Solutions to Bayesian Inverse Problems using Variational AutoEncoders

Submitted in partial fulfillment of the
requirements for the degree

of

Master of Technology

by

Ninad Chavan
142202021

Under the guidance of

Dr. Arijit Hazra

IIT PALAKKAD

DATA SCIENCE

INDIAN INSTITUTE OF TECHNOLOGY PALAKKAD

IIT PALAKKAD

Department of Data Science
Indian Institute of Technology, Palakkad
Kerala - 678623

CERTIFICATE

This is to certify that we have examined the thesis entitled **Solutions to Bayesian Inverse Problems using Variational AutoEncoders**, submitted by **Ninad Chavan**(Roll Number: *142202021*) a postgraduate student of **Department of Data Science** in partial fulfillment for the award of degree of Master of Technology. We hereby accord our approval of it as a study carried out and presented in a manner required for its acceptance in partial fulfillment for the Post Graduate Degree for which it has been submitted. The thesis has fulfilled all the requirements as per the regulations of the Institute and has reached the standard needed for submission.

Dr. Arijit Hazra
SERB Ramanujan Fellow
Indian Institute of Technology,
Palakkad

Place: Palakkad
Date: 15 May 2024

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my deepest gratitude to my supervisor and mentor, **Dr. Arijit Hazra**. His support and encouragement have been invaluable throughout the journey. I extend my gratitude to all the individuals who have directly and indirectly supported me throughout the Project.

Ninad Chavan
M.Tech Data Science
Department of Data Science
Indian Institute of Technology, Palakkad
May 2024

ABSTRACT

Inverse problems are integral to a broad range of applications across different scientific and engineering fields. We are particularly solving parameter estimation inverse problem. Bayesian methods is one of the method to solve inverse problem which gives probabilistic distribution of parameters. But this methods are computationally expensive. Variational autoencoders due to its probabilistic encoding techniques can be used for solving parameter estimation inverse problem. In this project, we have covered different Variational Autoencoders architectures to solve parameter estimation inverse problem.

Keywords: Variational Inference (VI), Variational Autoencoder (VAE), Conditional Variational Autoencoder (C-VAE), Uncertainty Quantification Variational Autoencoder (UQ-VAE), Markov Chain Monte Carlo(MCMC), Maximum Likelihood Estimate (MLE), Maximum A Posteriori (MAP), Parameter of Interest(PoI), Kullback-Leibler Divergence (KLD), Jensen-Shannon Divergence (JSD)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Inverse Problem

In mathematical modeling and engineering, the forward problem typically involves using a known set of input variables and a defined operator or function to predict an output. In this type of problem the inputs and the mathematical relations are well-defined, leading to a specific output. Conversely, the inverse problem is the process of calculating from the set of observations or outputs the causal factors or input variables that produced them[1].In general terms, given the effects finding the cause. e.g from the observations of temperature measurements, finding the thermal conductivity of the material.

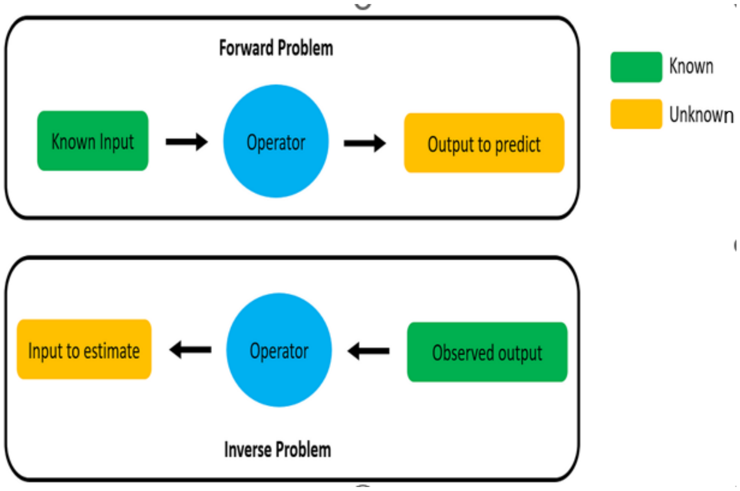

Figure 1.1: Inverse problem definition

Inverse problems are common in many fields including physics, engineering, medical imaging, geophysics, etc. Here are some of the examples in different domains. Magnetic Resonance Imaging (MRI) in an application in medical imaging. In MRI, constructing images of the interior of the body from measurements of the response of atomic nuclei to magnetic fields is an example of inverse problem. In geophysics, determining the density distribution inside the Earth from measurements of the gravitational field is an example of inverse problem. Let x_{obs} be the observed data or measurements and k be the parameter to be estimated. As there will be errors in measurements or observations, let e be the error or noise in observations and F be parameter to observations function. Then mathematically inverse problem can be represented as[1]:

$$x_{\text{obs}} = F(k) + e \tag{1.1}$$

Inverse problems are generally ill-posed. A problem is ill-posed [2] if it does not satisfy one of the three conditions suggested by Jacques Hadamard of a well-posed problem which are existence, uniqueness and stability. If there exists a solution, existence condition is satisfied. If the solution is unique, uniqueness condition is satisfied. If the solution depends continuously on initial conditions and small changes in input data result in small changes in output then stability condition is satisfied. Inverse problems does not satisfies one or more of these conditions.

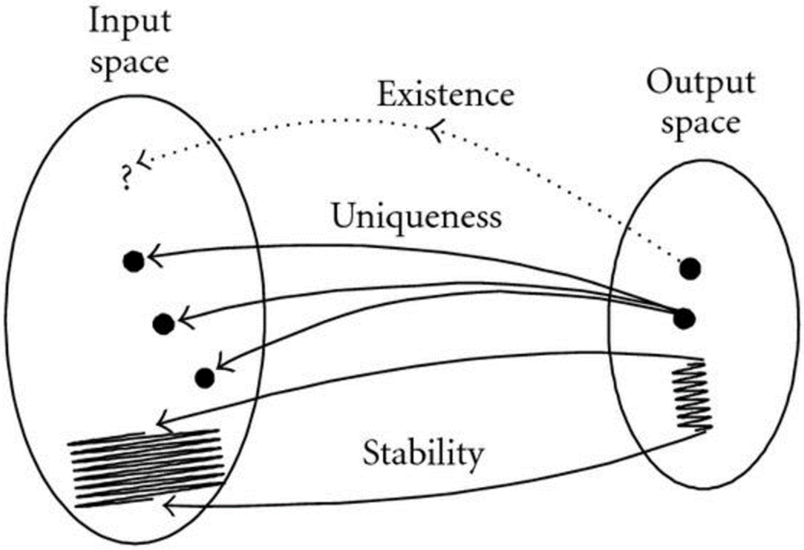

Figure 1.2: Inverse problems are ill-posed

1.2 Methods to solve Inverse Problems

Fundamentally there are two methods for solving inverse problems:
(a)Deterministic or Regularization based methods : Least Squares, Tikhonov Regularization are examples of this methods. Least squares methods are used for estimating the coefficients of a model by minimizing the sum of the squared differences (residuals) between observed and predicted values. Regularization based methods in inverse problems prevent ill-posedness by adding a penalty term to the objective function, promoting stability and ensuring a well-posed solution. These techniques constrain model complexity, yielding robust and generalizable solutions. The disadvantage of these methods is it provides a single or point estimate of the parameters and does not account for the variability in the results. Let x be the observed data and k be the parameter to be estimated. Mathematical representation of regularization based methods[3] :

$$\min_{k} \|x - F(k)\|^2 + R(k) \tag{1.2}$$

(b)Bayesian or Statistical methods : Provides a complete posterior distribution which helps to quantify uncertainty over parameters, rather than just point estimates. As based on Bayes' theorem, they allows us to incorporate prior information, which further helps to improve parameter estimation. Markov Chain Monte Carlo, Laplace Approximation are examples of this methods. While effective in quantifying uncertainty or giving the complete posterior distribution of the parameter, these methods are computationally expensive. Bayes theorem can be mathematically represented as [4]:

$$p(k \mid x) = \frac{p(x \mid k)p(k)}{\int p(x \mid k')p(k')\,dk'} \tag{1.3}$$

where $p(k|x)$ is the posterior probability, $p(x|k)$ is the likelihood of observing x given k, $p(k)$ is the prior probability of parameter k. The denominator is the evidence or marginal likelihood, integrating over all possible values of k.

Consider an example of curve fitting using various bayesian inference methods.The blue points are the data points and a curve needs to be fitted. For this a we need to get a polynomial equation. In this case, the co-efficients of the polynomial eqautions are the parameters to be estimated. **Maximum Likelihood Estimation (MLE)** finds the parameter values that maximize the likelihood of observed data, relying solely on the data without incorporating prior information. Using this values of the parameters

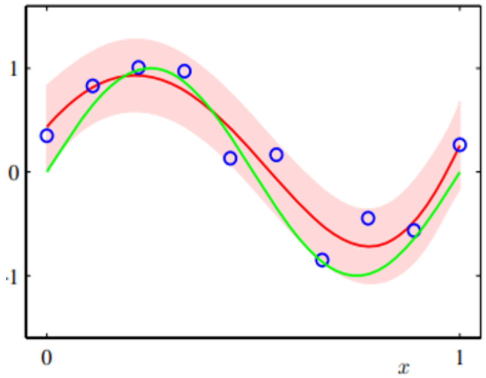

Figure 1.3: Curve Fitting using various methods
[5]

in the polynomial equation, we get a red curve which fits the datapoints. **Maximum A Posteriori (MAP)** estimation combines the likelihood of the data with prior information about the parameters, finding values of parameters that maximize the posterior distribution. This method enhances MLE by incorporating prior knowledge through a prior distribution. Using this values of parameters in the polynomial equation, we get a green curve which fits the datapoints. MLE is equivalent to MAP when the prior is uniform, as the prior has no influence, making the estimates identical. **Markov Chain Monte Carlo (MCMC)** is a technique used in bayesian statistics to understand complex probability distributions. It works by running a simulation that follows certain rules (Markov Chain) to create samples. These samples help us learn about important features of the distribution, such as mean values and variance, and gives the distribution estimates of the parameters [6].

With the advent of neural networks and their ability to approximate complex functions, we can utilize them to solve inverse problems. Also called **Deep Learning-based methods**, these methods can be both deterministic and probabilistic [7]. Deterministic deep learning methods provide a single, fixed output for a given input, such as using neural networks as surrogate models to approximate complex mappings efficiently. Probabilistic deep learning methods incorporate uncertainty into their predictions, offering a distribution of possible solutions rather than a single estimate. Deep learning methods trades off real-time estimation with offline training means it undergo intensive, time consuming training beforehand. Once trained, they can quickly provide real-time estimations, making them efficient for solving inverse problems.

Chapter 2

Motivation

As inverse problems being ill-posed in nature and having characteristics such as multiple solutions, stability or sensitivity issues such as small changes in outputs gives large changes in predictions of causal parameters. Hence, bayesian or statistical methods should be preferred as it gives the full distribution estimate of the parameters. Also other advantages of bayesian method includes incorporation of prior knowledge, which can allow in estimating the parameters accurately.But as seen earlier from equation 1.3, calculating the posterior distribution $p(k|x)$ is computationally expensive due to the calculation of evidence. Variational Inference (VI) lets us approximate a high-dimensional bayesian posterior with a simpler variational distribution by solving an optimization problem. It assumes a distribution say $q'(k|x)$ and approximates it to the target posterior $p(k|x)$ which was computationally expensive to calculate. Variational Autoencoders are a class of deep generative models that use principles of variational inference to learn the latent representations of input data. VAEs offer the advantages typically associated with Bayesian methods such as probabilistic distribution of parameters, while avoiding the disadvantages of bayesian methods such as high computational cost and intractable posterior distribution. Unlike Bayesian methods, such as those requiring re-running Markov Chain Monte Carlo (MCMC) sampling for each analysis and are computationally expensive, VAEs have a one-time computational cost for model training. They leverage the power of deep learning, optimized for efficient training on large datasets. Additionally, VAEs utilize the Variational Inference technique, which approximates complex posterior distributions with simpler, parameterized ones, making the inference process more tractable. We will be exploring the utilization of Variational Autoencoders (VAEs) as an alternative to Bayesian methods for solving inverse problems.

Chapter 3

Methodology

3.1 Variational Autoencoder

3.1.1 Variational Autoencoder Architecture

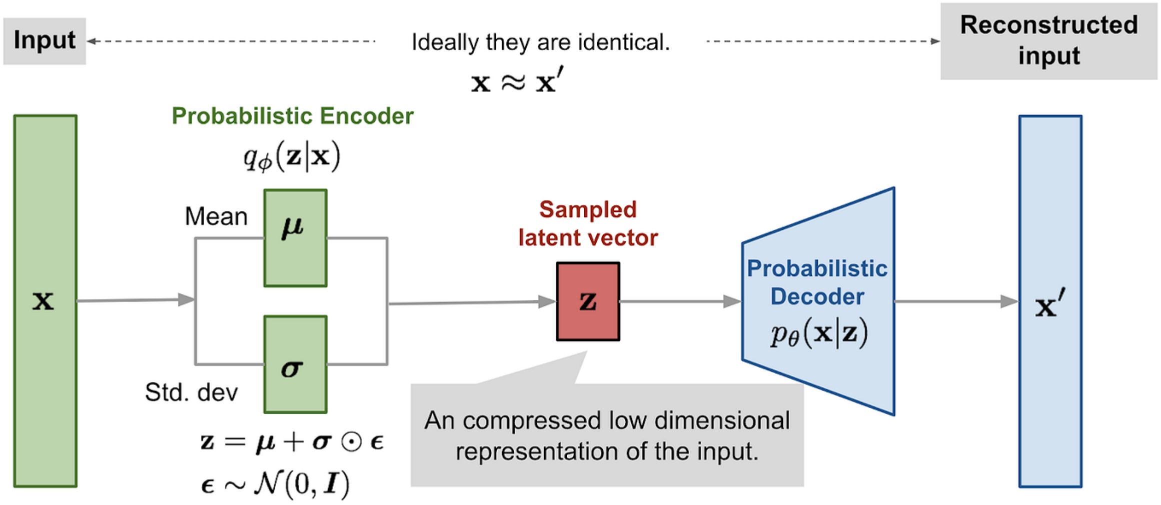
Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{x}$
Sampled latent vector
$\mathbf{z}$
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
An compressed low dimensional representation of the input.

Figure 3.1: Variational Autoencoder Architecture

A Variational Autoencoder (VAE) is a generative model that combines an encoder network to map data into a probabilistic latent space and a decoder network to generate new data samples from this space. Its components include the encoder, which computes the latent space parameters, and the decoder, which reconstructs data from sampled latent points, facilitating tasks like data generation and representation learning.

Encoder Architecture
The architecture of a Variational Autoencoder (VAE) encoder typically consists of neural network layers that map input data to the parameters of the latent space probability distribution usually a Gaussian distribution. The input layer takes the input data and its shape is equal to the number of features in the input data. The hidden layers in encoder reduce the dimensionality of the input data while preserving essential information. The final hidden layer provides the latent space parameters, such as the mean and log-variance. The reparameterization trick in Variational Autoencoders (VAEs) allows differentiable sampling from a Gaussian latent space distribution. Instead of directly sampling z from $\mathcal{N}(\mu, \sigma^2)$, it samples from a standard Gaussian $\mathcal{N}(0, 1)$ and transforms it using [8]:

$$z = \mu + \sigma \cdot \epsilon \tag{3.1}$$

where z is the sampled latent variable, μ is the mean, σ is the standard deviation, and ϵ is sampled from $\mathcal{N}(0, 1)$. This ensures gradients flow through the sampling, enabling VAE training while modeling complex data distributions.

Decoder Architecture
The decoder architecture in VAE is responsible for generating data samples from the latent space representations. It mirrors the encoder's architecture but in reverse. The decoder starts with an input layer that receives samples from the latent space. These samples are denoted as z and known as latent inputs. This layer dimension are same as latent space dimensions. The decoder's hidden layers take samples from the latent space and map them to the space of the reconstructed data. The output layer of vae has the same dimensions as input layer of encoder and is designed to recontruct the input data. The VAE architectures we are using have three hidden layers having eight, five and three neurons. The activation function we have choosen is tanh.

3.1.2 Variational Autoencoder Loss Function

In the framework of Variational Autoencoders (VAEs), the primary objective is to maximize the likelihood of the observed data under the model parameters. However, direct computation of the likelihood is intractable due to the integration over the latent variables and computationally expensive. Therefore, VAEs employ a variational inference approach to approximate this posterior with a simpler distribution.

Below are the detailed steps involved in deriving the loss function of a VAE, which is based on the maximization of the Evidence Lower Bound (ELBO).

Step 1: The initial log probability of the observed data x under the model parameters θ is expressed as [8]:

$$\log p_\theta(x) = \log \mathbb{E}_{z\sim q'(z|x)}[p_\theta(x)] \tag{3.2}$$

where z is the latent variable and $q'(z|x)$ is the variational distribution.

Step 2: Introducing the variational distribution $q'(z|x)$ and applying Bayes' rule, the log probability can be rewritten as [8]:

$$\log p_\theta(x) = \mathbb{E}_Z \left[\log \frac{p_\theta(x|z)p(z)}{p(z|x)}\right] \tag{3.3}$$

where $p_\theta(x|z)$ is the likelihood of the data given the latent variables, and $p(z)$ is the prior over the latent variables.

Step 3: The above expression can be further decomposed by introducing the variational distribution $q'(z|x)$ as follows:

$$\log p_\theta(x) = \mathbb{E}_Z \left[\log \frac{p_\theta(x|z)p(z)}{q'(z|x)} \cdot \frac{q'(z|x)}{p(z|x)}\right] \tag{3.4}$$

This can be split into two parts:

$$\log p_\theta(x) = \mathbb{E}_z[\log p_\theta(x|z)] - \mathbb{E}_z \left[\log \frac{q'(z|x)}{p(z)}\right] + \mathbb{E}_z \left[\log \frac{q'(z|x)}{p(z|x)}\right] \tag{3.5}$$

Writing in terms of Kullback-Leibler divergence to simplify the expression [8]:

$$\log p_\theta(x) = \mathbb{E}_z[\log p_\theta(x|z)] - D_{KL}(q'(z|x)\|p(z)) + D_{KL}(q'(z|x)\|p(z|x)) \tag{3.6}$$

Finally, the Evidence Lower Bound (ELBO) is derived as [8]:

$$\log p_\theta(x) \geq \mathbb{E}_z[\log p_\theta(x|z)] - D_{KL}(q'(z|x)\|p(z)) = L_{VAE}(x) \tag{3.7}$$

This ELBO is used as the objective function for training VAEs, where the aim is to maximize this lower bound to achieve efficient approximation of the posterior distribution. As seen from equation 3.7, standard VAE minimizes $D_{\mathrm{KL}}(q'(z|x)\|p(z|x))$. But as we are using VAE for regression, where k is the parameter to be estimated and x is the observed data. We will see different architectures of VAEs which will aid in doing parameter estimation inverse problem.

3.2 Conditional Variational Autoencoder (C-VAE)

3.2.1 Conditional VAE Architecture for one parameter estimation

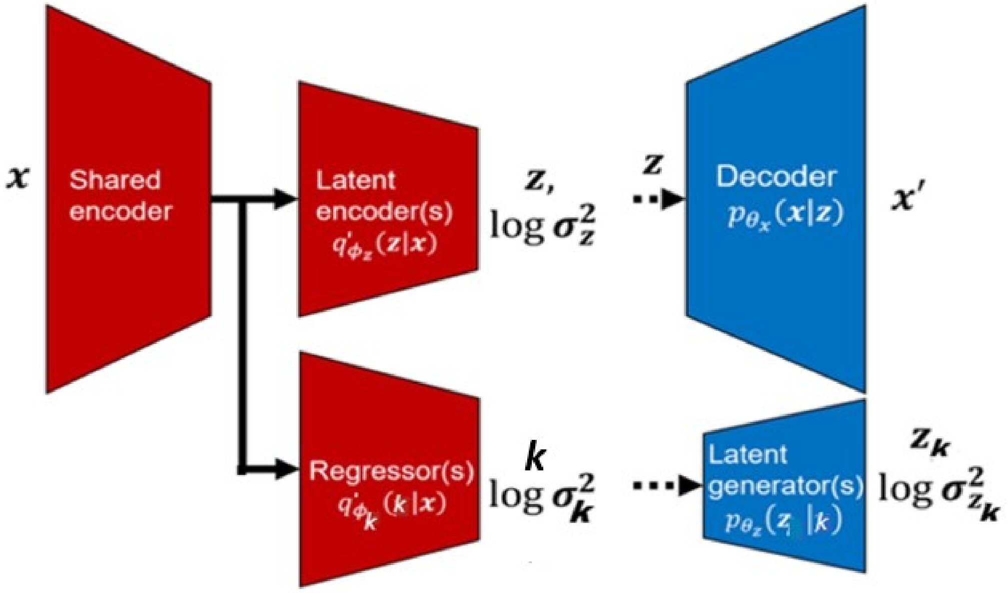
x
Shared encoder
Latent encoder(s) $q'_{\phi_z}(z|x)$
z, $\log \sigma_z^2$
z
Decoder $p_{\theta_x}(x|z)$
x'
Regressor(s) $q'_{\phi_k}(k|x)$
k $\log \sigma_k^2$
Latent generator(s) $p_{\theta_z}(z|k)$
z_k $\log \sigma_{z_k}^2$

Figure 3.2: C-VAE Architecture for 1 parameter estimation

The primary goal in training a Conditional Variational Autoencoder (C-VAE) is to maximize the probability of the observed data x while effectively capturing the underlying structure using latent variables z and conditioning variables k.

The log-likelihood of the observed data can be expressed in terms of latent and conditioning variables [9]:

$$\log p(x) = \log \int\int p(x|z,k)p(z,k)\,dz\,dk \tag{3.8}$$

where $p(x|z,k)$ is the probability of observing x given latent variables z and conditioning variables k and $p(z,k)$ represents the joint prior distribution of the latent and conditioning variables.

To make computation feasible, we introduce a variational distribution $q'(z,k|x)$ to approximate the true posterior $p(z,k|x)$. This approximation allows us to derive a lower bound on the log-likelihood, known as the Evidence Lower Bound (ELBO) [9]:

$$\log p(x) \geq \mathbb{E}_{q'(z,k|x)}[\log p(x|z,k)] - D_{KL}(q'(z,k|x)\|p(z,k)) \tag{3.9}$$

Based on mean field theory, assuming $q'(z,k|x) = q'(z|x)q'(k|x)$ and $p(z,k) = p(z|k)p(k)$, the KL divergence can be decomposed as [9]:

$$D_{KL}(q'(z,k|x)\|p(z,k)) = D_{KL}(q'(k|x)\|p(k)) + \mathbb{E}_{q'(k|x)}[D_{KL}(q'(z|x)\|p(z|k))] \quad (3.10)$$

With all these assumptions, the ELBO simplifies to [9]:

$$L(x) = \mathbb{E}_{q'(z|x)}[\log p(x|z)] - D_{KL}(q'(k|x)\|p(k)) - \mathbb{E}_{q'(k|x)}[D_{KL}(q'(z|x)\|p(z|k))] \quad (3.11)$$

This formulation combines a reconstruction term that enhances the model's ability to generate data similar to x from z and k. KL divergence terms aligns the variational posteriors $q'(z|x)$ and $q'(k|x)$ with their respective priors, ensuring that the distributions learned are informed by prior knowledge. The ELBO is a balance of expected log probabilities (reconstruction quality) against the penalties for deviating from the prior distributions. By maximizing this ELBO during training, the C-VAE learns to reconstruct the observed data.

3.2.2 Conditional VAE Architecture for two parameter estimation

Here the goal in training a Conditional Variational Autoencoder (C-VAE) is to maximize the probability of the observed data x while effectively capturing the underlying structure using latent variables z and conditioning variables k and q.

The log-likelihood of the observed data can be expressed in terms of latent and conditioning variables:

$$\log p(x) = \log \int \int \int p(x|z,k,q)p(z,k,q)\, dz\, dk\, dq \quad (3.12)$$

where $p(x|z,k,q)$ is the probability of observing x given latent variables z and parameters k and q. $p(z,k,q)$ represents the joint prior distribution of the latent variables, influenced by the parameters k and q.

To make computation feasible, we introduce a variational distribution $q'(z,k,q|x)$ to approximate the true posterior $p(z,k,q|x)$. This approximation allows us to derive a lower bound on the log-likelihood, known as the Evidence Lower Bound (ELBO):

$$\log p(x) \geq \mathbb{E}_{q'(z,k,q|x)}[\log p(x|z,k,q)] - D_{KL}(q'(z,k,q|x)\|p(z,k,q)) \quad (3.13)$$

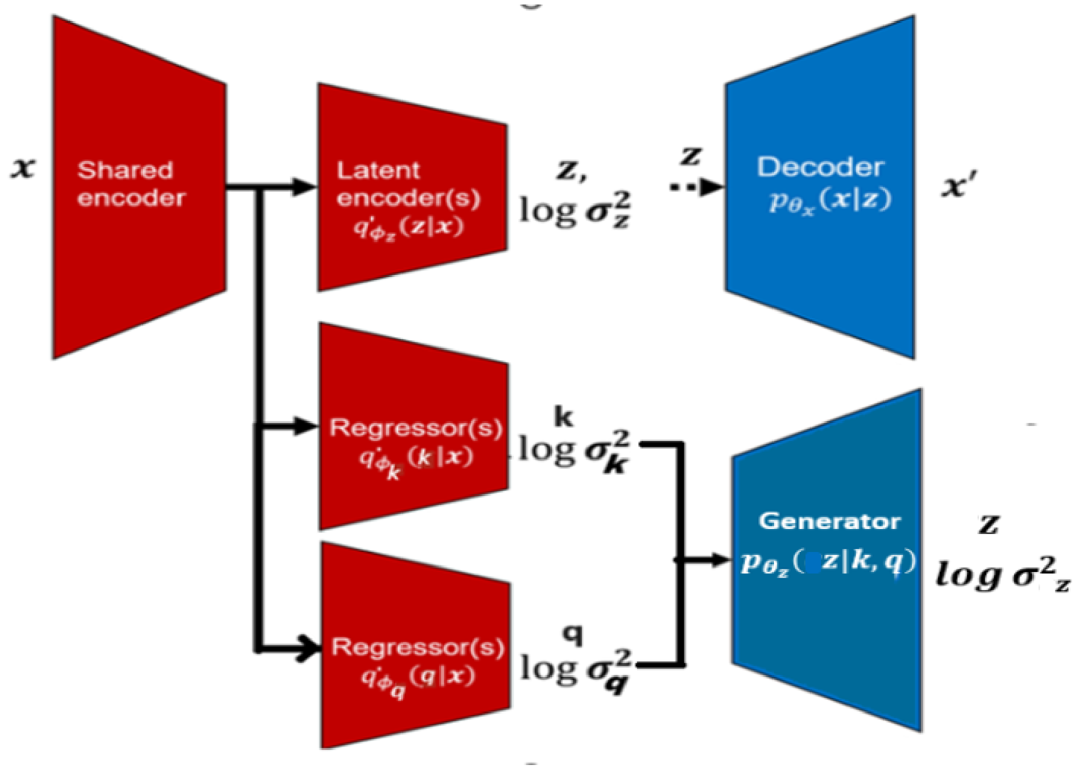
x
Shared encoder
Latent encoder(s) $q'_{\phi_z}(z|x)$
z, $\log \sigma_z^2$
z
Decoder $p_{\theta_x}(x|z)$
x'
Regressor(s) $q'_{\phi_k}(k|x)$
k $\log \sigma_k^2$
Regressor(s) $q'_{\phi_q}(q|x)$
q $\log \sigma_q^2$
Generator $p_{\theta_z}(z|k,q)$
z $\log \sigma^2{}_z$

Figure 3.3: Conditional VAE Architecture for 2 parameter estimation

Based on mean field theory ,assuming $q'(z,k,q|x) = q'(z|x)q'(k|x)q'(q|x)$ and $p(z,k,q) = p(z|k,q)p(k,q)$, the KL divergence can be decomposed as:

$$\begin{aligned} D_{KL}(q'(z,k,q|x)\|p(z,k,q)) = D_{KL}(q'(k|x)\|p(k)) + D_{KL}(q'(q|x)\|p(q)) \\ + \mathbb{E}_{q'(k|x)q'(q|x)}[D_{KL}(q'(z|x)\|p(z|k,q))] \end{aligned} \tag{3.14}$$

With all these assumptions, the ELBO simplifies to:

$$\begin{aligned} L(x) = \mathbb{E}_{q'(z|x)}[\log p(x|z)] - D_{KL}(q'(k|x)\|p(k)) - D_{KL}(q'(q|x)\|p(q)) \\ - \mathbb{E}_{q'(k|x)q'(q|x)}[D_{KL}(q'(z|x)\|p(z|k,q))] \end{aligned} \tag{3.15}$$

This formulation combines a reconstruction term that enhances the model's ability to generate data similar to x from z , k and q. KL divergence terms that align the variational posteriors $q'(z|x)$, $q'(k|x)$ and $q'(q|x)$ with their respective priors, ensuring that the distributions learned are informed by prior knowledge.

3.3 Uncertainty Quantification Variational Autoencoder

3.3.1 Uncertainty Quantification VAE Architecture

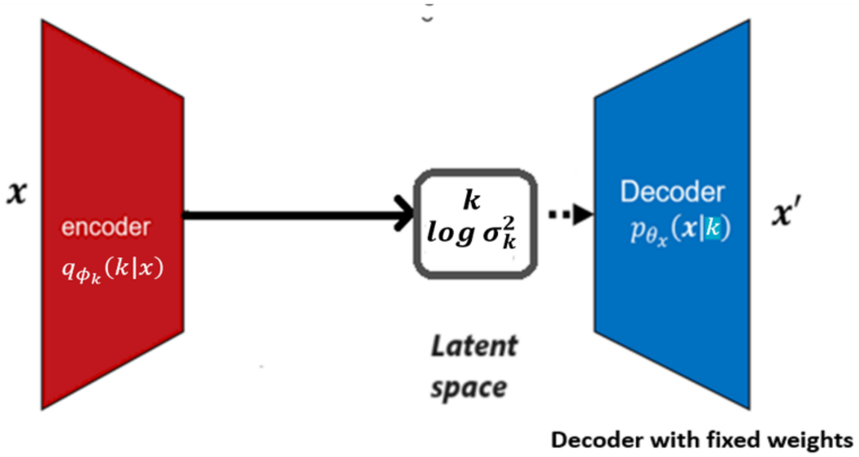

Figure 3.4: UQ-VAE Architecture

In C-VAE architecture the parameter estimated space and the latent space are different. In uncertainty quantification architecture, the parameter of interest space and latent space is same. If more parameters are to be estimated then the latent space dimension should be equal to the number of parameters estimated. Also the physical laws governing the problem can also be incorporated by training the decoder neural network using the parameter to observation (PtO) map F. This fixed weight decoder is then used in the uncertainty quantification architecture. Other sources of information required remains same as earlier VAE architectures which includes the physical properties of the parameter to be estimated through the prior model and the paired parameter to be estimated and observation datasets acquired through simulation or experimentation procedures.

3.3.2 Uncertainty Quantification VAE Loss Function

We are using a dataset containing M pairs of points of interest (PoI) and their corresponding observations, denoted as $(k^{(m)}, x_{\text{obs}}^{(m)})$. Our prior distribution for the parameters is denoted as p(k). To solve this, we are using a neural network ψ that takes the observation data x_{obs} as input and outputs the statistics of our posterior model, represented as $(\mu_{\text{post}}, \Gamma_{\text{post}})$ where μ denotes mean and Γ denotes variance. This means we are expressing the statistics of our Gaussian posterior model using the neural network's weights W. Let $p(k|x)$ denote the target posterior density we wish to estimate and let $q'(k|x)$ denote our model of the target distribution.To optimize the parameters effectively, we need a measure of how close our model posterior is to the target posterior. In this context, we will utilize the Jensen Shannon (JS) divergence as the metric to quantify this distance between the two distributions. For simplicity, let $p(k|x) = P$, $q'(k|x) = Q$ and p(k) = P_{pr} .The Jensen–Shannon divergence (JSD) is a symmetrized and smoothed version of the KL divergence. JS Divergence is defined by [3]:

$$JS_\alpha(Q||P) = \alpha D_{KL}(Q||(1-\alpha)Q + \alpha P) + (1-\alpha) D_{KL}(P||(1-\alpha)Q + \alpha P) \quad (3.16)$$

where α is the weighting factor used to balance the contributions of two probability distributions Q and P in the divergence calculation α value lies between 0 and 1.

Theorem

Let α in $(0, 1)$.Then [3]:

$$\frac{1}{\alpha} JS_\alpha(Q||P) = -E_Q[log(\alpha + \frac{(1-\alpha)Q}{P})] + log(P(x)) - L_{JS} \quad (3.17)$$

where [3]:

$$L_{JS} = \frac{1-\alpha}{\alpha} E_P[log(\alpha + \frac{(1-\alpha)Q}{P})] + E_Q[log(\frac{P(x,k)}{Q})] \quad (3.18)$$

The implementation perspective of this theorem does not provide significant insight. Hence, we present the following corollary, which provides valuable terms related to Kullback-Leibler divergence (KLD).

Corollary

Let α in $(0, 1)$. Consider the equations in the above theorem. By dropping the α from within the logarithms in both the equations we get[3]:

$$\begin{aligned} \frac{1}{\alpha} JS_\alpha(Q||P) \leq -D_{KL}(Q||P) + \log(p(x)) - \log(1-\alpha) - \frac{(1-\alpha)\log(1-\alpha)}{\alpha} \\ + \frac{1-\alpha}{\alpha} D_{KL}(P||Q) - E_Q[\log P] + D_{KL}(Q||P_{\text{pr}}) \end{aligned} \quad (3.19)$$

In Particular we have [3]:

$$-L_{JS} \leq \frac{(1-\alpha)log(1-\alpha)}{\alpha} + \frac{1-\alpha}{\alpha} D_{KL}(P||Q) - E_Q[log(P)] + D_{KL}(Q||P_{pr}) \quad (3.20)$$

The corollary's importance lies in its ability to minimize the upper bound [3]:

$$\frac{1-\alpha}{\alpha} D_{KL}(p(k|x)||q'_\phi(k|x)) - E_{k\sim q'_\phi}[log(p(x|k)] + D_{KL}(q'_\phi(k|x)||p(k)) \quad (3.21)$$

We start by approximating the first term in the upper bound. Initially, we leverage the established concept that minimizing the Kullback-Leibler Divergence (KLD) between empirical and model distributions is akin to maximizing the likelihood function. Next, we integrate deep learning into the framework by introducing a neural network denoted as ψ. This leads us to the following optimization problem. For Gaussian noise $N(\mu_E, \Gamma_E)$ and prior models $N(\mu_{\text{pr}}, \Gamma_{\text{pr}})$, we obtain the following form [3]:

$$\begin{aligned}
\min_W \frac{1}{M} \sum_{m=1}^{M} & \left\| x_{\text{obs}}^{(m)} - F\left(k_{\text{draw}}^{(m)}(W)\right) - \mu_E \right\|_{\Gamma_E^{-1}}^2 \\
& + \frac{1-\alpha}{\alpha}\left[\log\left|\Gamma_{\text{post}}^{(m)}\right| + \left\|\mu_{\text{post}}^{(m)} - k^{(m)}\right\|_{\Gamma_{\text{post}}^{(m)-1}}^2\right] \\
& + \operatorname{tr}\left(\Gamma_{\text{pr}}^{-1}\Gamma_{\text{post}}^{(m)}\right) + \left\|\mu_{\text{post}}^{(m)} - \mu_{\text{pr}}\right\|_{\Gamma_{\text{pr}}^{-1}}^2 + \log\left|\Gamma_{\text{pr}}\right| - \log\left|\Gamma_{\text{post}}^{(m)}\right|
\end{aligned} \quad (3.22)$$

where

$$\begin{aligned}
\mu_{\text{post}}^{(m)}, \Gamma_{\text{post}}^{(m)} &= \Psi\left(x_{\text{obs}}^{(m)}, W\right), \quad \text{Encoder} \\
k_{\text{draw}}^{(m)}(W) &= \mu_{\text{post}}^{(m)} + \Gamma_{\text{post}}^{(m)\frac{1}{2}}\epsilon^{(m)}, \quad \epsilon^{(m)} \sim \mathcal{N}(0, I_D), \quad \text{Reparameterization}
\end{aligned}$$

Chapter 4

Model Problem

4.1 One dimensional steady state heat conduction problem

In this problem, we will be estimating the thermal conductivity of material from temperature measurements taken along the length of the material subjected to one dimensional steady state heat conduction. General form of one dimensional heat transfer through a plain wall having thermal conductivity k, length L and A be the area of wall normal to the direction of heat transfer and having no internal heat generation is given by [10]:

$$\frac{\partial}{\partial l}\left(k\frac{\partial T}{\partial l}\right) + \dot{g} = \rho C\frac{\partial T}{\partial t} \tag{4.1}$$

The wall is subjected to constant heat flux q in $\mathrm{W\,m^{-2}}$ on one side and other side of the wall is subjected to convective heat transfer having heat transfer co-efficient h in $\mathrm{W\,m^{-2}\,K^{-1}}$.The surrounding temperature is fixed and is denoted by T_{surr}.

$$q = -k\left(\frac{dT}{dl}\right)_{l=0} \qquad -k\left(\frac{dT}{dl}\right)_{l=L} = h\left(T(L) - T_{\mathrm{surr}}\right) \tag{4.2}$$

Considering steady state with no heat generation, isotropic material and above boundary conditions, the equation takes the form,

$$\frac{d^2T}{dx^2} = 0 \tag{4.3}$$

Integrating wrt x, once we get, $\frac{dT}{dl} = C_1$
and twice we get, $T(x) = C_1 l + C_2$

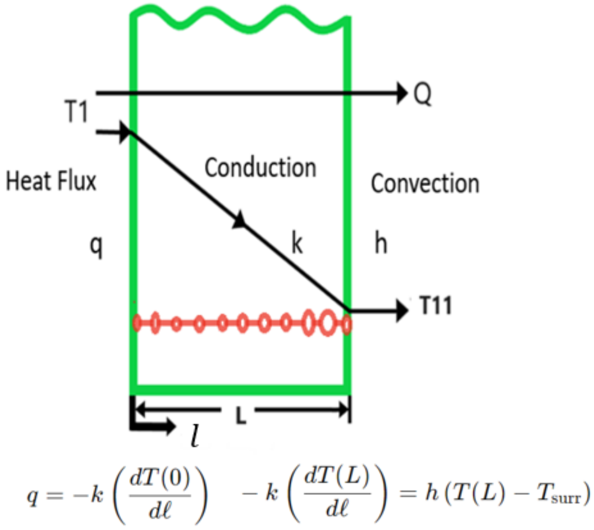
T1
Q
Heat Flux
Conduction
Convection
q
k
h
T11
L
l
$q = -k\left(\frac{dT(0)}{d\ell}\right)$
$-k\left(\frac{dT(L)}{d\ell}\right) = h\left(T(L) - T_{\text{surr}}\right)$

Figure 4.1: One dimensional Steady state heat conduction

Substituting above boundary conditions we get the co-efficients C_1 and C_2, the final equation of temperature distribution along the length of wall then is,

$$T(x) = q\left(\frac{L-l}{k} + \frac{1}{h}\right) + T_{\text{surr}} \tag{4.4}$$

From above 4.4, we can find Temperature T at any given l, if other parameters such as q, h and k are known. We have considered q = 1000 $\mathrm{W\,m^{-2}}$, h = 12.5 $\mathrm{W\,m^{-2}\,K^{-1}}$ and T_{surr} as 298K. The length of the wall is 1 meter and temperature T is measured at equal intervals of 10mm, we get 11 measurements of temperature along the length of the wall. We generate 1000 data points, randomly choosing 1000 distinct values of k in the range (10,200) and using in equation 4.4. This problem can also be extended to two parameter estimation in which the parameters to be estimated we have considered as q and k. Other conditions remains same as one parameter estimation problem. In this also, we generated 1000 data points by randomly choosing 1000 distinct k values in the range (10,200) and q values in the range (1000, 1100) and using in equation 4.4. We also require log variance of k for the prior distribution. The standard deviation in k is taken as 5% value of k and the log variance of k is calculated from this standard deviation value of k. For two parameter estimation log variance of q is calculated by same method as log variance of k.

4.2 Two dimensional steady state heat conduction problem

General form of two dimension steady state heat conduction with no internal heat generation is given by [3]:

$$-\nabla \cdot k\nabla T = 0 \; in \, \Omega \tag{4.5}$$

Considering isotropic material the governing PDE becomes [3]:

$$-k(\nabla T \cdot n) = 0.5T \; on \, \Omega^{ext} \tag{4.6}$$

$$-k(\nabla T \cdot n) = -1 \; on \, \Omega^{root} \tag{4.7}$$

where, k denotes the thermal heat conductivity. Ω is the physical domain, Ω^{root} is the bottom edge of the domain and $\Omega^{ext} = d\Omega - \Omega^{root}$ is the remaining edges of the domain [3]. We use the Fenicsx Library for data generation.The FEniCSx library employs the Finite Element Method (FEM) [11] to numerically solve partial differential equations (PDEs). The Finite Element Method (FEM) breaks down a complex system into smaller elements known as finite elements to solve a problem. This process involves formulating the PDE, generating a mesh for the domain, defining the variational form, solving the equations using FEM, and obtaining numerical solutions for data generation purposes. FEniCSx is particularly useful for efficiently solving complex PDEs in scientific and engineering simulations. The FEM minimizes an error function, derived from the calculus of variations, to approximate the solution effectively.We consider the computational domain to be a Unit Square, consisting of D = 2601 degrees of freedom. The observation data corresponds to sensor measurements of temperature at 10 randomly selected locations on the plate. The data values for k and log variance k follow the same method as one dimensional steady state heat transfer one parameter estimation. The difference lies in the temperature values now follows two dimension steady state heat conduction equation.

Chapter 5

Experimentation and Results

We will be plotting the results for R2 score and RMSE using different Variational autoencoder architectures. Also we will study the effect of increasing noise percentage on R2 score results and predicted standard deviation values by VAEs.

5.1 Results for one dimensional steady state heat conduction problem

Results for one parameter estimation inverse problem

VAE	R2 Score	RMSE
C-VAE	0.978	8.021
UQ-VAE	0.991	5.144

Table 5.1: R2 Score for thermal conductivity at 5% Gaussian Noise

As it can be seen from above results of one parameter estimation at 5% Gaussian noise, UQ-VAE performs better than C-VAE in terms of higher R2 score and lower RMSE.

Comparison with MCMC for one parameter estimation inverse problem

As it can be seen from the results of one parameter estimation at 5% Gaussian noise, MCMC performs better than UQ-VAE in terms of higher R2 score and lower RMSE. Also UQ-VAE fails to estimate when values of parameters are high.

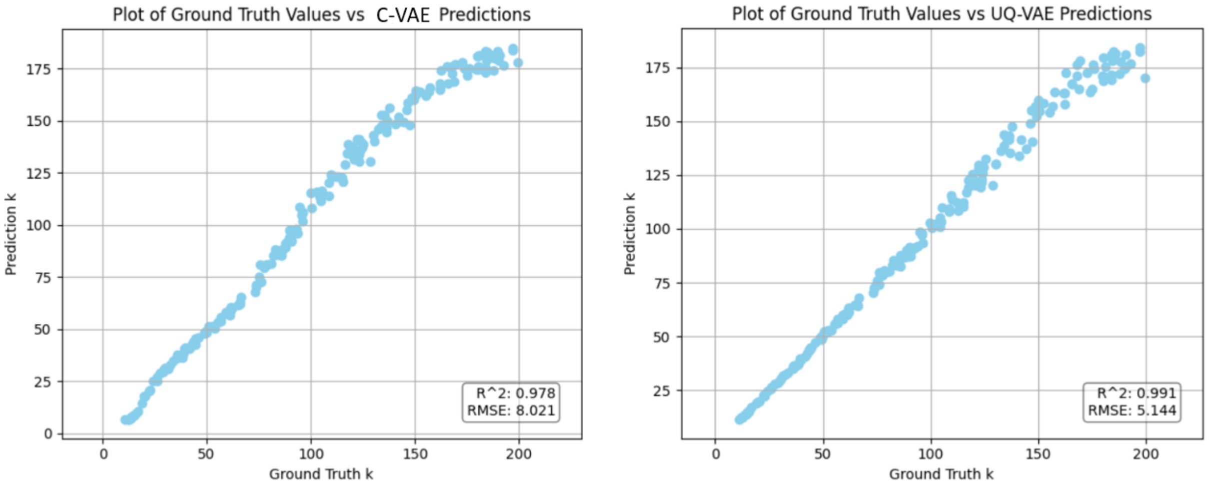

Figure 5.1: Result for one parameter estimation inverse problem at 5% Gaussian Noise

Method	R2 Score	RMSE
UQ-VAE	0.99	5.55
MCMC	1.00	0.80

Table 5.2: R2 Score for thermal conductivity at 5% Gaussian Noise

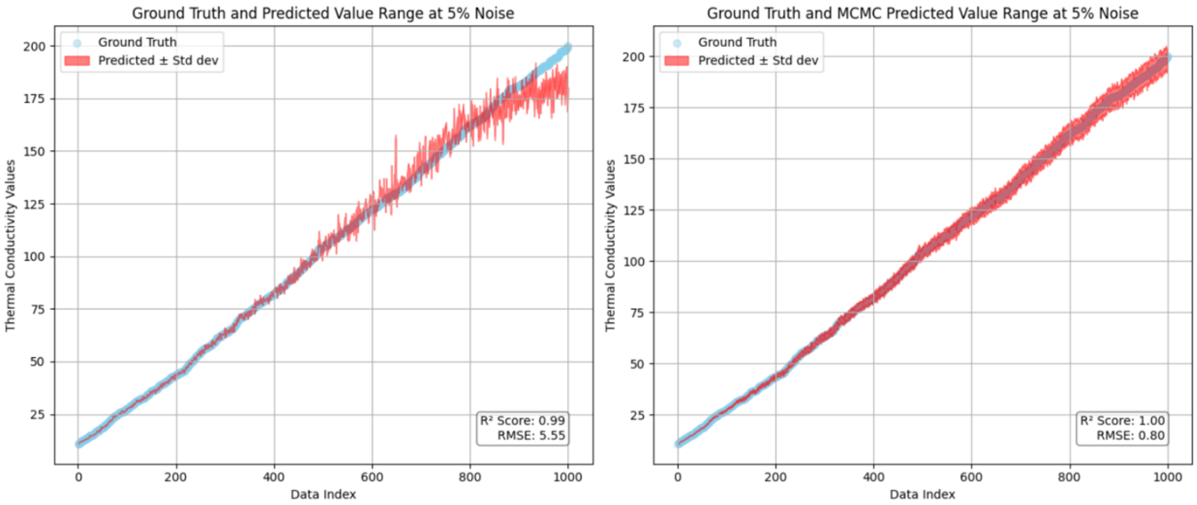

Figure 5.2: One parameter estimation comparison between UQ-VAE and MCMC.png

Results for two parameter estimation inverse problem

As it can be seen from the results of two parameter estimation at 5% Gaussian noise, UQ-VAE performs better than C-VAE in terms of higher R2 score and lower RMSE

VAE	R2 Score	RMSE
C-VAE	0.981	3.920
UQ-VAE	0.996	1.741

Table 5.3: R2 Score for heat flux at 5% Gaussian Noise

VAE	R2 Score	RMSE
C-VAE	0.947	4.988
UQ-VAE	0.987	2.475

Table 5.4: R2 Score for thermal conductivity at 5% Gaussian Noise

for both the parameters.

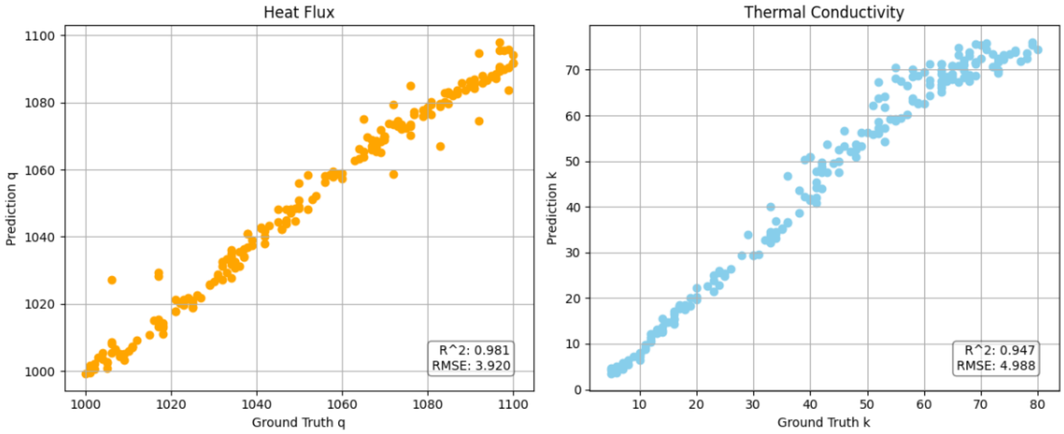

Figure 5.3: Results for C-VAE two parameter estimation at 5% Gaussian Noise

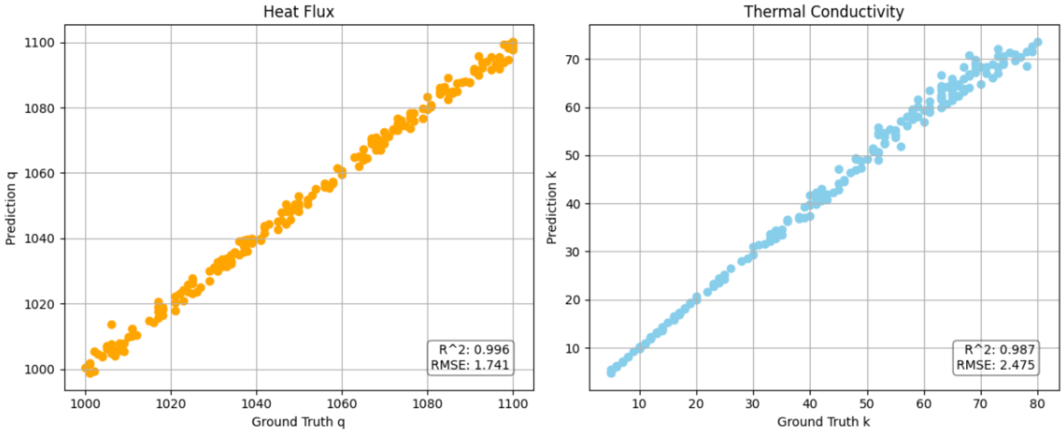

Figure 5.4: Results for UQ-VAE two parameter estimation at 5% Gaussian Noise

5.2 Results for two dimensional steady state heat conduction problem

Results for one parameter estimation inverse problem and 2 dimensional data

VAE	R2 Score	RMSE
C-VAE	0.930	14.346
UQ-VAE	0.982	7.313

Table 5.5: R2 Score for thermal conductivity at 5% Gaussian Noise

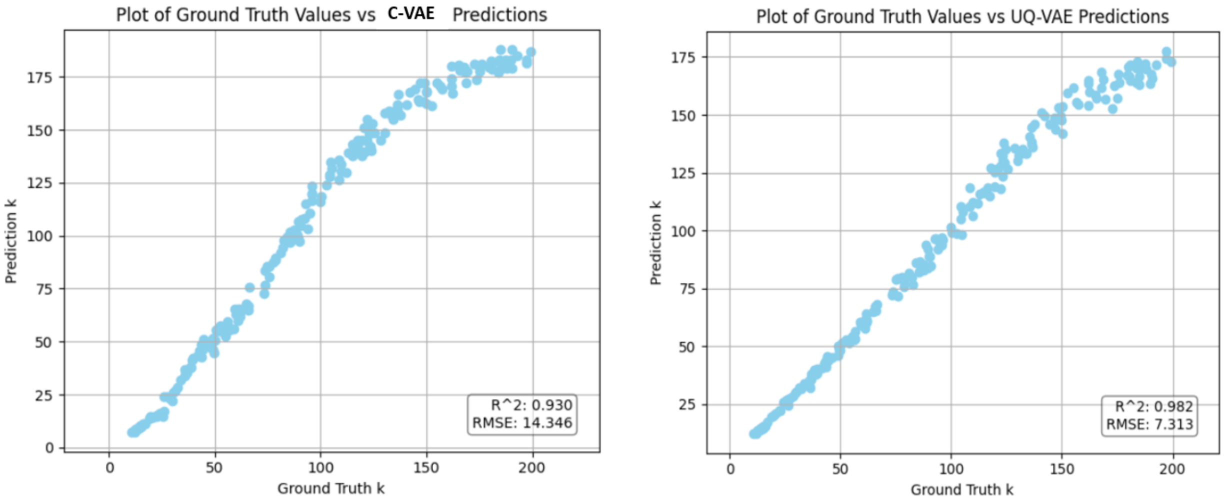

Figure 5.5: Results for one parameter estimation for 2-Dimensional data at 5% Gaussian noise

As it can be seen from above results of one parameter estimation at 5% Gaussian noise, UQ-VAE performs better than C-VAE in terms of higher R2 score and lower RMSE when the temperature changes in two dimensions.

5.3 Effect of Noise on estimated values of parameters

As it can be seen from the results of one parameter estimation , as noise percentage increases in the observations R2 score decreases, RMSE value increases and values

% Noise	R2 Score	RMSE
5	0.99	5.55
20	0.94	13.44

Table 5.6: R2 Score for thermal conductivity predictions by UQ-VAE at different noise percentage

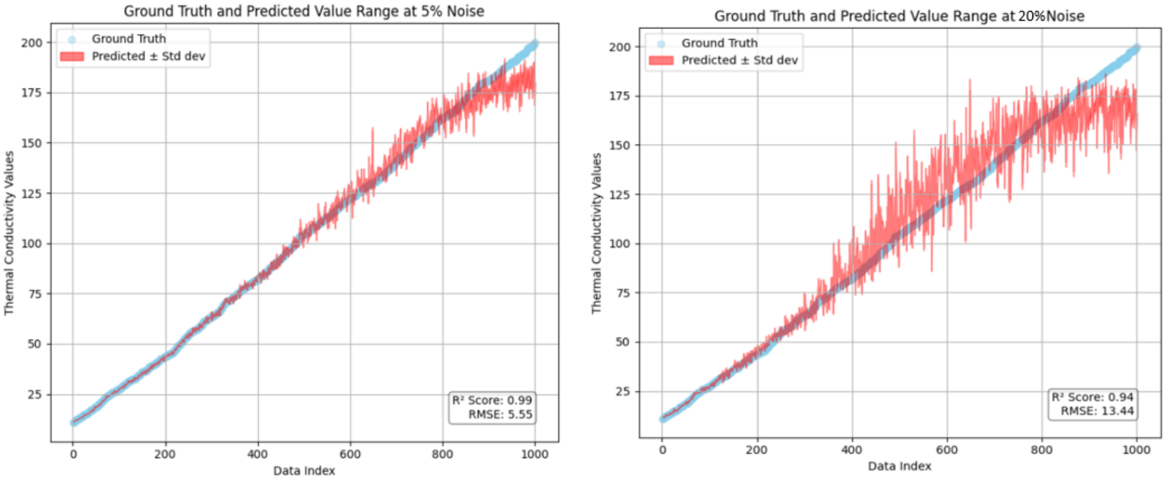

Figure 5.6: Effect of Noise on R2 score and predicted Standard deviation

predicted for standard deviation of parameters by VAE increases as percentage noise increases.

Chapter 6

Conclusion

After studying the results, we have seen that UQ-VAE performs better than C-VAE in terms of higher R2 score and lower RMSE at all noise levels. UQ-VAE architecture has the advantage of incorporating physics of the problem. Also the latent space and parameter of interest space is same in UQ-VAE unlike C-VAE. While studying the effect of increase in percentage noise, we see that R2 score decreases, RMSE increases and values predicted by VAEs for standard deviation of parameter increases. The increase in values of standard deviation is also an indication that the model has less confidence in its predictions. MCMC outperforms VAEs in terms of higher R2 scores and predicting uncertainty, ensuring the ground truth lies within the predicted range. At lower noise levels, MCMC and VAEs perform similarly, but MCMC excels as noise increases. As seen in Figure 5.6 VAEs are not able to predict close to actual values when the value of parameter to be estimated is high. We have also worked with two-dimensional data and two-parameter estimation, observing that VAEs can predict parameter values closer to the ground truth.

Outlook

This study can be extended to develop models for higher dimensions and multiple parameter estimation. There is significant scope for improving the model architecture to ensure that the predicted values are even closer to the ground truth. While our current study focuses on steady-state problems and parameter estimation, future research can extend these findings to function estimation and transient state problems.

Bibliography

[1] Albert Tarantola. *InverseProblemTheory*. 2004.

[2] Bernd Hofmann and Robert Plato. On ill-posedness concepts, stable solvability and saturation, 2017.

[3] Hwan Goh, Sheroze Sheriffdeen, Jonathan Wittmer, and Tan Bui-Thanh. Solving bayesian inverse problems via variational autoencoders, 2021.

[4] Moorad Alexanian. Nature, science, bayes' theorem, and the whole of reality, 2015.

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. 2006.

[6] Joshua S. Speagle. A conceptual introduction to markov chain monte carlo methods, 2020.

[7] Shima Kamyab, Zohreh Azimifar, Rasool Sabzi, and Paul Fieguth. Survey of deep learning methods for inverse problems, 2021.

[8] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[9] Qingyu Zhao, Ehsan Adeli, Nicolas Honnorat, Tuo Leng, and Kilian M. Pohl. Variational autoencoder for regression: Application to brain aging analysis, 2019.

[10] John H. Lienhard and John H. Lienhard. *A Heat Transfer Textbook*. Phlogiston Press, Cambridge, Massachusetts, third edition, August 2003.

[11] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals, Sixth Edition*. Butterworth-Heinemann, 6 edition, May 2005.