BACHELOR

Unreliable Uncertainty

How Bayesian Non-identifiability Influences the Performance of Uncertainty Quantification Methods in the Context of Reinforcement Learning

Göbbels, Lieve J.L.

Award date:
2022

Department of Mathematics and Computer Science
Bachelor Data Science

Unreliable Uncertainty: How Bayesian Non-identifiability Influences the Performance of Uncertainty Quantification Methods in the Context of Reinforcement Learning

Bachelor End Project

Lieve Göbbels

Supervisors:
Dr. Maryam Tavakol
Soroush Ghandi, MSc

Final thesis

Eindhoven, June, 2022

Abstract

Recently, there has been an increasing demand for explainable Artificial Intelligence (AI) which aims to develop algorithms that are interpretable for humans by using a framework of ethical principles and societal rights and therewith increase reliability of such algorithms. The context of this study is reinforcement learning as this is one of the main areas of origin for many of these algorithms. One of the subproblems of developing more reliable AI systems is to accurately quantify a system's uncertainty, which can be done using Bayesian inference methods. In this study, the focus lays on assessing the influence of Bayesian non-identifiability as a source of misestimation of uncertainty on several Bayesian uncertainty quantification methods, namely Hamiltonian Monte Carlo (HMC), No U-Turn Sampling (NUTS), Metropolis, Automatic Differentiation Variational Inference (ADVI) and Stein Variational Gradient Descent (SVGD). Prior-posterior overlap is used as a measure of non-identifiability. This is analyzed qualitatively and quantitatively by proposing an algorithm for calculating the exact prior-posterior overlap and by providing prior-posterior density plots to qualitatively determine the extent of said overlap. Additionally, the performance of each method is assessed by extracting the best negative log-likelihood and by providing visualizations of the method's uncertainty. Results show that Bayesian non-identifiability has a clear negative influence on the methods' ability to accurately quantify uncertainty. However, even when Bayesian non-identifiability is eliminated, most of the methods are still not able to provide approximations that are accurate enough. As this negatively affects the optimization process of a reinforcement learning agent and therewith decreases reliability of the AI system, it is vital to investigate what influences the accuracy of such uncertainty quantification methods besides Bayesian non-identifiability.

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

In recent years, there has been an increasing demand for explainable Artificial Intelligence (AI) (Yalcin, 2020; Wells and Bednarz, 2021). Explainable AI aims to develop algorithms that are interpretable for humans and focuses on the three ethical principles of transparency, interpretability and explainability together with the societal right to explanation of algorithmic decision-making processes (Wells and Bednarz, 2021; Goodman and Flaxman, 2017). Although there are different reasons for this demand, they all relate to reliability of AI and machine learning algorithms. The potentially catastrophic consequences linked to low reliability of AI systems in sensitive areas like self-driving cars, are one of the reasons for the high demand for more explainable and more reliable AI (Yalcin, 2020; Turek; Abdar et al., 2021). Many of these systems originate from the area of reinforcement learning, which is the context of this study (Wells and Bednarz, 2021). Reinforcement learning refers to a self-learning agent, like a neural network, trying to find the optimal solution to a certain problem by trial and error.

In this study, the focus lays on uncertainty quantification within the field of reinforcement learning. Therefore, several Bayesian uncertainty quantification (UQ) methods, that is Hamiltonian Monte Carlo (HMC), No U-Turn Sampling (NUTS), Metropolis, Automatic Differentiation Variational Inference (ADVI) and Stein Variational Gradient Descent (SVGD), are assessed and compared both empirically and theoretically by conducting experiments and a literature review. The goal of this assessment and comparison is to identify the strengths and weaknesses of these methods regarding the Bayesian non-identifiability problem as well as their performance in a reinforcement learning context. Ideally, these ideas form a starting point for improving the performance of the analyzed uncertainty quantification methods and therewith increase reliability. If uncertainty is incorrectly estimated but nonetheless used for improving the decision-making process of the agent, it is likely this agent-optimization process is suboptimal as it is based on incorrect information. That is, if the uncertainty quantification method provides erroneous information on what parts of the reinforcement learning environment are unexplored, indicated by high epistemic uncertainty, gathering more data on the parts with assumed high epistemic uncertainty are unlikely to improve the reinforcement learning agent's decision-making process, as there is sufficient data available already. So, for optimizing the agent in an efficient way and making the model's explanations informative, therewith increasing reliability and explainability of the system, it is vital to provide accurate estimations of the uncertainty of the decisions made by the agent.

One of the main causes of misestimation of uncertainty is non-identifiability (Yacoby et al., 2021; Wang et al., 2021; Papamarkou et al., 2021). In a general sense, non-identifiability of a model means that there are two or more parametrizations that are observationally equivalent (Koopmans, 1949; Rothenberg, 1971). Then, it is not theoretically possible to learn the true values of a model's underlying parameters with an infinite number of observations. In other words,

with non-identifiability, different parameter values produce equivalent probability distributions of the observed variables (Koopmans, 1949; Rothenberg, 1971). There are many forms of the non-identifiability problem, including Bayesian or posterior non-identifiability, which is especially relevant for the type of neural network, that is a Bayesian neural network with latent variables (BNN-LV), used in this study. However, in the context of reinforcement learning the topic of non-identifiability is still underexposed. Hence, the central research question in this study is:

How does posterior non-identifiability in a Bayesian neural network with latent variables influence the performance of different sampling methods for uncertainty quantification in the context of reinforcement learning?

To answer this research question, the most important theory and concepts are first described in Chapter 2. This is followed by a description of the used methods and the experimental set-up, after which each experiment and the corresponding sub-questions are specified in Chapters 3 and 4 respectively. Then, in Chapter 5 important results of the experiments and answers to the experiment-specific research questions are provided. In the chapter that follows, these results are further analyzed and interpreted.

Chapter 2

Theoretical Framework

2.1 Reinforcement learning

Reinforcement learning, one of the three machine learning (ML) paradigms, is concerned with finding optimal or near-optimal solutions in an interactive framework. Reinforcement learning focuses on sequential decision-making in a specific environment. This process is based on the concept of a Markov Decision Process (MDP) (Yi et al., 2020). That is, the formal definition of an MDP is the tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, p(s_0), \gamma\}$, which includes a transition ($\mathcal{T}$) and reward function ($\mathcal{R}$) in the environment. The general idea is as follows: at each time step t, some state $s_t \in \mathcal{S}$ is observed and an action $a_t \in \mathcal{A}$ is chosen, after which the next state (s_{t+1}) and the associated scalar reward (r_t) are returned by the environment. Repeating this process leads to a trace. Since for each possible choice the rewards are not identical, a policy is created, which serves as a guideline for the decision-making process of the agent (Yi et al., 2020; Puterman, 2005; Abdar et al., 2021; Depeweg et al., 2019). That is, the goal is for the agent (decision-maker) to learn the policy (strategy) that maximizes the cumulative reward (feedback based on the agent's actions) (Yi et al., 2020; Puterman, 2005; Abdar et al., 2021; Depeweg et al., 2019).

In general, there are two reinforcement learning approaches: model-based and model-free. In model-based learning, the agent learns an approximation of the environment, which can then be used to determine which action to take next (Swazinna et al., 2022). In model-free learning on the other hand, the agent only learns the states it has been in and the actions taken so far and generalizes inference to unexplored states, so that when the agent revisits a certain state, it is likely to base its decision on the action that leads directly to a good outcome (Swazinna et al., 2022). In other words, it starts with an arbitrary policy which it then aims to improve in an iterative way without learning the actual dynamics in the environment (Yi et al., 2020; Puterman, 2005). Since model-based reinforcement learning learns about the environment, therewith increasing sample efficiency, less data is required to learn a policy, which is not the case for the model-free approach (Swazinna et al., 2022). This also allows for simulating a sequence of actions rather than performing them in the actual environment, which means that the learning process is more generalizable. So, model-based reinforcement learning learns to predict the dynamics of the environment instead of a single policy that leads to a good outcome. This knowledge can then also be used for solving other tasks in a similar environment (Yi et al., 2020; Puterman, 2005). However, model-based reinforcement learning is more prone to error and more computationally expensive compared to the model-free approach because it learns the policy as well as a model of the environment dynamics (Swazinna et al., 2022). Since the focus lays on quantifying the uncertainty of a predictive model that is at the basis of the agent's decision-making process, model-based reinforcement learning is more suitable as context for analysis, because model-free reinforcement learning does not base its

decisions on the learned approximation of the transition probabilities as it uses the ground-truth transitions of the environment.

Yi et al. distinguish between different types of model-based reinforcement learning to better identify the different challenges each type needs to cope with. In this study, the distinction of model-based reinforcement learning categories made by Yi et al. is used to further specify the context (Yi et al., 2020). Accordingly, the context is specified as model-based reinforcement learning with a learned model (Yi et al., 2020; Sutton, 1991). That is, both the model and the policy or value are learned. This differs from model-based reinforcement learning with a known model, which uses a predefined model to learn the policy or value (Yi et al., 2020; Silver et al., 2017).

2.2 Uncertainty quantification

In addition to reinforcement learning, the concept of uncertainty (quantification) is also fundamental to this study. Uncertainty in machine learning refers to the use of incomplete or flawed information which causes a predictive model to be unsure. To make machine learning models more explainable and reliable, it is crucial to construct an accurate representation of the model's uncertainty, which is often done by modeling the uncertainty using probability theory (Hüllermeier and Waegeman, 2021). In machine learning, the more conventional method of modeling the uncertainty in a single probability distribution is still frequently used (Hüllermeier and Waegeman, 2021; Abdar et al., 2021). However, in light of the demand for reliable models, it is increasingly advocated to distinguishing between different types of uncertainty (Senge et al., 2014; Kull and Flach, 2014; Varshney and Alemzadeh, 2016; Kendall and Gal, 2017). The most common classification is aleatoric versus epistemic uncertainty, but multiple classifications have been proposed in recent years (Kläs and Vollmer, 2018; Booker and Ross, 2011; Zhang et al., 2016). Kläs et al. argue that the aleatoric versus epistemic classification is potentially problematic for practical application because "their boundaries are not sharp" (Kläs and Vollmer, 2018). However, by providing a clear definition of both forms of uncertainty and applying this within the boundaries of each model this problem can be circumvented (Kiureghian and Ditlevsen, 2009). Hence, aleatoric uncertainty is defined as "the variability in the outcome of an experiment which is due to inherently random effects" and epistemic uncertainty as "the ignorance of the agent or decision maker" (Hüllermeier and Waegeman, 2021). Aleatoric uncertainty refers to the irreducible part of uncertainty because the source of this uncertainty is the stochastic dependency between instances and outcomes. Epistemic uncertainty, on the other hand, is reducible, because it emerges from model and approximation uncertainty, which both refer to the lack of knowledge about the most suitable predictor (Hüllermeier and Waegeman, 2021).

Bayesian uncertainty quantification

There are several Bayesian techniques for uncertainty quantification. All these techniques use Bayesian inference to update the probability of a certain hypothesis as more information becomes available (Depeweg et al., 2019; Abdar et al., 2021; Depeweg et al., 2018). More specifically, Bayesian Neural Networks (BNN) or Bayesian Deep Learning (BDL) are used to model the pre-

dictive uncertainty of a model. In other words, BDL allows for interpreting the model parameters. Additionally, these techniques are generally robust to overfitting and are trainable on both small and big data sets (Abdar et al., 2021; Kucukelbir et al., 2017). Some of the most widely used techniques are variations of Monte Carlo (MC) dropout, Markov chain Monte Carlo (MCMC), Variational Inference (VI), Bayesian Active Learning (BAL), Bayes by Backprop (BBB), and Variational Auto-Encoders (VAE) (Abdar et al., 2021). MC dropout uses dropout as regularization term to compute the predictive uncertainty, and hence is used for approximating the posterior inference (Gal and Ghahramani, 2016). MCMC also approximates inference, but uses a stochastic transition operator for which the outcome distribution converges to the exact posterior after repeating the process for a certain number of times (Kupinski et al., 2003). VI, on the other hand, is a method that approximates the posterior distribution over the weights of the BNN. So, VI methods transform the Bayesian inference problem to an optimization problem to train the neural network (Swiatkowski et al., 2020). BAL uses unlabeled samples to learn, while BBB aims to learn a distribution over the weights of the NN by utilizing unbiased gradient estimates of the cost function (Hossain et al., 2017; Blundell et al., 2015). Lastly, VAEs aim to map high-dimensional input samples to low-dimensional latent variables and approach the learning representations for high-dimensional distributions as a VI problem (Ghosh et al., 2019).

Two examples of variational inference methods are Automatic Differentiation Variational Inference (ADVI) and Stein Variational Gradient Descent (SVGD). ADVI is a subclass of mean-field variational inference, where the goal is to approximate the posterior distribution by the variational posterior. Both of these distributions are normal distributions, such that the approximation is a mean-field approximation (Kucukelbir et al., 2016; Roeder et al., 2016; Kingma and Welling, 2014). SVGD, on the other hand, is based on Kernelized Stein Discrepancy, where the main idea is to move noisy particles so that they fit the target distribution as close as possible (Liu and Wang, 2016; Liu et al., 2017). While ADVI's objective is to maximize the Evidence Lower Bound (ELBO), SVGD's objective is to minimize the Kullback-Leibler divergence (Kucukelbir et al., 2016; Liu and Wang, 2016). So, both these methods have an optimization objective. Contrarily, MCMC methods like Metropolis, No-U-Turn Sampler (NUTS), and Hamiltonian Monte Carlo (HMC) use direct sampling of the posterior (Monnahan et al., 2017; Hoffman and Gelman, 2014). Metropolis simulates a stationary distribution π for each chain. Then, at each step in the chain, a new state is accepted or rejected depending on a dynamically calculated probability (Gundersen, 2019). Since HMC adopts physical system dynamics to propose future states allowing faster convergence, it is seen as more efficient than Metropolis. However, NUTS is considered the superior MCMC sampler, because it is a more robust version of HMC as it performs automatic tuning and therewith avoids random walk altogether (Monnahan et al., 2017; Hoffman and Gelman, 2014).

According to Depeweg et al., there is one major drawback to the standard Bayesian modeling approach: "Supervised learning methods that utilize Bayesian modeling, such as GPs or BNNs, only model the epistemic form of uncertainty [...] [T]o our knowledge there is no method that both models epistemic uncertainty via a Bayesian modeling approach and can model complex noise patterns with a high level of generality"(Depeweg et al., 2019). Depeweg et al., therefore propose a novel BNN, namely BNN with latent variables (BNN-LV), which allows for modeling and distinguishing between epistemic and aleatoric uncertainty. This is done by approximating the entropy, which represents a quantity of inherent variation. Entropy is approximated because

the posterior in the BNN-LV architecture is of unknown form.

2.3 (Bayesian) non-identifiability

Even though the BNN-LV architecture is considered an improvement of the standard BNN's, the issue of non-identifiability remains (Yacoby et al., 2021; Wang et al., 2021; Shi et al., 2017; Kurle et al., 2021; Pourzanjani et al., 2017; Papamarkou et al., 2021; Romeijn and Williamson, 2018). As described earlier, in a general sense, non-identifiability of a model means that there are two or more parametrizations that are observationally equivalent (Koopmans, 1949; Rothenberg, 1971). It is then theoretically impossible to learn the true values of a model's underlying parameters with an infinite number of observations. In other words, with non-identifiability, different parameter values produce equivalent probability distributions of the observed variables (Koopmans, 1949; Rothenberg, 1971).

One subproblem is non-identifiability of the likelihood, meaning that the likelihood function is unidentifiable due to different parameters being observationally equivalent (Allman et al., 2008; Wechsler et al., 2013). So, parameters $\theta_1,\ \theta_2 \in \Theta$ with $\theta_1 \neq \theta_2$ generate observationally equivalent likelihoods $L:\ L(\theta_1) = L(\theta_2)$ (Schmidt et al., 2020; Cole, 2020). The cause of this form of non-identifiability can be prescribed to the architecture of Bayesian neural networks themselves, making it hard to completely eliminate the problem: the weights of the network can be swapped internally without changing the likelihood (Cole, 2020; Shi et al., 2017; Kurle et al., 2021).

Another form of non-identifiability is Bayesian or posterior non-identifiability, which refers to non-identifiability of the posterior distribution of the model (Wang et al., 2021; Cole, 2020). That is, there are equivalent posterior distributions for different parameter values: $\theta_1,\ \theta_2 \in \Theta$ with $\theta_1 \neq \theta_2$ generate $f(\theta_1|y) = f(\theta_2|y)$. Additionally, when the latent variables of a model are unidentifiable, posterior collapse occurs, which means that the likelihood function does not depend on the latent variable(s) and is also referred to as a form of practical non-identifiability (Wang et al., 2021; Cole, 2020). Posterior collapse refers to the posterior distribution of the latent variables being equal to the prior distribution. So, if a substantial overlap between the prior and posterior exists, the prior may be dictating the posterior distribution (Wang et al., 2021; Youngflesh, 2018; Razavi et al., 2019). Even though this is not always considered problematic, it may have a negative influence on quantifying uncertainty of the model as it affects Bayesian inference (Cole, 2020).

Chapter 3

Methods

3.1 The methodological framework

This study is a theoretical and empirical comparative analysis of Bayesian uncertainty quantification techniques in the area of reinforcement learning, for which the scientific method is used as the overarching methodological framework. Even though there are differences in the practical application of the framework in various research areas, the overall process is generally similar: after making an observation or raising a question and performing exploratory analysis on the topic, one or more hypotheses are formulated and empirically tested, after which the data is analyzed and conclusions are reported (Gauch, 2003). Since it is likely that new questions are developed during the process, or that errors occur in any of the phases, the scientific method is an iterative framework rather than a sequential one. That is, in any of the phases it is possible to go back to one of the earlier stages to make adjustments. For example, during experimentation it might become clear that the hypothesis tests are wrongly formulated, which can then be adjusted by going back to (re)formulating the hypotheses, or even by doing more research on the topic and then adjusting the hypotheses. To make the findings more reproducible, the aim is to provide reproducible experiments and findings by developing explicit hypothesis tests and elaborate analysis of the results (Forde and Paganini, 2019; Pineau et al., 2021).

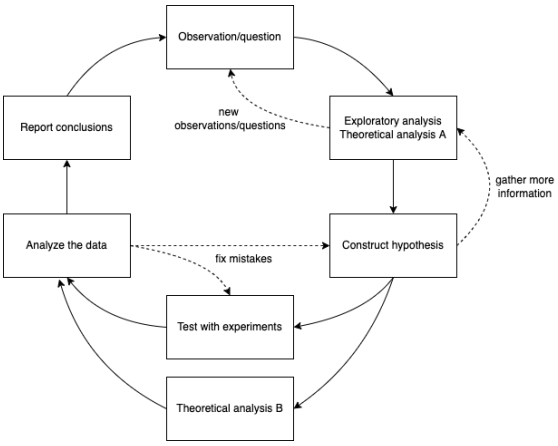

Figure 3.1: The adapted scientific method framework used in this study

Besides the empirical analysis, different Bayesian techniques have also been assessed theoretically by means of a literature review. On the one hand this theoretical comparison has formed the basis for the experiments, on the other hand, it has been part of the analysis phase after conducting the experiments, as the theoretical assessment has helped discovering potential errors and validating empirical findings. The results of both the theoretical and the empirical analyses have then been used to answer the research question. While the theoretical comparison has taken into account computational expensiveness to some extent, this has not been the case for the empirical part of this study directly. Computational expensiveness has however been indirectly part of the empirical assessment due to the constraints posed by the computational limitations for some of the models. In Figure 3.1, a sketch of the adapted scientific method framework, including the most important iterative procedures, is visualized. As multiple research questions have been defined,

the procedure illustrated in the framework has been performed for each separate question. The most important findings gathered throughout each procedure have then been combined to provide an elaborate answer to the overarching research question.

3.2 Experimental set-up

The architecture that has been used for conducting the experiments is a pre-implemented BNN with latent variables (BNN-LV) based on the architecture described in "Decomposition of Uncertainty in Bayesian Deep Learning for Efficient and Risk-sensitive Learning" (Depeweg et al., 2018; Callen et al., 2020). The key components of this architecture are: 1) two hidden layers, with 20 nodes per layer; 2) ReLU (Rectified Linear Unit) activation functions for the hidden layers; 3) a standard deviation of 1 for the noise Σ on each model output and a standard deviation γ of $1 * \textit{dimensionality}$ for the noise of each latent input. Since both benchmarks have been designed to be two-dimensional, the standard deviation for the noise of each latent input equals 2 for all experiments. In Figure 3.2, an outline of the architecture is visualized in a simplified form. To decompose the uncertainty, entropy has been approximated by using the single nearest neighbor method as described by Depeweg et al. (Depeweg et al., 2018).

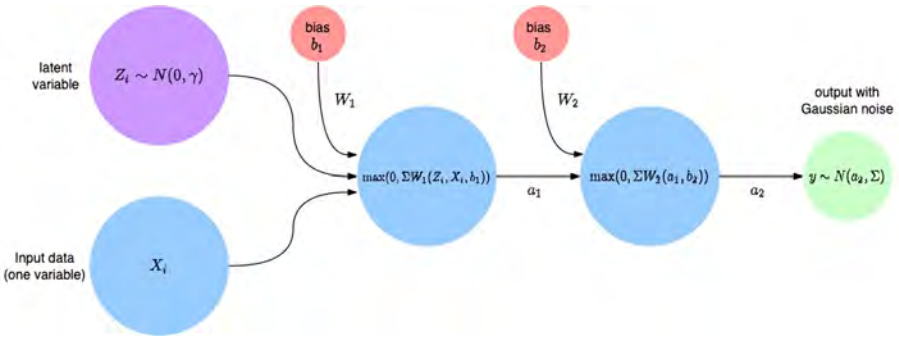

Figure 3.2: BNN-LV architecture with two hidden layers with ReLU activation functions and additive Gaussian noise

3.3 Benchmarks

To account for robustness of the methods and to assess the generalizability of the results, multiple data sets have been used in the experiments. Since the context of this study is reinforcement learning, the data used in the experiments are two reinforcement learning benchmarks: Wet-chicken and Lunar-lander. The Wet-chicken problem refers to a canoeist (the agent) trying to paddle as close to the waterfall at the end of the river as possible but without falling down the waterfall. This problem contains both heteroskedastic and bimodal noise dynamics which makes it complicated to solve for model-based reinforcement learning methods, hence there is always uncertainty involved (Depeweg et al., 2017). In this study, the river has been represented as a 3×5 grid with a (waterfall) boundary at the end, or bottom, of the grid (see Figure 3.3). The agent can take four actions: do nothing (indicated by $(0, 0)$), move left (indicated by $(-1, 0)$), move right (indicated by $(+1, 0)$), or move upward (indicated by $(0, +1)$). The applied Wet-chicken benchmark has been designed in such a way to allow easier assessment of the correctness of the

uncertainty quantification and its decomposition. For the epistemic uncertainty, by assigning the action taken to be 'none' each time, there is less data gathered by the model in the lower left part of the river. Additionally, the right-hand side of the river has been designed to be deterministic (so without randomness) and the left-hand side of the river to be completely random, which influences aleatoric uncertainty.

Somewhat similarly to the Wet-chicken benchmark, the objective of the Lunar-lander benchmark is for the agent to safely land a spacecraft in a given area on the moon without crashing. Since the original problem does not easily allow for quantifying uncertainty, this problem has been redesigned by hand in a slightly simplified fashion. Like the Wet-chicken benchmark, the environment has once again been represented as a grid and the agent can take the same four actions. Here, the grid is 5×4 with the landing area being in the middle column on the bottom row (see Figure 3.4). Since the original problem's environment contains randomness in the form of turbulence, this has also been added in the simplified design. Similar to the Wet-chicken benchmark, the left-hand side of the environment has been designed to be completely random, which gradually transforms to be completely deterministic on the right-hand side. Additionally, the agent has once again been assigned to not take any action, so that there is less data gathered on the left-hand side.

These two benchmarks have been used to assess the performance of uncertainty quantification of five sampling methods: HMC, NUTS, Metropolis, SVGD, and ADVI. These sampling methods have been selected as they are among the most frequently-used methods and classes of methods, namely variational inference and Markov Chain Monte Carlo. By implementing random seeds, all samplers use the exact same neural network and data sets as basis for their approximations to allow for better comparison between the methods and between the identifiable and non-identifiable variants of each method. As indicated before, in both environments the agent is assigned not to take any action, which refers to the policy for generating data of each episode. So, the policy for both benchmarks is $(0, 0)$. For both benchmarks, the number of episodes has been set to 100 with a maximum number of steps per episode of 20. The number of episodes refers to the number of reiterations of the game and the number of steps per episode refers to the maximum number of iterations allowed within each game before a new episode starts, provided that the agent has not fallen off the waterfall or crashed or that it has not achieved the objective.

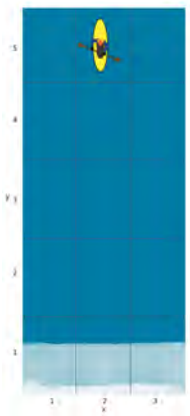

Figure 3.3: Sketch of Wet-chicken

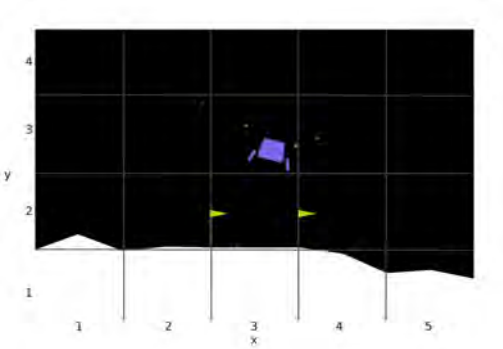

Figure 3.4: Sketch of Lunar-lander

3.4 Eliminating non-identifiability and model tuning

The focus of this research lays on non-identifiability of the posterior (from now on referred to as 'non-identifiability') and posterior collapse as an indication of non-identifiability, which can be eliminated by changing the distribution of the priors (Wang et al., 2021; Cole, 2020). In this study, this means that the prior distributions of the weights and latent variables have been adjusted from uniform to normal distributions (see Listings A.1, A.2, and A.3). To make sure differences in results are only accounted to the absence or presence of non-identifiability, all potentially influential parameters and hyperparameters have remained the same. Since the workings of the methods differ from each other even though they belong to the same classes, either MCMC or VI (see Chapter 2), the hyperparameters and their values differ across methods. However, the tuning process has had the same goal for all models - to achieve the best performance given the available time and computing power. The exact hyperparameter configurations are shown in Table 3.1[1].

Table 3.1: Configurations for each method

	Draws	Optimizer	Learning rate	Tuning samples	Target accept rate	Maximum tree depth	Tune interval
ADVI	30 000	Adam	0.01	-	-	-	-
SVGD	1 000	Adagrad	0.2	-	-	-	-
Metropolis	20 000	-	-	20 000	0.99	15	1 000
NUTS	400	-	-	500	0.9 (0.5 for non-id.)	15	-
HMC	2 500	-	-	4 000	0.99	15	-

[1] the relevant code written for this research can be found here: `https://github.com/Lieve2/nonidentifiability-and-UQ.git` (Göbbels, 2022).

Chapter 4

Experiments

As indicated in the previous chapter, in order to be able to analyze the influence of non-identifiability on the performance of the uncertainty quantification methods, these methods have been compared in two environments of which one contains non-identifiability and the other does not, and that are otherwise identical so that the difference in performance can solely be attributed to the aspect of non-identifiability. Firstly, the existence of non-identifiability has been empirically proven with the aim of verifying what was already proven theoretically (see Section 2.3). After this, the performance of each uncertainty quantification method has been assessed and compared. This is followed by an experiment to test whether non-identifiability has been eliminated successfully, after which the performance of each uncertainty quantification method has once again been assessed and compared.

Algorithm 1: Prior-Posterior Overlap (PPO)

Input: p, q, n, with arrays p and q and n the number of bins
Output: overlap coefficient r

$c_1 = 0$
$c_2 = 0$
$a = \min((\min(p), \min(q)))$ // lower bound of integration range
$b = \min((\max(p), \max(q)))$ // upper bound of integration range
$w = \frac{b-a}{n}$ // bin width
$l = a$ // lower bound for bin
for $i = 0$ *to* $n - 1$ **do**
  $h = l + w$ // upper bound for bin
  for $j = 0$ *to* $length(p) - 1$ **do**
    // count frequency of values in p within bin
    if $p[j] > a \wedge p[j] \leq h$ **then**
      $c_1 = c_1 + 1$
    $p_r = \frac{c_1}{length(p)-1}$
  for $k = 0$ *to* $length(q) - 1$ **do**
    // count frequency of values in q within bin
    if $q[k] > a \wedge q[k] \leq h$ **then**
      $c_2 = c_2 + 1$
    $q_r = \frac{c_2}{length(q)-1}$
  $r[i] = \min((p_r, q_r))$ // store lowest frequency on i
  $l = h$
$r = \sum_{i=1}^{n-1} r[i]$ // sum frequencies on entire interval
return r

4.1 Experiment 1: empirical proof of non-identifiability

This experiment centers around the first subquestion: *Is there posterior non-identifiability for each of the sampling methods?* Whether there is posterior non-identifiability (or posterior collapse) has been assessed using a (density) plot and a calculation of the prior-posterior overlap (PPO). If the overlap is significant, wherefore the standard threshold of (>)35% has been used, it is considered that there is posterior collapse and posterior non-identifiability (Cole, 2020; Youngflesh, 2018). Accordingly, the hypothesis central to this experiment is that the PPO is larger than 0.35 for all models.

Since it is impossible to get an exact value of the PPO by just using plots, an algorithm has been constructed to calculate the exact overlap, which is shown in the pseudocode in Algorithm 1 and in Listing B.1. Depending on how many bins are used, the value might diverge slightly from the true value. Considering that the posterior (predictive) distribution ranges approximately from -4 to 8 (see for example Figures D.1 and D.2), and after deliberate testing of different numbers of bins, the number of bins has been set to 100 in order to result in a PPO value that is close enough to the true value, while keeping the running time and memory use as low as possible.

4.2 Experiment 2: performance in the presence of non-identifiability

For answering the second sub-question, *How do the sampling methods perform in the presence of non-identifiability?*, the performance of each method has been assessed by plotting the (decomposed) uncertainty quantification. Additionally, the negative log-likelihood (NLL) has been computed to quantitatively analyze the performance. Therefore, data from the likelihood distribution $y \sim N(\mu, \sigma|Y)$ has been used. The implementation of this calculation in Python can be found in Listing C.1. Together with the results gathered from Experiment 4, this has allowed for comparison of performance on two levels: between the different methods and between the identifiable and non-identifiable variants of each method. In this experiment, there is no explicit hypothesis, but there is an overarching hypothesis that concerns both this experiment and Experiment 4: there is a difference in performance depending on the presence or absence of non-identifiability.

4.3 Experiment 3: eliminating non-identifiability

The central question answered with the third experiment is: *Is non-identifiability of the posterior successfully eliminated?* This experiment is mainly intended to verify the successful elimination of the non-identifiability in order to be able to conduct the following experiment (Experiment 4) correctly and to be able to compare the results from Experiment 4 with those acquired in Experiment 2 in a sound way. Similar to the first experiment, a plot and calculation of the prior-posterior overlap (PPO) have been used to assess the non-existence of posterior non-identifiability. However, as opposed to Experiment 2, the goal in this experiment has been to prove that for all models, the PPO is equal to or lower than 35%. So, the hypothesis central to this experiment is that the PPO is equal to or lower than 35% and thus there is no non-identifiability.

4.4 Experiment 4: performance in the absence of non-identifiability

In the fourth and final experiment, the goal has been to gather results on the performance of the methods in the absence of non-identifiability that allow for justly assessing the difference in performance depending on the presence of non-identifiability. Hence, this experiment has been used to answer the fourth sub-question: *How do the methods perform in the absence of non-identifiability?* The first part of the experiment is similar to Experiment 2: the performance of each method has been assessed by plotting the (decomposed) uncertainty quantification and the minimum negative log-likelihood has been computed to quantitatively analyze model performance and to allow for comparison of the performances. After gathering the results, they have been qualitatively (plots of the uncertainty quantification) and quantitatively (negative log-likelihoods) compared with the results from Experiment 2.

Chapter 5

Results

In this chapter, the main findings of the experiments are described with the aim to gather all necessary information to answer the experiment-specific research questions. For each of these subquestions, a brief answer and a description of the validity of the hypothesis is provided.

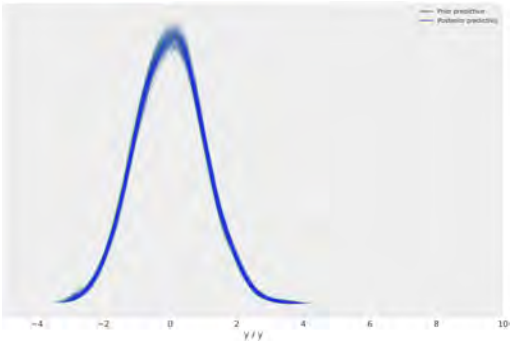

Figure 5.1: PPO of non-identifiable NUTS on the Wet-chicken benchmark (99.5% overlap)

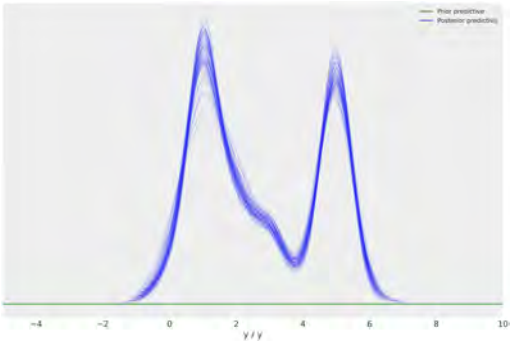

Figure 5.2: PPO of identifiable NUTS on the Wet-chicken benchmark (3.3% overlap)

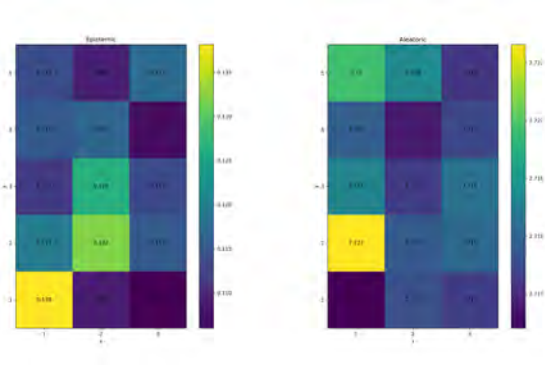

Figure 5.3: Uncertainty quantification plot of non-identifiable NUTS on the Wet-chicken benchmark

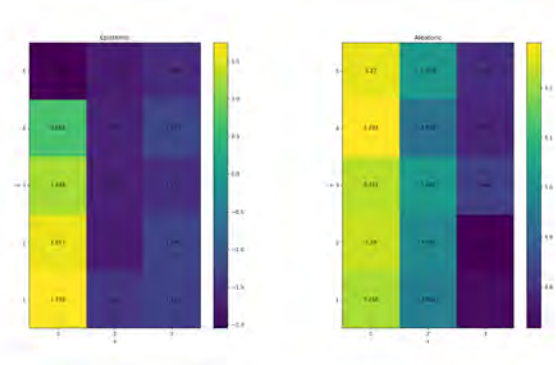

Figure 5.4: Uncertainty quantification plot of identifiable NUTS on the Wet-chicken benchmark

5.1 Empirical findings

In the first experiment, it is shown that all five models of the samplers with uniform priors with narrow bounds are non-identifiable due to a prior-posterior overlap coefficient that is larger than 0.35 (see Tables 5.1 and 5.2, middle column of top part). That there is severe overlap between the

Table 5.1: PPO and NLL of the sampling methods on the Wet-chicken benchmark

Method	PPO	NLL
Non-identifiable		
Metropolis	0.840	127 978.7
HMC	0.834	127 971.5
NUTS	0.995	158 012.2
ADVI	0.995	157 013.3
SVGD	0.994	157 025.8
Identifiable		
Metropolis	0.059	6240.7
HMC	0.036	1605.1
NUTS	0.033	**507.8**
ADVI	0.035	7624.5
SVGD	0.038	2860.1

Note: 'Non-identifiable' refers to Experiment 1 and 'Identifiable' to Experiment 3.

Table 5.2: PPO and NLL of the sampling methods on the Lunar-lander benchmark

Method	PPO	NLL
Non-identifiable		
Metropolis	0.866	191 265.3
HMC	0.853	197 131.3
NUTS	0.995	239 571.4
ADVI	0.995	226 546.7
SVGD	0.995	239 513.6
Identifiable		
Metropolis	0.038	11 882.9
HMC	0.067	3122.0
NUTS	0.041	**1874.0**
ADVI	0.060	17 162.8
SVGD	0.046	7395.1

Note: 'Non-identifiable' refers to Experiment 1 and 'Identifiable' to Experiment 3.

prior and posterior density distributions is also visible in Figures 5.1, D.1, D.3, D.7, D.9, D.11, D.13, D.15, D.17, and D.19. Additionally, the results from the third experiment show that each model of a sampler with normally distributed priors has a PPO coefficient that is significantly lower than 0.35 (see Tables 5.1 and 5.2, middle column of bottom part).

Considering the results of the first experiment, the corresponding hypothesis can be accepted as the PPO for each method is larger than 0.35, indicating that there is indeed posterior non-identifiability for each of the sampling methods. This finding then answers the first subquestion: *Is there posterior non-identifiability for each of the sampling methods?* Regarding the third experiment, the results are exactly opposite as the PPO is lower than 0.35 for each of the sampling methods. Therewith, the hypothesis that for all sampling methods, the PPO is equal to or lower than 0.35 can also be accepted. Hence, the answer to the respective subquestion is that non-identifiability is indeed successfully eliminated.

In the second experiment, the results from the performance of each uncertainty quantification method have been gathered. These results show that all models have a negative log-likelihood that falls in the range of $1.2 \cdot 10^5$ to $2.4 \cdot 10^5$ for both data sets (see Tables 5.1 and 5.2, right column of top part). The decomposed uncertainty quantification as a qualitative measure of performance is visualized in the plot-pairs in Figures 5.3, E.1, E.3, E.7, E.9, E.11, E.13, E.15, E.17, and E.19. In the fourth experiment, the results of the performance of each uncertainty quantification method have been gathered once again, but this time from the identifiable variant. These results are shown in Tables 5.1 and 5.2 (right column of the bottom part), and in Figures 5.4, E.2, E.4, E.8, E.10, E.12, E.14, E.16, E.18, and E.20. Here, the negative log-likelihood values are substantially lower than those from the non-identifiable variants. For both benchmarks, the ranking of the methods regarding the negative log-likelihood is the same, with NUTS having the best results, HMC being the second-best and SVGD being the best variational inference method. In light of these results, the hypothesis that there is a difference in performance in the absence of non-identifiability can be accepted.

Chapter 6

Discussion

In this chapter, interpretations of the main findings of both the theoretical analysis and the experiments are described with the aim to gather all necessary information to answer the experiment-specific research questions. This provides the basis for developing an answer to the main research question.

6.1 Theoretical implications of the used methods

In theory, the NUTS sampler is considered superior over Metropolis and HMC as it is able to automatically tune the settings preventing U-turns or random walk (Hoffman and Gelman, 2014). Especially Metropolis is seen as an outdated method, because it is sensitive to the choice of proposal distribution and hence requires trial and error when it comes to tuning the model to perform decently. The main problem of Metropolis is that it is too random, causing the method to re-explore the same parts of the target and, unless tuned well, rejects a lot of proposals (McElreath, 2017; Monnahan et al., 2017). So at the very least, it is considered computationally inefficient. Instead of making random proposals, both HMC and NUTS approach the sampling and approximation as a physics simulation where the vector of parameters is a particle in a space with n dimensions and a frictionless surface (e.g. a parabola). This surface represents the target distribution. In each iteration, the particle is released in a random direction in the space and eventually returns in such a way that information is gathered on the shape of the surface (McElreath, 2017). A major drawback of HMC is that it makes U-turns if the number of steps is not configured right, causing the method to be relatively inefficient as is re-explores local spaces. NUTS tries to avoid this by adaptively finding a suitable number of steps (Hoffman and Gelman, 2014; McElreath, 2017). However, the problem of re-exploring local spaces remains for some of the more complex targets, like multimodal targets or Neal's funnel (McElreath, 2017; Wijono, 2020). Another drawback of NUTS is that it computes the gradient at each step, which makes it a computationally expensive and not scalable sampler (Wijono, 2020; Liu and Wang, 2016).

Since VI methods like SVGD and ADVI reframe the objective into an optimization problem, they can be used on big data sets, which is impossible when using NUTS due to its high computational costs. The main difference between (mean-field) ADVI and SVGD is that ADVI uses mean-field approximation with the objective to maximize ELBO while SVGD does not (PyMC3, 2021a,b). However, ADVI can be easily redesigned to a full-rank version (Kucukelbir et al., 2016). This version has nonetheless not been implemented in this study due to time constraints. A drawback of ADVI is that it is generally not able to deal with multimodality and is more prone to random walk compared to SVGD and NUTS. The main drawback of SVGD is that it is computationally more expensive than ADVI (PyMC3, 2020). So, overall VI methods are more

computationally efficient than MCMC methods but this comes at the cost of performance. While MCMC methods are usually preferred due to better performance, VI methods are considered superior when it comes to large data sets, since MCMC methods are computationally expensive and therewith incompatible with big data.

6.2 Analysis of the empirical findings

Regarding the experiments concerning assessment of the prior-posterior overlap as a measure of non-identifiability, when comparing the prior-posterior distributions from the non-identifiable variants of the methods with those from the identifiable variants, there is a clear difference in the shape of the posterior. This indicates that the prior dictates the posterior as described in Section 2.3. Additionally, similar to what is described in the theoretical analysis, it appears that both Metropolis and ADVI do not seem to be able to capture multimodality compared to the other methods as they have fewer peaks than HMC, NUTS and SVGD (see Figures D.12, D.14, D.16, D.18, and D.20).

To determine what the lack of capturing multimodality means for the ability to quantify the uncertainty and to compare the performance of the methods with and without non-identifiability, both the qualitative results and the quantitative results can be used. For the qualitative results, it means that the uncertainty quantification plots need to be interpreted in comparison with the ideal plot. As described in Chapter 3, regarding epistemic uncertainty there ideally is high uncertainty on the left-hand side of the grid from $y = 1$ to $y = 4$ for the Wet-chicken and $y = 1$ to $y = 3$ for the Lunar-lander benchmark respectively, and low uncertainty for all other positions. For aleatoric uncertainty, the optimal plot has high uncertainty on the left-hand side which gradually decreases for each step to the right.

The results of the second experiment show that none of the methods containing non-identifiability are able to come close to this ideal. This poor performance is also visible in the negative log-likelihood, which is the quantitative measure used: with the lowest negative log-likelihoods being $1.3 \cdot 10^5$ and $1.9 \cdot 10^5$ on the Wet-chicken and Lunar-lander benchmark respectively, the non-identifiable variants of the five methods can all be considered to have poor performance. In the last experiment, on the other hand, the identifiable variants of NUTS, HMC, and SVGD seem to be able to capture the uncertainty to a certain degree in some cases. More specifically, NUTS is able to represent the ideal plot-pair for the Wet-chicken benchmark and has a decent approximation of the aleatoric uncertainty for the Lunar-lander benchmark. HMC, on the other hand, comes close to the true epistemic uncertainty regarding the Lunar-lander benchmark and has good estimations of both epistemic and aleatoric uncertainty regarding the Wet-chicken benchmark. SVGD, being the best-performing variational inference method, provides good approximation of the aleatoric uncertainty but misestimates both types of uncertainty in the Lunar-lander environment.

Metropolis and ADVI have the worst qualitative and quantitative performance results for both benchmarks in Experiment 4. This may be due to their inability to capture multimodality, which seems especially important in the Lunar-lander environment. Considering this, in combination with the given that both methods are considered problematic in theory, it can be concluded that these methods are not suitable when it comes to quantifying uncertainty in the context of reinforcement learning. In addition, besides the performance of identifiable NUTS in the Wet-

chicken benchmark, none of the other performance results are desirable when it comes to reliable uncertainty quantification. This may be due to the limited number of draws used for training the methods or it may be due to the implementation of the single nearest-neighbor algorithm to approximate and decompose the uncertainty, as this is a relatively frail method. However, increasing the sample size or changing the decomposition with a more complex, yet more robust algorithm would demand even more computing power, making it less utilizable.

Chapter 7

Conclusion

This study has assessed five uncertainty quantification methods, HMC, NUTS, Metropolis, ADVI, and SVGD, to investigate the effects of Bayesian non-identifiability on the performance of these methods in the context of reinforcement learning. Additionally, these methods have also been compared with each other to identify the strengths and weaknesses of these methods. Results show that Bayesian non-identifiability has a clear negative influence on the ability to accurately quantify uncertainty for each of the methods. However, even when Bayesian non-identifiability is eliminated, most of the methods are still not able to provide accurate enough approximations to the true uncertainty. As this negatively affects the optimization process of a reinforcement learning agent and therewith decreases reliability of the AI system, it is vital to investigate what influences the accuracy of such uncertainty quantification methods besides Bayesian non-identifiability. Regarding this, further research could for instance focus on investigating the influence of other types of non-identifiability, like likelihood non-identifiability, to further identify what improvements are needed to acquire well-performing methods. Despite the probability of other types of non-identifiability negatively influencing the performance of the five investigated methods, part of the misestimation could be a result of the limited number of samples used in the conducted experiments. Additionally, the decomposition of uncertainty could also be miscalculated at times due to using the single nearest-neighbor method instead of a more robust algorithm.

This research shows that Bayesian non-identifiability negatively influences the ability to accurately quantify uncertainty for both variational inference class models and Markov Chain Monte Carlo methods in the context of reinforcement learning, which may cause problems when trying to efficiently optimize reinforcement learning agents and provide reliable AI systems. As the topic of non-identifiability has remained underexposed in the context of reinforcement learning, this study highlights the importance of eliminating (Bayesian) non-identifiability and serves as a starting point to further investigate the effects of and solutions to other types of non-identifiability to ultimately develop efficient and accurate Bayesian uncertainty quantification methods.

Bibliography

M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021. ISSN 15662535. doi: 10.1016/j.inffus.2021.05.008. 1, 3, 4, 5

E. S. Allman, C. Matias, and J. A. Rhodes. Identifiability of parameters in latent structure models with many observed variables. *Annals of Statistics*, 37(6 A):3099–3132, 9 2008. doi: 10.1214/09-AOS689. URL http://arxiv.org/abs/0809.5032http://dx.doi.org/10.1214/09-AOS689. 6

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight Uncertainty in Neural Networks. *32nd International Conference on Machine Learning, ICML 2015*, 2:1613–1622, 5 2015. URL https://arxiv.org/abs/1505.05424v2. 5

J. M. Booker and T. J. Ross. An evolution of uncertainty assessment and quantification. *Scientia Iranica*, 18(3):669–676, 6 2011. ISSN 1026-3098. doi: 10.1016/J.SCIENT.2011.04.017. 4

O. Callen, G. Pestre, H. Sansum, and N. Vanderklaauw. GitHub - decomposition-of-uncertainty, 2020. URL https://github.com/2020fa-207-final-project/decomposition-of-uncertainty. 8

D. J. Cole. *Parameter Redundancy and Identifiability*. Chapman and Hall/CRC, Boca Raton, 1 edition, 5 2020. ISBN 9781315120003. doi: 10.1201/9781315120003. URL https://www-taylorfrancis-com.dianus.libr.tue.nl/books/mono/10.1201/9781315120003/parameter-redundancy-identifiability-diana-cole. 6, 10, 12

S. Depeweg, J. Miguel Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Learning and Policy Search in Stochastic Dynamical Systems with Bayesian Neural Networks, 2017. 8

S. Depeweg, J. Miguel Hernández-Lobato, F. Doshi-Velez, and S. Udluft. Decomposition of Uncertainty in Bayesian Deep Learning for Efficient and Risk-sensitive Learning. 2018. 4, 8

S. Depeweg, T. A. Runkler, A. Laura Leal-Taixé, and J. Miguel Hernández-Lobato. *Modeling Epistemic and Aleatoric Uncertainty with Bayesian Neural Networks and Latent Variables*. PhD thesis, Technische Universität München, 2019. 3, 4, 5

J. Z. Forde and M. Paganini. The Scientific Method in the Science of Machine Learning. Technical report, 4 2019. URL https://arxiv.org/abs/1904.10922v1. 7

Y. Gal and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. Technical report, 2016. URL https://arxiv.org/abs/1506.02142. 5

H. Gauch. *Scientific method in practice*. Cambridge University Press, 2003. ISBN 9780521017084. 7

P. Ghosh, M. S. M. Sajjadi, A. Vergari, M. Black, and B. Schölkopf. From Variational to Deterministic Autoencoders. Technical report, 3 2019. URL https://arxiv.org/abs/1903.12436v4. 5

L. Göbbels. GitHub Repository Nonidentifiability and UQ, 2022. URL https://github.com/Lieve2/nonidentifiability-and-UQ.git. 10

B. Goodman and S. Flaxman. European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Magazine*, 38(3):50–57, 10 2017. ISSN 2371-9621. doi: 10.1609/AIMAG.V38I3.2741. URL https://ojs.aaai.org/index.php/aimagazine/article/view/2741. 1

G. Gundersen. Why Metropolis–Hastings Works, 11 2019. URL https://gregorygundersen.com/blog/2019/11/02/metropolis-hastings/. 5

M. D. Hoffman and A. Gelman. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15:1593–1623, 2014. URL http://mcmc-jags.sourceforge.net. 5, 16

H. M. Hossain, M. A. A. H. Khan, and N. Roy. Active learning enabled activity recognition. *Pervasive and Mobile Computing*, 38:312–330, 7 2017. ISSN 1574-1192. doi: 10.1016/J.PMCJ.2016.08.017. 5

E. Hüllermeier and W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 3 2021. ISSN 15730565. doi: 10.1007/S10994-021-05946-3/FIGURES/17. URL https://link.springer.com/article/10.1007/s10994-021-05946-3. 4

A. Kendall and Y. Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? *Advances in Neural Information Processing Systems*, 2017-December:5575–5585, 3 2017. ISSN 10495258. doi: 10.48550/arxiv.1703.04977. URL https://arxiv.org/abs/1703.04977v2. 4

D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 5 2014. doi: 10.48550/arxiv.1312.6114. URL https://arxiv.org/abs/1312.6114v10. 5

A. D. Kiureghian and O. Ditlevsen. Aleatory or epistemic? Does it matter? *Structural Safety*, 31(2):105–112, 3 2009. ISSN 0167-4730. doi: 10.1016/J.STRUSAFE.2008.06.020. 4

M. Kläs and A. M. Vollmer. Uncertainty in Machine Learning Applications: A Practice-Driven Classification of Uncertainty. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11094 LNCS:431–438, 2018. ISSN 16113349. doi: 10.1007/978-3-319-99229-7{_}36. URL https://link.springer.com/chapter/10.1007/978-3-319-99229-7_36. 4

T. C. Koopmans. Identification Problems in Economic Model Construction. *Econometrica*, 17(2):125, 4 1949. ISSN 00129682. doi: 10.2307/1905689. 1, 2, 6

A. Kucukelbir, D. M. Blei, A. Gelman, R. Ranganath, and D. Tran. Automatic Differentiation Variational Inference. *Journal of Machine Learning Research*, 18:1–45, 3 2016. ISSN 15337928. doi: 10.48550/arxiv.1603.00788. URL `https://arxiv.org/abs/1603.00788v1`. 5, 16

A. Kucukelbir, D. Tran, A. Gelman, and D. M. Blei. Automatic Differentiation Variational Inference Rajesh Ranganath. *Journal of Machine Learning Research*, 18:1–45, 2017. URL `http://jmlr.org/papers/v18/16-107.html`. 5

M. Kull and P. A. Flach. Reliability Maps: A Tool to Enhance Probability Estimates and Improve Classification Accuracy. *Lecture Notes in Artificial Intelligence*, 8725:18–33, 2014. doi: 10.1007/978-3-662-44851-9{_}2. URL `http://www.bristol.ac.uk/red/research-policy/pure/user-guides/ebr-terms/`. 4

M. A. Kupinski, J. W. Hoppin, E. Clarkson, and H. H. Barrett. Ideal-observer computation in medical imaging with use of Markov-chain Monte Carlo techniques. *Journal of the Optical Society of America. A, Optics, image science, and vision*, 20(3):430, 3 2003. ISSN 1084-7529. doi: 10.1364/JOSAA.20.000430. URL `https://pubmed.ncbi.nlm.nih.gov/12630829/`. 5

R. Kurle, T. Januschowski, J. Gasthaus, and Y. Wang. On Symmetries in Variational Bayesian Neural Nets. *Bayesian Deep Learning workshop, NeurIPS*, 2021. URL `http://bayesiandeeplearning.org/2021/papers/30.pdf`. 6

Q. Liu and D. Wang. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. *Advances in Neural Information Processing Systems*, pages 2378–2386, 8 2016. ISSN 10495258. doi: 10.48550/arxiv.1608.04471. URL `https://arxiv.org/abs/1608.04471v3`. 5, 16

Y. Liu, P. Ramachandran, Q. Liu, and J. Peng. Stein Variational Policy Gradient. *Uncertainty in Artificial Intelligence - Proceedings of the 33rd Conference, UAI 2017*, 4 2017. doi: 10.48550/arxiv.1704.02399. URL `https://arxiv.org/abs/1704.02399v1`. 5

R. McElreath. Markov Chains: Why Walk When You Can Flow? , 11 2017. URL `https://elevanth.org/blog/2017/11/28/build-a-better-markov-chain/`. 16

C. C. Monnahan, J. T. Thorson, and T. A. Branch. Faster estimation of Bayesian models in ecology using Hamiltonian Monte Carlo. *Methods in Ecology and Evolution*, 8(3):339–348, 3 2017. ISSN 2041210X. doi: 10.1111/2041-210X.12681. 5, 16

T. Papamarkou, J. Hinkle, M. T. Young, and D. Womble. Challenges in Markov chain Monte Carlo for Bayesian neural networks. 10 2021. doi: 10.48550/arxiv.1910.06539. URL `https://arxiv.org/abs/1910.06539v6`. 1, 6

J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivì, A. Beygelzimer, F. D'alché-Buc, T. Paris, H. Larochelle, E. Florence D'alché-Buc, and H. L. Fox. Improving Reproducibility in Machine Learning Research (A Report from the NeurIPS 2019 Reproducibility Program). *Journal of Machine Learning Research*, 22:1–20, 2021. 7

A. A. Pourzanjani, R. M. Jiang, and L. R. Petzold. Improving the Identifiability of Neural Networks for Bayesian Inference, 2017. URL `http://bayesiandeeplearning.org/2017/papers/15.pdf`. 6

M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., Hoboken, 2 edition, 2005. ISBN 9780470316887. doi: 10.1002/9780470316887. 3

PyMC3. Variational API quickstart, 2020. URL `https://docs.pymc.io/en/v3/pymc-examples/examples/variational_inference/variational_api_quickstart.html`. 16

PyMC3. pymc.ADVI, 2021a. URL `https://docs.pymc.io/en/latest/api/generated/pymc.ADVI.html`. 16

PyMC3. pymc.SVGD , 2021b. URL `https://docs.pymc.io/en/latest/api/generated/pymc.SVGD.html`. 16

A. Razavi, A. v. d. Oord, B. Poole, and O. Vinyals. Preventing Posterior Collapse with delta-VAEs. *7th International Conference on Learning Representations, ICLR 2019*, 1 2019. doi: 10.48550/arxiv.1901.03416. URL `https://arxiv.org/abs/1901.03416v1`. 6

G. Roeder, Y. Wu, and D. Duvenaud. Sticking the Landing: A Simple Reduced-Variance Gradient for ADVI, 2016. 5

J. W. Romeijn and J. Williamson. Intervention and Identifiability in Latent Variable Modelling. *Minds and Machines*, 28(2):243–264, 6 2018. ISSN 15728641. doi: 10.1007/S11023-018-9460-Y/FIGURES/3. URL `https://link.springer.com/article/10.1007/s11023-018-9460-y`. 6

T. J. Rothenberg. Identification in Parametric Models. *Econometrica*, 39(3):577, 5 1971. ISSN 00129682. doi: 10.2307/1913267. 1, 2, 6

P. J. Schmidt, M. B. Emelko, and M. E. Thompson. Recognizing Structural Nonidentifiability: When Experiments Do Not Provide Information About Important Parameters and Misleading Models Can Still Have Great Fit. *Risk Analysis*, 40(2):352–369, 2 2020. ISSN 1539-6924. doi: 10.1111/RISA.13386. URL `https://onlinelibrary-wiley-com.dianus.libr.tue.nl/doi/full/10.1111/risa.13386`. 6

R. Senge, S. Bösner, K. Dembczyński, J. Haasenritter, O. Hirsch, N. Donner-Banzhoff, and E. Hüllermeier. Reliable classification: Learning classifiers that distinguish aleatoric and epistemic uncertainty. *Information Sciences*, 255:16–29, 1 2014. ISSN 00200255. doi: 10.1016/J.INS.2013.07.030. 4

J. Shi, S. Sun, and J. Zhu. Kernel Implicit Variational Inference. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 5 2017. doi: 10.48550/arxiv.1705.10119. URL `https://arxiv.org/abs/1705.10119v3`. 6

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Van Den Driessche, T. Graepel,

and H. Demis. Mastering the Game of Go without Human Knowledge. *Nature*, 550(7676), 2017. doi: 10.1038/nature24270. URL `https://discovery.ucl.ac.uk/id/eprint/10045895/1/agz_unformatted_nature.pdf`. 4

R. S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, 7 1991. ISSN 0163-5719. doi: 10.1145/122344.122377. URL `https://dl.acm.org/doi/abs/10.1145/122344.122377`. 4

P. Swazinna, S. Udluft, D. Hein, and T. Runkler. Comparing Model-free and Model-based Algorithms for Offline Reinforcement Learning, 2022. 3

J. Swiatkowski, K. Roth, S. V. Bastiaan, L. Tran, J. V. Dillon, J. Snoek, S. Mandt, T. Salimans, R. Jenatton, and S. Nowozin. The k-tied Normal Distribution: A Compact Parameterization of Gaussian Mean Field Posteriors in Bayesian Neural Networks. In *Proceedings of the 37th International Conference on Machine Learning*, Vienna, 2020. 5

M. Turek. Explainable Artificial Intelligence (XAI). URL `https://www.darpa.mil/program/explainable-artificial-intelligence`. 1

K. R. Varshney and H. Alemzadeh. On the Safety of Machine Learning: Cyber-Physical Systems, Decision Sciences, and Data Products. *Big Data*, 5(3):246–255, 10 2016. ISSN 2167647X. doi: 10.48550/arxiv.1610.01256. URL `https://arxiv.org/abs/1610.01256v2`. 4

Y. Wang, D. M. Blei, and J. P. Cunningham. Posterior Collapse and Latent Variable Non-identifiability. In *35th Conference on Neural Information Processing Systems*, 2021. URL `https://proceedings.neurips.cc/paper/2021/file/2b6921f2c64dee16ba21ebf17f3c2c92-Paper.pdf`. 1, 6, 10

S. Wechsler, R. Izbicki, and L. G. Esteves. A Bayesian Look at Nonidentifiability: A Simple Example. *The American Statistician*, 67(2):90–93, 5 2013. ISSN 00031305. doi: 10.1080/00031305.2013.778787. URL `https://www-tandfonline-com.dianus.libr.tue.nl/doi/abs/10.1080/00031305.2013.778787`. 6

L. Wells and T. Bednarz. Explainable AI and Reinforcement Learning—A Systematic Review of Current Approaches and Trends. *Frontiers in Artificial Intelligence*, 4:48, 5 2021. ISSN 26248212. doi: 10.3389/FRAI.2021.550030/BIBTEX. 1

W. Wijono. Bayesian Inference Algorithms: MCMC and VI, 3 2020. URL `https://towardsdatascience.com/bayesian-inference-algorithms-mcmc-and-vi-a8dad51ad5f5`. 16

Y. Yacoby, W. Pan, F. Doshi-Velez, and J. A. Paulson. Mitigating the Effects of Non-Identifiability on Inference for Bayesian Neural Networks with Latent Variables. 11 2021. doi: 10.48550/arxiv.1911.00569. URL `https://arxiv.org/abs/1911.00569v3`. 1, 6

G. O. Yalcin. 5 Significant Reasons Why Explainable AI is an Existential Need for Humanity , 12 2020. URL `https://towardsdatascience.com/5-significant-reasons-why-explainable-ai-is-an-existential-need-for-humanity-abe57ced4541`. 1

F. Yi, W. Fu, and H. Liang. Model-based Reinforcement Learning: A Survey. *Proceedings of the International Conference on Electronic Business (ICEB)*, 2018-Decem:421–429, 6 2020. ISSN 16830040. URL https://arxiv.org/abs/2006.16712v3. 3, 4

C. Youngflesh. Check your prior posterior overlap (PPO) – MCMC wrangling in R made easy with 'MCMCvis', 1 2018. URL https://www.r-craft.org/r-news/check-your-prior-posterior-overlap-ppo-mcmc-wrangling-in-r-made-easy-with-mcmcvis/. 6, 12

M. Zhang, B. Selic, S. Ali, T. Yue, O. Okariz, and R. Norgren. Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9764: 247–264, 2016. ISSN 16113349. doi: 10.1007/978-3-319-42061-5{_}16. URL https://link.springer.com/chapter/10.1007/978-3-319-42061-5_16. 4

Appendix A

Python code snippets of the priors

```
p_mu = 0
p_sigma = 5
l_sigma = 0.25
lv_gamma = 2

# Prior on w (same shape as MLE)
w_prior = pm.Normal(name='w', mu=p_mu, sigma=p_sigma, shape=bnn.weights.shape)

# Latent variable prior (same shape as number of datapoints)
lv_prior = pm.Normal(name='z', mu=0, sigma=lv_gamma, shape=(X_train.shape[0],1))
```

Listing A.1: Priors of identifiable variants

```
p_mu = 0
p_sigma = 5
l_sigma = 0.25
lv_gamma = 2

# Prior on w (same shape as MLE)
w_prior = pm.Uniform(name='w', lower=-0.01, upper=0.01, shape=bnn_lv_chicken.
    weights.shape)

# Latent variable prior (same shape as number of datapoints)
lv_prior = pm.Uniform(name='z', lower=-0.01, upper=0.01, shape=(X_train.shape[0],1)
    )
```

Listing A.2: Priors of non-identifiable NUTS, ADVI, SVGD

```
p_mu = 0
p_sigma = 5
l_sigma = 0.25
lv_gamma = 2

# Prior on w (same shape as MLE)
w_prior = pm.Uniform(name='w', lower=-0.05, upper=0.05, shape=bnn_lv_chicken.
    weights.shape)

# Latent variable prior (same shape as number of datapoints)
lv_prior = pm.Uniform(name='z', lower=-0.05, upper=0.05, shape=(X_train.shape[0],1)
    )
```

Listing A.3: Priors of non-identifiable Metropolis, HMC

Appendix B

Python function for calculating the PPO

```python
import numpy as np
from tqdm import tqdm

# function for PPO calculation
def PPO(arr1, arr2, nbins):

    # Determine the integration range
    min_value = np.min((arr1.min(), arr2.min()))
    max_value = np.min((arr1.max(), arr2.max()))

    # Determine the bin width
    bin_width = (max_value - min_value) / nbins

    # For each bin, find minimum frequency
    lower_bound = min_value  # Lower bound of the first bin is the lowest value of
        both arrays
    lower = np.empty(nbins)  # Array for storing lowest frequency for each bin

    for b in tqdm(range(nbins)):
        higher_bound = lower_bound + bin_width  # Set the higher bound for the bin

        # Determine the share of samples in the interval
        freq1 = np.ma.masked_where((arr1 < lower_bound) | (arr1 >= higher_bound),
            arr1).count() / len(arr1)
        freq2 = np.ma.masked_where((arr2 < lower_bound) | (arr2 >= higher_bound),
            arr2).count() / len(arr2)

        # Conserve the lower frequency and proceed
        lower[b] = np.min((freq1, freq2))
        lower_bound = higher_bound
        pass

    return lower.sum()
```

Listing B.1: PPO calculation

Appendix C

Python code snippet for calculating the NLL

```
# likelihood (part of sampler model initialization)
y = pm.Normal(name='y', mu = bnn.forward(X = X_train, input_noise = lv_prior,
    weights=w_prior), sigma = l_sigma, observed = Y_train)

# extracting the log likelihood and transforming it to NLL
obs_logp = pm_model.y.logp
loglikelihood = np.array([obs_logp(p) for p in trace.points()])
nll = -1 * loglikelihood

# extracting the best value
min_nll = min(nll)
round(min_nll, 3)
```

Listing C.1: NLL calculation

Appendix D

Plots of the prior-posterior overlap

D.1 Wet-chicken benchmark

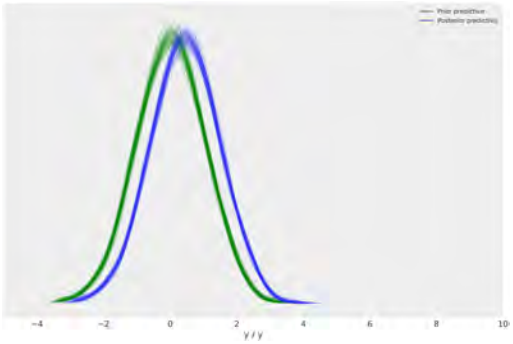

Figure D.1: PPO of non-identifiable HMC on the Wet-chicken benchmark

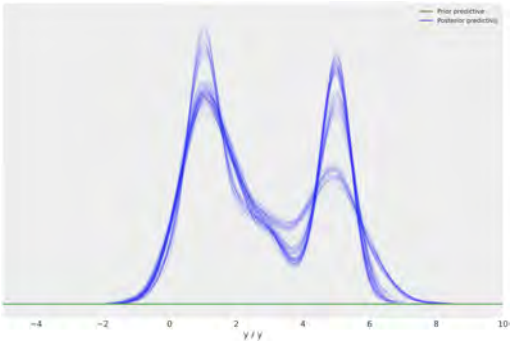

Figure D.2: PPO of identifiable HMC on the Wet-chicken benchmark

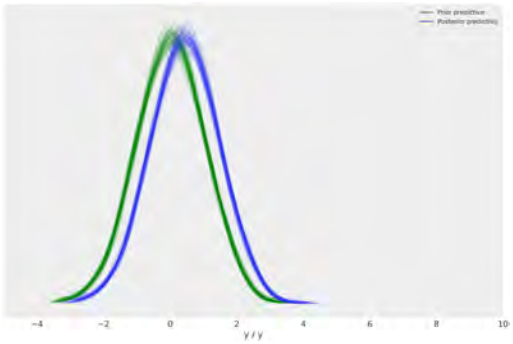

Figure D.3: PPO of non-identifiable Metropolis on the Wet-chicken benchmark

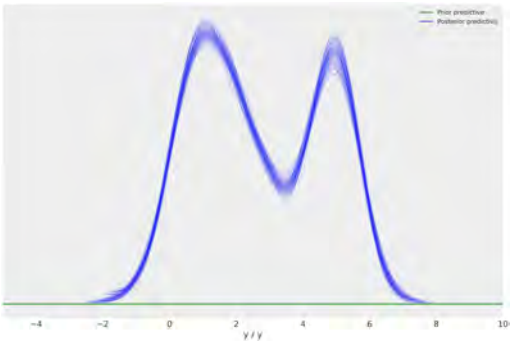

Figure D.4: PPO of identifiable Metropolis on the Wet-chicken benchmark

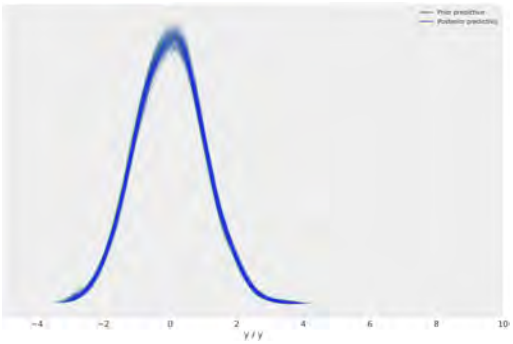

Figure D.5: PPO of non-identifiable NUTS on the Wet-chicken benchmark

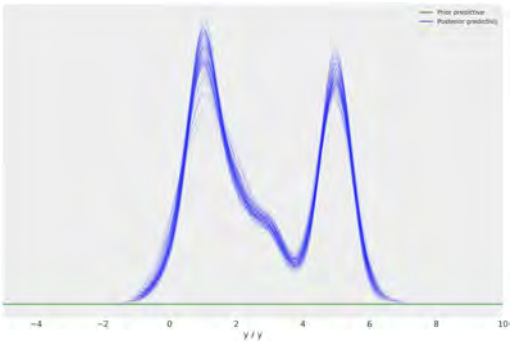

Figure D.6: PPO of identifiable NUTS on the Wet-chicken benchmark

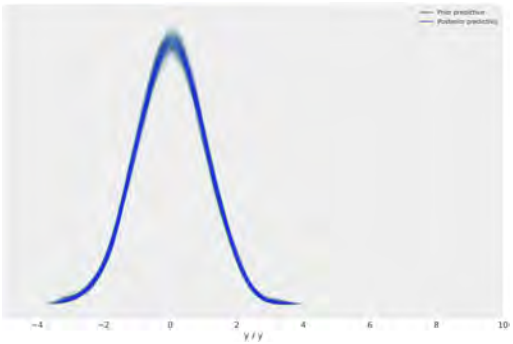

Figure D.7: PPO of non-identifiable ADVI on the Wet-chicken benchmark

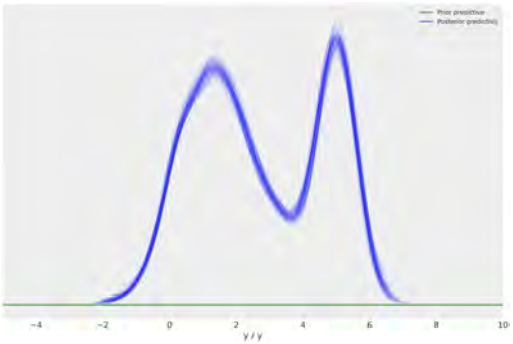

Figure D.8: PPO of identifiable ADVI on the Wet-chicken benchmark

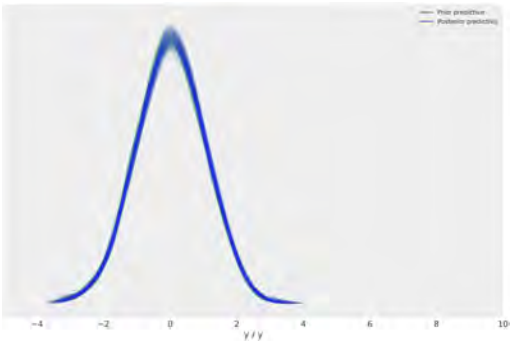

Figure D.9: PPO of non-identifiable SVGD on the Wet-chicken benchmark

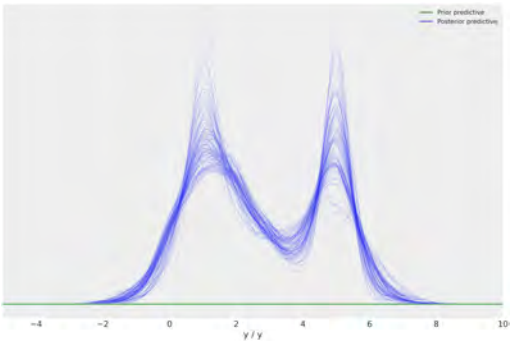

Figure D.10: PPO of identifiable SVGD on the Wet-chicken benchmark

D.2 Lunar-lander benchmark

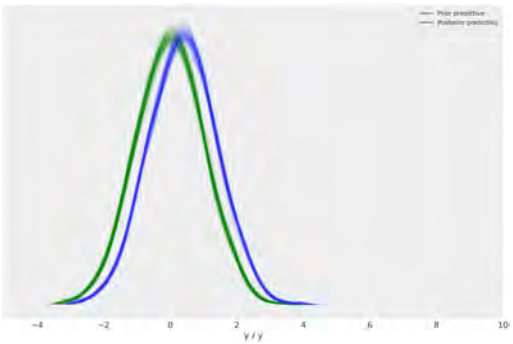

Figure D.11: PPO of non-identifiable HMC on the Lunar-lander benchmark

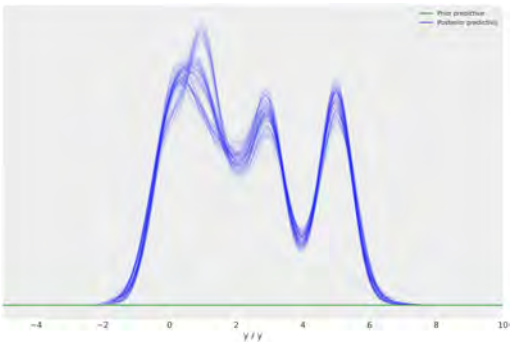

Figure D.12: PPO of identifiable HMC on the Lunar-lander benchmark

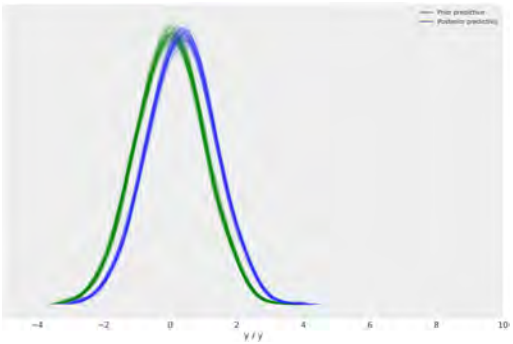

Figure D.13: PPO of non-identifiable Metropolis on the Lunar-lander benchmark

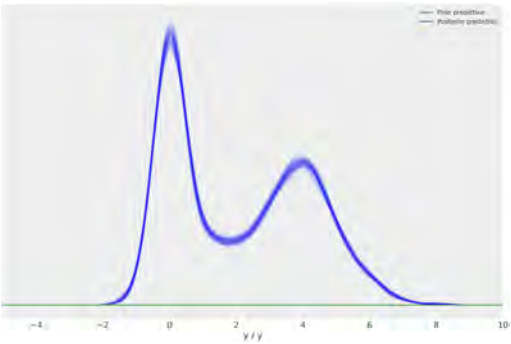

Figure D.14: PPO of identifiable Metropolis on the Lunar-lander benchmark

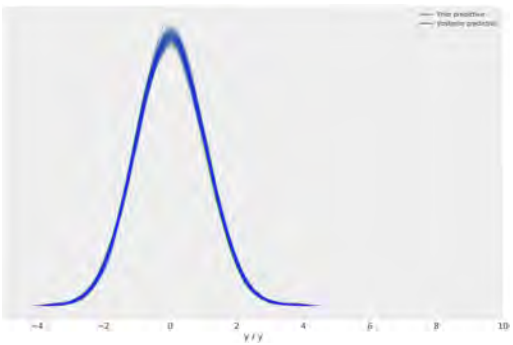

Figure D.15: PPO of non-identifiable NUTS on the Lunar-lander benchmark

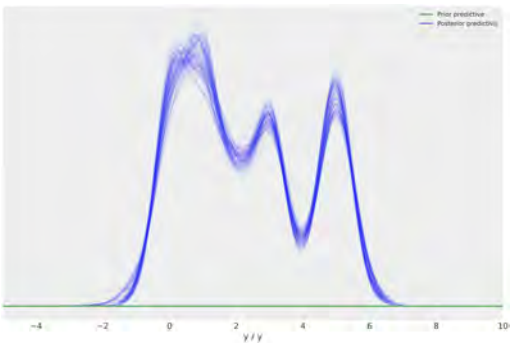

Figure D.16: PPO of identifiable NUTS on the Lunar-lander benchmark

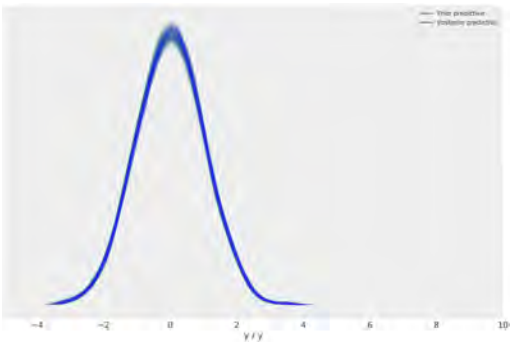

Figure D.17: PPO of non-identifiable ADVI on the Lunar-lander benchmark

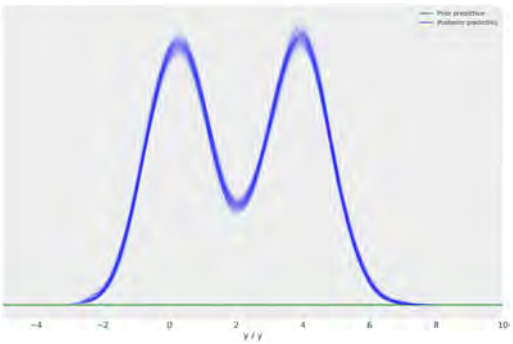

Figure D.18: PPO of identifiable ADVI on the Lunar-lander benchmark

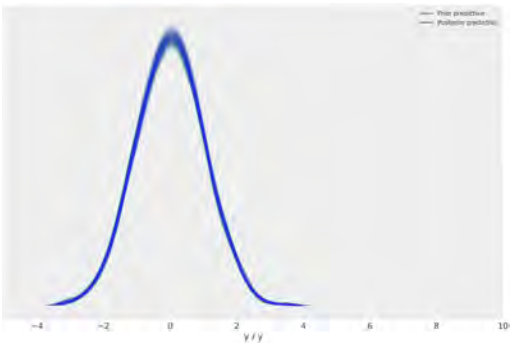

Figure D.19: PPO of non-identifiable SVGD on the Lunar-lander benchmark

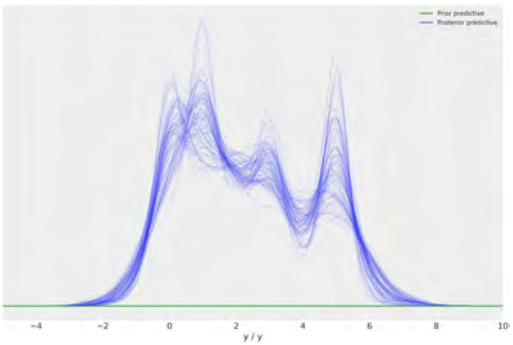

Figure D.20: PPO of identifiable SVGD on the Lunar-lander benchmark

Appendix E

Plots of the decomposed uncertainty quantification

E.1 Wet-chicken benchmark

For each pair of plots, epistemic uncertainty is presented in the left plot and aleatoric uncertainty in the right plot. Yellow indicates high uncertainty, and dark purple low uncertainty.

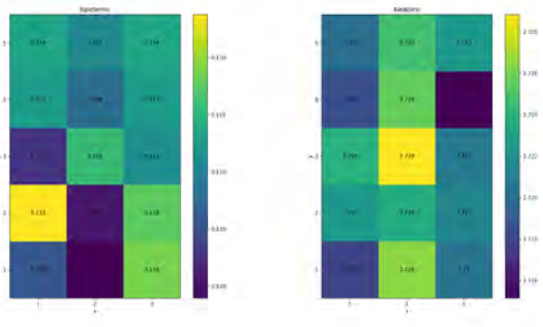

Figure E.1: UQ of non-identifiable HMC on the Wet-chicken benchmark

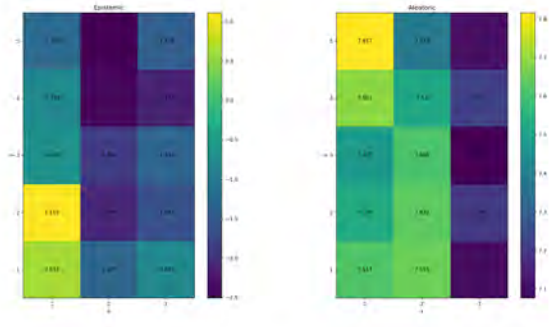

Figure E.2: UQ of identifiable HMC on the Wet-chicken benchmark

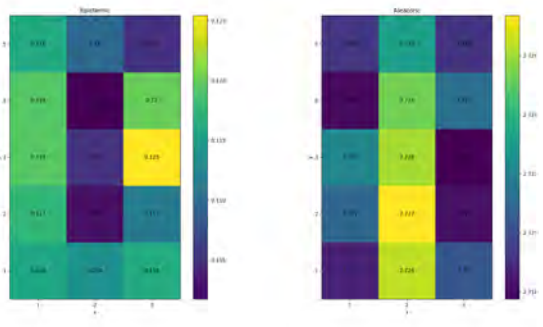

Figure E.3: UQ of non-identifiable Metropolis on the Wet-chicken benchmark

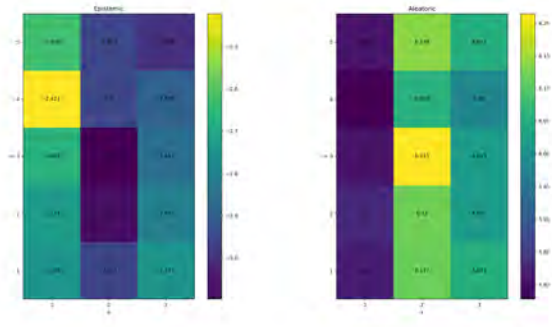

Figure E.4: UQ of identifiable Metropolis on the Wet-chicken benchmark

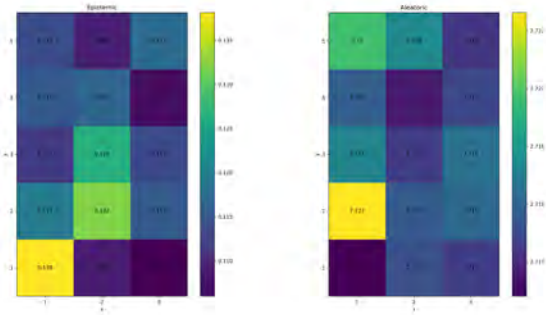

Figure E.5: UQ of non-identifiable NUTS on the Wet-chicken benchmark

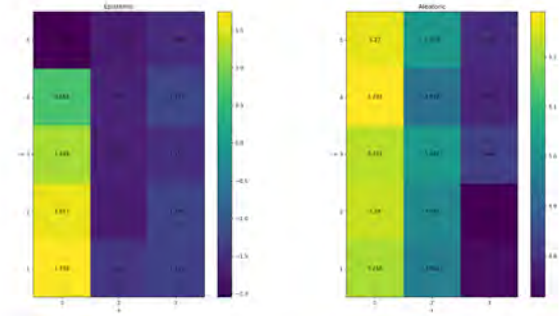

Figure E.6: UQ of identifiable NUTS on the Wet-chicken benchmark

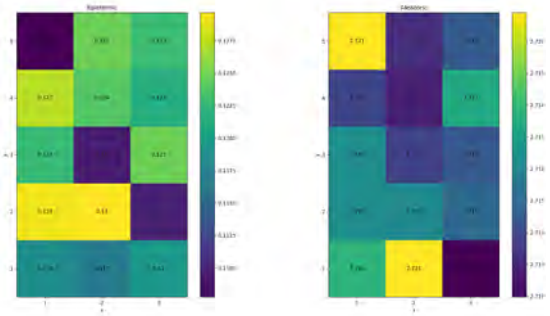

Figure E.7: UQ of non-identifiable ADVI on the Wet-chicken benchmark

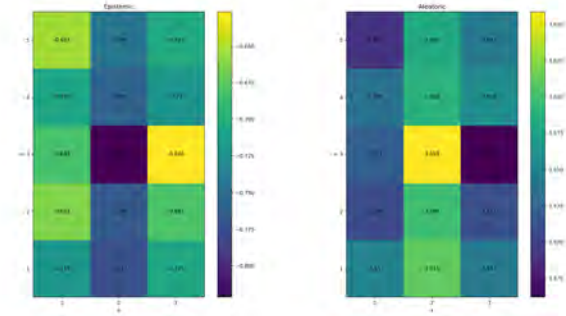

Figure E.8: UQ of identifiable ADVI on the Wet-chicken benchmark

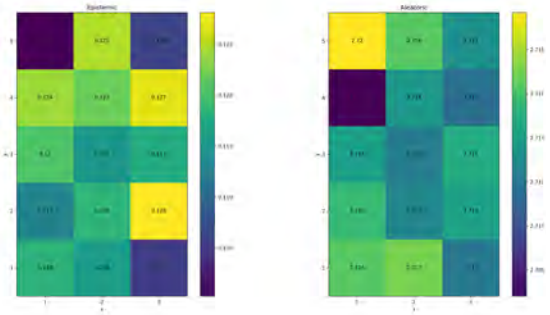

Figure E.9: UQ of non-identifiable SVGD on the Wet-chicken benchmark

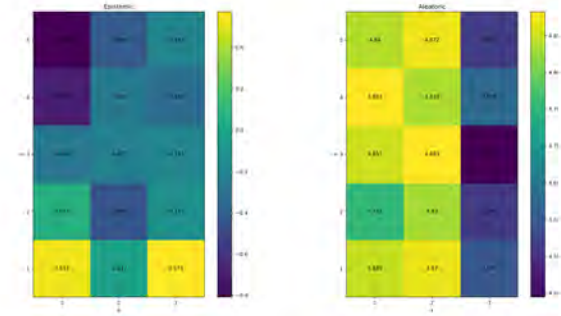

Figure E.10: UQ of identifiable SVGD on the Wet-chicken benchmark

E.2 Lunar-lander benchmark

For each pair of plots, epistemic uncertainty is presented in the left plot and aleatoric uncertainty in the right plot. Yellow indicates high uncertainty, and dark purple low uncertainty.

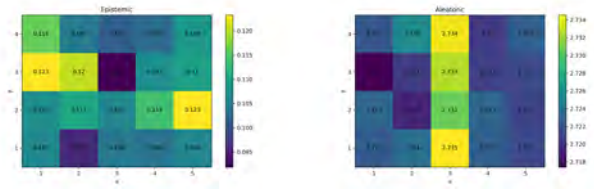

Figure E.11: UQ of non-identifiable HMC on the Lunar-lander benchmark

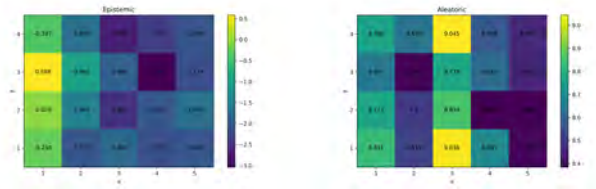

Figure E.12: UQ of identifiable HMC on the Lunar-lander benchmark

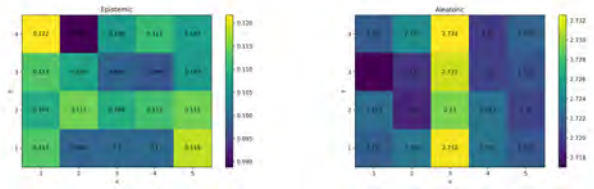

Figure E.13: UQ of non-identifiable Metropolis on the Lunar-lander benchmark

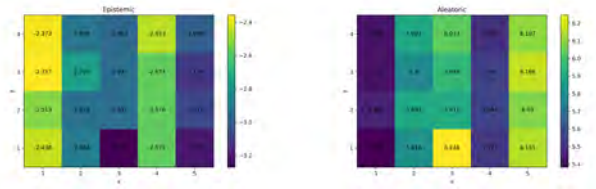

Figure E.14: UQ of identifiable Metropolis on the Lunar-lander benchmark

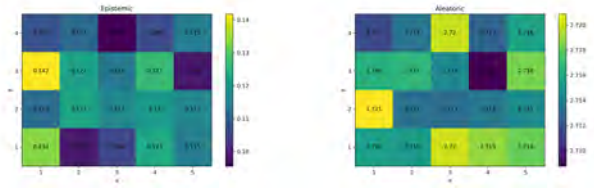

Figure E.15: UQ of non-identifiable NUTS on the Lunar-lander benchmark

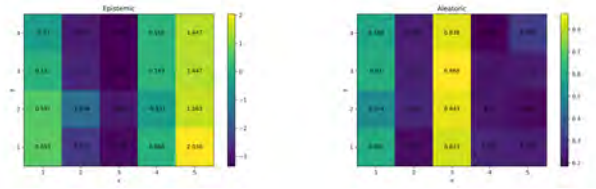

Figure E.16: UQ of identifiable NUTS on the Lunar-lander benchmark

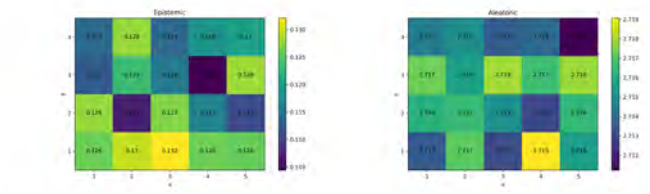

Figure E.17: UQ of non-identifiable ADVI on the Lunar-lander benchmark

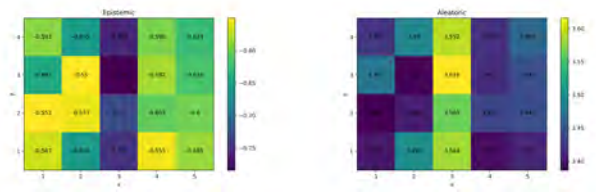

Figure E.18: UQ of identifiable ADVI on the Lunar-lander benchmark

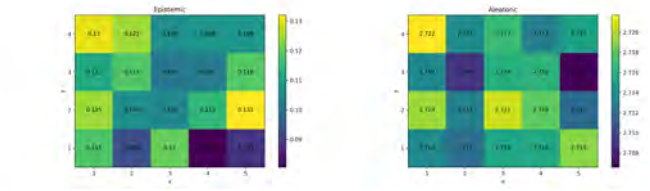

Figure E.19: UQ of non-identifiable SVGD on the Lunar-lander benchmark

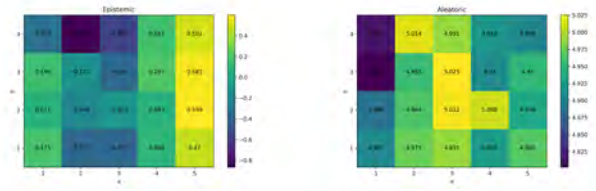

Figure E.20: UQ of identifiable SVGD on the Lunar-lander benchmark