0 $cm^2$ square anode and cathode are placed at the ends of a long e-coat bath and connected to a power source. A voltage is applied across the anode and the cathode according to either the VR or CC setup. During the course of each experiment, the voltage, current, and film resistance are all measured at a frequency of 10 Hz. For some experiments, thickness measurements are obtained by setting the voltage to zero at some time and measuring the thickness of the film at that time. Note that measuring the thickness terminates the experiment. A depiction of the experimental setup is given in Fig. 4.2

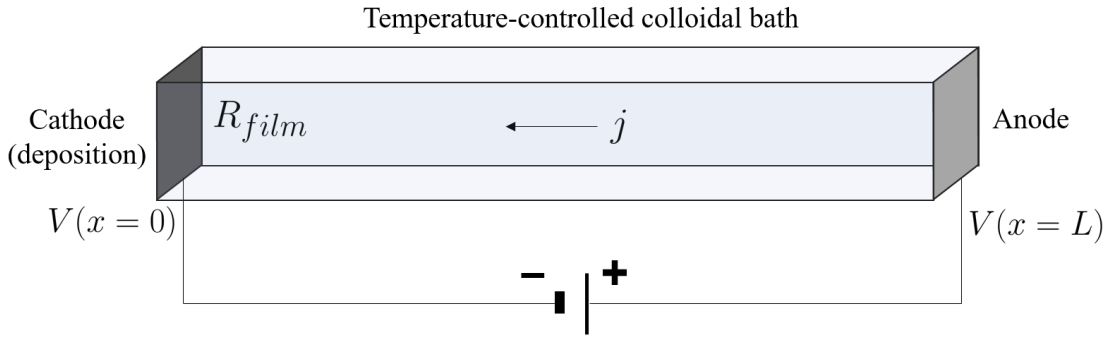



Figure 4.2: Experimental setup

Experiments are performed for both VR and CC at different experimental conditions, for a total of six configurations (see Table 4.2 for an overview). Multiple trials are repeated for each configuration, where each trial is performed to a different final time in obtaining its thickness measurement. The experimental data is collected in $\mathcal{D} = \{\{j\}_i, \{R\}_i\}_{i=1}^6$ where $\{j\}_i, \{R\}_i$ respectively represent the sets of current and resistance measurements for each of the six configurations. We do not use voltage data as they can be computed analytically from resistance and current. The data used during inference and learning is truncated for each experiment type $i$ to time $t_i$ such that data has been gathered for at least 3 trials at all times $t \leq t_i$. This is done to provide more accurate estimates of the variance during likelihood computation. An example of current measurements $\{j\}_1$ from the 13 trials under configuration VR, $V_R = 1V$ is shown in Fig 4.3.

| $i$ | Experiment | $n_i$ (# trials) | $t_i$ (s) | Total Data Points $(j, R, V)$ |
|---|---|---|---|---|
| 1 | VR, $V_R = 1.0$ V/s | 13 | 239 | 31,870 |
| 2 | VR, $V_R = 0.5$ V/s | 13 | 477 | 63,620 |
| 3 | VR, $V_R = 0.125$ V/s | 12 | 639 | 85,178 |
| 4 | CC, $j_0 = 10.0$ mA | 10 | 80 | 10,420 |
| 5 | CC, $j_0 = 7.5$ mA | 10 | 160 | 20,820 |
| 6 | CC, $j_0 = 5.0$ mA | 10 | 240 | 31,216 |

Table 4.2: Descriptions of the experimental data for each of the six configurations.

## 4.2 Parameter Inference With Experimental Data

In the baseline e-coat model, the unknown parameters of interest are $\theta = \{j_{min}, C_v, Q_{min}\}$. The initial goal is to infer (Section 1.5.1.1) these parameters given the experimental data described in Sec. 4.1.1.3 and the baseline model as the dynamics. The data random variable $Y$ consists of both current $j$ and film resistance $R_{film}$. We assume a measurement model of

$$y(\theta, t; \eta) = G(\theta, t; \eta) + \epsilon(t, \eta) \tag{4.15}$$

where $G(\theta, t; \eta) = \{R_{film}(\theta, t, \eta), j(\theta, t, \eta)\}$, computed by simulating the baseline model, and is a deterministic function of $\theta$, $t$, and the experimental configuration parameters $\eta$. The measurement noise $\epsilon(t, \eta) \sim \mathcal{N}(0, \Sigma(t, \eta))$ is a function of time and experimental configuration parameters, and the covariance matrix $\Sigma(t, \eta)$ is estimated from experimental data.

Considering the experimental data as $\mathcal{D}$ and baseline model as $\mathcal{M}$, the posterior we seek to approximate is given by Bayes' rule as

$$p(\theta|\mathcal{D}, \mathcal{M}) = \frac{p(\mathcal{D}|\theta, \mathcal{M})p(\theta|\mathcal{M})}{p(\mathcal{D}|\mathcal{M})} \, . \tag{4.16}$$

The integration involved in directly computing $p(\mathcal{D}|\mathcal{M}) = \int_\theta p(\mathcal{D}, \theta|\mathcal{M})d\theta$ (the "brute force" approach) is expensive to evaluate, particularly with the large amount of data in our dataset. Therefore, variational inference methods have are used in this work to approximate the posterior. Additionally, computing the posterior directly does not on its own necessarily allow samples to be easily drawn from the posterior. The brute force method of directly computing the posterior is referred to as the gridding approach hereafter due to computing the posterior on a discretized grid in the parameter space. Other methods investigated are all forms of variational inference in which the inference problem is transformed to an optimization problem [96, 157, 64, 62]. Truncated normal distributions are used for the prior

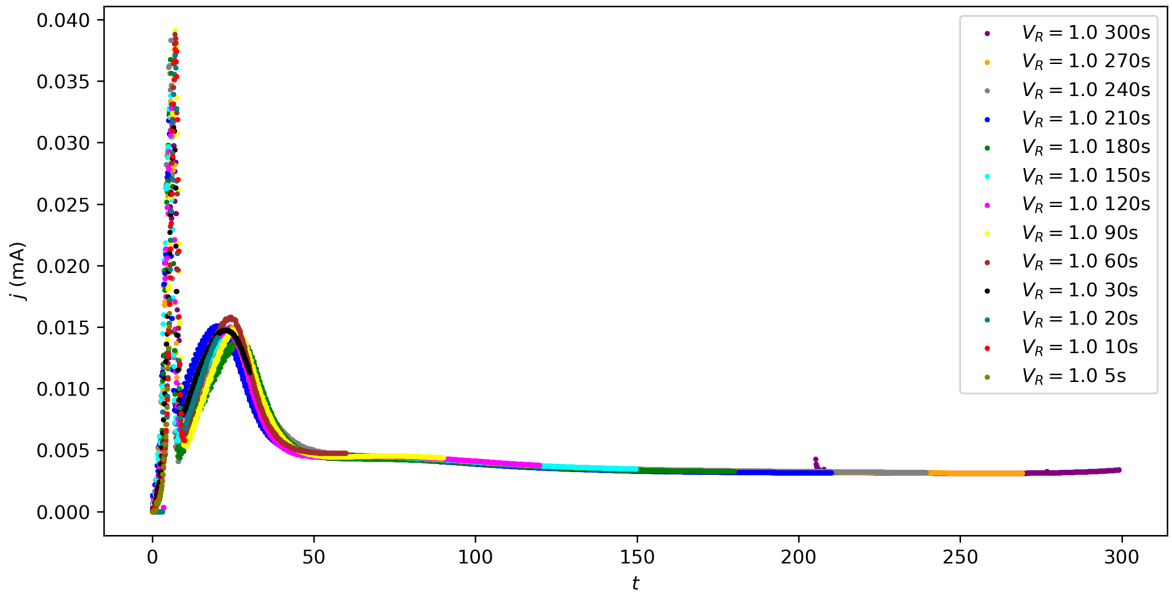



Figure 4.3: Visualization of the $\{j\}_1$ experimental data (all 13 trials) for configuration VR = 1.0. Each trial ends at a different time, and data is sampled at a rate of 10 Hz.

to ensure positive support while still enabling the specification of the mean and standard deviation of the prior for each parameter.

### 4.2.1 Likelihood

Any inference methods discussed in this work require the computation of the (log) likelihood. Given the assumed measurement model in Eq. 4.15, the likelihood $p(\mathcal{D}|\theta, \mathcal{M})$ is a Gaussian distribution with mean $G(\theta, t; \eta)$ (the output of the baseline model) and covariance matrix $\Sigma(t, \eta)$. We assume that all experiments, trials, and samples within each trial are independent. While the later assumption may not be strictly true, it significantly decreases the computational cost of computing the log likelihood due to a diagonal covariance matrix. The log-likelihood is therefore given by

$$\log p(\mathcal{D}|\theta, \mathcal{M}) = -\frac{1}{2}\sum_{i=1}^{6}\sum_{l=1}^{n_i}\sum_{r=1}^{10t_{l,i}}(G(\theta, 0.1r; \eta_i) - \mathcal{D}_{i,l,r})^T\Sigma(0.1r, \eta_i)(G(\theta, 0.1r; \eta_i) - \mathcal{D}_{i,l,r}) + Z \ , \tag{4.17}$$

where the time is given by $0.1r$ due to the constant sampling rate of 10Hz, and $\mathcal{D}_{i,l,r}$ denotes the 2D vector $[\{j\}_i^{(l)}(0.1r), \{R\}_i^{(l)}(0.1r)]^T$, the resistance and current for experiment $i$ in trial $l$ at time $0.1r$. Additionally, the matrix $\Sigma(0.1r, \eta_i)$ is diagonal and is computed for

experiment $i$ at time $0.1r$ by computing the variance of current and resistance data over all trials which contain data at time $0.1r$. The time $t_{l,i}$ denotes the final experiment time for trial $l$ of experimental configuration $i$. Finally, $Z$ is the negative log of the likelihood normalization constant which does not depend on $\theta$.

### 4.2.2 Gridding approach

Computing the Bayesian posterior is typically prohibitively expensive in practice. For our application, simulating the baseline model is the dominant bottleneck in terms of time complexity. Evaluating the forward model as little as possible will provide the greatest benefit for the efficiency of the inference process. More efficient methods of approximating the posterior are thus investigated to perform parameter inference. Each of these results is compared to the Bayesian posterior using a gridding approach, which is considered the 'true' posterior.

We employ a simple gridding approach to compute the Bayesian posterior in which the parameter space is discretized into a $d$-dimensional grid of uniform spacing. This grid is then sequentially refined with higher resolution near the maximum a posteriori (MAP) and any modes of the Bayesian posterior.

It is often computationally more stable to compute the log-distributions first, rather than the distribution directly. Considering a single grid point $\theta_i$, we first compute the quantity

$$\log p(\theta_i|y) + \log p(y) = \log p(\theta_i) + \log p(y|\theta_i) \tag{4.18}$$

at all grid points in the parameter space. Note that we ignore the *evidence* term $p(y)$ in the gridding approach until Eq. 4.18 is computed at each point. It is assumed that the grid bounds in parameter space are sufficiently large to capture the most significant aspects of the distribution.

The final distribution is computed at each point in the grid by normalizing the quantity computed in Eq. 4.18 using

$$p(\theta_i|y) = \exp(\log p(\theta_i) + \log p(y|\theta_i))/Z \ ,$$

where the normalization constant is given by $Z = \int_\Theta \exp(\log p(\theta) + \log p(y|\theta))d\theta$ and approximated using some numerical integration scheme based on the selected grid.

We note that the gridding approach can provide a reasonable approximation to the posterior, but it does not provide a straightforward method of sampling from this posterior.

### 4.2.3 Gradients Through ODE Solve

Our subsequent analysis will employ inference and ML methods that require gradient information. Therefore, ensuring that the ODE solve is differentiable and having an efficient method of computing gradients is critical. To obtain gradients of the baseline model ODE forward solve with respect to model parameters, we employ an adjoint-based method known as NeuralODE [26] which can compute gradients using any black-box ODE solver. This method computes gradients through the ODE forward solve very efficiently relative to automatic differentiation methods when solving ODEs. It is also available as part of existing ML libraries such as Pytorch [158], which facilitates its easy integration with other ML tools and frameworks.

Propagating gradients through the deposition onset criteria requires extra care. Before film deposition begins, the resistance and film thickness do not increase. Therefore, the following dynamical system is solved:

$$\frac{dh}{dt} = 0, \quad \frac{dR_{film}}{dt} = 0, \tag{4.19}$$

with $V$, $j$, and $Q$ computed analytically according to the experiment type. When both $j > j_{min}$ and $Q > Q_{min}$, deposition begins and the model dynamics instantaneously switch to

$$\frac{dh}{dt} = C_v j, \quad \frac{dR_{film}}{dt} = \rho(j)\frac{dh}{dt}. \tag{4.20}$$

As predicting the time of deposition onset is critical for predicting film growth, gradients of the output with respect to the onset condition parameters $j_{min}$ and $Q_{min}$ must be computed. However, the instantaneous change in model constitutes an in-place operation [158] and must be treated separately. We use ideas from an extension to NeuralODE which adds the ability the compute gradients through instantaneous event handling [131]. Adding an additional *switch state* $\xi$ to our model, we computationally solve the following equations instead of Eqs. 4.19 and 4.20:

$$\frac{dh}{dt} = \xi C_v j, \quad \frac{dR_{film}}{dt} = \xi\rho(j)\frac{dh}{dt}. \tag{4.21}$$

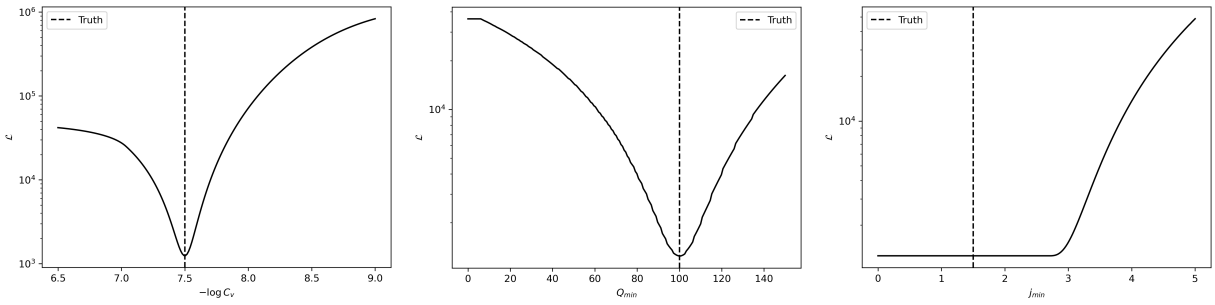
The initial switch state $\xi(t = 0) = 0$ is switched to $\xi(t = t_e) = 1$ at the event time, defined as the time $t_e$ such that both of the deposition onset criteria are met. This implementation detail allows computing the gradient of the forward solve with respect to the event time, and in turn with respect to the onset criteria parameters $j_{min}$ and $Q_{min}$, using the framework from Ref. [131].

### 4.2.4 Parameter identifiability of baseline model

The process of parameter inference on the baseline model provides some insight into the shortcomings of the model, and results in proposed updates to the baseline model to address some of these shortcomings.

One of such shortcomings is parameter unidentifiability of the deposition onset criteria parameters, $Q_{min}$ and $j_{min}$. The nature of this double criteria for the onset of deposition creates situations in which one parameter or the other may not be identifiable. In this case, the definition of *identifiability* is that of structural identifiability [159]. A model parameter is structurally identifiable from data if, for any distinct parameter values, the model output is different for the same input. Mathematically, given two parameter values $\theta_1$ and $\theta_2$ where $\theta_1 \neq \theta_2$, then $f(x, \theta_1) \neq f(x, \theta_2)$ for a model with output $f$ and input $x$.

Two of the baseline model parameters are unidentifiable from data, but with a complex dependency on the specific experiment from which the data was gathered. For example, consider the case in which the time $t_j$ at which $j > j_{min}$ is smaller than the time $t_Q$ at which $Q > Q_{min}$, such that $t_j < t_Q$. Changing $j_{min}$ over some range will thus not have any effect on the baseline model output because *both* conditions $j > j_{min}$ and $Q > Q_{min}$ must be satisfied for deposition to begin. In this case, only $Q_{min}$ controls the deposition onset time and data may be uninformative about $j_{min}$. However, the opposite can occur and there also exist cases in which data may be uninformative about $Q_{min}$.




(a) $-\log C_v$ (b) $Q_{min}$ (c) $j_{min}$

Figure 4.4: Negative log-likelihoods computed from simulated data on a voltage ramp experiment using the baseline model with experimental conditions $V_R = 0.125$, $\sigma = 0.14$, $-\log C_v = 8.5$, $Q_{min} = 100.0$, and $j_{min} = 1.5$.

To illustrate this identifiability issue more clearly, we generate artificial data from the baseline forward model by taking the 'true' parameters to be $[-\log C_v, Q_{min}, j_{min}] = [7.0, 150.0, 1.0]$. Other experimental parameters are given in Table 4.1. A total of 10 samples are obtained by simulating a voltage ramp (VR) experiment with a small amount of noise

added. Figure 4.4 illustrates the negative log-likelihood of this data for each parameter to be inferred assuming that the other two parameters are fixed to their 'true' values. For the first two parameters, a minimum exists at the 'true' values, indicating that these parameters can be accurately inferred from the simulated data. However, the negative log-likelihood of the minimum current threshold parameter $j_{min}$ is flat over some region, indicating that the model is not effected by changes in $j_{min}$ over this region. If the 'true' parameter value lies anywhere in the region in which the derivative of the log-likelihood is zero, that parameter is unidentifiable on that region. This indicates that the model output is identical for all values of $j_{min}$ on this region, rendering the parameter unidentifiable.

For voltage ramp experiments, there exists a set of boundaries in parameter space for which either $j_{min}$ or $Q_{min}$ will be unidentifiable, or both will be identifiable in the baseline model. These boundaries change with the conditions of the voltage ramp experiment; gathering data from additional experiments could provide information on a parameter that is not informed by data from a different experiment.

Consider a case in which data exists such that $t_Q < t_j$. This means that $t_j$ will control the deposition onset time. As $j(t) > 0 \ \forall \ t > 0$, then $Q(t > t_j) > Q(t_j) > Q_{min}$. Thus changing $Q_{min}$ on a range $0 \leq Q_{min} < Q(t_j)$ will have no effect on the output of the baseline model and the data will be uninformative about $Q_{min}$ on this range. Next consider a case in which data exists such that $t_j < t_Q$. Now $t_Q$ will control the deposition onset time. However, $j(t > t_Q) > j(t_Q)$ is *not* guaranteed, which can be easily seen by taking the time derivative of Eq. 4.13. If the voltage ramp $V_R < \rho(j)C_v j$, then $dj/dt < 0$ and it is not guaranteed that $j(t > t_Q) > j(t_Q)$. If $j(t > t_Q) < j(t_Q)$, then it is possible for $j(t > t_Q) < j_{min}$ and deposition stops, followed by an increase in current, restarting deposition, and the cycle repeats. Ultimately this indicates that the parameter $j_{min}$ will have an influence on the baseline model output, and data will be informative about $j_{min}$. Thus, if $t_j < t_Q$, data may or may not be informative about the parameter $j_{min}$, depending on the experimental conditions.

Two identifiability regions corresponding to different experiments are computed empirically and illustrated in Figure 4.5. This regions are computed by simulating the baseline model according to the experimental conditions for a set of discretized points in the $j_{min}, Q_{min}$ space. For each simulation, identifiability is checked by checking $t_Q$ and $t_j$. Purple regions indicate that data from the experiment is not informative about $j_{min}$ if the true value of $j_{min}$ and $Q_{min}$ lie in the region. In other words, $\partial\mathcal{L}/\partial j_{min} = 0$ on the purple region. Cyan regions indicate that data from the experiment is not informative about $Q_{min}$ if the true values of $j_{min}$ and $Q_{min}$ lie in the region, or $\partial\mathcal{L}/\partial Q_{min} = 0$. Yellow regions indicate that the experiment can inform both $j_{min}$ and $Q_{min}$. The boundary of the cyan region can be

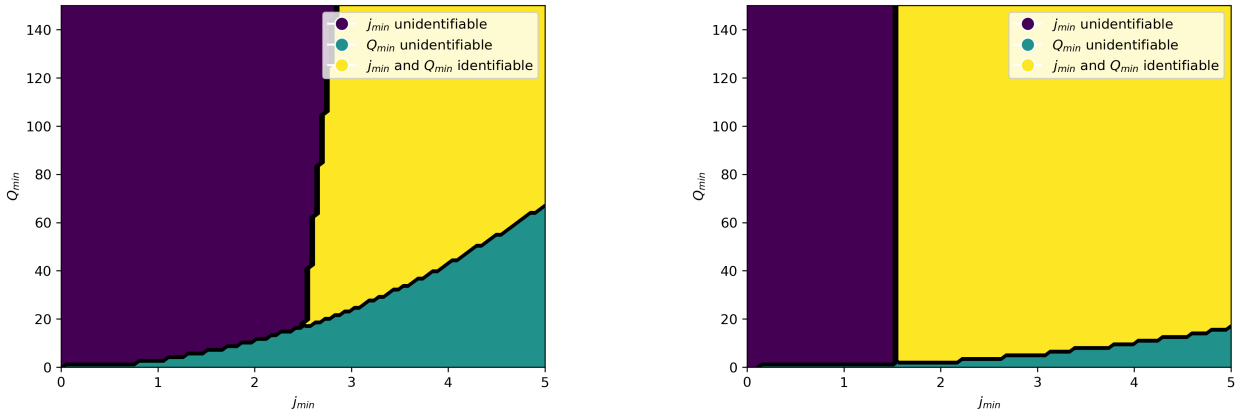
computed analytically, but the other is nontrivial and computed empirically. The times at which the deposition onset criteria are met in the voltage ramp experiment for the baseline model are given by

$$t_j = \frac{2Q_{min}}{\sigma V_R}(\sigma R_0 + L), \quad t_Q = \left[\frac{2j_{min}}{\sigma V_R}(\sigma R_0 + L)\right]^{1/2}. \tag{4.22}$$

Setting Eqs. 4.22 equal, we obtain a closed form solution to the $Q_{min}$ identifiability boundary as

$$j_{min} = \left[\frac{2Q_{min}\sigma V_R}{\sigma R_0 + L}\right]^{1/2}, \tag{4.23}$$

which has been validated against the empirically computed boundaries shown in Fig. 4.5. We note that the log-likelihoods in Fig. 4.4b and 4.4c correspond to the identifiability plane in Fig 4.5a. The log-likelihoods of $Q_{min}$ and $j_{min}$ have first derivative zero over some region as predicted by the identifiability boundaries. The true values of $j_{min}$ and $Q_{min}$ are such that $j_{min}$ is not informed by the simulated experimental data.




(a) VR experiment, $V_R = 0.125$, $\sigma = 0.14$, $-\log C_v = 8.5$

(b) VR experiment, $V_R = 0.5$, $\sigma = 0.14$, $-\log C_v = 7.5$

Figure 4.5: Identifiability regions of the baseline model for two different experimental conditions. The log-likelihood on the simulated experimental data will be constant in the purple and cyan regions, indicating that little information is gained about $j_{min}$ if the true value lies in the purple region or $Q_{min}$ if the true value lies in the cyan region. Note: the 'stepping' behavior observed in the identifiability boundaries here are a product of discretizing the $j_{min}$ and $Q_{min}$ domains, but the boundaries are in fact smooth.

If the experimental data does not inform one of the parameters, then that parameter does not influence the output of our baseline model, and it cannot be accurately inferred. However, as shown by Fig. 4.5, the identifiability boundaries change with the type of experiment. This

behaviour could cause very poor prediction results. Suppose none of the experimental data is informative about $j_{min}$, and inference is performed on the model parameters. Using the model in prediction could result in poor performance especially if predictions are made for an experimental condition in which $j_{min}$ does influence the output of the model.

It is also necessary to infer robust posteriors in this case. Suppose Gaussian variational inference is selected as the inference method. A Gaussian variational posterior will be computed for each of the parameters to be inferred, providing a unimodal distribution for each. This can be misleading about the Bayesian posterior distribution of the parameter and result in poor uncertainty quantification during prediction.

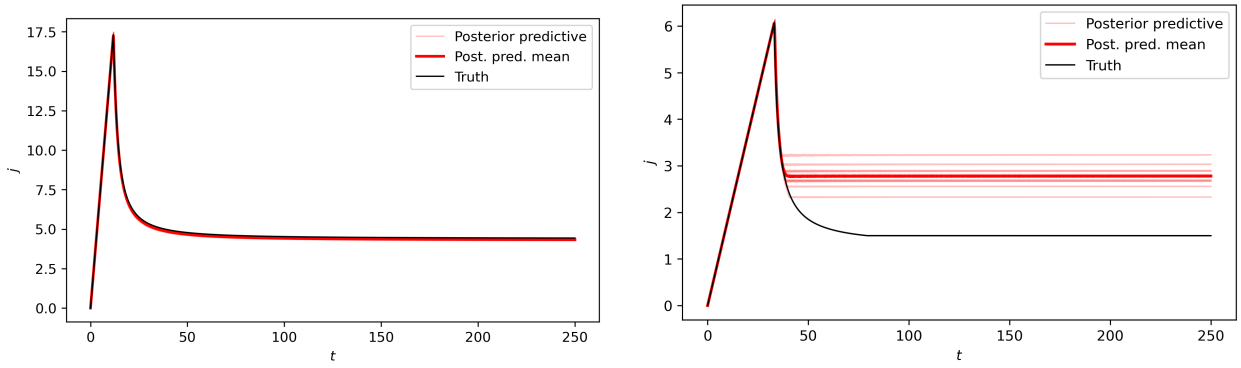
We explore two approaches towards the aim of improving model predictions by improving the form of the baseline model. First, we update the model based on insight from the shortcomings observed during inference, described in Sec. 4.3.1. We then pursue a different approach in which model augmentations are learning using machine learning tools to augment the baseline model with previously unmodeled dynamics, discussed in Sec. 4.3.2.

### 4.2.5 Inference on baseline model

In this section, we demonstrate inference results using Gaussian variational inference on the baseline model. The purpose of this experiment is to illustrate the shortcomings of the model form itself and the exact reason why the identifiability issues described in Section 4.2.4 are problematic. Inference is performed using simulated data from the baseline model, and we demonstrate that the posterior predictive results in good prediction performance for some experimental conditions, but poor performance on others.

Data is generated by simulating the baseline model 10 times up to a final time $T = 250s$ for a voltage ramp experiment with $V_R = 1.0$, $\log C_v = -8$, $j_{min} = 1.5$, $\sigma = 0.14$, and $Q_{min} = 100$, and add Gaussian random noise $\epsilon \sim \mathcal{N}(0, \eta^2)$, where $\eta = 0.01$, at each time step to simulate measurement noise. Only data from current measurements is considered in the inference process. We then perform Gaussian VI for parameters $\log C_v$, $j_{min}$, and $Q_{min}$ on the baseline model.

After learning the variational posterior $q_\phi$, samples are drawn and used to simulate the baseline model to obtain samples from the posterior predictive. Ten of these samples along with the mean are shown in Fig. 4.6a. For the experimental configuration on which the data is generated, accurate and low variance prediction is observed. However, we then use the variational posterior distribution to predict on a voltage ramp experiment in which all parameters are the same except the voltage ramp $V_R = 0.125$. Fig. 4.6b shows that prediction performance is very poor for this experiment.




(a) Voltage ramp experiment data, $V_R = 1.0$, $\sigma = 0.14$, $-\log C_v = 8$, $j_{min} = 1.5$, $Q_{min} = 100$

(b) VR experiment prediction, $V_R = 0.125$, $\sigma = 0.14$, $-\log C_v = 8$, $j_{min} = 1.5$, $Q_{min} = 100$

Figure 4.6: Posterior predictive results after performing Gaussian VI on data from a simulated voltage ramp experiment. The posterior predictive results in accurate simulations on the data (a), but poor predictions for other experiments (b). This is caused by unidentifiable $j_{min}$ in the data.

The reason for this stems from the identifiability issues previously discussed in Sec 4.2.4. In our experiment, we assume that the 'true' value of $j_{min}$ is 1.5. However, in the experiment which we gather data from, $j_{min}$ is unidentifiable in the baseline model. Thus the inference results in a variational posterior distribution for $j_{min}$ of $q(j_{min}|\mathcal{D}) = \mathcal{N}(2.66, 0.01)$, which is a low variance but poor estimate of the true parameter value. This posterior distribution is then used for prediction in an experiment in which $j_{min}$ *does* have an effect on model output, and because the posterior is not accurate, prediction is also inaccurate.

On top of identifiability issues potentially resulting in poor prediction performance, the parameter $Q_{min}$ is inconsistent accross different experiment types. To illustrate this issue, Gaussian VI is performed on the parameters $C_v$, $j_{min}$, and $Q_{min}$ twice - first using real experimental data from only the voltage ramp experiments and again using only data from the constant current experiments. The maximum a posteriori (MAP) of the variational posterior for each distribution are very different - in particular for $Q_{min}$. Using only voltage ramp experimental data, the MAP of the variational posterior is $Q_{min} \approx 261$; however, using only constant current experimental data, the MAP is located at $Q_{min} \approx 101$. This is indicative of $Q_{min}$ being problematic in allowing the baseline model to accurately predict experimental data with different boundary conditions. Our natural conclusion is that the model is incorrect, and a root cause may lie in a constant $Q_{min}$. Rather, $Q_{min}$ may be a function of the type of experiment being performed rather than a constant to be inferred. In Sec 4.3.1, physically intuitive model updates are introduced with the aim of alleviating the identifiability concerns of the baseline model and improving the modeling of the minimum

charge criterion.

## 4.3 Model updates

The baseline model was shown to exhibit identifiability as well as generalization deficiencies in Sec. 4.2. In this section, we aim to improve the prediction accuracy and generalizability of the baseline model. First, modifications are made based on the inference results of the baseline model to aid in improving parameter identifiability and minimum charge criterion modeling.

An alternative approach to improving the baseline model is also investigated in which machine-learning augmentations are introduced to model system dynamics which are absent from the baseline model. These augmentations are introduced with an emphasis on interpretability of the augmented model while allowing for greater flexibility in model expressiveness.

### 4.3.1 Inference-informed modifications

Based on the inference experiments of Sec. 4.2, model updates are proposed to alleviate the observed inadequacies during prediction. The issue of identifiability arises in the baseline model due to the double conditional statement that film deposition begins only if $j > j_{min}$ *and* $Q > Q_{min}$. This conditional statement, in particular $j > j_{min}$, creates non-physical, discontinuous behavior of the model. The film growth rate given by Eq. 4.19 is exactly zero until the conditional statement is true. Assuming that $j_{min} > 0$, the film growth rate will instantaneously increase, and a sharp discontinuity in the film thickness growth occurs. The model also does not accurately model the cases in which both onset criteria are met, but the minimum current condition is no longer met at a later time. This behavior is one of the reasons that prediction is observed to be inaccurate in Fig. 4.6b. We therefore propose a model update to create a model in which the dynamics are continuous which also allows for film dissolution by replacing the film thickness dynamics in Eq. 4.20 with

$$\frac{dh}{dt} = C_v(j_n - j_{min}) \text{ for } Q > Q_{min}, \text{ s.t. } h \geqslant 0 \, . \tag{4.24}$$

This also gives the benefit of $j_{min}$ being identifiable for all experimental configurations.

With $j_{min}$ now identifiable, the expressiveness of the parameter is investigated to improve generalizability. Assuming that the minimum charge criterion is related to the concentration of $OH^-$ present in the bath, B.3 illustrates that $Q_{min}$ is not constant accross experiment types, but depends on a different constant $K$. It is shown in B.3 that $Q_{min}$ is a function

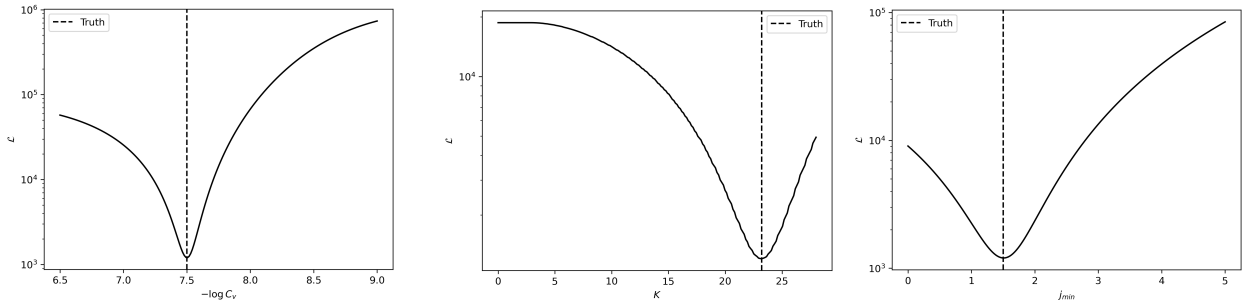
of this constant, and the function differs between the voltage ramp and constant current experiments. For VR (Eq. 4.25) and CC (Eq. 4.26) experiments, this function is given by

$$Q_{min} = \left(\frac{81}{128\beta}\right)^{1/3} K^{4/3} \qquad (4.25) \qquad\qquad Q_{min} = \frac{K^2}{j_0}, \qquad (4.26)$$

where $\beta = \sigma V_R/(\sigma R_0 + L)$

The updated film thickness dynamics of Eq. 4.24 along with introducing the parameter $K$ used to compute the minimum charge criterion constitute an updated model which we dub the 'inference-informed' model.

Performing the same exercise to visualize the negative log-likelihood as in Fig. 4.4, we visualize the negative log-likelihood of the inference-informed model on artificial data to illustrate that all parameters are now identifiable. In this case, we simulate the data from the same experimental configuration as Fig. 4.4, which is a VR experiment with $V_R = 0.125$, $\sigma = 0.14$, $-\log C_v = 8.5$, $Q_{min} = 100.0$, and $j_{min} = 1.0$. However, the parameter $K$ is used instead of $Q_{min}$; thus the value of $K$ is found using the relationship in Eq. 4.25, obtaining $K = 23.2$. The negative log-likelihoods of $-\log C_v$, $K$, and $j_{min}$ are illustrated in Fig. 4.7, and all parameters exhibit a global minimum at the true parameter values, indicating that all are now identifiable.




(a) $-\log C_v$ (b) $Q_{min}$ (c) $j_{min}$

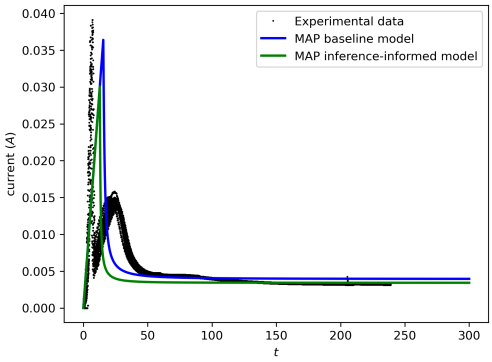
Figure 4.7: Negative log-likelihoods computed from simulated data on a voltage ramp experiment using the inference-informed model with experimental conditions $V_R = 0.125$, $\sigma = 0.14$, $-\log C_v = 8.5$, $Q_{min} = 100.0$ ($K = 23.2$), and $j_{min} = 1.5$.

#### 4.3.1.1 Model Comparisons

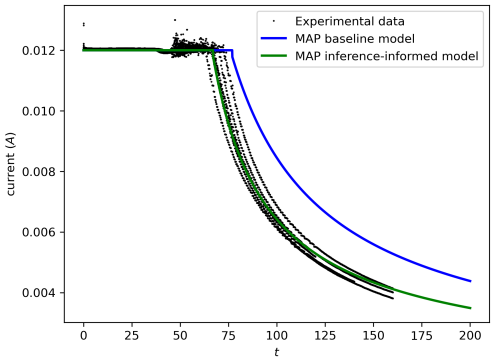
The baseline model and the inference-informed model are directly compared by performing Bayesian inference using the gridding approach discussed in Sec. 4.2.2. We then predict for each model based on the maximum a posteriori and compute the negative log-likelihood of

the data. Lower values of the negative log-likelihood at the MAP correspond to a model which better fits the experimental data.

The MAP of the approximated posterior using the gridding approach assuming the baseline model is located at $\theta_{MAP} = [-\log C_v, Q_{min}, j_{min}] = [7.58, 173.5, 0.0]$ with a negative log-likelihood at the MAP of $3.35 \times 10^7$. Assuming the updated model, the MAP is located at $\theta_{MAP} = [-\log C_v, K, j_{min}] = [7.32, 44.2, 0.63]$ while the negative log-likelihood is $2.65 \times 10^7$. This suggests that the inference-informed model performs better than the baseline model by having the potential to represent the experimental data more accurately.



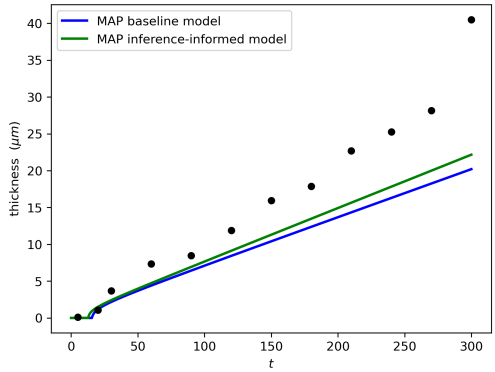

(a) Voltage ramp experiment prediction, $V_R = 1.0V/s$



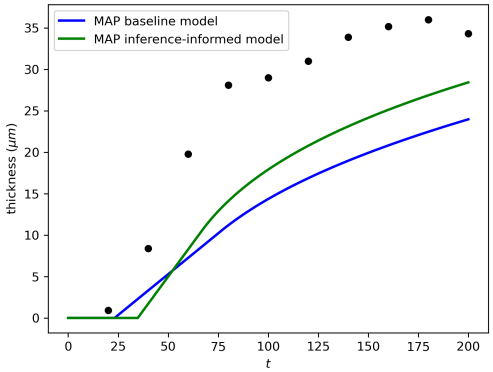

(b) Constant current experiment prediction, $j_0 = 7.5mA$

Figure 4.8: Comparisons between current prediction on the baseline model and inference-informed model at the MAP for each on (a) voltage ramp experiment with $V_R = 1.0V/s$ and (b) constant current experiment with $j_0 = 7.5mA$.

Each model prediction at the map is compared to the experimental data in Fig 4.8. The current predictions are shown for 2 experiments: VR with $V_R = 1.0V/s$ and CC with $j_0 = 7.5mA$. Predictions for all experiments on current, resistance, and thickness data are included in B.4. Prediction of the inference-informed model for constant current experiments exhibits significant improvement over the baseline model, likely due to the improved parameterization of the minimum charge criterion $Q_{min}$. However, there is clearly some physical behavior which exists in the voltage ramp experiments which is not present in our model. For instance, there are two current 'peaks' in experimental data for voltage ramp experiments, but only one is possible in model predictions. Additionally, although the thickness prediction of the inference-informed model is closer to the experimental data than the baseline model, Fig. 4.9 illustrates that neither are particularly accurate. Ultimately, thickness prediction is the most important quantity of interest. We thus turn towards improving the model through augmenting the model forming, and learning the augmentations via a machine-learning based framework.




(a) Voltage ramp experiment prediction, $V_R = 1.0$ (film thickness)




(b) Constant current experiment prediction, $j_0 = 7.5mA$ (film thickness)

Figure 4.9: Comparisons between film thickness prediction on the baseline model and inference-informed model at the MAP for each.

### 4.3.2 Machine-learning augmentations

Modifying the model using insights from the inference process aid in understanding some shortcomings due to model form, but results in limited improvement to predictive performance. Additional physical behavior is missing from the model itself, which results in poor performance even when the optimal model parameters are identified. There are two main features absent from the baseline model: the presence of two peaks in the current dynamics for the voltage ramp experiments, and smooth behavior of those peaks. Unstable behavior in data for constant current experiments observed just before the drop in current is due to the current controller in the experiments, which we do not account for in our model.

To incorporate the two missing physical features in our model, we turn towards the power of machine learning with an emphasis on maintaining interpretability, designing ML augmentations from the perspective of potential root causes which may not be well understood or difficult to model. In particular, we augment the right hand side of the dynamics model with parameterized neural networks and train the augmentations using NeuralODE [26]. This can be used effectively to utilize adjoint-based gradient computation alongside standard machine learning pipelines to learn flexible augmentations in the right hand side of a dynamical system.

We first attempt to characterize the underlying physics of each of the two peaks present in the experimental current data for voltage ramp experiments. The second peak corresponds to the onset of deposition, which is previously modeled in the baseline and inference-informed models. As film deposition begins, resistance increases and current decreases. However, the baseline model assumes an instantaneous onset of film deposition which results in a

discontinuous and physically impossible film growth rate. Experimental data illustrates smooth transitions to film growth in current data, further supporting the need for additional modeling. This smooth transition is likely the result of portions of the surface being coated in a non-uniform manner, resulting in only a fraction of the material surface being coated for a small time. We propose an augmentation to the baseline model which has the additional benefit of modeling the deposition onset time without the need of threshold parameters. We propose multiplying the right hand side of Eq. 4.20 by a learnable term $g_\phi(V, Q)$ which varies smoothly between zero and one. This term is dubbed the 'coverage fraction model', and it is a function of voltage and charge only, representing the coverage fraction of the film on the material. It is found empirically that using both voltage and charge as inputs to the model results in the best predictive performance. It is a function of charge due to the dependence of the deposition onset time of the baseline model on charge, and it is a function of voltage due to a change in the time derivative of voltage across experiment types.

The first peak present in the current data for voltage ramp experiments is caused by a phenomenon ignored by the baseline model altogether. This peak may be caused by an oxygen-reduction reaction [160] which occurs at the anode as the voltage increases. The relationship between electric potential and current in redox reactions is described by the Butler-Volmer equation [161] of the form

$$j = j_0 \left[\exp\left(aV\right) - \exp(bV)\right] \ , \tag{4.27}$$

where the parameters $a$ and $b$ depend on many physically relevant parameters. However, the particular form is not relevant to the discussion here as we aim to learn this component of the model while keeping the general form to ensure that the model is physically-interpretable. We assume that the combined effect of this redox reaction results in a resistance 'source' such that $j = c_1 \exp(c_2 V)$, based on the form of Eq. 4.27.

After some critical point is reached (which we do not explicitly model, but likely corresponds to some minimum charge criterion), we assume that the $OH^-$ starts being diffused at some rate [160] which decreases the $OH^-$ concentration. As the redox reaction continues, we assume that the combined effect of these two reactions results in a different exponent such that $j = c_3 \exp(c_4 V)$, where $c_4 < 0$.

To augment the voltage ramp experiment model with this behavior of switching between two different exponents resulting from a combination of reactions, we use an exponential function in which the argument includes a hyperbolic tangent function which can vary smoothly

between two different values. We thus assume an augmented model of the form

$$j = \frac{\sigma V}{\sigma R_{film} + L} + c_1 \exp(c_2 V f_\theta(V, Q)), \quad f_\theta(V, Q) = \tanh(-\hat{f}_\theta(V, Q)) \,, \tag{4.28}$$

and $\hat{f}_\theta$ is a parameterized function such as a feed-forward neural network (FNN). The first term in Eq. 4.28 corresponds to the baseline model current in Eq. 4.11, but the second term is informed by the aforementioned physical processes while allowing flexibility in learning the particular dynamics. Using the tanh function allows a smooth transition between two different exponents, effectively switching between the two dominant reactions at the beginning of voltage ramp experiments. We have added additional parameters to the model to allow for more flexibility. This model augmentation is then added to the current in voltage ramp experiments to capture the behavior of the first peak.

The augmented model is finally expressed by

$$f_\theta(V, Q) = \tanh\left(-\hat{f}_\theta(V, Q)\right), \quad g_\phi(V, Q) = \frac{1}{1 + c_3 \exp\left(-c_4 \hat{g}_\phi(V, Q)\right)} \tag{4.29}$$

$$\frac{dR_{film}}{dt} = g_\phi(V)\rho(j) j C_v, \quad j = c_1(\exp(c_2 V f_\theta(V))) + \frac{\sigma V}{\sigma R_{film} + L} \,, \tag{4.30}$$

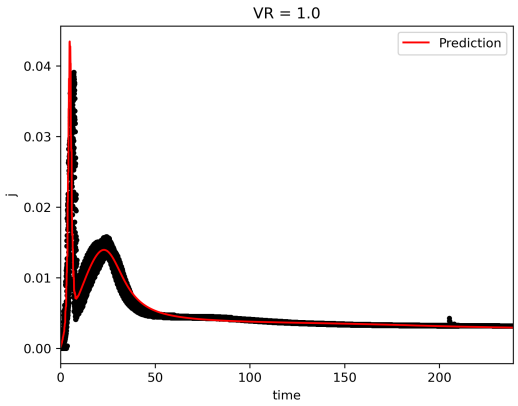
for VR experiments and

$$g_\phi(V, Q) = \frac{1}{1 + c_3 \exp\left(-c_4 \hat{g}_\phi(V, Q)\right)}, \quad \frac{dR_{film}}{dt} = g_\phi(V)\rho(j) j C_v, \tag{4.31}$$

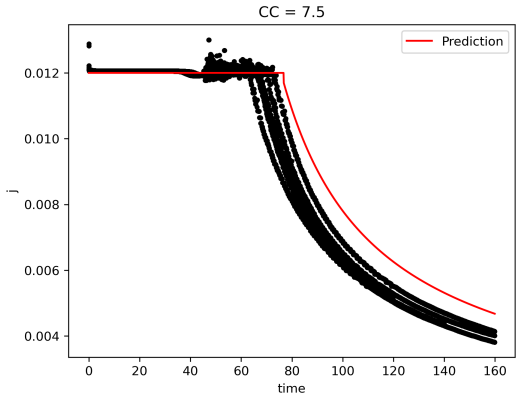
for CC experiments. Note that the relationship between voltage, current, and resistance are unchanged from the baseline model in the CC experiments even with the machine learning augmentations introduced to the model. The model augmentation functions $\hat{g}_\phi$ and $\hat{f}_\theta$ are parameterized by FNN's with 4 layers, 8 nodes per layer, and ReLU activation functions. These are learned along with the constants $c_1, c_2, c_3$, and $c_4$ using the NeuralODE framework using only current data from the 3 VR experiments. Thus, applying the model on the CC experiments is purely prediction as none of the data is seen during training. We learn the parameters $\theta, \phi, C_v$, $\sigma, c_1, c_2, c_3$, and $c_4$ in our experiments and show prediction results. Figure 4.10 illustrates the results of the final trained model, better capturing the 'double-peaked' nature of the current in experimental data for voltage ramp experiments. Here we show current prediction for only two experimental configurations, but current and resistance predictions for all other experimental configurations are provided in B.5.

Further, thickness predictions for all experimental configurations are illustrated in Fig. 4.11 along with predictions from the baseline and inference-informed models for compar-

ison. Film thickness predictions show an improvement over the baseline model in all cases except for a single experimental configuration: voltage ramp experiments with $V_R = 0.125$. We expect that this is due to some unmodeled behavior in the low voltage regime which is difficult to capture. However, in constant current experiments and larger voltage values in voltage ramp experiments, our augmentations provide significant improvement to model predictions.



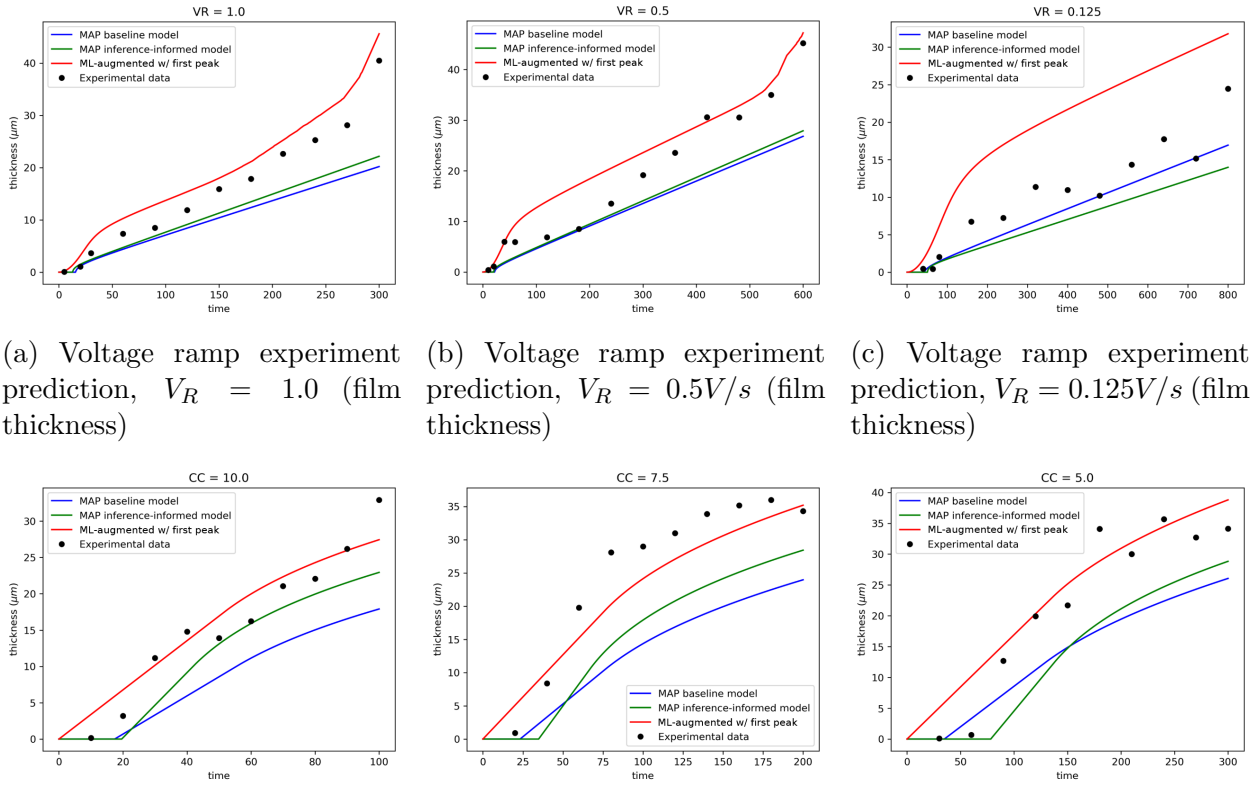

(a) Voltage ramp experiment prediction, $V_R = 1.0V/s$




(b) Constant current experiment prediction, $j_0 = 7.5mA$

Figure 4.10: Current prediction on the machine-learning augmented model trained with the first peak model.

Learning the first peak model function $f_\theta$ in the NeuralODE framework is significantly more expensive computationally than training only the coverage fraction model $g_\phi$ due to the large gradient change requiring a finely-discretized time step to adequately resolve. Additionally, the first peak model performs notably poorly for the voltage ramp experiment in the low voltage ramp regime. We thus remove the 'first peak' model and retrain only the augmentation function $g_\phi(V, Q)$ by masking out the experimental data corresponding to the first peak in current for voltage ramp experiments. This results in more efficient training and thickness prediction while providing quite similar predictions when the first peak model is included. These results show that modeling the dynamics of the first peak may unnecessarily decrease computational efficiency while still providing more accurate thickness prediction, which is ultimately the goal. Current predictions for two experiments (only voltage ramp and one constant current) are shown in Fig. 4.12, and thickness predictions are shown and compared to all other models presented in Fig. 4.13. These results again illustrate a significant improvement in thickness prediction without the need for including the more expensive first peak model, indicating that modeling such behavior may be largely unnecessary for




(a) Voltage ramp experiment prediction, $V_R = 1.0$ (film thickness)

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (film thickness)

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (film thickness)

(d) Constant resistance experiment prediction, $j_0 = 10.0mA$ (film thickness)

(e) Constant current experiment prediction, $j_0 = 7.5mA$ (film thickness)

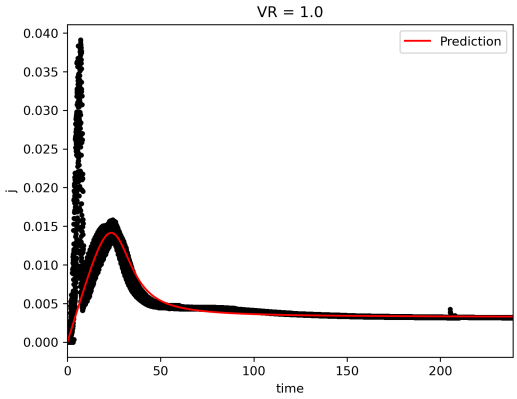
(f) Constant current experiment prediction, $j_0 = 5.0mA$ (film thickness)

Figure 4.11: Comparisons between film thickness prediction on the baseline model, inference-informed model, and ML-augmented model with first peak.

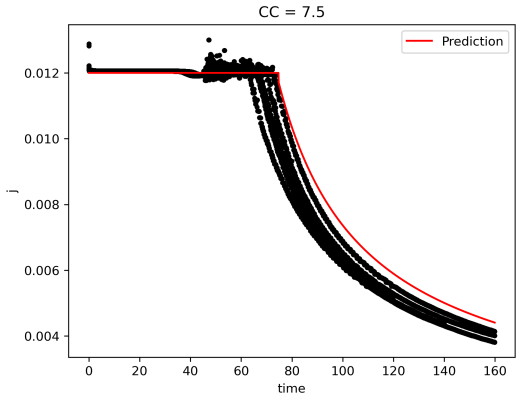
predicting thickness. However, prediction in the low voltage ramp regime is actually worse than with the first peak model included, further supporting our hypothesis that there exist additional unmodeled dynamics in such cases. Again, all other predictions for current and resistance using our model augmentations without the first peak model included are provided in Appendix B.6.

## 4.4 Summary

Throughout this chapter on enhancing dynamical system modeling through interpretable machine learning augmentations, we have presented an application which underscores the integration of adaptable yet interpretable machine learning-based model refinements. This process commences with the objective of parameter inference, which reveals significant limitations within the existing baseline model. These insights lead to the necessity for diverse




(a) Voltage ramp experiment prediction, $V_R = 1.0V/s$



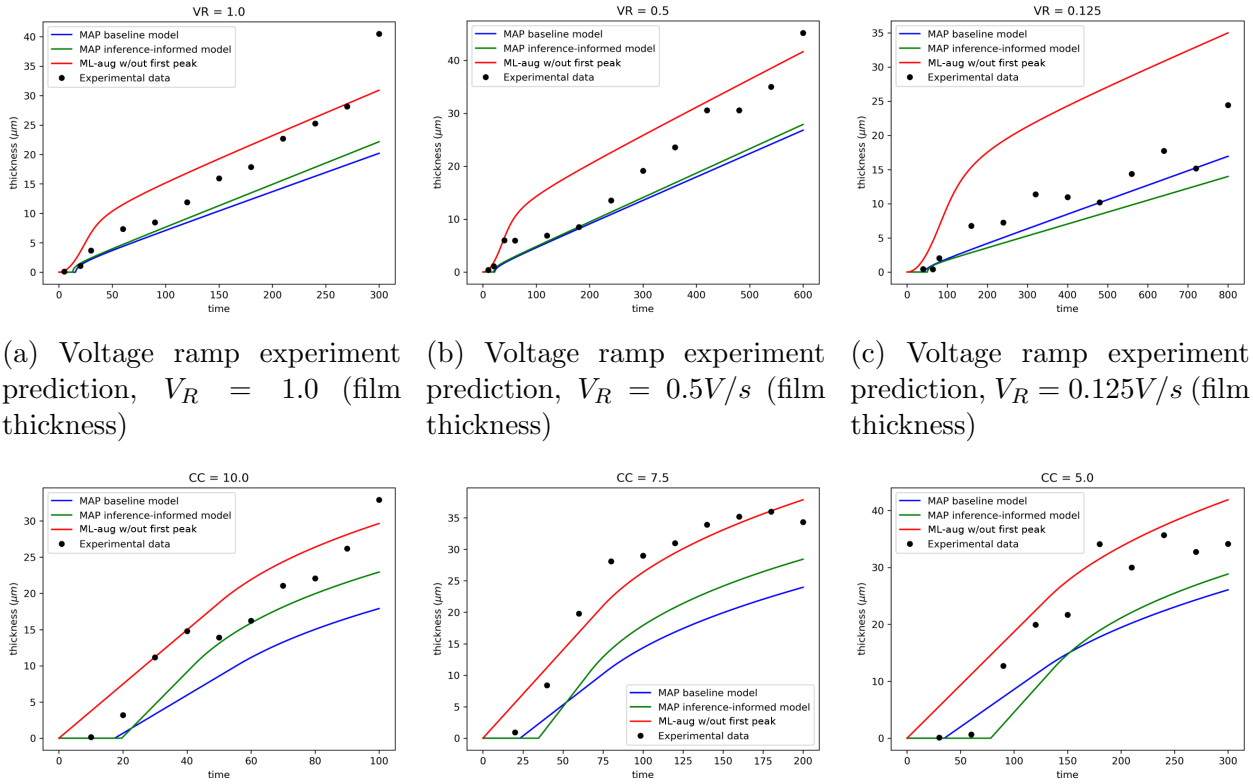

(b) Constant current experiment prediction, $j_0 = 7.5mA$

Figure 4.12: Current prediction on the machine-learning augmented model trained without the first peak model, shown for (a) voltage ramp experiment with $V_R = 1.0V/s$ and (b) constant current experiment with $j_0 = 7.5mA$.

model refinements aimed at rectifying these deficiencies and better aligning the model with experimental observations, such as addressing the elusive physical behaviors not fully replicated by the models, including the emergence of two peaks in current data during voltage ramp experiments.

Through the development and introduction of physically meaningful augmentations and the harnessing of a NeuralODE framework for implementing learnability, we have addressed the challenges of maintaining interpretability while exploiting the expressive power of neural networks for enhanced adaptability and generalizability. This duality of goals reflects the overarching aims of the thesis: to merge the empirical richness of machine learning with the foundational rigor of physics. The integration of these augmentations has yielded models with improved predictive accuracy and generalizability, except in the low voltage regime which may require additional modeling.

Although the ML-based augmentations introduced improve prediction accuracy, an integral part of the methodology presented is the diagnostic analysis performed prior to augmentation, which provides crucial insights into the model's limitations. This understanding is important and informs the design of augmentations that retain physical interpretability and removes the necessity for manual crafting, a practice of immense significance for both machine learning-driven dynamical system understanding and broader physics-based applications. It demonstrates a path towards future explorations into model improvements that remain physically-consistent, emphasizing the need for physical understanding and careful combination with ML-based ideas.




(a) Voltage ramp experiment prediction, $V_R = 1.0$ (film thickness)

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (film thickness)

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (film thickness)

(d) Constant resistance experiment prediction, $j_0 = 10.0mA$ (film thickness)

(e) Constant current experiment prediction, $j_0 = 7.5mA$ (film thickness)

(f) Constant current experiment prediction, $j_0 = 5.0mA$ (film thickness)

Figure 4.13: Comparisons between film thickness prediction on the baseline model, inference-informed model, and ML-augmented model without first peak.

The systematic process outlined in this work is performed on a low dimensional dynamical system for which the computational model is efficient to simulate. However, in more complex cases which necessitate the need for reduced order modeling techniques, ML and data-driven techniques can be leveraged in other ways to achieve improved accuracy.

# CHAPTER 5

# Rate Distortion Informed Clustering of Non-equilibrium Gas Dynamics

This chapter furthers exploration into the union of ML and physical modeling, building upon the insights from the preceding chapters. Chapter 3 presented a compelling case for the inherent limitations of purely data-driven approaches in capturing the nuances of physical phenomena. It highlighted how the infusion of physical information, through inductive biases, significantly enhances the robustness and fidelity of ML models in modeling physical systems. Following this thread, Chapter 4 illustrated the process of refining a baseline model with incompletely understood physics. By integrating inductive biases, we constructed model augmentations that not only preserved a degree of interpretability but also embraced the adaptability and precision offered by neural networks.

In this chapter, the focus is pivoted to the realm of reduced order models (ROMs) for non-equilibrium gas dynamics, illustrating a study that underscores the capabilities of ML to elevate ROM accuracy. This ultimately demonstrates an inherent balance between enhancing model accuracy and maintaining interpretability. While the resultant reduced order model retains a fundamental level of physical interpretability—thanks to the methodical construction of the ROM framework—the incorporation of ML techniques introduces a nuanced compromise on interpretability. Despite this, the compelling benefits of ML integration, particularly in significantly boosting ROM accuracy, are unequivocally demonstrated. This chapter continues the discourse on the integration of ML into physics-based models but also demonstrates and discusses the trade-offs involved, providing a deeper understanding of how ML can be harnessed to advance applications in reduced order modeling. The following discussions draw material directly from Ref. [162].

## 5.1 Introduction

Non-equilibrium fluid flows, characterized by complex thermochemical processes and highly non-linear dynamics of a high number of internal energy degrees of freedom present in molecular systems, play a pivotal role in a wide range of scientific and engineering applications. Understanding and modeling these phenomena are essential for advancing fields such as hypersonic fluid flows [163, 164], plasma physics [165], and materials science [166]. The most accurate approach to model non-equilibrium flows involves the direct solution of the master equation, which relies on quantum state-to-state (STS) chemistry models. While these models provide exceptional accuracy, they become impractical in large-scale multi-dimensional simulations due to the exponential increase in the number of degrees of freedom (molecule and atom energy levels). Thus reducing the number of degrees of freedom can render the simulations feasible but must be performed carefully to ensure as much accuracy as possible is retained. This work focuses primarily on improving the accuracy of reduced order modeling approaches for STS modeling on a path toward improving the efficiency and accuracy of multidimensional hypersonic simulations.

Reduced order models [167, 168, 23] seek to capture the essential features of the system dynamics while significantly reducing the computational cost compared to the full order model. In the case the STS model, commonly employed coarse-graining [169, 170, 171, 172] methods aggregate the individual internal energy states into clusters based on (typically) physics-driven criteria. The solution of the coarse-grained dynamics with the partial equilibration of the underlying microscopic structure provides enough information to approximately describe the thermochemical state of the gas undergoing non-equilibrium phenomena while mass / energy conservation and detailed balance are satisfied by construction.

The coarse-graining approach based on the maximum entropy principle [169, 170] has shown promise in reducing the dimensionality of non-equilibrium systems [173, 174, 175]. By (i) selecting a suitable number of clusters, (ii) defining a specific cluster-wise distribution function, and (iii) assigning internal energy levels to these clusters, the system's dynamics can be approximated by evolving cluster-related macroscopic quantities, i.e., moments of the state-specific distribution function computed within each cluster. The challenge of this approach lies in finding the optimal cluster assignments that best capture the system behavior. Previous works have focused on simple physically interpretable assignments [170, 174] or used graph clustering methods such as a modified version of the island algorithm [176, 177, 173] and spectral clustering [178, 173]. The former exhibits enhanced accuracy compared to energy-based clustering but falls short in accuracy when compared to the latter. However, each of these graph-clustering methods can be unstable to compute and rely only on inelastic

energy transition probabilities as the informative quantity for graph partitioning.

Other reduced order modeling techniques such as parametric variational inference [179, 180] have seen great success when applied to dynamical systems and stochastic reaction networks; however, this work differs in two primary aspects. First, this work deals with a deterministic dynamics model, negating the need for modeling moments or distributions of the dynamics, and we avoid a probabilistic reduced order prediction. Although we perform a probabilistic relaxation of the coarse-graining process to facilitate gradient-based optimization techniques, the final optimal result is deterministic as specified by rate distortion theory. Second, the reduced order model which we develop is not parametric in form, but rather based on the maximum entropy principle, maintaining a level of interpretability of the reduced order system regardless of method of cluster assignment.

In this work, we propose a framework to *learn* the optimal cluster assignments through an ML-based framework. Previous approaches have seen success using heuristic or algorithm-based assignment [170, 173]. These works focus on interpretability first with the aim of improving coarse-grained or reduced order model (ROM) accuracy as a result. Conversely, we focus on optimal ROM accuracy as the primary object. Although this renders the cluster assignment process itself less interpretable, the reduced order model based on maximum entropy principle coarse-graining maintains interpretability by construction. A trade off always exists when incorporating ML into physics - a trade off characterized by an improvement in accuracy corresponding to a reduction in interpretability.

The approach taken in this work takes inspiration from lossy compression techniques [56] and leverages ideas from rate-distortion theory [54, 55] to guide the learning process, resulting in a more robust optimization procedure. Although the optimization procedure is guided by constraints derived from rate distortion theory, the aim is not to determine the optimal encoding decoding system as in Ref. [59]. In contrast, the aim is to determine the optimal encoding system given the physics-constrained maximum entropy-based reduced order model as the decoder. By formulating the cluster assignment problem as an encoding-decoding system and constructing an optimization task, we seek to minimize the distortion between the ROM and STS models subject to some rate of compression (number of clusters).

To achieve this, the system dynamics are transformed into a probabilistic description, enabling the application of probabilistic learning techniques and allowing continuous gradient-based optimization techniques. We develop a fully differentiable solver for the probabilistic coarse-graining approach using an adjoint solver to efficiently backpropagate gradients. As a consequence of the feedback from the predictions to the cluster assignments, this technique is model-consistent [181, 50]. The adjoint computation is implemented with the help of NeuralODE [26] to preserve native automatic differentiation (AD) of the computational graph

in Pytorch [158], facilitating end-to-end training of the cluster assignments. Several other important implementation details and modifications have been made to ensure the accuracy and feasibility of the otherwise intractable training process which are further discussed in Sec. 5.4.2.1.

Our framework offers a significant improvement in ROM accuracy over existing methods by learning the optimal cluster assignments in a data-driven manner, rather than relying on hand-designed or heuristic approaches. The effectiveness and efficiency of this approach is demonstrated to be state of the art through numerical experiments on non-equilibrium flows for a high-temperature reacting $N_2 + N$ system, showcasing improved accuracy compared to traditional coarse-graining methods. Adopting the maximum entropy principles [169, 170] ensures that the data-driven ROM inherently satisfies conservation and detailed balance. Additionally, this formulation ensures the positive production of entropy in the system. Each of these is not guaranteed in fully data-driven ROMs unless imposed as a hard constraint. However, we note that learning the cluster assignments does not guarantee that the assignments will be interpretable. Our method aims to maximize accuracy as the primary object while inherently satisfying physical constraints at the cost of reduced interpretability of the learned clustering. The primary novel contributions of this work are summarized as follows:

- Transforming the maximum entropy-based and coarse-grained master equations to a probabilistic description
- Creating a fully-differentiable coarse-grained dynamics solver based on the probabilistic coarse-graining description
- Developing a machine learning framework using NeuralODE, automatic differentiation, and many implementation optimizations to leverage gradient-based optimization methods for learning optimal cluster assignments
- Viewing the coarse-graining process through the lens of information theory to develop a loss function which encourages rate distortion-optimal learning and a more robust optimization process
- Tractability and stability improvements such as equation transformations, careful scaling, and data limiting to make the computationally demanding optimization feasible.
- Demonstration of the framework on a high-dimensional (nearly 10,000 states) problem with state of the art accuracy compared to other methods.

By combining the power of reduced order modeling, coarse-graining techniques, and machine learning, this framework offers a promising avenue for improving the efficiency and accuracy of simulations in non-equilibrium flows.

## 5.2 Non-Equilibrium Gas Dynamics

We examine the behavior of an $N_2 + N$ system in an ideal isothermal-isochoric chemical reactor with both species considered in their electronic ground state. The STS model consists of evolving the dynamics of individual pseudo-species, or a particular species' internal degrees of freedom treated as state variables. For the system studied in this work, the rovibrational energy levels of $N_2$ are considered in its electronic ground state and N also in its electronic ground state. A system of master equations describes the evolution of the dynamical system by modeling elementary interactions between each of the pseudo-species in the system. An in-depth discussion of macrostates and microstates and how they relate to this problem is provided in Appendix C.1 and may serve as a useful introduction to the topic.

### 5.2.1 Master Equations

Non-equilibrium flows are widely observed in applications such as combustion [182], hypersonics [183, 184, 185, 186], and material processing [187]. These flows are characterized by a deviation of the microstates (given by $N_S^i$) from its Maxwell-Boltzmann distribution. Modeling such non-equilibrium flows requires solving the *master equations* which describe the dynamics of the microstates undergoing collisional and ionization processes.

For each particular species S, particles exist in one of the allowable energy states corresponding to that species. Each of the allowable internal states is specified with a particular internal energy and degeneracy. For internal state $i$, the degeneracy $g_S^i$ defines the number of possible molecular internal combinations corresponding to internal energy $\varepsilon_S^i$.

Consider a pair of atoms or molecules A and B. Each can undergo excitation, de-excitation, ionization, dissociation, and recombination during a mutual collision. Internal excitation and de-excitation processes begin with molecule A in one of its internal states $i$ interacting with another molecule B in one of its internal states $j$ such that one or both undergo internal excitation or de-excitation. This process is given by

$$\mathrm{A^i + B^j \rightleftharpoons A^k + B^l}\,. \tag{5.1}$$

We also consider that A can undergo ionization, dissociation, and recombination. This

process can be expressed as

$$\mathrm{A^i + B^j \rightleftharpoons C^p + D^q + B^l} \, . \tag{5.2}$$

From detailed balance, the microscopic state transition equations, or the *master equations* for general transition for a species (say A) is given as

$$\begin{aligned} \frac{dn_A^i}{dt} \triangleq & \sum_j \sum_k \sum_l (-\kappa_{ij,kl} n_A^i n_B^j + \kappa_{kl,ij} n_A^k n_B^l) \\ + & \sum_j \sum_l \sum_p \sum_q (-\kappa_{ij,lpq} n_A^i n_B^j + \kappa_{lpq,ij} n_B^l n_C^p n_D^q), \end{aligned} \tag{5.3}$$

where $n_S^i$ denotes the number density of state $i$ of species $S$ and $\kappa_{\eta\xi,\gamma\sigma}$ is the Maxwellian-distribution-based state-to-state rate coefficient for transition from an internal configuration pair $(\eta, \xi)$ to $(\gamma, \sigma)$. This essentially represents the frequency or probability with which the transition from $(\eta, \xi)$ to $(\gamma, \sigma)$ will occur during a collision.

In the special case where the internal configuration of the collision partner B does not change, and the products C and D are equal to B (which is considered in this work), the general transition in (Eq. 5.1) and (Eq. 5.2) can be reduced as

$$\mathrm{A^i + B \rightleftharpoons A^k + B} \tag{5.4}$$

$$\mathrm{A^i + B \rightleftharpoons B + B + B} \, . \tag{5.5}$$

This transforms Eq. 5.3 to

$$\frac{dn_A^i}{dt} \triangleq \sum_k (-\kappa_{i,k} n_A^i n_B + \kappa_{k,i} n_A^k n_B) + (-\kappa_i^d n_A^i n_B + \kappa_i^r n_B^3) \, . \tag{5.6}$$

where $\kappa_{i,k}$ / $\kappa_{k,i}$ represent the excitation and de-excitation rate coefficients describing the transition probability of jumping from state $k$ to state $i$ and vice-versa, and $\kappa_i^d$ / $\kappa_i^r$ describe the ionization/dissociation and recombination rate coefficients, respectively. We note that the dynamics to model species B are much simpler in this case, and only require a single equation for $n_B$:

$$\frac{dn_B}{dt} = \sum_i \kappa_i^d n_A^i n_B - \kappa_i^r n_B^3 \tag{5.7}$$

Consider the simplified state-to-state transition equation given in (Eq. 5.6) such that *total* number density $n_A$ and energy density $e_A$ dynamics for species A are given by

$$\frac{dn_A}{dt} \triangleq \sum_i \frac{dn_A^i}{dt} = \sum_i \sum_k (-\kappa_{i,k} n_A^i n_B + \kappa_{k,i} n_A^k n_B) + (-\kappa_i^d n_A^i n_B + \kappa_i^r n_B^3), \tag{5.8}$$

$$\begin{aligned}\frac{de_A}{dt} \triangleq \sum_i \varepsilon_A^i \frac{dn_A^i}{dt} = &\sum_i \sum_k \left(-\kappa_{i,k} n_A^i \varepsilon_A^i n_B + \kappa_{k,i} n_A^k \varepsilon_A^i n_B\right) \\ &+ \sum_i \left(-\kappa_i^d n_A^i \varepsilon_A^i n_B + \kappa_i^r \varepsilon_A^i n_B^3\right).\end{aligned} \tag{5.9}$$

Typically, even for a relatively simple system such as $N_2 + N$, solving such STS equations for density and energy is computationally demanding and impractical for multidimensional large scale simulations.

### 5.2.2 Coarse-grained equations

Solving the master equations is computationally expensive and impractical. Indeed, in common CFD solvers, the master equations must be solved for the chosen gas mixture in each single cell of the discretized domain, which is prohibitively expensive. For example, a system consisting of only $N_2$ and N results in a system of almost 10,000 coupled equations for the zero-dimensional dynamical system, and adding spatial degrees of freedom greatly increases the dimensionality of the problem. As a result, there have been significant efforts to develop reduce order models that can predict the time evolution of $n_S^i$ with high confidence. In this section, we discuss coarse-graining [170] along with some additional insights.

The coarse-grained equations are constructed by first distributing $\ell$ internal states among $m$ clusters, where $m \ll \ell$. That is, each state is assigned to a single cluster. A cluster-wise distribution function is then assumed to represent the internal state population within each cluster. In this case, we employ the maximum entropy (log-polynomial) function [169, 170]. Moments of the master equations are then taken using the assumed cluster-wise distribution functions, yielding the governing equations describing the evolution of cluster properties. This method of coarse-graining, or reduced order modeling, preserves the form of the master equations in the coarse-grained equations. It is also guaranteed to satisfy basic conservation and detailed balance as opposed to fully data-driven ROMs in which it is not guaranteed.

**Piecewise representation of microscopic quantities** The internal energy states can be *grouped*, *binned*, or *clustered* together to reduce the dimensionality of solving the state-to-state system of equations in Eqs. 5.8 and 5.9. The reduced system of equations can then be solved for the cluster density $n_S^j$ and energy $\varepsilon_S^j$, where $j$ is the bin or cluster index. There exists a set of state indices $\mathcal{B}^j = \{i \mid i \in \text{group } j\}$ for each group consisting of all state

indices assigned to group $j$.

The full internal state distribution is approximated (reconstructed) from the coarse-grained solution by assuming a cluster-wise distribution function. As the energy of each internal state is constant, [170] assumes that the number density of each state $n_S^i$ can be represented as a linear combination of monomials in terms of $\varepsilon_S^i$

$$n_S^i = \alpha_S + \beta_S \varepsilon_S^i + \gamma_S (\varepsilon_S^i)^2 + \dots, \tag{5.10}$$

where the coefficients $\alpha_S$, $\beta_S$, etc. are the unknown coefficients to be computed. These coefficients are used to reconstruct, or approximate, the full order solution. For all states in a single cluster $i \in \mathcal{B}^j$, the reconstructed number density for a single internal state $\hat{n}_S^i$ in the clsuter is given by

$$\hat{n}_S^i = \alpha_S^j + \beta_S^j \varepsilon_S^i + \gamma_S^j (\varepsilon_S^i)^2 + \dots; i \in \mathcal{B}^j, \tag{5.11}$$

such that the basis coefficients are unknown for each group rather than for each energy state individually. Based on the maximum entropy argument [169] (discussed in Appendix C.2), the relationship is more appropriately chosen as

$$\log\left(\frac{g_S^i}{\hat{n}_S^i}\right) = \alpha_S^j + \beta_S^j \varepsilon_S^i + \gamma_S^j (\varepsilon_S^i)^2 + \dots;\ i \in \mathcal{B}^j, \tag{5.12}$$

such that

$$\hat{n}_S^i = g_S^i e^{-\alpha_S^j - \beta_S^j \varepsilon_S^i - \gamma_S^j (\varepsilon_S^i)^2 + \dots};\ i \in \mathcal{B}^j, \tag{5.13}$$

subject to the macroscopic physical constraints

$$\tilde{n}_S^j = \sum_{i \in \mathcal{B}^j} \hat{n}_S^i; \quad \tilde{e}_S^j = \sum_{i \in \mathcal{B}^j} \hat{n}_S^i \varepsilon_S^i; \quad \tilde{f}_S^j = \sum_{i \in \mathcal{B}^j} \hat{n}_S^i (\varepsilon_S^i)^2 \dots. \tag{5.14}$$

where the $\tilde{n}_S^j$ and $\tilde{e}_S^j$ indicate the number density and energy density of cluster $j$, respectively. The maximum entropy formulation in Eq. 5.12 is an accurate representation of the internal states at thermal equilibrium as it reproduces the Boltzmann distribution exactly using one group truncated to a linear model, keeping only $\alpha_S^j$ and $\beta_S^j$ as the basis coefficients. We thus assume a bin-wise reconstruction function of

$$\log\left(\frac{\hat{n}_S^i}{g_S^i}\right) = -\alpha_S^j - \beta_S^j \varepsilon_S^i\ ;\ \forall i \in \mathcal{B}^j, \tag{5.15}$$

in this work.

#### 5.2.2.1 Bin-averaged reaction rate coefficients

The coarse-grained system of equations corresponding to the simplified state-to-state Eq. 5.6 and the corresponding energy evolution equation are very similar and given by

$$\frac{d\tilde{n}_A^j}{dt} \triangleq \sum_{w=1}^{m}(-{}^0K_{j,w}\tilde{n}_A^j n_B + {}^0K_{w,j}\tilde{n}_A^w n_B) + (-{}^0K_j^d\tilde{n}_A^j n_B + {}^0K_j^r n_B^3), \tag{5.16}$$

$$\frac{d\tilde{e}_A^j}{dt} \triangleq \sum_{w=1}^{m}(-{}^1K_{j,w}\tilde{e}_A^j n_B + {}^1K_{w,j}\tilde{e}_A^r n_B) + (-{}^1K_j^d\tilde{e}_A^j n_B + {}^1K_j^r n_B^3)\ , \tag{5.17}$$

where the coarse-grained rate coefficients are given by [170]

$${}^uK_{j,w} \triangleq \frac{1}{{}^uQ_A^j}\sum_{i\in\mathcal{B}^j}\sum_{k\in\mathcal{B}^w}\kappa_{i,k}(\varepsilon_A^i)^u g_A^i \exp(-\beta_A^j\varepsilon_A^i), \tag{5.18}$$

$${}^uK_j^d \triangleq \frac{1}{{}^uQ_A^j}\sum_{i\in\mathcal{B}^j}\kappa_i^d(\varepsilon_A^i)^u g_A^i \exp(-\beta_A^j\varepsilon_A^i), \tag{5.19}$$

$${}^uK_j^r \triangleq \sum_{i\in\mathcal{B}^j}\kappa_i^r(\varepsilon_B^i)^u, \tag{5.20}$$

and ${}^uQ_A^j$ are moments of the state-specific partition function

$${}^uQ_A^j = \sum_{i\in\mathcal{B}^j} g_A^i(\varepsilon_A^i)^u \exp(-\beta_A^j\varepsilon_A^i). \tag{5.21}$$

#### 5.2.2.2 Bin-averaged initial condition

After the set of cluster assignments $\mathcal{B}^j$ are determined, an initial condition must be defined to solve Eqs. 5.16 and 5.17. The initial condition is specified as number densities of each species and an initial internal temperature such as $n_A(0)$, $n_B(0)$, and $T_{int}(0)$. We are interested in computing the initial condition on the internal state distribution of species $A$ only as we assume species $B$ to be single state in this work. To compute the initial condition $n_A^i$ for each internal state individually is relatively straightforward. We assume that the entire system is initially in equilibrium and therefore follows the Maxwell-Boltzmann distribution, requiring only computing $\alpha_A$ and $\beta_A$. Each internal density $n_A^i$ can thus be computed and $\tilde{n}_A^j$, $\tilde{e}_A^j$ for each cluster computed by

$$\tilde{n}_A^j = \sum_{i\in\mathcal{B}^j} n_A^i, \quad \text{and} \quad \tilde{e}_A^j = \sum_{i\in\mathcal{B}^j} n_A^i\varepsilon_A^i.$$

#### 5.2.2.3 Reconstruction function

After computing the solution to the coarse-grained equations, we reconstruct the internal state distribution at some time $t$ using Eq. 5.15. The bin-wise coefficients $\alpha_S^j$ and $\beta_S^j$ are functions of time and computed by

$$\beta_S^j = \frac{1}{kT_S^j} \quad , \quad \alpha_S^j = \log\left(\frac{{}^0Q_S^j}{\tilde{n}_A^j}\right) \ , \tag{5.22}$$

where $T_S^j$ is an internal 'temperature' which is a modeled quantity computed during non-equilibrium. We refer the reader to [170] for more details on the derivation of these quantities.

### 5.2.3 Existing clustering methods

#### 5.2.3.1 Internal energy-based clustering

The range of the internal energy space is equally divided into $m$ bins, and all internal states are assigned to the corresponding bins. This method is the original clustering method developed for use with maximum-entropy-based coarse graining [170].

For a given species A, the bounds of the internal energy space are found such that all internal states lie on the range $\mathcal{R} = [a, b]$, where $a = \min_i \varepsilon_A^i$ and $b = \max_i \varepsilon_A^i$. This range is divided into sub-ranges $\mathcal{R}_j = [a + (j-1)(b-a)/m,\ a + j(b-a)/m]\ \forall j \in \{1, \ldots, m\}$. The sets of states $\mathcal{S}_j = \{i | \varepsilon_A^i \in \mathcal{R}_j\}\ \forall j \in \{1, \ldots, m\}$ are formed by associating all states with internal energy in range $\mathcal{R}_j$ with cluster $j$.

This method of clustering is physically interpretable by design, but results in relatively poor accuracy as illustrated in Sec. 5.5.

#### 5.2.3.2 Spectral clustering

Spectral clustering is a well established idea for partitioning connected graphs [178]. This idea was extended in [173] to improve coarse-grained ROM accuracy over clustering based on internal energy. The idea is to compute a degree of connectedness $S_{ik}$ between two internal states $i, k$ based on their proximity in internal energy space $|\varepsilon^i - \varepsilon^k|$ and the magnitude of the internal state transition coefficient $\kappa_{i,k}$. These are them manipulated to form the weights on edges in a connected graph of all the states. Spectral clustering is then used to partition the graph into a discrete number of bins.

One drawback of this method is that it does not consider all internal energy states. Although more accurate than energy-based clustering, it does not take into account any

quasi-bound states [173]. Neglecting the quasi-bound states results in reverting to energy-based clustering for such states. Additionally, this approach relies only on inelastic energy transition probabilities as the informative quantity for graph partitioning.

#### 5.2.3.3 Centrifugal barrier clustering

This method is a physics-driven clustering approach that groups internal states based on their distance from the molecule's centrifugal barrier, a distinct property of the diatomic potential [174]. This strategy exhibits significantly improved accuracy compared to vibrational-specific clustering [188], as it effectively captures the dissociation dynamics inherent in the STS model. However, it should be noted that this clustering strategy exclusively addresses dissociation processes, in contrast to spectral clustering, which considers excitation processes only. This requires the STS model to be computed for the excitation and de-excitation process, and we thus omit this strategy from the comparative analysis with our approach.

## 5.3 Mathematical Framework

The primary goal of this work is to find optimal cluster assignments for the reduced order models described in the previous section. We aim at solving the optimization problem

$$\mathbf{c}^* = \arg\min_{\mathbf{c}} \ \mathcal{D}(\mathbf{n}_A, \hat{\mathbf{n}}_A(\mathbf{c})) \ , \tag{5.23}$$

where the discrete cluster assignment vector $\mathbf{c} \in \{1, \dots, m\}^\ell$ is used to compute the coarse-grained rate coefficients by Eqs. 5.18, 5.19, and 5.20. The coarse-grained system is then simulated to steady state and reconstructed at each time $t$ as $\hat{\mathbf{n}} = [\hat{n}_A(\mathbf{c}, 0), \hat{n}_A(\mathbf{c}, 1), \dots, \hat{n}_A(\mathbf{c}, t), \dots, \hat{n}_A(\mathbf{c}, T)]$. This is then compared to the solution from the full state-to-state equations (Eq. 5.6) by some distortion (loss) $\mathcal{D} : \mathbb{R}^{\ell \times n_t} \times \mathbb{R}^{\ell \times n_t} \to \mathbb{R}$.

Solving this optimization problem directly is prohibitively expensive. It is a discrete optimization problem, which requires expensive combinatorial type optimization procedures. With a dimension of 10,000, directly optimizing the vector $\mathbf{c}$ is infeasible. We therefore transform the problem to a probabilistic description to facilitate the use of gradient-based optimization methods.

To solve the optimization problem, we turn towards rate-distortion theory to encourage learning optimal assignments. From this, a loss function is derived which guides the learning process toward the RD-optimal solution.

### 5.3.1 Probabilistic description of coarse graining

Transforming the system to a probabilistic description allows for a continuous representation and thus continuous optimization methods can be applied. Instead of directly optimizing the cluster assignment vector, we optimize the elements of a probability matrix which defines a distribution on the cluster assignment vector. We assume that all cluster assignments are independent such that the matrix $\mathbf{P} \in [0,1]^{\ell \times m}$ defines the entire probability space. Each element $P_{ij}$ represents the probability that state $i$ belongs to group $j$.

Using a probabilistic description is the basis of transforming the optimization problem to a continuous setting, but there exist many other factors accompanying this change which must be addressed. Firstly, an adjustment of Eq. 5.23 must be introduced to compute the distortion. With the cluster assignment vector now a random variable, the expectation is computed as $\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\mathcal{D}(\mathbf{n}_A, \hat{\mathbf{n}}_A)]$, as is the case in rate-distortion theory. The transformed optimization problem is thus as follows:

$$\mathbf{P}^* = \arg\min_{\mathbf{P}} \mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\mathcal{D}(\mathbf{n}_A, \hat{\mathbf{n}}_A(\mathbf{c}))]. \tag{5.24}$$

To use gradient-based optimization methods, we must also compute gradients through the solution of the ODE (Eqs. 5.16 and 5.17) and the coarse-grained reaction rate coefficient equations (Eqs. 5.18- 5.20). Computing these gradients along with the remaining challenges in solving Eq. 5.24 are discussed in Sec. 5.4. First we will discuss two methods of approximating the expectation $\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\mathcal{D}(\mathbf{n}_A, \hat{\mathbf{n}}_A)]$.

#### 5.3.1.1 Expectation computation

The expectation $\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\mathcal{D}]$ is nonlinear and thus must be approximated to avoid prohibitively high computational costs. We propose two methods for efficiently approximating this expectation for use in an optimization framework. Throughout this section, we explicitly indicate dependencies on the cluster assignment vector $\mathbf{c}$ and time $t$ for clarity. Further, we replace the distortion $\mathcal{D}(\mathbf{n}_A, \hat{\mathbf{n}}_A(\mathbf{c}))$ by $\mathcal{D}(\log \mathbf{n}_A, \log \hat{\mathbf{n}}_A(\mathbf{c}))$.

**Expectation approximation** The first method we use to approximate the expected distortion is to approximate all nonlinear expectations. The distortion we use is the common mean-squared distortion, which is convex. We make the approximation

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\mathcal{D}(\log \mathbf{n}_A, \log \hat{\mathbf{n}}_A(\mathbf{c})] \approx \mathcal{D}(\log \mathbf{n}_A, \mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\log \hat{\mathbf{n}}_A(\mathbf{c})]), \tag{5.25}$$

where for every time step $t$, each state $i$ is reconstructed by

$$\log \hat{n}_A^i(\mathbf{c}, t) = -\alpha_A^j(\mathbf{c}, t) - \beta_A^j \varepsilon_A^i \ ; \ \forall i \in \mathcal{B}^j.$$

Note that we drop the time index $t$ for notational simplicity, but the following discussion is for a particular time $t$ and must be repeated at each time which is used to compute the distortion.

As mentioned in the previous section, temperature is assumed constant in the optimization framework. Therefore $\beta_A^j = 1/k_B T$, where $k_B$ is the Boltzmann constant and $T$ is the translational temperature of the 0D reactor. The expectation is therefore

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\log \hat{n}_A^i(\mathbf{c}, t)] = -\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\alpha_A^j(\mathbf{c}, t)] - \beta_A^j \varepsilon_A^j \ ; \ \forall i \in \mathcal{B}^j. \tag{5.26}$$

Using Eq. 5.22), we make the approximation

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\alpha_A^j(\mathbf{c}, t)] = \mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\log {}^0Q_A^j(\mathbf{c})] - \mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\log \tilde{n}_A^j(\mathbf{c}, t)] \approx \log \mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})] - \log \mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\tilde{n}_A^j(\mathbf{c}, t)]. \tag{5.27}$$

The first term of the approximation has a closed form solution

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})] = \sum_{i=1}^{\ell} P_{ij} g_A^i \exp(-\beta_A^j \varepsilon_A^i), \tag{5.28}$$

which also avoids the non-differentiable summation condition from Eq. 5.21.

The second term in the approximation of Eq. 5.27 requires more simplification. Recall $\tilde{n}_A^j(\mathbf{c}, t)$ is the solution to the coarse-grained system of equations in Eq. 5.16 for cluster $j$ at time $t$. This solution is a function of the coarse-grained reaction rate coefficients, which are a function of the cluster assignment vector . We make a further approximation that $\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[\tilde{n}_A^j(\mathbf{c}, t)]$ is the solution to Eq. 5.16 computed using the expected reaction-rate coefficients

$$\begin{aligned}
&\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_{j,w}(\mathbf{c})] \ , \ \forall j, w \in \{1, \ldots, m\}, \\
&\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_j^d(\mathbf{c})] \ , \ \forall j \in \{1, \ldots, m\}, \\
&\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_j^r(\mathbf{c})] \ , \ \forall j \in \{1, \ldots, m\}.
\end{aligned}$$

Finally, we make the approximations

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_{j,w}(\mathbf{c})] \approx \frac{\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})^0K_{j,w}(\mathbf{c})]}{\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})]},$$

and

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_j^d(\mathbf{c})] \approx \frac{\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})^0K_j^d(\mathbf{c})]}{\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})]} \ .$$

The closed form equations we use to compute these expectations are thus

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_{j,w}(\mathbf{c})] \approx \frac{\sum_{i=1}^{\ell}\sum_{k=1}^{\ell} P_{ij}P_{kw}\kappa_{i,k}g_A^i \exp(-\beta_A^j\varepsilon_A^i)}{\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})]} \ , \ \forall j,w \in \{1,\ldots,m\}, \tag{5.29}$$

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_j^d(\mathbf{c})] \approx \frac{\sum_{i=1}^{\ell} P_{ij}\kappa_i^d g_A^i \exp(-\beta_A^j\varepsilon_A^i)}{\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0Q_A^j(\mathbf{c})]} \ , \ \forall j \in \{1,\ldots,m\}, \tag{5.30}$$

$$\mathbb{E}_{\mathbf{c}\sim\mathbf{P}}[{}^0K_j^r(\mathbf{c})] = \sum_{i=1}^{\ell} P_{ij}\kappa_i^r \ , \ \forall j \in \{1,\ldots,m\}. \tag{5.31}$$

Finally, we approximate the expected distortion using Eqs. 5.25, 5.26, 5.27, and 5.28, with the coarse-grained solution computed at the expected coarse-grained reaction rate coefficients using Eqs. 5.29, 5.30, and 5.31. This also removes the non-differentiable summation conditions to create a differentiable function.

**Sampled approximation** An additional method we propose to approximate the expected distortion is through Monte Carlo (MC) sampling. Similar to variational autoencoder training (VAE) [189], approximating the expectation with a single sample works well in practice due to the stochastic nature of the optimization itself. The sampling operation is made differentiable through the parameterization trick [189]. That is, a sample is drawn from some easily sampled distribution and transformed through some parameterization to sample from the desired distribution. Differentiable sampling of discrete one-hot vectors requires a more advanced approach than sampling from a Gaussian distribution. The recent work in [190] introduces the Gumbel-Softmax reparameterization trick. This parameterization allows sampling from a discrete categorical distribution such that the sample is 'one-hot' - only one element of the resulting vector is equal to 1, and all other elements are equal to 0. Taking the statistical mean with respect to infinitely many samples using the Gumbel-Softmax trick will approach the defined discrete distribution.

Using the Gumbel-softmax trick, we can approximate the expectation using a single sample from the cluster assignment vector $\mathbf{c}$. To avoid the non-differentiable summation conditions, we instead sample a matrix $\mathbf{C} \in \mathbb{R}^{\ell\times m}$, where each element is equal to

$$\begin{cases} C_{ij} & = 1 \ ; \ c_i = j \\ C_{ij} & = 0 \ ; \ c_i \neq j \ . \end{cases}$$

Each row of the $\mathbf{C}$ matrix should be one-hot, indicating that the state is assigned to a single cluster. To achieve this, we use the Gumbel-softmax trick to sample from each row of $\mathbf{P}$ and assign it to the corresponding row in $\mathbf{C}$.

The expected distortion is then computed using the single sample $\mathbf{C}$ by solving Eq. 5.16 using the coarse-grained rate coefficients defined by

$$ {}^0K_{j,w}(\mathbf{c}) = {}^0K_{j,w}(\mathbf{C}) = \frac{1}{{}^0Q_A^j(\mathbf{C})} \sum_{i=1}^{\ell} \sum_{k=1}^{\ell} C_{ij} C_{kw} \kappa_{i,k} g_A^i \exp(-\beta_A^j \varepsilon_A^i) \ , \ \forall j, w \in \{1, \ldots, m\}, \tag{5.32} $$

$$ {}^0K_j^d(\mathbf{c}) = {}^0K_j^d(\mathbf{C}) = \frac{1}{{}^0Q_A^j(\mathbf{C})} \sum_{i=1}^{\ell} C_{ij} \kappa_i^d g_A^i \exp(-\beta_A^j \varepsilon_A^i) \ , \ \forall j \in \{1, \ldots, m\}, \tag{5.33} $$

$$ {}^0K_j^r(\mathbf{c}) = {}^0K_j^r(\mathbf{C}) = \sum_{i=1}^{\ell} C_{ij} \kappa_i^r \ , \ \forall j \in \{1, \ldots, m\}, \tag{5.34} $$

and

$$ {}^0Q_A^j(\mathbf{c}) = {}^0Q_A^j(\mathbf{C}) = \sum_{i=1}^{\ell} C_{ij} g_A^i \exp(-\beta_A^j \varepsilon_A^i). \tag{5.35} $$

Note that these equations are equivalent to Eqs. 5.18, 5.19, and 5.20. This avoids the non-differentiable summation conditions, and allows a fully differentiable approximation of the distortion with MC-based sampling. The forward pass is exact in this case, and the approximation comes only from finite number of samples. However, we found in practice that this approximation is far less stable during training than the previously presented approximation method. We have empirically found that approximating expectations yields better results in practice, as it requires significantly fewer optimization iterations compared to the Monte Carlo (MC) sampling-based approximation to achieve similar outcomes. However, since there has been no formal analysis or guarantees regarding the quality of these approximations, it is possible that the MC approximation may be more versatile and generally applicable in a wider range of scenarios.

### 5.3.2 Rate Distortion Clustering

Information theory is a natural tool for analyzing clustering algorithms. Much of its theorems are extensively studied in the context of discrete problems. Clustering is discrete by nature; a collection of items are grouped into a discrete number of bins. Each element in the collection may be continuous or discrete in nature, but the clustering itself is discrete. Thus, we can apply discrete concepts from information theory which facilitate a detailed analysis

of clustering algorithms. In particular, RD provides a theoretic limit in performance which is achievable through clustering. We can also leverage conditions of optimality in an attempt to *learn* what the optimal clustering is, rather than relying on ad hoc clustering algorithms.

#### 5.3.2.1 Optimality of Deterministic Encoding-Decoding Systems

In this section, we draw comparisons between using deterministic encoding functions and probabilistic encoders.

Consider the case illustrated in Fig. 2.1 where we have the Markov chain $Y - X - C - \tilde{Y}$. The question we seek to answer is: what is the minimum achievable distortion using a finite number $m$ of outputs? We seek to learn the channel distribution $p(c|x)$ and decoding function $g(c)$ such that $\mathcal{D}$ is minimized for a given $m$.

First, we derive the channel capacity as

$$\begin{aligned} C_I &= \max_{p(x)} I(X;C), \\ &= \max_{p(x)} H(C) - H(C|X), \qquad \text{(a)} \\ &= \max_{p(x)} H(C), \qquad \text{(b)} \\ &= \log_2 m, \end{aligned} \tag{5.36}$$

where (a) is the definition of mutual information with $H(C)$, $H(C|X)$ defining Shannon entropy, (b) follows from the fact that $H(C|X) \geq 0$, and the final equality is due to $c \in \{1, \ldots, m\}$ and $p(c) = \mathbb{E}_{p(x)}[p(x)p(c|x)]$.

We now have the chain of inequalities

$$I(Y;\tilde{Y}) \leq_{(a)} I(Y;C) \leq_{(b)} I(X;C) \leq_{(c)} C_I \tag{5.37}$$

where

- (a) follows from the data processing inequality with equality if and only if $I(Y;C|\tilde{Y}) = 0$,
- (b) follows from the data processing inequality with equality if and only if $I(X;C|Y) = 0$,
- (c) follows from the definition of channel capacity with equality if and only if $H(C|X) = 0$ and $p(c = i) = 1/m, \ \forall\, i \in \{1, \ldots, m\}$.

From the source-channel separation theorem with distortion, some distortion $D$ is achievable if and only if $R(D) \leq C_I$, and from the definition of the rate distortion function,

$$R(D) \leq I(Y; \tilde{Y}) \ . \tag{5.38}$$

We also know that the rate-distortion function is monotonic in $D$; thus, the minimum distortion $D^*$ is obtained at $R(D^*) = C_I$. To achieve $D^*$, all inequalities in Eq. 5.37 must be equality.

For $D^*$ to be achievable, we require $R(D^*) = C_I$ and thus the following must hold:

1. $H(C|X) = 0$,
2. $I(X; C|Y) = 0$,
3. $I(Y; C|\tilde{Y}) = 0$,
4. $p(c = i) = \int_{\mathcal{X}} p(x)p(c = i|x)dx = 1/m, \ \ \forall i \in \{1, \ldots, m\}$.

The above 4 conditions can be enforced by ensuring

1. $C = h(X)$, where $h : \mathcal{X} \to \{1, \ldots, m\}$ is surjective.
2. $f$ is invertible
3. $g$ is invertible
4. $H(C) = \log_2 m \iff \mathbb{E}_{p(y)}[\mathbb{1}_{\{i=h(f(y))\}}] = \frac{1}{m}, \ \ \forall i \in \{1, \ldots, m\}$

Thus, if we define an overall encoding function $f_h = h \circ f$ the minimum distortion $D^*$ is a function of $m$ and is achieved by:

$$\begin{aligned} D^*(m) = & \min_{f_h, g} \mathbb{E}_{Y \sim p(y)}[D(Y, g(f_h(Y)))]. \\ \text{subject to} \ \ & f_h \ \text{ surjective} \\ & g \ \text{ invertible} \\ & \mathbb{E}_{p(y)}[\mathbb{1}_{\{i=f_h(y)\}}] = \frac{1}{m}, \ \ \forall i \in \{1, \ldots, m\} \end{aligned} \tag{5.39}$$

To solve Eq. 5.39 and enforce the constraints, we must carefully design the optimization framework.

Note that the optimization problem given by

$$D^*(m) = \min_{f_h, g} \mathbb{E}_{Y \sim p(y)}[D(Y, g(f_h(Y))], \tag{5.40}$$

has the same solution as Eq. 5.39, but the added conditions constrain the high dimensional optimization to ensure that the minimum achievable distortion remains *realizable*. Simply minimizing the average distortion often leads to local minimum solutions in our experiments and is not as robust without the constraints provided by 5.39. Enforcing the optimization constraints of Eq. 5.39 is a question of implementation details. We discuss here one way of enforcing these conditions.

Consider a case in which we have $R$ bits to represent a sample from the source $y \sim p(y)$. This means that there are $m = 2^R$ elements in a *codebook* $\mathcal{B} = \{1, \ldots, m\}$ with which we can use to represent samples of the source distribution. We use this codebook as an example for simplicity, but the codebook can contain any $m$ unique elements. A deterministic encoding function $f_h : \mathcal{Y} \to \mathcal{B}$ is desirable, but designing general and flexible functions $f_h$ which are guaranteed to only predict elements of the codebook (a discrete space) is non-trivial. Instead, we replace this with a function $f_\phi : \mathbb{Y} \to [0, 1]^m$ such that a conditional distribution is predicted over the codebook for some input by

$$p(\mathbf{c}|y) = f_\phi(y) \ ,$$

where $\mathbf{c} \in \mathbb{R}^m$ is a vector corresponding to each of the elements in the codebook. The function $f_\phi$ can be easily constructed as a neural network or any other parameterized function with a softmax output layer. This will ensure that a probability distribution on the codebook is predicted for each input sample. However from Sec. 5.3.2.1, it is clear that to obtain minimum distortion the encoding function must be deterministic. In other words, $H(C|Y) = 0$, and this can be easily encouraged in the optimization objective as

$$H(C|Y) = -\mathbb{E}_Y \left[ \sum_{j=1}^{m} f_{\phi,j}(y) \log f_{\phi,j}(y) \right],$$

is inexpensive to compute.

We can also parameterize the decoding function $g_\psi : \mathcal{B} \to \mathcal{R}$ as a neural network or some other flexible function. As we will only input codebook values, the decoding (or reconstruction) function will predict at most $m$ *reconstruction points*.

Finally, we arrive at the optimization objective which we use to perform rate-distortion

informed clustering

$$\hat{D}^*(m) = \min_{\phi,\psi} \mathbb{E}_{Y,C\sim p(y)f_\phi(y)}[D(Y, g_\psi(C))] + \alpha H(C|Y) + \beta \sum_{i=1}^{m} \left( \mathbb{E}_{Y,C\sim p(y)f_\phi(y)}[\mathbb{1}_{\{i=C\}}] - \frac{1}{m} \right)^2 , \tag{5.41}$$

where the argument gives the trainable loss function.

## 5.4 Computational framework

With a probabilistic description of the system defined (Sec. 5.3.1) and a loss function to optimize (Eq. 5.41), we connect our problem to the RD-based loss function and discuss implementation details. An overview of our framework is illustrated in Fig. 5.1

### 5.4.1 RD-Based Clustering of Internal States

The goal of our clustering problem is to assign each internal state $n_A^i$ of species $A$ to a cluster $j$. To this end, we introduce a classification model which predicts the cluster probabilities $p(\mathbf{c}_i|\mathbf{s}_i) = f_\phi(\mathbf{s}_i)$ from the state information $\mathbf{s}_i$ such as rotational / vibrational quantum numbers and internal energy. The classification model predicts the cluster assignments for each state individually, and the entire probability matrix is assembled by

$$\mathbf{P} = p(\mathbf{c}|y) = \begin{bmatrix} p(\mathbf{c}_1|\mathbf{s}_1) \\ \vdots \\ p(\mathbf{c}_\ell|\mathbf{s}_\ell) \end{bmatrix} \in \mathbb{R}^{\ell\times m} .$$

We use the optimization objective in Eq. 5.41 to train the classifier. The 'decoder' in this case does not need to be constructed. It is already prescribed and consists of computing the coarse-grained (reduced-order) solution to Eqs. 5.16, or more accurately using one of the methods in Sec. 5.3.1.1 to approximate the expectation of the solution with respect to $\mathbf{P}$. The distortion we assume during training is the $L_2$ norm between the log of number density:

$$D(\mathbf{n}_A, \hat{\mathbf{n}}_A(\mathbf{c}_\phi)) = \|\log \mathbf{n}_A - \log \hat{\mathbf{n}}_A(\mathbf{c}_\phi)\|_2^2 .$$

Note that the cluster assignments $\mathbf{c}_\phi$ are a function of the classifier parameters. Our final

loss function with which we train the classifier parameters $\phi$ is

$$\mathcal{L}(\phi) = \mathbb{E}_{\mathbf{c}_\phi}[\|\log \mathbf{n}_A - \log \hat{\mathbf{n}}_A(\mathbf{c}_\phi)\|_2^2] - \alpha \sum_{i=1}^{\ell} \sum_{j=1}^{m} f_{\phi,j}(\mathbf{s}_i) \log f_{\phi,j}(\mathbf{s}_i) + \beta \sum_{j=1}^{m} \left[ \left( \sum_{i=1}^{\ell} f_{\phi,j}(\mathbf{s}_i) \right) - \frac{1}{m} \right]^2, \tag{5.42}$$

where the first term on the right hand side is the expected distortion, the second is the conditional entropy $H(C|Y)$, and the final term corresponds to the final term in Eq. 5.41. This final term encourages equal number of states to be assigned to each cluster while the conditional entropy term encourages deterministic bin assignments.

### 5.4.2 Optimization framework

Computing the expectation in the loss function in Eq. 5.42 was addressed in Sec. 5.3.1.1, but computing the trainable parameter gradients via backpropagation through an ODE solve can be quite expensive if performed naively. Therefore we employ NeuralODE [26] to efficiently compute the gradients required. Any black-box solver can be used to solve the ODE using NeuralODE, and the gradients are automatically computed. Our entire computational framework is built in Pytorch [158], which allows us to combine the adjoint-based gradient computation of NeuralODE with the automatic differentiation (AD) tools present in Pytorch. Additionally, using AD models allows any modules containing trainable parameters to be introduced at any location in the framework and gradients will be computed with respect to those parameters automatically. In the end, this allows for relatively simple and efficient implementation of complex and large scale gradient computations and allows for flexible and efficient framework adjustments.

Having an efficient method of computing gradients using AD is just one piece towards optimizing our classification model. However, this comes with another task: ensuring that the computational graph is not severed. For example, the summation conditions of the coarse-grained equations are non-differentiable due to the existence of the cluster assignments in the summation index. This issue is avoided by transforming the equations to a probabilistic description.

We initially leveraged the Gumbel-Softmax trick [190] to approximate the expectation in Eq. 5.42 in a differentiable manner. This was replaced in favor of the empirically more robust way of approximating the expectation with the method presented in Sec. 5.3.1.1

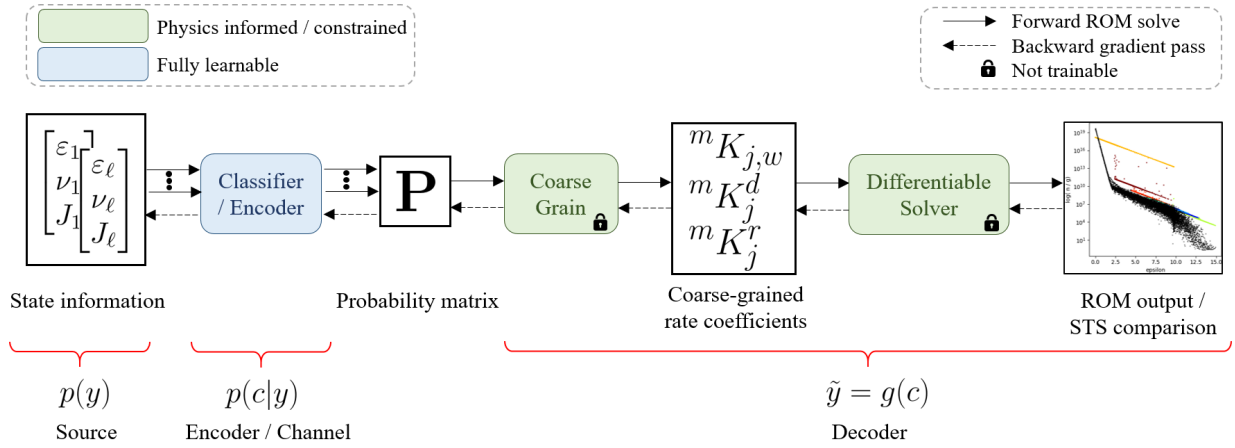



Figure 5.1: **Machine learning framework.** We train a classifier with a rate-distortion informed loss function to predict internal state cluster assignments. After predicting classes, we use the probabilistic description of coarse grained dynamics to approximate the state-to-state solution. Clear connections to Fig. 2.1 are illustrated.

#### 5.4.2.1 Tractability improvements

Even with an efficient method of computing gradients, the memory requirements can be too large to fit in modern hardware. To partially alleviate this, we make many tractability improvements to make the optimization process feasible.

In general, reaction rate coefficients vary with temperature. The full state-to-state rates are often modeled with an Arrhenius fit; however, the coarse-grained reaction rate coefficients are bi-variate functions of $T_{int}$ and $T$, as can be seen from Eqs. 5.18- 5.20, and can not be fit with uni-variate functions such as the Arrhenius law. A common approach for this issue is to compute the coarse-grained coefficients on a grid of $N_{T_{int}} \times N_T$ points, and interpolate on this grid during the forward solve. This greatly reduces the cost of computing the coarse-grained reaction rate coefficients by avoiding computing them at each temperature in the forward solve.

If we consider all outputs of the coarse-graining process (coarse-grained reaction rate coefficients) as $\mathbf{K}$, there are $(M^2 + 2M)N_{T_{int}}N_T$ values to compute. At each point in the temperature grid, there are $M^2$ excitation / de-excitation rates, $M$ recombination rates, and $M$ dissociation rates. Computing the gradients of $\mathbf{K}$ with respect to the cluster assignments $\mathbf{c}$ results in a matrix $\partial\mathbf{K}/\partial\mathbf{c} \in \mathbb{R}^{\ell \times (M^2+2M)N_{T_{int}}N_T}$

Conservative values for $N_{T_{int}}$ and $N_T$ are 10. With this temperature grid and 16 bins for the $N_2 + N$ system, the gradient $\partial\mathbf{K}/\partial\mathbf{c}$ will contain more than $2.7 \times 10^8$ values. Using 64 bit precision is required here to accurately compute state-specific partition functions. This results in well over 2GB of data that must be computed and stored at each optimization

iteration just for the forward pass.

Computing the expected distortion $\mathbb{E}[\mathcal{D}]$ and gradients of the ODE solve can also become quite expensive for a large number of time steps. The system is usually solved using adaptive time stepping methods which can result in $n_t > 10^3$ time steps until equilibrium is reached. The output of the ODE solve will therefore contain $N(M^2 + 2M)N_T N_{T_{int}} N_t$ values, which can be over 25GB using 64bit precision. The backward pass often requires far more memory, especially when considering that the adjoint system is considerably more stiff and requires many times more integration steps to evaluate accurately. This results in memory requirements per optimization iteration of nearly a terabyte. With potentially thousands of optimization iterations required, it is prohibitively expensive and infeasible on typical hardware.

Following the previous works on coarse-graining [169, 170, 173, 174], we can assume isothermal heat bath and constant internal temperature during the optimization process, specifically $T_{int} = T = T_0$ with $N_{T_{int}} = N_T = 1$, which greatly reduces the compute and storage costs of the framework. Additionally, only a subset of locations in time are used to compute the distortion. These simplifications combined greatly reduce the cost of computing and storing gradients by an estimated 3-6 orders of magnitude.

### 5.4.3 Stability Improvements

We transform the original system of equations for more stability during the ODE solve. These transformations are purely for computational convenience and do not alter the mathematical formulation of the dynamical systems. Both the number density and energy density equations are transformed for a more stable method of computing the solution. First, we solve for mass densities instead of number densities in Eq. 5.16. It is easy to convert between mass density and number density using the relationship $n_A = \rho_A / m_A$, where $m_A$ is the mass of a particle of species A. A substitution $n_A^j = \rho_A^j / m_B$ and $n_B = \rho_B / m_B$ is made; for simplicity, we always use the single particle mass to make the transformation. The resulting equation which we solve is instead

$$\frac{d\tilde{\rho}_A^j}{dt} = \sum_{w=1}^{m} (-{}^0\hat{K}_{j,w}\tilde{\rho}_A^j n_B + {}^0\hat{K}_{w,j}\tilde{\rho}_A^w n_B) + (-{}^0\hat{K}_j^d \tilde{\rho}_A^j n_B + {}^0\hat{K}_j^r \rho_B^3) \ , \tag{5.43}$$

where the rate coefficient matrices are simply scaled versions of the originals:

$${}^0\hat{K}_{j,w} = \frac{{}^0K_{j,w}}{m}, \quad {}^0\hat{K}_j^d = \frac{{}^0K_j^d}{m}, \quad {}^0\hat{K}_j^r = \frac{{}^0K_j^r}{m^2} \ .$$

This simple substitution greatly aids in computational stability and accuracy. Without it, scale differences in the rate coefficient matrices and number densities can be as high as 60 orders of magnitude. Solving the equations with such a large range of scales can cause memory issues and finite precision errors. With the transformation, the density and transformed rate coefficient matrices are closer to the same scale, alleviating some of such computational barriers.

Instead of solving the internal energy density dynamics (Eq. 5.6) and converting to temperature, we solve for internal temperature directly. Starting with the chain rule, we rewrite the time derivative of the energy density as

$$\frac{de}{dt} = \frac{de}{dT_{int}} \frac{dT_{int}}{dt} = c_v \frac{dT_{int}}{dt} \ ,$$

where $de/dT_{int} = c_v$ is the constant volume specific heat. Simply taking the derivative of Eq. 5.17 with respect to $T_{int}$, and using the relationship $\beta = 1/(k_B T_{int})$, the constant volume specific heat for cluster $j$ of species A is

$$c_{v,A}^j = \frac{{}^2Q_A^j {}^0Q_A^j - ({}^1Q_A^j)^2}{({}^0Q_A^j)^2 k T_{int}^2} \ , \tag{5.44}$$

where ${}^mQ_A^j$ is defined by Eq. 5.21. Finally, we solve for the temperature of each cluster directly by simulating

$$\frac{d\tilde{T}_{int,A}^j}{dt} = \frac{1}{c_{v,A}^j} \frac{d\tilde{e}_A^j}{dt} \ , \tag{5.45}$$

where $\tilde{T}_{int,A}^j$ is the internal energy of cluster $j$ of species A. We present here only the deterministic formulation of these transformations, but the approximated expectation using a probabilistic description is straightforward and follows the discussion of Sec. 5.3.1.1. Note that we do not solve Eq. 5.45 during training, but we do solve it when performing the final prediction with the optimized cluster assignments.

## 5.5 Numerical Results

The presented RD-informed loss function is demonstrated by training a classifier on two separate problems. The first is a simple 1D Gaussian source quantization problem for which the analytic RD curve is known. Training with our loss function can thus be demonstrated in a more controlled and efficient environment. The entire framework is then illustrated by clustering 9390 internal states of $N_2$ in a 0D reacting system of $N_2 + N$. All of the source

code for the following experiments is publicly available at https://www.github.com/christian-jacobsen/RDClustering

### 5.5.1 RD clustering analytic example

The quantization of a one-dimensional Gaussian source is a well studied problem [54], and one of the very few problems for which the analytic form of the rate-distortion curve exists. Consider samples $y$ from a 1D standard normal distribution such that $y \sim \mathcal{N}(0,1)$, and a squared-error distortion

$$D = (Y - \tilde{Y})^2.$$

We would like to *quantize* this source using some fixed number of bits $R = \log_2 m$. This means that there are $2^R$ elements in a *codebook* with which we can use to represent samples from the source. Alternatively, there are $m$ *clusters* which we can use in the quantization. Here we will consider a codebook $\mathcal{B} = \{1, \dots, m\}$ for simplicity, although any codebook with $m$ unique values can be used. Each sample from the source is assigned a particular cluster, and this cluster assignment is transmitted through a noisy channel. At the output of the channel, a decoder attempts to reconstruct the original sample from the received cluster assignment. There are two questions to answer:

1. On average, how well can the original signal be reconstructed?

2. How can the optimal encoding-decoding system be found?

We have already illustrated how rate-distortion theory provides an answer to the first question, and in this an analytic solution is known. The second question is one of optimization, and we can inform the optimization process through information theory, and in particular the conditions for optimality in Sec. 5.3.2.1.

The rate-distortion function for memoryless transmission of the Gaussian source is given by [54]

$$R(D) = \begin{cases} -\frac{1}{2}\log_2(D) & \text{if } 0 \leq D \leq 1 \\ 0 & \text{if } D > 1 \end{cases}. \tag{5.46}$$

Conversely, the distortion-rate function

$$D(R) = 2^{-2R} \tag{5.47}$$

is related to the number of clusters used in compression. We will approximately solve Eq. 5.39 using machine learning based methods to approximate functions $f_h$ and $g$ which minimize

distortion for some number of bins $m$. We will then compare the final trained models to the RD curve for this problem.

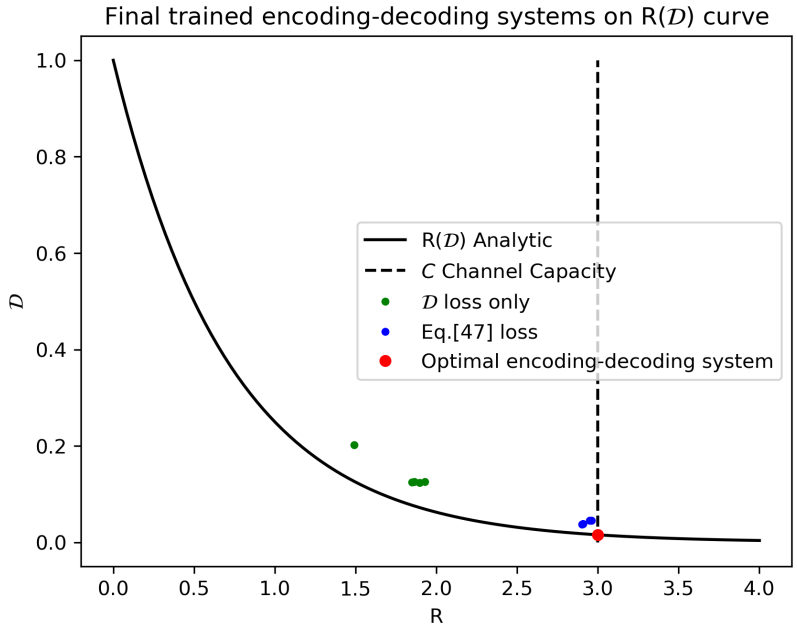



Figure 5.2: Learning an encoding-decoding system to quantize a 1D Gaussian source. Training with our RD-informed loss (Eq. 5.41) results in lower distortion over training with distortion only. The optimal point on the RD curve is highlighted in red.

We first parameterize a probabilistic encoding function $f_\phi : \mathbb{R} \to [0,1]^m$ using a simple fully connected feed-forward neural network which acts as the encoder, and transmission through a noisy channel. The network consists of 5 layers with 10 nodes per layer and a softmax layer at the end. The conditional distribution on the cluster assignment is given by

$$p(\mathbf{c}|y) = f_\phi(y)$$

where $\mathbf{c} \in \mathbb{R}^m$.

A deterministic decoding function $h_\psi : \mathcal{B} \to \mathbb{R}$ is also parameterized by a simple FNN with 5 hidden layers and 10 nodes per layer. There is no final activation function in the decoder to allow any continuous value to be reconstructed. However, note that there exist only up to $m$ *reconstruction points*. Although the decoding function is continuous, only the $m$ values in the codebook $\mathcal{B}$ are ever reconstructed. Thus, we will have at most $m$ reconstruction points.

The problem of quantization essentially consists of learning:

1. The *assignment regions*

2. The *reconstruction points*

We show that simply minimizing distortion (also called *reconstruction loss*) does not robustly and consistently find optimal encoding-decoding systems, but our formulation brings the learned encoding-decoding system significantly closer to the optimal system in the RD plane. Figure 5.2 shows the results of learning the encoding and decoding functions by minimizing both the distortion only and our loss formulation separately in Eq. 5.41. All model and training parameters are identical for each model trained, the differences are in the random initialization of the model parameters and the loss function trained with. The model has been trained 5 times for each loss function, and the rate and distortion computed using the final trained model with respect to the Gaussian source. The average distortion for models trained by minimizing distortion only is 0.1403 while the average distortion for models trained by minimizing Eq. 5.41 is 0.0412. The optimal distortion for an encoding-decoding system used to quantize a Gaussian source is 0.0156. This simply example illustrates that training using the RD-informed loss function can consistently achieve lower distortion than training with distortion only.

### 5.5.2 Optimal Clustering of $N_2 + N$ System

We implement the rate distortion based clustering optimization framework described in Sec. 5.4 to cluster $\ell$ states into $m$ bins. The coarse-graining process detailed in Sec. 5.2.2 is then used to form a reduced order system of equations based on the binning assignments.

The optimization is performed using a single solution of the STS master equations for a 0D $N_2 - N$ system reacting at 10,000K and 1,000Pa. The system is initially considered in equilibrium at 1,000K, and the temperature is instantaneously raised to 10,000K to begin the simulation. We set the initial molar fraction of N to 0.05. There are 9391 total degrees of freedom - 9390 rovibrational states of $N_2$ and the molar concentration of N. Full STS rate coefficients are obtained through rovibrational quasi-classical trajectory (QCT) calculations [191, 192], and the Plato library [175, 193, 194] is used to simulate the master equations. This is our only data sample, a single solution to the STS master equations.

A classification model is initialized as a simple fully-connected neural network with 4 layers, 10 nodes per layer, ReLU activation functions, and a softmax output layer. Three features for each internal state are input to the network: internal energy level $\varepsilon_i$, rotational quantum number $J_i$, and rotational quantum number $\nu_i$. An $m$-dimensional probability vector is output indicating the probability of an internal state of $N_2$ belonging to each of the clusters. The optimization objective in Eq. 5.42 is used to train the classifier. The distortion we assume here is the MSE between the N molar fraction of the STS solution and the coarse-grained solution based on the cluster assignments from the classifier prediction. We limit the

amount of STS data used in time to train the model. Only the first $t_f = 2 \times 10^{-6}$ seconds of the STS solution is considered during training. This distortion corresponds to an $L_2$ norm between the black and blue curves to the left of the vertical gray dashed lines in Fig. 5.4

#### 5.5.2.1 Optimization procedure

The optimization of such a high dimensional system contains many local minima. Here we have empirically found that two primary types of local minima exist in the loss landscape. The first type is characterized by a lack of using all clusters available to the classifier. For instance, $p(c_j|\mathbf{s}_i) \approx 0 \quad , \forall i$ for some cluster $j$. This violates condition (4) in Sec. 5.3.2.1, thus it is clear that this type of local minima is sub-optimal. One can 'escape' this type of local minimum by scheduling an increase in $\beta$ throughout training. A larger value of $\beta$ corresponds to encouraging an increase in conditional entropy $H(C|S)$, distributing states more evenly among the available clusters.

The second type of local minimum is characterized by low values of entropy $H(C|S_i)$ for some states. This indicates that the state $i$ has a relatively high probability of belonging to more than one cluster. This violates condition (2) in Sec. 5.3.2.1, and again this type of local minimum is sub-optimal. 'Escaping' this type of local minimum can be accomplished by introducing a scheduler for the $\alpha$ hyperparameter in our training loss.

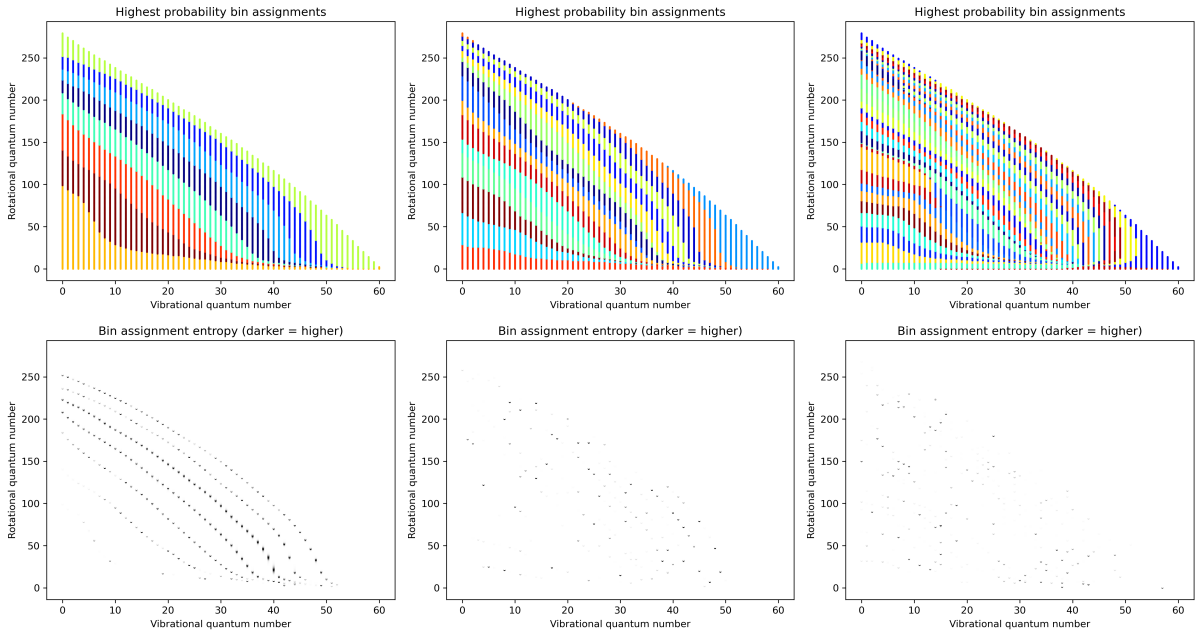
Our $\alpha$ and $\beta$ schedules follow typical $\beta$-type schedulers used in training variational autoencoders (VAEs) to prevent mode collapse [195, 74]. We linearly increase $\alpha$ from 0 to $10^{-3}$ over the course of 5,000 optimization iterations, followed by a repeat of this for another 5,000 optimization iterations then $\alpha = 10^{-3}$ until convergence. We follow a similar procedure for $\beta$, varying from 0 to $5 \times 10^{-4}$ over 7,500 iterations and hold at $5 \times 10^{-4}$ until convergence. The implemented $\alpha$ and $\beta$ schedules aid in avoiding the aforementioned local minimum solutions, but the model may still be stuck in a local minimum after the hyperparameter scheduling procedure. This may often be the case, but it can easily be alleviated with some additional manual scheduling of the hyperparamters $\alpha$ and $\beta$. If any clusters are unused, $\beta$ should be increased temporarily to encourage all clusters to be used. If many cluster assignments remain highly uncertain, $\alpha$ should be increased to reduce the uncertainty in cluster assignment predictions. Doing so will allow the optimization procedure to 'escape' local minima and continue with the optimization process.

Setting $\alpha,\ \beta = 0$ corresponds to minimizing distortion only without any enforcement of the conditions obtained from RD. In this setting, none of our experiments were able to improve upon the existing spectral clustering algorithm. However, when applying the aforementioned schedulers for $\alpha$ and $\beta$ during optimization, effectively leveraging the RD-informed loss function, cluster assignments which result in greatly reduced ROM error are

learned.

#### 5.5.2.2 Constant internal temperature optimization results

The optimization is performed assuming a constant internal temperature in the reduced-order system, set to the translational temperature of the system $T = 10,000K$. The optimization is far less memory-intensive than using variable temperature in the reduced-order system, as described in Sec. 5.4.2.1. However, the final learned cluster assignments are still valid when solving for the internal temperature. We perform experiments for varying number of clusters. In particular, we use $m = 8, 16, 32$.

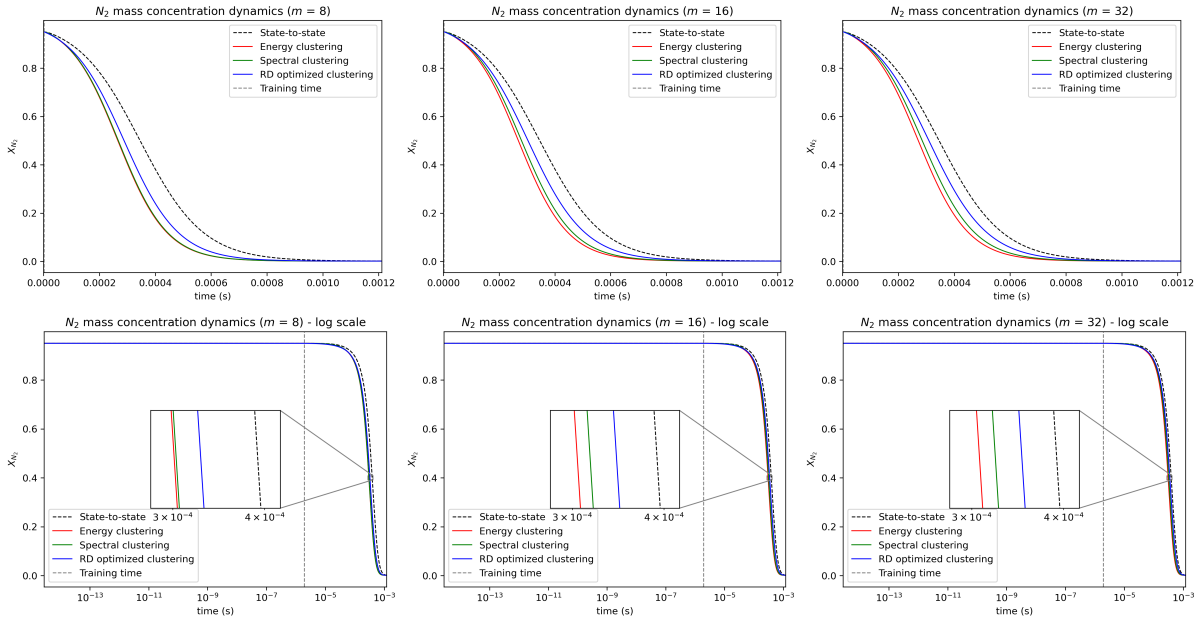



(a) **m = 8 clusters** (b) **m = 16 clusters** (c) **m = 32 clusters**

Figure 5.3: Final trained classifier predictions. *(upper)* Maximum probability cluster assignments for all states. *(lower)* Entropy of cluster assignment predictions for all states. White indicates that the cluster assignment is nearly deterministic.

Figure 5.3 shows the final trained classifier predictions for each value of $m$. We plot the vibrational (rotational) quantum number of each state on the x (y) axis with colors corresponding to the maximum probability cluster that the state belongs to, predicted by the trained classifier. We also show the entropy of $p(\mathbf{c}_i|\mathbf{s}_i)$ for each state, which is an illustration of the degree to which the classifier deterministically predicts cluster assignments. Darker colors indicate a higher entropy (or uncertainty) in the cluster assignment predictions. There is clearly some structure in the cluster assignment predictions, and the uncertainty is larger

on the boundaries of 'decision regions' where cluster assignments change. However, for the overwhelming majority of states, the entropy $H(C|S_i)$ is nearly zero.




(a) **m = 8 clusters** (b) **m = 16 clusters** (c) **m = 32 clusters**

Figure 5.4: *(upper)* Dynamics of molar fraction evolution for $N_2$. Note the training time line near the y-axis. *(lower)* In log scaled time.

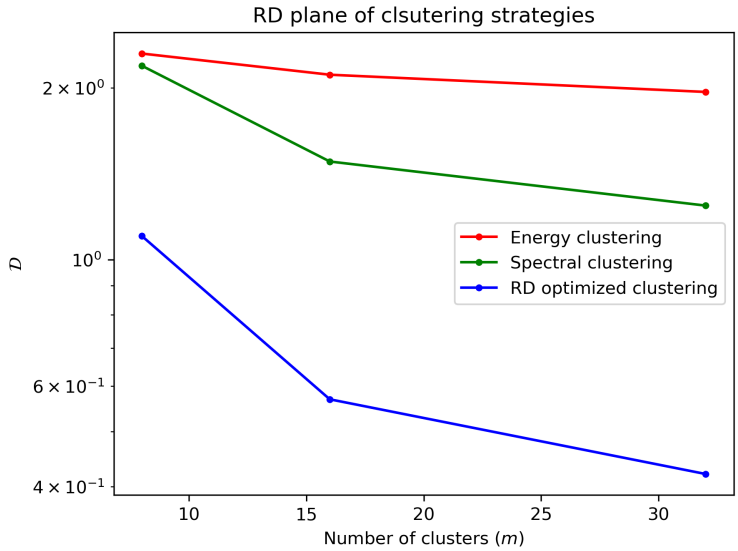



Figure 5.5: RD plane comparison of clustering strategies (lower is better).

We also illustrate in Fig. 5.4 how the final predictions compare to other clustering strate-

gies described in Section 5.2.3. Qualitatively, the results obtained from our method are closer to the STS solution than other previous methods. Quantitatively, we plot energy-based clustering, spectral clustering, and our RD-informed clustering on the rate-distortion (RD) plane in Figure 5.5. The number of bins are plotted on the x-axis, and the distortion is plotted on the y-axis. Lower distortion is desirable, thus our method clearly outperforms the other two methods shown. Additionally, the distortion also decreases at a faster rate with increased number of clusters using the RD-informed method compared to energy-based clustering and spectral clustering.

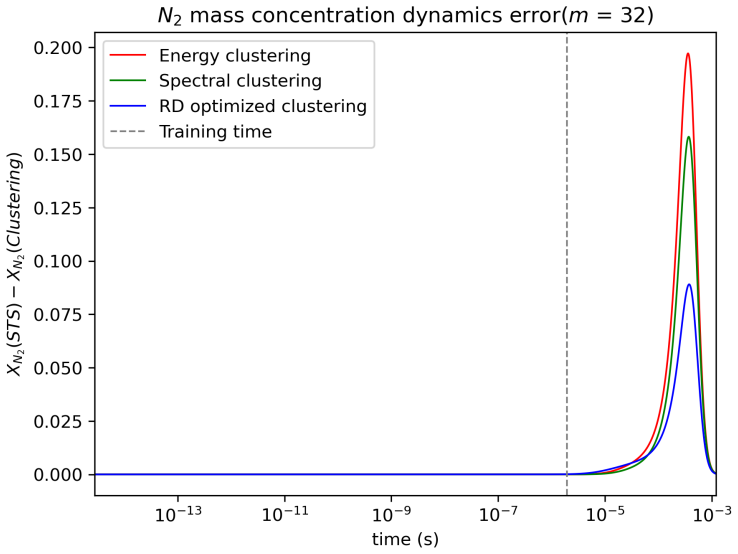



Figure 5.6: Prediction error as a function of time for $m = 32$ clusters (lower is better).

Finally, we show the error with respect to the STS model as a function of time for prediction on the $m = 32$ cluster system in Figure 5.6. The RD-informed clustering shows improved performance compared to the other methods on nearly the entire prediction time, with the exception of a small window of time at the onset of prediction.

## 5.6 Summary

In this chapter, a framework is presented to integrate ML techniques with the principles of RD theory to enhance reduced-order modeling of non-equilibrium gas dynamics. This approach leverages the maximum entropy principle and RD theory to navigate the complexities of optimizing cluster assignments, thereby capturing the dynamics of systems with greater accuracy. By implementing a fully-differentiable dynamics solver, which includes Neural ODEs and adjoint methods, this framework facilitates the efficient learning of cluster assignments

through gradient-based optimization. Additionally, the implementation details discussed and modifications incorporated into the optimization process are critical to rendering the optimization feasible. The final cluster assignments learned by the classifier demonstrate state of the art performance in terms of accuracy in the reduced order modeling of the master equations describing state-to-state dynamics.

The presented results demonstrate that training the classifier assuming constant internal temperature results in improved ROM performance; however, optimizing the classifier with the internal energy density equations included may lead to a further increase in performance while additionally improving performance across a range of initial conditions. Doing so poses some additional challenges partially discussed in Section 5.4.2.1. One such hurdle is to massive leap in memory requirements due to computing a grid of coarse-grained rate coefficients in internal temperature space. This requires 2-3 orders of magnitude more memory to compute just in the forward pass, and even more in the gradient computation. Performing such an optimization is a significant challenge which requires many more implementation optimizations and perhaps some new techniques.

A goal of future works is to develop a surrogate capable of dynamically adjusting both cluster assignments and the number of clusters required to represent the true dynamics in real time. However, we note that the work presented in this chapter is a first step towards that goal. Generalizing the classifier to a wider range of initial conditions is of primary importance to improve accuracy broadly across a range of pressure and temperature values. It may be possible to generalize the classifier to other species, provided enough data is available.

The incorporation of ML techniques, while beneficial for the accuracy of the ROMs, also introduces a trade-off in terms of interpretability. Specifically, the complexity of the learned cluster assignments, despite their improved accuracy, challenges their straightforward interpretability, highlighting a common theme in the intersection of ML and physics: the balance between enhancing model performance and maintaining clarity in the model's physical underpinnings. However, the reduced order model itself is interpretable and physically consistent by construction, and so it is deemed acceptable in this work to sacrifice a small portion of interpretability in the model for great increases in accuracy.

This exploration contributes to the thesis's broader objective of demonstrating the potential and limitations of applying ML to physical modeling. Our findings emphasize the promise of ML in advancing the capabilities of physical models, albeit with considerations regarding the interpretability of such models. As we advance beyond application-specific works towards a more general framework for solving forward and inverse problems, this chapter marks a step towards the integration of ML in physics, demonstrating a nuanced

approach where ML complements and enriches physical modeling practices.

# CHAPTER 6

# Physically Consistent Diffusion Model Sampling for Solving Forward and Inverse Problems

This chapter marks a slight transition, venturing from specific case studies of physics-aware ML towards a more generalized framework designed to tackle the nuanced challenges of solving forward and inverse problems using physics-aware ML. The focus of this chapter is twofold. Firstly, it underscores the unique position of score-based generative models as a powerful tool for solving forward and inverse problems, especially those embedded with probabilistic uncertainties. These models, renowned for their success in fields such as image generation and discussed in Section 2.3.2, encounter a distinct set of challenges when applied to physical problem-solving. The crux of these challenges lies in the necessity for generated samples to not only be quantitatively rigorous but also to strictly adhere to the physical laws governing the system of interest. Our aim is towards developing a methodology that ensures every sample generated by these models remains faithful to the specified governing physical equations, thereby mitigating the risk of predicting non-physical behaviors.

Despite the promising initial applications of these models in fluid dynamics and other areas governed by partial differential equations (PDEs), there exists a notable gap in ensuring their adherence to physical laws. Yang et al. develop a surrogate model named FluidDiff [196] based on diffusion models to predict flows governed by the 2D incompressible Navier-Stokes equations given a source function. The model is shown to achieve greater accuracy in prediction over other methods such as conditional GANs [82], U-Nets [197], and even physics-informed neural networks [46] (PINNs) in some cases. The primary metric of comparison is the root mean-squared error (RMSE) without emphasis on the adherence of predictions to the physical equations. Ultimately, without enforcement of the physical equations, the model may inevitably predict non-physical behavior. Another work by Shu et al. [198] develops an iterative method for performing field reconstruction from sparse mea-

surements on 2D flows governed by the Kolmogorov flow equations. The model is based on conditional DDPMs in which the conditioning information is a low resolution guess of the full field along with gradients of the physical residual. Accurate high-fidelity reconstructions are observed along with reasonably good adherence to the physical equations of interest; however, there is no enforcement of physical behavior. Residual gradients are included as conditioning information, but there is no enforcement towards minimizing the residual during sampling. This in turn also allows for the model to potentially predict non-physical behavior. These observations underline the essence of our work: to establish a robust framework that not only enhances the accuracy and applicability of score-based generative models but also guarantees their compliance with physical laws.

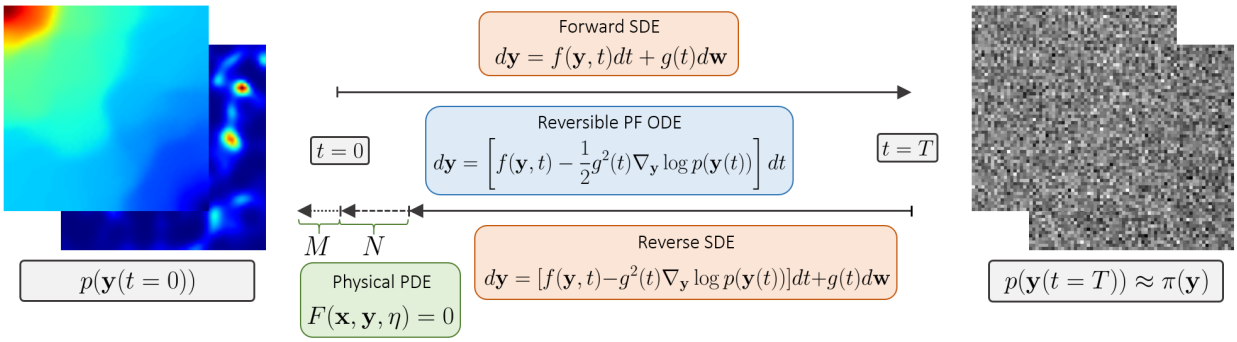



Figure 6.1: We propose a method of injecting the governing equations into the sampling process of score-based generative models to enforce consistency of samples with the underlying PDE.

Furthermore, we delve into the intricacies of our proposed framework (Figure 6.1), emphasizing its novelty in enforcing physical consistency during the sample generation phase. This approach is distinguished from prior methods, and in particular PINNs, in many ways. Three primary differences are: i) The goal is to not solve a PDE, but rather to generate samples constrained by them; ii) Rather than using the PDE residual during the training stage, the residual is utilized during the sample generation stage; iii) Rather than relying on the continuous form of the residual, a discrete form is utilized based on the defined data.

We further remark that the work by Chung et al. [121] is most closely related to some aspects of the work presented here, although applied to image reconstruction. Corrector steps are applied during sampling from the score-based model to encourage sample consistency with a linear system modeling the data. Our work similarly contains correction steps; however, the nature of the correction steps is quite different and achieved in a different manner. The 'correction steps' in our work constitute minimizing the residual from general nonlinear physical PDEs rather than a linear system to perform image reconstruction.

As we navigate through the example application of this framework (2D predictions governed by Darcy's law), we illustrate the framework's effectiveness in predicting fluid flows and reconstructing field data. These examples not only showcase the framework's capacity to ensure physical consistency but also highlights its utility in both forward and inverse tasks. A flexible form of model architecture is additionally demonstrated such that architectural augmentations can be rapidly trained for particular forward or inverse problems based on a baseline pretrained model.

Score based generative models can effectively solve forward and inverse problems, and with improvements to the generative sampling process, they may be able to solve such problems with state of the art efficiency. They are also well suited for solving very high dimensional inverse problems (which are notoriously expensive to solve) with expressive and unconstrained distributions. New works such as consistency trajectory models [104] and higher order solvers [199] have demonstrated remarkably faster sampling strategies which are not included in this work - strategies which could provide orders of magnitude speedup in sampling time, resulting in state of the art flexibility and efficiency in solving forward and inverse problems with score-based generative models.

In conclusion, Chapter 6 not only encapsulates the synthesis of machine learning and physical sciences explored throughout this thesis but also sets the foundation for future research at their intersection. By demonstrating the effectiveness and flexibility of score-based models in solving forward and inverse problems while adhering to physical laws, this chapter paves the way for a broader application of these models, ensuring that the future of computational physics is one where accuracy does not come at the expense of physical integrity.

## 6.1 Enforcing Physical Consistency and Conditioning

This chapter considers solving forward and inverse problems governed by general steady-state partial differential equations (PDEs) of the form

$$F(\mathbf{x}, \mathbf{y}, \boldsymbol{\eta}) = 0 \,, \tag{6.1}$$

where $\mathbf{x} \in \mathbb{R}^2$ (2D) are the spatial coordinates, $\mathbf{y} \in \mathcal{X}$ are the physical variables of interest, and $\boldsymbol{\eta} \in \mathbb{R}^p$ are parameters describing the physical system. In our examples, solutions $\mathbf{y}$ are computed on a discretized spatial domain such that a single solution – or data sample – is given on a 2D physical domain $\mathcal{X} = \mathbb{R}^{n \times n}$. We remark that constraining samples to follow the governing PDE is done **only at inference/sample time, and does not have**

**any effect on the training procedure.** This allows for much greater flexibility without the need for re-training the model. We first discuss training a score-based generative model before describing our approach to enforcing physical consistency in generated samples.

### 6.1.1 Unconditional Model Training

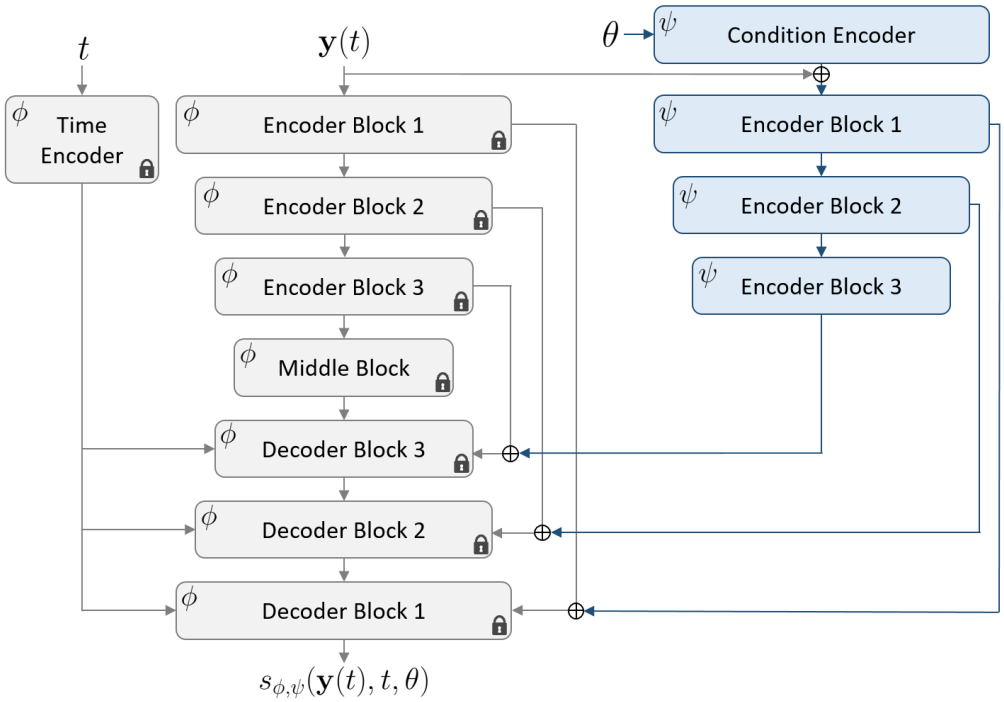



Figure 6.2: ControlNet-type architecture based on a UNet-type unconditional model architecture. The conditional model contains a fixed pretrained unconditional model with parameters $\phi$ and a conditional model augmentation with trainable parameters $\psi$.

We train a convolution-based UNet-type [197] architecture to approximate the score function in time with the weighting function set to a constant $\lambda(t) = 1$ (see Section 2.3.2 for details). The overall architecture for both our unconditional and conditional generative models is illustrated in Fig. 6.2. Our choice of $\mathbf{f}(\mathbf{y}, t)$ and $g(t)$ in the SDE form correspond to the variance-preserving (VP) SDE [99] in which

$$\mathbf{f}(\mathbf{y}, t) = -\frac{1}{2}\beta(t)\mathbf{y}, \text{ and} \tag{6.2}$$

$$g(t) = \sqrt{\beta(t)}\,, \tag{6.3}$$

giving a final forward SDE form of

$$d\mathbf{y} = -\frac{1}{2}\beta(t)\mathbf{y}dt + \sqrt{\beta(t)}dw \ . \tag{6.4}$$

Further, we define $\beta(t)$ as a linear function with two hyperparameters:

$$\beta(t) = \beta_{min} + (\beta_{max} - \beta_{min})t \ , \tag{6.5}$$

where $\beta_{min}, \beta_{max} > 0$ and $\beta_{max} > \beta_{min}$. This SDE form induces a transition kernel of

$$p(\mathbf{y}(t)|\mathbf{y}(0)) = \mathcal{N}\left(\boldsymbol{\mu}(\mathbf{y}, t), \boldsymbol{\Sigma}(t)\right) \ , \tag{6.6}$$

where

$$\boldsymbol{\mu}(\mathbf{y}, t) = \mathbf{y}(0) \exp\left[-\frac{1}{4}t^2(\beta_{max} - \beta_{min}) - \frac{1}{2}t\beta_{min}\right] \tag{6.7}$$

$$\boldsymbol{\Sigma}(t) = \mathbf{I}\left(1 - \exp\left[-\frac{1}{2}t^2(\beta_{max} - \beta_{min}) - t\beta_{min}\right]\right) \tag{6.8}$$

With a Gaussian transition kernel, the score function can be computed analytically such that we train with the following loss function:

$$\min_{\theta} \mathbb{E}_{t\sim\mathcal{U}[0,T]}\left[\mathbb{E}_{p(\mathbf{y}(0))p(\mathbf{y}(t)|\mathbf{y}(0))}[||s_\phi(\mathbf{y}(t), t) + \mathbf{z}\boldsymbol{\Sigma}^{1/2}(t)||_2^2]\right] \ , \tag{6.9}$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\mathbf{z} \in \mathcal{X}$. During training, we set $\beta_{min} = 1 \times 10^{-4}$, $\beta_{max} = 10$, and $T = 1$ for all of our experiments.

The optimal values for hyperparameters $\beta_{min}$ and $\beta_{max}$ may be application dependent. The value $\beta_{min}$ is related to the minimum noise level of the data during training. It should be set to a small value ($\beta_{min} << 1$) to allow the model to learn what the data distribution looks like without noise added. The value of $\beta_{max}$ is thus related to the maximum noise level of the data during training. If it is too small, generated samples will likely show a lack of diversity and may be constrained to a single mode. If it is too large, the model will struggle to fully denoise samples such that the generated samples are very noisy. Beginning with the hyperparameter values discussed in this work, is suggested that the values of $\beta_{min}$ and $\beta_{max}$ are tuned for each application and model.

### 6.1.2 Learning Conditional Score Functions

Unconditional generators have diverse applications, such as sampling from $p(\mathbf{y})$ for data augmentation, distribution learning, and anomaly detection. Moreover, they can serve as a robust foundation for constructing conditional generative models. Such models aim to sample from $p(\mathbf{y}|\boldsymbol{\theta})$, where $\boldsymbol{\theta} \in \mathbb{R}^s$ represents conditioning information, guiding the sampling process. Integrating conditioning into existing unconditional score-based generative models extends their utility, particularly in generating physical fields based on various input conditions like generative parameters, boundary conditions, partial field measurements, or macroscopic quantities. A conditional score-based generative model has the potential to undertake tasks like field reconstruction, field inversion, and effective probabilistic surrogate modeling. In some instances, an approximate conditional sampling can be achieved from a pre-trained unconditional model without additional training or conditional modeling. However, in most cases, generating samples from the desired conditional distribution requires extra data, modeling, and/or training efforts.

In scenarios where an analytical approximation to the conditional score function is unavailable, we resort to training a conditional model to approximate the conditional score function. Developing a conditional score-based generative model requires not only data samples $\mathbf{y}$ but also corresponding conditional information $\boldsymbol{\theta}$ for each sample, forming pairs $\{\mathbf{y}^{(i)}, \boldsymbol{\theta}^{(i)}\}_{i=1}^{N}$ in the dataset. The conditioning information could be a macroscopic quantity derived from the physical field, the generative parameters defining the physical field, or partial field measurements, among other possibilities. With these data samples, a conditional generative model aims to sample from the true conditional distribution $p(\mathbf{y}|\boldsymbol{\theta})$.

Since the forward Stochastic Differential Equation (SDE) noising process defined by Eq. 6.4 is Markovian, it remains independent of conditioning, even when conditioning information is available. The forward process stays the same: $p(\mathbf{y}(t)|\mathbf{y}(0), \boldsymbol{\theta}) = p(\mathbf{y}(t)|\mathbf{y}(0))$. However, the score approximation model $s_\phi(\mathbf{y}(t), t, \boldsymbol{\theta})$ is designed to accept conditioning information as an additional input. In the general conditional case, the score approximation model is trained by

$$\min_{\phi} \mathbb{E}_t \left[ \lambda(t) \mathbb{E}_{p(\mathbf{y}(0),\boldsymbol{\theta}) p(\mathbf{y}(t)|\mathbf{y}(0))} [||s_\phi(\mathbf{y}(t), t, \boldsymbol{\theta}) - \nabla_{\mathbf{y}(t)} \log p(\mathbf{y}(t)|\mathbf{y}(0), \boldsymbol{\theta})||_2^2] \right] , \tag{6.10}$$

effectively embedding conditioning information into the score approximation to learn $\nabla_{\mathbf{y}} \log p(\mathbf{y}(t)|\boldsymbol{\theta})$.

Training a full-scale generative model is often quite expensive and can be difficult to properly tune with optimal hyperparameters, architectures, and training schedules. However, in some scenarios in which multiple downstream objectives are of interest from the same appli-

cation, we aim to train a single unconditional model and augment the unconditional model with smaller conditional models which can be easily connected and disconnected to facilitate generation using multiple different types of conditioning. To achieve this, we turn towards recent advancements in conditional score-based generative modeling and adapt them for our applications. In particular, we leverage ControlNet [117], a form of model architecture specifically designed to augment pretrained unconditional models with conditional generative capabilities. The idea of ControlNet is to freeze the pretrained unconditional model and connect it to a smaller trainable model which will incorporate the conditional information. The overall model architecture including the unconditional model and conditional augmentation which we employ is shown in Fig. 6.2. The conditional portion of the model to be trained is initialized with special zero-convolution [117] layers such that the model will produce unhindered unconditional samples at the onset of training. Thus, if a pretrained unconditional model is validated to generate physical fields which are consistent with the underlying physical PDE, the conditional model is already guaranteed to produce physically accurate solutions at the onset of training.

During training, the unconditional model parameters are frozen. The conditional augmentation is trained using a very similar objective function to Eq. 6.10 with the main exception that the conditional score function approximation $s_{\phi,\psi}(\mathbf{y}(t), t, \boldsymbol{\theta})$ contains non-trainable parameters $\phi$ and trainable parameters $\psi$. The training objective we train conditional augmentations with therefore becomes

$$\min_{\psi} \mathbb{E}_t \left[ \lambda(t) \mathbb{E}_{p(\mathbf{y}(0),\boldsymbol{\theta}) p(\mathbf{y}(t)|\mathbf{y}(0))} [||s_{\phi,\psi}(\mathbf{y}(t), t, \boldsymbol{\theta}) - \nabla_{\mathbf{y}(t)} \log p(\mathbf{y}(t)|\mathbf{y}(0), \boldsymbol{\theta})||_2^2] \right] . \tag{6.11}$$

### 6.1.3 Sampling

Sampling from the trained score-based generative model is achieved by solving the reverse SDE or PF ODE backward in time. In our work, the particular form of reverse SDE solved during sampling is given by

$$d\mathbf{y} = \left[ -\frac{1}{2}\beta(t)\mathbf{y} - \beta(t) s_\phi(\mathbf{y}(t), t) \right] dt + \sqrt{\beta(t)} dw , \tag{6.12}$$

and the PF ODE by

$$d\mathbf{y} = \left[ -\frac{1}{2}\beta(t)\mathbf{y} - \frac{1}{2}\beta(t) s_\phi(\mathbf{y}(t), t) \right] dt . \tag{6.13}$$

These equations can be easily constructed from Eqs. 2.22, 2.26, 6.4, and 6.5. We note that the unconditional score function approximation $s_\phi(\mathbf{y}(t), t)$ is replaced with the conditional approximation $s_{\phi,\psi}(\mathbf{y}(t), t, \boldsymbol{\theta})$ in Eqs. 6.12 and 6.13 to achieve conditional sampling.

We assume a prior distribution of $\pi(\mathbf{y}) = \mathcal{N}(0, \mathbf{I})$. The transition kernel in Eq. 2.24 approaches this prior at $t \to \infty$, but with large enough difference in $\beta_{max} - \beta_{min}$, the transition kernel at $t = T = 1$ will be close to the selected prior. After sampling from the prior, this becomes the initial condition of the reverse SDE or reverse PF ODE. Solving either of Eqs. 6.12 or 6.13 from $t = 1$ to $t = 0$ will result in a sample drawn from the original data distribution $p(\mathbf{y}(0))$, as long as the score function approximation is accurate. These equations can be solved using any discretized SDE or ODE solver of choice. For simplicity, we opt to solve the reverse SDE in Eq. 6.12 using the first-order accurate Euler-Maruyama (EM) [200] method, and the PF ODE in Eq. 6.13 with the first-order accurate forward Euler (FE) scheme. Defining $\tau$ as the number of discrete time steps, this corresponds to a timestep of $\Delta t = T/\tau$.

To enforce physical consistency, we turn towards modifying the sampling process after training by reducing the physical residual $r$ at each point in the computational domain, where

$$\mathbf{r} = F(\mathbf{x}, \mathbf{y}, \boldsymbol{\eta}) \ . \tag{6.14}$$

The residual constitutes computing the PDE operator on a **discretized** approximate solution to the PDE. In practice, this residual will be nonzero, indicating that the discretized physical field $\mathbf{y}$ does not perfectly satisfy the physical equations. Our goal is to generate samples from a score-based generative model which minimizes this residual. Generated samples should contain residuals which are similar to the residual obtained by solving the PDE using traditional means. To reduce the residual at the final time of the sampling process, we aim to minimize the following expression at each point in the computational domain:

$$\min_{\mathbf{y}(t=0)} \|\mathbf{r}\|_2^2 \ .$$

To this end, we propose appending a residual minimization step to the SDE or ODE solver for the last $N$ of $\tau$ time steps, aiming to minimize $\|\mathbf{r}\|_2^2$ by appending a small step in the negative gradient direction $\nabla_{\mathbf{y}} \|\mathbf{r}\|_2^2 = 2\mathbf{r}\nabla_{\mathbf{y}}\mathbf{r}$. This requires evaluating the full order operator of the PDE and gradient of the residual by updating the sample according to

$$\mathbf{y}_{i-1} = \text{Solver}(\mathbf{y}_i, t_i) - 2\epsilon\mathbf{r}\nabla_{\mathbf{y}}\mathbf{r} \ , \tag{6.15}$$

where $\epsilon$ is the residual step size hyperparameter and $\text{Solver}(\mathbf{y}_i, t_i)$ indicates a step using the particular solver (EM for reverse SDE or FE for PF ODE). The residual is computed at each point in the computational domain, and each value is updated via physical consistency steps. After the reverse SDE process is solved to $t = 0$, we propose performing an additional

$M$ physical consistency steps by iteratively updating the final sample according to

$$\mathbf{y}_{i-1} = \mathbf{y}_i - 2\epsilon \mathbf{r} \nabla_{\mathbf{y}} \mathbf{r} \ .$$

The sampling process is flexible, with tunable hyperparameters $\epsilon$, $\tau$, $N$, and $M$, which can balance the need for efficiency over the degree to which samples follow the governing equations. An in-depth investigation into the effects of these hyperparameters is performed in Sec. 6.2.2. In our experiments, we find empirically that setting $\epsilon$ too large or too small results in unimproved sampling. We use $\epsilon = 2 \times 10^{-4} / \max \nabla_{\mathbf{y}} \mathbf{r}$ during sampling in all of our experiments, though we suspect that this form may not be optimal for all PDEs and datasets.

However, the residual minimization step appended to the solver can be implemented for any governing PDE. As the training process is not altered, a trained generative model is expected to produce samples which are relatively close to satisfying the equations. Thus, with an accurately trained generative model, relatively few physical consistency steps will be required to greatly reduce residuals and bring the generated samples closer to adhering to the physics. We illustrate this process on an example from fluid dynamics involving flow through porous media.

## 6.2 Unconditional Generation of Darcy Flow Fields

Before addressing forward and inverse problems, we first demonstrate the capabilities of physical consistency sampling by training an unconditional model. Samples from the unconditional model are distributed approximately as the true data distribution. However, we find empirically that such samples will generally not satisfy the PDE. It is possible that developing more advanced architectures and finely tuning the training procedure may result in consistently small physical residuals, but we illustrate that the use of physical consistency sampling negates this need altogether. Our primary demonstrative example is that of Darcy flow (see Section 3.2).

We employ two training and test datasets throughout the work. The first training dataset contains 10,000 samples generated by solving the Darcy flow equations with an intrinsic dimensionality of $s = 16$ on a discretized grid of $n = 64$. The generative parameters corresponding to each sample are randomly generated according to $p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ and saved to be used in conditional experiments. An additional 1,000 samples are generated and withheld from training as a test set used only in conditional experiments. The second training and test sets are generated with an intrinsic dimensionality of $s = 256$, keeping all other parameters

identical to the $s = 16$ dataset.

### 6.2.1 Residual Computation

Implementing the score-based generative model requires a method of evaluating the degree to which generated samples satisfy the Darcy flow equations. One way of doing this is to evaluate the PDE residual (Eq. 6.16) on generated samples. Note that the integral condition of Eq. 3.8 can be easily satisfied for any generated sample by regularizing the intermediate output $\tilde{\mathbf{p}}$ according to

$$\mathbf{p} = \tilde{\mathbf{p}} - \int_{\mathcal{X}} \tilde{\mathbf{p}} d\mathbf{x} \, .$$

We thus ignore this as part of the residual computation and satisfy the integral condition by construction. The residual is therefore considered only to be

$$\mathbf{r} = K(\mathbf{x}) \frac{\partial^2 p(\mathbf{x})}{\partial x_1^2} + \frac{\partial K(\mathbf{x})}{\partial x_1} \frac{\partial p(\mathbf{x})}{\partial x_1} + K(\mathbf{x}) \frac{\partial^2 p(\mathbf{x})}{\partial x_2^2} + \frac{\partial K(\mathbf{x})}{\partial x_2} \frac{\partial p(\mathbf{x})}{\partial x_2} + f(\mathbf{x}) \, , \tag{6.16}$$

following Eq. 3.6. This residual is a function of the spatial coordinate $\mathbf{x}$ and can be approximated at each point in the computational domain given $K(\mathbf{x})$ and $p(\mathbf{x})$. However, when applying physical consistency steps according to Eq. 6.15, we apply the residual gradient only to pressure fields, avoiding direct updates of the permeability field.

When solving the Darcy flow equations as outlined above, the gradients $\partial p / \partial x_1 = 0$ and $\partial p / \partial x_2 = 0$ are approximately satisfied on the domain boundaries by construction of the linear system. However, in computing the residual, we must compute gradients on these boundaries. We therefore use second order forward and backward difference approximations to the first and second order derivatives on the boundaries to compute the residual. This facilitates approximating the residual on both data generated by solving the linear system $\mathbf{Ap} = \mathbf{f}$ and samples obtained from the generative model by solving the reverse SDE.

### 6.2.2 Physically-Consistent Unconditional Generative Model

Constraining generated samples to follow physical equations as described in Sec. 6.1 is quite flexible and inherently contains hyperparameters which can be set and altered after training. In particular, the number of discrete time steps $\tau$ used in the solver, the number of residual minimization steps $N$ prior to $T = 0$, and the number of residual minimization steps $M$ after solving the reverse SDE or reverse PF ODE all play an important role in enforcing the physical laws which samples are to follow. We demonstrate our method of enforcing physical consistency by training an unconditional generative model on the Darcy flow training dataset

with dimensionality $s = 16$.

#### 6.2.2.1 Training

Our unconditional model architecture is illustrated as the left side of Fig 6.2. Each of the components labeled with parameters $\phi$ constitute the unconditional generative model. Additional architectural, training, and hyperparameter details are provided in Appendix D.1.

#### 6.2.2.2 Physically-Consistent Sampling

After training, the sampling process is quite flexible as discussed in 6.1.3. Sampling amounts to solving Eq. 2.23 or 2.26 backwards in time augmented with physical consistency steps, but the particular method of solving has a large impact on the quality of the final samples at $t = 0$. We investigate the effect that each equation and various combinations of hyperparameters have on the average physical residual for generated samples.

Higher-order solvers can also be used to provide gains in sampling efficiency [199], but is outside the scope of this work. Instead, we aim to minimize sample residuals without an initial emphasis on sampling efficiency. The use of physical consistency steps defined in Sec. 6.1.3 along with the number of time steps used to solve either the reverse SDE or PF ODE constitute our investigation into reducing the physical residual of generated samples.

When implementing physical consistency steps to reduce the PDE operator residual, we exclusively apply the steps to the pressure component of the samples. This deliberate choice ensures that no adjustments are made to the permeability field. Since the pressure field is a deterministic function of the permeability field, we refrain from directly modifying the permeability field generated by the model. Instead, we allow the model to fully generate the permeability field without interference.

Sampling is performed by varying $\tau$, $N$, and $M$ and solving either the PF ODE or the reverse SDE to generate a synthetic dataset of 1,000 samples. For each dataset, the average physical residual is computed as outlined in Sec. 6.2.1. The primary results of this experiment are illustrated in Fig. 6.3, where $p_D$ denotes the training data distribution and $p_S$ denotes the distribution of 1,000 samples generated according to the indicated method.

Initially, we observe that the choice of the number of discrete timesteps $\tau$ is crucial when employing the Euler-Maruyama method for solving the reverse SDE or the forward Euler method for solving the PF ODE. Insufficiently small values of $\tau$ result in high errors in approximating the dynamics described by Eqs. 6.12 and 6.13, leading to suboptimal sampling quality and increased physical residuals. In both scenarios, solving either the reverse SDE or PF ODE with a value of $\tau$ which is too small renders physical consistency enforcement

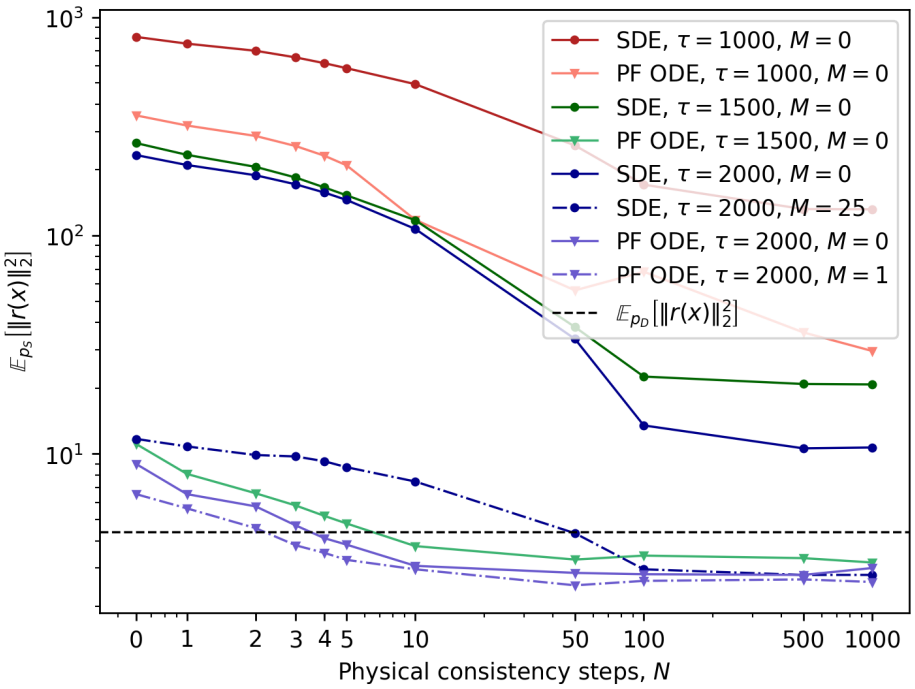


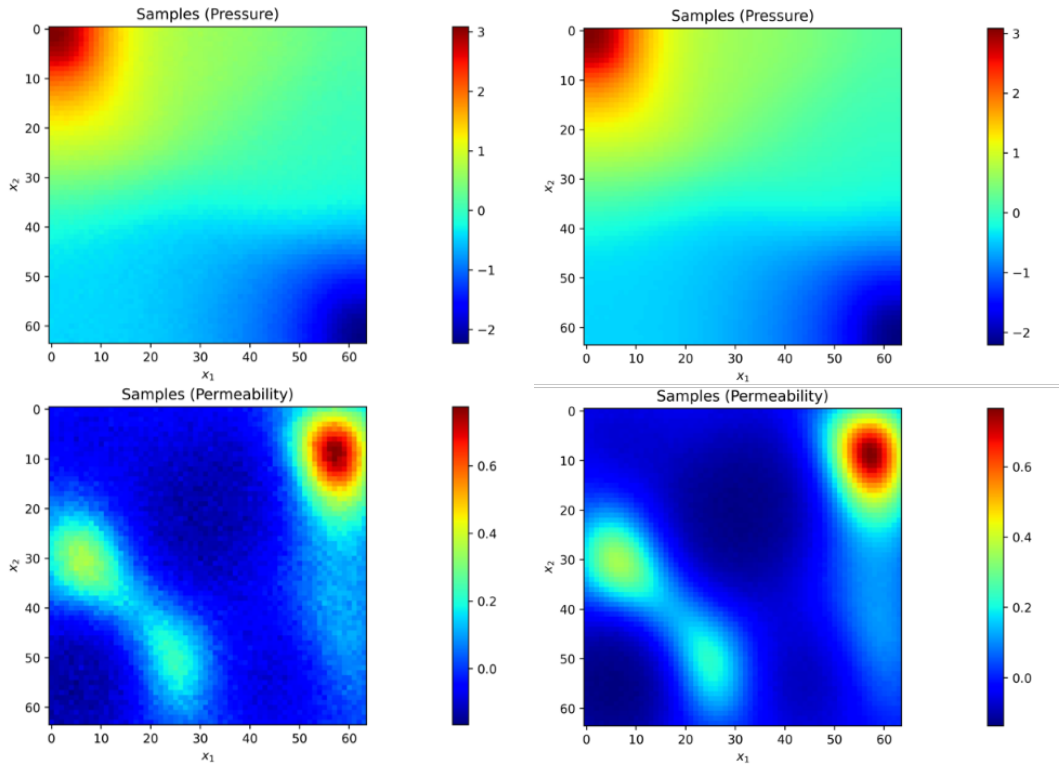

Figure 6.3: Darcy flow equations are enforced through physical consistency sampling. Hyperparameters $\tau$, $N$, and $M$ as well as the sampling equation have a large impact on physical residuals.

incapable of reducing residuals in generated samples to match the quality of the training dataset.

However, with a sufficiently large value of $\tau$, physical consistency can effectively reduce residuals on the generated datasets to be equal to or even lower than those in the training dataset. In the context of solving the reverse SDE, the stochastic nature of the process necessitates large numbers of physical consistency steps to reduce residuals towards the training dataset levels. However, when solving the PF ODE, relatively fewer physical consistency steps are required. The PF ODE generally yields significantly smaller residuals on generated samples compared to the reverse SDE, regardless of the presence of physical consistency steps. The additional noise provided by solving the reverse SDE may be beneficial in some cases in which the importance of diversity may outweigh that of fidelity. However, in many physical problems where fidelity is paramount, the PF ODE emerges as the superior choice.

The minimum number of PDE operator evaluations needed to achieve a generated dataset residual less than or equal to that of the training set was attained by solving the PF ODE with $\tau = 2000$, $M = 1$, and $N = 3$, resulting in a total of 4 evaluations.

We observe that the number of physical consistency steps, denoted as $N$, preceding the time $t = 0$ serves a dual purpose beyond reducing the physical residual. As mentioned, we




(a) $N = 0$ (b) $N = 50$

Figure 6.4: Physical consistency steps applied only to pressure fields provide a denoising effect on permeability fields while reducing the physical residual. Samples are obtained by solving the reverse SDE using EM solver with $\tau = 2000$ time steps and $M = 0$ additional physical consistency steps.

exclusively apply physical consistency steps to pressure fields. Given that the score function approximation jointly predicts the score functions for both pressure and permeability fields, a correlation emerges in the denoising process for each field. Consequently, directly modifying the pressure field during the reverse process indirectly influences the permeability field through the score function prediction. This, in turn, produces a denoising effect on the permeability fields while concurrently reducing the physical residual. Figure 6.4 illustrates the denoising effect that physical consistency steps have on pressure and permeability fields. Applying physical consistency steps provides two main benefits: encouraging generated samples to follow the physical equations of interest as well providing a denoising effect during sampling. Such benefits are not exclusive to unconditional models and can also be applied to conditional sample generation for forward and inverse problems.

## 6.3 Conditional Generation

Conditional generative models hold immense potential in physics-based applications, offering versatile utility in various tasks such as field prediction, uncertainty quantification, and field reconstruction. Their broad range of applications makes it advantageous to train a single unconditional generative model that can be leveraged by numerous conditional models.

For instance, in the context of field prediction, conditional generative models can be trained to generate specific fields based on various input conditions such as generative parameters, boundary conditions, partial field measurements, or other relevant factors, effectively solving a forward prediction problem. This allows for the flexible generation of fields tailored to specific scenarios. Field reconstruction involves the task of reconstructing a complete field based on partial or incomplete information, and constitutes a reverse problem. Conditional generative models can be trained to reconstruct fields by leveraging available data and conditioning information, offering a powerful tool in scenarios where complete field data may be limited or noisy. In both cases of solving a forward or inverse problem, physical consistency sampling ensures that all predictions will adhere to the underlying physical behavior governed by the PDE.

The advantage of training a single unconditional generative model lies in its ability to capture the underlying distribution of the data. This model can then be flexibly adapted for various conditional tasks, eliminating the need to train separate models for each specific condition. This approach not only streamlines the training process but also facilitates efficient model reuse across a diverse set of applications.

We train two separate conditional augmentations on the Darcy flow dataset with $s = 16$, each of which are connected to the unconditional model architecture as illustrated in Fig. 6.2. The conditional portion of the model (blue) can be removed from the overall architecture to leave behind an unconditional model. This renders it easy to add or remove different conditional augmentations to the same pretrained unconditional model. We copy the same unconditional model trained in Sec. 6.2, and train two separate augmentations as described in Sec. 6.1.2: one for pressure and permeability field prediction from the underlying generative parameters of the permeability field and one for pressure / permeability field reconstruction from partial pressure measurements.

### 6.3.1 Conditional Field Generation from Generative Parameters

We illustrate the first application for conditional generative models as that of surrogate modeling, a forward problem. A conditional augmentation is trained with a baseline unconditional model on the Darcy flow dataset with an underlying dimensionality of $s = 256$.

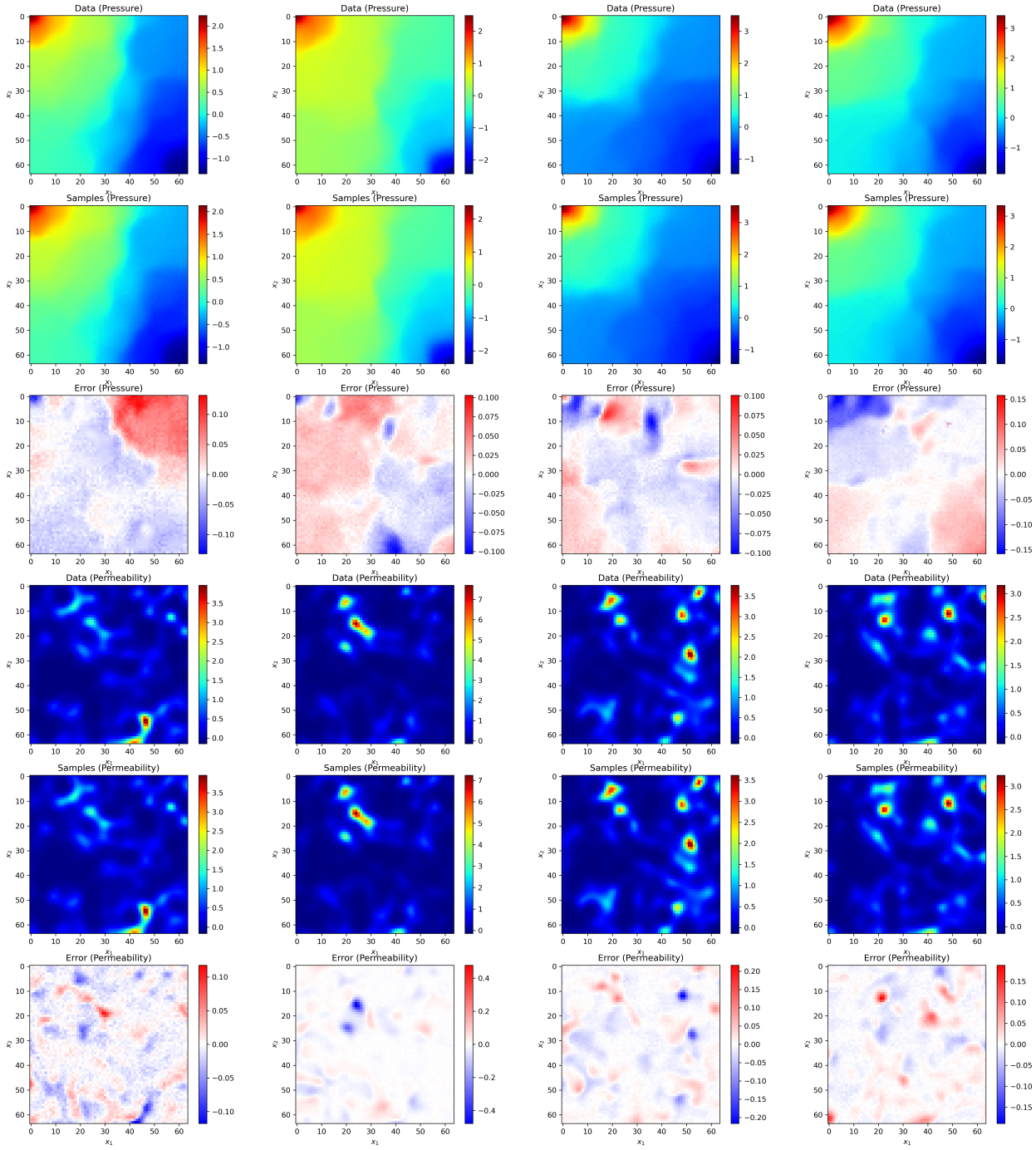



Figure 6.5: Samples from a conditional model trained on Darcy flow data ($s = 256$) where the conditional augmentation takes permeability field generative parameters as input.

The conditioning input $\boldsymbol{\theta} \in \mathbb{R}^s$ are the parameters $\boldsymbol{\theta}$ determining the permeability field in Eq. 3.5. The conditional generative model will output samples from $p(\mathbf{y}|\boldsymbol{\theta})$, given the particular parameterization. As data samples $x$ are determined by an injective mapping

$G(\boldsymbol{\theta}) : \mathbb{R}^s \to \mathbb{R}^{n^2}$, the true conditional p.d.f. is zero everywhere except $\mathbf{y} = G(\boldsymbol{\theta})$.

The training procedure is outlined in Sec. 6.1.2, and additional details are included in Appendix D.2. In this case, the three control encoding blocks in Fig 6.2 are identical in structure to the encoding blocks of the unconditional U-Net [197] with the exception of a zero-convolution at the output of each block. The input to the first control encoding block must therefore be the same shape as the noisy inputs $\mathbf{y}(t)$. To achieve this, the condition encoder consists of linear layers which encode the conditioning from $\mathbb{R}^s$ to $\mathbb{R}^{n^2}$ and reshape it to be directly summed with $\mathbf{y}(t)$.

After training, sampling is performed using conditions $\boldsymbol{\theta}$ from a separate test set which were not present in the training data. We sampling with physical consistency enforcement by solving the PF ODE with $\tau = 2000$, $N = 50$, and $M = 10$. Both qualitative and quantitative reconstructions are provided in Fig 6.5, illustrating accurate predictions for both pressure and permeability fields from the underlying parameterization of the permeability field. Predictions on the test set result in an average $L_2$-norm of $7.0 \times 10^{-4}$ on pressure field predictions and $6 \times 10^{-4}$ on permeability field predictions. We note that prediction is performed by drawing a single sample from $p(\mathbf{y}|\boldsymbol{\theta})$ for each value of $\boldsymbol{\theta}$ in the test set.

#### 6.3.1.1 Conditional Field Inversion and Reconstruction with Sparse Measurements

In addition to solving forward problems, we also illustrate the application of score-based generative models to solve inverse problems. We train a conditional augmentation to perform probabilistic field inversion and reconstruction from sparse measurements. The goal is to perform field inversion to obtain permeability field approximations and reconstruction to obtain pressure field approximations from only measurements of the pressure field. During training, we select a random number $m$ of pressure sensors, place them randomly in the physical domain, and assume accurate pressure measurements at each location. These pressure measurements are taken as the conditioning variables $\theta$ such that we sample from $p(\mathbf{y}|\boldsymbol{\theta})$ given the measurements. Further training details are provided in Appendix D.2.

Figure 6.6 illustrates a single sample drawn from $p(\mathbf{y}|\boldsymbol{\theta})$ along with the conditioning information (pressure measurements with $m = 250$ sensors) and ground truth. However, as the generative model provides the ability to sample from the conditional distribution, uncertainty quantification is possible when predicting the pressure and permeability fields. This is illustrated in Fig. 6.7, showing the mean and standard deviation of $p(\mathbf{y}|\boldsymbol{\theta})$ compared to the true data sample. In reality, the conditional distribution is likely not a Gaussian distribution, but is very flexible due to the use of score-based generative models. We visualize only the mean and standard deviation in Fig. 6.7 for simplicity. In contrast to the surrogate

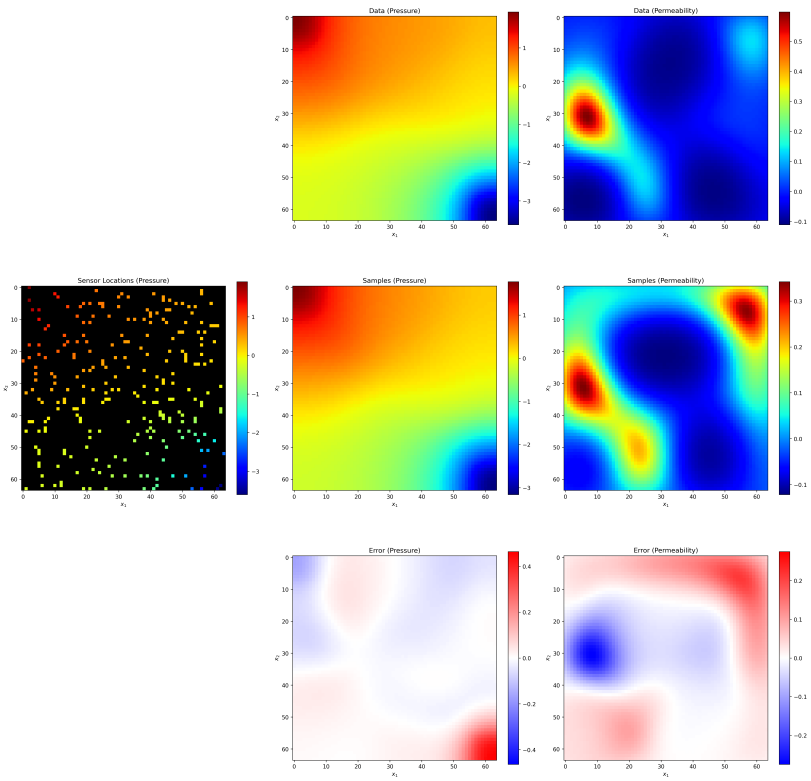



Figure 6.6: Sampling from $p(\mathbf{y}|\theta)$ where $\theta$ are $m = 250$ partial measurements of the pressure field. Samples are always consistent with the underlying PDE, even if prediction accuracy is low.

modeling example in Sec. 6.3.1, it is difficult to guarantee that $\mathbf{y}$ is a deterministic function of $\boldsymbol{\theta}$. For example, consider a case in which a single pressure measurement is given. There exists no unique solution to the entire pressure and permeability fields which will exhibit the correct pressure at the single sensor location. Therefore, uncertainty quantification may be very useful in a real-world situation to determine the reliability or capacity of the current measurements to fully determine the physical fields. Further, as sampling is always performed with physical consistency enforcement (PF ODE, $\tau = 2000$, $N = 50$, $M = 10$), all samples will approximately adhere to the governing PDE.

In an effort to quantify the average field reconstruction error for the case illustrated in Fig. 6.6, we apply the same $m = 250$ pressure measurements to each of the 1,000 cases in the test set, predict the pressure and permeability fields, and measure the error. The pressure reconstruction error is computed as $\mathbb{E}_{p(\mathbf{y}|\boldsymbol{\theta})}[\|P - \hat{P}\|_2^2]$, and the permeability reconstruction error is computed as $\mathbb{E}_{p(\mathbf{y}|\boldsymbol{\theta})}[\|K - \hat{K}\|_2^2]$. The pressure field approximation $\hat{P}$ and permeability field approximation $\hat{K}$ are compared to the true pressure $P$ and permeability $K$ fields. This expectation is approximated by sampling only a single time from $p(x|\theta)$, essentially

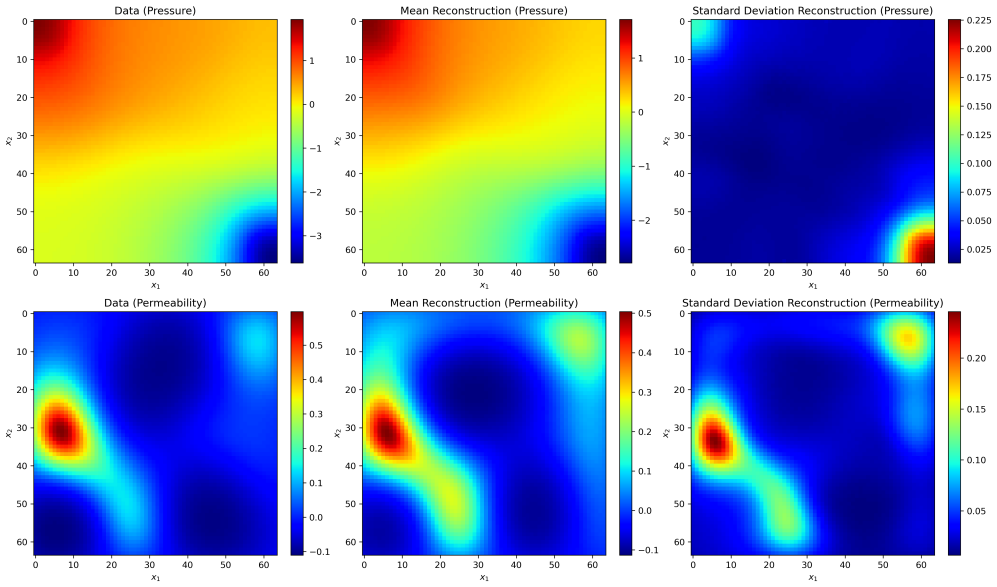



Figure 6.7: Many samples can be drawn from $p(\mathbf{y}|\theta)$, facilitating uncertainty quantification. The data sample (left) is compared to the expectation $\mathbb{E}_{p(\mathbf{y}|\theta)}[\mathbf{y}]$ (center) and standard deviation (right) for both pressure (top) and permeability (bottom) fields.

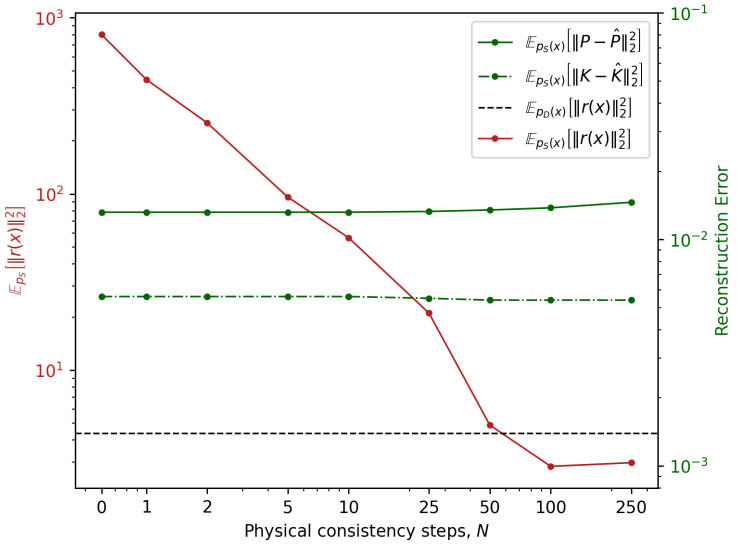
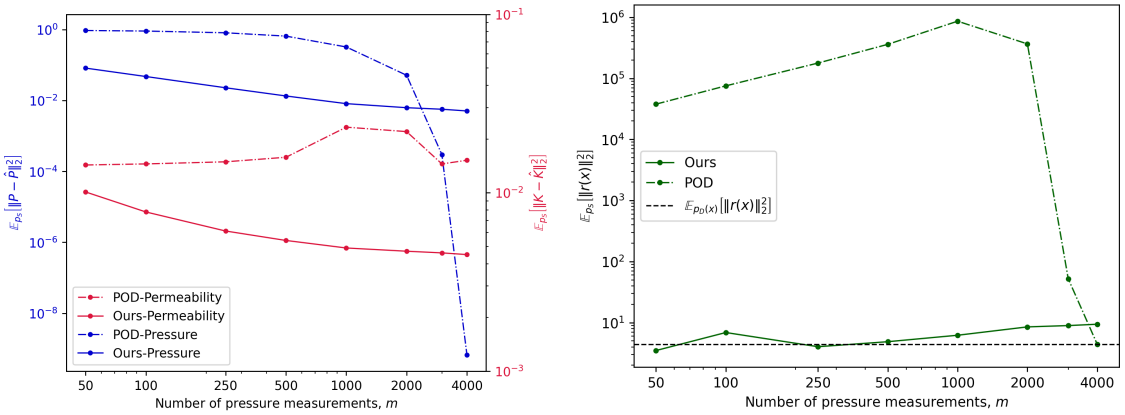



Figure 6.8: Generating conditional samples with physical consistency enforcement for field inversion and reconstruction with sparse measurements ($m = 250$) has minimal impact on reconstruction / inversion errors while improving sample consistency with the physical PDE.

sampling from the model a single time for each data sample. Although Fig. 6.7 illustrates that uncertainty may be high in $p(\mathbf{y}|\boldsymbol{\theta})$, we use this as an illustrative example which does not require a thorough estimation of the expectation.

Our first objective is to illustrate the effect that physical consistency sampling has on the

reconstructive performance of the model. Figure 6.8 shows that as the number of physical consistency steps $N$ is increased, there is nearly zero effect on the reconstructive performance until a large number of steps are used. Even then, in the worst case investigated, only a 2% increase in pressure reconstruction error is observed while permeability reconstruction remains constant. However, notably the physical residual decreases consistently with increasing number of physical consistency steps. This shows that physics can be enforced without negatively impacting the reconstructive performance on the model, even though physics are not enforced during training. One possible explanation for this is that the measurements themselves are included as conditioning. Although PDE information is not included at training time, the measurement information is. As the sampling process is performed, iteratively passing the sample through the score estimation model, physical consistency steps will ultimately result in small changes to the sample before the sampling process is complete. The samples including these small changes are then input into the score estimation function along with the measurement information as conditioning, allowing the model to adapt itself to the small changes caused by physical consistency sampling. In effect, this facilitates the reduction of residuals while maintaining constant reconstruction error.




(a) Reconstruction errors.

(b) Physical residuals.

Figure 6.9: Our method greatly outperforms a POD-based method on field inversion and reconstruction from sparse measurements in terms of reconstruction error and physical residual. Sampling for our method is performed with physical consistency by solving the PF ODE with $\tau = 2000$, $N = 50$ and $M = 10$.

Solving the inverse problem using score-based generative models is further investigated by comparing it to a more established method of reconstructing data from measurements using a basis $\mathbf{\Psi}$ obtained by performing proper orthogonal decomposition (POD) on the training dataset. Given a measurement matrix $\mathbf{P}$ and partial field measurements $\mathbf{q}$, the full field is

reconstructed by approximating the reconstruction in terms of the basis $\boldsymbol{\Psi}$. This approximate reconstruction is computed as $\hat{\mathbf{y}} = \boldsymbol{\Psi}[\mathbf{P}\boldsymbol{\Psi}]^{\dagger}\mathbf{q}$ [201, 202]. The aim of comparison is to illustrate the effect that physical consistency sampling has on predictions when compared to another method. Here, we investigate the reconstructive performance of each method as a function of the number of pressure measurements taken, but also illustrate the average physical residual of reconstructions as a function of the number of measurements. Figure 6.9a shows that for more sparsely sampled measurements, the score-based generative model is able to more accurately reconstruct both pressure and permeability fields. As there are no permeability measurements taken, the POD-based method cannot accurately reconstruct the permeability field, even in the limit of measuring the entire pressure field. However, this method does approach perfect reconstruction of the pressure field as the entire field is measured, while the score-based model does not. Regardless of the number of measurements taken, Fig. 6.9b clearly shows that the score-based model with physical consistency sampling reliably produces samples which adhere to the physical PDE while the POD-based method produces predictions with orders of magnitude higher residuals.

This particular example is intended to illustrate the potential for score-based generative models with physical consistency sampling in solving inverse problems, not to compare against advanced solutions. Whether the conditional distribution $p(\mathbf{y}|\boldsymbol{\theta})$ has little uncertainty as in the forward problem demonstrated in Sec. 6.3.1 or is quite uncertain as in the case of few measurements here, generated samples will adhere to the underlying PDE.

### 6.3.2 Comparisons with Traditional Solvers

It is not straightforward to directly compare the computational complexity of solving the Darcy flow equations with state of the art methods such as multigrid [203]. However, there are some comparisons which can inform the selection of one method or the other in a particular application.

In the case of generating solutions to the Darcy flow equations, the key advantage of direct solving with multigrid is that once set up, the solver solves the Darcy flow equations in a computationally efficient manner, with complexity close to $\mathcal{O}(N)$, where $N$ is the number of unknowns. This method is deterministic and provides high accuracy and reliability for any given problem setup.

However, after the initial expensive training phase, generating new solutions can be relatively fast, especially if generating multiple solutions from a single trained model or leveraging it for solving forward and inverse problems. However, the quality and accuracy of these solutions depend on the training data, the capacity of the model, and how well the training

process has converged.

If the goal is to solve the Darcy flow equations for a few instances, directly solving with a multigrid solver is likely the more efficient approach. If one needs to generate a large number of solutions quickly after an initial setup, and especially if these solutions cover a range of conditions similar to the training set, a score-based generative model could offer advantages in speed and flexibility at the inference phase. Additionally, if the goal is to solve forward and/or inverse problems with the generative model, or especially adapt the model for solving a range of forward and inverse problems, the generative model may be much more efficient than setting up solvers for each application individually. Additionally, particularly for inverse problems, traditional techniques may require many evaluations of the forward model (multigrid solver), whereas the score-based generative model can more efficiently sample from the distribution describing the solution to the inverse problem.

The primary online cost associated with generating samples from score-based generative models is solving the reverse SDE or probability flow ODE. The cost of evaluating the right hand side is very dependent on the cost of a forward evaluation of the neural network architecture used to approximate the time dependent score function $\nabla_{\mathbf{y}} \log p(\mathbf{y}(t))$. If a simple architecture can be leveraged to accurately model the score function, sampling efficiency can be improved over the use of complex network architectures.

Additionally, this work employs simple first order SDE / ODE solvers for solving the reverse SDE and probability flow ODE solvers. However, recent works such as Consistency Trajectory Models [104] have demonstrated far more efficient methods of sampling from score based generative models. Thus, the sampling efficiency has the potential to be greatly improved.

Generally, the characteristics of the particular application should be greatly considered when determining which methods to employ. State of the art solvers may be the superior choice in some applications, while the flexibility provided by score-based generative models may offer greater benefits in others.

## 6.4 Summary

In this chapter, we explore the implementation of score-based generative models to tackle both forward and inverse problems, establishing a framework that seamlessly incorporates physical laws into the generative model sampling process. This chapter outlines an ML methodology that is not only compliant with the physical equations governing the systems under study but also exhibits a broad range of applicability within the physics sector. Central to our strategy is the idea of physical consistency sampling, which proves instrumental

in producing consistently minimal residuals across various implementations. This approach suggests a reduced necessity for the extensive fine-tuning of models and hyperparameters, traditionally required for customized problem-solving, thus facilitating the creation of samples that inherently respect the physics of the system.

Our investigation highlights the effectiveness of solving the probability flow ordinary differential equation backwards in time over the reverse stochastic differential equation, particularly for its capability to minimize physical residuals. Furthermore, the deliberate modification of the number of physical consistency steps, $N$, emerges as a key element in minimizing residuals and improving noise reduction, thereby reinforcing the model's stability. This emphasizes the importance of careful hyperparameter adjustments, especially for parameters such as $\tau$, to achieve optimal physical consistency without compromising the potential to reach minimal physical residuals.

The framework's design emphasizes versatility and the ability to accurately solve both forward and inverse problems while maintaining physical authenticity in the outputs. This methodological choice does not detract from predictive accuracy; on the contrary, it is demonstrated through a consistent decrease in physical residuals, highlighting its advantage over traditional sampling approaches. The example involving Darcy flow illustrates this point effectively, showing how adjustments in the pressure field can influence the permeability field through score function estimation, enhancing the quality of the permeability fields and reducing physical residuals simultaneously.

Reflecting on the entirety of this dissertation, Chapter 6 not only aids in achieving its set objectives but also establishes a benchmark for employing score-based generative models in physics-oriented machine learning applications. By presenting a foundational framework, it encourages further investigation and development within this area. The adaptability, effectiveness, and potential of our approach highlight its capacity to innovate the integration of physical consistency into generative modeling, fostering novel solutions to applications with the intersection of physics and machine learning.

# CHAPTER 7

# Concluding Remarks and Future Directions

This work presents a view into the evolving field of physics-aware ML, illustrating how leveraging data with ML techniques can be used to improve the efficiency and accuracy of predictive physical models. By incorporating physical laws into machine learning algorithms, we have shown improvements in model interpretability, reliability, accuracy, efficiency, and applicability in various physical modeling scenarios. This interdisciplinary strategy not only enhances our theoretical comprehension but also opens up new possibilities for practical applications, ranging from system optimizations to the resolution of intricate inverse problems.

The journey from specific cases to a broad framework reveals the adaptability of physics-aware machine learning. Although each study contains unique applications and goals, a common thread is the synergy between physical understanding and computational advancements, providing a foundation for future research in the area.

## 7.1 Main Contributions and Conclusions

### 7.1.1 Extracting Physical Parameters from Data with Variational Autoencoders

The initial contribution presented is in demonstrating use of VAEs in computational physics to transform complex physical data into low-dimensional, interpretable formats that closely reflect the underlying physically significant generative parameters. This approach enhances the interpretability of data analysis for physical systems by converting complex datasets into simpler, more meaningful forms, facilitating a deeper understanding of the phenomena under investigation.

The strategic use of VAEs, highlighted by innovative techniques aimed at refining the training process, stands out in our methodology. Notably, the introduction of a loss-weighting schedule effectively counters the issue of over-regularization and the resulting zero mutual

information, a frequent challenge in VAE training. This strategy helps the training process avoid local minima that could impede the acquisition of significant representations.

Furthermore, the implementation of hierarchical priors within VAEs represents a methodological enhancement, enabling the capture of complex prior distributions and thus boosting the effectiveness and robustness of unsupervised learning in pinpointing physically correlated representations. This advancement is crucial for improving the model's ability to learn with limited supervision.

An essential insight from our work is the notable improvement in the robustness, generalizability, and accuracy of the learning process when even minimal prior physical information is included. Introducing labels for a select set of samples demonstrates that integrating physical knowledge into machine learning algorithms significantly bolsters their ability to recognize and model complex physical systems. This highlights the value of melding domain-specific insights with machine learning model design and application, bridging the divide between data-driven methods and classical physical science approaches.

However, the use of VAEs in computational physics is not without its challenges. The complexity of physical systems, alongside the difficulty of accurately capturing their dynamics with low-dimensional representations, necessitates ongoing refinement of both models and methodologies. Moreover, even minimal reliance on labeled data emphasizes the need for thoughtful integration of physical knowledge into the modeling process, which can be challenging when such information is scarce or difficult to precisely define.

### 7.1.2 Enhancing Dynamics Modeling with Data-driven Inference and Interpretable Machine-learning Augmentations

In the second contribution, we focus on modeling cathodic electrophoretic deposition through the use of variational inference, applying it to refine a baseline model with the help of extensive experimental data. This approach helps identify the baseline model's shortcomings, such as issues with parameter identifiability and its inability to adapt to different experimental conditions. These findings prompt a reevaluation of the deposition onset time model towards a more generalized approach that could handle diverse experimental settings. While these improvements enhanced the model's versatility, they did not fully capture all critical behaviors of the deposition process.

Addressing the accurate modeling of electrophoretic deposition, especially the initiation phase, posed a significant challenge. We responded by incorporating interpretable machine learning augmentations to boost the model's predictive accuracy and maintain its generalizability. These augmentations, rooted in physical principles, were designed to keep the model's

interpretability intact while adding the necessary flexibility. Nevertheless, the model's performance in predicting film deposition under low voltage conditions revealed a notable shortfall, highlighting its limitations in fully accounting for the governing physical principles in such scenarios.

This situation illustrates a fundamental limitation inherent in even the most adaptable machine learning models: the difficulty of capturing complex physical phenomena without a solid foundational understanding of the underlying mechanisms. The versatility and capability of machine learning are fully realized when they are integrated with a deep understanding of physical theory. This relationship underscores the importance of foundational modeling in physics and the role of ML as a supplementary, not substitute, tool that should be thoughtfully combined with traditional scientific methods.

Looking ahead, this work outlines several avenues for further exploration. Although the work demonstrates insights towards improving current electrophoretic deposition models, improving the model's performance in low voltage conditions and other challenging scenarios will likely involve a closer fusion of physical insight with machine learning innovation. This could mean developing more advanced machine learning methods that better incorporate physical laws or adopting new data collection strategies to enhance model training. Expanding the model to cover a wider range of electrophoretic deposition phenomena could also yield a more thorough understanding of these complex processes. The confluence of machine learning and physical modeling offers a promising area for future innovation, with the potential to deepen our comprehension and control of electrophoretic deposition and similar complex physical systems.

### 7.1.3 Rate Distortion Informed Clustering of Non-equilibrium Gas Dynamics

In the third contribution, we explore non-equilibrium gas dynamics using a machine learning framework that applies rate-distortion theory to identify state clusters for a reduced-order model, ensuring adherence to essential physical constraints such as mass and energy conservation. This method, grounded in the well-known maximum entropy principle, effectively simplifies the dynamics into manageable clusters within the high-dimensional energy state space, thus significantly lowering the computational demands of prediction tasks. Although the reduced order model is developed in zero spatial dimensions, it has been shown by Zanardi et al. [204] that the learned cluster assignments work well in simulations in higher spatial dimensions. Thus, achieving state of the art accuracy in the 0D reduced order model presented in this work is likely to result in state of the art performance in complex hypersonic

simulations in higher dimensions. Such simulations are computationally infeasible without the adoption of reduced order models for the state-to-state master equations.

The primary challenge with this approach is the detailed process of assigning states to clusters, which has traditionally been addressed through ad-hoc methods and iteratively refined. Our research enhances this process by implementing a systematic strategy to identify the specific characteristics that guide state-to-cluster assignments, employing a probabilistic framework that supports the use of gradient-based optimization techniques for more accurate and efficient model training. This innovation includes the creation of two effective approximations for estimating the expected value of reduced-order predictions related to cluster assignments.

A key aspect of our methodology is the application of a rate-distortion (RD) informed loss function, greatly improving the model's robustness and the accuracy of the encoding-decoding system designed to minimize reconstruction errors. The strategic adjustment of loss weighting constants during training is vital for avoiding local minima, leading to a learning outcome that is both more precise and broadly applicable. This careful tuning process draws parallels with strategies used in variational autoencoders, emphasizing the need for balanced training to attain high predictive performance.

However, our study does face limitations, notably in the generalizability of learned cluster assignments under different ambient conditions, primarily because the model was trained with constant temperature settings. This highlights an area for potential enhancement in the model's versatility by training under a wider array of conditions, though such efforts are currently limited by computational resources.

This research highlights the ability of machine learning to increase the accuracy of physical models while maintaining a significant degree of their interpretability. Nevertheless, it also points to a crucial trade-off: improvements in model accuracy may sometimes obscure interpretability. The gains in prediction accuracy through learned cluster assignments illustrate the advantages of merging machine learning with physical modeling. However, interpreting these assignments within a physical context remains a complex task, emphasizing the delicate balance needed when integrating machine learning with the physical sciences.

Looking ahead, this study also highlights opportunities for further investigation, especially in terms of expanding the model's training to include variable temperature conditions and employing more sophisticated optimization techniques to navigate computational constraints. Continuing to refine these machine learning methods will improve our modeling of complex physical phenomena, potentially leading to significant breakthroughs in understanding and applying non-equilibrium gas dynamics and similar areas. The ongoing effort to balance accuracy and interpretability in models is crucial, promising valuable insights as

machine learning and traditional physical modeling methods become increasingly integrated.

### 7.1.4 Physically Consistent Diffusion Model Sampling for Solving Forward and Inverse Problems

In the final part of this work, we introduce a framework aimed at ensuring physical consistency within score-based generative models. This development transforms these models into dynamic tools capable of tackling both forward and inverse problems in the context of physics-aware ML. By incorporating physical laws directly into the sampling process, this methodology circumvents the high computational demands usually associated with calculating physical residuals during training such as in PINNs, thereby reducing the need for extensive tuning and hyperparameter adjustments typically required in physics-based machine learning projects.

A key feature of the approach is the concept of physical consistency sampling, which is shown to yield consistently minimal residuals in a variety of applications, highlighting its capacity to standardize the production of physically plausible samples. Notably, the use of the probability flow ordinary differential equation stands out for its efficiency compared to traditional reverse stochastic differential equation solvers, mainly because of its ability to reduce physical residuals. Furthermore, the careful calibration of physical consistency steps, particularly through the adjustment of the hyperparameter defining the number of sampling steps, plays a vital role in diminishing residuals and bolstering the model's resilience.

Examining this framework's application to both forward and inverse problems reveals its broad utility and flexibility. For instance, the Darcy flow scenario effectively demonstrates how altering the pressure field impacts the permeability field via score function estimation, illustrating the twofold advantage of refining the permeability fields while simultaneously reducing the physical residual. Additionally, first order SDE and ODE solvers result in generating samples with similar efficiency to solving the Darcy flow equations with a second order finite difference solver. Recent works have illustrated 3-4 orders of magnitude improvement to sampling efficiency in score-based generative models. The implementation of such sampling strategies along with the tuning of model architectures is likely to advance the efficiency of solving forward and inverse problems with score-based generative model well past that of even the most advanced traditional solvers.

Moreover, we present an architecture specifically designed for swift and efficient adaptation to new applications, leveraging a pre-existing pretrained model. The pretrained model has the potential to serve as a foundational tool, capable of generating approximate solutions for a wide array of PDE operators. Through employing conditional networks, the

model could be adeptly guided to address specific forward or inverse problems across diverse applications.

Given the rapid progress in generative modeling, such techniques are poised to become increasingly influential in physics-based applications and computational science. The future holds numerous opportunities for extending these concepts, including the use of more efficient sampling techniques, the development of models that can address time-evolving dynamics, the creation of general foundation models for solving and generating solutions to various PDEs under different conditions, integrating boundary conditions into the model guidance process, and producing mesh-free solutions. This exploration into the use of generative AI within physics applications is set to potentially transform how scientific inquiries and problems are approached and resolved.

### 7.1.5 Final Remarks

Together, these studies present a unified perspective on physics-aware machine learning, underscoring the essential role of integrating physical principles into machine learning models to improve their interpretability, reliability, and general applicability while improving the predictive efficiency and accuracy of such physical models. A recurring theme throughout this work is the significant impact that embedding physical insights into machine learning frameworks can have, ranging from achieving disentangled representations in computational physics to enhancing physical models. This contribution enriches the dialogue between machine learning and physics and lays the groundwork for future research that connects these fields.

However, it is important to acknowledge that the problems addressed in this thesis, while significant, represent a simplified subset of the complex challenges faced by industry applications. Extending these ideas to meet industry needs would likely necessitate tackling problems of greater scale and complexity, often under constraints of time sensitivity and computational efficiency. This extension would require not only advanced algorithms and robust theoretical frameworks but also a comprehensive approach including large-scale data handling, the use of sophisticated computational libraries, and the development of scalable machine learning models.

Recognizing the limitations of our current models such as data scarcity, scalability issues, and the balance between model complexity and computational feasibility, underscore the hurdles that must be overcome. Addressing these challenges is critical for advancing the practicality of physics-aware machine learning in real-world applications. It calls for innovative solutions that leverage cutting-edge algorithms, theory, and data management tech-

niques, alongside the deployment of computational libraries designed for high-performance computing environments.

Looking forward, the path is ripe with opportunities for exploration and growth. The development of more complex models capable of handling the intricacies of industrial problems, the application of these methodologies in new physical domains, and a deeper integration of theoretical physics with cutting-edge machine learning strategies all represent promising avenues for research. The potential for making a meaningful impact is immense, suggesting that we can make significant strides in our understanding and manipulation of complex systems.

As we look to the future, the opportunities for further inquiry are plentiful, whether it involves developing more intricate models, applying these ideas across new physical arenas, or a deeper integration of theoretical physics with cutting-edge machine learning strategies. The prospects for meaningful impact in both arenas are substantial, forecasting significant progress in our ability to understand and influence complex systems. The capacity of physics-aware machine learning to transform our approach to scientific challenges is profound, and this work is merely a step in the continuous journey of discovery and innovation.

# APPENDIX A

# Extracting Physical Parameters from Data with Variational Autoencoders

## A.1 Solving Darcy Flow with a Linear System

Section 3.2.1 describes the construction of the linear system $\mathbf{Ap} = \mathbf{f}$ to solve the Darcy flow equations for pressure using finite differences on a discretized computational domain. The matrix $\mathbf{A}$ is formed by constructing $n^2 + 1$ equations for pressure values at $n^2$ spatial locations in the computational domain. The equations are broken down into 4 categories: corner nodes, edge nodes, interior nodes, and integral enforcement.

There are four corner equations corresponding to nodes $(1, 1)$, $(n, 1)$, $(1, n)$, and $(n, n)$. These equations are given by

$$\begin{aligned}
\text{corner node } 1,1: &-\frac{K(\mathbf{x}_{1,1})}{\Delta x^2}[2p(\mathbf{x}_{2,1}) - 4p(\mathbf{x}_{1,1}) + 2p(\mathbf{x}_{1,2})] = f_s(\mathbf{x}_{1,1}) \\
\text{corner node } n,1: &-\frac{K(\mathbf{x}_{n,1})}{\Delta x^2}[2p(\mathbf{x}_{n-1,1}) - 4p(\mathbf{x}_{n,1}) + 2p(\mathbf{x}_{n,2})] = f_s(\mathbf{x}_{n,1}) \\
\text{corner node } 1,n: &-\frac{K(\mathbf{x}_{1,n})}{\Delta x^2}[2p(\mathbf{x}_{2,n}) - 4p(\mathbf{x}_{1,n}) + 2p(\mathbf{x}_{1,n-1})] = f_s(\mathbf{x}_{1,n}) \\
\text{corner node } n,n: &-\frac{K(\mathbf{x}_{n,n})}{\Delta x^2}[2p(\mathbf{x}_{n-1,n}) - 4p(\mathbf{x}_{n,n}) + 2p(\mathbf{x}_{n,n-1})] = f_s(\mathbf{x}_{n,n})
\end{aligned}$$

There are an additional $4(n-2)$ equations corresponding to nodes along edges which are not also corner nodes. The equations for each of these edges are given by

$$\begin{aligned}
\text{edge } i = 1: \; &-\frac{K(\mathbf{x}_{1,j})}{\Delta x^2}[p(\mathbf{x}_{1,j-1}) - 4p(\mathbf{x}_{1,j}) + p(\mathbf{x}_{1,j+1}) + 2p(\mathbf{x}_{2,j})] \\
&\quad - \frac{1}{2\Delta x}[K(\mathbf{x}_{1,j+1}) - K(\mathbf{x}_{1,j-1})][p(\mathbf{x}_{1,j+1}) - p(\mathbf{x}_{1,j-1})] = f_s(\mathbf{x}_{1,j})
\end{aligned}$$

$$\text{edge } i = n: \; -\frac{K(\mathbf{x}_{n,j})}{\Delta x^2}[p(\mathbf{x}_{n,j-1}) - 4p(\mathbf{x}_{n,j}) + p(\mathbf{x}_{n,j+1}) + 2p(\mathbf{x}_{n-1,j})] \\ - \frac{1}{2\Delta x}[K(\mathbf{x}_{n,j+1}) - K(\mathbf{x}_{n,j-1})][p(\mathbf{x}_{n,j+1}) - p(\mathbf{x}_{n,j-1})] = f_s(\mathbf{x}_{n,j})$$

$$\text{edge } j = 1: \; -\frac{K(\mathbf{x}_{i,1})}{\Delta x^2}[p(\mathbf{x}_{i-1,1}) - 4p(\mathbf{x}_{i,1}) + p(\mathbf{x}_{i+1,1}) + 2p(\mathbf{x}_{i,2})] \\ - \frac{1}{2\Delta x}[K(\mathbf{x}_{i+1,1}) - K(\mathbf{x}_{i-1,1})][p(\mathbf{x}_{i+1,1}) - p(\mathbf{x}_{i-1,1})] = f_s(\mathbf{x}_{i,1})$$

$$\text{edge } j = n: \; -\frac{K(\mathbf{x}_{i,n})}{\Delta x^2}[p(\mathbf{x}_{i-1,n}) - 4p(\mathbf{x}_{i,n}) + p(\mathbf{x}_{i+1,n}) + 2p(\mathbf{x}_{i,n-1})] \\ - \frac{1}{2\Delta x}[K(\mathbf{x}_{i+1,n}) - K(\mathbf{x}_{i-1,n})][p(\mathbf{x}_{i+1,n}) - p(\mathbf{x}_{i-1,n})] = f_s(\mathbf{x}_{i,n})$$

A remaining $(n-2)^2$ equations correspond to interior nodes (not on corners or edges) and are given by

$$\text{interior node } i,j: \; -\frac{K(\mathbf{x}_{i,j})}{\Delta x^2}[p(\mathbf{x}_{i,j-1} + p(\mathbf{x}_{i-1,j}) - 4p(\mathbf{x}_{i,j}) + p(\mathbf{x}_{i+1,j}) + p(\mathbf{x}_{i,j+1})] \\ - \frac{1}{2\Delta x}[K(\mathbf{x}_{i+1,j}) - K(\mathbf{x}_{i-1,j})][p(\mathbf{x}_{i+1,j}) - p(\mathbf{x}_{i-1,j})] \\ - \frac{1}{2\Delta x}[K(\mathbf{x}_{i,j+1}) - K(\mathbf{x}_{i,j-1})][p(\mathbf{x}_{i,j+1}) - p(\mathbf{x}_{i,j-1})] = f_s(\mathbf{x}_{i,j})$$

The final equation is formed by approximating the integral condition given in Eq. 3.1 using 2-dimensional trapezoidal rule. This equation is given by

$$\frac{\Delta x^2}{4}\left[p(\mathbf{x}_{1,1}) + p(\mathbf{x}_{n,1}) + p(\mathbf{x}_{1,n}) + p(\mathbf{x}_{n,n}) + 2\sum_{i=2}^{n-1} p(\mathbf{x}_{i,1}) + 2\sum_{i=2}^{n-1} p(\mathbf{x}_{i,n}) \right. \\ \left. +2\sum_{j=2}^{n-1} p(\mathbf{x}_{1,j}) + 2\sum_{j=2}^{n-1} p(\mathbf{x}_{n,j}) + 4\sum_{i=2}^{n-1}\sum_{j=2}^{n-1} p(\mathbf{x}_{i,j})\right] = 0$$

The explicit form of the matrix $\mathbf{A}$ is not provided due to the large form of the matrix, but it can be easily constructed from the provided equations. The overdetermined system is solved through least squares minimization where the solution is given by $\mathbf{p} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{f}$.

## A.2 Rotationally-Invariant Distributions

A matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$ is a rotation matrix if for all $z \in \mathbb{R}^n$, $\|\mathbf{R}z\|_2 = \|z\|_2$.

A probability distribution $p(z)$ is said to be *rotationally-invariant* if $p(z) = p(\mathbf{R}z)$ for all $z \in \mathbb{R}^n$ and for all rotation matrices $\mathbf{R} \in \mathbb{R}^{n \times n}$.

The ELBO loss is unaffected by rotations of the latent space when training with a rotationally-invariant prior. This is shown in detail in [97].

## A.3 Architecture Description and Optimization

A convolutional layer is first applied to the input. A series of dense blocks and encoding blocks followed by a flatten and fully connected layers then encode the input to the parameterized latent distribution. A series of decoding blocks and dense blocks then decode samples from the latent space to the output. Figure A.1 illustrates the general architecture we have selected for the latent mean and log-variance along with the output mean (with the output log-variance being constant but trainable).

A convolutional architecture was initially implemented without the use of dense blocks, but reconstruction of data is not very accurate with this architecture, even with some amount of hyperparameter tuning. Ref. [141] illustrated that the architecture implemented there can accurately predict the data of our problem. The architecture contains many hyperparameters such as number of dense blocks, number of layers in each dense block, dense block growth rate, stride of convolutions, fully connected layer width, and others. There were three main goals for us in tuning the hyperparameters: accurate reconstruction, ability to produce disentangled representations, and high computational efficiency. As an example of hyperparameter tuning, we consider changes in the dense block growth rate keeping all other hyperparameters constant. Ten VAEs were trained with the ELBO loss for each growth rate value on the KLE2 dataset with $p(\theta)$ being standard normal. Figure A.2 illustrates some statistics on this study. The overall ELBO loss, and in particular the reconstruction loss continues to decrease with and increase in growth rate, which is desirable. Good conclusions cannot be easily drawn from the disentanglement statistics, although at each growth rate a disentangled representation was observed. However, as the growth rate increases, the probability of convergence decreases. This may be improved by introducing lower learning rates, but in our case increase training time was highly undesirable. Thus, a growth rate of 4 was selected.

## A.4 Over-Regularization

To illustrate the avoidance of over-regularized local minima using our training method, Figure A.3 shows the training losses and $\beta$ as a function of epoch. The VAE loss reaches a local

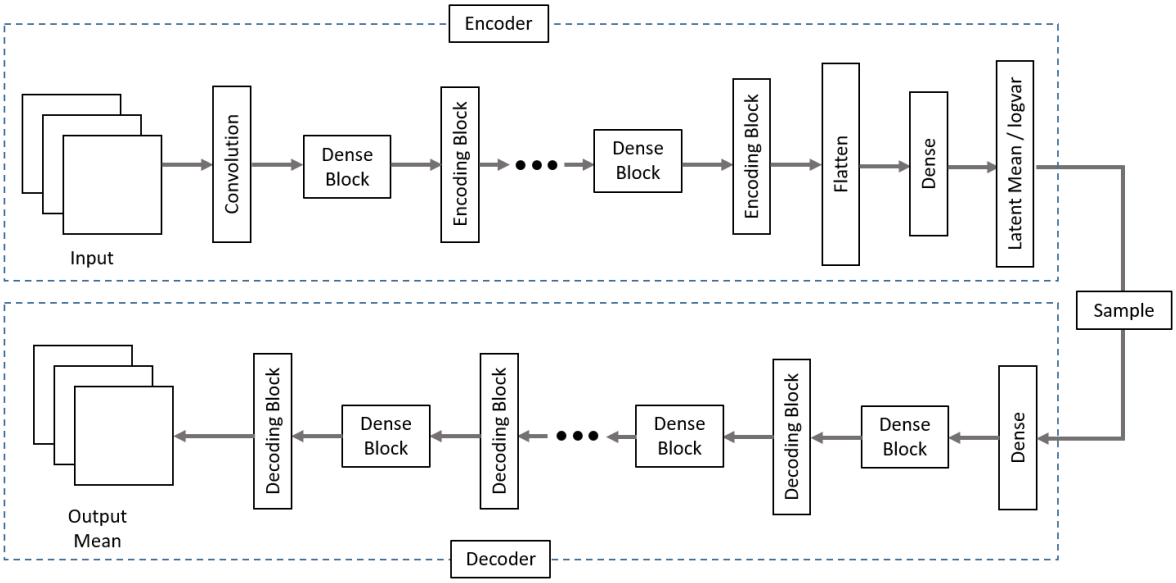



Figure A.1: Dense VAE architecture.

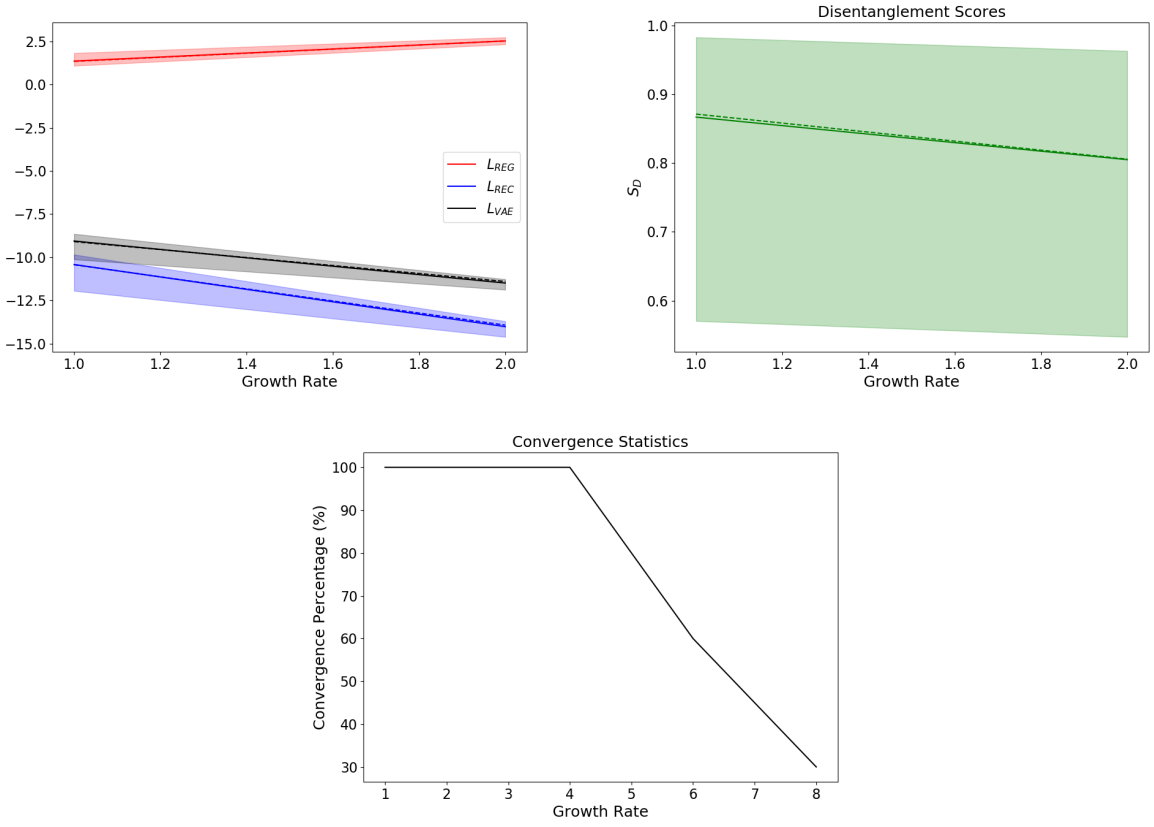



Figure A.2: Hyperparameter selection example.

minimum but continues to increase as the true training loss decreases. With a small initial $\beta$ ($10^{-7}$), great emphasis is placed on the reconstruction loss. When $\beta$ begins to increase, the VAE is 'past' the over-regularized region and the training loss rapidly converges to the VAE loss, obtaining a desirable solution.

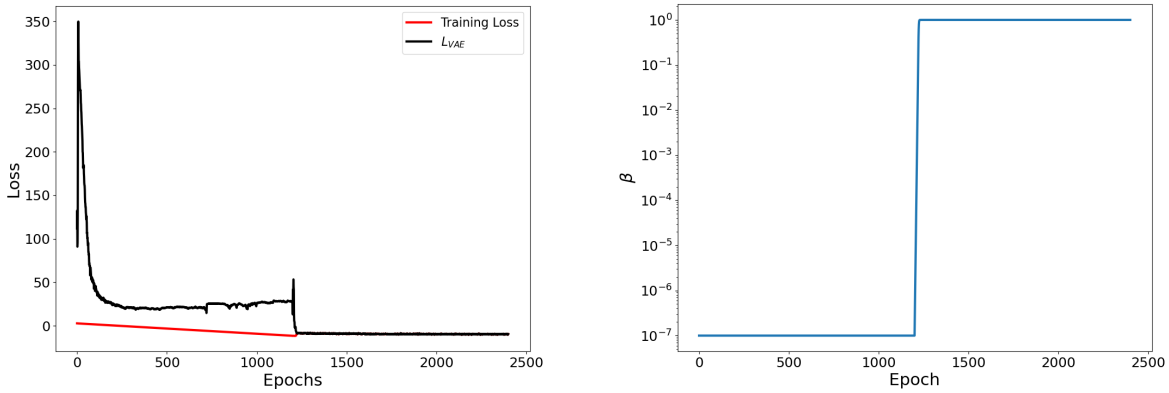



Figure A.3: Training with initial increased weight on reconstruction loss helps to avoid over-regularized local minima.

## A.5 Loss Analysis with Increasing Number of Training Samples

This study shows the relationship of the loss function and disentanglement with respect to the number of data samples used to train the VAE (Figure A.4). All results are obtained with $\beta = 1$ during training and a latent dimension $n = 2$. For every number of training data ([32, 64, 128, 256, 512]), 10 VAEs are trained.

## A.6 Local Minima in Regularization Loss From Rotation of Latent Space

We hypothesize that local minima exist in the regularization loss with respect to rotations in the latent space for the multimodal generative parameter distribution case. This results in the aggregated posterior being rotated 45 degrees relative to the generative parameter distribution (Figure A.5).

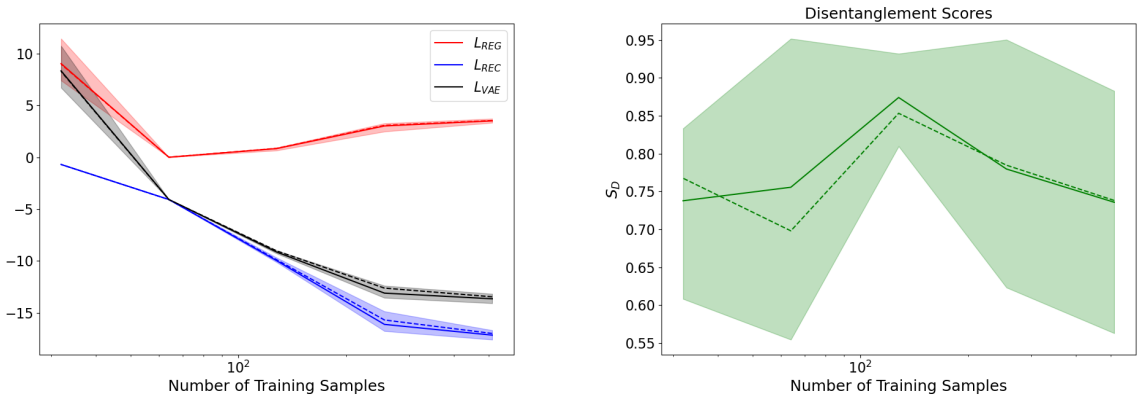



Figure A.4: Solid lines indicate averages over training data for 10 VAEs trained at each point. Dashed lines represent averages over testing data. Ranges indicate minimum and maximum values. *left* Converged VAE losses for various numbers of training samples. *right* Converged VAE disentanglement score as a function of number of training samples.

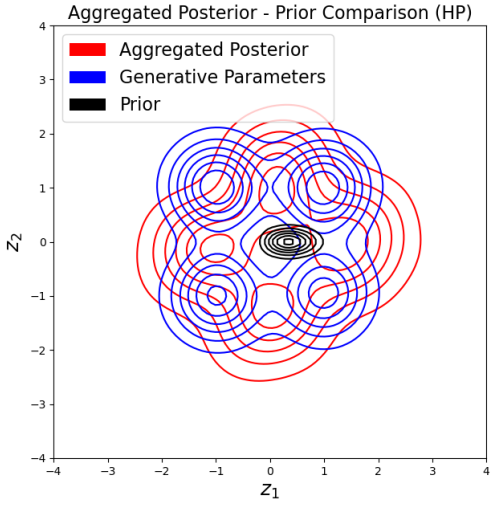



Figure A.5: A 45 degree rotation of the latent space may be the result of local minima in the regularization loss during training.

## A.7 Regularization Loss as a Function of Rotation of Latent Angle

Using rotationally-invariant priors does not enforce any particular rotation of the learned aggregated posterior distribution. In contrast, a non-rotationally-invariant prior can be used to enforce a particular rotation of the latent space (Figure A.6).

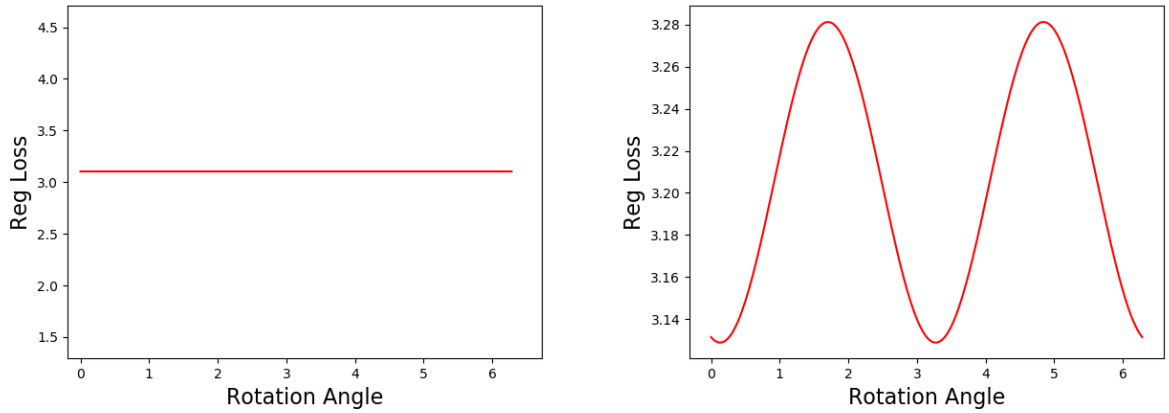



Figure A.6: (*left*) Regularization loss unaffected by latent rotation when training with rotationally-invariant priors, (*right*) regularization loss is affected by latent rotation when training with non-rotationally-invariant priors.

## A.8 Disentanglement of Correlated Generative Parameters

Disentanglement has not been observed using our architecture when generative parameter distributions exhibit correlations between dimensions. Figure A.7 shows that the aggregated posteriors are rotated relative to the generative parameter distributions, which does not facilitate learning a disentangled latent representation.

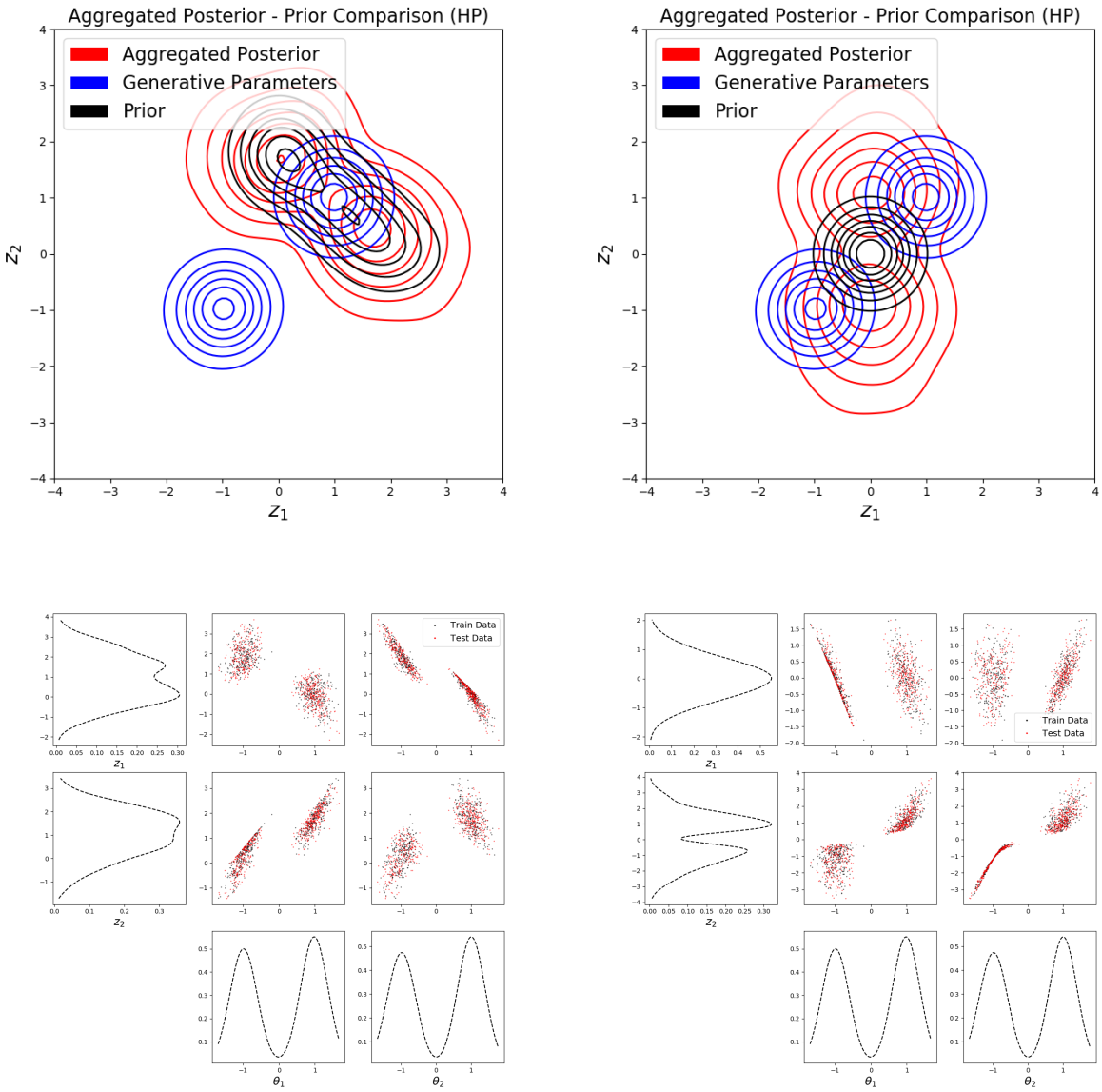



Figure A.7: (*top*) aggregated posterior comparison correlations / rotations relative to the generative parameter distribution, (*bottom*) worse disentanglement when correlations not expressed in latent space.

# APPENDIX B

# Enhancing Dynamics Modeling with Data-driven Inference and Interpretable Machine-learning Augmentations

## B.1 Voltage ramp

For the electric field, we solve a Poisson equation with Robin boundary condition on the interface bath/film and Dirichlet condition at the anode.

$$\sigma_{\text{bath}} \frac{\partial^2 \phi}{\partial x^2} = 0 \qquad \text{in the bath} \tag{B.1}$$

$$\phi - R_{\text{film}} \sigma_{\text{bath}} \frac{\partial \phi}{\partial x} = 0 \qquad \text{at the interface film-bath} \tag{B.2}$$

$$\phi_{\text{anode}}(t) = \phi_{t=0} + \phi_{\text{ramp}}(t) \qquad \text{at the anode} \tag{B.3}$$

where $\sigma_{\text{bath}}$ is the bath conductivity, $j$ is the normal component of the current density, $\phi$ is the electrical potential, $R_{\text{film}}$ is the film resistance and $h$ is the film thickness.

## B.2 Constant current

For the electric field, we solve a Poisson equation with Robin boundary condition on the interface bath/film and Neumann condition at the anode.

$$\sigma_{\text{bath}} \frac{\partial^2 \phi}{\partial x^2} = 0 \qquad \text{in the bath} \tag{B.4}$$

$$\phi - R_{\text{film}} \sigma_{\text{bath}} \frac{\partial \phi}{\partial x} = 0 \qquad \text{at the interface film-bath} \tag{B.5}$$

$$\sigma_{\text{bath}} \frac{\partial \phi}{\partial x} = j_0 \qquad \text{at the anode} \tag{B.6}$$

## B.3 Evolution of hydroxide concentration

### B.3.1 Solution of the diffusion equation

We consider the diffusion equation in the domain [0;L]

$$\frac{\partial u}{\partial t} = D \frac{\partial^2 u}{\partial x} \tag{B.7}$$

with the following boundary conditions:

$$\left. \frac{\partial u}{\partial x} \right|_{x=0} = -\frac{j(t)}{DF} = g(t) \quad \text{and} \quad u(x = L, t) = h(t) \tag{B.8}$$

where u is the $OH^-$ concentration, $F$ is the Faraday constant, $D$ is the diffusion coefficient, $h$ and $g$ are time-dependent functions. The initial condition is defined by $u(x, 0) = u_0$, $u_0$ being the initial concentration.

Applying the Laplace transform

$$\mathcal{L}\left\{ \frac{\partial u}{\partial t} - D \frac{\partial^2 u}{\partial x} \right\} = \mathcal{L}\left\{ \frac{\partial u}{\partial t} \right\} - D \left\{ \frac{\partial^2 u}{\partial x} \right\} = s\bar{u} - u_0 - D\bar{u}_{xx} = 0 \tag{B.9}$$

The solution of the homogeneous equation $s\bar{u} - D\bar{u}_{xx} = 0$ is given by

$$\bar{u}_h(x, s) = C_1 \exp(r_1 x) + C_2 \exp(r_2 x) \tag{B.10}$$

where $r_1 = \sqrt{s/D}$ and $r_2 = -\sqrt{s/D}$

The specific solution, assuming a constant $\bar{u}_{sp.} = A$ is given by $\bar{u}_{sp.} = u_0/s$. Therefore, the solution has the following form

$$\bar{u}(x, s) = C_1 \exp(r_1 x) + C_2 \exp(r_2 x) + \frac{u_0}{s} \tag{B.11}$$

where $C_1$ and $C_2$ are functions derived using the boundary and initial conditions.

Using the Neumann boundary condition at x=0 and the definition $\mathcal{L}\{g(t)\} = G(s)$, then

$$\bar{u}_x(x=0,s) = G(s) = C_1 r_1 + C_2 r_2. \tag{B.12}$$

Therefore $C_1 = \dfrac{G(s) - C_2 r_2}{r_1} = \dfrac{G(s)}{r_1} + C_2$.

Hence

$$\bar{u}(x,s) = \left(\frac{G(s)}{r_1} + C_2\right)\exp(r_1 x) + C_2 \exp(r_2 x) + \frac{u_0}{s} \tag{B.13}$$

Using the Dirichlet boundary condition at x=L:

$$\bar{u}(x=L,s) = H(s) = \left(\frac{G(s)}{r_1} + C_2\right)\exp(r_1 L) + C_2 \exp(r_2 L) + \frac{u_0}{s}, \tag{B.14}$$

the solution is given by

$$\begin{aligned}\bar{u}(x,s) = &\left[\frac{G(s)}{r_1} + \frac{H(s) - \dfrac{u_0}{s} - \dfrac{G(s)}{r_1}\exp(r_1 L)}{\exp(r_1 L) + \exp(r_2 L)}\right]\exp(r_1 x) \\ &+ \left[\frac{H(s) - \dfrac{u_0}{s} - \dfrac{G(s)}{r_1}\exp(r_1 L)}{\exp(r_1 L) + \exp(r_2 L)}\right]\exp(r_2 x) + \frac{u_0}{s}\end{aligned} \tag{B.15}$$

If $L \to \infty$ and $h(t) = u_0$ (i.e. $H(s) = u_0/s$), the solution simplifies to

$$\bar{u}(x,s) = \left[\frac{G(s)}{r_1} - \frac{G(s)}{r_1}\right]\exp(r_1 x) + \left[-\frac{G(s)}{r_1}\right]\exp(r_2 x) + \frac{u_0}{s} \tag{B.16}$$

$$= -\frac{G(s)}{r_1}\exp(r_2 x) + \frac{u_0}{s} \tag{B.17}$$

Using the inverse Laplace transform yields

$$\mathcal{L}^{-1}\{\bar{u}(x,s)\} = \mathcal{L}^{-1}\left\{-G(s)\sqrt{\frac{D}{s}}\exp\left(-\sqrt{\frac{s}{D}}x\right) + \frac{u_0}{s}\right\} \tag{B.18}$$

$$= -\sqrt{D}\mathcal{L}^{-1}\left\{G(s)\frac{1}{\sqrt{s}}\exp\left(-\frac{x}{\sqrt{D}}\sqrt{s}\right)\right\} + u_0. \tag{B.19}$$

Knowing that

$$\mathcal{L}^{-1}\left\{\frac{1}{\sqrt{s}}\exp\left(-\frac{x}{\sqrt{D}}\sqrt{s}\right)\right\}=\frac{1}{\sqrt{\pi t}}\exp\left(-\frac{x^2}{4Dt}\right) \tag{B.20}$$

and

$$\mathcal{L}^{-1}\left\{G(s)K(s)\right\}=\int_0^t g(\tau)k(t-\tau)d\tau, \tag{B.21}$$

the concentration as a function of space and time is given by

$$u(x,t)=u_0+\frac{1}{F\sqrt{\pi D}}\int_0^t j(\tau)\frac{1}{\sqrt{t-\tau}}\exp\left(-\frac{x^2}{4D(t-\tau)}\right)d\tau. \tag{B.22}$$

The concentration at the cathode (x=0) is given by

$$u(0,t)=u_0+\frac{1}{F\sqrt{\pi D}}\int_0^t j(\tau)\frac{1}{\sqrt{t-\tau}}d\tau. \tag{B.23}$$

Let $u_{\min}$ be the minimum concentration required to start deposition and $\xi$ be the time when the minimum concentration is reached. Therefore

$$u_{\min}=u_0+\frac{1}{F\sqrt{\pi D}}\int_0^{\xi} j(\tau)\frac{1}{\sqrt{\xi-\tau}}d\tau. \tag{B.24}$$

Let $K=\frac{1}{2}\left(u_{\min}-u_0\right)F\sqrt{\pi D}$ be a constant characterizing the deposition onset.

#### B.3.1.1 Constant current density

For a constant current density $j$, the concentration is defined as

$$u(0,t)=u_0+\frac{2j_0}{F\sqrt{\pi D}}\sqrt{t} \tag{B.25}$$

and the electric charge is defined as

$$Q(t)=\int_0^t j dt=j_0 t. \tag{B.26}$$

At the deposition onset, this yields the well-known Sand's equation [154]:

$$\frac{1}{2}\left(u_{\min}-u_0\right)F\sqrt{\pi D}=j_0\sqrt{\xi}. \tag{B.27}$$

The minimum charge can be derived accordingly:

$$Q_{\min} = j_0 \xi = \frac{K^2}{j_0} \tag{B.28}$$

#### B.3.1.2 Linear voltage ramp

If the voltage ramp is a linear function of time. Before the deposition, due to the Ohm's law, the current density is a linear function of time $j = \beta t + j_0$ and the concentration is expressed as

$$\begin{aligned} u(0,t) =& u_0 + \frac{1}{F\sqrt{\pi D}} \int_0^t j(\tau) \frac{1}{\sqrt{t-\tau}} d\tau & \text{(B.29)} \\ =& u_0 + \frac{1}{F\sqrt{\pi D}} \left[ 2 j_0 \sqrt{t} + 2 \int_0^t \frac{\partial j}{\partial \tau} \sqrt{t-\tau} d\tau \right] & \text{(B.30)} \\ =& u_0 + \frac{1}{F\sqrt{\pi D}} \left[ 2 j_0 \sqrt{t} + \frac{4}{3} \beta t^{3/2} \right]. & \text{(B.31)} \end{aligned}$$

At the deposition onset, the induction time is the solution of the following equation

$$\beta \xi^{3/2} + \frac{3}{2} j_0 \xi^{1/2} = \frac{3}{2} K. \tag{B.32}$$

Assuming that the voltage at the anode is initially 0, we can derive the following expression for the induction time

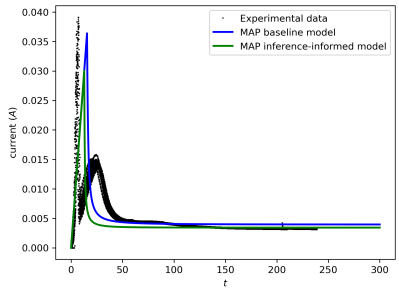
$$\xi = \beta^{-2/3} \left[ \frac{3}{2} K \right]^{2/3} \tag{B.33}$$

Finally, the minimum electric charge for the linear voltage ramp is

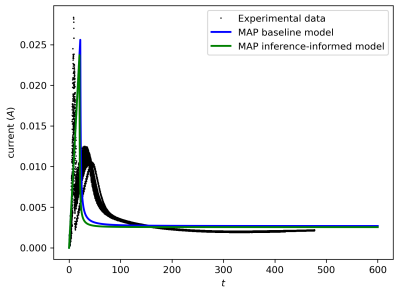
$$Q_{\min} = \left( \frac{81}{128\beta} \right)^{1/3} K^{4/3}. \tag{B.34}$$

## B.4 Baseline / Inference-informed model comparisons

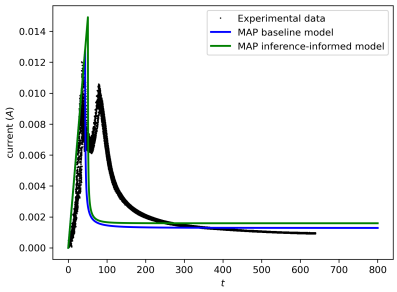
We include here in Figs B.1- B.2 all comparisons between the baseline and inference-informed models. This includes current and resistance comparisons for all 6 experiments. The inference-informed model performs better than the baseline model in all experiments, but the notably greater improvement on predictions for CC experiments. However, thickness prediction is still inaccurate in many cases.



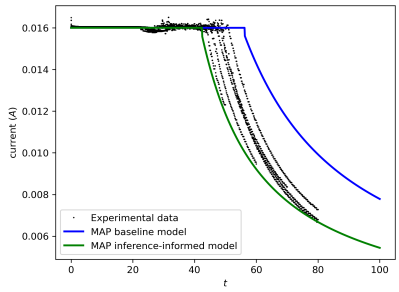

(a) Voltage ramp experiment prediction, $V_R = 1.0V/s$ (current)



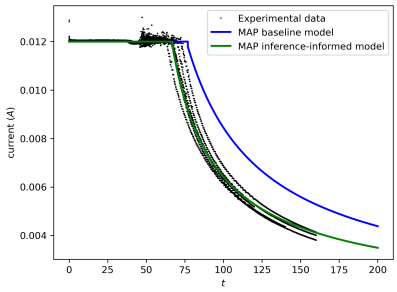

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (current)



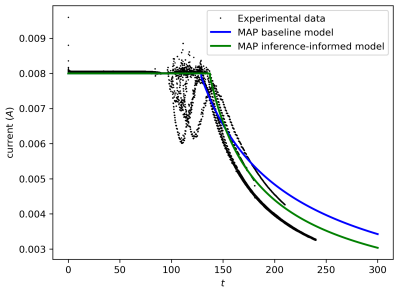

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (current)




(d) Constant current experiment prediction, $j_0 = 10.0mA$ (current)




(e) Constant current experiment prediction, $j_0 = 7.5mA$ (current)




(f) Constant current experiment prediction, $j_0 = 5.0mA$ (current)

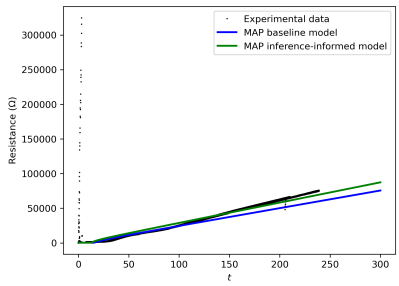
Figure B.1: Comparisons between current prediction on the baseline model and inference-informed model at the MAP for each.

## B.5 ML-augmented model with first-peak

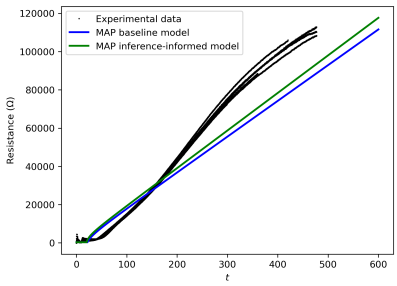
We present all prediction results on the ML-augmented model with first peak. Noteably, predictions are poor for VR experiments in the low $V_R$ regime, but accurate for all other experimental configurations. In particular, thickness prediction is much more accurate compared to the baseline and inference-informed models. Predictions for current and resistance are included in Figs. B.3- B.4, but the ML model is trained with current data for the 3 voltage ramp experiments only.

## B.6 ML-augmented model without first-peak

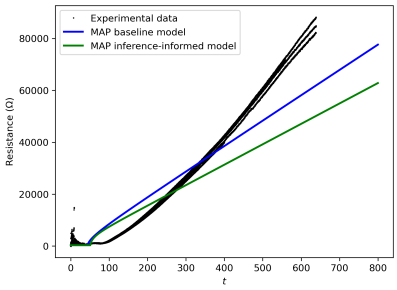
We present all prediction results on the ML-augmented model without the first peak model included. Removing the first peak model greatly improves model efficiency during prediction time (and training time). Predictions are still poor for VR experiments in the low $V_R$ regime,



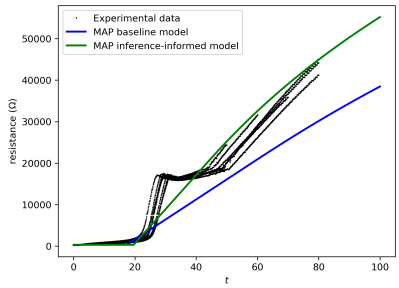

(a) Voltage ramp experiment prediction, $V_R = 1.0$ (resistance)



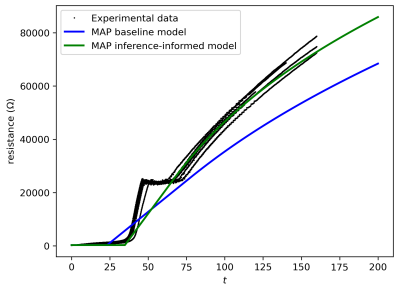

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (resistance)



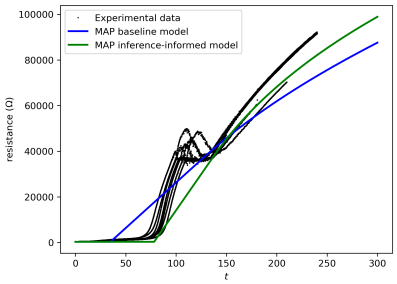

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (resistance)




(d) Constant resistance experiment prediction, $j_0 = 10.0mA$ (resistance)



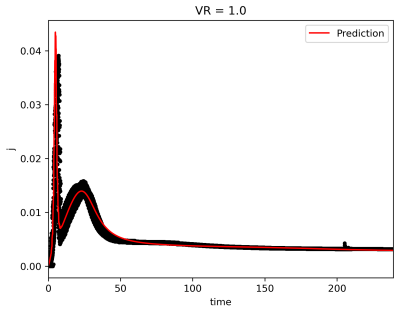

(e) Constant current experiment prediction, $j_0 = 7.5mA$ (resistance)



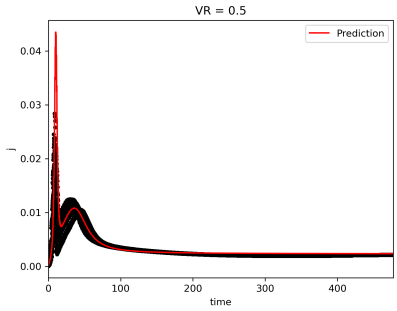

(f) Constant current experiment prediction, $j_0 = 5.0mA$ (resistance)

Figure B.2: Comparisons between resistance prediction on the baseline model and inference-informed model at the MAP for each.

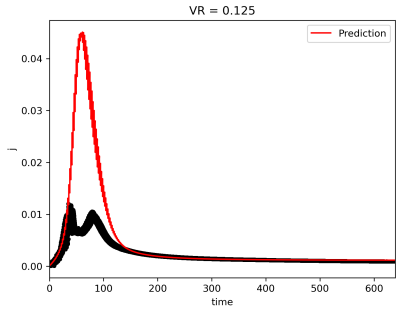
but improved over the baseline and inference-informed models. Again, thickness predictions are more accurate compared to previous models. Predictions for current, resistance, and thickness are included for all six experimental configurations in Figs. B.3- B.6, but the ML model is trained with current data for the 3 voltage ramp experiments only.



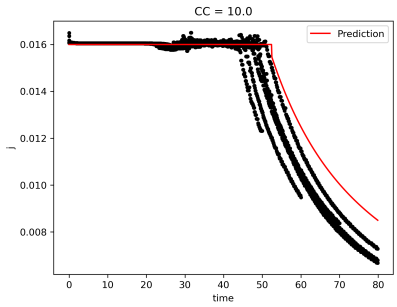

(a) Voltage ramp experiment prediction, $V_R = 1.0V/s$ (current)



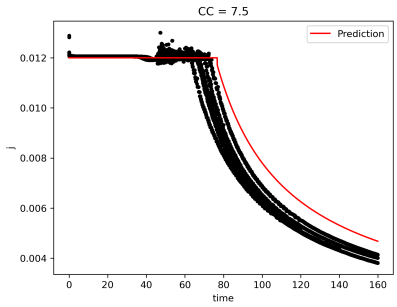

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (current)



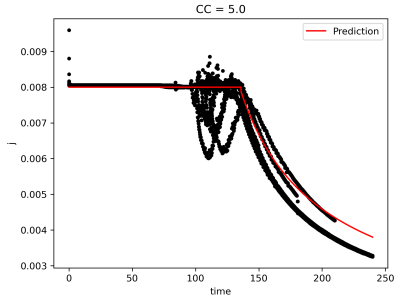

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (current)




(d) Constant current experiment prediction, $j_0 = 10.0mA$ (current)

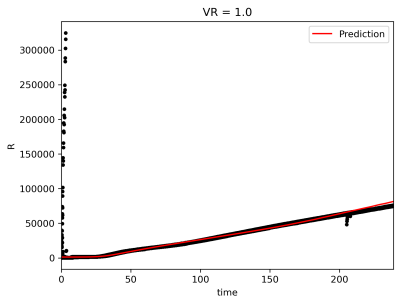



(e) Constant current experiment prediction, $j_0 = 7.5mA$ (current)

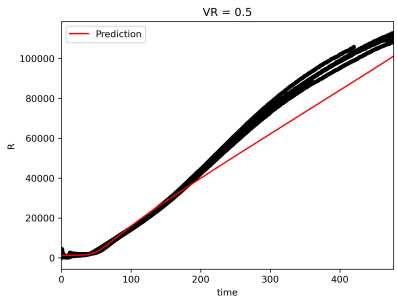



(f) Constant current experiment prediction, $j_0 = 5.0mA$ (current)

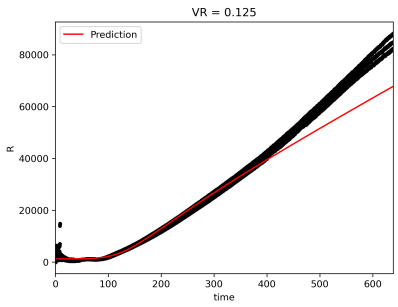
Figure B.3: ML-augmented model with first peak current predictions compared to experimental data.



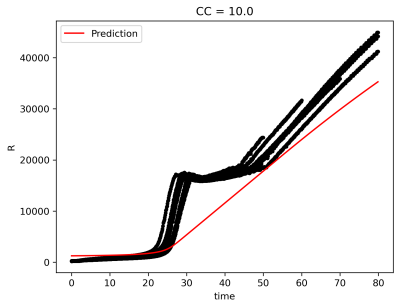

(a) Voltage ramp experiment prediction, $V_R = 1.0$ (resistance)



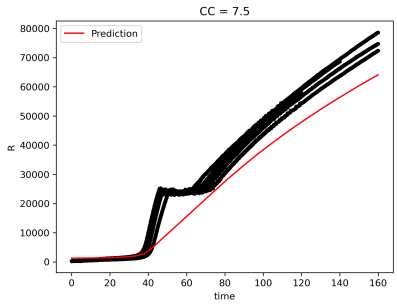

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (resistance)



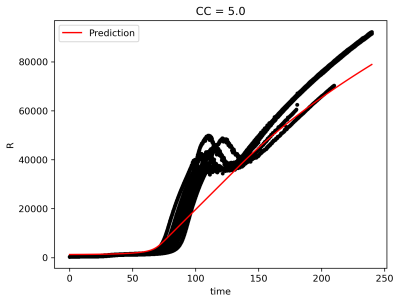

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (resistance)




(d) Constant resistance experiment prediction, $j_0 = 10.0mA$ (resistance)

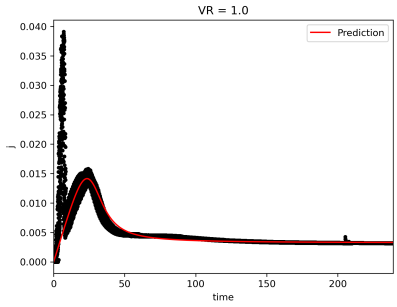



(e) Constant current experiment prediction, $j_0 = 7.5mA$ (resistance)

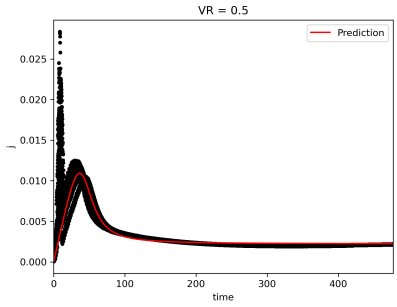



(f) Constant current experiment prediction, $j_0 = 5.0mA$ (resistance)

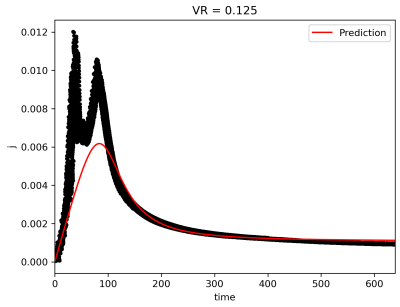
Figure B.4: ML-augmented model with first peak resistance predictions compared to experimental data.



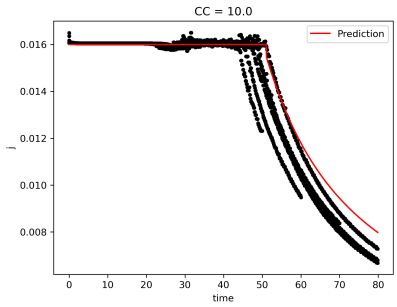

(a) Voltage ramp experiment prediction, $V_R = 1.0V/s$ (current)



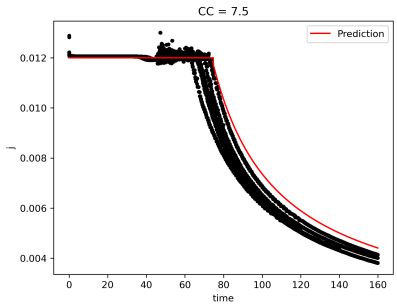

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (current)



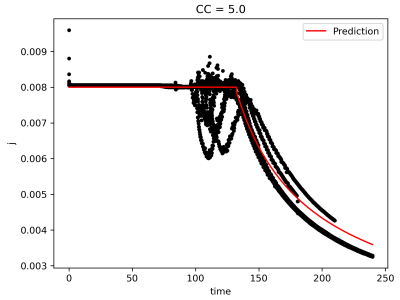

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (current)




(d) Constant current experiment prediction, $j_0 = 10.0mA$ (current)

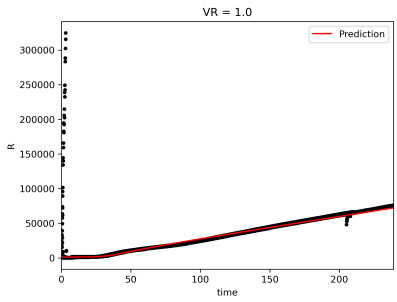



(e) Constant current experiment prediction, $j_0 = 7.5mA$ (current)

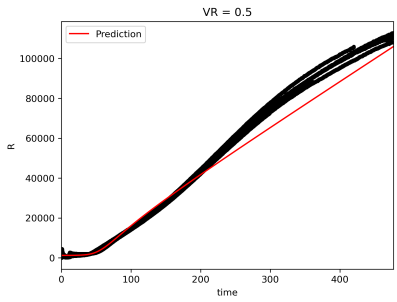



(f) Constant current experiment prediction, $j_0 = 5.0mA$ (current)

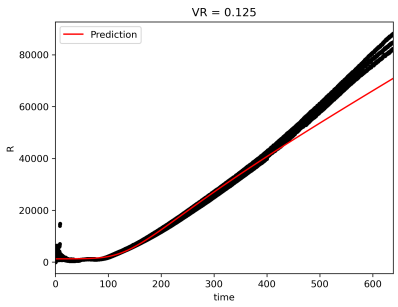
Figure B.5: ML-augmented model without first peak current predictions compared to experimental data.



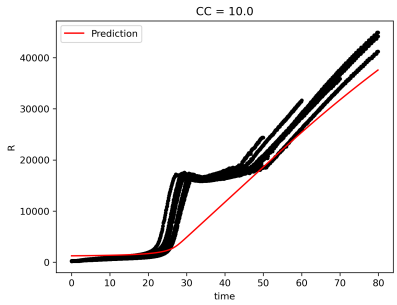

(a) Voltage ramp experiment prediction, $V_R = 1.0$ (resistance)



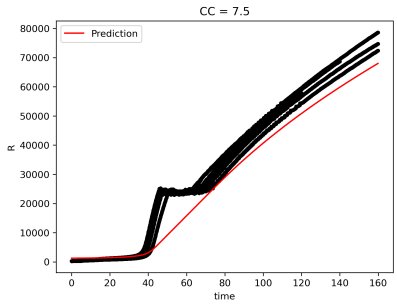

(b) Voltage ramp experiment prediction, $V_R = 0.5V/s$ (resistance)



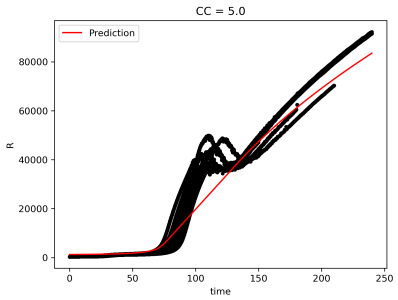

(c) Voltage ramp experiment prediction, $V_R = 0.125V/s$ (resistance)




(d) Constant resistance experiment prediction, $j_0 = 10.0mA$ (resistance)




(e) Constant current experiment prediction, $j_0 = 7.5mA$ (resistance)




(f) Constant current experiment prediction, $j_0 = 5.0mA$ (resistance)

Figure B.6: ML-augmented model without first peak resistance predictions compared to experimental data.

# APPENDIX C

# Rate Distortion Informed Clustering of Non-equilibrium Gas Dynamics

## C.1 Microstates and Macrostates

Consider a system of $N_S$ molecules or atoms of the chemical species $S$ with a total internal energy of the species $U_S$. Given that the system has $\ell$ quantized energy states, then at any instance, a molecule in this system can exist in only one of these finite energy states. The energy state in which a molecule will be present is a direct consequence of the distribution of energy among its different energy modes such as translation, vibrational, rotation, or electronic. If the molecule does not interact with its surroundings, such that there is no re-distribution of energy, then its internal configuration and thereby energy state will remain unchanged. As an analogy, consider $\ell$ boxes that are labeled according to the energy state $\varepsilon_S^i$ for $i \in \{1, \ldots, \ell\}$. At any given time, the $N_S$ number of molecules can be distributed in different ways among $\ell$ boxes corresponding to the *microstate* of the system. The microstate of the system can be thought of as the time snapshot of the configuration of each molecule of the system. The microstate information of each molecule is often excessive, and of interest, is the number of particles that are placed in the $i^{th}$ box $N_S^i$. The number $N_S^i$ along with $\varepsilon_S^i$ specify the *macrostate* of the system. Each macrostate, specified by $N_S^i$, can be realized by a number $w_k$ of microstates, also known as the *thermodynamic probability* of the $k^{th}$ macrostate. This means that molecules can be distributed among the boxes differently, however, while still preserving $N_S^i$. Now consider the energy state $i$ with $N_S^i$ molecules. Even within the energy state $\varepsilon_S^i$ there are $g_S^i$ different molecular internal configurations that will result in the same energy state. Such configurations result in the degeneracy of the energy state $i$, such that for a fixed $N_S^i$ there are $g_S^i$ internal configurations that will lead to the energy state $\varepsilon_S^i$. Think of the degenerate states for an energy state as finite compartments within the box $i$. For a volume, $V$ isolated system with $N_S$ particles and specified total internal energy $U_S$ the

conservation laws are given as

$$N_S \triangleq \sum_i^{\ell} N_S^i, \quad U_S \triangleq \sum_i^{\ell} N_S^i \varepsilon_S^i \quad \text{or} \quad n_S = \sum_i^{\ell} n_S^i, \quad e_S = \sum_i^{\ell} n_S^i \varepsilon_S^i \tag{C.1}$$

where $n_S$ is the number density and $e_S$ is the total internal energy per unit volume for species $S$.

**Remark 1.** *When considering a system of dissimilar species it is often the case that not all of the quantized energy states can be realized. Since the energy state observed is a consequence of the internal configuration (or distribution of energy over modes) of the molecule, it is possible that certain combinations of energy modes will not result in a particular energy state.*

Fig. C.1 illustrates the macrostates and microstates for a system with $N_S = 4$. Configuration 1 and 2 illustrate the same macrostate with microstate configuration $N_S^3 = 1$ and $N_S^4 = 3$. $\varepsilon_S^3$ has a degeneracy $g_S^3 = 5$ and similarly $\varepsilon_S^4$ has a degeneracy $g_S^4 = 7$. In other words, for $\varepsilon_S^3$ all 5 quantized states will result in the same energy state. Similarly, for $\varepsilon_S^4$, all 7 quantized states will result in the same energy state.

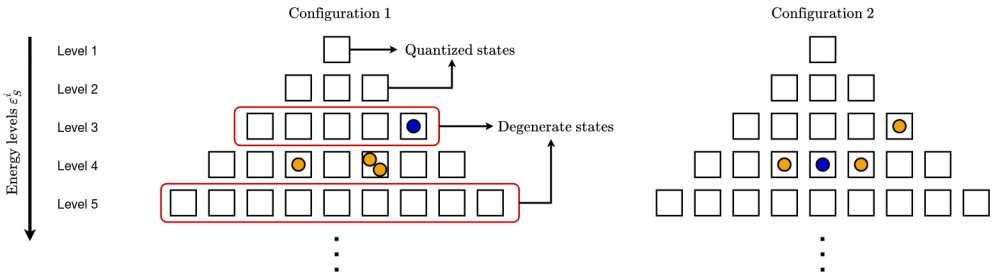



Figure C.1: Illustration of microstates and macrostates using $N_S = 4$. Molecule colors are changed for illustration purposes only.

## C.2 Connections of maximum entropy principle with the Boltzmann distribution

In statistical thermodynamics the Boltzmann distribution is a probability distribution over the realization of the energy states $\varepsilon^i$. Consider a system of volume $V$ with $N$ particles and a total energy $E$. We are interested in the number of ways in which $N$ particles can be distributed among $\ell$ internal energy levels such that there are $N^i$ particles in the $i^{th}$ energy

level. This is given by the multiplicity function as

$$Q(N^1, N^2, \ldots, N^\ell) = Q \triangleq \frac{N!}{\prod_i^\ell N^i!}, \tag{C.2}$$

where $N^i$ is the number of particles in energy state $i$. Since for each energy level there exists a degeneracy $g^i$ such that $N^i$ particles can be further arranged in $(g^i)^{(N^i)}$ ways. Thus, the total multiplicity function is given as

$$Q = \frac{N!}{\prod_{i=1}^\ell N^i!} \prod_{i=1}^{\ell} (g^i)^{(N^i)}, \tag{C.3}$$

$$\ln Q = \ln N! + \sum_{i=1}^{\ell} N^i \ln g^i - \sum_{i=1}^{\ell} \ln N^i!. \tag{C.4}$$

Applying Stirling's approximation ($\ln x! \approx x \ln x - x$ for large $x$),

$$\ln Q = N \ln N - N + \sum_{i=1}^{\ell} N^i \ln g^i - \sum_{i=1}^{\ell} N^i \ln N^i + \sum_{i=1}^{\ell} N^i. \tag{C.5}$$

Maximizing (C.5) with respect to $N^i$ under the constraints (C.1) means that the entropy of the system is being maximized. Consider the Lagrangian

$$\frac{\partial \ln Q}{\partial N^j} + \alpha \underbrace{\frac{\partial \phi}{\partial N^j}}_{\triangleq 0} - \beta \underbrace{\frac{\partial \psi}{\partial N^j}}_{\triangleq 0} = 0, \tag{C.6}$$

where $\phi = N$ and $\psi = U$ are constants by conservation principles. Substituting (C.6) into (C.5) gives

$$\begin{aligned} \frac{\partial}{\partial N^j}\Big(N \ln N - N + \sum_{i=1}^{\ell} N^i \ln g^i - \sum_{i=1}^{\ell} N^i \ln N^i + \sum_{i=1}^{\ell} N^i\Big) \\ + \alpha \frac{\partial}{\partial N^j}\Big(\sum_{i=1}^{\ell} N^i\Big) - \beta \frac{\partial}{\partial N^j}\Big(\sum_{i=1}^{\ell} N^i \varepsilon^i\Big) = 0, \end{aligned} \tag{C.7}$$

$$\ln g^j - \Big(1 \cdot \ln N^j + N^j \cdot \frac{1}{N^j}\Big) + 1 + \alpha - \beta \varepsilon^j = 0, \tag{C.8a}$$

$$\ln g^j - \ln N^j + \alpha - \beta \varepsilon^j = 0. \tag{C.8b}$$

Rearranging (C.8b) gives

$$\frac{N^j}{g^j} = e^{\alpha - \beta\varepsilon^j}. \tag{C.9}$$

Using the constraint (C.1)

$$N = \sum_{j=1}^{\ell} N^j = e^{\alpha} \sum_{j=1}^{\ell} g^j e^{-\beta\varepsilon^j}. \tag{C.10}$$

Therefore

$$\alpha = \ln \frac{N}{\sum_{j=1}^{\ell} g^j e^{-\beta\varepsilon^j}}, \tag{C.11}$$

which gives

$$\frac{N^j}{N} = g^j \frac{e^{-\beta\varepsilon^j}}{\sum_{j=1}^{\ell} g^j e^{-\beta\varepsilon^j}}, \tag{C.12}$$

$$\frac{n^j}{n} = g^j \frac{e^{-\beta\varepsilon^j}}{\sum_{j=1}^{\ell} g^j e^{-\beta\varepsilon^j}}. \tag{C.13}$$

The value of $\beta = 1/k_N T$ comes from energy conservation.

# APPENDIX D

# Physically Consistent Diffusion Model Sampling for Solving Forward and Inverse Problems

## D.1 Unconditional Model and Training Details

Unconditional models for both the Darcy flow $s = 16$ and $s = 256$ datasets are identical in architecture and identical in training, containing 17.7M trainable parameters. We use a U-Net-type architecture consisting of convolution-based encoding and decoding layers described in the main text as the unconditional score-approximation model. This model is trained for 5000 epochs with a learning rate of $10^{-4}$ and batch size of 128. The SDE process which we employ is the variance-preserving SDE [99] with a linear $\beta(t)$ function corresponding to $\beta_{min} = 10^{-4}$, $\beta_{max} = 10$, and $T = 1$. The optimization took 24.5 hours to complete 200,000 optimization iterations on 4 NVIDIA A6000 GPUs, which is sufficient to achieve convergence. This time includes sampling a batch of 8 fields every 1,000 optimization steps with each batch of samples taking an average of 31 seconds to obtain.

## D.2 Conditional Model and Training Details

The conditional model architectures are exemplified by the ControlNet-based architecture in the main text. Each conditional augmentation has a slightly different architecture due to the need for the condition encoder to accept various forms in input. However, training is performed in an identical way to the unconditional training in D.1.

In the case of surrogate modeling, we use a condition encoder consisting of 3 linear layers followed by a reshaping operator and convolutional layers. The entire conditional augmentation in this case consists of 5.8M trainable parameters, with the 17.7M parameters in the unconditional model frozen during training. This results in a total of 23.5M parameters

in the model. The optimization took 11.6 hours to achieve convergence for a total of 100,000 optimization iterations, less than half as long as training the unconditional model.

The ControlNet architecture for performing field reconstruction and inversion is identical to that of the surrogate model ControlNet with the exception of the condition encoder. Here, the condition encoder consists of 3 convolutional layers only as the conditioning data is of the same shape as the input data. This results in a conditional augmentation with 5.7M trainable parameters, bringing the total parameters in the model to 23.4M including the frozen unconditional model. The optimization took 18.1 hours to achieve convergence for a total of 160,000 optimization iterations, a significant speedup over the unconditional model training.

## D.3 Analysis of Conditional Score-based Generative Modeling

This section provides a rigorous analysis of conditional score-based generative models in representing a conditional distribution when the true conditional is known. It further demonstrates the limitations of learning the true time dependent score function brought on by data requirements.

### D.3.1 Dataset description

The conditional relationship which the score-based generative model (SGBM) is trained to learn in this examples is given by $p(\mathbf{y}|\theta)$, where samples satisfy the relationship

$$\mathbf{y} = \begin{bmatrix} \theta x_1 \\ x_2/\theta - \theta^2(x_1^2 + 1) \end{bmatrix}, \tag{D.1}$$

and $p(\mathbf{x}) = \mathcal{N}(0, \mathbf{I})$).

The distribution $p(\theta)$ is set to be a uniform distribution $\mathcal{U}[\theta_{min}, \theta_{max}] = \mathcal{U}[0.2, 1.0]$, and the training dataset is generated by sampling $N$ times from the joint distribution $p(\mathbf{y}, \theta)$. However, the unconditional model SBGM is first trained on only the marginal distribution $p(\mathbf{y})$ before incorporating conditioning information.

The marginal distribution $p(\mathbf{y})$ is illustrated approximately in Figure D.2, while the true conditional PDF $p(\mathbf{y}|\theta)$ is illustrated in Figure D.1 for a range of conditioning values $\theta$.

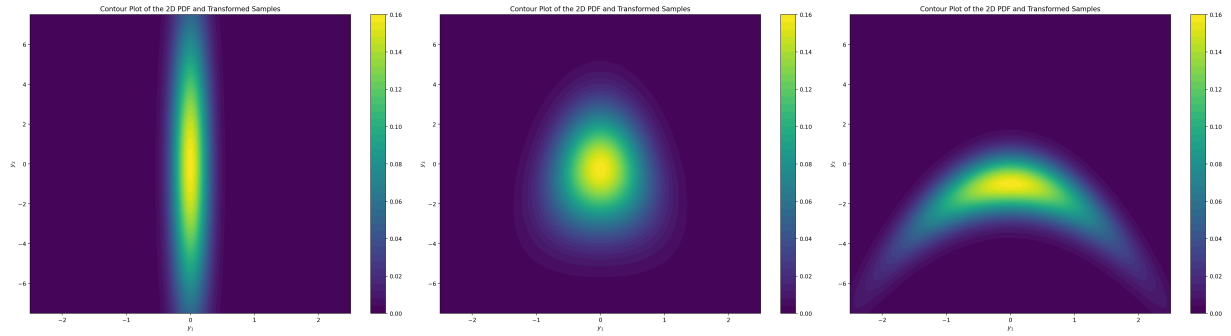



Figure D.1: Probability density functions of the conditional relationship $p(\mathbf{y}|\theta)$ *(left)* $p(\mathbf{y}|\theta = 0.2)$ *center* $p(\mathbf{y}|\theta = 0.5)$ *(right)* $p(\mathbf{y}|\theta = 1.0)$

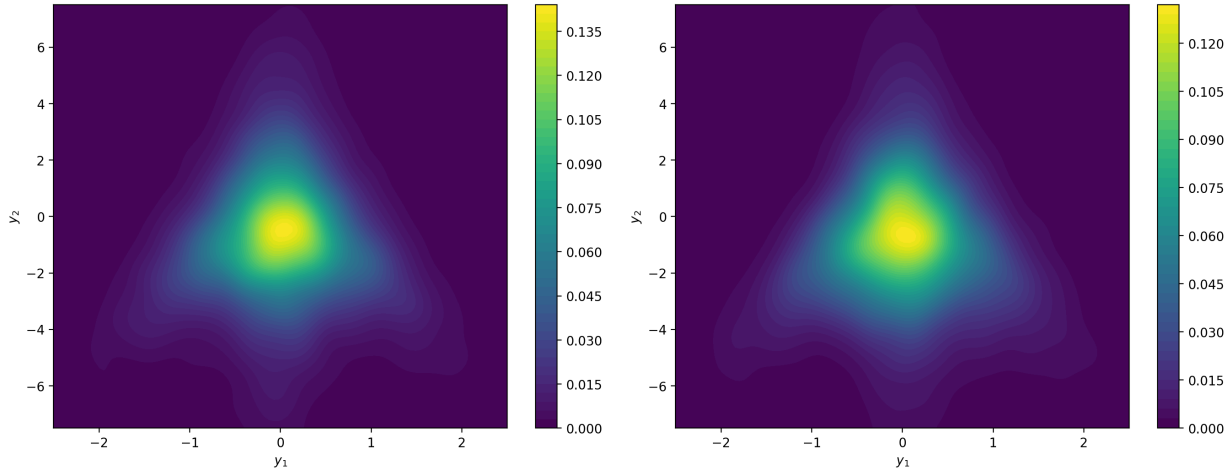



Figure D.2: *(left)* Empirical distribution of dataset sampled with $N = 10,000$ from $p(\mathbf{y})$. *(right)* Empirical distribution of dataset sampled $10,000$ times from a trained unconditional SBGM $\hat{p}(\mathbf{y})$.

### D.3.2 Unconditional SBGM Training

We train an unconditional SBGM on the data distribution $p(\mathbf{y})$ to approximate it by $\hat{p}(\mathbf{y})$. We use the same VP SDE form discussed in Chapter 6, along with identical hyperparameters. However, the model architecture is a much simpler fully-connected architecture with 4 layers and 128 nodes per layer. The input to the network is the concatenation of a sinusoidal embedding of the input data and a sinusoidal embedding of the SDE time.

The unconditional SBGM is trained multiple times, each time with a different number of training samples. We train the model with $N =$ 1,000, 5,000, 10,000, and 100,000 training samples. Figure D.2 illustrates the empirical distribution (approximated through kernel density estimation (KLE)) of the $N = 10,000$ dataset sampled from $p(\mathbf{y})$ along with that generated by $10,000$ samples from the trained unconditional SBGM $\hat{p}(\mathbf{y})$. Qualitatively, the

distributions appear to be very similar and match well. Quantitatively, the KL divergence is approximated between the KLE approximations to be $DKL[\hat{p}(\mathbf{y})||p(\mathbf{y})] = 0.0146$.

### D.3.3 Conditional SBGM Training

We next train a conditional augmentation to learn a conditional SBGM which approximately samples from $p(\mathbf{y}|\theta)$. Note that the training dataset is generated by sampling from the joint distribution $p(\mathbf{y}, \theta)$. Although the unconditional SBGM is trained only on data from marginal $p(\mathbf{y})$, the dataset has the corresponding values $\theta$ recorded. We thus train a conditional augmentation to sample from $p(\mathbf{y}|\theta)$ using the same training dataset, but with the conditioning information included. We train the conditional augmentations on the four different training datasets corresponding to $N = 1{,}000$, 5,000, 10,000, and 100,000 samples.

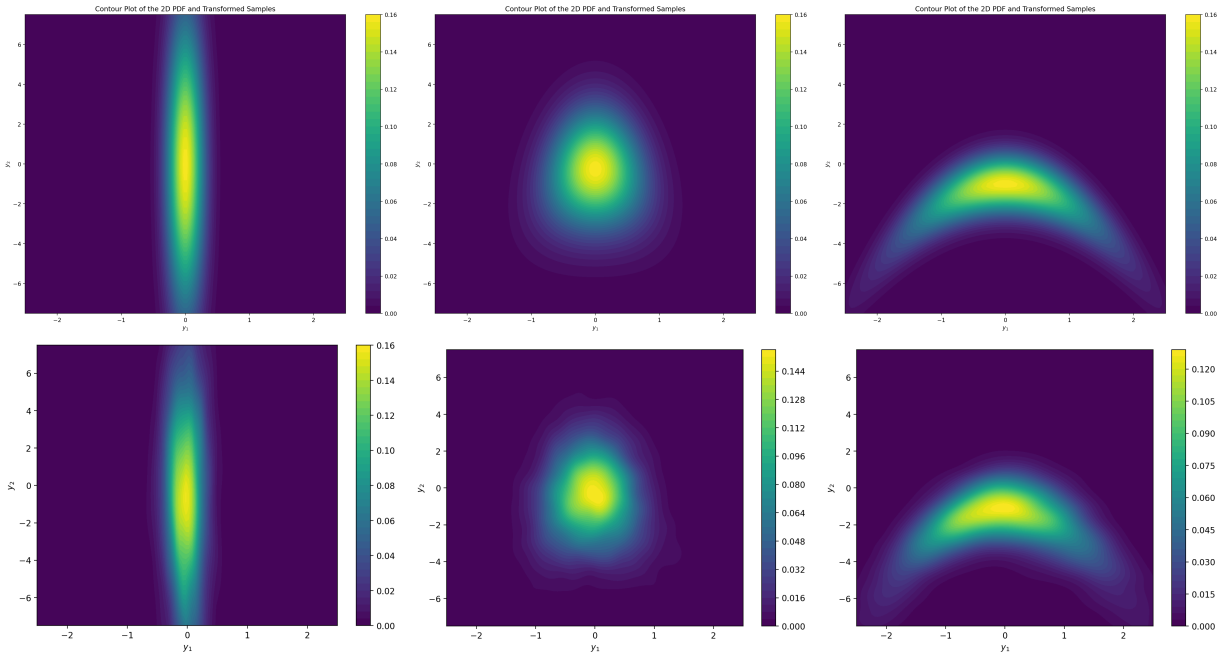



Figure D.3: Comparison of conditional probability density functions for models trained with $N = 10,000$ samples. *(upper)* Analytic relationship $p(\mathbf{y}|\theta)$ *(lower)* Approximate distribution of trained conditional SBGM $\hat{p}(\mathbf{y}|\theta)$ by generating 10,000 samples with *(left)* $\theta = 0.2$ *(center)* $\theta$ *(right)* $\theta$

After model training, we sample from the SBGM for various values of $\theta$. For each value, a dataset of 10,000 samples is generated, and KDE is used to estimate the distribution. Figure D.4 illustrates the KL divergence between the true conditional distribution (computed analytically) and the approximated distribution of the generated data for different values of $\theta$. This experiment is repeated by training the unconditional model from scratch

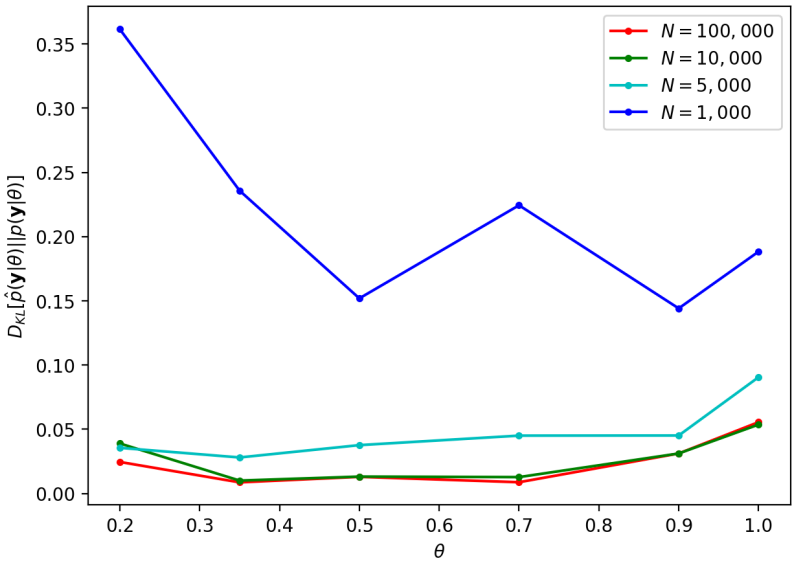



Figure D.4: KL divergence between analytic conditional distribution $p(\mathbf{y}|\theta)$ and approximate distribution from sampling 10,000 times from conditional SBGM $\hat{p}(\mathbf{y}|\theta)$ as a function of conditioning parameter $\theta$. Results shown for models trained by various training dataset sizes $N$.

with $N$ samples, and training the conditional augmentation afterwards (repeated for $N =$ 1,000, 5,000, 10,000, and 100,000) samples. We find that the KL divergence increases as $\theta$ approaches the bounds of the training data ($p(\theta) = \mathcal{U}[0.2, 1.0]$), likely due to the well known extrapolative difficulty which most ML models demonstrate. Additionally, without sufficient data during training, the SBGM fails to accurately model the time dependent score function, reducing the accuracy in sampling from the true data distribution. However, once a sufficient number of samples are used in training, there appears to be diminishing returns to the accuracy in representing the conditional distributions. The remaining inaccuracy in modeling the true distribution is likely due to the expressiveness (more accurately, inexpressiveness) of the model architecture and the discretization of the reverse SDE.

Further, qualitative comparisons between the generated distributions and the true conditional PDFs are shown in Figure D.3, demonstrating a good approximation to the conditional distribution on the range of training data for $N = 10,000$ samples.

### D.3.4 Discussion on Physical Consistency Steps

In the experiments presented in Chapter 6, we apply physical consistency steps, which may lead to one hypothesizing that this will alter the dynamics of the reverse SDE, and potentially alter the distribution of samples generated by the SBGM. However, this is not the case.

Consider an example in which the data is generated through the relationship

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x \\ x^2 \end{bmatrix} , \tag{D.2}$$

where $x \sim p(x)$. In this case, $y_2$ is a function of $y_1$. Thus, accurately modeling $p(y_1) = p(x)$ removes the necessity for modeling the uncertainty in $y_2$, as long as the relationship between the two ($y_2 = y_1^2$) is accurately modeled. Therefore, if a SBGM generates both $y_1$ *and* $y_2$, as long as the distribution $p(y_1)$ and the relationship $y_2 = f(y_1)$ are accurately modeled, the SBGM will accurately generate samples from $p(y_1, y_2) = p(\mathbf{y})$. We assume that the SBGM will accurately sample from $p(y_1)$, but applying 'physical consistency steps' to enforce the relationship $y_2 = f(y_1)$ will only improve the modeling of $p(\mathbf{y})$.

This argument can be easily extended to the applications of SBGM's to Darcy flow in Chapter 6. Given a permeability field $K(\mathbf{x})$, the pressure field $p(\mathbf{x})$ is a deterministic function of the permeability related through some function $p(\mathbf{x}) = f(K(\mathbf{x}))$. Therefore, if the SBGM accurately approximates the distribution on permeability fields $p(K(\mathbf{x}))$ and the relationship $p(\mathbf{x}) = f(K(\mathbf{x}))$, then the distribution $p(K(\mathbf{x}), p(\mathbf{x}))$ will be accurately modeled. We rely on the SBGM to model $p(K(\mathbf{x}))$, but we aid it in representing the relationship $p(\mathbf{x}) = f(K(\mathbf{x}))$ through physical consistency sampling, enforcing this relationship through the minimization of the PDE residual on the pressure fields only.

## D.4 Analytic Approximation to Conditional Sampling - Field Reconstruction and Inversion

There exist a few special cases of conditional sampling in which it is possible to approximate conditional sampling after training only an unconditional generative model. Data imputation is one of such cases and attempts to predict or reconstruct unknown dimensions of data given some particular measured values of known dimensions. This is a conditional problem in which the aim is to sample from a conditional distribution in which conditioning information is the particular measured values $\omega$ of some known dimensions $\Omega(\mathbf{y}(0))$. The particular distribution to sample from is defined as $p(\mathbf{z}|\Omega(x(0)) = \omega)$, where $\bar{\Omega}(\mathbf{y}(0)) = \mathbf{z}$ corresponds to the unknown dimensions. Given a trained unconditional score-based generative model, an approximation to this conditional sampling can be performed without any need for further training [99]. In our experiments, we measure only Darcy flow pressure fields at various spatial locations. From these measurements, the entire pressure field is reconstructed and an inverse problem is effectively solved to also approximate the permeability field. The

examples in this work are intended to illustrate a framework which can be applied to any number of physical systems, effectively using measurements to predict physical fields which are consistent with governing equations. Although we perform this procedure with a closed form approximation based on a pretrained unconditional model, the technique can also be performed by training a conditional model as described in Sec. 6.1.2 of the main text.

We perform data imputation by following the ideas from [99] in which $\Omega(\mathbf{y})$ denotes the known (measured) dimensions of $\mathbf{y}$ and $\bar{\Omega}(\mathbf{y})$ denotes the unknown dimensions of $\mathbf{y}$. Further, $f_{\bar{\Omega}}(\cdot, t)$ and $g_{\bar{\Omega}}(t)$ define a restriction of the operators $f(\cdot, t)$ and $g(t)$ in the SDE to the unknown dimensions only.

As some portion of $\mathbf{y}$ is known, a new diffusion process is defined such that $\mathbf{z}(t) = \bar{\Omega}(\mathbf{y}(t))$ and

$$d\mathbf{z} = f_{\bar{\Omega}}(\mathbf{z}, t)dt + g_{\bar{\Omega}}(t)dw \ .$$

However, we do not desire to simply sample from $p(z(t = 0))$, but rather $p(\mathbf{z}(t = 0)|\Omega(\mathbf{y}(t = 0)) = \hat{\mathbf{y}})$, where $\hat{\mathbf{y}}$ are the measured values. The reverse process is therefore defined as

$$d\mathbf{z} = [f_{\bar{\Omega}}(\mathbf{z}, t) - g_{\bar{\Omega}}(\mathbf{z}, t)^2 \nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\Omega(\mathbf{y}(0)) = \hat{\mathbf{y}})]dt + g_{\bar{\Omega}}(t)dw$$

It is shown in [99] that $\nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\Omega(\mathbf{y}(0)) = \hat{\mathbf{y}})$ can be approximated by $\nabla_{\mathbf{z}} \log p_t(\mathbf{u}(t))$ where $\mathbf{u}(t) = [\mathbf{z}(t); \hat{\Omega}(\mathbf{y}(t))]$ is a vector such that $\Omega(\mathbf{u}(t)) = \hat{\Omega}(\mathbf{y}(t))$ and $\bar{\Omega}(\mathbf{u}(t)) = \mathbf{z}(t)$. The quantity $\hat{\Omega}(\mathbf{y}(t))$ denotes a random sample from $p_t(\Omega(\mathbf{y}(t))|\Omega(\mathbf{y}(0)) = \hat{\mathbf{y}})$, which can typically be obtained by sampling the corresponding known dimensions from the forward SDE process as the dimensions of $\mathbf{y}$ are uncorrelated in the forward process (Eq. 5.42 of the main text). The vector $\mathbf{u}(t)$ contains all known and unknown components of $\mathbf{y}$ and therefore $s_\theta(\mathbf{y}(t), t) \approx \nabla_{\mathbf{z}} \log p_t(\mathbf{u}(t)) \approx \nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\Omega(\mathbf{y}(0)) = \hat{\mathbf{y}})$. Thus, an approximation to $\nabla_{\mathbf{z}} \log p_t(\mathbf{z}|\Omega(\mathbf{y}(0)) = \hat{\mathbf{y}})$ can be computed by leveraging the score function approximation $s_\theta(\mathbf{y}(t), t)$ without any additional special training by sampling the known dimensions of $\mathbf{y}(t)$ at each time step according to the forward process. Further, sampling with physical consistency steps (Sec. 6.1 of the main text) can still be performed in the same manner, leading to samples which are consistent with the governing PDE.

It is worth mentioning that performing data imputation in this special case can be orders of magnitude more expensive when solving the PF ODE reverse process rather than the reverse SDE process. It is illustrated in Sec. 6.1 of the main text that solving the PF ODE reverse process results in improved physical residual minimization with smaller PDE operator evaluations when compared to the reverse SDE. However, the outlined method of data imputation requires sampling from the forward process to obtain $\hat{\Omega}(\mathbf{y}(t))$. Sampling in time via the forward PF ODE process involves solving the PF ODE forward in time. In

Table D.1: Field reconstruction and inversion (no RePaint) with various number of pressure measurements (Darcy Flow, $s = 16$). Analytic approximation to data imputation used in sampling.

| Equation | $m$ | $\mathbb{E}_{p_S(\mathbf{y})}[\|\|\mathbf{r}\|\|_2^2]$ | $\mathbb{E}_{p_S(\mathbf{y})}[\|\|\mathbf{r}\|\|_2^2] - \mathbb{E}_{p_D(\mathbf{y})}[\|\|\mathbf{r}\|\|_2^2]$ | $\mathbb{E}_{p_D(\mathbf{y})}[\|\|P - \hat{P}\|\|_2^2]$ | $\mathbb{E}_{p_D(\mathbf{y})}[\|\|K - \hat{K}\|\|_2^2]$ |
|---|---|---|---|---|---|
| SDE | 0 | 3.6 | -0.9 | $1.02 \times 10^{-1}$ | $8.8 \times 10^{-3}$ |
| SDE | 10 | 224.1 | 219.6 | $1.04 \times 10^{-1}$ | $9.5 \times 10^{-3}$ |
| SDE | 50 | 1463.5 | 1459.0 | $8.73 \times 10^{-2}$ | $9.1 \times 10^{-3}$ |
| SDE | 100 | 3986.1 | 3981.6 | $5.92 \times 10^{-2}$ | $8.5 \times 10^{-3}$ |
| SDE | 250 | 4142.1 | 4137.6 | $3.25 \times 10^{-2}$ | $7.8 \times 10^{-3}$ |
| SDE | 500 | 1970.2 | 1965.7 | $9.1 \times 10^{-3}$ | $6.4 \times 10^{-3}$ |
| SDE | 1000 | 183.8 | 179.3 | $5.0 \times 10^{-4}$ | $4.3 \times 10^{-3}$ |
| SDE | 2000 | 7.5 | 3.0 | $3.6 \times 10^{-6}$ | $2.8 \times 10^{-3}$ |
| SDE | 4000 | 7.6 | 3.1 | $2.4 \times 10^{-7}$ | $2.0 \times 10^{-3}$ |

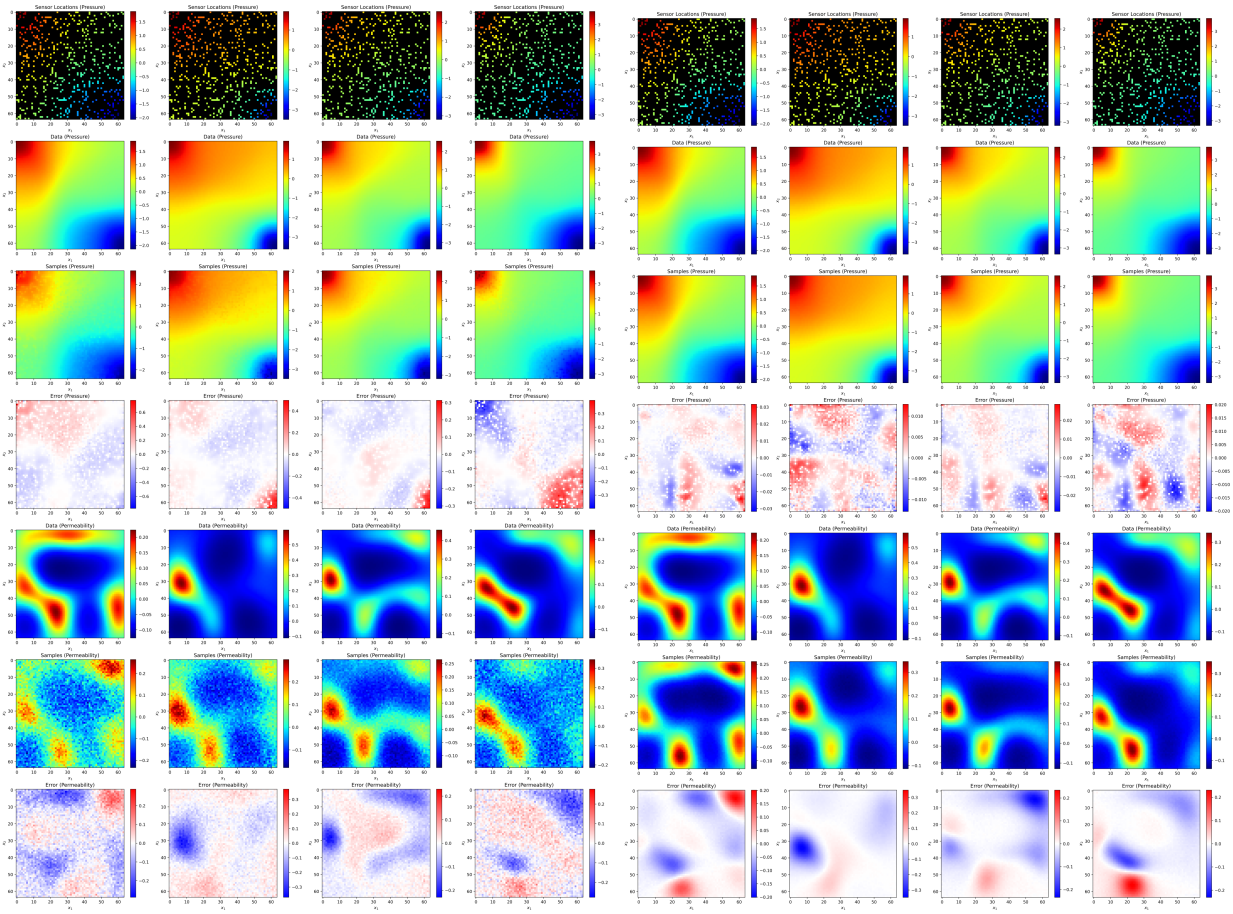
contrast, the forward SDE has a closed form solution to obtain samples at any future time (Eq. 2.24 in the main text), rendering data imputation (field reconstruction and inversion) far more feasible and inexpensive.

Table D.2: Field reconstruction and inversion with RePaint performance with the number of repainting steps $r$ (Darcy Flow, $s = 16$, $m = 500$).

| Equation | $r$ | $\mathbb{E}_{p_S(\mathbf{y})}[\|\|\mathbf{r}\|\|_2^2]$ | $\mathbb{E}_{p_S(\mathbf{y})}[\|\|\mathbf{r}\|\|_2^2] - \mathbb{E}_{p_D(\mathbf{y})}[\|\|\mathbf{r}\|\|_2^2]$ | $\mathbb{E}_{p_D(\mathbf{y})}[\|\|P - \hat{P}\|\|_2^2]$ | $\mathbb{E}_{p_D(\mathbf{y})}[\|\|K - \hat{K}\|\|_2^2]$ |
|---|---|---|---|---|---|
| SDE | 1 | 750.7 | 746.2 | $4.9 \times 10^{-3}$ | $4.3 \times 10^{-3}$ |
| SDE | 2 | 26.2 | 21.7 | $3.0 \times 10^{-4}$ | $2.9 \times 10^{-3}$ |
| SDE | 3 | 8.7 | 4.2 | $3.1 \times 10^{-5}$ | $3.8 \times 10^{-3}$ |
| SDE | 5 | 10.6 | 6.1 | $1.9 \times 10^{-5}$ | $2.9 \times 10^{-3}$ |
| SDE | 10 | 9.6 | 5.1 | $1.7 \times 10^{-5}$ | $4.8 \times 10^{-3}$ |

We use the same score-based generative model defined in the experiments of Sec. 6.1 in the main text in our experiments. However, only the reverse SDE is solved and not the PF ODE due to aforementioned inefficiencies. We perform the baseline data imputation approximation for various number of measured dimensions. The number of measurements is defined as $m$. To perform data imputation, we randomly select $m$ spatial locations to measure only pressure in the physical domain, similar to Sec. 6.3.1.1 in the main text. The random seed is kept constant such that for two cases in which $m_2 > m_1$, the spatial locations of case 1 are a subset of the spatial locations of case 2.

For each experiment, we vary only the number of pressure measurements used to reconstruct the pressure field and invert the permeability field. The reverse process uses an



(a) $r = 0$ (b) $r = 5$

Figure D.5: Field reconstruction and inversion (a) without and (b) with RePaint. RePaint incorporates more information about the known dimensions into the unknown dimensions. Figure is high quality, zoom in for more clarity.

Euler-Maruyama time integration method to solve the reverse SDE with $\tau = 2,000$ time steps. We also use $N = 50$ physical consistency steps and $M = 10$ additional physical consistency steps. For these experiments, we illustrate results on the Darcy flow $s = 16$ dataset only.

Pressure data is denoted $P$ with reconstructions denoted $\hat{P}$. Similarly, permeability data and reconstructions are denoted $K$ and $\hat{K}$ respectively. Table D.1 gives quantitative reconstruction results, and Fig. D.5a illustrates qualitative reconstruction on the test set. Although the reconstruction for both pressure and permeability fields improves with increasing number of samples, the physical residual is very high, particularly for smaller number of measurements, even with physical consistency enforcement. This is likely due to the incoherence between the known (measured) and unknown regions of the data samples. It is evident in

Fig. D.5 that the measured points do not match well with the reconstructed points.

To reduce this discrepancy, we turn towards a recent work addressing this issue called RePaint [110]. This effectively alters between reverse and forward steps to reduce the boundary discrepancy between measured and unmeasured regions of the data samples. At each discrete time step $i$ of the reverse SDE solver, an additional $r$ *resampling* steps are performed. A resampling step consists of solving the forward SDE for a single time step, followed by solving the reverse SDE for the same time step, effectively alternating $r$ times between stepping forward and stepping backward in time according to the SDE. This allows better mixing of the measured and unmeasured dimensions of the data, adding artificial noise before denoising, effectively incorporating more information about the known dimensions to the unknown dimensions at each resampling step. However, this comes at a significant cost: the number of score function evaluations scales linearly with $r$. This can be partially alleviated by performing $r$ resampling steps only at some discrete pre-selected number of time steps instead of at each time step. However, this presents additional difficulties such as determining the optimal time steps to perform resampling at. In our experiments, we perform resampling $r$ times at each time step.

We investigate the power of RePainting by setting a constant number of pressure measurements ($m = 500$), and solving the reverse SDE with various numbers of resampling steps. Table D.2 illustrates the results of this experiment, showing that the resampling procedure greatly improves reconstruction performance and drastically reduces physical residuals over the standard method of sampling without RePaint ($r = 0$). Additionally, reconstructions rapidly improve with increasing number of resampling steps. Sampling with RePaint and $r = 3$ reduces the average physical residual from 1970.2 to 8.7 over sampling without RePaint. The average reconstruction error is also reduced from $9.1 \times 10^{-3}$ to $3.1 \times 10^{-5}$ and the field inversion error is reduced from $6.4 \times 10^{-3}$ to $3.8 \times 10^{-3}$. Figure D.5b qualitatively shows the effect that RePaint has on sample quality. When compared to sampling without RePainting (Fig. D.5a), reconstructions are far more accurate and predictions less noisy, especially permeability fields. This is a direct result of reducing the boundary discrepancy using the RePainting algorithm, leading to de-noised samples.

Performing field reconstruction and inversion as described in this section outperforms the method of conditional generation using ControlNet detailed in the main text in terms of reconstruction performance. However, using the repainting algorithm linearly increases the cost of sampling as a function or $r$. Thus, the best results here require around 5x more compute time to obtain over those obtained using ControlNet-based augmentations for conditional generation.

# BIBLIOGRAPHY


[1] Jiayang Xu and Karthik Duraisamy. Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics. Computer Methods in Applied Mechanics and Engineering, 372:113379, 2020.

[2] Steven L. Brunton, J. Nathan Kutz, Krithika Manohar, Aleksandr Y. Aravkin, Kristi Morgansen, Jennifer Klemisch, Nicholas Goebel, James Buttrick, Jeffrey Poskin, Adriana W. Blom-Schieber, Thomas Hogan, and Darren McDonald. Data-driven aerospace engineering: Reframing the industry with machine learning. AIAA Journal, 59(8):2820–2847, 2021.

[3] Venkat Raman and Malik Hassanaly. Emerging trends in numerical simulations of combustion systems. Proceedings of the Combustion Institute, 37(2):2073–2089, 2019.

[4] Joaquim R.R.A. Martins. Aerodynamic design optimization: Challenges and perspectives. Computers & Fluids, 239:105391, 2022.

[5] Marco Ceze and Krzysztof J. Fidkowski. Drag prediction using adaptive discontinuous finite elements. Journal of Aircraft, 51(4):1284–1294, 2014.

[6] Tony Hey, Stewart Tansley, Kristin Tolle, and Jim Gray. The Fourth Paradigm: Data-Intensive Scientific Discovery. Microsoft Research, October 2009.

[7] Soledad Le Clainche, Esteban Ferrer, Sam Gibson, Elisabeth Cross, Alessandro Parente, and Ricardo Vinuesa. Improving aircraft performance using machine learning: A review. Aerospace Science and Technology, 138:108354, 2023.

[8] Jichao Li, Xiaosong Du, and Joaquim R.R.A. Martins. Machine learning in aerodynamic shape optimization. Progress in Aerospace Sciences, 134:100849, 2022.

[9] Miles J. McGruder, Aniruddhe Pradhan, and Krzysztof Fidkowski. A neural-network based adaptive discontinuous galerkin method for turbulent flow simulations. AIAA SCITECH Forum, 2023.

[10] Romesh Prasad, Malik Hassanaly, and Xiangyu Zhang. Discovery of false data injection attacks on power grid frequency controllers with reinforcement learning. 12 2023.

[11] Waqar Ahmed Khan, Hoi-Lam Ma, Sai-Ho Chung, and Xin Wen. Hierarchical integrated machine learning model for predicting flight departure delays and duration in series. Transportation Research Part C: Emerging Technologies, 129:103225, 2021.

[12] George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. Nature Reviews Physics, 3(6):422–440, Jun 2021.

[13] S. Chakraborty, S. Adhikari, and R. Ganguli. The role of surrogate models in the development of digital twins of dynamic systems. Applied Mathematical Modelling, 90:662–681, 2021.

[14] Bruno Sudret. Global sensitivity analysis using polynomial chaos expansions. Reliability Engineering & System Safety, 93(7):964–979, 2008. Bayesian Networks in Dependability.

[15] Ilias Bilionis, Nicholas Zabaras, Bledar A. Konomi, and Guang Lin. Multi-output separable gaussian process: Towards an efficient, fully bayesian paradigm for uncertainty quantification. Journal of Computational Physics, 241:212–239, 2013.

[16] Souvik Chakraborty and Rajib Chowdhury. Graph-theoretic-approach-assisted gaussian process for nonlinear stochastic dynamic analysis under generalized loading. Journal of Engineering Mechanics, 145(12):04019105, 2019.

[17] Somdatta Goswami, Cosmin Anitescu, Souvik Chakraborty, and Timon Rabczuk. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. Theoretical and Applied Fracture Mechanics, 106:102447, 2020.

[18] Chi Zhang and Abdollah Shafieezadeh. Simulation-free reliability analysis with active learning and physics-informed neural network. Reliability Engineering & System Safety, 226:108716, 2022.

[19] Gene H. Golub, Alan J. Hoffman, and G. W. Stewart. A generalization of the eckart-young-mirsky matrix approximation theorem. Linear Algebra and its Applications, pages 317–327, 1987.

[20] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. Journal of Machine Learning Research, 9(86):2579–2605, 2008.

[21] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders, 2021.

[22] B.G. Galerkin. Rods and plates, series occurring in various questions concerning the elastic equilibrium of rods and plates. Vestnik Inzhenerov i Tekhnikov (Engineers and Technologists Bulletin), 19:897–908, 1915. (in Russian), (English Translation: 63-18925, Clearinghouse Fed. Sci. Tech. Info., 1963).

[23] Cheng Huang and Karthik Duraisamy. Predictive reduced order modeling of chaotic multi-scale problems using adaptively sampled projections. Journal of Computational Physics, 2023.

[24] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org.

[25] R. Abadía-Heredia, M. López-Martín, B. Carro, J.I. Arribas, J.M. Pérez, and S. Le Clainche. A predictive hybrid reduced order model based on proper orthogonal decomposition combined with deep learning architectures. Expert Systems with Applications, 187:115910, 2022.

[26] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 31. Curran Associates, Inc., 2018.

[27] Simo Särkkä. Bayesian Filtering and Smoothing. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013.

[28] Bradley Efron. Bayes' theorem in the 21st century. Science, 340(6137):1177–1178, 2013.

[29] José M Bernardo and Adrian F M Smith. Bayesian theory. Measurement Science and Technology, 12(2):221, feb 2001.

[30] Pawel Cichosz. Data Mining Algorithms: Explained Using R, pages 118–133. 2015.

[31] Alexander Jordan, Fabian Krüger, and Sebastian Lerch. Evaluating probabilistic forecasts with scoringrules. Journal of Statistical Software, 90(12):1–37, 2019.

[32] Julia Slingo and Tim Palmer. Uncertainty in weather and climate prediction. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 369(1956):4751–4767, 2011.

[33] Jethro Browell and Ciaran Gilbert. Probcast: Open-source production, evaluation and visualisation of probabilistic forecasts. In 2020 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS), pages 1–6, 2020.

[34] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. Science, 349(6245):255–260, 2015.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. Nature, 521(7553):436–444, May 2015.

[36] Ziad Obermeyer and Ezekiel J. Emanuel. Predicting the future — big data, machine learning, and clinical medicine. New England Journal of Medicine, 375(13):1216–1219, 2016. PMID: 27682033.

[37] Andrea Dal Pozzolo and Gianluca Bontempi. Adaptive machine learning for credit card fraud detection. Universite Libre de Bruxelles, 2015.

[38] Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer, 2006.

[39] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006.

[40] Andrew D. Selbst, Danah Boyd, Sorelle A. Friedler, Suresh Venkatasubramanian, and Janet Vertesi. Fairness and abstraction in sociotechnical systems. FAT* 2019 - Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency, pages 59–68, January 2019.

[41] Davide Castelvecchi. Can we open the black box of ai? Nature, 538:20–23, 10 2016.

[42] Tomasz Szandała. Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks, pages 203–224. Springer Singapore, Singapore, 2021.

[43] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. Nature Machine Intelligence, 3(3):218–229, Mar 2021.

[44] Yibo Yang and Paris Perdikaris. Conditional deep surrogate models for stochastic, high-dimensional, and multi-fidelity systems. Computational Mechanics, 64(2):417–434, Aug 2019.

[45] Fabian H. Sinz, Xaq Pitkow, Jacob Reimer, Matthias Bethge, and Andreas S. Tolias. Engineering a less artificial intelligence. Neuron, 103(6):967–979, 2019.

[46] M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378:686–707, 2019.

[47] Yibo Yang and Paris Perdikaris. Physics-informed deep generative models, 2018.

[48] Quercus Hernández, Alberto Badías, Francisco Chinesta, and Elías Cueto. Port-metriplectic neural networks: thermodynamics-informed machine learning of complex physical systems. Computational Mechanics, 72(3):553–561, Sep 2023.

[49] Jean-Christophe Loiseau and Steven L. Brunton. Constrained sparse galerkin regression. Journal of Fluid Mechanics, 838:42–67, 2018.

[50] Vishal Srivastava and Karthik Duraisamy. Generalizable physics-constrained modeling using learning and inference assisted by feature-space engineering. Physical Review Fluids, 6(12):124602, 2021.

[51] Karthik Duraisamy, Gianluca Iaccarino, and Heng Xiao. Turbulence modeling in the age of data. Annual Review of Fluid Mechanics, 51(1):357–377, 2019.

[52] C. E. Shannon. A mathematical theory of communication. The Bell System Technical Journal, 27(3):379–423, 1948.

[53] Thomas M. Cover and Joy A. Thomas. Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing). Wiley-Interscience, USA, 2006.

[54] Thomas Cover and Joy Thomas. Rate Distortion Theory. John Wiley & Sons, Ltd, 2001.

[55] David J. C. MacKay. Information Theory, Inference, and Learning Algorithms. Copyright Cambridge University Press, 2003.

[56] Khalid Sayood. Introduction to data compression (third edition). The Morgan Kaufmann Series in Multimedia Information and Systems, pages xvii–xxii, 2006.

[57] Toby Berger. Rate Distortion Theory. A Mathematical Basis for Data Compression. Prentice-Hall, Englewood Cliffs, 1971.

[58] Jerry Gibson. Information Theory and Rate Distortion Theory for Communications and Compression. 2013.

[59] Eric Lei, Hamed Hassani, and Shirin Saeedi Bidokhti. Neural estimation of the rate-distortion function with applications to operational source coding. IEEE Journal on Selected Areas in Information Theory, 3(4):674–686, 2022.

[60] Lingyi Chen, Shitong Wu, Wenhao Ye, Huihui Wu, Wenyi Zhang, Hao Wu, and Bo Bai. A constrained ba algorithm for rate-distortion and distortion-rate functions. arXiv, 2024.

[61] F. Dupuis, W. Yu, and F.M.J. Willems. Blahut-arimoto algorithms for computing channel capacity and rate-distortion with side information. In International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings., pages 179–, 2004.

[62] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Francis Bach and David Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 1530–1538, Lille, France, 07–09 Jul 2015. PMLR.

[63] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. Journal of the American Statistical Association, 112(518):859–877, apr 2017.

[64] Fangjian Guo, Xiangyu Wang, Kai Fan, Tamara Broderick, and David B. Dunson. Boosting variational inference. arXiv, 2017.

[65] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. Journal of Machine Learning Research, 14(40):1303–1347, 2013.

[66] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. jcp, 21(6):1087–1092, June 1953.

[67] Christian P. Robert and George Casella. Monte Carlo Statistical Methods (Springer Texts in Statistics). Springer-Verlag, Berlin, Heidelberg, 2005.

[68] Christian Robert and George Casella. A short history of markov chain monte carlo: Subjective recollections from incomplete data. Statistical Science, 26(1), feb 2011.

[69] Alp Kucukelbir David M. Blei and Jon D. McAuliffe. Variational inference: A review for statisticians. Journal of the American Statistical Association, 112(518):859–877, 2017.

[70] J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. The Computer Journal, 7(4):308–313, 01 1965.

[71] I. Kobyzev, S. D. Prince, and M. A. Brubaker. Normalizing flows: An introduction and review of current methods. IEEE Transactions on Pattern Analysis & Machine Intelligence, 43(11):3964–3979, nov 2021.

[72] Diederik Kingma and Max Welling. Auto-Encoding Variational Bayes. International Conference on Learning Representations, 12 2013.

[73] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In Eric P. Xing and Tony Jebara, editors, Proceedings of the 31st International Conference on Machine Learning, volume 32 of Proceedings of Machine Learning Research, pages 1278–1286, Bejing, China, 22–24 Jun 2014. PMLR.

[74] Christian Jacobsen and Karthik Duraisamy. Disentangling generative factors of physical fields using variational autoencoders. Frontiers in Physics, 10, 2022.

[75] Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, volume 15 of Proceedings of Machine Learning Research, pages 29–37, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[76] Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In Francis Bach and David Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 881–889, Lille, France, 07–09 Jul 2015. PMLR.

[77] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In Maria Florina Balcan and Kilian Q. Weinberger, editors, Proceedings of The 33rd International Conference on Machine Learning, volume 48 of Proceedings of Machine Learning Research, pages 1747–1756, New York, New York, USA, 20–22 Jun 2016. PMLR.

[78] Yann Lecun, Sumit Chopra, Raia Hadsell, Marc Aurelio Ranzato, and Fu Jie Huang. A tutorial on energy-based learning. MIT Press, Cambridge, 2006.

[79] Yang Song and Diederik P. Kingma. How to train your energy-based models. arXiv, 2021.

[80] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. arXiv, 2015.

[81] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In International Conference on Learning Representations, 2017.

[82] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 27, 2014.

[83] Kacper Chwialkowski, Heiko Strathmann, and Arthur Gretton. A kernel test of goodness of fit. In Maria Florina Balcan and Kilian Q. Weinberger, editors, Proceedings of The 33rd International Conference on Machine Learning, volume 48 of Proceedings of Machine Learning Research, pages 2606–2615, New York, New York, USA, 20–22 Jun 2016. PMLR.

[84] I. Higgins, L. Matthey, A. Pal, Christopher P. Burgess, Xavier Glorot, M. Botvinick, S. Mohamed, and Alexander Lerchner. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. International Conference on Learning Representations, 2017.

[85] Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, and P. V. D. Smagt. Learning Hierarchical Priors in VAEs. Conference on Neural Information Processing Systems, 2019.

[86] Hyunjik Kim and A. Mnih. Disentangling by Factorising. International Conference on Machine Learning, 2018.

[87] Shengjia Zhao, Jiaming Song, and S. Ermon. InfoVAE: Information Maximizing Variational Autoencoders. arXiv, 1706.02262, 2017.

[88] Y. Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35:1798–1828, 08 2013.

[89] Daniel J. Tait and Theodoros Damoulas. Variational Autoencoding of PDE Inverse Problems. arXiv, 2006.15641, 2020.

[90] Peter Y. Lu, Samuel Kim, and Marin Soljačić. Extracting Interpretable Physical Parameters from Spatiotemporal Systems Using Unsupervised Learning. Physical Review X, Sep 2020.

[91] Ryan Lopez and Paul Atzberger. Variational Autoencoders for Learning Nonlinear Dynamics of Physical Systems. arXiv, 2012.03448, 12 2020.

[92] Jérémie Donà, Jean-Yves Franceschi, sylvain lamprier, and patrick gallinari. PDE-Driven Spatiotemporal Disentanglement. International Conference on Learning Representations, 2021.

[93] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial Autoencoders. International Conference on Learning Representations, 2016.

[94] Stephen G. Odaibo. Tutorial: Deriving the Standard Variational Autoencoder (VAE) Loss Function. arXiv, 1907.08956, 2019.

[95] Ronald Yu. A Tutorial on VAEs: From Bayes' Rule to Lossless Compression. arXiv, 2006.10273, 2020.

[96] Karthik Duraisamy. Variational Encoders and Autoencoders : Information-theoretic Inference and Closed-form Solutions. arXiv, 2101.11428, 2021.

[97] Michal Rolinek, Dominik Zietlow, and Georg Martius. Variational Autoencoders Pursue PCA Directions (by Accident). arXiv, 1812.06775, 06 2019.

[98] James Lucas, G. Tucker, Roger B. Grosse, and Mohammad Norouzi. Understanding Posterior Collapse in Generative Latent Variable Models. International Conference on Learning Representations, 2019.

[99] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In International Conference on Learning Representations, 2021.

[100] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 6840–6851, 2020.

[101] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32, 2019.

[102] Yilun Xu, Ziming Liu, Max Tegmark, and Tommi S. Jaakkola. Poisson flow generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, Advances in Neural Information Processing Systems, 2022.

[103] Yilun Xu, Ziming Liu, Yonglong Tian, Shangyuan Tong, Max Tegmark, and Tommi Jaakkola. Pfgm++: Unlocking the potential of physics-inspired generative models. arXiv, 2023.

[104] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. arXiv, 2023.

[105] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, Advances in Neural Information Processing Systems, 2021.

[106] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. Diffusion-LM improves controllable text generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, Advances in Neural Information Processing Systems, 2022.

[107] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. In The Eleventh International Conference on Learning Representations, 2023.

[108] Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. ACL: Annual Meeting of the Association for Computational Linguistics.

[109] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(4):4713–4726, 2023.

[110] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 11451–11461, 2022.

[111] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In ACM SIGGRAPH 2022 Conference Proceedings, SIGGRAPH '22, New York, NY, USA, 2022. Association for Computing Machinery.

[112] J. Xu, S. Liu, A. Vahdat, W. Byeon, X. Wang, and S. De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 2955–2966, Los Alamitos, CA, USA, jun 2023. IEEE Computer Society.

[113] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10684–10695, 2022.

[114] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. arXiv, 2022.

[115] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In International Conference on Machine Learning, 2022.

[116] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. arXiv preprint arxiv:2208.12242, 2022.

[117] Maneesh Agrawala Lvmin Zhang, Anyi Rao. Adding conditional control to text-to-image diffusion models. In International Conference on Computer Vision, 2023.

[118] Jiaqi Guan, Wesley Wei Qian, Xingang Peng, Yufeng Su, Jian Peng, and Jianzhu Ma. 3d equivariant diffusion for target-aware molecule generation and affinity prediction. In The Eleventh International Conference on Learning Representations, 2023.

[119] Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, and Jianzhu Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, Advances in Neural Information Processing Systems, 2022.

[120] Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. In International Conference on Learning Representations, 2022.

[121] Hyungjin Chung and Jong Chul Ye. Score-based diffusion models for accelerated mri. Medical Image Analysis, 80:102479, 2022.

[122] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[123] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. ACM Comput. Surv., 56(4), nov 2023.

[124] G. Parisi. Correlation functions and computer simulations. Nuclear Physics B, 180(3):378–384, 1981.

[125] Ulf Grenander and Michael I. Miller. Representations of knowledge in complex systems. Journal of the Royal Statistical Society. Series B (Methodological), 56(4):549–603, 1994.

[126] Yang Song. Estimating gradients of the data distribution. https://yang-song.net/blog/2021/score/, 05 2021.

[127] Brian D.O. Anderson. Reverse-time diffusion equation models. Stochastic Processes and their Applications, 12(3):313–326, 1982.

[128] Pascal Vincent. A connection between score matching and denoising autoencoders. Neural Computation, 23(7):1661–1674, 2011.

[129] Joaquim R. R. A. Martins and Andrew Ning. Engineering Design Optimization. Cambridge University Press, 2021.

[130] Christian Jacobsen, Jiayuan Dong, Mehdi Khalloufi, Xun Huan, Karthik Duraisamy, Maryam Akram, and Wanjiao Liu. Enhancing dynamical system modeling through interpretable machine learning augmentations: A case study in cathodic electrophoretic deposition. arXiv, 2024.

[131] Ricky T. Q. Chen, Brandon Amos, and Maximilian Nickel. Learning neural event functions for ordinary differential equations. In International Conference on Learning Representations, 2021.

[132] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In International Conference on Learning Representations, 2021.

[133] Defang Li, Min Zhang, Weifu Chen, and Guocan Feng. Facial Attribute Editing by Latent Space Adversarial Variational Autoencoders. International Conference on Pattern Recognition, pages 1337–1342, 2018.

[134] Tao Wang, Junzhe Liu, Cong Jin, Jianguang Li, and Shihao Ma. An Intelligent Music Generation Based on Variational Autoencoder. International Conference on Culture-oriented Science Technology, pages 394–398, 2020.

[135] Alexander Amini, Wilko Schwarting, Guy Rosman, Brandon Araki, Sertac Karaman, and Daniela Rus. Variational Autoencoder for End-to-End Control of Autonomous Driving with Novelty Detection and Training De-biasing. IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 568–575, 10 2018.

[136] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Y. Bengio. Generative Adversarial Networks. Advances in Neural Information Processing Systems, 3, 2014.

[137] Abdon Atangana. Chapter 2 - principle of groundwater flow. In Abdon Atangana, editor, Fractional Operators with Constant and Variable Order with Application to Geo-Hydrology, pages 15–47. Academic Press, 2018.

[138] David J. Griffiths. Introduction to Electrodynamics. Cambridge University Press, 5 edition, 2023.

[139] Suresh Kumar Govindarajan. An overview on extension and limitations of macroscopic darcy's law for a single and multi-phase fluid flow through a porous medium. International Journal of Mining Science, 5:1–21, 12 2019.

[140] Jingyi. Leng, Xiaobo. Lin, and Linlin. Wang. Effects of osmosis on darcy flow in shales. Energy & Fuels, 35(6):4874–4884, 2021.

[141] Yinhao Zhu and Nicholas Zabaras. Bayesian Deep Convolutional Encoder–Decoder Networks for Surrogate Modeling and Uncertainty Quantification. Journal of Computational Physics, 366:415–447, Aug 2018.

[142] Christopher P. Burgess, I. Higgins, A. Pal, L. Matthey, Nicholas Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding Disentangling in $\beta$-VAE (2018). arXiv, 1804.03599, 2018.

[143] Cian Eastwood and Christopher K. I. Williams. A Framework for the Quantitative Evaluation of Disentangled Representations. International Conference on Learning Representations, 2018.

[144] Francesco Locatello, Stefan Bauer, Mario Lucic, S. Gelly, B. Schölkopf, and Olivier Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. arXiv, 1811.12359, 2019.

[145] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, A. Çelikyilmaz, and L. Carin. Cyclical Annealing Schedule: A Simple Approach to Mitigating KL Vanishing. North American chapter of the Association for Computational Linguistics, 2019.

[146] Vincent Sitzmann, Julien NP Martel, Alexander W Bergman, David B Lindell, and Gordon Wetzstein. Implicit Neural Representations with Periodic Activation Functions. arXiv, 2006.09661, 2020.

[147] Qing Wang, Sanjeev R. Kulkarni, and Sergio Verdu. Divergence Estimation for Multidimensional Densities Via k-Nearest-Neighbor Distances. IEEE Transactions on Information Theory, 55(5):2392–2405, 2009.

[148] Percy E Pierce. The physical chemistry of the cathodic electrodeposition process. J. Coat. Technol., 53(672):52–67, 1981.

[149] Saeed Rastegar, Zahra Ranjbar, and Ghasem Fakhrpour. A generalized model for the film growth during the constant voltage electro-deposition of resin dispersions. Colloids and Surfaces A: Physicochemical and Engineering Aspects, 317(1):17–22, 2008.

[150] Donald W. Boyd and Robert R. Zwack. Predicting electrocoat efficiency: a simple data-based model for predicting electrocoat usage. Progress in Organic Coatings, 27(1):25–32, 1996. Proceedings of the 20th International Conference in Organic Coatings Science and Technology.

[151] Vesna B Mišković-Stanković. The mechanism of cathodic electrodeposition of epoxy coatings and the corrosion behaviour of the electrodeposited coatings. Journal of the Serbian Chemical Society, 67(5):305–324, 2002.

[152] Kevin Ellwood, Janice L. Tardiff, Leonard J. Gray, Patrick Gaffney, Jacob Braslaw, Knut Moldekleiv, and Arne Halvorsen. Development of a full vehicle electrocoat paint simulation tool. SAE International Journal of Materials and Manufacturing, 2(1):234–240, 2009.

[153] Fritz Beck. Fundamental aspects of electrodeposition of paint. Progress in Organic Coatings, 4(1):1–60, 1976.

[154] Henry J.S. Sand Ph.D. Iii. on the concentration at the electrodes in a solution, with special reference to the liberation of hydrogen by electrolysis of a mixture of copper sulphate and sulphuric acid. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 1(1):45–79, 1901.

[155] Tyler Marlar, Andrew Trainor, Wanjiao Liu, Kevin Ellwood, Brian Okerberg, Fardin Padash, and John N. Harb. Effect of convection on initial deposition during electrocoating of galvanized steel. Journal of The Electrochemical Society, 167(10):103502, jun 2020.

[156] Ford Motor Company Wanjiao Liu, Kevin Ellwood. Simulation of the full vehicle electrodeposition for paint virtual manufacturing. SAE World Congress, 2017.

[157] Minh-Ngoc Tran, Trong-Nghia Nguyen, and Viet-Hung Dao. A practical tutorial on variational bayes. arXiv, 2021.

[158] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems 32, pages 8024–8035. Curran Associates, Inc., 2019.

[159] J. Guillaume, J. Jakeman, S. Marsili-Libelli, Michael Asher, P. Brunner, B. Croke, M. C. Hill, A. Jakeman, K. Keesman, S. Razavi, and J. D. Stigter. Introductory overview of identifiability analysis: A guide to evaluating whether you have the right type of data for your modeling purpose. Environ. Model. Softw., 119:418–432, 2019.

[160] Hisashi NAGAI, Yuki ONISHI, and Kenji AMAYA. Exact paint resistance and deposition models for accurate numerical electrodeposition simulation. TRANSACTIONS OF THE JAPAN SOCIETY OF MECHANICAL ENGINEERS Series A, 78(794):1446–1461, 2012.

[161] Edmund J.F. Dickinson and Andrew J. Wain. The butler-volmer equation in electrochemical theory: Origins, value, and practical application. Journal of Electroanalytical Chemistry, 872:114145, 2020. Dr. Richard Compton 65th birthday Special issue.

[162] Christian Jacobsen, Ivan Zanardi, Sahil Bhola, Karthik Duraisamy, and Marco Panesi. Information theoretic clustering for coarse-grained modeling of non-equilibrium gas dynamics. Journal of Computational Physics, page 112977, 2024.

[163] Walter Guido Vincenti and Charles H. Kruger. Introduction to physical gas dynamics. 1965.

[164] Chul Hi Park. Nonequilibrium hypersonic aerothermodynamics. 1989.

[165] J. Reece Roth. Industrial plasma engineering: Volume 2 - applications to nonthermal plasma processing (1st ed.). Routledge., 2001.

[166] Abhilash Harpale, Marco Panesi, and Huck Beng Chew. Communication: Surface-to-bulk diffusion of isolated versus interacting C atoms in Ni(111) and Cu(111) substrates: A first principle investigation. The Journal of Chemical Physics, 142(6):061101, 02 2015.

[167] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. SIAM review, 57(4):483–531, 2015.

[168] Cheng Huang, Christopher R Wentland, Karthik Duraisamy, and Charles Merkle. Model reduction for multi-scale transport problems using model-form preserving least-squares projections with variable transformation. Journal of Computational Physics, 448:110742, 2022.

[169] Yen Liu, Marcel Vinokur, Macro Panesi, and Thierry Magin. A multi-group maximum entropy model for thermo-chemical non-equilibrium. In 10th AIAA/ASME Joint Thermophysics and Heat Transfer Conference, page 4332, 2010.

[170] Yen Liu, Marco Panesi, Amal Sahai, and Marcel Vinokur. General multi-group macroscopic modeling for thermo-chemical non-equilibrium gas mixtures. The Journal of Chemical Physics, 142(13):134109, 2015.

[171] I. Armenise and E.V. Kustova. State-to-state models for co2 molecules: From the theory to an application to hypersonic boundary layers. Chemical Physics, 415:269–281, 2013.

[172] Hai P. Le, Ann R. Karagozian, and Jean-Luc Cambier. Complexity reduction of collisional-radiative kinetics for atomic plasma. Physics of Plasmas, 20(12):123304, 12 2013.

[173] A. Sahai, B. Lopez, C. O. Johnston, and M. Panesi. Adaptive coarse graining method for energy transfer and dissociation kinetics of polyatomic species. The Journal of Chemical Physics, 147(5):054107, 08 2017.

[174] S. Venturi, M. P. Sharma, B. Lopez, and M. Panesi. Data-inspired and physics-driven model reduction for dissociation: Application to the o2 + o system. The Journal of Physical Chemistry A, 124(41):8359–8372, 2020. PMID: 32886505.

[175] A. Munafò, Y. Liu, and M. Panesi. Modeling of dissociation and energy transfer in shock-heated nitrogen flows. Physics of Fluids, 27(12):127101, 12 2015.

[176] Robert Tarjan. Depth-first search and linear graph algorithms. SIAM Journal on Computing, 1(2):146–160, 1972.

[177] John Hopcroft and Robert Tarjan. Algorithm 447: Efficient algorithms for graph manipulation. Commun. ACM, 16(6):372–378, jun 1973.

[178] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, Advances in Neural Information Processing Systems, volume 14. MIT Press, 2001.

[179] Markos A. Katsoulakis and Pedro Vilanova. Data-driven, variational model reduction of high-dimensional reaction networks. Journal of Computational Physics, 401:108997, 2020.

[180] Sebastian Kaltenbach and Phaedon-Stelios Koutsourelakis. Incorporating physical constraints in a deep probabilistic machine learning framework for coarse-graining dynamical systems. Journal of Computational Physics, 419:109673, 2020.

[181] Karthik Duraisamy. Perspectives on machine learning-augmented reynolds-averaged and large eddy simulation models of turbulence. Physical Review Fluids, 6(5):050504, 2021.

[182] Andrey Starikovskiy. Physics and chemistry of plasma-assisted combustion. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 373(2048):20150074, 2015.

[183] Peter A. Gnoffo. Planetary-entry gas dynamics. Annual Review of Fluid Mechanics, 31(1):459–494, 1999.

[184] Sergey T. Surzhikov. Radiative-Collisional Models in Non-Equilibrium Aerothermodynamics of Entry Probes. Journal of Heat Transfer, 134(3):031002, 01 2012.

[185] Marco Panesi, Thierry Magin, Anne Bourdon, Arnaud Bultel, and O. Chazot. Fire ii flight experiment analysis by means of a collisional-radiative model. Journal of Thermophysics and Heat Transfer, 23(2):236–248, 2009.

[186] M. Panesi, T. E. Magin, A. Bourdon, A. Bultel, and O. Chazot. Electronic excitation of atoms and molecules for the fire ii flight experiment. Journal of Thermophysics and Heat Transfer, 25(3):361–374, 2011.

[187] Annemie Bogaerts, Erik Neyts, Renaat Gijbels, and Joost van der Mullen. Gas discharge plasmas and their applications. Spectrochimica Acta Part B: Atomic Spectroscopy, 57(4):609–658, 2002.

[188] Gianpiero Colonna, Iole Armenise, Domenico Bruno, and Mario Capitelli. Reduction of state-to-state kinetics to macroscopic models in hypersonic flows. Journal of Thermophysics and Heat Transfer, 20(3):477–486, 2006.

[189] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014.

[190] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In International Conference on Learning Representations, 2017.

[191] David W. Schwenke. Calculations of rate constants for the three-body recombination of h2 in the presence of h2. The Journal of Chemical Physics, 89:2076–2091, 8 1988.

[192] Galina Chaban, Richard Jaffe, David W. Schwenke, and Winifred Huo. Dissociation cross sections and rate coefficients for nitrogen from accurate theoretical calculations. 46th AIAA Aerospace Sciences Meeting and Exhibit, 2008.

[193] Alessandro Munafò and Marco Panesi. Plato: a high-fidelity tool for multi-component plasmas. American Institute of Aeronautics and Astronautics (AIAA), 6 2023.

[194] Alessandro Munafò, Andrea Alberti, Carlos Pantano, Jonathan B. Freund, and Marco Panesi. A computational model for nanosecond pulse laser-plasma interactions. Journal of Computational Physics, 406:109190, 4 2020.

[195] Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 240–250, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[196] Gefan Yang and Stefan Sommer. A denoising diffusion model for fluid field prediction. arXiv, 2023.

[197] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, pages 234–241, Cham, 2015. Springer International Publishing.

[198] Dule Shu, Zijie Li, and Amir Barati Farimani. A physics-informed diffusion model for high-fidelity flow field reconstruction. Journal of Computational Physics, 478:111972, 2023.

[199] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, Advances in Neural Information Processing Systems, 2022.

[200] P.E. Kloeden and E. Platen. Numerical Solution of Stochastic Differential Equations. Stochastic Modelling and Applied Probability. Springer Berlin, Heidelberg, 2011.

[201] Saifon Chaturantabut and Danny C Sorensen. Nonlinear model reduction via discrete empirical interpolation. SIAM Journal on Scientific Computing, 32(5):2737–2764, 2010.

[202] Christopher R Wentland, Karthik Duraisamy, and Cheng Huang. Scalable projection-based reduced-order models for large multiscale fluid systems. AIAA Journal, 61(10):4499–4523, 2023.

[203] William Briggs, Van Henson, and Steve McCormick. A Multigrid Tutorial, 2nd Edition. 01 2000.

[204] Ivan Zanardi, Simone Venturi, and Marco Panesi. Towards Efficient Simulations of Non-Equilibrium Chemistry in Hypersonic Flows: Application of Neural Operators in Multidimensional CFD Simulations.