



STOCHASTIC VARIATIONAL INFERENCE AND BAYESIAN NEURAL NETWORK PRIORS FOR INVERSION

Lappeenranta–Lahti University of Technology LUT

Degree Programme in Computational Engineering, Master's thesis

2024

Akseli Suutari

Examiners: Professor Lassi Roininen

Professor Tapio Helin

Abstract

Lappeenranta–Lahti University of Technology LUT

LUT School of Engineering Sciences

Degree Programme in Computational Engineering

Double-degree in co-operation with partner university: Università della Svizzera italiana

Akseli Suutari

Stochastic Variational Inference and Bayesian Neural Network Priors for Inversion

Master's thesis

2024

44 pages, 17 figures

Examiners: Professor Lassi Roininen and Professor Tapio Helin

Keywords: probabilistic deep learning, computational statistics, deconvolution, uncertainty quantification

Inverse problems are computational problems in which, using a known forward model, an unknown property is solved from measured data caused by the unknown in interest. Inversion, in general, is considered ill-posed, which causes the solutions to be unstable if applicable regularization is not used. Regularization can be done with the Bayesian framework, in which the solution is found by updating random variables and well-designed prior information to posterior distribution using the likelihood function.

In various inverse problems, the unknown is a function, and thus, the prior is also constructed in a function space. In this thesis, the prior functions are created using Bayesian neural networks with parameters endowed with α -stable prior distributions. Because of the large number of parameters, sampling the posterior on the Bayesian neural network is slow with traditional Markov chain Monte Carlo methods. We propose the usage of stochastic variational inference to perform inversion with neural network priors efficiently.

We study the proposed method in deconvolution inverse problems. Stochastic variational inference proved to be an efficient alternative to traditional posterior sampling when constructing neural network priors in applications where the reconstruction needs to be done within time limits.

Tiivistelmä

Lappeenrannan–Lahden teknillinen yliopisto LUT

LUTin insinööritieteiden tiedekunta

Laskennallisen tekniikan koulutusohjelma

Kaksoistutkinto yhteistyössä: Università della Svizzera italiana

Akseli Suutari

Stokastinen variaatioinferenssi ja Bayesilaiset neuroverkkopriorit inversiossa

Diplomityö

2024

44 sivua, 17 kuvaa

Tarkastajat: Professori Lassi Roininen ja Professori Tapio Helin

Avainsanat: tilastollinen syväoppiminen, laskennallinen tilastotiede, dekonvoluutio, epävarmuuden kvantifiointi

Inversio-ongelmat ovat laskennallisia ongelmia, joissa tunnetun suoran mallin ja mitatun datan avulla ratkaistaan tuntematon ominaisuus, josta mitattu data riippuu. Inversio-ongelmat ovat huonosti aseteltuja ongelmia, mikä johtaa epästabiileihin ratkaisuihin, jos tarvittavaa regularisointia ei ole tehty. Regularisointia voi käyttää Bayesilaista lähestymistapaa, jossa ratkaisua etsitään muodostamalla posteriorijakauma satunnaismuuttujien, a priori -tiedon ja todennäköisyysfunktion avulla.

Usein inversio-ongelmissa ratkaistava ominaisuus on funktio, jolloin priori-tiedon tulee myös olla funktioavaruudessa. Tässä työssä priorifunktiot muodostetaan käyttäen Bayesilaista neuroverkkoa, jonka parametreilla on α -stabiilit priorijakaumat. Neuroverkoilla on suuri määrä parametreja, minkä takia Bayesilaisen neuroverkon posteriorijakauma on hidasta muodostaa perinteisten Markovin ketju Monte Carlo -menetelmien avulla. Tässä työssä ehdotamme stokastisen variaatioinferenssin käyttöä neuroverkon painojen posteriorin muodostamiseen.

Tutkimme esiteltyä lähestymistapaa dekonvoluutio-ongelman avulla. Työssä näytetään stokastisen variaatioinferenssin olevan tehokas vaihtoehto perinteisille metodeille neuroverkkopriorreja käytettäessä erityisesti, kun ratkaisu tulee löytää aikarajoitteiden puitteissa.

Acknowledgements

First, I would like to thank Professor Lassi Roininen and Professor Tapio Helin from Lappeenranta-Lahti University of Technology LUT for sharing their expertise as my supervisors and examiners. I would also like to thank Dr. Angelina Senchukova for all the help regarding the Bayesian neural networks and Dr. Teemu Härkönen for sharing his knowledge of Bayesian statistics.

I would also like to thank my family for supporting me on my journey. I want to thank my friends, bandmates, and members of the computational engineering guild Lateksii ry for the extracurricular activities that have made life at the university even more enjoyable. Lastly, I want to thank my partner, Alla, for being supportive and understanding throughout my studies and the thesis process.

This thesis has been spell-checked using Grammarly software to guarantee linguistic quality. The author verified all the suggested corrections.

Lappeenranta, October 18, 2024

Akseli Suutari

Symbols and abbreviations

Abbreviations

ANN	Artificial Neural Network
BNN	Bayesian Neural Network
CAVI	Coordinate Ascent Mean-field Variational Inference
CM	Conditional Mean
ELBO	Evidence Lower Bound
HMC	Hamiltonian Monte Carlo
KL	Kullback-Leibler
MAP	Maximum a Posteriori
MCMC	Markov chain Monte Carlo
NN	Neural Network
NUTS	No U-Turn Sampler
RKHS	Reproducing Kernel Hilbert Space
RV	Random Variable
SVI	Stochastic Variational Inference
TV	Total Variation
UQ	Uncertainty Quantification
VI	Variational Inference

Table of contents

Abstract	
Acknowledgements	
Symbols and abbreviations	
Table of contents	6
1 Introduction	8
1.1 Contributions	10
1.2 Structure of the thesis	11
2 Inverse problem	12
2.1 Regularization	13
2.2 Bayesian Inverse Problems	14
2.3 Likelihood function	16
2.4 Prior	16
2.5 Function space priors	17
3 Bayesian Neural Networks and SVI	19
3.1 Neural Networks	19
3.2 Bayesian Neural Network	20
3.3 Markov chain Monte Carlo and Hamiltonian Monte Carlo	22
3.4 No U-Turn Sampler	24
3.5 Variational Inference	24
3.6 Stochastic Variational Inference	28
3.7 Modeling the measurement noise	29
4 Numerical experiments	30
4.1 Deconvolution	30
4.2 Numerical experiments on deconvolution	31
4.3 Increasing the number of discretization points	32
4.4 Noise in the measurements	34
4.5 SVI training iterations	36

4.6	Comparison between SVI and MCMC	38
4.7	Gaussian parameters with high prior standard deviation	39
5	Conclusions	40
	References	42

1 Introduction

Computers are being integrated into more and more fields of life. This is caused by the advancements in computer chip and sensor manufacturing, which have led to both increased computation power and more accurate sensing while allowing the components to be smaller at the same time. This has enabled the development of solutions for more complex tasks in various different fields. The solutions can be fully algorithmic or combine measurements from sensors with algorithms. Fields such as medicine with the usage of machine learning in cancer detection (Kourou et al., 2015) and biology through the protein structure research with the Alphafold (Jumper et al., 2021) have benefitted from the increased computational capacity and powerful algorithms.

A computational problem consists of three components: input, model, and output. To solve the problem, we need to know at least one of these. Based on which of the components are known, computational problems are categorized into *forward problems*, *inverse problems*, and *learning problems*. These categories are illustrated in Figure 1. In the figure, the forward model $f(x, e)$ maps the x to the y . It also has a parameter e that contains the errors in the system (Kaipio & Somersalo, 2005).

In this thesis, we focus on inverse problems. From Figure 1, we can see that in inversion, we assume to know the forward model with a goal of solving an unknown \mathbf{x} using the forward model and measurements of \mathbf{y} . For a linear forward model, this can be written as

$$\mathbf{y} = (\mathcal{O} \circ \mathcal{A})\mathbf{x} + \varepsilon \quad (1)$$

where noise ε is additive and \mathcal{O} is an observation operator that maps the forward solution to the observation space. To obtain a stable solution in inversion, regularization needs to be used to constrain the solution. In this thesis, the inversion will be done in a Bayesian setting, in which *Bayes' rule*

$$\pi(\mathbf{x} | \mathbf{y}) = \frac{\pi(\mathbf{y} | \mathbf{x})\pi(\mathbf{x})}{\pi(\mathbf{y})} \quad (2)$$

is used to formulate the *posterior* distribution $\pi(\mathbf{x} | \mathbf{y})$ for the unknown based on the data using the *likelihood* $\pi(\mathbf{y} | \mathbf{x})$ and *a priori* knowledge $\pi(\mathbf{x})$. In a Bayesian setting, the regularization is done by choosing a suitable prior (Kaipio & Somersalo, 2005). If the unknown \mathbf{x} is a function, we want our prior to also be a function.

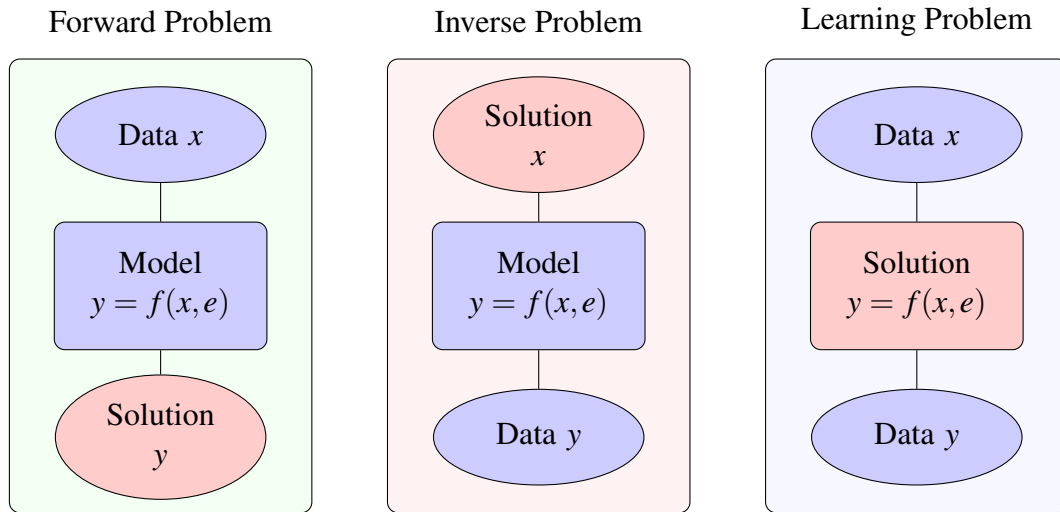


Figure 1: The three different types of computational problems. Blue nodes are the known properties, and red illustrates what the problem aims to learn.

One way to construct prior functions is in function space by using Karhunen-Loève expansion to produce continuous functions with covariance functions. (Uribe et al., 2020) We will introduce this method later in Chapter 2, where we discuss the creation of priors in more detail. In their study Roininen, Huttunen & Lasanen (2014) constructed Whittle-Matérn correlation functions to model the correlations in the unknown \mathbf{x} . These functions were used as a prior in solving two-dimensional electrical impedance tomography. The Whittle-Matérn correlation functions are smooth because of the Gaussian random fields used. Both priors with Karhunen-Loève expansion and Whittle-Matérn correlation functions work well for inversion, where the underlying x is smooth.

Many applications of inverse problems require the detection of features with sharp jumps. For this type of problem, smooth priors should not be used since they are not able to produce sharp realizations. To solve this problem, Lassas & Siltanen (2004) utilized Total Variation (TV) priors in inversion. TV prior is a discretization-based method for prior construction. In essence, TV utilizes derivatives to measure how a function varies. The goal is to penalize the solution with large variations. This property encourages smoothness similar to Gaussian priors, but it also allows sharp jumps making TV priors suitable for edge-preserving inversion, where discontinuities are expected. However, it was shown that TV prior is not discretization invariant. That is, it converges to Gaussian prior as the discretization size decreases.

For edge-preserving inversion, Besov prior has been proposed by Lassas, Saksman & Siltanen (2009). Essentially Besov priors are constructed similarly to Karhunen-Loève expansion priors, but the functions are created with a wavelet expansion instead of the Karhunen-Loève expansion. This allows discontinuities in the functions, making them suitable for

edge-preserving solutions. In addition, unlike the TV priors, Besov priors are discretization invariant. Markkanen et al. (2019) introduced a method for edge-preserving inversion by utilizing Cauchy difference priors. The upside of the suggested method is that it is discretization invariant. That is, the realizations do not converge to smooth functions as the number of discretization steps increases.

Neural networks (NN) can model complex functions by utilizing perceptrons developed by Rosenblatt (1958). A neural network consists of perceptrons, that are connected. The connections are endowed with deterministic weights and biases that are trained with input-output pairs of data and a gradient-based optimization algorithm. A trained NN can be used to make predictions for unseen input data. Neural networks have evolved into more complex methods, such as convolutional neural networks by LeCun, Bengio & Hinton (2015) for computer vision tasks and attention-preserving transformers by Vaswani et al. (2017), enabling the development of large language models.

The problem with neural networks is that they cannot provide ways to do uncertainty quantification (UQ), that is, to evaluate the confidence of the predictions. Neal (1996) proposed the usage of distributions instead of fixed parameters for the neural network, enabling the UQ for the neural networks formulating the Bayesian neural networks (BNN).

Bayesian neural networks in inversion have been previously studied by Li, Dunlop & Stadler (2022), Suuronen (2023) and Senchukova (2024). It has been proven that a BNN can be used as a functional prior and that it can produce both smooth and sharp realizations just by endowing the parameters with different prior distributions. In general smooth realizations can be produced with a network that has normally distributed parameters. If the parameter priors are Cauchy distributions, the network promotes rough features.

In addition to the introduced methods for prior generation, there are also additional methods such as machine learning (Marschall et al., 2023) and data-driven priors (Holden, Pereyra & Zygalakis, 2021) that have been proposed. However, we will not discuss these in this thesis as we focus on the function-space priors.

1.1 Contributions

Previous studies by Li, Dunlop & Stadler (2022), Suuronen (2023) and Senchukova (2024) have utilized *Markov chain Monte Carlo* (MCMC) methods for the model inference. Because MCMC methods formulate the posterior by sampling random variables, they do not scale well when the parameter space increases. To overcome this limitation, this thesis studies the usage of *Stochastic Variational Inference* (SVI) in Bayesian inversion (Hoffman, Blei,

et al., 2013). Instead of sampling the posterior, the goal in SVI is to optimize the parameters of a variational distribution so that it approximates the posterior. We will show later that this can be done without knowing the true posterior. The variational inference has previously been compared to MCMC methods by David M. Blei & McAuliffe (2017), but in their study, they did not experiment with BNNs or inversion.

1.2 Structure of the thesis

This thesis is organized as follows: In Chapter 2, we define inverse problems and define the need for regularization. We discuss the classical methods for regularization and solving inverse problems. After that, we discuss the statistical approach to inversion and address Bayesian methods and the importance of the prior. We conclude the chapter by introducing Karhunen-Loève expansion as a method of constructing Gaussian prior functions and addressing the need for more flexible priors.

In Chapter 3, we introduce neural networks and how they can be extended to BNNs. We lay the foundation for the methods of using BNNs as a functional prior in inversion and discuss MCMC methods, No U-turn sampler, and define variational inference.

In Chapter 4, we first introduce the deconvolution inverse problem and then study the BNN and SVI in the deconvolution. In the experiments, we reconstruct three different signals: a continuous function, a piecewise constant function, and a function with a combination of continuous and sharp jumps. With the experiments, we will study how SVI works with the BNN in different settings. Lastly, we compare the SVI and No U-turn sampler. Finally, we conclude the study in Chapter 5 by discussing the key findings of the study and the possibilities for future work.

2 Inverse problem

Let us suppose that we have two quantities $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^m$. To take into account measurement noise and poorly known parameters, we also introduce a quantity $\mathbf{e} \in \mathbb{R}^k$. We can relate these to each other via mapping $f : \mathbb{R}^n \times \mathbb{R}^k \mapsto \mathbb{R}^m$. Using the defined properties, we can formulate a computational problem as

$$\mathbf{y} = f(\mathbf{x}, \mathbf{e}). \quad (3)$$

We call the mapping f the forward model (Kaipio & Somersalo, 2005). By utilizing the forward model, given the \mathbf{x} , we are able to compute the corresponding \mathbf{y} that is to solve a forward problem. If, however, we have measurements of \mathbf{y} along with known $f(\cdot)$, solving the \mathbf{x} from the Equation (3) would be an inverse problem. Inversion is essential in various applications such as in x-ray tomography (Siltanen et al., 2003; Springer et al., 2023) and in electrical impedance tomography (Roininen, Huttunen & Lasanen, 2014).

Inverse problems are generally considered difficult to solve due to them often being *ill-posed problems*. Hadamard (1923) stated that a problem is ill-posed if it is not well-posed that is, it does not satisfy at least one of the following conditions:

1. **Existence:** The problem has a solution.
2. **Uniqueness:** The solution is unique.
3. **Stability:** The solution depends continuously on the data. Small changes in the data result in small changes in the solution.

The ill-posedness of inversion is typically caused by instability for the changes in the data, which arises when dealing with measurements since measured data always contains noise. In the Equation (3), the noise is contained in the parameter \mathbf{e} . The effect of the noise varies in different problems since it can be of different types and from various distributions. Commonly noise is standard mean-zero Gaussian or Poisson-distributed, which usually affects discrete measurements. For example, in X-ray measurements, the noise is Poisson-distributed because the photon counts in X-ray are discrete (Kaipio & Somersalo, 2005). Mean-zero Gaussian noise is typical for continuous measurements. It is worth noting, that Poisson distribution can be approximated with Gaussian distribution given enough samples due to the law of large numbers. In this thesis, we focus on Gaussian noise.

As briefly discussed in Chapter 1, supposing, that the forward model in Equation (3) is a linear operator $\mathcal{A}(t)$, we can formulate a linear inverse problem as finding the \mathbf{x} given measurement data \mathbf{y} seen in Equation (1), in which the noise is mean-zero additive Gaussian

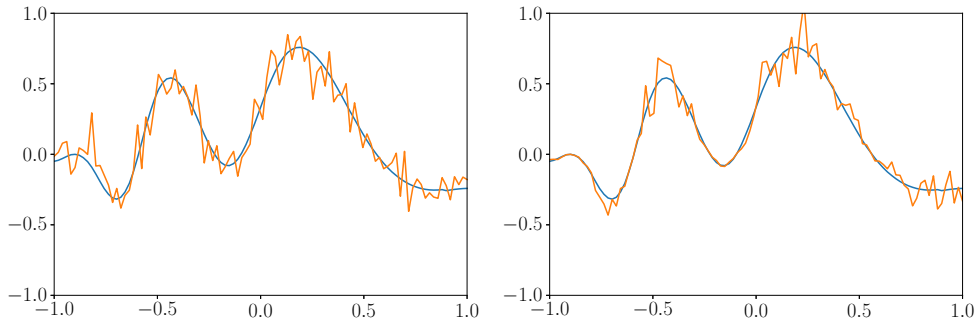


Figure 2: Examples of noise with additive noise on the left and multiplicative noise on the right.

$\varepsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$. Another common form of noise is multiplicative $\mathcal{A}\mathbf{x}(t)(1 + \varepsilon)$. In Figure 2, we illustrate additive and multiplicative Gaussian noise where $\varepsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$. We can notice that additive noise has the same effect on the whole signal, meaning that the noise level is constant and multiplicative noise affects the signal more the higher the absolute value of the signal is.

In Chapter 4, we will show that with naive methods for inversion, the measurement noise makes the solutions unstable because of the ill-posedness. We can mitigate the instabilities by *regularizing* the solution.

2.1 Regularization

To preserve stability in ill-posed problems, we apply regularization. This means that the model is constrained or penalized to make it behave like a well-posed problem. A simple example of classical regularization is *Tikhonov regularization* for a linear inverse problem (Miller & Karl, 2003; Vauhkonen et al., 1998). To define Tikhonov regularization, let us first suppose that we want to solve the \mathbf{x} from the Equation (1). We can define the solution to be the \mathbf{x}^\dagger , that minimizes the norm $\|\mathcal{A}\mathbf{x}^\dagger - \mathbf{y}\|^2$.

Tikhonov regularization constrains the solution by introducing a regularization term $\lambda \|\mathbf{x}\|^2$ with λ as a regularization parameter. This regularization term penalizes the model by having large values of \mathbf{x} , which helps to remove extreme fluctuations from the solution. The Tikhonov regularized solution \mathbf{x}^λ is then the minimizer of functional (Kaipio & Somersalo, 2005)

$$\min_{\mathbf{x}} \{ \|\mathcal{A}\mathbf{x} - \mathbf{y}\|^2 + \lambda \|\mathbf{x}\|^2 \}. \quad (4)$$

The solution for the minimization problem (4) can be found with the ordinary least squares method

$$\mathbf{x}^\lambda = \left(\mathcal{A}^\top \mathcal{A} + \lambda \mathbf{I} \right)^{-1} \mathcal{A}^\top \mathbf{y} \quad (5)$$

The regularization parameter λ controls how much regularization is applied to the solution. With a too small λ the model fits to the noise in the data whereas with too large λ the model over-smooths the solution. The optimal λ can be found by utilizing *Morozov discrepancy principle*. Assuming that we have an estimate of the norm of the error $\varepsilon > 0$ from the data vector \mathbf{y} , we can then conclude that any \mathbf{x}_λ such that

$$\|\mathcal{A} \mathbf{x}^\lambda - \mathbf{y}\| \leq \varepsilon$$

is an acceptable solution to the problem. Practically we want the solution for which the discrepancy principle is equal to the noise. This way, the model does not fit into the noise (Kaipio & Somersalo, 2005).

Regularization can also be done with statistical methods. As discussed in Chapter 1, this thesis will focus on Bayesian statistics in inversion. In the next section, we will introduce the Bayesian framework for inversion and discuss how we can regularize with Bayesian methods.

2.2 Bayesian Inverse Problems

The goal of statistical inversion is to extract information and assess uncertainty about the variables based on all the knowledge we have about the system (Kaipio & Somersalo, 2005). By taking a Bayesian approach to inverse problems, we evaluate the variables using random variables (RV) and *Bayes' theorem* described in Equation (2) with the goal of constructing a posterior distribution for the solution \mathbf{x} .

An RV X is defined as a measurable function $X : \Omega \rightarrow E$ that maps from the sample space Ω to the measurement space E . Random variables are denoted as capital letters, and their realizations as lowercase letters. In inverse problems, the random variable Y is called the measurement, and the random variable X is the unknown.

A collection of RVs is called a stochastic process, that given probability space (Ω, \mathcal{F}, P) , formulates a stochastic process using three components:

- *Sample space* Ω consisting of all the possible outcomes.

- *Event space* that has all the events \mathcal{F} . An event is a set of outcomes in the sample space.
- A probability function P assigning each event in event space a probability between 0 and 1.

We can define a stochastic process $X(\mathbf{t}, \omega)$ with $\mathbf{t} \in D \subset \mathbb{R}^n$ and $\omega \in \Omega$. From the stochastic process, for a fixed spatial coordinate $t_i \in D$ we get a random vector $\mathbf{X} = X(t_i, \omega)$ and for a fixed element of the sample space $\omega_i \in \Omega$ we produce a *realization* for the vector \mathbf{t} . We can write the Equation (1) for random vectors as

$$\mathbf{Y} = \mathcal{A}\mathbf{X} + \varepsilon. \quad (6)$$

By having measured data that is a realization of $\mathbf{Y} = \mathbf{y}_{\text{observed}}$ that for convenience we will denote as \mathbf{y} , we can construct the *posterior distribution*, which is essentially a distribution where the solution \mathbf{x} belongs to. The posterior is defined with the Bayes' rule introduced in Equation (2) as the conditional probability of \mathbf{x} and \mathbf{y}

$$\pi_{\text{post}}(\mathbf{x}) = \pi(\mathbf{x} | \mathbf{y}) = \frac{\pi(\mathbf{y} | \mathbf{x})\pi(\mathbf{x})}{\pi(\mathbf{y})}, \quad (7)$$

where the denominator, *evidence*, $\pi(\mathbf{y})$ is

$$\pi(\mathbf{y}) = \int_{\mathbb{R}^n} \pi(\mathbf{y} | \mathbf{x})\pi(\mathbf{x})d\mathbf{x} \quad (8)$$

and is a constant.

The posterior probability density is the estimation of how the unknown \mathbf{x} is distributed. To get a point estimation for the \mathbf{x} we can use *maximum a posteriori* (MAP) estimate

$$\mathbf{x}_{\text{MAP}} = \underset{\mathbf{x} \in \mathbb{R}^n}{\text{argmax}} \pi(\mathbf{x} | \mathbf{y}). \quad (9)$$

In some cases, it is possible to find a closed-form solution for the MAP, but when the dimensionality increases, this becomes difficult. If the closed form can not be formulated, numerical methods can be used. It is also worth noting that the maximizer of the posterior might not exist or it might not be unique.

Another common point estimate is called *conditional mean* (CM) that is defined as

$$\mathbf{x}_{\text{CM}} = \mathbb{E}\{\mathbf{x} | \mathbf{y}\} = \int_{\mathbb{R}^n} \mathbf{x} \pi(\mathbf{x} | \mathbf{y}) d\mathbf{x}. \quad (10)$$

Because the conditional mean is an integral function, it produces smooth solutions, whereas MAP might contain rough features. However, integration space \mathbb{R}^n is often high-dimensional, making the computation of the integral not feasible. Because of this limitation, we will use the MAP estimator in this thesis. We will use variational Bayes methods to formulate the posterior and get the MAP estimation. This will be covered more thoroughly in Chapter 3.

2.3 Likelihood function

In the Equation (7), the density $\pi(\mathbf{y} | \mathbf{x})$ is called the *likelihood function*. The likelihood describes how likely the model would produce the measured data \mathbf{y} given \mathbf{x} . In inversion, the likelihood is defined by the difference between the data \mathbf{y} and the predicted values transformed with the forward model. For a model (6) with a zero-mean Gaussian additive noise with a variance of σ^2 and the data $\mathbf{y} \in \mathbb{R}^m$, the likelihood is defined as

$$\pi(\mathbf{y} | \mathbf{x}) = \frac{1}{(2\pi)^{m/2} \sigma^m} \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{y} - f(\mathbf{x})\|_2^2\right). \quad (11)$$

For the linear inverse problem with Gaussian noise introduced in Equation (1), the likelihood (11) can be further formulated to

$$\pi(\mathbf{y} | \mathbf{x}) = \frac{1}{(2\pi)^{m/2} \sigma^m} \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{y} - \mathcal{A}(\mathbf{x})\|_2^2\right). \quad (12)$$

The likelihood is used in combination with the prior to formulate the posterior distribution. In the next section, we will discuss the different methods of constructing priors.

2.4 Prior

In Bayesian statistics, we assume to have some *a priori* belief or knowledge of the unknown \mathbf{X} before the measurement of \mathbf{Y} . This information is expressed in Equation (7) as the probability distribution $x \mapsto \pi_{\text{pr}}(x)$ called *prior*. The prior is an essential part of Bayesian inversion since it can act as a regularizer if it is designed to limit the parameters into the desired range (Kaipio & Somersalo, 2005).

If we have knowledge of how the unknown is distributed before measurements, we can con-

struct an *informative* prior that assumes some distribution for the unknown. An example of an informative prior could be a Gaussian prior, which assumes the unknown to be normally distributed. If there is no relevant information about the model parameters, an *uninformative* prior, such as uniform prior, that assigns equal probability for all values, should be used.

In Bayesian statistics, if the unknown is a function, we want our prior to also be a function. As discussed in Chapter 1, the unknown to be reconstructed can be either a smooth function such as in spectroscopy (Härkönen et al., 2024), a function with discontinuities for example, X-ray tomography (Markkanen et al., 2019) or a combination of both. For each problem, we want to design a prior that promotes the desired structure of the unknown. That is, the prior should be able to have smooth and rough features.

Generally, there are two ways to create priors for functions: *function space-based* and *discretization-based*. In Chapter 1, we introduced various methods for creating priors. In this thesis, we focus on priors created with neural networks, which will be discussed in Chapter 3, but we will first briefly introduce the creation of priors with Karhunen-Loève expansion.

2.5 Function space priors

Let \mathcal{H} be a Hilbert space of real functions f on an index set D . This space is a Reproducing Kernel Hilbert Space (RKHS), if there exists a function $k : D \times D \mapsto \mathbb{R}$ that has properties:

- for every $\mathbf{x}, k(\mathbf{x}, \mathbf{x}') \in \mathcal{H}$ and
- k has reproducing property, $\langle f(\cdot), k(\cdot, \mathbf{x}') \rangle_{\mathcal{H}} = f(\mathbf{x}')$.

The $k(\cdot, \mathbf{x}')$ is called a kernel function. The reproducing property of $k(\cdot, \mathbf{x}')$ allows to represent any $f(\mathbf{x})$ as the inner product of $f(\cdot)$ and $k(\cdot, \mathbf{x}')$. In essence, we can use the kernel function to approximate the $f(\cdot)$ by only knowing \mathbf{x} and \mathbf{x}' . The selection of the kernel functions determines the properties of the produced feature space.

If we have a RKHS and a continuous kernel k , we can define a Hilbert-Schmidt operator, that is compact and has a complete set of eigenvectors in $L^2(D)$. Then there exists an orthonormal basis with eigenfunctions $\{\phi_i\} : D \mapsto L^2(D)$ and eigenvalues $\{\lambda_i\} \in [0, \infty)$ such that $\lambda_i \geq \lambda_{i+1}$ and $\lim_{i \rightarrow \infty} \lambda_i = 0$. Using these properties, we can write the covariance kernel as

$$C_{\text{HH}}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{\infty} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{x}'). \quad (13)$$

by utilizing *Merzel expansion*. A common choice for the covariance kernel is the *Matérn covariance* kernel.

We can represent Gaussian second-order random field $H(\mathbf{x}, \boldsymbol{\omega})$ in D defined over probability space (Ω, \mathcal{F}, P) and equipped with covariance function $C_{HH}(\mathbf{x}, \mathbf{x}')$ for a truncation level k as a Karhunen - Loève expansion as

$$H(\mathbf{x}, \boldsymbol{\omega}) \approx H(k, \mathbf{x}, \boldsymbol{\omega}) = \mu_H + \sum_{i=1}^{\infty} \mathbb{1}(i \leq k) \sqrt{\lambda_i} \phi_i(\mathbf{x}) \theta_i(\boldsymbol{\omega}) \quad (14)$$

where $\mathbb{1}(\cdot)$ is an indicator function and $\boldsymbol{\theta} = \{\theta_i(\boldsymbol{\omega}) : \Omega \rightarrow \mathbb{R}\}$ are mean zero, unit variance mutually uncorrelated RVs.

It is worth noting that the Karhunen-Loève expansion generates only continuous function realizations since it constructs Gaussian prior. Thus the priors with it are not applicable for edge-preserving tasks. For these tasks, a *Besov space prior* (Lassas, Saksman & Siltanen, 2009) utilizing wavelet expansion instead of Karhunen-Loève expansion has been developed. A new approach by Li, Dunlop & Stadler (2022) constructed function spaced priors with neural networks, and this will be the chosen method of this thesis and will be discussed in detail in Chapter 3.

3 Bayesian Neural Networks and SVI

As discussed in Section 2.5 Li, Dunlop & Stadler (2022) introduced a method of utilizing neural networks as priors by taking advantage of their flexibility in modeling functions to overcome the limitations of Gaussian and Besov priors. In this chapter, we introduce Bayesian neural networks and discuss their usage as a prior.

Bayesian neural networks are an extension of the traditional artificial neural networks (ANN), in which the parameters are statistical distributions instead of fixed parameters. As discussed in Chapter 1, one of the upsides of BNNs compared to traditional neural networks is that BNNs allow us to do uncertainty quantification in neural networks because of their usage of Bayesian statistics.

3.1 Neural Networks

A neural network is a mathematical model that takes inspiration from the biological brain. A fully connected feed-forward neural network consists of multiple layers of perceptrons connected. The first layer is called an input layer, the last layer is an output layer, and the layers in between are hidden layers.

Defining the NN to have $L \geq 1$ layers with n_l units in each layer, we can write the network in vector-matrix form for the domain vector $\mathbf{t} \in \mathcal{D}$ as

$$\begin{aligned} F^{(0)}(\mathbf{t}) &= \mathbf{t} \\ F^{(\ell+1)}(\mathbf{t}) &= \mathbf{W}^\ell \varphi(\mathbf{F}^\ell + \mathbf{B}^\ell), 0 \leq \ell \leq L-1 \\ \Phi^{(\text{NN})}(\mathbf{t}) &= \mathbf{W}^L \mathbf{F}^L + \mathbf{B}^L, \end{aligned}$$

where $\mathbf{W}^\ell \in \mathbb{R}^{n_{l+1} \times n_l}$ are the weights of the layer ℓ , $\mathbf{W}^L \in \mathbb{R}^{1 \times n_L}$ the weights of the output layer, $\mathbf{B}^\ell \in \mathbb{R}^{n_{l+1} \times 1}$ contains the biases of the layer and $\varphi : \mathbb{R} \rightarrow \mathbb{R}$ is a non-linear activation function to introduce non-linearity to the network. We denote the output of the NN as $\Phi(\mathbf{t})$.

We do not apply the activation function to the output layer to allow the most flexibility for the output. In this thesis, a hyperbolic tangent $\tanh(\cdot)$ is used as an activation to allow the output to be differentiable. This property is mandatory for training with variational methods that will be introduced later in the chapter.

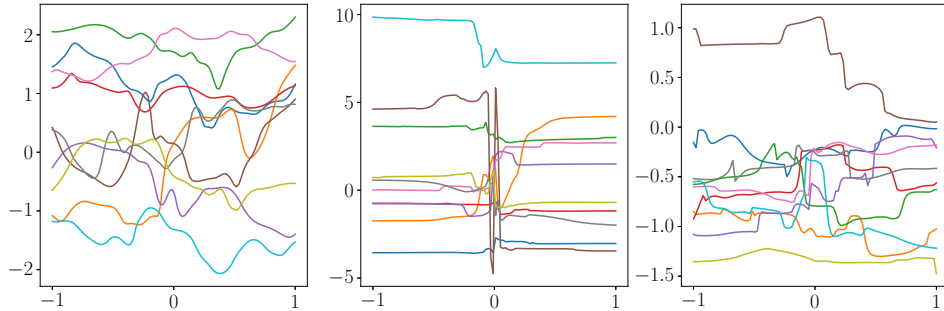


Figure 3: Example realizations from BNNs with different priors for the weights and biases. On the left only Gaussian layers, on the middle only Cauchy layers and on the right a combination of Gaussian and Cauchy layers.

With NNs, we are interested in model parameters, which are the weights and biases. We denote the model parameters as a combined vector $\boldsymbol{\theta}$. When training the network, we will optimize the $\boldsymbol{\theta}$. To calculate the number of parameters in $\boldsymbol{\theta}$, we can use the following formula

$$d_{\text{NN}} = \sum_{\ell=0}^L (n_{\ell}n_{\ell+1} + n_{\ell+1}).$$

The number of parameters affects the complexity of the model in the sense that with more nodes, the network can represent a more complex function. However, increasing the d_{NN} will also increase the time to train the model.

Training a traditional NN is an optimization problem, where the predicted output of the model is evaluated against the measured dataset with a *loss-function*. The loss is then used to evaluate how much the parameters need to be updated. The updating process is done by utilizing a gradient-descent algorithm, in which the parameters are updated in the opposite direction of the gradient. It allows the largest descent in the loss to find the minimum efficiently. The gradients with respect to the weights and biases are calculated with *back-propagation* algorithm.

3.2 Bayesian Neural Network

In ANNs, the parameters are deterministic. That is, each parameter in $\boldsymbol{\theta}$ is a fixed number that is tuned in the training process. As discussed in Chapter 1, a well-trained neural network can model complex functions. However, the downside of fixed parameters is that there is no way to evaluate the confidence or the error of the model that is to do UQ. To tackle

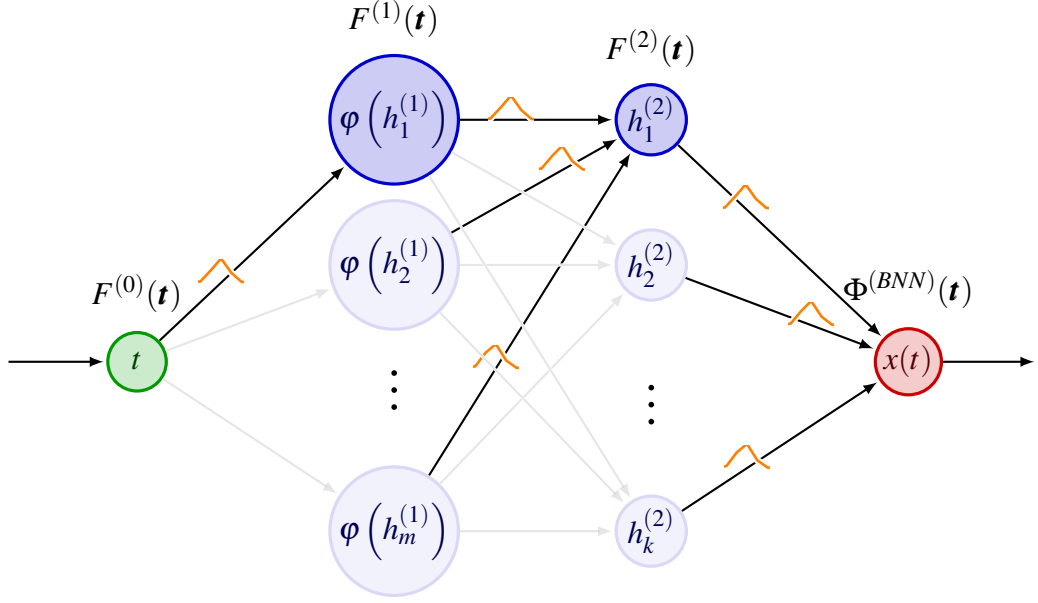


Figure 4: Illustration of a BNN with two hidden layers. The model parameters are shown as distributions.

this limitation, a Bayesian neural network replaces the deterministic parameters with random variables equipped with prior distributions. These priors are then updated to posterior distribution utilizing the likelihood and data.

A proven choice for the parameter priors are α -stable distributions such as Gaussian ($\alpha = 2$) and Cauchy ($\alpha = 1$) distributions (Li, Dunlop & Stadler, 2022; Suuronen et al., 2023). The choice of prior for the parameter RVs affects the realizations the BNN produces. This is illustrated in Figure 3. From the realizations, we can see that the fully Gaussian network produces smooth realizations, and the fully Cauchy network promotes rough features. A network with a combination of Cauchy and Gaussian layers is able to produce realizations that have both smooth and rough parts.

As said before, the activation function $\varphi(\cdot)$ is not applied for the last layer. By having multiple units in the last layer, the variance of the last layer will converge to

$$\mathbb{V}[x(t)] = n_\ell \sigma_w^2 V(t) + \sigma_b^2 \quad (15)$$

where n_ℓ is the number of units in the layer and σ_w^2 is the standard deviation of the layer weights prior. From the Equation 15, we can see that by letting $n_\ell \rightarrow \infty$, the variance of the layer is not bounded. To have the variance bounded, the standard deviations of the hidden-to-output weights need to be scaled with

$$\sigma_w = n_\ell^{-1/\alpha} \sigma_{\text{unscaled}} \quad (16)$$

where α is the stability index of the α -stable prior distribution to obtain a well-defined limit for the output variance.

The difference between a deterministic NN and a BNN comes from the tuning of network parameters $\boldsymbol{\theta}$. In deterministic inversion, we try to find the $\boldsymbol{\theta}$ that minimizes the loss function

$$\mathcal{L}^{(\text{inversion})}(\boldsymbol{\theta}) = \frac{1}{\sigma_{\text{obs}}^m} \|\mathbf{y} - \mathcal{O}(\mathcal{A}\Phi^{(\text{NN})}(\mathbf{t}; \boldsymbol{\theta}))\|_2^2, \quad (17)$$

that is, to find the $\boldsymbol{\theta}$ that minimizes the norm between the data and the output of the network $\Phi^{(\text{NN})}(\mathbf{t}; \boldsymbol{\theta})$ transformed with the matrix \mathcal{A} . The network will be given the discretized domain $\mathbf{t} \in \mathcal{D}$ as an input instead of utilizing the traditional neural network method of input-output pair training process.

As discussed in Section 2.3, in the Bayesian framework, we formulate the *likelihood* function that is used to update the prior distributions of the $\boldsymbol{\theta}$ to the posterior distribution that is to do *Bayesian inference*. We formulate the likelihood for Bayesian inversion as

$$\pi^{(\text{inversion})}(\mathbf{y} | \boldsymbol{\theta}) \propto \frac{1}{\sigma_{\text{obs}}^m} \exp\left(-\frac{1}{2\sigma_{\text{obs}}^2} \|\mathbf{y} - \mathcal{O}(\mathcal{A}\Phi^{(\text{NN})}(\mathbf{t}; \boldsymbol{\theta}))\|_2^2\right). \quad (18)$$

In recent studies by Li, Dunlop & Stadler (2022), Suuronen (2023), and Senchukova (2024), the Bayesian inference was done using Markov chain Monte Carlo methods, which are generally popular in Bayesian statistics since given enough time and space, they are guaranteed to converge to true posterior. The drawback of MCMC methods is that because they explore the parameter space by sampling from the posterior distribution, they suffer from the curse of dimensionality, making it not scale well into large-scale problems. To overcome this limitation, we utilize the *stochastic variational inference* method by Hoffman, Blei, et al. (2013) to efficiently form the posterior distribution for the BNN parameters. In the following sections, we will give a brief introduction to MCMC methods and No U-Turn Sampler (NUTS) for Hamiltonian Monte Carlo (HMC) before introducing the SVI method.

3.3 Markov chain Monte Carlo and Hamiltonian Monte Carlo

MCMC algorithms are a class of methods used to sample distributions by generating Markov chains with their stationary distributions being the same as the target distribution (Tierney,

1994). As previously discussed, MCMC methods always converge to the true posterior given enough time, with the cost of sometimes taking a long time to converge due to the random walks used to explore the parameter space.

Hamiltonian Monte Carlo method replaces the random walks with a concept of Hamiltonian dynamics where the target distribution determines the potential energy in the system. The Hamiltonian dynamics are defined with

$$\frac{dq_i}{dt} = \frac{\partial H(\mathbf{q}, \mathbf{r})}{\partial r_i} \quad (19) \quad \frac{dr_i}{dt} = -\frac{\partial H(\mathbf{q}, \mathbf{r})}{\partial q_i} \quad (20)$$

where \mathbf{q} is a position vector and $\mathbf{r} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{I}_d)$ is a momentum vector. $H(\mathbf{q}, \mathbf{r})$ is the *Hamiltonian operator* defined as

$$H(\mathbf{q}, \mathbf{r}) = -\log p(\mathbf{q}) + \sum_{i=1}^d \frac{1}{2} r_i^2. \quad (21)$$

By combining the equations and discretizing them with a *leapfrog scheme* with a step size δ , we get

$$\begin{aligned} \mathbf{r}\left(t + \frac{\delta}{2}\right) &= \mathbf{r}(t) - \frac{\delta}{2} \frac{\partial U}{\partial \mathbf{q}} \mathbf{q}(t), \\ \mathbf{q}(t + \delta) &= \mathbf{q}(t) + \delta \mathbf{r}\left(t + \frac{\delta}{2}\right), \\ \mathbf{r}(t + \delta) &= \mathbf{r}\left(t + \frac{\delta}{2}\right) - \frac{\delta}{2} \frac{\partial U}{\partial \mathbf{q}} \mathbf{q}(t + \delta). \end{aligned} \quad (22)$$

HMC will perform sampling by first sampling a new \mathbf{r} and then proposing a new state according to the Hamiltonian dynamics simulated with N leapfrog steps. This new state is then accepted or rejected based on the *Metropolis acceptance probability*

$$\alpha(\mathbf{q}^* | \mathbf{r}^*) = \min\left(1, \frac{p(\mathbf{q}^*)q(\mathbf{r}^* | \mathbf{q}^*)}{p(\mathbf{r}^*)q(\mathbf{q}^* | \mathbf{r}^*)}\right) \quad (23)$$

where q is the proposal density. The new state will be accepted with probability α or rejected with probability $1 - \alpha$. The logarithm of the α is dependent on the total energy in the beginning and the end

$$\log \alpha = H(\mathbf{q}, \mathbf{r}) - H(\mathbf{q}^*, \mathbf{r}^*).$$

The HMC is run until the resulting Markov chain has reached a stationary state. The distribution of values in the Markov chain corresponds to the distribution of the sampled parameter.

3.4 No U-Turn Sampler

HMC is dependent on tuning the step size and the number of steps in the leapfrog scheme δ and N . By having too few steps, the Markov chain will not explore the space efficiently, whereas having too many steps makes the chain explore the same parts multiple times and thus waste computational power. This problem is solved by the NUTS sampler, in which the Hamiltonian dynamics are run forwards and backward, adding the results to a binary tree. This is stopped when a U-turn is detected, that is, when the next step would result in the chain going closer to the starting point (Hoffman & Gelman, 2014).

3.5 Variational Inference

With MCMC methods, we are solving a sampling problem where the posterior is sampled multiple times to get the estimates for the desired parameters. As discussed before, this might cause problems when the dimensionality increases since the sampling is generally quite slow. An alternative way to formulate the posterior is to transform the problem from a sampling problem to an optimization problem to enable the use of efficient optimization algorithms. This transformation can be done with *variational inference* (VI) methods.

Before deriving the VI methods, we will define some terminology. Let us suppose, that the BNN model has observations $\mathbf{y}_{1:N}$, local hidden variables $z_{1:N}$, and global hidden variables $\boldsymbol{\theta}$. The global hidden variables, that is, the BNN weights and biases, are shared across all the observations, whereas the local hidden variables, the noise level, are unique to each observation. The hidden variables are assumed conditionally independent. We also assume that the conditional distributions of hidden variables given other hidden variables and the observations are in the exponential family,

$$p(\boldsymbol{\theta} | \mathbf{y}, \mathbf{z}, \alpha) = h(\boldsymbol{\theta}) \exp\left(\boldsymbol{\eta}_g(\mathbf{y}, \mathbf{z}, \alpha)^\top t(\boldsymbol{\theta}) - a_g(\boldsymbol{\eta}_g(\mathbf{y}, \mathbf{z}, \alpha))\right) \text{ and} \quad (24)$$

$$p(z_{nj} | x_n, z_{n,-j}, \boldsymbol{\theta}) = h(z_{nj}) \exp\left(\boldsymbol{\eta}_\ell(x_n, z_{n,-j}, \boldsymbol{\theta})^\top t(z_{nj}) - a_\ell(\boldsymbol{\eta}_\ell(x_n, z_{n,-j}, \boldsymbol{\theta}))\right), \quad (25)$$

where $h(\cdot)$ is the base measure, $\boldsymbol{\eta}(\cdot)$ is the natural parameter with the subscript g for global and ℓ for local variables, $t(\cdot)$ is the sufficient statistic and $a(\cdot)$ is the log-normalizer. For

the conditional distributions, the natural parameter is a function of the conditioned variables (Hoffman, Blei, et al., 2013). These assumptions indicate that both the local context given global variables

$$p(y_n, z_n | \boldsymbol{\theta}) = h(y_n, z_n) \exp\left(\boldsymbol{\theta}^\top t(y_n, z_n) - a_\ell(\boldsymbol{\theta})\right), \quad (26)$$

and the prior $p(\boldsymbol{\theta})$

$$p(\boldsymbol{\theta}) = h(\boldsymbol{\theta}) \exp\left(\boldsymbol{\alpha}^\top t(\boldsymbol{\theta}) - a_g(\boldsymbol{\alpha})\right) \quad (27)$$

must be in the exponential family as well. We can do this by having $t(\boldsymbol{\theta}) = (\boldsymbol{\theta}, -a_\ell(\boldsymbol{\theta}))$ and $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2)$, so we can define natural parameter

$$\boldsymbol{\eta}_g(\mathbf{y}, \mathbf{z}, \boldsymbol{\alpha}) = \left(\boldsymbol{\alpha}_1 + \sum_{n=1}^N t(z_n, x_n), \boldsymbol{\alpha}_2 + N \right), \quad (28)$$

which leads to having (26) and the prior (27) in the same exponential family as (24) and (25) (Hoffman, Blei, et al., 2013).

The goal in VI is to approximate the true posterior with variational distribution $q(\mathbf{z}, \boldsymbol{\theta})$ by minimizing the Kullback-Leibler (KL) divergence between the variational distribution and the true posterior of the network parameters $\pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y})$. We will first show that we can minimize the KL divergence by maximizing the *evidence lower bound* (ELBO).

Let us first define the KL divergence as

$$D_{\text{KL}}(q(\mathbf{z}, \boldsymbol{\theta}) | \pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y})) = \sum_{\mathbf{y} \in \mathcal{Y}} q(\mathbf{z}, \boldsymbol{\theta}) \log \left(\frac{q(\mathbf{z}, \boldsymbol{\theta})}{\pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y})} \right), \quad (29)$$

which can also be written as

$$D_{\text{KL}}(q(\mathbf{z}, \boldsymbol{\theta}) || \pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y})) = \mathbb{E}_q \left[\frac{\pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y})}{q(\mathbf{z}, \boldsymbol{\theta})} \right]. \quad (30)$$

The problem in both Equations (29) and (30) is that they require the true posterior to be computed. To overcome this, we can derive ELBO by taking the posterior of the hidden variables given observations

$$\pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y}) = \frac{\pi(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta})}{\int \pi(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta}) d\mathbf{z} d\boldsymbol{\theta}}, \quad (31)$$

where $y_{1:N}$ are the observations and $\boldsymbol{\theta}$ the vector of hidden variables, and binding the log $\pi(\mathbf{y})$ using Jensen's inequality

$$\begin{aligned} \log \pi(\mathbf{y}) &= \log \int \pi(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta}) d\mathbf{z} d\boldsymbol{\theta} = \log \int \pi(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta}) \frac{q(\mathbf{z}, \boldsymbol{\theta})}{q(\mathbf{z}, \boldsymbol{\theta})} d\mathbf{z} d\boldsymbol{\theta} \\ &= \log \left(\mathbb{E}_q \left[\frac{\pi(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta})}{q(\mathbf{z}, \boldsymbol{\theta})} \right] \right) \\ &\geq \mathbb{E}_q [\log \pi(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta})] - \mathbb{E}_q [\log q(\mathbf{z}, \boldsymbol{\theta})] \triangleq \mathcal{L}(q). \end{aligned}$$

We define the ELBO as $\mathcal{L}(\cdot)$. By substituting the ELBO to the Equation 30, we can write the KL-divergence as

$$\begin{aligned} D_{\text{KL}}(q(\mathbf{z}, \boldsymbol{\theta}) || \pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y})) &= \mathbb{E}_q \left[\frac{p(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta})}{q(\mathbf{z}, \boldsymbol{\theta})} \right] \\ &= \mathbb{E}_q [\log q(\mathbf{z}, \boldsymbol{\theta})] - \mathbb{E}_q [\log \pi(\mathbf{z}, \boldsymbol{\theta} | \mathbf{y})] \\ &= \mathbb{E}_q [\log q(\mathbf{z}, \boldsymbol{\theta})] - \mathbb{E}_q [\log \pi(\mathbf{y}, \mathbf{z}, \boldsymbol{\theta})] + \log \pi(\mathbf{y}) \\ &= -\mathcal{L}(q) + \log \pi(\mathbf{y}). \end{aligned}$$

Because the evidence $\log \pi(\mathbf{y})$ is constant and does not depend on q , we can minimize the KL divergence by maximizing the $\mathcal{L}(q)$.

To choose variational distribution q with suitable properties, Hoffman, Blei, et al. (2013) used the mean-field variational family of distributions

$$q(\mathbf{x}, \boldsymbol{\theta}) = q(\boldsymbol{\theta} | \boldsymbol{\lambda}) \prod_{n=1}^N \prod_{j=1}^J q(\theta_{nj} | \boldsymbol{\phi}_{nj}), \quad (32)$$

in which each hidden variable is independent and governed by its own parameters. The ELBO is a function of $\boldsymbol{\lambda}$ and $\boldsymbol{\phi}_n$, that govern the global variables and local variables in the n th context respectively. Again, we want to set the distributions $q(\boldsymbol{\theta} | \boldsymbol{\lambda})$ and $q(z_{nj} | \boldsymbol{\phi}_{nj})$ to be in the same exponential family as (24) and (25) with the $\boldsymbol{\lambda}$ and $\boldsymbol{\phi}_{nj}$ being the natural parameters

$$q(\boldsymbol{\theta} | \boldsymbol{\lambda}) = h(\boldsymbol{\theta}) \exp\left(\boldsymbol{\lambda}^\top t(\boldsymbol{\theta}) - a_g(\boldsymbol{\lambda})\right), \quad (33)$$

$$q(z_{nj} | \boldsymbol{\phi}_{nj}) = h(z_{nj}) \exp\left(\boldsymbol{\phi}_{nj}^\top t(z_{nj}) - a_\ell(\boldsymbol{\phi}_{nj})\right). \quad (34)$$

Using the defined variational distributions, we can define the ELBO as a function of $\boldsymbol{\lambda}$ as

$$\mathcal{L}(\boldsymbol{\lambda}) = \mathbb{E}_q[\eta_g(\mathbf{y}, z, \boldsymbol{\alpha})]^\top \nabla_\lambda a_g(\boldsymbol{\lambda}) - \boldsymbol{\lambda}^\top \nabla_\lambda a_g(\boldsymbol{\lambda}) + a_g(\boldsymbol{\lambda}) + \text{constant}. \quad (35)$$

By taking the gradient from the Equation (35), we can set it to zero by choosing

$$\boldsymbol{\lambda} = \mathbb{E}_q[\eta_g(\mathbf{y}, z, \boldsymbol{\alpha})] \quad (36)$$

and thus getting the update rule for the global variational parameters $\boldsymbol{\lambda}$. We can similarly derive the update rule for the local variational parameters $\boldsymbol{\phi}_{nj}$ to be

$$\boldsymbol{\phi}_{nj} = \mathbb{E}_q[\eta_\ell(y_n, z_{n,-j}, \boldsymbol{\theta})] \quad (37)$$

Algorithm 1 Coordinate Ascent Mean-field Variational Inference

Initialize random $\boldsymbol{\lambda}^{(0)}$

while ELBO has not converged **do**

for each $\boldsymbol{\phi}_{nj}$ **do**

$$\boldsymbol{\phi}_{nj}^{(t)} = \mathbb{E}_{q^{(t-1)}}[\eta_\ell(y_n, z_{n,-j}, \boldsymbol{\theta})].$$

\triangleright Update $\boldsymbol{\phi}_{nj}$

end for

$$\text{Update } \boldsymbol{\lambda}, \boldsymbol{\lambda}^{(t)} = \mathbb{E}_q[\eta_g(\mathbf{y}, z, \boldsymbol{\alpha})].$$

end while

Using the update rules (36) and (37), we can form the coordinate ascent mean-field variational inference (CAVI) algorithm that is shown in Algorithm 1 (Hoffman, Blei, et al., 2013). In essence, each iteration of CAVI first updates the local parameters and then the global parameters. The problem with the CAVI is that the first values of $\boldsymbol{\lambda}$ are initialized randomly, causing the algorithm to inefficiently go through all the data points with the random values. To solve this problem, Hoffman, Blei, et al. (2013) proposed the usage of stochastic optimization with the variational inference to efficiently update the global variational parameters $\boldsymbol{\lambda}$. Next, we will discuss how the stochasticity is introduced into the VI.

3.6 Stochastic Variational Inference

As discussed, stochastic variational inference is a modification of the CAVI, where we use stochastic optimization to make VI scalable and able to handle large-scale models and datasets. The goal in SVI is to sample a data point and repeat it N times and, with it, calculate an intermediate global parameter $\hat{\boldsymbol{\lambda}}_t$, which is then used in a stochastic gradient algorithm to compute the next $\boldsymbol{\lambda}$ (Hoffman, Blei, et al., 2013). To formulate SVI, we first define the stochastic gradient algorithm with the objective function $f(\boldsymbol{\lambda})$ and random function $B(\boldsymbol{\lambda})$ with and $\mathbb{E}_q[B(\boldsymbol{\lambda})] = \nabla_{\boldsymbol{\lambda}} f(\boldsymbol{\lambda})$ as

$$\boldsymbol{\lambda}^{(t)} = \boldsymbol{\lambda}^{(t-1)} + \rho_t b_t(\boldsymbol{\lambda}^{(t-1)}), \quad (38)$$

where b_t is an independent draw from B and ρ_t is an element of the step size sequence. If the following properties are satisfied:

$$\begin{aligned} \sum \rho_t &= \infty, \\ \sum \rho_t^2 &< \infty, \end{aligned} \quad (39)$$

then $\boldsymbol{\lambda}^{(t)}$ will converge to global optimum if f is convex and to local optimum if f is not convex. By utilizing the intermediate global parameter

$$\hat{\boldsymbol{\lambda}}_t = \boldsymbol{\alpha} + N \mathbb{E}_{\phi_t(\boldsymbol{\lambda})} [(t(y_i, z_i), 1)], \quad (40)$$

that in essence is the estimate of $\boldsymbol{\lambda}$, if the drawn sample is repeated N times, the Equation (38) can be formulated into the update rule for $\boldsymbol{\lambda}$ as

$$\boldsymbol{\lambda}^{(t)} = (1 - \rho_t) \boldsymbol{\lambda}^{(t-1)} + \rho_t \hat{\boldsymbol{\lambda}}_t. \quad (41)$$

By repeating this, we can find the variational parameters that maximize the ELBO. The final algorithm of the SVI can be seen in Algorithm 2. By setting the ρ_t to satisfy the conditions in Equation (39), SVI will converge to a local optimum of ELBO. (Hoffman, Blei, et al., 2013)

It is worth noting that VI methods usually underestimate the variance of the posterior. This

Algorithm 2 Stochastic Variational Inference

Initialize random $\boldsymbol{\lambda}^{(0)}$
 Set the step-size schedule ρ_t appropriately.
while forever **do**
 Sample a data point y_i
 Compute $\boldsymbol{\phi} = \mathbb{E}_{\lambda^{(t-1)}} [\boldsymbol{\eta}_g(y_i^{(N)}, z_i^{(N)})]$.
 Compute $\hat{\boldsymbol{\lambda}} = \mathbb{E}_{\boldsymbol{\phi}} [\boldsymbol{\eta}_g(y_i^{(N)}, z_i^{(N)})]$
 Update $\boldsymbol{\lambda}^{(t)} = (1 - \rho_t) \boldsymbol{\lambda}^{(t-1)} + \rho_t \hat{\boldsymbol{\lambda}}$.
end while

is caused when the variational distribution $q(\boldsymbol{\theta})$ is less than $p(\mathbf{y}, \boldsymbol{\theta})$, in which the VI does not reduce the KL divergence in these points, since the D_{KL} is already negative. The VI only reduces the D_{KL} if the $q(\boldsymbol{\theta})$ is larger than $p(\mathbf{y}, \boldsymbol{\theta})$ thus leading to the q having low variance. We can see this in the numerical examples in Chapter 4, where the reconstructions have quite narrow confidence intervals.

3.7 Modeling the measurement noise

As stated in Chapter 2, Equation (6), noise is present in the modeling process. In real-world use cases, it is unknown, and because of this, we want to incorporate the noise into our modeling process. With the SVI, we can run inference on the noise $\boldsymbol{\sigma}$ to make sure that the model does not fit into the noise in the data.

4 Numerical experiments

In this chapter, we evaluate the method of inversion with functional priors with BNNs in deconvolution. The BNN parameters will be trained using the SVI. The first section of this chapter introduces the deconvolution inverse problem and then we will go through the numerical examples. The experiments are implemented in Python utilizing PyTorch and Pyro libraries for the BNNs and for the training with SVI and MCMC.

4.1 Deconvolution

Convolution is a linear integral operation on two functions to produce a third function, where one of the functions is added to every point in the other function. *Deconvolution* is the inverse problem of convolution, where the aim is to use the noisy convolved measurement of \mathbf{y} to construct the unknown signal $\mathbf{x} : [-1, 1] \rightarrow \mathbb{R}$. The convolution has been done by applying a convolution kernel to the true \mathbf{x} . In this thesis, we use the Gaussian kernel

$$\mathcal{A}(t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}t^2\right). \quad (42)$$

Convolution is defined as a continuous integral

$$y(t) = (x * \mathcal{A})(t) = \int_{\mathbb{R}^n} \mathcal{A}(t - \tau)x(\tau)d\tau. \quad (43)$$

For practicality, the domain \mathbf{t} is discretized to N points $\mathbf{t} \in \mathbb{R}^N$. Thus $\mathbf{x} := (x(t_1), \dots, x(t_N))^T \in \mathbb{R}^N$ and similarly the data $\mathbf{y} \in \mathbb{R}^M$.

Assuming additive noise, the Equation (43) can be further written as a linear system of

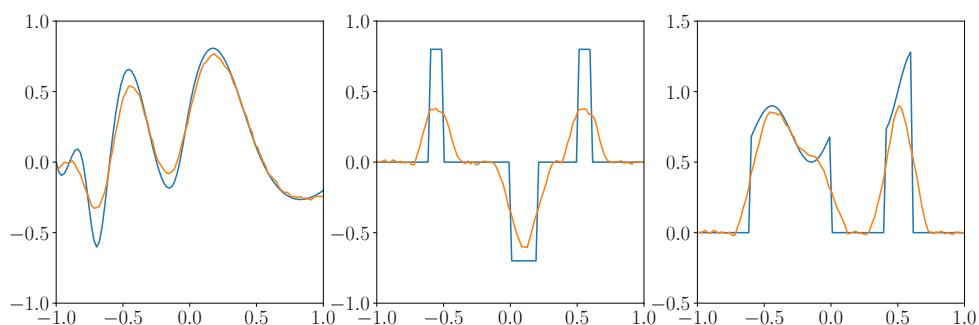


Figure 5: Initial deconvolution inverse problems with noise level $\sigma^2 = 0.01^2$. The true signal is in blue and the noisy measurements are in orange.

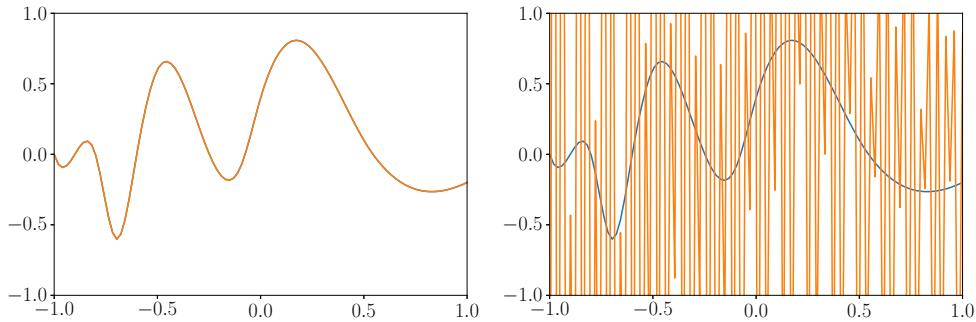


Figure 6: The effect of noise in direct inversion. The true signal is in blue, and the solution is in orange. On the left, there is no noise in the measurements, and on the right, the noise is additive Gaussian $\mathcal{N}(0, 0.01^2)$.

equations

$$\mathbf{y}^\varepsilon(t) = \mathcal{A}(t)\mathbf{x}(t) + \varepsilon \quad (44)$$

where $\mathcal{A}^{n \times m}$ and ε is the measurement noise. In this thesis, the noise is assumed to be zero-mean additive Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$.

It might be tempting to try to do the deconvolution by finding the direct inverse for the linear system of Equations (44) by multiplying the data \mathbf{y}^ε with the inverse of $\mathcal{A}(t)$. This method only works if the measurements do not contain any noise, that is, the amount of noise ε is zero. With $\varepsilon \neq 0$, there are an infinite number of possible solutions for $x(t)$ due to ill-posedness. The problem caused by the ill-posedness is illustrated in Figure 6. We can see that the noise causes the solution to become unstable. Because of this, it is crucial to limit the solution to not contain extreme values by utilizing regularization as discussed in Chapter 2.

4.2 Numerical experiments on deconvolution

In our numerical examples, we consider three deconvolution problems: a continuous function, a piecewise constant function, and a function containing both smooth and rough features. The convolution is done by using a Gaussian kernel described in the Equation (42). The signals are illustrated in Figure 5. The cases are designed in a way that the functions contain both positive and negative values and are approximately zero on the border.

Depending on the problem type, we use different prior distributions for the BNN parameters. For the continuous function, we utilize a fully Gaussian network, for the piecewise constant

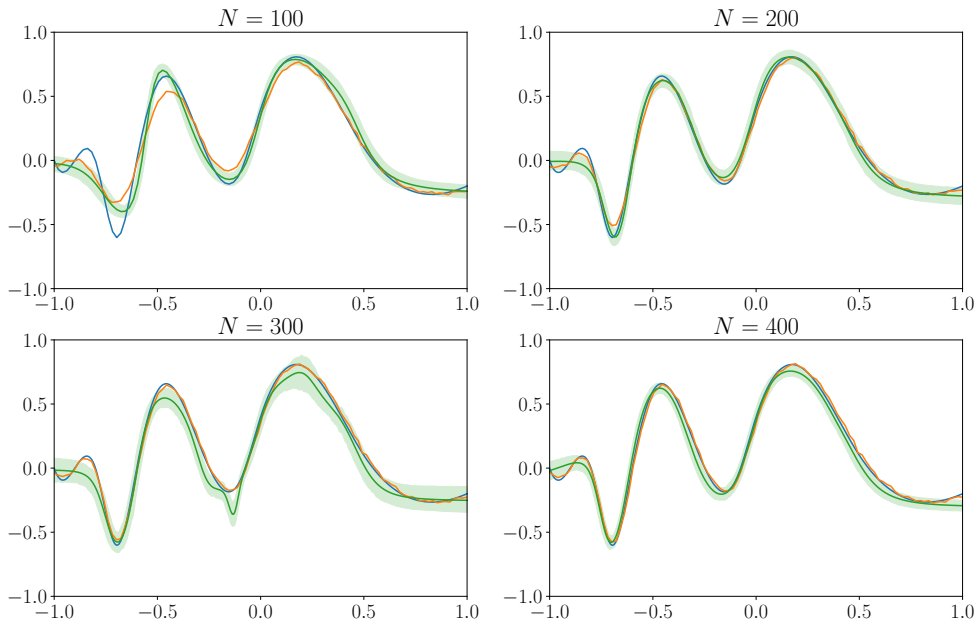


Figure 7: Reconstructions with different discretizations. The true continuous signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

function a fully Cauchy network and for the combined function a Cauchy-Gaussian network that contains both Gaussian and Cauchy layers. All the networks had four layers with layer sizes $n_\ell \in \{40, 80, 40, 80\}$.

We evaluate different discretizations for the domain t introduced in Section 3.1 and the number of SVI iterations in the BNN training process. Since real-world use cases require methods to be reliable in various conditions, we address the performance of the prior with different noise levels σ by studying how sensitive it is to the noise. Lastly, we compare the performance between the different inference methods: HMC with NUTS sampler and the proposed alternative SVI.

4.3 Increasing the number of discretization points

From Figure 7 we can see, that increasing the number of discretization points N made the continuous signal reconstruction clearly better. With 400 discretization points, the true signal is within the confidence interval at every point, whereas with 100 points, the first peaks and valleys are not captured by the model.

As shown in Figure 8, the discretization steps had a more drastic effect on the solution. We can see that with $N = 100$ and $N = 400$, the solution has some clear errors. This is likely due

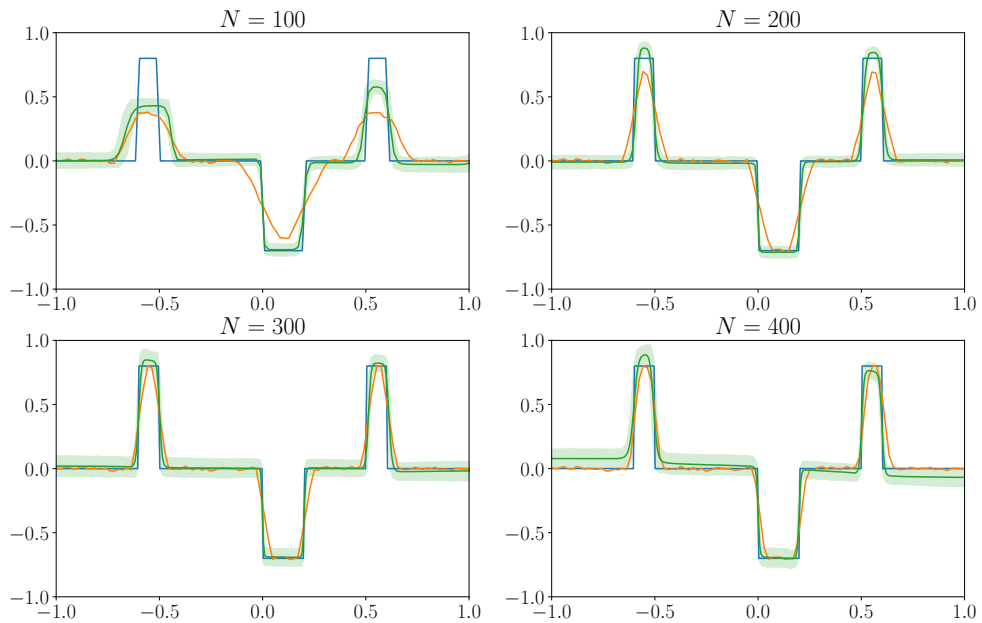


Figure 8: Reconstructions with different discretizations. The true discrete signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

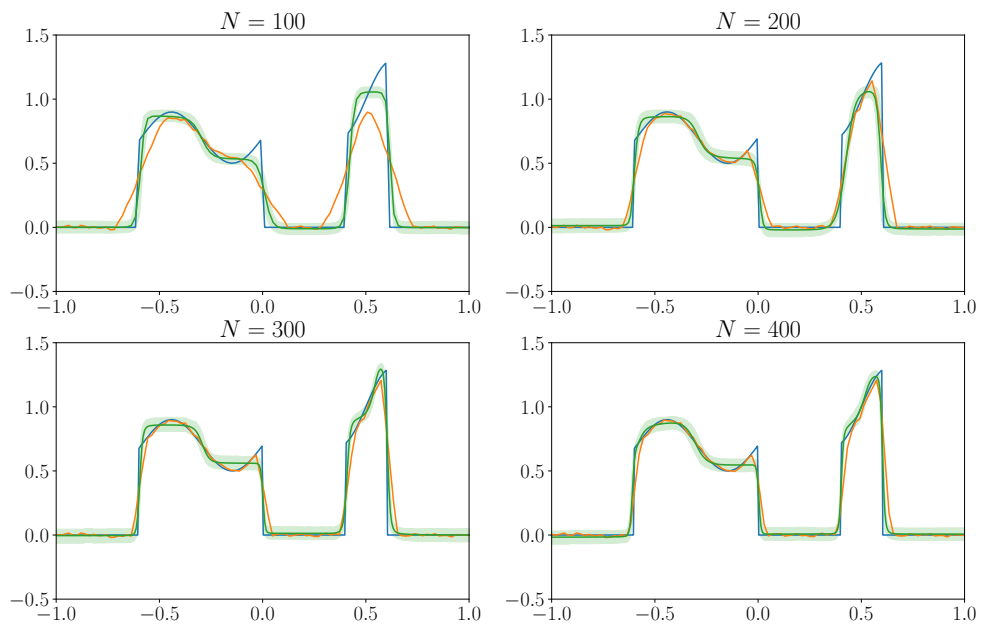


Figure 9: Reconstructions with different discretizations. The true combined signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

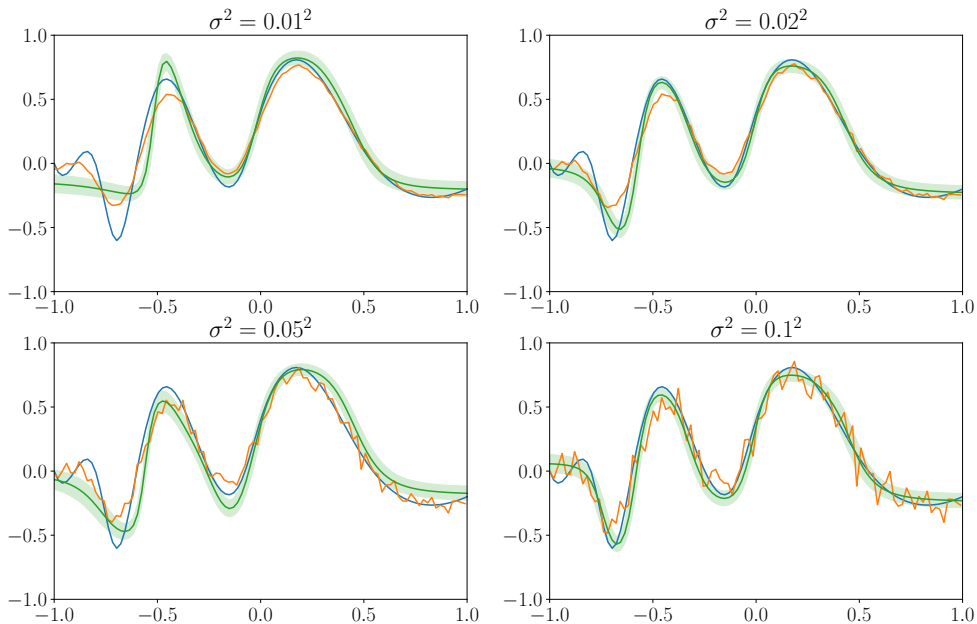


Figure 10: Reconstructions with different amounts of measurement noise and continuous signal. The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

to the fact that the SVI had not yet converged to the optimal variational distribution.

More discretization points also help with the Cauchy-Gaussian BNN and combined target function, as shown in Figure 9. We can see that with a more sparse discretization, the model was not able to capture the shape of the second spike correctly, but with $N = 300$ and $N = 400$ the second spike is fairly accurate. It is worth noting that in the two last cases, the reconstruction of the first part of the function has piecewise constant values.

4.4 Noise in the measurements

We evaluated the performance of the model with different levels of noise with standard deviations of $\sigma \in \{0.01, 0.02, 0.05, 0.1\}$. The results can be seen in Figures 10, 11 and 12. Surprisingly, increasing the noise did not make the reconstructions unstable. This might be because the noise helps the SVI to find the variational parameters better by making larger changes in the parameter updating process that help avoid local minima. However, we can see that with the combined signal, the noise affected the reconstruction, making it worse within the first part of the signal.

We can also notice the same phenomenon as with the discretization experiments in Figure 8, that the discrete signal does not always have time to fully update the parameters to get the

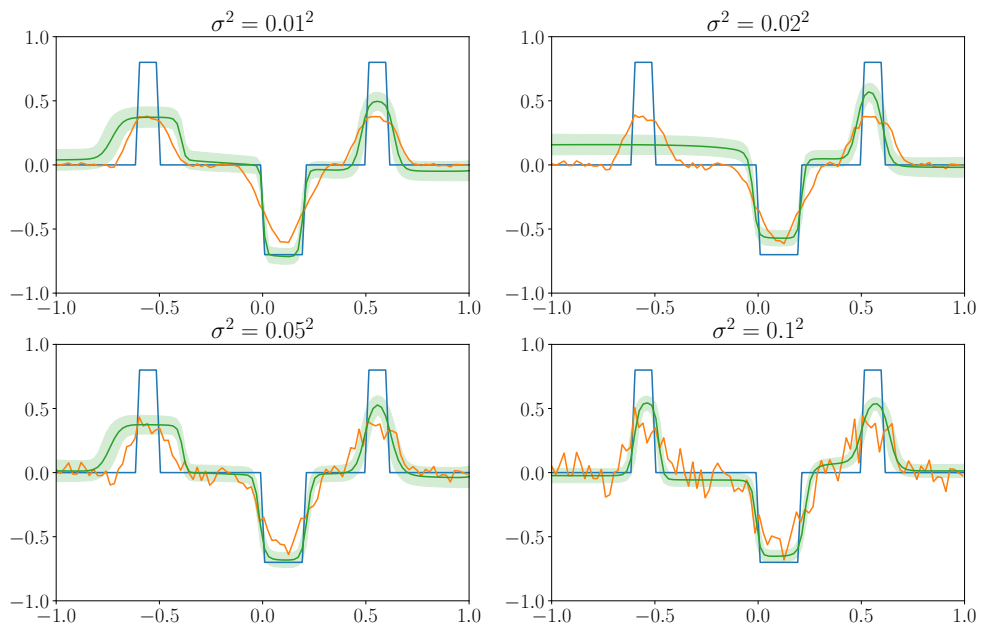


Figure 11: Reconstructions with different amounts of measurement noise and discrete signal. The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

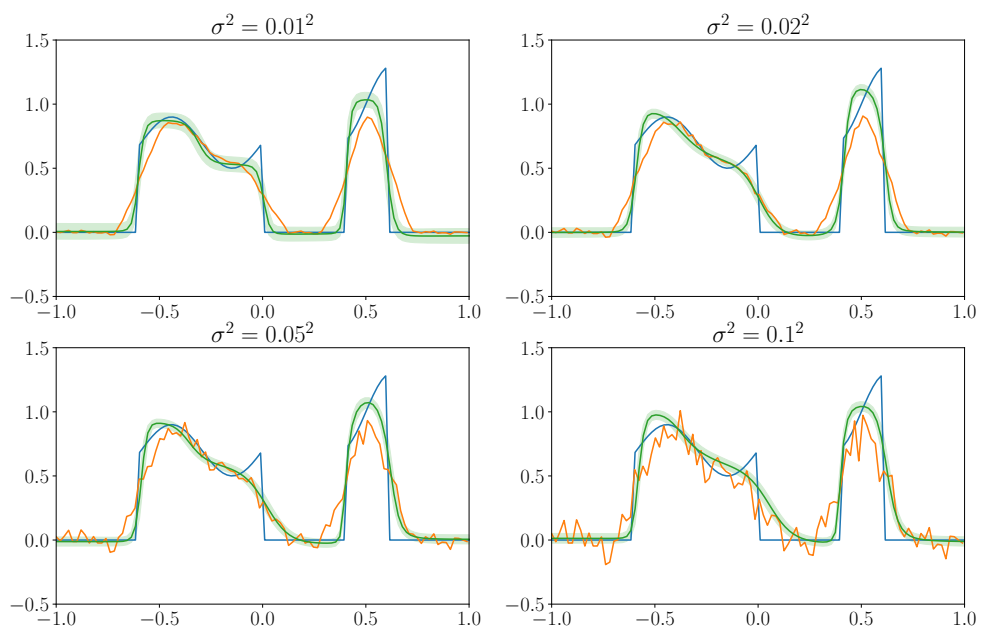


Figure 12: Reconstructions with different amounts of measurement noise and a signal with discrete and continuous parts. The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

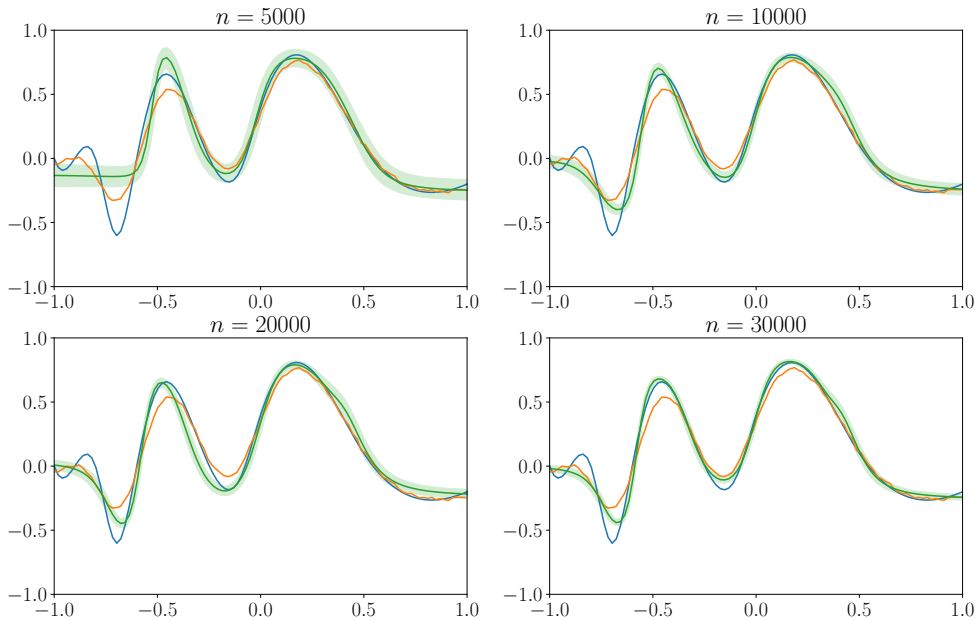


Figure 13: Posteriors with increasing number of SVI training iterations for continuous signal. The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

full reconstruction. Instead, the first step in the function is not reconstructed.

4.5 SVI training iterations

As discussed in Section 3.5, SVI is a gradient-based optimization method and thus is dependent on the number of iterations the algorithm is run. We evaluated the reconstructions across multiple values of the parameter n , using $n \in \{5000, 10000, 20000, 30000\}$.

From Figure 13, we can conclude that with 5000 iterations, the model has not yet converged since increasing the number of iterations improved the accuracy of the continuous function reconstruction. However even though the MAP estimate was more accurate, running the SVI longer made the confidence intervals narrower. The SVI already struggles with the confidence intervals as discussed in Section 3.5, and because of the true \mathbf{x} not being inside the confidence interval, we could say that the iterations make the model confidence worse. For the piecewise constant function, the reconstruction benefitted from more iterations, as seen in Figure 14. Interestingly the confidence intervals were narrower with 20000 iterations than with 30000 iterations.

As seen in the experiments with the noise level and number of discretization steps, the reconstruction of the combined signal is the most difficult one of the three signals. From Figure

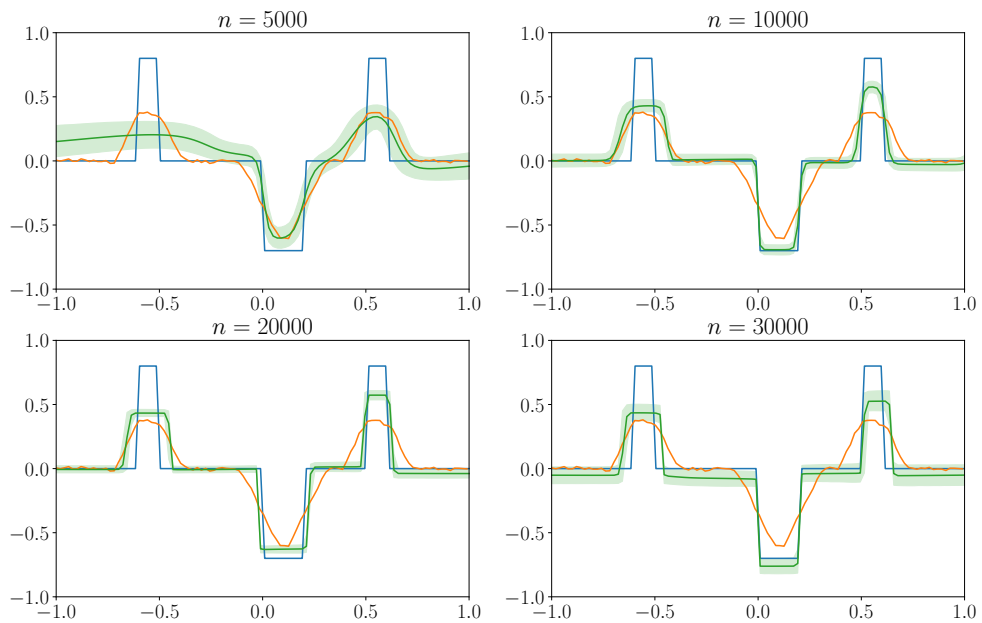


Figure 14: Posteriors with increasing number of SVI training iterations for discrete signal. The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

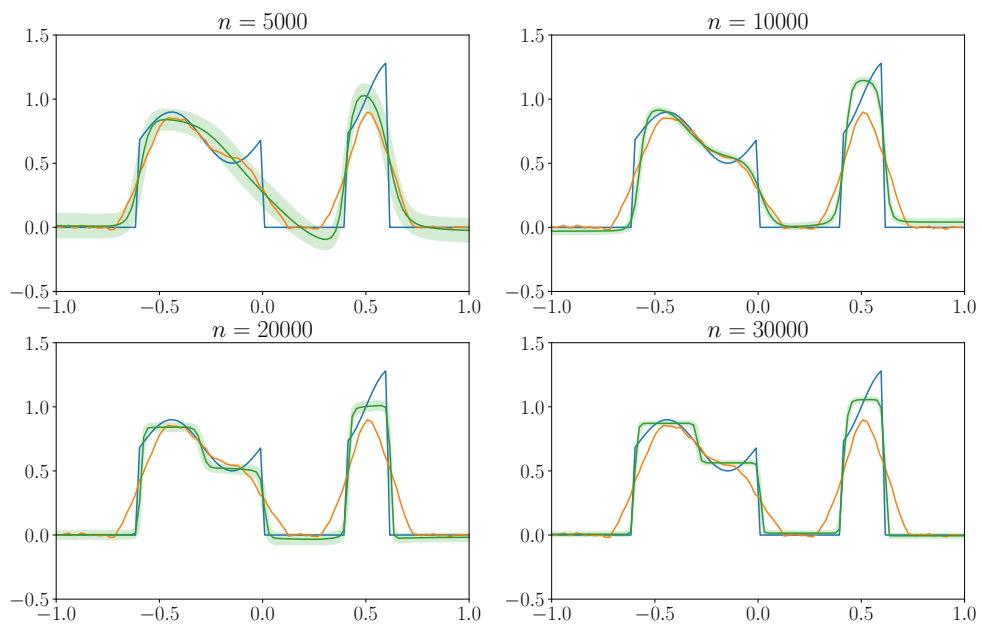


Figure 15: Posteriors with increasing number of SVI training iterations for combined signal. The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

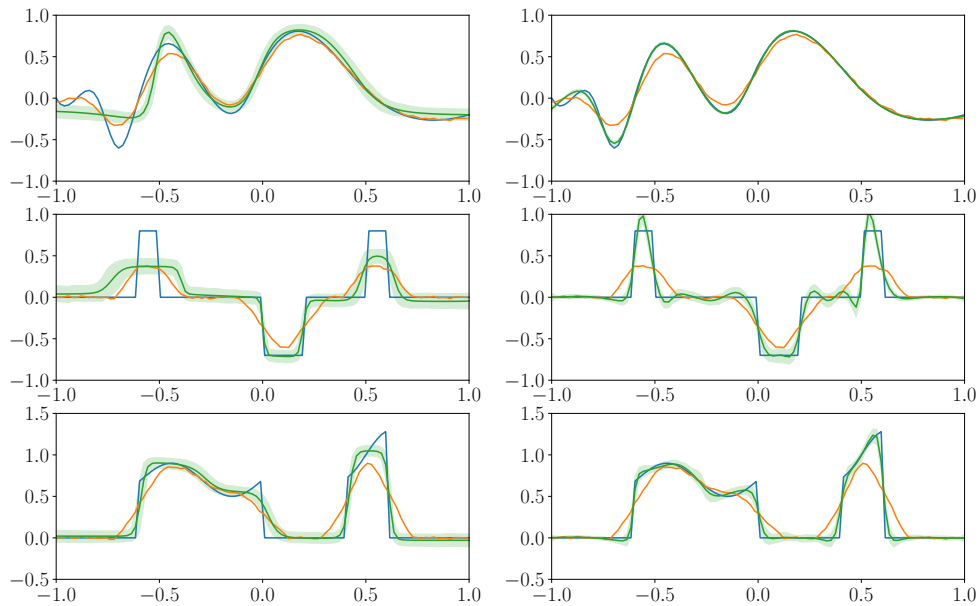


Figure 16: Comparison between SVI (left) and HMC with NUTS sampler (right). The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

15, we see that the first two reconstructions oversmooth the solution. The last two realizations are able to detect the sharp edges but fail to capture the shape of the continuous parts by making the reconstruction fairly piecewise constant.

4.6 Comparison between SVI and MCMC

Lastly, we address the differences between the SVI and HMC NUTS inference methods in Figure 16. We can see that for the continuous function, MCMC was able to create an almost flawless reconstruction, while the SVI was not able to correctly reconstruct the first part of the signal.

For the piecewise constant signal, SVI was able to correctly identify the shape of the signal. MCMC produced some fluctuations in the constant parts of the solution.

For the signal with both flat and continuous parts, MCMC was more accurate. It reproduced the shape of both continuous parts really well, whereas the SVI was not able to do that for the second part of the signal. It is worth noting the same phenomenon as with the piecewise constant signal that the SVI method was able to model flat parts of the signal better than MCMC. In both cases, MCMC produces fluctuations in the solution.

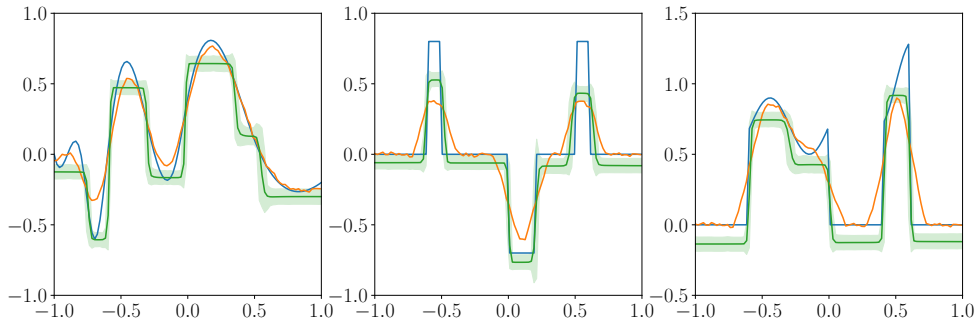


Figure 17: Gaussian layers with high σ on the parameter priors. The true signal is in blue, noisy measurements in orange, and the inverse problem solution with 95% confidence interval in green.

When comparing the two methods, it is also necessary to compare the time it took to obtain the solution. In the examples in Figure 16, the inference with SVI took only 2% of the time the MCMC needed for the sampling. Because of this, SVI would be more suitable for applications where the inverse problem needs to be solved with time constraints.

4.7 Gaussian parameters with high prior standard deviation

During the experiments, we found that by utilizing a Gaussian network with high standard deviations on the parameter priors, the network produces piecewise constant reconstructions for deconvolution. An example of this is shown in Figure 17. We can see that for the discrete signal, the shape of the reconstruction is fairly good, but the value of the whole signal is lower than it should be. It is interesting to also see that the network produces piecewise constant reconstructions even for the continuous signal.

It is not clear, what causes this behavior, but it might be because of the activation function $\tanh(\cdot)$ being asymptotically limited to -1 and 1 , which would then lead to discrete values in the nodes. This is an interesting property that could be usable in inverse problems where both continuous and sharp features are present.

5 Conclusions

In this thesis, we studied the usage of Bayesian neural networks as a function prior for Bayesian inverse problems. In Chapter 2, we introduced inverse problems and addressed the importance of a prior in Bayesian inversion.

In Chapter 3, we defined the Bayesian neural networks and discussed how they can be used in inversion. This has been studied before by Li, Dunlop & Stadler (2022), Suuronen et al. (2023) and Senchukova (2024). In these studies, the inference on the BNN was done using Markov chain Monte Carlo methods. Because of the scalability limitations of the MCMC methods, in this thesis the inference for the BNN was done using the stochastic variational inference method (Hoffman, Blei, et al., 2013), which has been proven to provide performance boosts on large-scale problems by transforming the posterior sampling problem into an optimization problem.

In the numerical examples in Chapter 4, we demonstrated the neural network priors with SVI by solving deconvolution for three different one-dimensional signals convolved with a Gaussian kernel: a continuous, a piecewise constant and a signal with both continuous and piecewise constant parts. We found that increasing the number of discretization points improves the quality of the reconstructions. Increasing the number of iterations, the SVI was run also had a positive impact on the reconstructions. This was expected since SVI is an iterative optimization algorithm that benefits from more steps.

Surprisingly, we found out that the effect of measurement noise was quite minimal. The model was able to create fairly good results even with 10% of noise.

Lastly, we compared the two methods: SVI and MCMC, for doing the inference on the BNN. We found out, that MCMC produced more accurate solutions for all the different signals. Especially it performed really well with the continuous and combined signals. However, MCMC had some trouble with the constant parts of the signals. The reconstruction of the piecewise constant signal had some fluctuations. Also, it is worth noting, that doing the inference with MCMC algorithms is computationally much slower. SVI, on the other hand, performed well with the continuous parts of the signals and was able to more correctly identify the shape of the piecewise constant signal.

In further work, the proposed method of this thesis could be extended to two-dimensional applications, in which we could greatly benefit from the computational efficiency of SVI over traditional MCMC methods due to having typically larger data and parameter space. The method could also be applied to some real-world applications.

In our experiments, we also noticed that a BNN is able to produce piecewise constant reconstructions with fully Gaussian layers given a large enough standard deviation for the parameter priors. For the discrete signal, this is a desired property. However, the high prior σ resulted in sharp reconstructions even with the true function being a continuous function, which is problematic. It should be studied, why the high standard deviation leads to sharp realizations since this property could allow the usage of only Gaussian layers to provide flexible priors.

References

- David M. Blei, A. K. & McAuliffe, J. D. (2017). Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association* 112(518), pp. 859–877. DOI: 10.1080/01621459.2017.1285773.
- Hadamard, J. (1923). *Lectures on Cauchy's Problem in Linear Partial Differential Equations*. Dover phoenix editions. Dover Publications. ISBN: 9780486495491.
- Härkönen, T., Vartiainen, E. M., Lensu, L., Moores, M. T. & Roininen, L. (2024). Log-Gaussian gamma processes for training Bayesian neural networks in Raman and CARS spectroscopies. *Phys. Chem. Chem. Phys.* 26 (4), pp. 3389–3399. DOI: 10.1039/D3CP04960D.
- Hoffman, M. D., Blei, D. M., Wang, C. & Paisley, J. (2013). Stochastic variational inference. *J. Mach. Learn. Res.* 14(1), pp. 1303–1347. ISSN: 1532-4435.
- Hoffman, M. D. & Gelman, A. (2014). The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research* 15(47), pp. 1593–1623.
- Holden, M., Pereyra, M. & Zygalakis, K. C. (2021). *Bayesian Imaging With Data-Driven Priors Encoded by Neural Networks: Theory, Methods, and Algorithms*. arXiv: 2103.10182 [stat.ME].
- Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Židek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S., Ballard, A., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J. & Hassabis, D. (July 2021). Highly accurate protein structure prediction with AlphaFold. *Nature* 596, pp. 1–11. DOI: 10.1038/s41586-021-03819-2.
- Kaipio, J. P. & Somersalo, E. (2005). *Statistical and Computational Inverse Problems*. eng. Applied mathematical sciences (Springer-Verlag New York Inc.) ; 160. New York, NY: Springer Science+Business Media, Inc. ISBN: 9780387271323. DOI: <https://doi.org/10.1007/b138659>.
- Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V. & Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. *Computational and Structural Biotechnology Journal* 13, pp. 8–17. ISSN: 2001-0370. DOI: <https://doi.org/10.1016/j.csbj.2014.11.005>.
- Lassas, M., Saksman, E. & Siltanen, S. (2009). *Discretization-invariant Bayesian inversion and Besov space priors*. DOI: 10.3934/ipi.2009.3.87.
- Lassas, M. & Siltanen, S. (Aug. 2004). Can one use total variation prior for edge-preserving Bayesian inversion? *Inverse Problems* 20(5), p. 1537. DOI: 10.1088/0266-5611/20/5/013.
- LeCun, Y., Bengio, Y. & Hinton, G. (May 2015). Deep Learning. *Nature* 521, pp. 436–44. DOI: 10.1038/nature14539.

- Li, C., Dunlop, M. & Stadler, G. (2022). *Bayesian neural network priors for edge-preserving inversion*. DOI: 10.3934/ipi.2022022.
- Markkanen, M., Roininen, L., Huttunen, J. M. & Lasanen, S. (2019). Cauchy difference priors for edge-preserving Bayesian inversion. *Journal of Inverse and Ill-posed Problems* 27(2), pp. 225–240. DOI: doi:10.1515/jiip-2017-0048. URL: <https://doi.org/10.1515/jiip-2017-0048>.
- Marschall, M., Wübbeler, G., Schmähling, F. & Elster, C. (July 2023). Machine learning based priors for Bayesian inversion in MR imaging. *Metrologia* 60(4), p. 044003. DOI: 10.1088/1681-7575/ace3c2.
- Miller, E. & Karl, W. (2003). *Fundamentals of Inverse Problems*.
- Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. eng. 1st ed. 1996. Lecture Notes in Statistics, 118. New York, NY: Springer New York. ISBN: 1-4612-0745-2.
- Roininen, L., Huttunen, J. M. J. & Lasanen, S. (2014). *Whittle-Matérn priors for Bayesian statistical inversion with applications in electrical impedance tomography*. DOI: 10.3934/ipi.2014.8.561.
- Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review* 65 6, pp. 386–408.
- Senchukova, A. (2024). *Flexible priors for rough feature reconstruction in Bayesian inversion*. eng.
- Siltanen, S., Kolehmainen, V., Järvenpää, S., Kaipio, J. P., Koistinen, P., Lassas, M., Pirttilä, J. & Somersalo, E. (2003). Statistical inversion for medical x-ray tomography with few radiographs: I. General theory. *Physics in Medicine and Biology* 48(10), pp. 1437–1463. DOI: 10.1088/0031-9155/48/10/314.
- Springer, S., Glielmo, A., Senchukova, A., Kauppi, T., Suuronen, J., Roininen, L., Haario, H. & Hauptmann, A. (2023). *Reconstruction and segmentation from sparse sequential X-ray measurements of wood logs*. DOI: 10.3934/ammc.2023002.
- Suuronen, J. (2023). *On numerical implementation of α -stable priors in Bayesian inversion*. eng.
- Suuronen, J., Soto, T., Chada, N. K. & Roininen, L. (Aug. 2023). Bayesian inversion with α -stable priors. *Inverse Problems* 39(10), p. 105007. DOI: 10.1088/1361-6420/acf154.
- Tierney, L. (1994). Markov Chains for Exploring Posterior Distributions. *The Annals of Statistics* 22(4), pp. 1701–1728. DOI: 10.1214/aos/1176325750.
- Uribe, F., Papaioannou, I., Betz, W. & Straub, D. (2020). Bayesian inference of random fields represented with the Karhunen–Loève expansion. *Computer Methods in Applied Mechanics and Engineering* 358, p. 112632. ISSN: 0045-7825. DOI: <https://doi.org/10.1016/j.cma.2019.112632>.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2017). Attention is All you Need. In: *Advances in Neural Information*

Processing Systems. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan & R. Garnett. Vol. 30. Curran Associates, Inc.

Vauhkonen, M., Vadász, D., Karjalainen, P., Somersalo, E. & Kaipio, J. (May 1998). Tikhonov regularization and prior information in electrical impedance tomography. *IEEE transactions on medical imaging* 17, pp. 285–93. DOI: 10.1109/42.700740.