Constrained Bayesian Neural Networks

Citation

Permanent link
https://nrs.harvard.edu/URN-3:HUL.INSTREPOS:37364639

Terms of Use
This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA

Share Your Story
The Harvard community has made this article openly available. Please share how this access benefits you. Submit a story.

Accessibility
Constrained Bayesian Neural Networks

A thesis presented by

Lars Lorch

To

Applied Mathematics

in partial fulfilment of the honors requirement

for the degree of Bachelor of Arts

John A. Paulson School of Engineering and Applied Sciences

Harvard College

Cambridge, Massachusetts

March 29, 2019
Abstract

Neural networks are central to many of the recent empirical breakthroughs in machine learning, but their inability to model the uncertainty of predictions makes them inadequate for safety-critical domains. Bayesian neural networks (BNNs) extend standard neural networks by modeling probability distributions over predictions, allowing us to judge how confident we should be in a given prediction. However, even though BNNs define a theoretical framework of incorporating prior beliefs about the parameters, they are unable to encode interpretable prior knowledge in function space, where most experts have prior domain knowledge.

In this thesis, I present a rigorous and interpretable approach of imposing prior constraints in the input-output space onto the distributions modeled by Bayesian neural networks. By formulating a general constraint prior, the novel method can be applied to arbitrary inequality constraints and treated as a black box with any inference technique normally used with BNNs. Extensive evaluations show qualitatively and quantitatively that Constrained Bayesian neural networks are able to successfully incorporate complex and yet interpretable constraints onto the functions they model. Furthermore, Constrained BNNs do not affect any objectives or advantages of inference methods negatively and can even guide mean-field variational inference approaches to higher unconstrained ELBO values than standard BNNs on average. Finally, novel multimodal posterior predictive distributions are shown in special constraint cases. A new formulation of variational inference with a general Gaussian mixture variational family is derived to obtain these results and compared to a state-of-the-art sampling method.
Acknowledgements

First and foremost, I want to thank Finale Doshi-Velez for being a very encouraging and supportive thesis advisor. Without her inspiration, constructive criticism, and shared wisdom, this thesis could not have been possible, and many of the accomplishments are due to her vision, guidance, and advice. I feel incredibly lucky to have had the chance of spending a whole year working with her. I also want to thank my initial project partner Moritz Graule who has actively contributed a great amount when collaborating on the earlier theoretical foundations of this thesis, especially the constraint prior. In addition, I thank Wanqian Yang for the insightful brainstorming sessions and Jiayu Yao, Melanie F. Pradier, Weiwei Pan, Jazz Zhao, Srivatsan Srinivasan, and Anirudh Suresh for their helpful discussions and suggestions in the effort of turning these ideas and more into a greater research contribution.
## Contents

1 Introduction .................................................. 5

2 Bayesian Neural Networks ................................. 8
   2.1 Neural Networks ......................................... 8
      2.1.1 Model ............................................ 9
      2.1.2 Training and inference ............................ 10
   2.2 Bayesian Approach .................................... 10
      2.2.1 Model ............................................ 11
      2.2.2 Prediction with the posterior predictive distribution ... 12
      2.2.3 Inference methods ................................. 14

3 Constrained Bayesian Neural Networks ................... 18
   3.1 Constraints and Constraint Function ................. 20
      3.1.1 Defining the infeasible set $\mathcal{S}$ ........... 20
      3.1.2 Guiding constraint function ...................... 21
   3.2 The Constraint Prior .................................. 24
   3.3 Effects on Inference: Bayes by Backprop ............ 25
      3.3.1 Relationship to constrained optimization ........ 26
      3.3.2 Other related work .............................. 28

4 Experiments and Evaluation .............................. 29
   4.1 Examples and Use Cases ............................... 33
      4.1.1 Linear and nonlinear bounds ...................... 33
<table>
<thead>
<tr>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.2 Enforcing asymptotic behavior only</td>
</tr>
<tr>
<td>4.1.3 Equality constraints</td>
</tr>
<tr>
<td>4.2 Ablation Experiments</td>
</tr>
<tr>
<td>4.2.1 Effect of the constraint enforcement parameter $\gamma$</td>
</tr>
<tr>
<td>4.2.2 Effect of parameterizations of $\Psi_{\tau_0,\tau}$</td>
</tr>
<tr>
<td>4.3 Additional Considerations</td>
</tr>
<tr>
<td>5 Advanced Posterior Inference and Results</td>
</tr>
<tr>
<td>5.1 Advanced Inference Methods</td>
</tr>
<tr>
<td>5.1.1 Variational inference with mixture of Gaussians using Gumbel-Softmax</td>
</tr>
<tr>
<td>5.1.2 Generalized Darting MCMC</td>
</tr>
<tr>
<td>5.2 Splitting Posterior Predictives</td>
</tr>
<tr>
<td>5.2.1 Infeasible sets with separated feasible outputs</td>
</tr>
<tr>
<td>5.2.2 Arbitrary and multiple constrained regions</td>
</tr>
<tr>
<td>6 Conclusion</td>
</tr>
<tr>
<td>Bibliography</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Machine learning and artificial intelligence have advanced technology more than any other field over the last decade. Researchers built systems that reach superhuman performance at complex strategy games such Chess, Go, or StarCraft II, often without any human training examples [1]. Novel language representation models are able to achieve state-of-the-art results for a wide range of natural language understanding tasks [2], and improved deep learning techniques help surpass the best benchmarks in image recognition [3].

Neural networks are the constant element in all of these and more advances in pattern recognition, machine learning, and artificial intelligence. Recent progress in hardware and in particular fast parallel computation have enabled the stacking and optimization of more and deeper consecutive layers of these complex function approximators. Hence, the field is now often referred to as deep learning [4, 5].

However, higher levels of care are necessary once neural networks are applied in safety critical domains such as drones and robotics, self-driving cars, health care, and voice command recognition [6]. Then, models need to live up to sensitive requirements by not only giving accurate predictions, but also by giving a sense of uncertainty on predictions, since humans could suffer extreme damage as a result of prediction errors in these domains. Standard neural networks themselves are not robust and interpretable enough by any means. They have been shown
to exhibit extremely unnatural and unpredictable behavior in various cases. In
so-called adversarial attacks, small and imperceptible perturbations in the inputs
were shown to lead to extreme misclassifications by neural networks [6, 7].

For deep learning to become applicable to direct interactions with humans,
predictions must come with a sense of uncertainty and multimodality. The dis-
tribution of confidence across predictions as well as the existence of different and
equally likely predictions should be taken into consideration. In case of high un-
certainty, predictions ought to express low confidence rather than proceeding with
unreasonable accurateness, suffering potentially harsh consequences.

Bayesian deep learning and Bayesian neural networks (BNNs) present a prin-
cipled approach to modeling problems probabilistically while retaining most of the
advantages of standard neural networks. Given an observed data set, BNNs define
a probability distribution over predictions as opposed to the point estimate of a
standard neural network. This class of models gives scientists not only a sense
of uncertainty and spread of possible predictions but is also expressive enough to
model multiple significant modes in the predictive distribution. In cases where
prediction is uncertain, human experts could thus intervene when making an ul-
timate decision call, often equipped with prior experience, expert knowledge, and
opinions by peers.

In addition to more principled predictions, BNNs enable the incorporation of
prior beliefs into the structure of the neural network. Beliefs derived from prior
knowledge can be used to select prior distributions over parameters, guiding pre-
dictions in more likely directions. In the same way experts might take over when
uncertainty is high, there are many domains in which prior expert knowledge is
available from past experiments and other branches of science. Especially the
above-mentioned safety critical domains have inherent constraints that are known
a priori and important to the model in many cases. For instance, the diagnosis
of patients in the medical sector can be assisted by machine learning systems.
In addition to the requirements of uncertainties on predictions, the model could include the vast corpus of expert knowledge available to physicians, which would be able to exclude many diagnoses completely under specific circumstances of the patient. However, the prior beliefs imposed by the BNN are in parameter space rather than function space. Unfortunately, the parameters of neural networks are vast, extremely uninterpretable, and as a result there is no way for experts to know how to encode prior beliefs into these expressive models.

This thesis introduces a principled and intuitive approach of imposing prior constraints in function space onto the distributions modeled by BNNs, called Constrained Bayesian neural networks. By formulating a prior that encodes arbitrary constraints in the input-output space, the concept can be treated as a black box with common Bayesian inference techniques. Competently incorporating expert knowledge has the potential to not only enforce feasibility in a given domain but to also make the predictive behavior of BNNs on unfamiliar or qualitatively new samples more accurate. The expert constraints help when predicting far away from prior observations, where the BNN needs to extrapolate rather than interpolate, and where standard neural networks often fail in unreasonable ways [6]. Furthermore, to enable qualitatively accurate and sufficient inference results, a new variational inference approach using Gaussian mixture distributions is derived and juxtaposed with a state-of-the-art sampling method to suit the application with Constrained Bayesian neural networks.

To provide the reader with a formal understanding of this new method and contribution as well as the inference techniques, neural networks and their Bayesian extension are formally introduced in chapter 2. Then, the core of the thesis with the novel constraint function, constraint prior, and its effects on inference are derived in chapter 3. The introduced ideas are tested and evaluated in synthetic experiments in chapter 4 and the novel and more complex inference methods are ultimately derived and discussed in relation to Constrained BNNs in chapter 5.
Chapter 2

Bayesian Neural Networks

As explained in the introduction, it is essential to lay out the mathematical model of a Bayesian neural network before presenting the new concepts for constraints on Bayesian neural networks at the core of this thesis. The neural network model and common concepts of statistical inference are fundamental to the reader for understanding later contributions in variational inference as well as the advantages and limitations of BNNs, in particular their quantification of uncertainty and the inherent limitations of interpretability in neural networks.

2.1 Neural Networks

An artificial neural network is a function $\phi(x; W)$ of an input $x = [x_1, \ldots, x_{D_x}]^T \in \mathbb{R}^{D_x}$ parameterized by a set of parameters $W$. Most often, the goal of a neural network is to model some ground truth function $f(x) \in \mathbb{R}^{D_y}$ for which there exists a collection of $N$ data points $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$ that we observed. This setup is referred to as supervised learning. Normally, the assumption is made that the data was observed under normally distributed and independent noise with variance $\sigma^2$ such that $y \sim \mathcal{N}(f(x), \sigma^2 I)$. 
2.1.1 Model

The simplest neural network is the fully connected feedforward neural network, also referred to as multilayer perceptron, and will be the only type of neural network discussed in this thesis. Despite its simplicity, multilayer perceptrons have been shown to be universal function approximators to arbitrary degree of accuracy even with only one hidden layer [8]. Even though a neural network is simply a parameterized function $\phi(x; W)$, its structure can be visualized as a network of layers that can be thought of as roughly resembling the web of neurons in the brain, hence the name “neural” network. See Figure 2.1.

A multilayer perceptron is composed of $L$ hidden layers $h^{(1)}, \ldots, h^{(L)}$ and an output layer $y$. Starting with the input $x$, the values of the hidden and output layers are *forward propagated* through the network using $L + 1$ sets of weights $W = \{W^{(l)}, b^{(l)}\}_{l=0}^L$. Setting $x = h^{(0)}$, the *forward pass* is computed recursively as follows:

\[
\begin{align*}
    z^{(l)} &= W^{(l-1)} h^{(l-1)} + b^{(l-1)} \\
    h^{(l)} &= \text{squash}(z^{(l)})
\end{align*}
\]

for all $1, \ldots, L + 1$, where $y = z^{(L+1)}$. The squash(.) function is called *activation*

![Figure 2.1: Visualization of a fully connected feedforward neural network with $L$ hidden layers. The input $x$ is forward propagated under a series of nonlinear operations to yield the output $y$, here a scalar. Every node represents a scalar $h_i$ of the vector $h$ and every arrow from $h^{(l-1)}$ to $h^{(l)}$ corresponds to the element $w_{ji}$ of $W^{(l-1)}$ in the linear transformation of Eq. 2.1.](image)

9
function and is applied element-wise to all hidden layers. It must be nonlinear for
the neural network $\phi(x; W)$ to be expressive enough to model nonlinear functions
$f(x)$. Popular choices for squash$(.)$ are ReLU$\left(z\right) = \max(z, 0)$ (rectified linear
unit), the sigmoid $\sigma(z) = \frac{1}{1+\exp(-z)}$, tanh$(x)$, or the radial basis function $\text{rbf}(z) = \exp(z^2)$.

### 2.1.2 Training and inference

The goal is to find parameters $W = \{W^{(l)}, b^{(l)}\}_{l=0}^L$ such that $\phi(x; W) \approx f(x)$.
Since this thesis is about Bayesian neural networks, I will not go into depth about
how to find these parameters for standard neural networks or the process of back-
propagation [9]. However, it is important from a probabilistic perspective to point
out that $W$ is generally found via maximum likelihood estimation (MLE) in stan-
dard neural networks, i.e. by finding parameters that maximize the likelihood of
the observed data $D$ given the model $W$.

$$
\max_W p(D|W) = \max_W \sum_{i=1}^N p(y_i|x_i; W) = \max_W \sum_{i=1}^N \mathcal{N}(y_i; \phi(x_i; W), \sigma^2 I)
$$

The likelihood $p(y|x; W) = \mathcal{N}(y; \phi(x; W), \sigma^2 I)$ follows from the assumption of
Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ in the observations. This estimation reduces $W$ to
a point estimate, a central difference to Bayesian neural networks.

### 2.2 Bayesian Approach

There are two fundamental differences between standard neural networks and the
Bayesian approach of Bayesian neural networks, both of which are discussed in
detail below, one of which the novel method of this thesis effectively addresses.

First, contrary to MLE parameter estimation in standard neural networks de-
scribed above, prediction in BNNs is done using proper statistical inference, which
in itself poses a whole set of difficulties. In BNNs, prediction is made via the poste-
rior predictive distribution, which yields a probability distribution over predictions rather than a single prediction, giving a sense of uncertainty and multimodality of our prediction. The posterior predictive is obtained by integrating predictions of the model with respect to the posterior distribution obtained by updating the prior \[10\].

Secondly, in BNNs and the Bayesian modeling approach in general, beliefs derived from prior knowledge are used to select prior distributions over the parameters. Using Bayes’ theorem, these prior beliefs are updated to yield the posterior of the parameters given the observed data \[10\]. Unfortunately, individual weights in neural networks do not have an interpretable or intuitive meaning and are as such unsuitable for directly encoding prior beliefs. This thesis will deliver an interpretable way of incorporating prior beliefs and expert knowledge into the functions modeled by BNNs.

### 2.2.1 Model

The computation of the neural network outputs via forward propagation described in Eq. 2.1 as well as the fully connected architecture remain the same in the Bayesian neural network. The fundamental difference is that the network weights are *probability distributions* over weights rather than single weights, i.e. we work with a distribution $p(W)$ instead of a single parameter set $W$. The simplest model of the prior over parameters $W$ is the fully-factorized prior

$$p(W) = \prod_{m=1}^{M} p(w_m) = \prod_{m=1}^{M} \mathcal{N}(w_m; \mu_m, \sigma^2_m) \tag{2.3}$$

where $M$ is the total number of individual weight scalars that make up the weight matrices and bias vectors in $W = \{W^{(l)}, b^{(l)}\}_{l=0}^{L}$. The factorization of $p(W)$ into individual priors over the individual weights $p(w_m)$ is called fully-factorized or diagonal Gaussian prior in the case of $p(w_m) = \mathcal{N}(w_m; \mu_m, \sigma^2_m)$ and will be used in this thesis due to its simplicity.
The first drawback of traditional Bayesian neural networks becomes apparent. Even though we have the benefit of being able to encode prior beliefs in $p(W)$, we do not have any intuition of what the prior probability distribution of a single weight $p(w_m)$ means with respect to the output, and even less so what prior belief we have about it. Thus, we often assume $\forall m : p(w_m) = \mathcal{N}(w_m; 0, \sigma^2_{w_m})$ for lack of a better hypothesis. Starting with this assumption despite choosing to use a Bayesian model in part due to the ability of encoding prior beliefs can be both disadvantageous and dissatisfying. The central part of this thesis in chapter 3 will provide a way of encoding prior knowledge and constraints into Bayesian neural networks by defining a prior more informative than $p(W)$ that operates in input-output space instead of the parameter weight space, tackling one of the fundamental interpretability issues of BNNs.

\subsection*{2.2.2 Prediction with the posterior predictive distribution}

Ultimately, we want to use BNNs to make predictions given some observations, just like with standard neural networks. Given the prior $p(W)$ and observed data $\mathcal{D}$, the Bayesian methodology provides us with a more principled and informative approach to making predictions. In particular, Bayes’ theorem \cite{B11}

$$p(W|\mathcal{D}) = \frac{p(\mathcal{D}, W)}{p(\mathcal{D})} = \frac{p(\mathcal{D}|W)p(W)}{p(\mathcal{D})} \quad (2.4)$$

defines what the posterior over parameters $p(W|\mathcal{D})$ given the data $\mathcal{D}$ looks like. In other words, Bayes’ rule tells us how to update our prior belief about $W$ given our observations. The posterior is then used to obtain the posterior predictive distribution

$$p(y'|x', \mathcal{D}) = \int p(y'|x', W) \ p(W|\mathcal{D}) \ dW = E_{p(W|\mathcal{D})} \left[p(y'|x', W)\right] \quad (2.5)$$

as the expected prediction given the posterior parameters. This provides us with a probability distribution over new predictions $y'$ given an input $x'$ and the ob-
served data \(D\), independent of \(W\) as we have marginalized or integrated over the parameters. Here, the conditional distribution \(p(y'|x', W)\) is simply based on the forward pass of the neural network given parameters \(W\) and often set to \(\mathcal{N}(y'; \phi(x'; W), \sigma^2I)\) due to the typical noise assumptions.

This is a major advantage of Bayesian neural networks. We have a principled way of making predictions using probability theory and statistics and along the way obtain a distribution over predictions, giving us a sense of uncertainty and multimodality. This contrasts the training and prediction of standard neural networks discussed in section 2.1.2 where the parameters \(W\) are reduced to a point estimate \(W^*\) in maximum likelihood estimation (MLE), leading to a single prediction given an input. Thus, apart from the location of one local mode of the likelihood, all information about the posterior \(p(W|D)\) is discarded when training standard neural networks. Even in maximum a posteriori estimation (MAP) where the posterior is maximized as opposed to the likelihood, we get no sense of uncertainty, contrary to the informative posterior predictive distribution of BNNs.

However, this favorable modeling advantage also leads to most difficulties in practice. There is no closed-form solution or computationally efficient way of finding the posterior predictive distribution \(p(y'|x', D)\). In fact, even the posterior over parameters \(p(W|D) = p(D|W)p(W)/p(D)\) alone is intractable due to the normalization constant \(p(D)\), also called marginal likelihood or evidence \([12]\).

Since the parameter space is vast and likelihood and prior distributions are not conjugate, there is neither a closed-from solution nor a computationally efficient way of computing the integral in \(p(D) = \int p(D|W)p(W)dW\).

Instead, we need to approximate the integral of the posterior predictive using random samples from the posterior to be able to make predictions. More specifically, using \(J\) Monte Carlo samples from the posterior \(p(W|D)\), we obtain the following unbiased estimate of the posterior predictive using Monte Carlo integra-
\[ p(y'|x', D) \approx \frac{1}{J} \sum_{j=1}^{J} p(y'|x', W^{(j)}) \quad \text{where } W^{(j)} \sim p(W|D) \quad (2.6) \]

which follows straight from Eq. \[ 2.5 \] However, it does not immediately follow how to obtain these random samples from \( p(W|D) \), and in fact such samples are anything but trivial to obtain because as discussed the posterior is complicated and intractable for BNNs. Hence, I briefly discuss the two major approaches to sampling from \( p(W|D) \) before moving on to the contributions of the thesis in chapters \[ 3 - 5 \].

### 2.2.3 Inference methods

Generating samples from the posterior \( p(W|D) \) or alternatively the unnormalized posterior or joint \( p(W, D) = p(D|W)p(W) \) are special cases of the broader problem of sampling from any target distribution with intractable normalization constant. Methods can broadly be divided into two main approaches. On the one hand, sampling methods attempt to generate samples using the unnormalized density \( p(D|W)p(W) \propto p(W|D) \) directly, often with theoretical guarantees of converging to the true target distribution. On the other hand, variational inference aims to find a parameterized variational distribution \( q(W; \lambda) \) that is as close as possible to \( p(W|D) \) via optimization of \( \lambda \). Variational inference methods are discussed in more detail because this thesis not only draws interesting parallels to constrained optimization under the variational inference framework but also derives a novel formulation of variational inference under a general Gaussian mixture family \( q(W; \lambda) \). In practice, variational inference is also often considerably more efficient than sampling approaches.
Variational inference

In variational inference, the goal is to approximate the target distribution with a variational distribution \( q(W; \lambda) \) by using optimization instead of sampling \[12\]. \( q(W; \lambda) \) belongs to a family of distributions parameterized by \( \lambda \), and we want to optimize \( \lambda \) to bring \( q(W; \lambda) \) as close as possible to \( p(W|D) \). In this thesis, most sections use the Gaussian mean-field variational family in which every target variable is governed by an independent factor in the variational distribution. Let \( \lambda = \{\mu_m, \sigma_m\}_{m=1}^M \) where as in Eq. 2.3 \( M \) denotes the number of total parameters \( w_m \) in the BNN. Define the Gaussian mean-field variational family as

\[
q(W; \lambda) = \prod_{m=1}^M \mathcal{N}(w_m; \mu_m, \sigma_m^2) \tag{2.7}
\]

More specifically, for the variational inference considered in this thesis, we aim to find the optimal variational parameters \( \lambda^* \) such that the Kullback-Leibler (KL) divergence between \( q(W; \lambda) \) and \( p(W|D) \) is minimized, i.e.

\[
\lambda^* = \arg \min_{\lambda} D_{KL}\left(q(W; \lambda) \parallel p(W|D)\right) \tag{2.8}
\]

where the KL divergence is a measure of difference between two distributions and is defined as \( D_{KL}(q(x)||p(x)) = \int q(x) \log \left( \frac{q(x)}{p(x)} \right) dx \) for two continuous distributions \( p(x), q(x) \) \[11\].

An important result is that instead of minimizing the KL divergence, we can equivalently maximize the so-called evidence lower bound (ELBO) \[12\]

\[
D_{KL}(q(W; \lambda)||p(W|D)) = \int q(W; \lambda) \log \left( \frac{q(W; \lambda)}{p(W|D)} \right)
= \mathbb{E}_{q(W; \lambda)}[\log q(W; \lambda)] - \mathbb{E}_{q(W; \lambda)}[\log p(W, D)] + \mathbb{E}_{q(W; \lambda)}[\log p(D)]
= \mathbb{E}_{q(W; \lambda)}[\log q(W; \lambda)] - \mathbb{E}_{q(W; \lambda)}[\log p(W, D)] + \log p(D)
= - \mathbb{E}_{q(W; \lambda)}[\log p(W, D) - \log q(W; \lambda)] + \log p(D) \tag{2.9}
\]
Note that our original objective $D_{KL}(q(W; \lambda)||p(W|D))$ is the same as the negative of ELBO($\lambda$) up to the evidence $\log p(D)$ in the last expression. Since the evidence is a constant with respect to the variational parameters $\lambda$, it can be ignored when optimizing the ELBO with respect to $\lambda$. Thus, minimizing the KL divergence between target and approximating distribution is equivalent to maximizing the ELBO, which is more convenient by avoiding the intractable evidence $p(D)$. The expression is called evidence lower bound because it forms a lower bound of the evidence $\log p(D)$, the marginal likelihood assigned to the data by the model. This follows from $D_{KL}(q||p) \geq 0$, and hence $\log p(D) \geq \text{ELBO}(\lambda)$.

**Bayes by Backprop.** There are several approaches to optimizing the ELBO for various distributions and settings. For my thesis and the use for Bayesian neural networks, *Bayes by Backprop* by Blundell et al. [13] has proven universally applicable and easy to use. The algorithm leverages the reparameterization trick made popular by Kingma and Welling [14] of rewriting the Gaussian distribution to separate the parameters we want to optimize for ($\lambda$) and the distribution we take the expectation over ($q(W; \lambda)$). If we can reparameterize the random variable $W$ as $W = g(\lambda, \epsilon)$ for some deterministic function $g(.)$ of $\lambda$ and a source of randomness $\epsilon \sim \pi(\epsilon)$ such as e.g. $\epsilon \sim \mathcal{N}(0,1)$, we can separate optimized parameters and expectation into

$$
\text{ELBO}(\lambda) = \mathbb{E}_{q(W; \lambda)} \left[ \log p(W, D) - \log q(W; \lambda) \right] \\
= \mathbb{E}_{\pi(\epsilon)} \left[ \log p(g(\lambda, \epsilon), D) - \log q(g(\lambda, \epsilon); \lambda) \right] \\
\approx \frac{1}{J} \sum_{j=1}^{J} \log p(g(\lambda, \epsilon^{(j)}), D) - \log q(g(\lambda, \epsilon^{(j)}); \lambda) \quad \text{where } \epsilon^{(j)} \sim \pi(\epsilon)
$$

(2.10)

and apply common gradient-based optimization algorithms to the unbiased Monte Carlo approximation of the ELBO($\lambda$) using $\pi(\epsilon)$. Note that for the Gaussian mean-field variational family, it is simple to reparameterize the random network parameters as $w_m = \mu_m + \sigma_m \epsilon_m$ with $\epsilon_m \sim \mathcal{N}(0,1)$ [13], thus separating random-
ness from the variational parameters we want to optimize.

For the optimization of the ELBO in Bayes by Backprop, I used gradient-based methods. The ADAM optimizer \[15\] worked well in practice and was used in all experiments using variational inference.

**Sampling based methods**

Contrary to efficient variational inference algorithms such as Bayes by Backprop \[13\], sampling methods are computationally intensive but often have guarantees of converging to the true distribution. The most effective algorithms for high-dimensional densities such as \(p(W|D)\) of BNNs are often so-called Markov Chain Monte Carlo (MCMC) methods. When resorting to sampling based methods in subsequent sections of this thesis, I use Hamiltonian Monte Carlo (HMC) \[16\] as well as the mode-exploring variant Generalized darting MCMC \[17\] together with HMC due to its smart exploration behavior of high-dimensional and multimodal target densities. Even though often not achievable in practice due to computational constraints, both algorithms theoretically converge to the true target distribution, enabling us to find the true posterior predictive distribution of BNNs in small-scale examples.
Chapter 3

Constrained Bayesian Neural Networks

As discussed in sections 1 and 2.2, there are many conceivable domains in which we not only hope to have accurate predictions when using our model, but also require a sense of uncertainty on the prediction. This is particularly crucial when individual predictions and its decisions are of high stakes. We can think of the diagnosis of patients as a problem where we only want to rely on machine learning when relatively certain about the prediction, and let a human expert decide when the prediction expresses high uncertainty or various modes. Bayesian neural networks which were presented in detail in section 2 fit well for many of these tasks as they give a full distribution over predictions rather than the point estimate of standard neural networks.

In the same way domain experts might take over when uncertainty is high, we can think of various settings where specialists have prior expert knowledge about the modeled output distribution. However, contrary to the prior beliefs encoded in the weight values through the prior $p(W)$, humans generally work and think in function space and as such only know of feasible and infeasible regions in the concrete input-output space. The weight space is not only less intuitive, but also immensely complicated and almost uninterpretable for deep BNNs with many hid-
den layers and units. For example, when a doctor wants to model a patient’s future health metrics given today’s lab results, she likely has specific feasible ranges for the predicted metrics (e.g. blood sugar, body core temperature, etc.) that make sense physiologically given past studies, and hence wants the fitted probabilistic model to obey these natural constraints. Yet, she does not have a good intuition as to what the distribution over weight parameters should look like to incorporate this prior belief. At the same time, the goal of an expert in robotics might be to find the optimal route in an unknown real-world terrain using reinforcement learning, but he already knows of certain states that are dangerous and to be avoided for safe exploration. Prior knowledge about system dynamics in model-based reinforcement learning is also conceivable. When solving path-planning tasks of robots the expert might know which areas are occupied by human co-workers and ought to be avoided to enable safe planning and cooperation in a shared environment. Adequately incorporating expert knowledge has the potential to not only enforce feasibility in a given domain but also make the predictive behavior of BNNs on particularly out-of-distribution (OOD) samples more accurate. OOD data is test data that is far from the majority of observations $\mathcal{D}$ where the model needs to extrapolate rather than interpolate and where highly nonlinear neural networks often fail in unreasonable ways \cite{6}.

In this thesis, I present a principled way of incorporating prior functional constraints onto the functions modeled by Bayesian neural networks. More specifically, I will define a new prior that constrains the weights $\mathcal{W}$ in a way that enables encoding prior beliefs about the posterior predictive $p(y'|x',\mathcal{D})$, i.e. in function space, in addition to the prior beliefs in $p(\mathcal{W})$. To do so, I formulate a new type of sigmoidal function in section 3.1 that can encode arbitrary inequality constraints. The constraint prior is then formally introduced and its properties analyzed in section 3.2. The effects on inference and its relation to constrained optimization in general are finally discussed in section 3.3.
3.1 Constraints and Constraint Function

The concept of incorporating constraints onto the posterior predictive of BNNs presented in this thesis relies on the fact that we can express our prior knowledge about a domain as infeasible regions in the input-output space.

More formally, let $\mathcal{S}$ be the set in the input-output space where all input-output tuples $(x, y) \in \mathcal{S}$ are denoted infeasible according to our prior knowledge. In other words, $\mathcal{S}$ denotes the infeasible region in the set of possible inputs and outputs $\mathbb{R}^{D_x} \times \mathbb{R}^{D_y}$. We want to encode $\mathcal{S}$ into the distribution over functions modeled by the BNN such that

$$p(y \text{ s.t. } (x', y') \in \mathcal{S}|x', \mathcal{D}) = 0$$

In practice, the property in Eq. 3.1 will only be enforced with arbitrarily high probability rather than strictly, which is sufficient for most practical purposes.

3.1.1 Defining the infeasible set $\mathcal{S}$

In this method, the set $\mathcal{S}$ can consist of $S$ different and not necessarily disjoint regions $\mathcal{R} \subset \mathbb{R}^{D_x} \times \mathbb{R}^{D_y}$ s.t. \( \bigcup_{s=1}^S \mathcal{R}_s = \mathcal{S} \). Every in itself infeasible region $\mathcal{R}$ must be definable by a set of constraints of the form $g(x, y) \leq 0$ such that

$$\mathcal{R} = \{ x \in \mathbb{R}^{D_x}, y \in \mathbb{R}^{D_y} \mid g_\mathcal{R}(x, y) \leq 0 \}$$

where $g_\mathcal{R} : \mathbb{R}^{D_x} \times \mathbb{R}^{D_y} \mapsto \mathbb{R}^k$ is a vector-valued function encoding $k$ different inequality constraints that define the infeasible region $\mathcal{R}$, where in practice $y = \phi(x; \mathcal{W})$ in the application with BNNs. This gives $\mathcal{S}$ a very general and potentially nonlinear shape, implying that we are able to encode many conceivable sorts of constraints onto the posterior predictive.

Note that the generality of constraints that define $\mathcal{S}$ allows for a distinction of $\mathcal{S}$ into two classes of infeasible sets. Let $\mathcal{S}$ be called an infeasible set with connected...
feasible outputs if for any given input $x \in \mathbb{R}^D_x$, the set of all corresponding outputs $y \in \mathbb{R}^D_y$ feasible under $\mathcal{S}$ is a connected set. More formally, if we denote the feasible outputs given an input $x$ as $E_{[x]} = \{y \in \mathbb{R}^D_y \mid (x, y) \notin \mathcal{S}\}$, then $\mathcal{S}$ is called an infeasible set with connected feasible outputs if

$$\forall x \in \mathbb{R}^D_x : E_{[x]} \text{ is a connected set} \tag{3.3}$$

By definition of connectedness, $E_{[x]} \subset \mathbb{R}^D_y$ is connected if it cannot be decomposed into two disjoint nonempty open sets [18]. If $\mathcal{S}$ is not an infeasible set with connected feasible outputs, then denote $\mathcal{S}$ an infeasible set with separated feasible outputs.

This definition appears abstract at first, and in this setting the notion of connectedness from analysis is not meant to imply anything more than the intuitive idea of whether the constrained set $\mathcal{S}$ splits the feasible outputs into two disconnected regions. Chapters 4 and 5 will make use of this distinction, since qualitatively more sophisticated results are possible with infeasible sets $\mathcal{S}$ of separated feasible outputs.

### 3.1.2 Guiding constraint function

We are missing the connecting piece between an infeasible region $\mathcal{S}$ that we defined and a prior expressing the violation of the posterior predictive. Furthermore, we want such a function to have informative gradients with respect to the network weights $\mathcal{W}$ such that inference algorithms are able to move away from parameters that violate $\mathcal{S}$.

**Guiding classifier**

The desired constraint function will consist of several sigmoidal functions that allow the “soft” classification of whether a constraint is satisfied or not. More
specifically, I define the *guiding classifier* $\Psi$ as

$$
\Psi_{\tau_b, \tau_s}(z) = \frac{1}{4} \left( \tanh(-\tau_b \cdot z) + 1 \right) \left( \tanh(-\tau_s \cdot z) + 1 \right)
$$

(3.4)

where $\tau_b, \tau_s$ are hyperparameters controlling the shape of the guiding classifier. Let $\tau_b$ be called the *base scale* and $\tau_s$ the *shape scale* with $\tau_s < \tau_b$ without loss of generality. $\Psi_{\tau_b, \tau_s}(z)$ is close to 1 when $z < 0$ and close to 0 when $z > 0$ and is thus able to evaluate whether a constraint of the form $z \leq 0$ is satisfied. The product of two hyperbolic tangent functions with scales $\tau_b$ and $\tau_s$ enables a sharp and steep overall classification of violating values in $z > 0$ from the positive side and a smoother and flatter classification for satisfying values in $z \leq 0$ from the negative side.

Fig. 3.1 visualizes the idea behind the definition. Since the goal of variational inference will be to find variational parameters that *do not* satisfy the constraints of the a priori defined infeasible set $\mathcal{S}$, the flatter curve for $z < 0$ is designed to provide more informative and non-saturated gradients for negative values of $z$.

The choice of tanh instead of the sigmoid $\sigma(z) = 1/(1 + \exp(-z))$ comes from a practical point of view, as the reciprocal of exponentials can cause numerical issues and “NaN” values for high absolute values of $z$ in practice due to over- and underflows. These issues are avoided by using the hyperbolic tangent function.

![Figure 3.1](image-url)

**Figure 3.1**: Plot of the guiding classifier $\Psi_{\tau_b, \tau_s}(z)$ for parameter tuples $(\tau_b, \tau_s)$ of (3, 1), (10, 1), (3, 0.5), and (10, 0.5) respectively, left to right. The naive tanh classifier $f(z) = \frac{1}{2} \tanh(-z) + \frac{1}{2}$ is plotted in dashed blue for comparison.
Constraint function

With the guiding classifier laid out, we are set to define the guiding constraint function \( c_S(x, W) \) which will finally classify whether a set of BNN parameters \( W \) violates our prior constraints by satisfying the conditions of the infeasible set \( S \) given an input \( x \). I define

\[
c_S(x, W) = \sum_{R \in S} \prod_{i=1}^{k} \Psi_{\tau_b, \tau_s}(g_R(x, \phi(x; W)))_i
\]  

(3.5)

Intuitively and informally, the sum of products of guiding classifiers \( \Psi \) acts as an “or” of “and”s of the constraints provided by the regions \( R \) in \( S \). Hence, if all constraints for at least one region \( R \) in the infeasible set are satisfied, our prior knowledge is violated and \( c_S(x, W) \) is far from 0. The constraint function value will be close to \( k \) if \( k \) is the overall number of violated regions by \((x, \phi(x; W)) \). Otherwise, at least one constraint of all infeasible regions is violated and our prior beliefs satisfied; \( c_S(x, W) \) is close to 0. Note that inputs to the guiding classifier

Figure 3.2: Heat map of the constraint function \( c_S \) for one-dimensional inputs \( x \) and outputs \( y \). Brighter parts on the heat map indicate higher values of \( c_S \), hence the constrained regions. Here, the prior constrained set \( S \) consists of two regions \( R_1 = \{x, y \mid x^2 \leq y\} \) and \( R_2 = \{x, y \mid -2 \exp \left(-\frac{1}{2}x^2\right) \geq y+1\} \); \( \tau_b = 5, \tau_s = 0.5 \).
that satisfy constraints $g_R(x, y)$ of the infeasible region $R$ are in fact results of violating BNN parameters. Conversely, violating constraints imply feasibility in the input-output space and are a result of satisfying BNN parameters. The juxtaposition of desired constraint violation and avoidance of constraint satisfaction can be counterintuitive at first. Recall that $\phi(x; W)$ is the forward pass of the BNN and $g_R(x, y) \in \mathbb{R}^k$ a vector of $k$ inequality constraints $g_R(x, y) \leq 0$ defining an infeasible region $R$. Fig. 3.2 visualizes the values of the constraint function $c_S$ in 2D for an example of $S$ consisting of two constrained regions.

3.2 The Constraint Prior

Given my definition of the guiding constraint function $c_S$ in Eq. 3.5 and our arbitrary constrained region $S$ from Eq. 3.2 I define the constraint prior as

$$\tilde{p}(W; S) \propto p(W) \cdot \nu(W; S)$$

with

$$\nu(W; S) := \exp \left( -\gamma \mathbb{E}_{x' \sim \pi_S(x')} [c_S(x', W)] \right)$$

where $\gamma$ is the constraint enforcement hyperparameter controlling how strictly the infeasible $S$ ought to be avoided. The additional term $\nu(W; S)$ in the prior encodes the satisfaction of constraints using the expectation of the constraint function $c_S$ in the constrained input space. The distribution $\pi_S(x')$ samples from the subset of the input domain $\mathbb{R}^{D_x}$ that contains members in the infeasible set $S$ for at least one possible output, i.e. the subset $S_x = \{ x' \in \mathbb{R}^{D_x} \mid \exists y' \in \mathbb{R}^{D_y} \text{ s.t. } (x', y') \in S \} \subset \mathbb{R}^{D_x}$. In practice, often $\pi_S(x') = \text{Unif}(x'; \mathcal{X})$. Finally, note that $\tilde{p}(W; S)$ is an improper prior and not a proper distribution because $\nu(W; S)$ is unnormalized.

Since we work with the unnormalized log posterior $\log \tilde{p}(W; D; S)$ in inference for all experiments in this thesis and most often in practice, this does not pose a problem.

Taking a step back, we see that the novel prior $\tilde{p}(W; S)$ is able to encode arbitrary prior knowledge about the modeled function by working in the interpretable
function space and with general inequality constraints \( g(x, y) \leq 0 \). High prior probability mass is placed not only on parameters \( W \) that conform to our often uninformed choice of \( p(W) \) but also almost entirely on regions in the function space implied by \( W \) that do not violate the constraints set by \( S \).

### 3.3 Effects on Inference: Bayes by Backprop

After the practical inference algorithm \textit{Bayes by Backprop} by Blundell et al. [13] was introduced in section 2.2.3, we can formally analyze what consequences the definition of the constraint prior \( \tilde{p}(W; S) \) has on variational inference.

Recall from section 2.2.3 that the primary objective function for the optimization in variational inference is the evidence lower bound or ELBO. The constraint prior amends this objective by an additional term. Starting from the definition of the ELBO in Eq. 2.9, we obtain

\[
\text{ELBO}_S(\lambda) = \mathbb{E}_{q(W;\lambda)} \left[ \log \tilde{p}(W, D; S) - \log q(W; \lambda) \right] \\
= \mathbb{E}_{q(W;\lambda)} \left[ \log p(D|W) + \log \tilde{p}(W; S) - \log q(W; \lambda) \right] \\
= \mathbb{E}_{q(W;\lambda)} \left[ \log p(D|W) + \log p(W) - \gamma \mathbb{E}_{x' \sim \pi_S(x')} [c_S(x', W)] - \log q(W; \lambda) \right] \\
\tag{3.8}
\]

The additional expectation term in the \( \text{ELBO}_S(\lambda) \) works as a black box in Bayes by Backprop and other variational inference and sampling methods because the gradients with respect to \( \lambda \) can be computed through the Monte Carlo approximation of the constraint function expectation over \( \pi_S(x') \), independent of \( \lambda \). Then, for practical purposes, applying the constraint prior together with the Bayes by Backprop algorithm [13] results in the procedure summarized in algorithm [1]. The standard deviations \( \sigma_m \) of the variational Gaussians are parameterized as \( \sigma_m = \exp(\sigma'_m) \) to enforce non-negativity of \( \sigma_m \) in gradient-based optimization. Furthermore, several random variables \( \epsilon \) can be sampled simultaneously in batches to get better gradi-
ent estimates by averaging the gradients of individual parameters over the batch dimension in steps 6 and 7 of algorithm 1.

Algorithm 1 Bayes by Backprop using the Constraint Prior

Input: \( \lambda_{\text{init}} \), learning rate schedule \( \alpha \), data \( \mathcal{D} \), \( \gamma \), sampler \( \pi_S(\mathbf{x}') \)

Output: optimized variational params. \( \lambda^* \) respecting constraints \( S \)

1. while \( f(\mathcal{W}) \) not converged do
2. \( \mu, \sigma \leftarrow \) unpack \( \lambda \)
3. \( \epsilon \sim \mathcal{N}(0, I_m) \)
4. \( \mathcal{W} \leftarrow \mu + \sigma \epsilon \quad \triangleright \) reparam. trick
5. \( \{\mathbf{x}^{(j)}\}_{j=1}^B \sim \pi_S(\mathbf{x}') \)
6. \( f(\mathcal{W}) \leftarrow \log q(\mathcal{W}; \lambda) - \log p(\mathcal{D}|\mathcal{W}) - \log p(\mathcal{W}) + \gamma \frac{1}{B} \sum_{j=1}^B c_S(\mathbf{x}^{(j)}, \mathcal{W}) \)
7. \( \lambda \leftarrow \lambda - \alpha \nabla_{\lambda} f(\mathcal{W}) \quad \triangleright \) gradients with auto-differentiation
8. update \( \alpha \)
9. end while

3.3.1 Relationship to constrained optimization

Constrained nonlinear optimization covers the optimization of general and non-linear objectives under constraints. Since the topic of this thesis is probabilistic modeling and inference rather than a detailed discussion of constrained optimization – an area which itself is vast enough to fill several works alone – I will not go into detail when discussing the relationship of the ideas of this work and constrained optimization. However, not surprisingly, section \[ \text{3} \] and algorithm \[ \text{1} \] establish interesting connections and similarities to approaches from this related field, specifically to penalty and log-barrier methods \[ \text{19} \].

Applying the constraint prior to variational inference algorithms such as Bayes by Backprop \[ \text{13} \] transforms the standard ELBO objective into \( \mathcal{L}(\lambda) = -ELBO(\lambda) + \gamma \mathbb{E}_{\mathbf{x}' \sim \pi_S(\mathbf{x}')} [c_S(\mathbf{x}', \mathcal{W})] \), see Eq. \[ \text{3.8} \]. The expectation of the constraint function \( c_S \) is a special case of a penalty term when seen from the angle of penalty-based optimization approaches. In these methods, constrained programming problems are converted into unconstrained problems by adding a term to the objective that encodes the constraint violation. Minimizing the new objective then guides the un-
constrained optimization to feasible solutions of the constrained problem. Previous
log-barrier and penalty methods have used penalty terms such as \( \sum_i \log(g_i(x)) \),
\( \sum_i (g_i(x))^2 \), \( \sum_i (\max\{0, g_i(x)\})^2 \), and others to encode constraints into the objective function [19].

However, in addition to not having been considered in the domain of prob-
abilistic modeling of neural networks, they share one or more of the following
shortcomings that are crucial to this application. Interior point and log-barrier
methods require a feasible starting point, which in general is not easy to find in
weight space when imposing complicated infeasibility constraints onto the BNN.
Furthermore, other penalty functions leave infeasible points either undefined in ex-
terior regions or map them to values that approach infinity, resulting in overflows
in practice.

Conversely, the constraint function \( c_S \) and the constraint prior \( \tilde{p}(W; S) \) intro-
duced earlier allow the optimization to start in infeasible regions of the weight
space. By design, the curvature of the novel constraint function encourages non-
vanishing gradients in infeasible regions, thus guiding the variational parameters
to agree with our prior beliefs. From a probabilistic point of view, close to no
prior probability mass is then placed on weights that imply a distribution over
functions that violate our constrained region \( S \).

At its essence, this connection to constrained optimization is intuitive. The
goal of this thesis is to define a prior distribution that captures constraints in
function space. Hence, optimization in variational inference using this prior can
be viewed as constrained optimization with respect to the constraints that define \( S \).
Because the ELBO is generally a non-convex objective [12], we obtain the general
setup of constrained nonlinear optimization when combining the constraint prior
with algorithms such as Bayes by Backprop.
3.3.2 Other related work

To the best of my knowledge, there is no literature on how to effectively impose general and yet interpretable constraints in the input-output space onto the distribution over functions modeled by Bayesian neural networks. However, a thorough literature review found that some related state-of-the-art literature exists. The work *Constraining the Dynamics of Deep Probabilistic Models* by Lorenzi and Filippone [20] recently published at the 35th International Conference on Machine Learning (ICML 2018) provides a distinct but also related framework where the modeled functions and derivatives of a given order are subject to constraints.

However, the authors focus on deep Gaussian processes, center their experiments around constraints on derivatives, and focus most of the theoretical efforts on equality constraints and specific assumptions about noise models of the dynamics. Hence, this thesis’s core research contributions of defining the intuitive constraint prior of arbitrary shape, the discussion of differences in classes of infeasible sets as well as its detailed evaluation through advanced posterior results and a novel variational inference method in chapters 4 and 5 form a definite contribution to the state-of-the-art research.
Chapter 4

Experiments and Evaluation

This section is designed to provide proofs of concept of the theoretical formulation derived in section 3 with interpretable two-dimensional examples. With the constraint prior $\tilde{p}(W; S)$, it is possible to encode arbitrary prior knowledge about our model in input-output space using constraints of the general form $g(x, y) \leq 0$. The analysis, plots, and visualizations will show that with the new method, high probability mass is placed almost entirely on regions in the function space implied by parameters $W$ that do not violate the constraints set by $S$. Furthermore, its superiority over standard BNNs in these use cases as well as the importance of certain modeling choices are established.

This chapter only discusses well-behaved constraints with connected feasible outputs, as introduced in section 3.1.1. Intuitively, these constrained regions do not split the feasible output regions of the BNN’s posterior predictive into disconnected regions, and hence standard inference techniques are able to successfully approximate the posterior. Later, chapter 5 will extend these standard cases by deriving novel as well as amending existing inference techniques that are more advanced and discussing infeasible sets with separated feasible outputs and their multimodal posterior results. Specifically, in part 4.1 of this chapter, I will demonstrate that simple linear and nonlinear bounds, equality constraints, and even only asymptotic behavior can be successfully enforced by the constraint prior. Many
of these types of inequality constraints are for instance commonly found in linear and nonlinear programming. All these examples are demonstrations of the ability of Constrained BNN to enforce straightforward constraints with connected feasible outputs. After the empirical results were demonstrated, part 4.2 performs ablation studies over the central modeling aspects of Constrained Bayesian Neural Networks, to better understand the relative importance of specific hyperparameters and the relationship to standard unconstrained BNNs. In doing so, I find that Constrained BNNs can converge to better unconstrained ELBO values on average than standard BNNs in variational inference, everything else held constant.

When performing the experiments in this and the following chapter, two univariate functions \( f(x) \) and \( g(x) \) will serve as canonical examples to demonstrate expert knowledge about upper and lower bounds, asymptotic behavior, other infeasible regions, and missing in-distribution and out-of-distribution data. For all following sections, denote

\[
\begin{align*}
    f(x) &= 2 \cdot \exp(-x^2) \cdot \sin(5x) \\
    g(x) &= \frac{2}{3} x^4 + \frac{4}{3} x^2 + 1
\end{align*}
\]

The factor \( \exp(-x^2) \) results in \( f(x) \) converging to zero as \( x \to +\infty \) and \( x \to -\infty \) and thus allows for suitable demonstrations of expert knowledge about asymptotic behavior. Both ground truth functions \( f(x) \) and \( g(x) \) are shown in the panels of Figs. 4.1 and 4.2 together with the posterior predictive of two BNNs with one hidden layer of 20 nodes and rbf activation function, fitted to observations from the corresponding ground truth function. The BNNs used the prior \( p(W) = \prod_{m=1}^{M} \mathcal{N}(w_m; 0, 10) \). See section 2.2.1 for details. Unless otherwise stated, all experiments in the empirical sections of this thesis use this neural network architecture, standard prior, and activation function.

To evaluate scenarios when either in-distribution and out-of-distribution data is missing, the function \( f(x) \) has only 10 observed in-distribution data points \( D_f \).
available, specifically testing extrapolation behavior. On the other hand, \( g(x) \) will have only 6 observed data points \( D_g \) with data missing from an enclosed distribution space. More specifically, let \( D_f = \{x_i, y_i\}_{i=1}^{10} \) with \( x_i \sim \text{Unif}(-1.5, 1.5) \) and \( y_i \sim \mathcal{N}(x_i, 0.1^2) \). Likewise, let \( D_g = \{x_i, y_i\}_{i=1}^{6} \) with the mixture distribution \( x_i \sim \frac{1}{2}\text{Unif}(-3, -1) + \frac{1}{2}\text{Unif}(1, 3) \) and \( y_i \sim \mathcal{N}(x_i, 0.1^2) \). Thus, the observation noise \( \epsilon \) was normally distributed as \( \epsilon \sim \mathcal{N}(0, 0.1^2) \). For better comparability, the data was sampled once and then kept constant across experiments and comparisons.

In all plots below, the ground truth function is drawn as a black dashed line and observations are marked with a black cross marker. Red regions indicate the constrained regions \( \mathcal{R} \) that form the constrained set \( S \). Blue is used for plots of the posterior predictive. More specifically, when applicable, the posterior predictive sample mean is plotted in dark blue and fading shades of blue indicate \( \pm \sigma \) and \( \pm 2\sigma \) confidence intervals calculated from samples of the posterior predictive. As discussed in detail in section 5.2 plotting the mean only makes sense when the posterior predictive is approximately unimodal. In more complex cases, individual particles from sampling are plotted in light blue to resemble the density when several functions are overlaid. Thus, the mean-based plots are only used for approximately unimodal posterior predictive distributions in chapter 4 and plots of particles in the experiments of chapter 5 on multimodal posteriors resulting from the constraint prior.

In general, we aim to enforce arbitrary constraints in function space with the constraint prior \( \tilde{p}(\mathcal{W}; \mathcal{S}) \). Hence to evaluate the degree to which constraints are satisfied, we need a metric of success, a metric of constraint violation in the Bayesian setting. I define the posterior constraint violation (PCV) of the con-
Figure 4.1: Ground truth function \( f(x) \) next to posterior predictive of a BNN with one hidden layer of 20 nodes using HMC for observed data \( D_f \). For this baseline, no constraint prior was used for modeling. In-distribution and out-of-distribution test points used in later experiments are marked as red stars.

Figure 4.2: Analogous to 4.1, ground truth function \( g(x) \) next to posterior predictive of the same BNN for observed data \( D_g \). No constraint prior.

strained regions \( S \) and BNN posterior given data \( D \) as

\[
PVC(S;D) = \int_S p(y'|x', D) \, dy' \, dx' \\
= \int_S \mathbb{E}_{W \sim p(W|D)}[p(y'|x', W)] \, dy' \, dx' \\
= \int_S \int p(y'|x', W) \, p(W|D) \, dW \, dy' \, dx'
\]  

The posterior constraint violation is the total mass of the posterior predictive probability that is placed into the constrained set \( S \) by the posterior of the BNN given the data. Hence, the PCV is an interpretable evaluation metric for a set of
constraints rather than for example the mere constraint function \( c_S(x, W) \) which is part of the objective and only defined given an input \( x \). The intention behind the constraint prior is clearly that close to zero probability mass of the posterior predictive is placed in these defined regions. The subsequent sections confirm that this goal is achieved an all settings.

4.1 Examples and Use Cases

As mentioned earlier, recall that all of the constraints in this section have in common that they define a constrained region \( S \) with connected feasible outputs, see section 3.1.1 for the definition. Intuitively, this means that the constrained set does not split the feasible outputs into disconnected segments, i.e. the function modeled by the BNN is not forced into a single constrained region of the input-output space. The focus here is thus on well-behaved linear and nonlinear bounds as well as equality constraints, as for example often used in linear programming. As stated above, all experiments of this section use a simple BNN with one hidden layer of 20 nodes, rbf activation, and prior \( p(W) = \prod_{m=1}^{M} \mathcal{N}(w_m; 0, 10) \).

We assume we have some prior knowledge about the process we are modeling. In particular, based on our domain knowledge in this hypothetical scenario, we know upper and lower bounds as well as asymptotic functional behavior of the function \( f \) and \( g \) that we model, while we are uncertain what the function looks like around \( x = 0 \).

4.1.1 Linear and nonlinear bounds

As a first detailed example, assume we know that the function \( f(x) \) we are modeling is linearly bounded above and below as follows

\[
\forall x: \quad -2.5 \leq f(x) \leq 2.5
\] (4.6)
This constraint corresponds to two regions \(\mathcal{R}\) in the constrained set \(\mathcal{S}\) since \(\mathcal{S} = \mathcal{R}_1 \cup \mathcal{R}_2\) with \(\mathcal{R}_1 = \{ \, x \in \mathbb{R}, y \in \mathbb{R} \mid y \leq -2.5 \, \}\) and \(\mathcal{R}_2 = \{ \, x \in \mathbb{R}, y \in \mathbb{R} \mid y \geq 2.5 \, \}\). The left plot of Fig. 4.3 summarizes the results of running Hamiltonian Monte Carlo (HMC) \cite{16}, providing the theoretical gold-standard in inference as its samples from the posterior theoretically converge to the true distribution. Qualitatively, we confirm that by using the constraint prior, the posterior predictive of the BNN completely respects the imposed constraints whereas the standard posterior of the BNN violates or prior knowledge, especially in out-of-distribution areas.

Going one step further, assume we know that the process roughly follows the nonlinear upper and lower bound given by a bell curve, i.e. \(\exp(-x^2)\). We can also enforce the following constraint onto our BNN using the constraint prior

\[
\forall x : \quad -2\exp(-x^2) - \frac{1}{2} \leq f(x) \leq 2\exp(-x^2) + \frac{1}{2}
\]  

(4.7)

again corresponding to two but this time nonlinear constrained regions in the set \(\mathcal{S}\). Again using HMC \cite{16} for inference, the right plot of Fig. 4.3 demonstrates that even nonlinear bounds are easily expressible using the prior \(\tilde{p}(W; \mathcal{S})\). After careful tuning of the leap frog method for HMC \cite{16} for moving through the energy landscape of the posterior, both times 5000 samples were collected after 2000 samples of burn-in, i.e. the first 2000 of 7000 samples were discarded, allowing the Markov chain in HMC to first travel to regions of high probability mass. The thinning rate was 5, implying that only every fifth sample was stored to reduce sample cross-correlation. HMC used \(L = 25\) and \(L = 20\) steps respectively and a step size of \(\epsilon = 5e - 3\). These hyperparameters ensured an acceptance probability of between 0.6 and 0.9 on average.

Table 4.1 summarizes the quantitative comparison of standard and Constrained BNNs for different sized neural network architectures, including a deeper two layer network. The constraint prior not only qualitatively incorporates the prior knowledge better than the usual prior \(p(W)\), but the likelihood of held-out in-
distribution (ID-LL) and out-of-distribution data (OOD-LL) is also quantitatively
different for the constraint prior model. The standard and constrained posterior
predictive are compared on modeling held-out-data as well as their prediction root
mean squared error (RMSE) using the sample mean $\mu = \mathbb{E}_{p(W|D_f)}[p(y'|x', W)]$
as the prediction, both in and out-of-distribution. The log-likelihood of the held-out
data is computed using a Gaussian approximation of the posterior with $\mathcal{N}(y; \mu, \sigma^2)$
using the sample mean and sample variance $\sigma^2 = \text{Var}_p(W|D_f)[p(y'|x', W)]$. For $f(x)$, the held-out in-distribution data was sampled as $D^\text{id}_f = \{x_i, y_i\}_{i=1}^{10}$ with
$x_i \sim \mathcal{U}(-1.5, 1.5)$, i.e. the same distribution as $D_f$. Conversely, the held-out
out-of-distribution data is $D^\text{ood}_f = \{x_i, y_i\}_{i=1}^{10}$ with $x_i \sim \frac{1}{2}\mathcal{U}(-4, -1.5) + \frac{1}{2}\mathcal{U}(1.5, 4)$.
Like the observations $D_f, D_g$, the held-out data was held constant across all of the

Figure 4.3: Comparing posterior predictives of the standard BNN of Fig. 4.1 (top)
with BNNs using the constraint prior $\tilde{p}(W; S)$ (bottom). In the bottom-left plot,
the Constrained BNN successfully enforced the linear bounds of Eq. constraint 4.6 in the bottom-right plot nonlinear bounds of Eq. 4.7. Note that the y-axis
scale and range of uncertainty is a lot larger for the standard BNN.

35
following experiments and plotted once for illustrative purposes in Figs. 4.1 and 4.2.

While the in-distribution metrics are comparable for standard and Constrained BNNs, the ability to extrapolate, being guided by the enforced constraints, is much better for the model using the constraint prior. The out-of-distribution error and held log-likelihood are consistently and significantly better for the Constrained BNN. Since we incorporate specific information about the location of the held-out out-of-sample data, this might not seem surprising, but it provides definite evidence that the constraint prior and Constrained BNNs succeed at incorporating constraints without using specific data but rather general constraints in function space. Furthermore, note that differences in OOD log-likelihood of standard and Constrained BNNs of e.g. -48 and 48, or -81 and 25 are in log space and thus very significant. In Bayesian modeling, the log-likelihood bears greater weighting than prediction RMSE, hence it is positive to find the OOD log-likelihood to be much higher for the constrained Bayesian neural network. Moreover, the posterior

<table>
<thead>
<tr>
<th>$p(W; S)$ ?</th>
<th>L</th>
<th>h</th>
<th>PCV</th>
<th>ID-LI</th>
<th>ID-RMSE</th>
<th>OOD-LI</th>
<th>OOD-RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>1</td>
<td>20</td>
<td>5.4721</td>
<td>-1148.5211</td>
<td>1.4593</td>
<td>-48.2928</td>
<td>0.2744</td>
</tr>
<tr>
<td>Yes</td>
<td>1</td>
<td>20</td>
<td>0.0003</td>
<td>-1744.1942</td>
<td>0.7683</td>
<td>47.9888</td>
<td>0.0128</td>
</tr>
<tr>
<td>No</td>
<td>1</td>
<td>50</td>
<td>6.1055</td>
<td>-905.4302</td>
<td>0.4947</td>
<td>-64.7072</td>
<td>0.4099</td>
</tr>
<tr>
<td>Yes</td>
<td>1</td>
<td>50</td>
<td>0.0000</td>
<td>-1220.0722</td>
<td>0.5332</td>
<td>26.3226</td>
<td>0.0195</td>
</tr>
<tr>
<td>No</td>
<td>2</td>
<td>50</td>
<td>8.7585</td>
<td>-798.1259</td>
<td>1.0868</td>
<td>-81.9233</td>
<td>0.6789</td>
</tr>
<tr>
<td>Yes</td>
<td>2</td>
<td>50</td>
<td>0.0000</td>
<td>-909.8671</td>
<td>0.4751</td>
<td>25.4097</td>
<td>0.0207</td>
</tr>
</tbody>
</table>

Table 4.1: Comparison of regular and Constrained BNNs for nonlinear inequality constraints. The constraint prior achieves the enforcement of constraints of $S$, which results in consistently lower out-of-distribution RMSE and held-out log-likelihood. All Constrained BNNs had $\gamma = 2000$.

constraint violation (PCV), defined in Eq. 4.3, to measure the violation of the constrained region $S$, fully reflects the ability of the constraint prior to incorporate the prior functional knowledge defined by the constraints. Across all experiments, the total probability mass of the posterior predictive placed into constrained regions by the posterior was close to zero for Constrained BNNs. Conversely, high PCV values indicate that standard BNNs violate these regions significantly.
The in-distribution held-out log-likelihood is found to be slightly worse for the Constrained BNN in this experiment setting. However, the in-sample log-likelihood values overall are very low, mostly because the available data $D_f$ is sparse and the provided constraints lack significant information about in-distribution data. Hence, the quantitative in-distribution metrics do not carry much weight compared to the significant quantitative differences in OOD behavior and qualitative improvement of the posterior predictive in Figs. 4.1 and 4.3 with respect to the provided constraints.

4.1.2 Enforcing asymptotic behavior only

Taking the previous scenario one step further, assume that our domain expertise tells us that the described physical upper and lower bounds only hold asymptotically. In other words, we are uncertain about the behavior of $f(x)$ around $x = 0$ but know the bounds to be true for $|x| \geq 1$. We can express even this type of conditional expert constraints

$$
\forall x \geq 1 : \quad -\exp(-x^2) - \frac{1}{2} \leq f(x) \leq \exp(-x^2) + \frac{1}{2}
$$

$$
\forall x \leq -1 : \quad -\exp(-x^2) - \frac{1}{2} \leq f(x) \leq \exp(-x^2) + \frac{1}{2} \quad (4.8)
$$

using the constraint prior and four disjoint regions. Figure demonstrates the results of applying the asymptotic constraints to the BNN modeling $f(x)$.

The previous experiments used Hamiltonian Monte Carlo [16] for inference, providing samples that provably converge to the true posterior. However, in practice, HMC and sampling based methods are often not be scalable enough for large BNNs. Hence, optimization-based variational inference as introduced in section 2.2.3 is often used to efficiently find a simplified approximation to the posterior, paying the price of having samples that slightly diverge from the true posterior.

To demonstrate the applicability of the constraint prior to mean-field variational inference methods, the efficient Bayes by Backprop algorithm [13] was used
for this experiment, see section 2.2.3. The algorithm was run for 5 random restarts, choosing the run achieving the maximum objective. The variational parameters were initialized as $\mu_n \sim \mathcal{N}(0,2)$ and $\rho_n = -8$. Every restart of the algorithm was run for 35,000 iterations to achieve convergence. As mentioned in section 2.2.3, optimization of the ELBO objective was done using ADAM [15]. Note that variational inference was used not because the nature of this experiment is different from the previous ones, but to show that Constrained BNNs are not only applicable to sampling methods but also to variational inference.

As for the inequality bounds in section 4.1.1, Constrained BNNs quantitatively outperform standard BNN models which are unable to incorporate expert knowledge a priori. The results are summarized in Table 4.2. As previously, the Constrained BNN shows much greater log-likelihood of the held-out data especially out-of-distribution, where we specifically applied our expert knowledge. In addition, the posterior constraint violation (PCV) values show that the novel prior for BNNs achieves its goal in fully incorporating expert knowledge without violation. PCV for Constrained BNNs is consistently much lower than for the equivalent regular BNN. In-distribution metrics are qualitatively equivalent for both methods as no constraints applied to in-distribution regions.
<table>
<thead>
<tr>
<th>$\hat{p}(W; S)$ ?</th>
<th>$L$</th>
<th>$h$</th>
<th>PCV</th>
<th>ID-LL</th>
<th>ID-RMSE</th>
<th>OOD-LL</th>
<th>OOD-RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>1</td>
<td>20</td>
<td>12.5959</td>
<td>-932.3903</td>
<td>0.3822</td>
<td>-287.7710</td>
<td>1.2404</td>
</tr>
<tr>
<td>Yes</td>
<td>1</td>
<td>20</td>
<td>0.0000</td>
<td>-1058.5134</td>
<td>0.7877</td>
<td>40.3288</td>
<td>0.0601</td>
</tr>
<tr>
<td>No</td>
<td>1</td>
<td>50</td>
<td>14.9199</td>
<td>-390.1461</td>
<td>1.1189</td>
<td>-428.2799</td>
<td>1.9963</td>
</tr>
<tr>
<td>Yes</td>
<td>1</td>
<td>50</td>
<td>0.1760</td>
<td>-489.2482</td>
<td>0.4195</td>
<td>29.0278</td>
<td>0.0797</td>
</tr>
<tr>
<td>No*</td>
<td>2</td>
<td>50</td>
<td>13.8200</td>
<td>-98.5591</td>
<td>0.9670</td>
<td>-70.1272</td>
<td>1.5737</td>
</tr>
<tr>
<td>Yes*</td>
<td>2</td>
<td>50</td>
<td>0.2680</td>
<td>-223.8693</td>
<td>0.8819</td>
<td>28.5774</td>
<td>0.0140</td>
</tr>
</tbody>
</table>

Table 4.2: Comparison of various standard and Constrained BNNs enforcing asymptotic behavior only. The constraint prior achieves the enforcement of the constraints defined by $S$ and outperforms the standard BNN in terms of PCV and OOD accuracy and likelihood. Bayes by Backprop [13] variational inference was used.

4.1.3 Equality constraints

In addition to the upper and lower bounds that are realistically often available in the form of inequalities in the prediction space, it is possible to imagine cases where equality constraints are useful and available as well. This is analogous to having additional data points in certain input regions collected under the assumption of no noise. In fact, as established below, enforcing an equality constraint over a short interval in the input region even sets a first-order notion of slope, which further restricts the posterior distribution.

Consider the scenario where it is known that

$$\forall x \in [-\epsilon, \epsilon]: \ g(x) = 1$$  \hspace{1cm} (4.9)

for small positive $\epsilon$. This can be practically expressed as two overlapping inequality constraints, hence by two constrained regions encoded in the constraint prior $\hat{p}(W; S)$. The result of applying the equality constraint with a Constrained BNN modeling the second function $g(x)$ is summarized in Fig. 4.5 and is contrasted by the standard BNN baseline of Fig. 4.2 depicted again on the left. Using Hamilto-

---

*Deeper BNNs make Bayes by Backprop overfit when optimizing until convergence. The result is a degenerate tunnel-shaped approximation to the predictive distribution while individual functions $\phi(x; W)$ fit all data points. Early stopping was used and should be taken into account when interpreting the results. True posteriors obtained by HMC are not degenerate.
Figure 4.5: Posterior predictive of the Constrained BNN (right) incorporating specific knowledge in form of equality constraints, contrasted with the predictive distribution of the standard BNN of Fig. 4.2 (left). Empirically, the constraint prior appears to also enforce the slope by extending over an interval rather than a single point, further constraining the posterior.

4.2 Ablation Experiments

Although convincing empirical results have been demonstrated, the experiments in section 4.1 have not isolated the specific modeling choices and hyperparameters of Constrained BNNs and how the speed of learning and convergence in particular compare to standard BNNs altogether. In this section, ablation experiments are performed over the key aspects of Constrained BNNs to better understand the necessity and relative importance of specific hyperparameters and parts of the method itself. Here, an ablation study refers to the analysis of a certain aspect of the model by completely removing or changing that aspect while holding
everything else equal, to see how behavior and performance are affected.

### 4.2.1 Effect of the constraint enforcement parameter $\gamma$

The choice of the constraint enforcement parameter $\gamma$ plays an important role in variational inference and the enforcement of constraints in Constrained BNNs. Recall from the definition of the constraint prior in Eq. 3.6 that

$$\tilde{p}(\mathbf{W}; \mathcal{S}) \propto p(\mathbf{W}) \cdot \exp \left( -\gamma \mathbb{E}_{\mathbf{x}' \sim \pi_{\mathcal{S}}(\mathbf{x}')} \left[ c_{\mathcal{S}}(\mathbf{x}', \mathbf{W}) \right] \right)$$

Hence, the parameter $\gamma$ strongly influences how the normal prior $p(\mathbf{W})$ is weighted compared to the exponential of the expectation of the negated constraint function $c_{\mathcal{S}}(\mathbf{x}, \mathbf{W})$.

This section isolates the importance of $\gamma$ and shows how certain choices are necessary to achieve specific properties and variations of the model in the setting of variational inference, such as for example the incorporation of softer over stricter constraints. We also observe that the unconstrained objectives and advantages of variational inference are not negatively affected, essentially making Constrained BNNs a universal extension to BNNs for all practical use cases. In fact, a series of experiments reveals that Constrained BNNs consistently reach better unconstrained ELBO values than standard BNNs in mean-field variational inference with everything held constant. All comparisons will be done using Bayes by Backprop with the hyperparameters used when applying variational inference in section 4.1.2.

Once more considering a small BNN with one hidden layer of 20 nodes on the constraint setting of *nonlinear* bounds of section 4.1.1 and Fig. 4.3, the parameter choice of $\gamma$ is assessed using the metrics of previous sections such as held-out log-likelihoods and posterior constraint violation as well as the unconstrained ELBO, i.e. the ELBO using only the conventional prior $p(\mathbf{W})$. As explained above and in the section on the constraint prior (Eq. 3.6 section 3.2) as well as the
discussion of effects on variational inference (Alg. 1, section 3.3), the enforcement
parameter $\gamma$ weighs the constraint violation term $c_S(x, \mathcal{W})$ against the standard
ELBO objective. Thus, small values of $\gamma$ in the prior $\tilde{p}(\mathbf{W}; S)$ will enforce the prior
constraints of $\mathcal{S}$ much more softly than higher values of $\gamma$, whose strict constraints
will be enforced with arbitrarily high probability depending on $\gamma$.

Table 4.3 lists different choices of $\gamma$ and their effects on the standard variati-
onal objective and posterior constraint violation as well as previous metrics. All
values are averaged over five runs of the algorithm to account for variations in
random restarts when analyzing trends. We observe and confirm our intuition

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>ELBO</th>
<th>PCV</th>
<th>ID-LL</th>
<th>ID-RMSE</th>
<th>OOD-LL</th>
<th>OOD-RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-126.0698</td>
<td>4.8201</td>
<td>-712.8677</td>
<td>0.7924</td>
<td>-213.1542</td>
<td>0.8004</td>
</tr>
<tr>
<td>1</td>
<td>-114.5556</td>
<td>4.9891</td>
<td>-979.9139</td>
<td>0.5115</td>
<td>-264.2013</td>
<td>0.8589</td>
</tr>
<tr>
<td>10</td>
<td>-112.7101</td>
<td>1.9795</td>
<td>-1406.8196</td>
<td>0.8834</td>
<td>-1207.1119</td>
<td>0.4990</td>
</tr>
<tr>
<td>100</td>
<td>-126.9794</td>
<td>0.1881</td>
<td>-1297.3185</td>
<td>0.5231</td>
<td>51.9268</td>
<td>0.0194</td>
</tr>
<tr>
<td>1,000</td>
<td>-117.9976</td>
<td>0.0086</td>
<td>-903.1517</td>
<td>0.6167</td>
<td>47.9614</td>
<td>0.0402</td>
</tr>
<tr>
<td>10,000</td>
<td>-112.3648</td>
<td>0.0000</td>
<td>-1528.3200</td>
<td>0.6035</td>
<td>48.4614</td>
<td>0.0455</td>
</tr>
</tbody>
</table>

Table 4.3: Choices of the enforcement parameter $\gamma$ in the constraint prior $\tilde{p}({\mathbf{W}}; \mathcal{S})$
in the setting of nonlinear bounds of section 4.1.1. Higher values enforce con-
straints more strictly but do not interfere with reaching qualitatively similar values
of the ELBO as for the unconstrained case, i.e. $\gamma = 0$. Bayes by Backprop
was run for 10,000 iterations in each setting using ADAM optimization [15] and
learning rate 0.01. All values are averages of 5 random initializations.

that for small values of $\gamma$, the constraint prior does not at all or only very softly
enforce the prior knowledge on impossible values in $\mathcal{S}$, hence resulting in higher
PCV values. The higher the enforcement parameter $\gamma$, the greater the ratio be-
tween violation term $c_S(x, \mathcal{W})$ and ELBO in the variational objective of Bayes by
Backprop, and constraints get enforced more strictly. For $\gamma = 100$ and higher, the
differences between runs of the constraint prior are only marginal and insignificant.
At that point, the constraint function part of the variational objective outweighs
the standard ELBO, and thus constraints get enforced very strictly. Note that
equal or very similar ELBO values are achieved with Constrained BNNs in gen-
eral compared to the unconstrained model. This strongly supports the use of the
constraint prior as we notice that the same ultimate and unconstrained Bayesian
objective is reached even with the additional prior. In other words, the prior defined according to our prior beliefs in specific domain regions only contributes to the favorable behavior in the constrained settings, without limiting the goal of achieving ELBO levels as high as possible.

Furthermore, not only does the constraint prior not negatively affect the achieved result of unconstrained inference for BNNs, but the speed of convergence of variational inference with Constrained BNNs is also marginally faster and the reached ELBO levels slightly higher as for standard BNNs. Figure 4.6 summarizes the qualitative comparison of variational objective, expected constraint function val-

![Variational inference metrics over iterations of Bayes by Backprop using the constraint prior with different values of $\gamma$. The data is from Table 4.3 and thus averaged over 5 random restarts. The rate of convergence is consistently at least as high as for standard BNNs and the unconstrained ELBO values reached during optimization are found to be higher when using the constraint prior.](image-url)
ues and unconstrained ELBO values of constrained and unconstrained BNNs during the optimization that resulted in the values of Table 4.3. Using the results summarized in the table, five random restarts of each setting of $\gamma$ are averaged to yield a relatively reliable and stable trend in the plots. Furthermore, every datapoint in the plot was the average of the last 100 values in the time-series. Since the advantage of variational inference in general is its efficiency compared to sampling methods, it is very positive to observe that the rate of convergence is qualitatively equivalent in these experiments, not hindering the applicability of these methods in practice.

As confirmed by Table 4.3, the bottom plot of Fig. 4.6 shows that the unconstrained ELBO values reached during optimization are consistently higher when using Constrained BNNs. This implies that including the constraint prior when optimizing the variational objective helps the converge to variational parameters $\lambda$ that provide a better lower bound on the marginal likelihood of the data on average, the ultimate maximization objective in variational inference. This is a powerful result and further supports the applicability of Constrained BNNs in practice.

However, note that this does not imply that the constraint prior helps reach better ELBO values in all cases. The results show that the ELBO values are higher on average, but individual random restarts can vary. Yet, this does not mean unconstrained optimization would not prefer the solutions reached by Constrained BNNs, which they should due to the higher unconstrained ELBO objective. When initializing Bayes by Backprop on a standard BNN with pre-optimized parameters of a Constrained BNN with $\gamma >> 0$, I observe that the ELBO immediately stays at and improves from the initialized constrained solution. This confirms not only that standard inference prefers the solutions with better ELBO objectives found by Constrained BNNs but also that solutions to the constrained optimization are valid and sensible in the unconstrained objective space.

According to these results, it appears to be the case that the constraints en-
forced by the constraint prior are able to guide the optimization to better solutions on average than standard BNN densities themselves. Thus, even though this phenomenon was only observed when averaging over restarts, it can be seen as a promising step towards improving optimization in variational inference. It also remains to be determined what specific settings of $\gamma$ optimize this behavior. The table and plots reveal that neither extremely low nor extremely higher values of $\gamma$ outperform the other, although it appears to be the case from analyzing the plots that Constrained BNNs with high $\gamma$ values converge slightly slower but converge to marginally higher ELBO levels.

The top right plot of Fig. 4.6 nicely contrasts how different magnitudes of $\gamma$ affect the violation values computed by the constraint function $c_S(x,W)$, the heart of the constraint prior in Constrained BNNs. With $\gamma$ increasing by factors of 10, the violation implied by the variational distribution is monotonically lowered across iterations of Bayes by Backprop. The top left panel of Fig. 4.6 plots the variational objective of Constrained BNNs across optimization iterations with ADAM [15] and shows that consistently lower variational objectives are obtained using Constrained BNNs. Since the constraint violation term $\mathbb{E}_{x \sim \pi_S(x)}[\gamma \cdot c_S(x,W)]$ in the objective is non-negative, this observation is effectively the reason for the findings of higher unconstrained ELBOs shown in the bottom plot of Fig. 4.6.

Why exactly the constraint prior helps in converging to better variational parameters during optimization even according to unconstrained Bayesian metrics is an open question at this point. One explanation could be that the information incorporated by means of the constraint prior respects the observations $\mathcal{D}$ and as such, indirectly guides the $q$ distribution to match the likelihood better. When rearranging the constrained ELBO formulation of Eq. 3.8 as

$$\text{ELBO}_S(\lambda) = \mathbb{E}_{q(W;\lambda)} \left[ \log p(W,\mathcal{D};\mathcal{S}) - \log q(W;\lambda) \right] = \mathbb{E}_{q(W;\lambda)} \left[ \log p(\mathcal{D}|W) \right] - D_{KL}(q(W;\lambda)||\tilde{p}(W;\mathcal{S}))$$

(4.10)
we see that optimizing the ELBO is equivalent to the optimization of the data log-likelihood while not diverging from the prior. Thus, when being directly regularized to match the constraint prior, it could be hypothesized that the optimization of the likelihood gets indirectly improved.

 Ultimately, these ablation experiments established that the choice of $\gamma$ not only affects how strictly prior knowledge and constraints are enforced by Constrained BNNs and what specific settings lead to optimal results, but also revealed that the constraint prior helps converge to higher unconstrained ELBO values on average in optimization compared to equivalent standard BNNs.

### 4.2.2 Effect of parameterizations of $\Psi_{\tau_b, \tau_s}$

Defined in section 3.1.2, the constraint function $c_S(x, \mathcal{W})$ is a sum of products of the guiding classifier $\Psi_{\tau_b, \tau_s}$ of Eq. 3.4. Thus, the constraint function actively depends on the classifier and its parameterization via $\tau_b, \tau_s$, which were designed to control the sigmoidal shape in both positive and negative regimes of the input. Specifically, the shape and parameterization of the constraint classifier was intended to improve the convergence of gradient-based optimization by allowing for more significant gradients for satisfying inputs to the classifier $\Psi$ and less gradient information for inputs that violate constraints $^1$. For visualization, recall the right plot for $\tau_b = 10, \tau_s = 0.5$ in Fig. 3.1.

![Figure 4.7: Heat maps of the constraint function $c_S$ for different parameterizations of $(\tau_b, \tau_s)$: (3, 1), (15, 1), (3, 0.5), and (15, 0.5) respectively, left to right.](image)

$^1$Recall that inputs that satisfy constraints $g(x, y)$ of the infeasible set $S$ are in fact results of violating BNN parameters while violated constraints imply feasibility in the input-output space and thus satisfying BNN parameters.
This set of experiments is designed to test the specific modeling choices of the classifier $\Psi_{\tau_b, \tau_s}(z)$ in $c_S(x, W)$ while holding everything else constant. The choice of base and scale parameters $\tau_b, \tau_s$ is shown to be in fact significant to the performance of variational inference methods and the speed of convergence of gradient-based optimization techniques. Using the same experiment design and inference settings as in section 4.2.1 previously, the rate of convergence of the constrained variational objective, the sampled expectation of the constraint function, and the value of the unconstrained ELBO are qualitatively compared during iterations of Bayes by Backprop [13] using ADAM [15], for different parameterizations of the constraint function $c_S(x, W)$. The choices of $\tau_b$ and $\tau_s$ result in different landscapes of $c_S$ in

![Graphs showing variational objective, expected constraint function values, and ELBO over iterations](image)

Figure 4.8: Variational objective, expected constraint function values, and ELBO over the iterations of Bayes by Backprop. The choice of $\tau_b = 15$ outperformed $\tau_b = 3$ absolutely in terms of the obtained constrained variational objective but $\tau_b = 3$ reached the best unconstrained ELBO nevertheless.
the input-output space, as summarized in Figure 4.7. Observe that higher values of \( \tau_b = 15 \) result in a sharper and more definite steepness in the increase from the satisfying side of the input-output space (black) while lower values of \( \tau_s = 0.5 \) make violating regions less plateau-like (white). This agrees with the plots of the guiding classifier \( \psi_{\tau_b, \tau_s} \) in Fig. 3.1.

Figure 4.8 summarizes the results of the comparison during inference. Although recommendations cannot be derived in general from this and do most certainly depend on the size and type of the constrained set \( S \), we see that the tuning of the shape parameters \( \tau_b \) and \( \tau_s \) can indeed positively affect the speed of convergence and violation. Interestingly, here the the choice of \( \tau_b = 15 \) outperformed \( \tau_b = 3 \) absolutely in terms of the obtained constrained variational objective in the top-left plot of Fig. 4.8 but nonetheless the opposite setting of \( \tau_b = 3, \tau_s = 0.5 \) reached the best unconstrained ELBO. The better objective values reached by \( \tau_b = 15 \) independently of \( \tau_s \) can be explained by the observation that \( \tau_b = 3 \) never reaches the expected constraint function values of \( \tau_b = 15 \) in the top-right plot of Fig. 4.8. Nevertheless, this contrasts the ablation experiments in section 4.2.1 on the effect of \( \gamma \), which found that better values of the constrained variational objective implied higher unconstrained ELBO values when comparing standard with Constrained BNNs. Here, we observe that the choice of \( \tau_b \) and \( \tau_s \) in \( \psi_{\tau_b, \tau_s} \) can in fact affect the way this effect plays out. Worse minimization objectives as for \( \tau_b = 3 \) could result in better ELBO values, which should be taken into consideration when tuning the hyperparameters of the prior.

4.3 Additional Considerations

Having applied and evaluated BNNs and the constraint prior extensively in the above examples and studies, there are a few additional considerations that ought to be discussed before moving on to the advanced inference methods and results in chapter 5.
High-dimensional input and output spaces

Although ultimately outside of the scope of this thesis, Constrained Bayesian neural networks will be used primarily in high-dimensional feature or input spaces when applied in real-world scenarios, sometimes even computing high-dimensional outputs, such as in classification problems. BNNs generally are less scalable than standard neural networks when increasing the number of units and layers as a result of computationally-expensive inference methods, but they have still been shown to be effective in settings of high-dimensional input and output spaces [21].

The extensions made by Constrained BNNs through the constraint prior and constraint function do not pose any additional difficulties in dealing with high-dimensional inputs and outputs. In fact, the only point where the higher dimensionality of the input-output space comes into play is the constraints $g(x, y) \leq 0$ that define the constrained set $S$ for $c_S(x', W)$; see Eq. 3.2. Since $g(x, y)$ simply needs to be evaluated in $c_S(x', W)$ at $x = x'$ and $y = \phi(x; W)$ to compute the constraint violation, the higher dimensions do not affect inference through the constraint prior, and in fact even more complicated classes of constrained sets are conceivable in these cases. Hence, being theoretically unaffected by the undesirable properties of high dimensional spaces – or “curse of dimensionality” [22, 23] as discussed in more detail later on – is another strength of the constraint prior and makes Constrained BNNs just as applicable as standard BNNs in real-world applications.

Conflicting observations and constraints

When imposing constraints onto the distributions modeled by BNNs using the constraint prior, it is possible for given subsets of the observations $D$ to violate the infeasible set $S$ implied by the constraints. Depending on the application, the desired way of handling these cases might vary, but generally data points that violate a priori imposed expert constraints could result from high levels of noise in the observations and in that case ought to be overwritten by constraints.
Figure 4.9: Two ways of handling conflicting observations and constraints when using Constrained BNNs. The enforcement parameter $\gamma$ determines whether violating data points dominate more loosely defined prior constraints (left; $\gamma = 10$) or whether they get overwritten by the constraints (right; $\gamma = 10,000$).

As one might expect from the ablation studies in section 4.2.1 on the enforcement parameter $\gamma$, the choice of $\gamma$ allows us to model different kinds of behavior. Introducing a violating data point at $(-\frac{1}{2}, 4)$ to form

$$\mathcal{D}_f' = \mathcal{D}_f \cup \{(-\frac{1}{2}, 4)\}$$

in the simplest linear bounding setting of Eq. 4.6, Figure 4.9 shows that a Constrained BNN with low $\gamma = 10$ is fitting the violating data point while a more strictly enforcing Constrained BNN with high $\gamma = 10,000$ is ignoring the observation. The simple variational inference algorithm Bayes by Backprop was once again used jointly with ADAM optimization using the same hyperparameters and initializations as in the above experiments with variational inference. Depending on whether the we assume high observational noise or only loose prior knowledge, both ways of handling conflicting constraints and observations can be employed by Constrained BNNs via the tuning of $\gamma$. In practice, it might be helpful to increase the focus of the sampling distribution $\pi_S(x)$ on constrained input regions that are more exposed to violating observations, to ensure the enforcement of the desired posterior behavior.
Chapter 5

Advanced Posterior Inference and Results

The evaluation and experiments of chapter 4 established qualitatively and quantitatively that the constraint prior defined in this thesis is able to successfully impose simple and well-behaved constraints onto the functions modeled by a Bayesian neural network. Furthermore, it was shown that the modeling choices of Constrained BNNs were both specific and necessary to optimize constraint enforcement and inference. Overall, the constraint prior did not negatively affect any qualitative objectives or advantages of variational inference and could even improve the ELBO values reached on average during inference, essentially making Constrained BNNs a universal extension to BNNs for all practical use cases.

However, as briefly mentioned in chapters 3 and 4, the constraint formulation allows for even more sophisticated constrained regions that block off closed regions in the input-output space, leading to the possibility of splitting behavior of the posterior predictive around the enclosed region when no data is available. In section 3.1.1, these constrained regions are defined as infeasible sets with separated feasible outputs, as these constraints can allow the output regions given certain inputs to be separated, i.e. formally decomposable into two disjoint nonempty open sets. Intuitively and informally, such constraints would allow the functions
modeled by a BNN to go more than one way to satisfy a constraint.

Real-world scenarios where BNN modeling is advantageous and prior expert functional knowledge available can often demand enclosed constrained regions. For instance, consider the general task of safe robot motion planning in an environment with obstacles [24]. If we were to use a Constrained BNN to model a robot's path in the environment and incorporate the locations of the obstacles or other humans and robots a priori, the constrained regions in the output space could be enclosed and separate the feasible space with respect to some inputs, thus warranting a splitting posterior predictive behavior. Constrained BNNs could then enable safe planning and interaction with the real-world environment.

This chapter will extend the standard cases discussed in chapter 4 by introducing novel and advanced inference methods and analyzing the splitting posterior results of infeasible sets with separated feasible outputs. More specifically, in section 5.1 I both derive a novel variational inference method and present an existing state-of-the-art sampling approach with a custom preprocessing procedure suitable for complex BNN posteriors. The conventional HMC [16] and Bayes by Backprop [13] algorithms used in chapter 4 have several weaknesses that make them unsuitable to express multimodal behavior. In section 5.2 I then finally discuss fully general experiments using infeasible sets with separated feasible outputs in relation to their multimodal posterior results and splitting predictive distributions.

5.1 Advanced Inference Methods

What is the difficulty of infeasible sets $S$ with separated feasible outputs, and why do we need more sophisticated inference methods to converge to the true posterior distribution of the BNN? Informally, if the infeasible regions in the input-output space block off fully enclosed regions of the domain – resulting in separated feasible outputs given certain inputs – then the function modeled by a neural network could go on either side of the constrained region, assuming we have no local data
available. Hence, in a Bayesian neural network, the distribution over functions should place probability mass on either side of a constrained region.

![Figure 5.1: Example of an infeasible set $\mathcal{S}$ with separated feasible outputs. Depicted is the heat map of the constraint function $c_\mathcal{S}$ for a single box with $\mathcal{S} = \{x, y \mid -1 \leq x, y \leq 1\}$.](image)

Take as an example $\mathcal{S} = \{x, y \mid -1 \leq x, y \leq 1\}$ whose corresponding constraint function is illustrated in Figure 5.1. Here, for all $-1 \leq x \leq 1$, it is possible for either $y \geq 1$ or $y \leq -1$, i.e., the subset of feasible outputs is separated. The plot explains the need for the posterior predictive to place probability mass on either side of the box under certain circumstances. Unless there is data available in an ambiguous region, the correct posterior ought to model the possibility of functions going both ways around the constraint.

After running preliminary studies using the conventional variational inference and sampling algorithms Bayes by Backprop [13] and HMC [16], it became clear that neither of the methods is complex enough to capture the phenomenon of the “splitting posterior”. As laid out in detail in section 2.2.3, recall that posterior variational inference and sampling for BNNs is run in *weight space*, and the particles are thus very high dimensional for already medium-sized BNNs.
The conjecture I make is that the desired splitting posterior predictive implies a multimodal posterior in weight space. Due to the curse of dimensionality, the distance between modes loses their meaning and the sparsity of probability mass become extremely large [23]. Hence, both conventional methods used so far fail to capture more than a single significant BNN posterior mode. The term “curse of dimensionality” was originally coined by Richard Bellman in his book on control theory [22] and now commonly refers to any issues in optimization or inference arising from the properties of high-dimensional spaces.

On the one hand, the particles generated by a sampling-based approach such as HMC correspond to actual neural network weights of the BNN posterior \( p(W|D) \). Then, the energy barrier of moving between the various very distant modes of probability density becomes almost insurmountable and makes mode traversal very unlikely over iterations of HMC in these high dimensions [16]. On the other hand, since the mean-field variational approximation \( q(W; \lambda) \) of Eq. 2.7 as used in Bayes by Backprop [13] places fully-factored and unimodal Gaussian distributions over each weight’s parameterization, a multimodal posterior of the BNN weights over the data cannot be matched appropriately. Thus, two more sophisticated inference methods – one novel variational approach and one sampling method – are derived from scratch and elected from the state-of-the-art literature, respectively. The specific goal is to enable multimodal approximation as well as mode traversal in high dimensions, to be able to successfully capture the conjectured splitting posterior predictive.

5.1.1 Variational inference with mixture of Gaussians using Gumbel-Softmax

In concluding that the mean-field assumption in Bayes by Backprop is not complex enough to allow approximations of multimodal posteriors, the straightforward next step is to introduce a multimodal variational family.

In Nonparametric variational inference, Gershman et al. propose a uniform
mixture of isotropic Gaussians for variational inference [25]. However, preliminary studies with standard BNNs showed that this approximating family is not able to infer the qualitatively correct posterior. Following their definition and optimization of $q(W; \lambda) = \frac{1}{K} \sum_{k=1}^{N} \mathcal{N}(W; \mu_k, \sigma_k^2 \mathbf{I})$ resulted in poor posterior distributions, often appearing as narrow strings in function space and lacking the correct uncertainty in out-of-sample regions, even though the splitting behavior was observed.

The lacking complexity in dealing with Bayesian neural network posteriors is likely due to several fundamental simplifications. Most importantly, the authors only optimize a scalar $\sigma_k^2$ for the variance of each trivial isotropic Gaussian component and use first and second-order Taylor expansions to approximate the likelihood term $\log p(W, D)$. These lower-order likelihood approximations can not only be drastic and inexact, but also do not offer a straightforward way of improving the variance of the ELBO gradient estimate. In methods such as Bayes by Backprop [13], expectations are replaced by unbiased Monte Carlo estimates which can be improved by simply increasing the number samples of the random variable the expectation is taken over. In addition, even though a closed-form approximation is used, the second order Taylor expansion requires third-order derivatives in gradient-based optimization, which are very inefficient to compute.

Therefore, I propose using a fully general Gaussian mixture distribution for variational inference

$$q(W; \lambda) = \sum_{k=1}^{K} \pi_k \mathcal{N}(W; \mu_k, \Sigma_k) = \sum_{k=1}^{K} \pi_k \prod_{n} \mathcal{N}(w_n; \mu_{kn}, \sigma_{kn}^2)$$  \quad (5.1)$$

with $\lambda = \{\pi, \mu_{1:K}, \Sigma_{1:K}\}$, $\sum_{k=1}^{K} \pi_k = 1$, vectors $\pi$ and $\mu_k$ and diagonal covariance matrices $\Sigma_k = \text{diag}(\sigma_k^2)$. The added complexity will allow for sufficient approximation of the BNN posterior to capture qualitatively correct uncertainty as well as multimodality.
As in Bayes by Backprop [13], the optimization of the ELBO with respect to the variational parameters is accomplished using the reparameterization trick, which allows the use of unbiased Monte Carlo gradient estimates and doesn’t need any simplifying first or second-order approximations of the likelihood. The difficulty lies in reparameterizing the discrete Categorical random variable Cat(\(\pi\)) implicit in the generative process of the mixture of Gaussians. In this approach, the reparameterization of the Categorical distribution is solved using the Gumbel-Softmax trick [26], a relaxation that defines a deterministic transformation of random noise onto the probability simplex and converges to a one-hot encoded categorical distribution in the limit.

Following Jang. et al. [26] and Maddison et. al [27], the Gumbel-Max trick tells us how to sample from categorical distribution using uniform random noise. In particular, given \(U_k \sim \mathcal{U}(0,1)\) for \(k = 1 \ldots K\), then if

\[
X = \arg \max_k (\log \pi_k - \log(-\log U_k)) = \arg \max_k (\log \pi_k + G_k)
\]

then \(X \sim \text{Cat}(\pi)\). The name follows from the fact that the random variables \(G_k = -\log(-\log U_k)\) have the Gumbel(0, 1) distribution. Now, using the softmax operator and a temperature parameter \(\tau\), we can define the smooth relaxation of Eq. 5.2 as

\[
X_k = \frac{\exp\left(\left(\log \pi_k + G_k\right)/\tau\right)}{\sum_{j=1}^{K} \exp\left(\left(\log \pi_j + G_j\right)/\tau\right)} = \text{softmax}\left(\left(\log \pi_{1:K} + G_{1:K}\right)/\tau\right)_k
\]

which turns the discrete categorical random variable into a continuous vector on the probability simplex. As \(\tau \to 0\), we have \(X \sim \text{Cat}(\pi)\), one-hot encoded as a discrete vector [26]. The distribution of the relaxed variable \(X\) is referred to as the Gumbel-Softmax distribution [26] or the concrete distribution [28].

Recall that we want to optimize the evidence-lower bound of Eq. 2.9 as a means to maximize the marginal likelihood of the data. Using the Gumbel-Softmax trick,
we can now directly apply the reparameterization trick as in Bayes by Backprop

$$ELBO(\lambda) = \mathbb{E}_{q(W;\lambda)} \left[ \log p(W, D) - \log q(W; \lambda) \right]$$

relaxed to

$$\mathbb{E}_{\xi \sim \mathcal{N}(0,1) \mid \epsilon \sim \text{Gumbel}(0,1)} \left[ \log p(g(\lambda, \xi, \epsilon), D) - \log q(g(\lambda, \xi, \epsilon); \lambda) \right] \quad (5.4)$$

where $g(\lambda, \xi_{1:K}, \epsilon_{1:K:1:N})$ with $\xi_k \sim \text{Gumbel}(0,1), \epsilon_{kn} \sim \mathcal{N}(0,1)$ returns the relaxed sample $W$ from the mixture of Gaussians as

$$g(\lambda, \xi, \epsilon)_n = \sum_{k=1}^{K} \text{softmax} \left( \log \pi_{1:K} + \xi_{1:K} \right)_k \mathcal{N}(w_n; \mu_{kn}, \sigma_{kn}^2)$$

$$= \sum_{k=1}^{K} \text{softmax} \left( \log \pi_{1:K} + \xi_{1:K} \right)_k (\mu_{kn} + \sigma_{kn} \epsilon_{kn}) \quad (5.5)$$

Hence, we reparametrize samples from the mixture by taking weighted averages of the samples of all mixture components using relaxed categorical one-hot vectors via the Gumbel-Softmax trick, which will weigh one mixture component much stronger than the rest. As $\tau$ goes to zero, this is equivalent to a true sample from the Gaussian mixture distribution of Eq. 5.1. By using the relaxation, we can still apply the reparameterization trick to obtain Monte Carlo gradient estimates that are slightly approximated and perform optimization using any conventional gradient-based optimization method, as in Bayes by Backprop.

It should be noted that Miller et al. [29] also developed a method solving the reparameterization trick for mixture distributions by rewriting the expectation over the variational distribution as a weighted sum of component expectations. However, contrary to this approach, they approximate the target distribution by adding individual components sequentially during optimization rather than optimizing all components jointly.

In practice, we can maximize the entropy term $\mathbb{E}_{q(W;\lambda)} [-\log q(W; \lambda)]$ of the ELBO via a closed-form lower bound established for mixture of Gaussians [30]
and used in simplified form by Gershman et al. [25].

\[
\mathbb{E}_{q(W; \lambda)}[-\log q(W; \lambda)] \geq -\sum_{k=1}^{K} \pi_k \log \left( \sum_{j=1}^{K} \pi_j \mathcal{N}(\mu_k; \mu_j, \Sigma_k + \Sigma_j) \right) 
\] (5.6)

The algorithm is summarized in Alg. 2 and results on complicated constraint settings using Constrained BNN posteriors are reported in the following section.

In practice, \( \pi_k \) is parameterized as \( \pi_k = \text{softmax}(\pi'_k) \) and \( \sigma = \log(1 + \exp(\sigma')) \) as previously in Bayes by Backprop to satisfy the simplex and non-negative domain of the respective variational parameters.

**Algorithm 2** Constrained Mixture of Gaussian VI using Gumbel-Softmax

**Input:** \( \lambda_{\text{init}}, \) learning rate schedule \( \alpha, \) data \( D, \) sampler \( \pi_S(\mathbf{x}'), \gamma, K \)

**Output:** optimized variational params. \( \lambda^* \) respecting constraints \( S \)

1. def softmax(z) = \[\exp(z_k) / \sum_j \exp(z_j)\]_k
2. while \( f(W) \) not converged do
3. \( \pi, \mu, \sigma \leftarrow \) unpack \( \lambda \)
4. \( \xi_{1:K} \sim \text{Gumbel}(0, 1) \)
5. \( \epsilon_{1:K:1:N} \sim \mathcal{N}(0, 1) \)
6. \( W_n \leftarrow \sum_{k=1}^{K} \text{softmax}(\log \pi + \xi)_k (\mu_{kn} + \sigma_{kn} \epsilon_{kn}) \) \( \triangleright \) reparam. trick
7. \( H \leftarrow -\sum_{k=1}^{K} \pi_k \log \left( \sum_{j=1}^{K} \pi_j \mathcal{N}(\mu_k; \mu_j, \Sigma_k + \Sigma_j) \right) \) \( \triangleright \) entropy bound
8. \( \{\mathbf{x}^{(j)}\}_{j=1}^{B} \sim \pi_S(\mathbf{x}') \)
9. \( f(W) \leftarrow -H - \log p(D|W) - \log p(W) + \gamma \frac{1}{B} \sum_{j=1}^{B} c_S(\mathbf{x}^{(j)}, W) \)
10. \( \lambda \leftarrow \lambda - \alpha \nabla_{\lambda} f(W) \) \( \triangleright \) gradients with auto-differentiation
11. update \( \alpha \)
12. end while

### 5.1.2 Generalized Darting MCMC

Having formulated an efficient variational inference algorithm that is complex enough to approximate multimodal Constrained BNN posteriors, it is also beneficial to present a sampling-based method that entails theoretical guarantees of converging to the true posterior distribution and forms a baseline of comparison.

**Generalized Darting MCMC** [17] is an algorithm by Sminchisescu and Welling that attempts to resolve the inability of standard Markov Chain Monte Carlo samplers to mix and traverse between multiples modes of the target distribution. With
prior knowledge of the mode locations, mode-hopping moves that satisfy detailed balance enable jumping between different high-probability regions of the probability support. Since the procedure applies to any MCMC method and results of chapter 4 were promising, Hamiltonian Monte Carlo \[16\] is used in combination with Generalized Darting MCMC in the application with BNN posteriors. The main algorithm is summarized in Algorithm 3.

### Algorithm 3 Generalized Darting HMC (uniform region weighting and sampling)

- **Input:** target $\theta(W|D; S)$, $S$ samples, $K$ darting regions, $\epsilon$, $p_{\text{check}}$
- **Output:** $S$ particles from the target distribution

1. $M^{(1:K)} \leftarrow$ modes found through preprocessing Algorithm 4
2. Initialize $W^{(1:S)}$ randomly
3. **while** less than $S$ samples collected **do**
   4. $s \leftarrow$ number of samples accumulated in $W^{(1:S)}$
   5. $u \sim U(0, 1)$
   6. **if** $u > p_{\text{check}}$ **then**
      7. perform one HMC step w/ leapfrog method; see \[16\]
   8. **else**
   9. $N \leftarrow$ number of darting regions $M$ containing $W^{(s)}$
   10. **if** $N > 0$ **then**
      11. $M' \leftarrow$ new region sampled uniformly at random from $M^{(1:K)}$
      12. $W' \leftarrow$ new location inside $\epsilon$-neighborhood $M'$ uniformly at random
      13. $N' \leftarrow$ number of darting regions $M$ containing $W'$
      14. $p_{\text{accept}} \leftarrow \min \left\{ 1, \frac{N \cdot \theta(W'|D; S)}{N' \cdot \theta(W'|D; S)} \right\}$
      15. $v \sim U(0, 1)$
      16. **if** $v < p_{\text{accept}}$ **then**
         17. $W^{(s+1)} \leftarrow W'$ (accept)
      18. **else**
         19. $W^{(s+1)} \leftarrow W^{(s)}$ (reject)
   20. **end if**
21. **end if**
22. **end if**
23. **end while**
24. **return** $W^{(1:S)}$

The procedure relies on a preprocessing step that finds the modes of the posterior density, which are later used by the main algorithm for mode-hopping. Note that this preprocessing step only needs to be computed once for each experimental setting, and the actual MCMC algorithm runs on top of and independent of the time it takes to run the mode exploration. To make Generalized Darting HMC suitable for BNN posterior inference, I designed the preprocessing procedure in
Algorithm 4 to explore the posterior density in weight space and return a set of modes as distinct as possible.

The idea is to use random initializations in weight space to climb to different prevalent modes in the posterior density using gradient ascent. After exploring a specified number of modes, the best mode positions are reduced to mode representatives that are as unique as possible using clustering. Clustering is done using the hierarchical agglomerative clustering algorithm (HAC), which creates clusters of all sizes bottom-up and often produces better clustering [31]. In addition, this allows determining more easily what cluster sizes might be sufficient or appropriate to express all mode representatives for Darting HMC.

**Algorithm 4 Mode Identification Preprocessing**

**Input:** target $\tilde{p}(W|D; S)$, $K$ darting regions, $M$ search attempts, $\delta$, learning rate schedule $\alpha$, $s \in (0, 1)$, $d$ cluster distance metric

**Output:** $K$ representative modes of the target distribution

1: Randomly initialize $W^{(1:M)}$
2: for $m = 1 \ldots M$ do  \hspace{1cm} $\triangleright$ randomly restarted grad. ascent
3: \hspace{1cm} while change in log $\tilde{p}(W^{(m)}|D; S) < \delta$ do
4: \hspace{2cm} $W^{(m)} \leftarrow W^{(m)} + \alpha \nabla_w \log \tilde{p}(W^{(m)}|D; S)$
5: \hspace{2cm} Update $\alpha$
6: \hspace{1cm} end while
7: end for
8: $W^{(1:s \cdot M)} \leftarrow$ top $|s \cdot M|$ modes $W^{(1:M)}$ according to $\tilde{p}(W|D; S)$
9: for $|s \cdot M| - K$ times do  \hspace{1cm} $\triangleright$ HAC clustering
10: \hspace{1cm} $i, j \leftarrow$ closest pair of clusters $d(W^{(i)}, W^{(j)})$
11: \hspace{1cm} $W^{(i)} \leftarrow$ merge($W^{(i)}, W^{(j)}$)
12: \hspace{1cm} delete $W^{(j)}$ from $W$
13: end for
14: reduce every cluster $W^{(j)}$ to its mediod
15: return $W^{(1:K)}$

Furthermore, I use theoretical results about the degeneracy of Euclidean distances in high dimensions or the “curse of dimensionality” [23, 22] to choose a more meaningful distance for the high-dimensional BNN weight space. Specifically, Aggarwal et al. [32] show that fractional distance metrics $d_f$, i.e. $L_f$ norms
with

\[ d_f(u, v) = \left( \sum_{m=1}^{M} |u_m - v_m|^f \right)^{1/f} \]  \hspace{1cm} (5.7)

and \( f \in (0, 1) \) are more contrasting in high dimensions and more effective in clustering than the conventional and interpretable Manhattan (\( L_1 \)) or Euclidean (\( L_2 \)) distances with \( f \geq 1 \). Using preliminary experiments and Aggarwal et al.’s results \[32\], the \( L_f \) norm with \( f = 0.3 \) was found to be most effective in clustering the density modes found via gradient ascent.

Even though more sophisticated exploration strategies of the probability density are conceivable, Algorithm 3 in combination with Algorithm 4 produced sufficient performance in demonstrating the positive results summarized later in section 5.2. In doing so, the choice of using the more informative \( L_f \) norms with \( f \in (0, 1) \) in high dimension in combination with clustering proved to be essential.

### 5.2 Splitting Posterior Predictives

The methods of Gaussian mixtures variational inference and Generalized Darting MCMC \[17\] introduced above reflect the strengths and weaknesses of their respective algorithm class – sampling and optimization-based inference. As mentioned in previous chapters, the computational efficiency of the former trade off with the expressivity and convergence guarantees of the latter. Both introduced methods will be discussed and contrasted in relation to the more challenging constrained sets \( S \) with separated feasible outputs, and the Generalized Darting HMC gold standard will serve as a baseline of evaluating the newly derived Gaussian mixture variational inference. The more sophisticated inference was chosen to specifically capture the multimodality that is expected to result from these constrained regions, the task that the conventional inference methods used in chapter 4 failed at, while providing both a theoretically convergent as well as computationally efficient method.
5.2.1 Infeasible sets with separated feasible outputs

Consider the enclosed box setting visualized earlier in Figure 5.1 and the ground truth function \( g(x) \) with observations \( D_g \) introduced in chapter 4. We assume knowing a priori that \( S = \{x, y| -\frac{1}{2} \leq x \leq \frac{1}{2}, 0 \leq y \leq 2\} \) is infeasible for the underlying process \( g(x) \) we want to model. At the same time, as previously, no observations are available for \( x \in (-1, 1) \). Thus, the posterior predictive should place probability mass on either side of the box constraint and exhibit the expected posterior behavior.

Figure 5.2 shows the posterior predictive distributions for the single enclosed constrained region. The first row of plots contrasts the efficacy of conventional

![Figure 5.2: Posterior predictives of Constrained BNNs on the enclosed box setting using conventional and advanced sampling and variational inference methods. Both conventional methods – HMC (top-left) and Bayes by Backprop (bottom-left) – fail to capture the splitting posterior predictive due to the multimodal posterior \( \tilde{p}(W|D; S) \). Advanced inference methods of section 5.1 – Generalized Darting HMC (top-right) and Gaussian mixture variational inference using Gumbel-Softmax (bottom-right) – exhibit the desired nontrivial distribution.](image)

The advanced inference methods succeed at showing the desired posterior predictive. In the absence of data, the predictive distribution splits around the constrained region and thus enables the possibility of a more general set of feasible neural network functions. The plots indicate that Generalized Darting HMC is superior in smoothly capturing posterior of the Constrained BNN. When analyzing the posterior predictive of the Gaussian mixture approximation, we can make out individual mixture components in the approximated posterior as bundles of sampled functions in the plot. Thus, even though the splitting behavior is inferred, the BNN predictive distribution is not as natural as for Generalized Darting HMC. However, the speed of optimization when using only a small number of modes in Mixture of Gaussian VI is significantly higher compared to sampling-based Darting HMC, and the approximation can be seen as quite good for all practical purposes.

After repeated tuning, the hyperparameters chosen for Generalized Darting HMC were $L = 20$ steps of step size 0.005, and a thinning rate of 5 for the standard HMC steps. No burn-in samples were taken as the Markov chain was initialized inside a preprocessed darting region. The preprocessing procedure explored 30 modes with individual weights randomly initialized as $w \sim 3 \max \{1, |r|\} s$ with $r, s \sim \mathcal{N}(0, 1)$ and reduced them to 6 representative darting regions using the described algorithm with $L_{0.3}$ norm. $p_{\text{check}}$ was 0.03 and the darting region radius was $\epsilon = 15$.

Gaussian Mixture variational inference using Gumbel-Softmax was run for 3 restarts and $N = 10$ mixture components, and the variational parameters were initialized as $\log \pi_k = 1$, $\mu_n \sim \mathcal{N}(0, 9)$ and $\log \sigma_n = -10$. The algorithm was run for 20,000 steps, and the Gumbel-$\tau$ parameter was set to 0.1. The classifier shape of the constraint function was set to $\tau_b = 5.0, \tau_s = 0.5$. 
Effect of new observations on the splitting posterior

When imposing constraints with separated feasible outputs and lacking relevant data, the true predictive distribution is expected and shown to split around the corresponding region. Conversely, if an additional observation in the constrained input region is observed, the likelihood term of the true posterior should result in placing almost the entire probability mass on the correct side of the constraint. Hence, extending the above single box constraint experiment, assume we add one observation to our data set. Specifically, let

$$D'_g = D_g \cup \{(0, 4)\}$$

Figure 5.3: Adding an observation in the splitting posterior setting. Posterior predictives using Generalized Darting HMC for $D_g$ (left) and $D'_g$ (right). The additional data point correctly bundles the posterior over functions above the constrained set.

Generalized Darting HMC is run on both $D_g$ and $D'_g$ to assess the ability for the posterior to “pick a side” under substantial evidence of where the modeled process would go. Indeed, Figure 5.3 shows that the additional data point influences the posterior predictive with the constraint prior sufficiently through the model likelihood term to bundle the predictive distribution. The experimental setup and hyperparameters for Generalized Darting HMC were the same as for the single box constraint experiment in Figure 5.2 and were held constant across both experiments with observations $D_g, D'_g$. 

64
5.2.2 Arbitrary and multiple constrained regions

After comprehensive evaluation, the constraint prior has been shown to exhibit the expected posterior features even in more challenging constraint settings. Variational inference with a Gaussian mixture family as well as specific preprocessing steps and amendments to Darting HMC made it possible to obtain the true posterior predictive for the highly complicated and nonlinear posterior over weights \( \tilde{p}(\mathcal{W}|\mathcal{D};\mathcal{S}) \) of the Constrained Bayesian Neural Network.

To unify the results obtained in chapters 4 and 5 and conclude the analysis of the introduced concepts of this thesis, consider a last setting of arbitrary and multiple constrained regions incorporated into a Bayesian neural network using the constraint prior. Modeling the observations \( \mathcal{D}_g \) of the ground truth process \( g(x) \), assume we have very specific prior constraint knowledge about the function we intend to model with the BNN. Let \( \mathcal{S} = \bigcup_i \mathcal{R}_i \) with the following subsets of input-output space \( \mathcal{R}_i \) known to be infeasible

\[
\begin{align*}
\mathcal{R}_1 &= \{ x, y \in \mathbb{R} \mid x + 3 \leq -y^2 \} \\
\mathcal{R}_2 &= \{ x, y \in \mathbb{R} \mid x - 3 \geq y^2 \} \\
\mathcal{R}_3 &= \{ x, y \in \mathbb{R} \mid -\frac{1}{2} \leq x \leq \frac{1}{2}, 0 \leq y \leq 2 \} \\
\mathcal{R}_4 &= \{ x, y \in \mathbb{R} \mid -\frac{3}{2} \leq x \leq \frac{3}{2}, y \leq -x^4 - 5 \}
\end{align*}
\]

Then, applying both Generalized Darting HMC as well as the newly derived Mixture of Gaussian variational inference using Gumbel-Softmax to the posterior of the Constrained BNN, we successfully capture the intended posterior predictive distribution that both retains all advantages of Bayesian modeling and incorporates our prior beliefs about the observations in function space. Figure 5.4 shows the constraint function and posterior predictive. The variational approximation found by Gaussian mixture variational inference is qualitatively very similar to the benchmark set by Generalized Darting HMC.

Gaussian mixture variational inference used the same hyperparameters and
number of mixture components as in the single box setting of the experiment in section 5.2.1 except here the constraint function shape was parameterized with $\tau_b = 5.0$ and $\tau_s = 2.0$ to help contrast the comparably small constrained region $\mathcal{R}_3$. Generalized Darting HMC was the same as for the single box constraint setting.

Figure 5.4: Posterior predictives of Constrained BNNs for the most general setting of multiple and arbitrary constrained regions. The top plot used Generalized Darting HMC with the custom preprocessing procedure; the bottom plot used Gaussian mixture variational inference with Gumbel-Softmax.
Chapter 6

Conclusion

Bayesian modeling has the opportunity of extending the great success of neural networks to safety critical domains and human interactions through Bayesian neural networks. The ability of modeling the uncertainty on predictions is necessary for allowing the use of deep learning in these domains, since precision can be mistaken for confidence and extrapolation is often mediocre. The advantage of Bayesian neural networks is not only the sense of predictive confidence but also the theoretical framework of incorporating prior beliefs about the modeled function in parameter space. However, BNNs are not yet equipped with the ability to encode interpretable prior knowledge in function space where most domain specialists have expert knowledge about the functions they model.

In this thesis, I introduced a principled and intuitive approach of imposing prior constraints in input-output space onto the distributions modeled by Bayesian neural networks. By formulating a general constraint prior in chapter 3, the novel method can be applied as a black box to any inference technique normally used for BNNs. Furthermore, the constraint prior did not negatively affect any qualitative objectives or advantages of inference and could empirically guide mean-field variational inference approaches to higher unconstrained ELBO values than standard BNNs.

Constrained Bayesian neural networks were shown to not only make the pre-
dictive behavior of BNNs more accurate but also enforce feasibility in the given
domains. The evaluations of chapter 4 and 5 showed qualitatively and quantita-
tively that the Constrained BNNs are able to successfully incorporate arbitrary
and complex yet interpretable constraints onto the functions they model, excelling
particularly in out-of-distribution regions.

Finally, novel posterior predictive distributions were obtained using an amended
version of a state-of-the-art sampling algorithm and a newly derived variational
inference algorithm which leveraged the expressivity of the Gaussian mixture dis-
tribution together with the Gumbel-Softmax and reparameterization trick.

Nevertheless, the conceivable extensions of this project are plenty. Most re-
cent breakthroughs in deep learning relied on special neural network architec-
tures such as convolutional neural networks (CNNs) or recurrent neural networks
(RNNs). If computer vision or natural language processing researchers have ac-
tionable knowledge in function space whose incorporation into the Bayesian setting
looks promising, it would be interesting to extend the framework of Constrained
Bayesian neural networks to these more complicated architectures and evaluate
their modeling performance.

Likewise, the experiments and evaluation of this work focus on pedagogical
small-scale examples that illustrate the use cases, practical applications, and diffi-
culties of the constraint prior, but a real-world application for example in robotics
is yet to be fully explored. The full and sufficient treatment of a realistic problem
at hand with actionable expert knowledge was ultimately outside the scope of this
thesis.

Ultimately, I hope that Constrained Bayesian neural networks can contribute
to making the empirical success of deep learning applicable and safe for real-world
human and society-level interactions in the future.
Bibliography


