Robust Variational Inference via Imprecise Probabilities

A REPORT SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF BACHELOR OF SCIENCE
IN THE FACULTY OF SCIENCE AND ENGINEERING

Author
Sourabh Roy

Supervisor
Michele Caprio

Department of Computer Science

April 2025

Abstract

Bayesian inference plays a pivotal role in many state-of-the-art algorithms due to its principled approach to quantifying uncertainty and incorporating prior knowledge. A key challenge to this lies in computing the posterior distribution, an often-times intractable task in real-world scenarios. Variational Inference (VI) provides an efficient alternative to exact Bayesian inference by approximating the posterior with a tractable surrogate distribution, making inference computationally feasible in practice. Yet, traditional VI methods are limited by their reliance on a single distributional family and often the mean-field approximation, which makes strong independence assumptions. While this aids scalability, it restricts expressiveness and can lead to inaccurate approximations and, consequently, severely flawed predictions.

In this work, we address this problem by proposing Robust Variational Inference (RVI), a VI method which draws upon the concept of a credal set (closed, convex set of probabilities) from Imprecise Probability (IP) theory. Our method utilises multiple parametric families to variationally approximate the posterior. The VI approximations are then used to derive a credal set, from which the best approximation, a mixture of the VI approximations, is efficiently selected. We show that our credally-selected distribution outperforms traditional single-distribution VI approximations. To demonstrate this improvement, we conduct empirical studies on various benchmark datasets, introducing significant noise and outliers manually to assess the robustness of our model across different noise levels. We further demonstrate the versatility of our approach by applying it to various tasks and integrating our VI method into different models, showcasing its broad compatibility and effectiveness.

Contents

Word Count: 10425

Chapter 1

Introduction

In order to set the scene for our work, we begin the report by motivating the study of variational inference. We then discuss the problem we are addressing in this work, its aims, and finally we outline our main contributions.

1.1 Motivation

A central challenge in modern statistics lies in approximating complex, intractable probability distributions. This is particularly crucial in Bayesian statistics, where inference revolves around computing posterior distributions over unknown quantities. Modern Bayesian statistics often rely on models with intractable exact posteriors. To address this, approximation techniques such as Markov Chain Monte Carlo (MCMC) ([41]; [17]) and Variational Inference (VI) ([21]; [51]) have become widely used, offering practical ways to obtain well-behaved posterior estimates that closely approximate the true distribution.

Markov Chain Monte Carlo (MCMC) methods generate samples from the posterior distribution through stochastic simulation and are widely regarded for their asymptotic accuracy and minimal approximation bias [44]. In contrast, Variational Inference (VI) reframes posterior estimation as an optimisation problem by approximating the true posterior with a simpler, tractable distribution. This fundamental difference in approach means that while MCMC aims to provide representative samples of the posterior, VI yields an analytical approximation of it. Consequently, VI is significantly faster and more scalable, offering a practical alternative for complex or data-intensive Bayesian models. As a result, it is especially useful in situations where we need to quickly explore a wide range of models [3]. It is important to recognise that MCMC *samples the posterior*, whereas VI *approximates it*; these are distinct tasks with different implications for downstream analysis and interpretation.

Variational Inference (VI) has been successfully applied across a wide range of fields, demonstrating its versatility in solving complex, real-world problems. While this is by no means an exhaustive list, the following examples illustrate the diversity of its applications:

- **Computational Biology:** VI plays a crucial role in computational biology, where probabilistic models are essential for analysis of genetic data. Notable applications include genome-wide association studies ([7], [29]), population genetics [35], and gene expression analysis [45].

- **Computational Neuroscience:** The field of neuroscience often deals with large, high-dimensional datasets, such as high-frequency time series or high-resolution functional magnetic resonance imaging (fMRI) data. VI has been extensively used in this area, with applications ranging from brain-computer interfaces [46] to software toolboxes designed for neuroscience and psychology research [9].

- **Natural Language Processing and Speech Recognition:** In natural language processing (NLP), VI has been employed for tasks like parsing, topic modelling [4], and part-of-speech tagging with hidden Markov models [52]. In speech recognition, VI has been used to fit complex coupled hidden Markov models [38].
- **Other Applications:** Beyond these domains, VI has found utility in a wide array of other fields, including marketing [5], optimal control and reinforcement learning ([48], [13]), astrophysics [37], and the social sciences ([12], [16]).

1.2 Existing Problems in Variational Inference

Formally, Variational Inference approximates the intractable true posterior $p^{\text{true}}(\theta \mid D)$ (where θ denotes the parameter of interest, and D the gathered data) with the best-fitting candidate $\hat{q}^{\text{VI}}$ from a fixed family $\mathcal{Q}$ of (parametrised) distributions. In other words, VI searches for element $\hat{q}^{\text{VI}} \in \mathcal{Q}$ that is the closest to $p^{\text{true}}(\theta \mid D)$. This "closeness" is measured by the Kullback-Leibler divergence (see section 3.1). Generally, the Gaussian family is used as $\mathcal{Q}$. This usage of a single family of distributions may result in the posterior $p^{\text{true}}(\theta \mid D)$ being poorly approximated ([39], [57]).

A typical cause of this, of great interest to the machine learning community, is noisy data. Noise can arise from the absence of data entries, measurement errors, and other unknown sources. They are associated with unconsidered sources of variation that affect the target variable. This renders the model unable to distinguish between random noise and systematic effects not captured by the model. In the context of VI for practical applications, noise significantly distorts Gaussian distribution, which in turn skews the mean estimate and results in overly optimistic prediction intervals. Given that real-world data is often noisy, there is a significant demand for machine learning models that are robust to such noise.

1.3 Aims and Objectives

To address the issues outlined above, this project aims to develop a novel variational inference procedure that leverages multiple parametric distributional families, rather than relying on a single family like the Gaussian, and is robust to noise.

The objectives of the project are as follows:

- Review the related works in Variational Inference and Imprecise Probability Theory.
- Discuss the application of Imprecise probability theory to our Variational Inference method. Specifically, we utilise the idea of credal sets as illustrated in [6].
- Present mathematical proofs of the proposed improvements of our method over other VI methods.
- Provide comparisons of our developed procedure with popular VI methods such as Stochastic Variational Inference (SVI), Automatic Differentiation Variational Inference (ADVI), etc. We will do so by training different models with different VI methods on two different tasks: binary classification and image reconstruction. We then compare their performance on datasets with varying degrees of noise.

Our evaluation process is mainly based on [28] and [54], where the authors performed experiments on the same machine learning tasks as mentioned above, to compare their VI methods with the state-of-the-art.

We aim for our work to bridge the gap between traditional probabilistic machine learning methods and credal machine learning techniques based on imprecise probability.

1.4 Contributions of the Work

We then make the following contributions:

- We propose the RVI algorithm that can be summarised as:
  - Specify multiple "well-behaved" distribution families $\mathcal{Q}_1, \dots, \mathcal{Q}_k$ (of course, what families to choose is application-specific task);
  - Find a VI approximation $\hat{q}_j^{\text{VI}}$ of $p^{\text{true}}(\cdot \mid D)$ for every family $\mathcal{Q}_j$, $j \in \{1, \dots, k\}$;
  - Obtain a credal set by taking the convex hull (the set of all possible convex combinations) of $\{\hat{q}_j^{\text{VI}}\}_{j=1}^k$. In formulas, $\mathcal{P}_{\text{VI}} = \text{CH}(\{\hat{q}_j^{\text{VI}}\}_{j=1}^k)$, where $\text{CH}(\cdot)$ denotes the convex hull operator;
  - Find $\hat{q}^{\text{RVI}} \in \mathcal{P}_{\text{VI}}$ that minimises the KL divergence between the elements of $\mathcal{P}_{\text{VI}}$ and $p^{\text{true}}(\cdot \mid D)$.
- We show mathematically that $\hat{q}^{\text{RVI}}$ is a "better approximation" (i.e. it is "closer" to $p^{\text{true}}(\cdot \mid D)$ in the KL divergence) than any of the $\hat{q}_j^{\text{VI}}$'s taken singularly.
- We show experimentally that our RVI method systematically outperforms classical VI methods in tasks on noisy datasets (Chapter 6).

These new results have the potential for broad application in machine learning and statistics, extending beyond just variational inference.

1.5 Project Structure

The report is structured to be read from start to finish. Every section builds on the previous one and provides additional background and motivation behind the main result.

- Chapter 2 presents the essential background knowledge and a review of related literature.
- Chapter 3 provides a discourse on Variational Inference and lays theoretical groundwork for Variational Autoencoders (VAEs) and Bayesian Logistic Regression (BLR) which are then investigated empirically in the subsequent sections of this work.
- Chapter 4 gives an introduction to Imprecise Probability Theory and justifies its relevance to our work.
- Chapter 5 introduces RVI algorithm along with comprehensive mathematical proofs establishing its key properties. It also presents the optimisation techniques that help us derive the best model from the **credal set** (discussed in Chapter 4).
- Chapter 6 includes a comparison of our method with traditional popular VI methods in a wide range of tasks on benchmark datasets.
- Chapter 7 summarises the work and proposes future research avenues.

Chapter 2

Background

This chapter first reviews key contributions from the literature that have shaped modern approaches to Variational Inference. Then it introduces the fundamental ideas of statistical inference, illustrating how data and models interact to inform conclusions about unknown quantities. Through a simple example, the mechanics of inference are made concrete, naturally leading to a discussion of one of its core difficulties: the often intractable task of computing posterior distributions. These foundations are essential for understanding the motivations behind the methods discussed in the following chapters.

2.1 Review of relevant works

Traditional VI methods approximate the posterior distribution $p^{\text{true}}(\cdot \mid D)$ by minimising the Kullback-Leibler (KL) divergence between the latter and the elements of a chosen variational family $\mathcal{Q}$. Mean-field VI [3] is widely used due to its simplicity, but often underestimates posterior correlations.

A notable limitation of standard VI is its dependence on a single family $\mathcal{Q}$, which makes the procedure sensitive to misspecification and may lead to poor estimates. Several advanced methods have been proposed to address this limitation. Stein Variational Gradient Descent (SVGD) [28] is a non-parametric method that optimises a set of particles to approximate the posterior distribution. By leveraging Stein's identity, SVGD updates particles using kernelised gradient flows, capturing posterior dependencies. Unlike standard VI, SVGD does not assume a restrictive variational family. However, SVGD still relies on a single prior, and its performance is sensitive to kernel selection.

Hamiltonian Monte Carlo (HMC) is an MCMC technique that explores the posterior using Hamiltonian dynamics [34]. HMC-based VI methods, such as HMC-VI [43], integrate gradient-based updates from HMC into variational inference to achieve more accurate posterior approximations. However, it is computationally expensive due to the need for leapfrog integrators and momentum resampling.

Normalizing flows [39] enhance VI by using a sequence of invertible transformations to map a simple base distribution (e.g., Gaussian) to a complex posterior approximation. Recent works, such as RealNVP [10] and Neural ODEs [8], further improve expressiveness using deep neural networks. However, normalizing flows require careful design of transformation functions and often suffer from optimisation instability ([24], [1]).

Another potentially powerful alternative would be to specify the approximate posterior as a mixture model, such as those developed by [20] and [14]. However, the mixture approach limits the potential scalability of variational inference since it requires evaluation of the log-likelihood and its gradients for each mixture component per parameter update, which is typically computationally expensive [39].

Our work is mostly inspired by [6] and [30] which demonstrated how using credal sets $\mathcal{P}_{\text{prior}}$ and $\mathcal{P}_{\text{lik}}$ of priors and likelihoods, respectively, in the context of Bayesian Deep Learning, outperforms the uncertainty quantification capabilities and the downstream task performances of single and ensemble of Bayesian Neural Networks.

By finding the RVI approximation $\hat{q}^{\text{RVI}}$ among the elements of the credal set $\mathcal{P}_{\text{VI}}$ of surrogate (i.e. VI approximated) posteriors, we introduce a novel way to handle misspecification while maintaining computational efficiency. Unlike SVGD and normalizing flows, our method does not rely on kernel-based updates or deep transformations, making it more stable. Compared to HMC-based VI, it remains computationally lightweight while still improving posterior robustness. Our approach can be viewed as an extension of adaptive VI methods, but instead of adjusting a single distribution, we construct a richer distribution space from the start, leading to improved inference quality.

2.2 Statistical Inference

Introduction to Statistical Inference

Statistical inference is concerned with the process of drawing conclusions about a population based on information obtained from a subset of it, known as a sample. The primary objective is to infer properties of the population, such as the mean, variance, or proportion, through the analysis of sampled data.

To facilitate this task, it is standard practice to assume that the population can be described by a family of probability distributions, parametrised by a finite-dimensional parameter. This parametrisation enables a tractable framework for inference and allows one to connect statistical models directly to the real-world quantity of interest.

Illustrative Example

As a concrete example, consider the problem of estimating the average height of middle-school students in a given region. Measuring the height of every student in the population is often infeasible. Therefore, a random sample, say of 100 students, is selected, and the sample mean is used to estimate the population mean.

To model this, one might assume that student heights follow a truncated normal distribution, with parameters μ and σ^2. This assumption captures the empirical observation that human heights tend to cluster around a central value and exhibit natural variability within a reasonable range. The parameter of interest is the mean μ, which represents the average height in the population. The standard deviation σ^2 captures the spread. By estimating these parameters from the observed data, we aim to make an informed inference about the population's average height.

Formal Framework

Formally, a statistical model consists of the following components:

- A **sample space** $\mathcal{X}$, representing the set of all possible outcomes;
- A **family of probability distributions** $\mathcal{Q} = \{p_\theta : \theta \in \Theta\}$, where:
  - θ is a parameter (or a vector of parameters);

  - $\Theta \subseteq \mathbb{R}^q$ is the parameter space;
  - Each p_θ represents a candidate distribution for the data-generating process.

The assumption is that there exists a true but unknown parameter value $\theta^* \in \Theta$ such that the data are generated according to the distribution p_{θ^*}. The task of inference is to draw conclusions about θ^*, and thereby about the properties of the population.

Probability Functions

Depending on the nature of the data, the distributions p_θ are either discrete or continuous:

- In the **discrete case**, p_θ is characterised by a probability mass function (PMF)

$$f_\theta(x) = \mathbb{P}_\theta(X = x), \quad \forall x \in \mathcal{X}.$$

- In the **continuous case**, p_θ is characterized by a probability density function (PDF)

$$\mathbb{P}_\theta(X \in [a, b]) = \int_a^b f_\theta(x)\, dx, \quad \forall a, b \in \mathbb{R}$$

  where $\mathcal{X} \subset \mathbb{R}$.

Rationale for Parametrisation

The use of parametrised models in statistical inference serves to reduce complexity by focussing attention on a limited set of interpretable parameters rather than attempting to estimate the entire underlying distribution. For instance, one may be interested in the population mean μ, the success probability p in a Bernoulli process, or the variance σ^2 representing the dispersion in the data. This focus is motivated by the fact that most inferential tasks aim to extract meaningful summaries or characteristics of the population, rather than reconstructing its full distributional form.

Bayesian Inference The development of reliable inference procedures has given rise to multiple foundational paradigms, among which *Bayesian inference* stands out as a widely adopted and principled approach. In the Bayesian framework, prior knowledge about an unknown parameter θ is expressed as a probability distribution $\pi(\theta)$ over the parameter space Θ, known as the *prior distribution.* The statistical model is specified by a family of likelihood functions $f(x \mid \theta)$, which describe the probability (or probability density) of the observed data X given the parameter θ.

Bayesian inference updates the prior belief in light of observed data using Bayes' rule. The result is the *posterior distribution*, denoted by $\pi(\theta \mid x)$, which reflects the updated belief about θ after observing the data.

Theorem 1 (Bayes' Rule)**.** *Let θ be a parameter and x be the observed data. Then the posterior distribution of θ given x is given by*

$$\pi(\theta \mid x) = \frac{f(x \mid \theta)\pi(\theta)}{f(x)}, \tag{2.1}$$

While the numerator in Equation (2.1) is typically straightforward to compute, the denominator poses a significant computational challenge.

2.3 The problem of computing posterior distributions

The denominator $f(x)$ in (2.1) is referred to as the **marginal likelihood** or **evidence**. It is defined as:

$$f(x) = \int_{\theta \in \Theta} f(x \mid \theta)\pi(\theta)\, d\theta.$$

The difficulty in calculating the exact marginal likelihood arises from two main challenges:

- The integral often does not have a closed-form solution, particularly when modelling complex real-world data where the underlying distribution cannot be represented analytically.
- Exact computation of the integral requires summing or integrating over potentially millions or billions of parameters, which is infeasible for complex models and large-scale applications.

As a result, the inability to calculate the marginal likelihood directly makes it impossible to obtain the posterior distribution analytically. This highlights the necessity of approximation techniques in modern Bayesian inference.

Chapter 3

Variational Inference

3.1 Kullback-Leibler Divergence

Before describing variational inference, we introduce the concept of Kullback-Leibler Divergence.

Definition 1. *The Kullback-Leibler (KL) divergence, denoted by $D_{KL}(q\|p)$, measures how much a probability distribution q differs from another probability distribution p by computing the expected logarithmic difference between them, where the expectation is taken under q. It is defined as:*

$$D_{KL}(q\|p) = \mathbb{E}_{q(x)}\left[\log \frac{q(x)}{p(x)}\right].$$

Explicitly, this is given by

$$D_{KL}(q\|p) = \sum_{x \in \mathcal{X}} q(x) \log \frac{q(x)}{p(x)}$$

for the discrete case, and

$$D_{KL}(q\|p) = \int_{x \in \mathcal{X}} q(x) \log \frac{q(x)}{p(x)}\, dx$$

for the continuous case.

The KL divergence can be interpreted as a form of measure of ”distance” between two distributions q and p. However, it does not satisfy the properties of a metric (or distance function) and, therefore, is not a true measure of ”distance” between distributions. Specifically:

- $D_{KL}(q\|p) \geq 0$ with equality iff $q = p$.
- KL divergence is not symmetric i.e. $D_{KL}(q\|p) \neq D_{KL}(p\|q)$.
- KL divergence does not satisfy the triangle inequality.

Since symmetry and the triangle inequality are essential properties of a metric, KL divergence does not define a proper distance function. Nevertheless, it can be viewed as a generalisation of squared Euclidean distance in certain contexts.

In this work, $D_{KL}(q\|p)$ has been used to quantify the discrepancy between the approximate posterior distribution q and the true posterior distribution p.

3.2 Introduction to Variational Inference

Variational Inference transforms posterior inference into a problem of optimisation. Let p^{true} be the true posterior distribution. We approximate p^{true} by introducing a surrogate distribution q from a "well-behaved" distribution family Q. We then minimise the loss:

$$\hat{q}^{\mathrm{VI}}(\theta) = \arg\min_{q \in \mathcal{Q}} D_{\mathrm{KL}}(q(\theta) \| p^{\mathrm{true}}(\theta \mid D)).$$

Assuming the elements of Q are parametrised by $\psi \in \Psi$, we can rewrite the optimisation problem as:

$$\begin{aligned}
\hat{q}^{\mathrm{VI}}(\theta) &= \arg\min_{\psi \in \Psi} D_{\mathrm{KL}}(q_\psi(\theta) \| p^{\mathrm{true}}(\theta \mid D)) \\
&= \arg\min_{\psi \in \Psi} \mathbb{E}_{q_\psi}\left[\log q_\psi(\theta) - \log \frac{p(D \mid \theta) p(\theta)}{p(D)}\right] \\
&= \arg\min_{\psi \in \Psi} \mathbb{E}_{q_\psi}\left[\log q_\psi(\theta) - \log p(D \mid \theta) - \log p(\theta)\right] + \log p(D)
\end{aligned}$$

Since the last term $\log p(D)$ does not depend on ψ, we define the loss function

$$L(\psi \mid D) = \mathbb{E}_{q_\psi}\left[\log q_\psi(\theta) - \log p(D \mid \theta) - \log p(\theta)\right].$$

Minimising this is equivalent to maximising:

$$\bar{L}(\psi | D) = -L(\psi | D).$$

We call $\bar{L}(\psi|D)$ the **Evidence Lower Bound (ELBO)**. Hence, maximising the ELBO minimises the KL divergence between the true and approximate posterior distributions. Current best practices in variational inference optimise the ELBO loss using mini-batches and off-the-shelf optimisers such as Adam or stochastic gradient descent. This enables variational inference to scale to problems with very large datasets. See figure 3.1 for an illustration.

3.3 Form of the variational posterior

There are two main approaches for choosing the form of the variational posterior $q_\psi(\theta)$: **fixed form** and **free form** variational inference. In fixed form VI, the functional form of the surrogate distribution q belonging to some class of distributions $\mathcal{Q}$, is specified (for example: multivariate Gaussian), and then the ELBO is optimised using gradient-based methods.

In free form VI, the functional form of q is not specified. An example of free form approach is Mean Field Variational Inference. In mean-field approach, we assume our approximate posterior factorises as

$$q_\psi(\theta) = \prod_{j=1}^{J} q_j(\theta_j)$$

where $q_j(\theta_j) = q_{\psi_j}(\theta_j)$ is the posterior over the j-th group of parameters. The optimal $q_\psi(\theta)$ is derived by maximising the ELBO with respect to each group of variational parameters, one at a time, in a coordinate ascent manner.

Let us now look at the different kinds of variational inference methods that we will encounter in this work. Interested readers can go through [33] and [47] for more details.

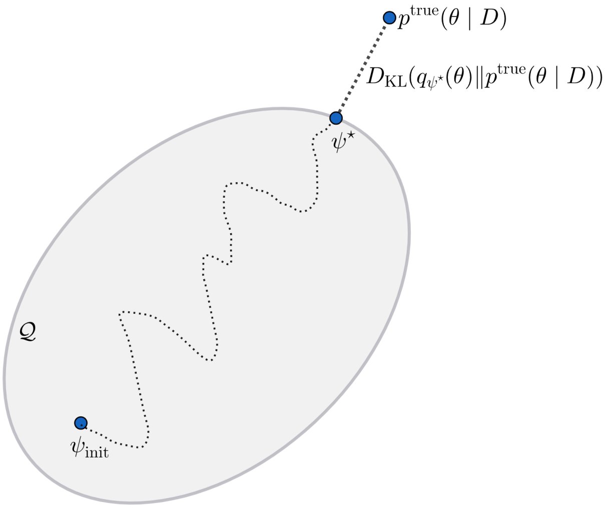

$p^{\text{true}}(\theta \mid D)$
$D_{\text{KL}}(q_{\psi^\star}(\theta) \| p^{\text{true}}(\theta \mid D))$
$\psi^\star$
$\mathcal{Q}$
ψ_{init}

Figure 3.1: Illustration of variational inference. The large oval represents the set of variational distributions $\mathcal{Q} = \{q_\psi(\theta) : \psi \in \Psi\}$, where Ψ is the set of possible variational parameters. The true posterior distribution $p^{\text{true}}(\theta \mid D)$ is assumed to lie outside this set. Our goal is to find the best approximation to $p^{\text{true}}(\theta \mid D)$ within our variational family—this is the point ψ^* that minimises the KL divergence to the true posterior. The optimization begins from a randomly initialized point ψ_{init}. Figure inspired from [33]

3.3.1 Stochastic Variational Inference

Stochastic Variational Inference [18] or SVI was designed to address the computational challenges of variational inference, particularly when dealing with large datasets, by using stochastic optimisation. To find the variational parameters, we usually need to optimise the ELBO with respect to the entire dataset, which requires computing the sum of the ELBOs for each of the N data samples. For large N, this is computationally expensive. In SVI, instead of using the full dataset, we use a random mini-batch of $M = |\mathcal{M}|$ examples from the dataset, and then make the approximation:

$$\bar{L}(\psi|D) = \sum_{i=1}^{N} \bar{L}(\psi_i|x_i) \approx \frac{N}{M} \sum_{x_i \in \mathcal{M}} \bar{L}(\psi_i|x_i)$$

This enables us to incorporate SVI with other stochastic optimisation algorithms such as SGD. SVI is commonly implemented using Pyro which we will see later in the Experiments section.

3.3.2 Amortized Variational Inference

Amortized Variational Inference (A-VI) [32] is famously used in VAEs (which we will see in detail in the later sections). While traditional variational inference methods such as mean field fit a separate parametric distribution over each latent variable (or parameter θ), A-VI learns an *inference network* which maps each observation to the posterior of its corresponding latent variable (or parameter). In other words, the inference function is *shared* across all latent variables (or parameters). This sharing significantly reduces the computational burden associated with Stochastic Variational Inference (SVI). Recall that in SVI, separate variational parameters ψ_i must be optimised for each data point i in the minibatch, which can be computationally expensive.

In A-VI, we train a network such that $\psi_i = f_\phi^{inf}(x_i)$, where f^{inf} is the inference network. Thus:

$$q(\theta_i|\psi_i) = q(\theta_i|f_\phi^{inf}(x_i)) = q_\phi(\theta_i|x_i)$$

and the corresponding ELBO is:

$$\bar{L}(\theta, \phi|D) = \sum_{i=1}^{N} (\mathbb{E}_{q_\phi(\theta_i|x_i)}[log(p(x_i, \theta_i)) - log(q_\phi(\theta_i|x_i))])$$

and we can now follow the example of SVI and use minibatches for faster optimisation.

3.3.3 Automatic Differentiation Variational Inference

Automatic Differentiation Variational Inference or ADVI [25] is one of the most commonly used variational inference methods. In fact, most python libraries such as PyMC3 use ADVI as the default implementation. The key idea in ADVI is transform the constrained parameters to unconstrained form, in $\mathbb{R}^k$.

As before, we are approximating $p^{true}(\theta|D)$, where $\theta \in \Theta$ with Θ as a k-dimensional parameter space. Let $T : \Theta \to \mathbb{R}^k$ be a bijective mapping. Let $u = T(\theta)$ be the unconstrained variables formed as a result of mapping θ to $\mathbb{R}^k$. Now we use a distribution to approximate the posterior for u. Usually, a Gaussian distribution is used. In other words, $q_\psi(u) = \mathcal{N}(u|\mu_d, \Sigma)$, where $\psi = (\mu, \Sigma)$.

Change of variables formula Let f be a bijection that maps $\mathbb{R}^n$ to $\mathbb{R}^n$. The change of variables formula tells us that:

$$p_y(y) = p_x(f^{-1}(y)) \left|\det\left(J_{f^{-1}}(y)\right)\right|,$$

where $J_{f^{-1}}(y)$ is the Jacobian matrix of the inverse mapping f^{-1} evaluated at y, and $|\det(J_{f^{-1}}(y))|$ denotes the absolute value of the determinant.

Using the change of variables formula, we have:

$$p(u) = p(T^{-1}(u))|det(J_{T^{-1}}(u))|$$

where $J_{T^{-1}}$ is the Jacobian. Accordingly, the ELBO is:

$$\bar{L}(\psi) = \mathbb{E}_{u \sim q_\psi(u)}[log(p(D|T^{-1}(u))) + log(p(T^{-1}(u))) + log|(det(J_{T^{-1}}(u)))|] + \mathbb{H}(\psi)$$

where $\mathbb{H}(\psi)$ denotes the entropy of $q_\psi(u)$. This is tractable and we can use SGD to optimise it along with minibatches for large datasets.

Note that this technique is applicable for any distribution for which we can define a bijection to $\mathbb{R}^k$. We will see more about this in the experiments section.

3.4 Applications

In this section, we present two of the most widely used applications of variational inference. These serve as benchmarks for evaluating our proposed method, which we will revisit and assess in Chapter-6.

3.4.1 Bayesian Logistic Regression

Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ the output space. Suppose we have access to a large population of labelled objects, denoted by $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where each $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, with $x_i \in \mathbb{R}^k$ representing a feature vector for some $k \in \mathbb{N}$, and $y_i \in \{0, 1\}$ indicating the corresponding class label. For instance, in a spam classification task, $x_r = [x_{1r}, x_{2r}]$ may represent the r-th email, where x_{1r} denotes the word count and x_{2r} the number of slang words; $y_r \in \{0, 1\}$ indicates whether the email is spam.

The goal is to approximate the underlying function

$$f(x) : \mathcal{X} \to \mathcal{Y} \quad \text{such that} \quad f(x) \approx P(Y = 1 \mid X = x),$$

so as to minimise the empirical risk

$$\hat{R}(f) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i)).$$

Note that in Logistic Regression L is the cross-entropy loss function.

Logistic Regression. In classical logistic regression, we aim to find a parameter vector $a \in \mathbb{R}^k$ and a scalar bias term $b \in \mathbb{R}$ such that the sigmoid function

$$\sigma(a^\top x + b) = \frac{1}{1 + e^{-(a^\top x + b)}}$$

approximates the conditional probability $P(Y = 1 \mid X = x)$. The sigmoid ensures that the output lies in the interval $[0, 1]$, making it suitable for probabilistic interpretation.

Bayesian Logistic Regression. In the Bayesian framework, we incorporate prior beliefs about the parameters by placing a prior distribution over them. Typically, the parameters a and b are concatenated into a single parameter vector θ, and a prior distribution $P(\theta)$ is defined over this space. Using Bayes' rule, we can express the posterior distribution as:

$$P(\theta \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \theta) P(\theta)}{P(\mathcal{D})},$$

where $P(\mathcal{D} \mid \theta)$ is the likelihood of the observed data, and $P(\mathcal{D})$ is a normalizing constant, also known as the *marginal likelihood* or *evidence*.

The likelihood function in Bayesian logistic regression is given by:

$$P(\mathcal{D} \mid \theta) = \prod_{i=1}^{N} P(y_i \mid x_i, \theta),$$

where

$$P(y_i = 1 \mid x_i, \theta) = \sigma(x_i^\top \theta), \quad P(y_i = 0 \mid x_i, \theta) = 1 - \sigma(x_i^\top \theta).$$

Hence, the full likelihood becomes:

$$P(\mathcal{D} \mid \theta) = \prod_{i=1}^{N} \sigma(x_i^\top \theta)^{y_i} \big(1 - \sigma(x_i^\top \theta)\big)^{1 - y_i}.$$

Posterior Approximation via Variational Inference. As discussed earlier, computing the marginal likelihood $P(\mathcal{D}) = \int P(\mathcal{D} \mid \theta)P(\theta)\,d\theta$ is intractable in most real-world scenarios. To circumvent this, we employ *Variational Inference* (VI) to approximate the posterior distribution $P(\theta \mid \mathcal{D})$ with a simpler, tractable distribution $q(\theta)$ drawn from a predefined family. The objective is to make $q(\theta)$ as close as possible to the true posterior, typically by minimising the Kullback–Leibler (KL) divergence.

Benefits of the Bayesian Approach Unlike classical logistic regression, which yields a single point estimate of the parameters, the Bayesian framework produces a full posterior distribution over the parameter vector θ. This distributional perspective enables several advantages. First, it naturally provides a principled quantification of uncertainty in parameter estimates, which is especially valuable in scenarios with limited or noisy data. Rather than relying on a single, potentially overconfident estimate, the Bayesian approach reflects the range of plausible parameter values given the observed data. Second, it allows for the construction of *credible intervals*. These are intervals within which the parameters lie with a specified posterior probability (e.g., 95%). These intervals are directly interpretable and offer a meaningful way to communicate uncertainty in applications where decisions are sensitive to estimation error, such as healthcare or finance. Finally, predictions are made by marginalising over the posterior distribution of the parameters, rather than conditioning on a fixed point estimate. This sampling-based prediction integrates model uncertainty into the decision-making process, leading to more robust and calibrated predictions.

3.4.2 Variational Autoencoders

Variational Autoencoders (VAEs) [22] are a class of deep generative models that learn to generate new data samples by modelling the underlying distribution of the training data. Their capabilities extend beyond generation. They are also used for tasks such as denoising, representation learning, and dimensionality reduction. A prominent example is image generation, where a VAE trained on a dataset of images can produce novel images that resemble the originals in a coherent and structured manner.

Conceptually, the VAE architecture comprises three main components: an *encoder*, a *bottleneck*, and a *decoder*. The encoder maps the input data x to a latent representation z, capturing the essential features of the data in a compressed form. The bottleneck forms the interface between the encoder and decoder, representing the latent space in a lower-dimensional manifold that encodes high-level attributes of the data. The decoder then attempts to reconstruct the input from this latent representation, effectively reversing the encoding process. Through this reconstruction objective, the VAE learns both a meaningful latent space and a generative model of the data. An illustration will help us visualise the entire mechanism. As shown in the figure below, the encoder maps the input data to a latent space, and the decoder reconstructs the input from these latent variables.

Before delving into the specifics of the architecture and training procedure, we begin by discussing the concept of latent space in greater detail.

Latent Space The latent space is a lower-dimensional space that captures the essential, unobserved factors called *latent variables* that underlie the structure of the data. These latent variables, typically denoted by z, are not directly observed but are assumed to influence the observed data x. For instance, if we are analysing the weights of vehicles without access to their physical appearance, the type of vehicle (e.g., truck or bike) acts as a latent variable. It affects the weight but is not part of the recorded features.

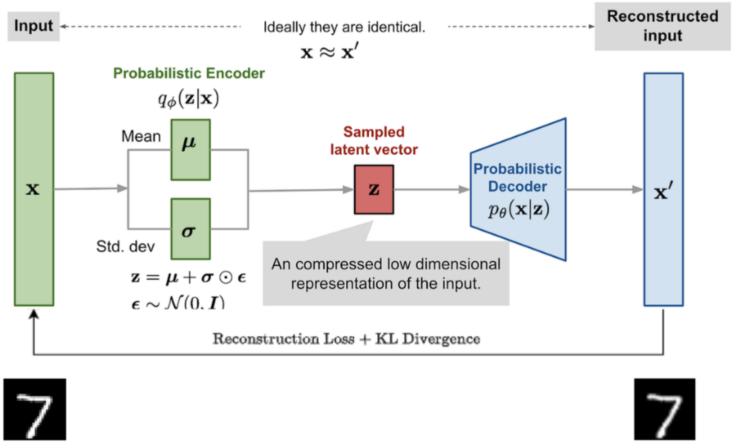

Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
$\mathbf{x}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon}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

Figure 3.2: Illustration of how a Variational Autoencoder (VAE) works in image reconstruction tasks. Diagram adapted from [53]

By projecting data points x into this latent space, we obtain a more compact and structured representation. Mathematically, this involves associating the observed data distribution $p(x)$ with a corresponding distribution over the latent variables $p(z)$. Understanding how this mapping is learnt and used is central to the functioning of variational autoencoders, and we will soon explore this in detail.

We are now ready to explore the mechanism behind Variational Autoencoders (VAEs), following the exposition in [23]. Let x represent an observed variable that is randomly sampled from an unknown underlying process with an unknown true distribution $p^*(x)$. The goal is to find the model parameters θ such that the chosen model $p_\theta(x)$ approximates the true distribution $p^*(x)$, i.e.,

$$p_\theta(x) \approx p^*(x) \quad \text{for any observed data point } x.$$

To achieve this, we employ *Maximum Likelihood Estimation (MLE)*, which seeks to optimise the parameters θ by maximising the likelihood of the observed data. Specifically, we maximise the log-likelihood over the dataset D:

$$\theta^{MLE} = \arg\max_\theta \sum_{x_i \in D} \log(p_\theta(x_i)),$$

where D represents the dataset. This optimisation problem is typically solved using gradient-based approaches, where we need to compute the gradient of the marginal log-likelihood $\log(p_\theta(x))$.

The gradient of the log-likelihood is given by:

$$\nabla \log(p_\theta(x)) = \int p_\theta(z|x) \nabla_\theta \log(p_\theta(x, z))\, dz.$$

However, directly computing the gradient is challenging because $p_\theta(z|x)$, the posterior distribution of the latent variables z given the observed data x, is intractable. To address this, VAEs use *amortized variational inference* to approximate the posterior $p_\theta(z|x)$.

Encoder The encoder, or the *inference model*, denoted as $q_\phi(z|x)$, addresses the challenge of intractable posterior inference by approximating the true posterior $p_\theta(z|x)$ with a tractable distribution. Here, ϕ represents the variational parameters, which are optimised to ensure that

$$q_\phi(z|x) \approx p_\theta(z|x).$$

The variational distribution $q_\phi(z|x)$ is typically modelled as a directed graphical model parametrised by a neural network. Consequently, the variational parameters ϕ correspond to the weights and biases of the neural network. For example, the encoder neural network may output the mean and the log of the standard deviation:

$$(\mu, \log(\sigma)) = \text{EncoderNeuralNet}_\phi(x),$$

where μ and σ are the parameters of the approximate posterior distribution. The distribution itself is typically a Gaussian with diagonal covariance:

$$q_\phi(z|x) = \mathcal{N}(z; \mu, \text{diag}(\sigma)).$$

As we discussed earlier, this approach of sharing variational parameters across data points is known as amortized variational inference. The advantage of amortization over traditional variational inference is that it eliminates the need for a per-datapoint optimisation loop over the entire dataset. Instead, by leveraging shared parameters and using stochastic gradient descent (SGD), we achieve more efficient optimisation.

Loss function for VAE We now present the derivation of the loss function used in VAEs, based on [22].

$$\begin{aligned}
\log p_\theta(x) &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x)\right] && (3.1)\\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{p_\theta(z|x)}\right)\right] && (3.2)\\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\cdot\frac{q_\phi(z|x)}{p_\theta(z|x)}\right)\right] && (3.3)\\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right)\right] + \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{q_\phi(z|x)}{p_\theta(z|x)}\right)\right] && (3.4)\\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right)\right] + D_{\text{KL}}(q_\phi(z|x)\|p_\theta(z|x)) && (3.5)\\
&= L_{\theta,\phi}(x) + D_{\text{KL}}(q_\phi(z|x)\|p_\theta(z|x)), && (3.6)
\end{aligned}$$

where the first term $L_{\theta,\phi}(x)$ is the Evidence Lower Bound (ELBO), and the second term $D_{\text{KL}}(q_\phi(z|x)\|p_\theta(z|x))$ represents the Kullback-Leibler (KL) divergence between the approximate posterior $q_\phi(z|x)$ and the true posterior $p_\theta(z|x)$.

Maximising the ELBO serves two primary purposes. First, it approximately maximises the marginal likelihood $p_\theta(x)$, thereby improving the performance of the generative model. Second, it minimises the KL divergence between the approximate posterior $q_\phi(z|x)$ and the true posterior $p_\theta(z|x)$, ensuring that the approximation becomes more accurate. The KL divergence term functions as a regularizer, serving two roles. It quantifies the discrepancy between the approximate posterior and the true posterior, and it measures the gap between the ELBO and the marginal likelihood $\log p_\theta(x)$, often referred to as the tightness of the bound. The smaller the KL divergence, the better $q_\phi(z|x)$ approximates the true posterior, which in turn reduces the gap and improves the overall model.

Reparameterization Trick The ELBO is an expectation whose gradients cannot be computed directly. The *Reparameterization Trick* allows us to differentiate the ELBO. The key idea is to express the random variable $z \sim q_\phi(z|x)$ as a differentiable (and invertible) transformation of another random variable ϵ, independent of x or ϕ, such that:

$$z = g(\epsilon, \phi, x)$$

Here, ϵ is a random variable whose distribution is independent of x or ϕ. This change of variables allows us to rewrite the expectation in terms of ϵ. Under this reparameterization, we can replace the expectation with respect to $q_\phi(z|x)$ with an expectation with respect to $p(\epsilon)$. The ELBO is then rewritten as:

$$L_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x,z) - \log q_\phi(z|x)\right]$$

$$= \mathbb{E}_{p(\epsilon)}\left[\log p_\theta(x,z) - \log q_\phi(z|x)\right]$$

where $z = g(\epsilon, \phi, x)$.

In summary, the variational parameters ϕ influence the objective function f through the random variable $z \sim q_\phi(z|x)$. To optimise with SGD, we need to compute the gradients $\nabla_\phi f$. Direct differentiation is not feasible because gradients cannot flow through the random variable z. However, by reparameterizing z as a deterministic function of ϕ, x, and a new random variable ϵ, we can ”externalise” the randomness and backpropagate through z to compute $\nabla_\phi f$.

The following illustration shows the reparameterization trick in action.

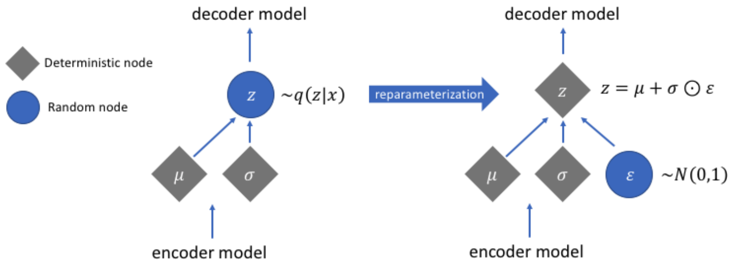


Figure 3.3: The Reparameterization Trick: transforming the random variable $z \sim q_\phi(z|x)$ into a deterministic function of ϵ, which allows for the gradient of the ELBO to be computed via backpropagation. Figure inspired from [23].

In this chapter, we have explored the general framework of Variational Inference (VI) Variational Autoencoders (VAEs) and Bayesian Linear Regression (BLR). While these methods offer powerful solutions, they often rely on precise assumptions about distributions and model structure. In the next chapter, we broaden this perspective by introducing Imprecise Probabilities.

Chapter 4

Imprecise Probabilities

In traditional Bayesian methods, a single prior distribution is arbitrarily selected and subsequently updated based on observed data. However, this approach is susceptible to *prior misspecification*, which can significantly impact the reliability of the results. To mitigate this issue, an alternative approach is to define the prior as a *set* or an *interval* of distributions rather than a fixed choice. By considering a range of plausible priors, this framework enhances robustness and provides more reliable inferences. This concept forms the foundation of *Imprecise Probability Theory*, which aims to account for uncertainty more comprehensively than standard Bayesian methods. The material presented in this chapter draws largely from [2] and [6].

4.1 Introduction to Imprecise Probability Theory

Imprecise Probabilities (IPs) allow to capture ambiguity in the user's beliefs. More formally, in a Bayesian setting, they allow to take into account the difficulty of eliciting unique prior and likelihood distributions. This might happen when either little is known initially regarding the parameter of interest, or when the available information is insufficient to specify a single distribution regulating the data generating process.

By *ambiguity*, then, we refer to the user's epistemic condition of not being able to fully determine true ideal distributions. To represent ambiguity, Imprecise Probabilists use a set of distributions $\mathcal{P}$. If such a set represents ambiguity around the parameter distribution (as it will be the case in the present paper), we have that each element in $\mathcal{P}$ is a reasonable fit for the agent's beliefs about the parameters of interest. The greater the distance between the infimum and supremum of $\mathcal{P}$, that is, its lower and upper envelopes, the higher the ambiguity.

4.2 Credal Sets

Definition 2 (Finitely Generated Credal Set)**.** *Let $\{q_1, q_2, \dots, q_k\}, \quad k \in \mathbb{N}$, be a finite set of probability distributions on a generic parameter space Θ, such that for all $j \in \{1, \dots, k\}$, q_j cannot be written as a convex combination of the other $k-1$ elements of the collection. Then, a* ***Finitely Generated Credal Set (FGCS)*** *(induced by $\{q_j\}_{j=1}^k$) is the set $\mathcal{P} = CH(\{q_j\}_{j=1}^k)$. That is, for all $q \in \mathcal{P}$,*

$$q(\theta) = \sum_{j=1}^{k} \lambda_j q_j(\theta), \quad \theta \in \Theta$$

where $\lambda_j \in [0,1]$ *for all* $j \in \{1,\dots,k\}$, *and* $\sum_{j=1}^{k} \lambda_j = 1$.

We refer to the collection that generates FGCS $\mathcal{P}$ as the extreme elements of $\mathcal{P}$, which we denote by ex$\mathcal{P}$. We illustrate the idea with the following diagram.

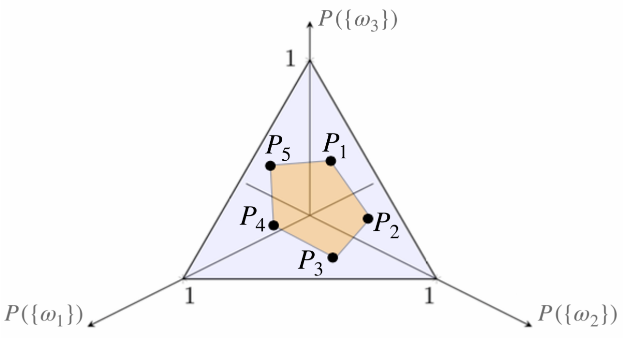


Figure 4.1: In a 3-class classification setting, let $\Omega = \{\omega_1, \omega_2, \omega_3\}$. A probability measure P on Ω can be represented as a probability vector. Since the entries are positive and sum to 1, P lies in the unit simplex, shown as the purple triangle in the figure. Now, suppose we have a set $\Pi = \{P_1, \dots, P_5\}$ of probability vectors. Taking their convex hull, $\Pi' = \text{Conv}(\Pi)$, gives the orange pentagon in the figure. This pentagon is a convex shape with a finite number of extreme points, and it represents a finitely generated credal set. This illustration is adapted from [6].

How does this approach benefit our Variational Inference (VI) problem? Recall that in VI, we approximate a complex posterior distribution using a single, simple family of distributions (e.g., the Normal distribution). However, by restricting the approximation to a single family, VI methods also become susceptible to *model misspecification*, as discussed earlier. To address this issue, we can extend the standard approach by representing our surrogate as a *set* of distribution families rather than a single choice. By constructing its corresponding *credal set* and using optimisation techniques, as discussed in the upcoming sections, we can obtain a more robust and accurate approximation.

Chapter 5

Methodology

Robust Variational Inference (RVI) aims to enhance traditional Variational Inference by considering multiple ”well-behaved” distributional families rather than relying on a single family. The core idea is to construct a credal set: a collection of VI approximations, and identify the one that best represents the true distribution. Fig 5.1 gives an illustration.

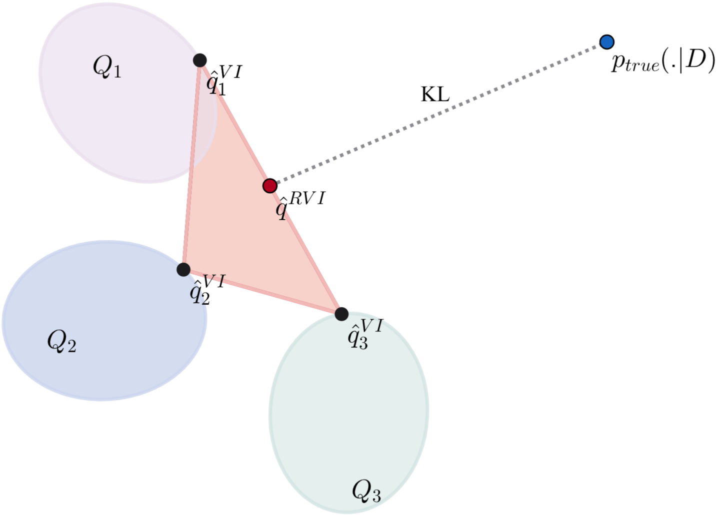


Figure 5.1: Illustration of RVI.

However, directly constructing this credal set poses significant challenges. To address this, we first reformulate the problem into an equivalent but more tractable approach, which we will now explore.

5.1 Constructing Credal Set of Surrogate Distributions

Let $p^{\text{true}}(\theta \mid D)$ be the true posterior that our procedure needs to approximate. As we briefly mentioned earlier, we begin our analysis by specifying $k \in \mathbb{N}$ “well-behaved” distribution families $\mathcal{Q}_1, \ldots, \mathcal{Q}_k$. We then denote by $\hat{q}_1^{\text{VI}}, \ldots, \hat{q}_k^{\text{VI}}$ the VI approximations for $p^{\text{true}}(\theta \mid D)$ obtained from $\mathcal{Q}_1, \ldots, \mathcal{Q}_k$, respectively. Let $\mathcal{P}_{\text{VI}} = \text{CH}(\{\hat{q}_j^{\text{VI}}\}_{j=1}^k)$ be the finitely generated (variational) credal set, and denote the RVI approximator as $\hat{q}^{\text{RVI}}(\theta) := \arg\min_{q \in \mathcal{P}_{\text{VI}}} D_{\text{KL}}[q(\theta) \| p^{\text{true}}(\theta \mid D)]$, for all $\theta \in \Theta$.[1] That is, our RVI method finds the element in $\mathcal{P}_{\text{VI}}$ that is closest (in the KL sense) to the true posterior $p^{\text{true}}(\theta \mid D)$.

[1] We tacitly assume that the argmin is a singleton.

Once we have the credal set, we can treat it as a distributional family, perform variational inference over its elements, and derive a final approximation in the form of a mixture model composed of $\hat{q}_1^{\mathrm{VI}}, \dots, \hat{q}_k^{\mathrm{VI}}$. However, it is important to note that $\mathcal{P}_{\mathrm{VI}}$ contains *uncountably many* elements, as there are infinitely many possible combinations of mixture weights $\lambda \in [0,1]$. This makes the explicit construction of $\mathcal{P}_{\mathrm{VI}}$ computationally intractable. One naive approach would be to sample a large number of weight vectors from a Dirichlet distribution, perform variational inference on each resulting mixture, and select the one with the highest ELBO. However, this strategy is computationally expensive and inefficient, as we are unlikely to stumble upon the optimal mixture model through random sampling alone.

Alternatively, we can rephrase the problem as one of finding the best weights for which the KL Divergence is minimised. Mathematically:

$$\min_{\lambda_j,\, j \in \{1,\dots,k\}} D_{\mathrm{KL}} \left(\sum_{j=1}^{k} \lambda_j \hat{q}_j^{\mathrm{VI}}(\theta) \middle\| p^{\mathrm{true}}(\theta \mid D) \right),$$

subjected to

$$\sum_{j=1}^{k} \lambda_j = 1, \quad \lambda_j \in [0,1], \quad \forall j \in \{1,\dots,k\}.$$

This is a standard constraint optimisation problem, which is tractable. It also eliminates the need to perform an additional variational inference, saving computation. But before looking at the optimisation techniques for solving this problem, let us first review some results which guarantee that such a set of optimum weights exist and that the resulting model is the best model.

Theorem 2 (The RVI Approximator is a Mixture of $\hat{q}_1^{\mathrm{VI}}, \dots, \hat{q}_k^{\mathrm{VI}}$)**.** *There exists a vector $\lambda^\star = (\lambda_1^\star, \dots, \lambda_k^\star)^\top$ in the $(k-1)$-unit simplex such that $\hat{q}^{RVI}(\theta) = \sum_{j=1}^{k} \lambda_j^\star \hat{q}_j^{VI}(\theta)$, for all $\theta \in \Theta$.*

Proof. Notice that $\mathcal{P}_{\mathrm{VI}}$ is convex by construction, and compact. The latter is true for any topological vector space Θ, since $\mathcal{P}_{\mathrm{VI}}$ is the convex hull of finite set. Then, by its definition, $\hat{q}^{\mathrm{RVI}}$ is an element of $\mathcal{P}_{\mathrm{VI}}$. In turn, this means that it can be written as a convex combination of its extreme elements $\mathrm{ex}\mathcal{P}_{\mathrm{VI}}$. In other words, there exists a vector $\lambda^\star = (\lambda_1^\star, \dots, \lambda_k^\star)^\top$ in the $(k-1)$-unit simplex such that $\hat{q}^{\mathrm{RVI}} = \sum_{j=1}^{k} \lambda_j^\star \hat{q}_j^{\mathrm{VI}}$, as desired. □

This tells us that, to find $\hat{q}^{\mathrm{RVI}}$, it is sufficient to determine the optimal weights $\lambda_j^\star$ that minimise the Kullback-Leibler (KL) divergence of $\sum_{j=1}^{k} \lambda_j \hat{q}_j^{\mathrm{VI}}$ from the true posterior p^{true}. We now show that the variational distribution $\hat{q}^{\mathrm{RVI}}$ selected by our method is indeed closer to the true posterior than any of the VI approximations that generate $\mathcal{P}_{\mathrm{VI}}$.

Theorem 3 (The RVI is a better approximator than each of the individual $\hat{q}_1^{\mathrm{VI}}, \dots, \hat{q}_k^{\mathrm{VI}}$)**.**

$$D_{KL}\left[\hat{q}^{RVI}(\theta) \| p^{true}(\theta \mid D)\right] \leq D_{KL}\left[\hat{q}_j^{VI}(\theta) \| p^{true}(\theta \mid D)\right], \quad \forall j \in \{1,\dots,k\}.$$

Proof. Immediate from the definition of $q^{\mathrm{RVI}}(\theta)$ and [[6], Proposition 8]. □

5.2 Optimisation techniques for best model

We present two different approaches to solving the constrained optimisation problem. Due to the lack of support for certain distributions (e.g., the skewed normal in Pyro) and other implementation limitations in Python libraries, we explored multiple methods to determine the weights in our experiments.

5.2.1 Stacking with bootstrapped-Pseudo-BMA

Stacking, as introduced in [55], is a principled method for model averaging. Rather than selecting a single best approximation from a set of candidates, stacking forms an optimal convex combination of the candidate models. In the context of *variational inference (VI)*, stacking seeks to construct a superior approximation to the true posterior distribution by averaging multiple variational approximations, each potentially capturing different aspects of the underlying posterior.

Step 1: Scoring Rules and Divergences

To assess the quality of each candidate model and their combinations, we employ *scoring rules.*

A *scoring rule* is a function

$$S : \mathcal{P} \times \Omega \to \overline{\mathbb{R}} = [-\infty, \infty],$$

where $\mathcal{P}$ is a class of probability distributions on a measurable space Ω, and for each $P \in \mathcal{P}$, the function $S(P, \cdot)$ is P-quasi-integrable. For continuous distributions, each $P \in \mathcal{P}$ can be represented by its density p.

The *expected score* of a predictive distribution P under a data-generating distribution Q is defined as:

$$S(P, Q) = \int S(P, \omega)\, dQ(\omega).$$

A scoring rule is said to be *proper* if $S(Q, Q) \geq S(P, Q)$ for all $P \in \mathcal{P}$, and *strictly proper* if equality holds only when $P = Q$ almost surely. Proper scoring rules induce a *divergence function*:

$$d(P, Q) = S(Q, Q) - S(P, Q),$$

which measures the discrepancy between distributions P and Q.

An important example is the *logarithmic scoring rule*, defined as $S(P, y) = \log p(y)$, where p is the density of P. This scoring rule induces the well-known *Kullback–Leibler (KL) divergence.*

Step 2: The Stacking Objective

Let $\hat{q}_1^{\mathrm{VI}}, \ldots, \hat{q}_K^{\mathrm{VI}}$ denote K distinct variational approximations to the true posterior $p(\theta \mid D)$, each obtained via a separate variational inference procedure. Stacking constructs a mixture distribution:

$$q_{\mathrm{mix}}(\theta) = \sum_{k=1}^{K} w_k\, \hat{q}_k^{\mathrm{VI}}(\theta),$$

where $w = (w_1, \ldots, w_K) \in \mathcal{S}_K$ lies in the $(K-1)$-simplex:

$$\mathcal{S}_K = \left\{ w \in \mathbb{R}^K : \sum_{k=1}^{K} w_k = 1,\ w_k \geq 0 \right\}.$$

The optimal weights w are those maximising the expected score under the true posterior:

$$\max_{w \in \mathcal{S}_K} S\left(\sum_{k=1}^{K} w_k\, \hat{q}_k^{\mathrm{VI}}(\theta),\, p(\theta \mid D) \right),$$

or, equivalently, minimising the associated divergence:

$$\min_{w \in \mathcal{S}_K} d\left(\sum_{k=1}^{K} w_k\, \hat{q}_k^{\mathrm{VI}}(\theta),\, p(\theta \mid D)\right).$$

When the logarithmic scoring rule is employed, this is equivalent to minimising the KL divergence between the stacked approximation and the true posterior.

Step 3: Leave-One-Out Approximation

Since the true posterior $p(\theta \mid D)$ is unknown, the direct computation of the above objectives is infeasible. Additionally, using the same dataset for both training and evaluation risks overfitting. To address these issues, *leave-one-out* (LOO) cross-validation is used to approximate the expected predictive performance.

Let $D = \{y_1, \ldots, y_n\}$ denote the dataset. For each y_i, we consider the LOO predictive distribution for model k as:

$$\hat{q}_{k,-i}(y_i) \approx \int p(y_i \mid \theta_k)\, \hat{q}_k^{\mathrm{VI}}(\theta_k \mid y_{-i})\, d\theta_k,$$

where y_{-i} denotes the dataset excluding the i-th observation.

The stacking weights are then optimised by solving:

$$\max_{w \in \mathcal{S}_K} \frac{1}{n} \sum_{i=1}^{n} S\left(\sum_{k=1}^{K} w_k\, \hat{q}_{k,-i}(y_i),\, y_i\right).$$

In the case of the logarithmic score, the objective becomes:

$$\max_{w \in \mathcal{S}_K} \frac{1}{n} \sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} w_k\, \hat{q}_{k,-i}(y_i)\right).$$

Step 4: Final Posterior Distribution

After obtaining the optimal weights $\hat{w}_1, \ldots, \hat{w}_K$, the final stacked posterior is given by:

$$\hat{q}^{\mathrm{RVI}}(\theta \mid D) = \sum_{k=1}^{K} \hat{w}_k\, \hat{q}_k^{\mathrm{VI}}(\theta \mid D).$$

This *stacked variational posterior* integrates information from all candidate approximations, often resulting in improved predictive performance and greater robustness.

Intuition

The fundamental insight behind stacking is to exploit the *diversity* among variational approximations. Each approximation may capture different aspects of the true posterior, such as skewness, multimodality, or tail behaviour. By combining these approximations, stacking can construct a more expressive and accurate posterior.

In addition to this, the use of *leave-one-out scoring* ensures that model weights are assigned based on out-of-sample predictive performance, thereby mitigating overfitting and enhancing generalisation.

Step 5: Pareto Smoothed Importance Sampling (PSIS)

A primary computational challenge in the stacking procedure is the evaluation of the leave-one-out (LOO) predictive densities:

$$\hat{q}_{k,-i}(y_i) \approx \int p(y_i \mid \theta_k)\, \hat{q}_k^{\mathrm{VI}}(\theta_k \mid y_{-i})\, d\theta_k,$$

which requires retraining each model n times, once for every data point y_i, to obtain the posterior $\hat{q}_k^{\mathrm{VI}}(\theta_k \mid y_{-i})$. This repeated refitting is computationally prohibitive, especially for large datasets or complex models.

To alleviate this burden, we employ an efficient approximation based on importance sampling. For the k-th model, we perform inference using the full dataset $y = \{y_1, \ldots, y_n\}$, yielding S posterior draws $\theta_k^s \sim \hat{q}_k^{\mathrm{VI}}(\theta_k \mid y)$ for $s = 1, \ldots, S$. We then compute the importance ratios:

$$r_{i,k}^s = \frac{1}{p(y_i \mid \theta_k^s)} \propto \frac{\hat{q}_k^{\mathrm{VI}}(\theta_k^s \mid y_{-i})}{\hat{q}_k^{\mathrm{VI}}(\theta_k^s \mid y)},$$

which approximate the reweighting needed to estimate the LOO predictive density without retraining the model on y_{-i}.

However, these raw importance ratios $r_{i,k}^s$ often exhibit high variance due to their heavy-tailed distribution. Following the method introduced in [50], we mitigate this instability by fitting a generalised Pareto distribution to the upper tail of $r_{i,k}^s$ and using the resulting smoothed weights:

$$w_{i,k}^s = \mathrm{ParetoSmooth}(r_{i,k}^s).$$

These smoothed importance weights $w_{i,k}^s$ yield a stabilised approximation to the LOO predictive density:

$$\hat{q}_{k,-i}(y_i) = \int p(y_i \mid \theta_k) \frac{\hat{q}_k^{\mathrm{VI}}(\theta_k \mid y_{-i})}{\hat{q}_k^{\mathrm{VI}}(\theta_k \mid y)}\, \hat{q}_k^{\mathrm{VI}}(\theta_k \mid y)\, d\theta_k \approx \frac{\sum_{s=1}^S w_{i,k}^s\, p(y_i \mid \theta_k^s)}{\sum_{s=1}^S w_{i,k}^s}.$$

This method is referred to as *Pareto Smoothed Importance Sampling* (PSIS).

The reliability of the PSIS approximation is diagnosed via the shape parameter $\hat{k}$ of the fitted Pareto distribution. Values of $\hat{k} < 0.7$ typically indicate a well-behaved approximation, whereas larger values suggest potential instability and the need for caution or alternative inference strategies.

Interested readers are referred to [50] for comprehensive derivations and diagnostic tools associated with PSIS.

Step 6: ELPD and Bayesian Bootstrapping

The *Expected Log Pointwise Predictive Density (ELPD)* is a widely used criterion for evaluating the out-of-sample predictive performance of a probabilistic model. In practical settings, one is often interested in assessing how well a model is likely to perform on unseen data. The ELPD provides a principled means to quantify this by estimating the expected log-likelihood of new observations under the model's predictive distribution, conditioned on the training data.

Formally, the ELPD is defined as:

$$\mathrm{ELPD} = \sum_{i=1}^{n} \mathbb{E}_{\tilde{y}_i \sim p_t} \left[\log p(\tilde{y}_i \mid y) \right],$$

where $\tilde{y}_i$ denotes a hypothetical future observation generated from the true data-generating distribution p_t, and $p(\tilde{y}_i \mid y)$ is the model's predictive density conditioned on the observed dataset y. This formulation rewards models that assign high probability to future data points, thereby capturing their generalisation ability.

Leave-One-Out Cross-Validation (LOO-CV) Since the true data-generating distribution p_t is typically unknown, we approximate the ELPD using Leave-One-Out Cross-Validation (LOO-CV). This technique estimates the predictive density for each data point by refitting the model on the dataset excluding that particular point. The LOO-CV approximation of the ELPD is given by:

$$\widehat{\mathrm{ELPD}}_{\mathrm{LOO}} = \sum_{i=1}^{n} \log p(y_i \mid y_{-i}),$$

where y_{-i} denotes the dataset with the i-th observation removed. This provides a near-unbiased estimate of the model's predictive performance and is especially suitable for Bayesian models where refitting is computationally tractable or can be efficiently approximated.

Model-Specific ELPD Estimation To assess the individual predictive quality of each model in an ensemble, we compute a model-specific ELPD using:

$$\mathrm{elpd}_k = \sum_{i=1}^{n} \int p_t(\tilde{y}_i) \log p_k(\tilde{y}_i \mid y) d\tilde{y}_i,$$

where $p_k(\cdot \mid y)$ is the posterior predictive distribution of model k. In practice, this is approximated via LOO-CV:

$$\mathrm{elpd}_k^{\mathrm{loo}} = \sum_{i=1}^{n} \log \left(\frac{1}{S} \sum_{s=1}^{S} p_k(y_i \mid \theta_k^s) \right),$$

where $\theta_k^s \sim p_k(\theta \mid y_{-i})$ are posterior draws from model k excluding y_i, and S is the number of samples.

Pseudo-Bayesian Model Averaging (Pseudo-BMA) To combine models in a principled way, one may use their LOO-CV-based ELPD estimates to form weights via a softmax transformation:

$$w_k = \frac{\exp(\mathrm{elpd}_k^{\mathrm{loo}})}{\sum_{j=1}^{K} \exp(\mathrm{elpd}_j^{\mathrm{loo}})}.$$

This weighting scheme emphasises models with higher predictive performance. However, it does not account for uncertainty in the elpd estimates themselves, leading to potentially overconfident weights, especially when differences in model performance are small.

Incorporating Uncertainty via Standard Errors To mitigate the issue of overconfidence, we incorporate the standard error of the LOO estimate:

$$\mathrm{se}(\mathrm{elpd}_k^{\mathrm{loo}}) = \sqrt{\sum_{i=1}^{n} \left(\mathrm{elpd}_k^{\mathrm{loo},i} - \frac{1}{n} \mathrm{elpd}_k^{\mathrm{loo}} \right)^2}.$$

A log-normal approximation of the uncertainty-adjusted predictive density then yields the modified weights:

$$w_k = \frac{\exp\left(\text{elpd}_k^{\text{loo}} - \frac{1}{2}\text{se}(\text{elpd}_k^{\text{loo}})^2\right)}{\sum_{j=1}^{K} \exp\left(\text{elpd}_j^{\text{loo}} - \frac{1}{2}\text{se}(\text{elpd}_j^{\text{loo}})^2\right)}.$$

These weights, known as *Pseudo-BMA+*, balance predictive accuracy with robustness to estimation variability, yielding a more stable ensemble.

Bayesian Bootstrap for Weight Uncertainty While Pseudo-BMA+ accounts for variability in a fixed dataset, it does not fully characterise the posterior uncertainty over the model weights. To address this, we employ the *Bayesian Bootstrap* ([49], [42]). The idea is to generate multiple weighted versions of the dataset by drawing from a Dirichlet distribution:

$$\alpha_{1:n} \sim \text{Dirichlet}((1, \ldots, 1)^\top),$$

and compute a weighted ELPD for each model using:

$$\hat{\phi}_k^{(b)} = \sum_{i=1}^{n} \alpha_i^{(b)} \cdot \text{elpd}_k^{\text{loo},i},$$

where $\alpha_i^{(b)}$ are the Dirichlet weights for the b-th bootstrap sample. Each $\hat{\phi}_k^{(b)}$ induces a set of weights $w_k^{(b)}$ as in the softmax or log-normal formulation above.

By repeating this process over B bootstrap replicates, we obtain a posterior distribution over model weights. The final stacking weights are then computed by averaging:

$$w_k = \frac{1}{B} \sum_{b=1}^{B} w_k^{(b)}.$$

Application in Robust Variational Inference (RVI) In the context of our proposed *Robust Variational Inference (RVI)* framework, we adopt the Bayesian Bootstrap-adjusted stacking procedure to robustly combine multiple probabilistic models. This approach ensures that ensemble weights reflect both predictive performance and associated uncertainty. As demonstrated in sections 6.1 and 6.2, this technique substantially improves performance in univariate distribution approximation and also leads to accurate and stable ensembles in binary classification via Bayesian logistic regression.

5.2.2 Sequential Least Squares Quadratic Programming (SLSQP)

In order to determine the optimal weights for our mixture model in Variational Inference (VI), we formulate the problem as a nonlinear programming (NLP) problem. One powerful algorithm for solving such problems is Sequential Least Squares Quadratic Programming (SLSQP) [31]. This method belongs to the broader class of Sequential Quadratic Programming (SQP) methods, which iteratively solve simpler subproblems, typically quadratic approximations, to make progress on the original nonlinear objective.

Unlike traditional SQP, which approximates the original NLP by solving a quadratic programming (QP) subproblem at each step, SLSQP solves a least-squares (LSQ) problem instead. This adjustment often makes the algorithm more numerically stable and applicable to a broader class of problems, particularly when the objective function is non-convex. This is especially relevant for our case, as the Evidence Lower Bound (ELBO), our objective function, is non-convex [3].

General NLP Formulation

We begin with the standard form of an NLP problem:

$$\min_{x \in \mathbb{R}^n} f(x)$$

subject to

$$h(x) = 0, \quad g(x) \geq 0,$$

where:

- $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function (in our case, the negative ELBO).
- $h : \mathbb{R}^n \to \mathbb{R}^{m_E}$ encodes equality constraints.
- $g : \mathbb{R}^n \to \mathbb{R}^{m_I}$ encodes inequality constraints.
- All functions are assumed to be at least twice continuously differentiable.

The gradient notation is defined as:

$$\nabla h := [\nabla h_1, \ldots, \nabla h_{m_E}], \quad \nabla g := [\nabla g_1, \ldots, \nabla g_{m_I}]$$

Least Squares Subproblem

At each iteration, SLSQP solves a constrained least-squares problem to generate the search direction d:

$$\min_{d \in \mathbb{R}^n} \frac{1}{2} \|R^k d - q^k\|^2,$$

subject to:

$$\nabla h(x^k)^T d + h(x^k) = 0, \quad \nabla g(x^k)^T d + g(x^k) \geq 0 \tag{LSQ}$$

Here:

- R^k is an upper triangular matrix obtained from a Cholesky-like factorisation.
- q^k is derived from the gradient of the objective.

Specifically:

$$(R^k)^T R^k = B^k, \tag{5.1}$$

$$(R^k)^T q^k = -\nabla f(x^k), \tag{5.2}$$

where B^k approximates the Hessian of the Lagrangian. Using LDL$^\mathrm{T}$ factorisation:

$$R^k = (D^k)^{1/2} (L^k)^T \tag{5.3}$$

and the Hessian is updated via the following low-rank formula:

$$L^{k+1} D^{k+1} (L^{k+1})^T = L^k D^k (L^k)^T + \frac{r^k (r^k)^T}{(r^k)^T s^k} - \frac{B^k s^k (s^k)^T B^k}{(s^k)^T B^k s^k}. \tag{5.4}$$

Penalty Terms and Merit Function

To ensure constraint satisfaction while optimising the objective, penalty parameters are introduced for both equality and inequality constraints. These parameters guide the algorithm toward feasible regions of the search space.

The penalty parameters are updated at each iteration k using:

$$\rho^k = \max\left(|\lambda^k|, \frac{\rho^{k-1} + |\lambda^k|}{2}\right), \tag{5.5}$$

$$\nu^k = \max\left(|\mu^k|, \frac{\nu^{k-1} + |\mu^k|}{2}\right), \tag{5.6}$$

where λ^k, μ^k are the Lagrange multipliers of the LSQ problem. A combined **merit function**, which evaluates both the objective and the degree of constraint violation, is defined as:

$$\varphi(x; \rho^k, \nu^k) = f(x) + \sum_{j \in E} \rho^k |h_j(x)| + \sum_{j \in I} \nu^k g_j(x)^-, \tag{5.7}$$

where $g_j(x)^- := \max(0, -g_j(x))$ captures violations of the inequality constraints.

The directional derivative of this merit function is:

$$D\varphi(x^k, d; \rho^k, \nu^k) = \nabla f(x^k)^T d - \sum_{j \in E} \rho_k |h_j(x^k)| - \sum_{j \in I} \nu^k g_j(x)^-, \tag{5.8}$$

This quantity is used in line search to ensure progress toward both feasibility and optimality. The **Armijo condition** ensures sufficient descent at each step:

$$\varphi(x^k + \alpha d) - \varphi(x^k) < \alpha \cdot \eta \cdot D\varphi(x^k, d), \tag{5.9}$$

where $\eta \in (0, 0.5)$ is a small constant ensuring conservative steps.

Convergence Criteria

The algorithm uses two groups of convergence criteria:

Group 1: After solving LSQ subproblem

$$\text{acc}_{inf} = \sum_{j \in E} \|h_j(x^k)\| + \sum_{j \in I} g_j(x^k)^- < \text{tol}, \tag{5.10}$$

$$\text{acc}_{opt} = \|\nabla f(x^k)^T d\| + \|\lambda_k\|^T \|h(x^k)\| + \|\mu_k\|^T g(x^k)^- < \text{tol}, \tag{5.11}$$

$$\text{acc}_{step} = \|d\| < \text{tol}. \tag{5.12}$$

Group 2: After line search

$$\widetilde{\text{acc}}_{inf} = \sum_{j \in E} \|h_j(x^k + \alpha d)\| + \sum_{j \in I} g_j(x^k + \alpha d)^- < \widetilde{\text{tol}}, \tag{5.13}$$

$$\widetilde{\text{acc}}_{opt} = |f(x^k + \alpha d) - f(x^k)| < \widetilde{\text{tol}}, \tag{5.14}$$

$$\widetilde{\text{acc}}_{step} = \|d\| < \widetilde{\text{tol}}. \tag{5.15}$$

Here, acc_{inf}, acc_{opt} and acc_{step} represent the feasibility, optimality, and step length, respectively. acc_{inf} is the summation of infeasibilities in all the constraints. acc_{opt} indicates the decrease potential of the objective function and the weighted constraint infeasibility. acc_{step} is the 2-norm of the descent direction. The optimisation is deemed successful if either: (5.12 and 5.13) or (5.12 and 5.14) are satisfied.

Framing Weight Selection as an NLP Problem

Recall that in our RVI (Robust Variational Inference) framework, we aim to find an optimal convex combination of candidate approximate posteriors $\hat{q}_i^{VI}(\theta)$. The ELBO associated with the mixture $\hat{q}^{RVI}(\theta) = \sum_i w_i \hat{q}_i^{VI}(\theta)$ is a function of the weights $\{w_i\}$, which must be non-negative and sum to 1.

We therefore formulate the following constrained optimisation problem:

$$\max_{w \in \mathbb{R}^K} \bar{L}(w) := \text{ELBO}(w) \quad \text{subject to} \quad \sum_{i=1}^{K} w_i = 1, \quad w_i \geq 0 \, \forall i,$$

This is a nonlinear programming (NLP) problem due to the non-convex nature of the ELBO with respect to the weights. SLSQP is thus well-suited for this setting: it handles nonlinear objectives, incorporates both equality and inequality constraints, and does not require convexity. This allows us to reliably search for the optimal set of weights that form the most expressive and robust approximate posterior from among our candidates. In our experiments with Variational Autoencoders (VAEs), we employ the SLSQP algorithm to optimise the mixture weights associated with the approximate posterior distributions produced by the encoder network.

Below is a summary of the SLSQP algorithm:

```
Algorithm 1: A basic SLSQP algorithm
  Data: $x^0$, $B^0$, $\rho^0$, $\nu^0$ and evaluate $f(x^0)$, $g(x^0)$, $h(x^0)$, $\nabla f(x^0)$, $\nabla g(x^0)$, $\nabla h(x^0)$
  Result: $x^k$, $\lambda^k$, $\mu^k$, $f(x^k)$
  Initialize: $k \leftarrow 0$, given $x^0$, $B^0$, $\rho^0$, $\nu^0$ and evaluate $f(x^0)$, $g(x^0)$, $h(x^0)$, $\nabla f(x^0)$, $\nabla g(x^0)$, $\nabla h(x^0)$;
  Solve the Least Squares Problem to obtain the search direction $d$ and Lagrange multipliers of LSQ: $\lambda^k$, $\mu^k$;
  Check the convergence criteria for the NLP problem. If they are satisfied, return to the output, else proceed;
  Update the penalty parameters $\rho^k$, $\nu^k$ using Eqs. (5.5), (5.6) above and calculate the directional derivative $D\varphi(x^k, d; \rho^k, \nu^k)$ using Eq. (5.8);
  Conduct the line search with the merit function defined in Eq. (5.7) to get a step length $\alpha$ that satisfies the Armijo condition Eq. (5.9), set $x^{k+1} \leftarrow x^k + \alpha d$, evaluate $f(x^{k+1})$, $g(x^{k+1})$, $h(x^{k+1})$;
  Update $L^k$ and $D^k$ by the formula in Eq. (5.3), then set $k \leftarrow k + 1$ and proceed to the next iteration;
  Return $x^k$, $\lambda^k$, $\mu^k$, $f(x^k)$;
```

Chapter 6

Experiments and Results

We evaluate the performance of our algorithm on both toy and real-world examples, considering both univariate and multivariate settings. To assess robustness, we introduced noise and outliers into each dataset.

In the univariate setting, our method demonstrates clear improvements over classical baselines, particularly in capturing complex posterior structures and handling discontinuities. However, in the multivariate setting, where we test our method on tasks such as binary classification and image reconstruction using the MNIST dataset [27], we do not observe a significant improvement in performance. Although there are slight gains in reconstruction quality, they are modest, and in classification tasks like Bayesian Logistic Regression, our approach performs comparably to standard methods. These results suggest that the benefits of our variational inference technique may be limited in higher-dimensional settings and are sensitive to both the nature of the task and the chosen distributional families.

All experiments were conducted on a standard laptop with 16.0 GB RAM and an 11th Gen Intel(R) Core(TM) i5-1155G7 @ 2.50GHz processor. Our code is available at https://github.com/bublaiSAURUS/RVI-via-IP.

6.1 Univariate Setting

Set up

We set our target distribution to $\mathcal{N}(x; 4, 1)$. To stress-test our model, we artificially injected Gaussian noise drawn from $\mathcal{N}(x; 0, \sigma^2)$, where $\sigma = 1.5$ represents a relatively high noise level. For comparison, we used Automatic Differentiation Variational Inference (ADVI) (see section 3.3), a widely adopted method for variational inference in Python, Stan, and other probabilistic programming languages.

In our Robust Variational Inference (RVI) framework, we employed three families of approximate posterior distributions: Student t, Gaussian, and Skew-Normal. Student t distribution was selected for its heavy tails and robustness to outliers, the Gaussian distribution for its ability to model the bulk of the data effectively, and the Skew-Normal distribution to account for potential skewness in the data.

Figure 6.1 demonstrates that under high noise conditions, both ADVI and RVI perform comparably.

Keeping the noise level fixed at $\sigma = 1.5$, we then introduced outliers into the dataset, varying the outlier fraction as $0.05, 0.10, 0.15, 0.20$. Figures 6.2a–6.2d show that as the percentage of outliers increases, RVI continues to accurately capture the central structure of the dataset and remains robust to the presence of outliers. In contrast, ADVI begins to

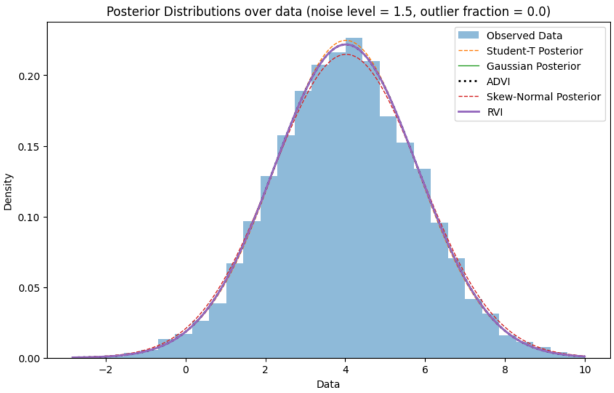
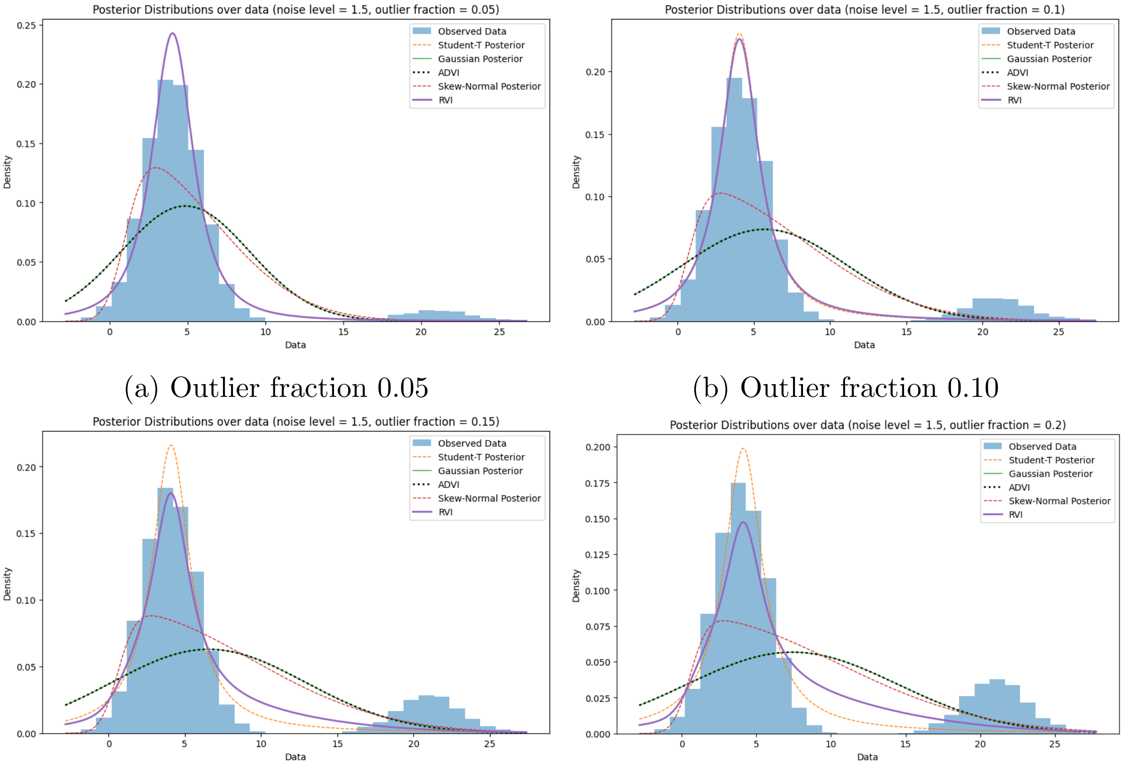


Figure 6.1: Comparison of RVI and ADVI with no outliers.

under-represent the bulk of the data on the right and gradually shifts its estimated mean toward the centre of the two modes.



(a) Outlier fraction 0.05

(b) Outlier fraction 0.10

(c) Outlier fraction 0.15

(d) Outlier fraction 0.20

Figure 6.2: Comparison of RVI and ADVI with varying outliers.

Analysis of results

This behaviour can be explained by the nature of the approximating distributions. ADVI fits a Gaussian distribution to approximate the posterior, which is inherently sensitive to extreme values due to its light tails. Since it cannot assign significant probability mass to outliers, it is forced to compromise by shifting its mean closer to the centre of the outlier cluster and the main data mass.

In contrast, the Student t-distribution used in RVI has heavy tails, allowing it to assign reasonable probability to outliers without significantly altering the representation of the main data cluster. Because it models the majority of the data more faithfully, stacking assigns it the highest weight in the mixture, thus dominating the final RVI approximation. This explains why RVI remains stable and continues to model the main distribution accurately even as the number of outliers increases, while ADVI becomes increasingly biased toward the middle.

6.2 Multivariate Setting

To assess the performance of our model in higher dimensions, we consider Bayesian Logistic Regression for binary classification and Variational Autoencoders for image reconstruction. As before, we introduced noise and outliers in each dataset and compared the performance of RVI against the standard approaches.

6.2.1 Binary Classification

Following [40], we use the 5 datasets given below for our experiments with Bayesian Logistic Regression for binary classification.

Set up

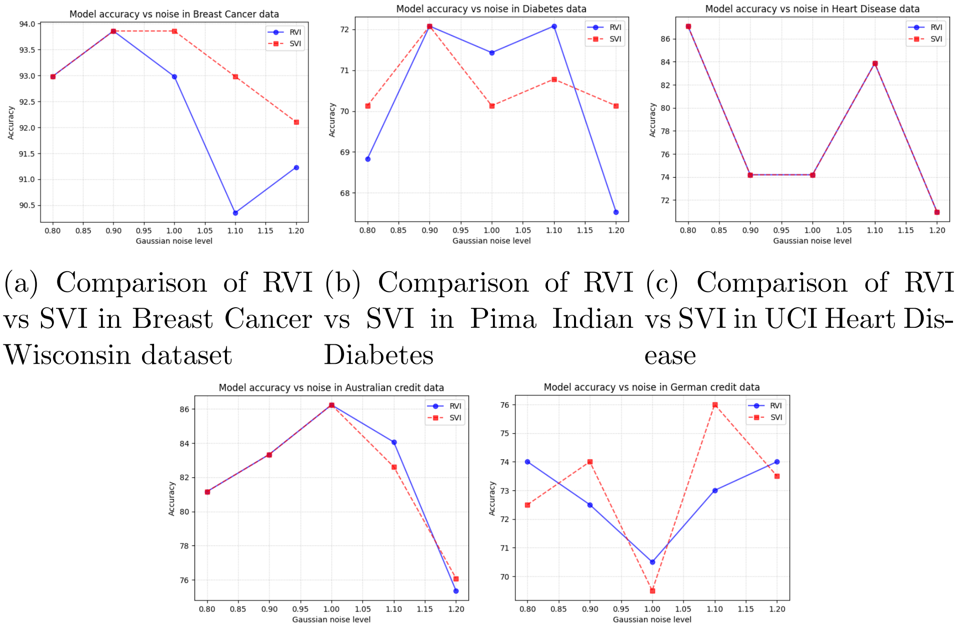
Name of dataset	Number of samples	Number of features
Breast Cancer Wisconsin (Diagnostic)	569	30
Pima Indian Diabetes	768	9
UCI Heart Disease	920	16
Australian Credit Approval	690	14
German Credit Data	1000	20

Table 6.1: Benchmark datasets for Bayesian Logistic Regression experiments.

We introduced noise into the training data by adding Gaussian noise sampled from $\mathcal{N}(0, \sigma^2)$ and injected outliers by randomly selecting a fixed percentage of the data points and scaling their values by a constant factor of 5. Given that our datasets contain between 500 and 1000 samples ($500 \leq N \leq 1000$), we fixed the outlier proportion at 5%. To assess model robustness under increasing noise, we varied the noise level σ across the values $\{0.8, 0.9, 1.0, 1.1, 1.2\}$. In contrast to the training data, the test dataset was kept clean.

For all five datasets, we employed Robust Variational Inference (RVI), which utilises both multivariate Gaussian and multivariate Student-t distributions to approximate the posterior over the weight parameters of a Bayesian Logistic Regression model. The corresponding bias terms were modelled using univariate Gaussian and univariate Student-t distributions, respectively. Rather than fixing the degrees of freedom (ν) of the Student-t distribution a priori, we treated it as a learnable variational parameter. To enable this, we placed a Gamma prior over ν and inferred its value directly from the data during training. During training, the negative ELBO was minimised (which is equivalent to maximising the ELBO) using the Adam optimiser. The learning rate was set at $\eta = 0.01$ for all experiments. The models were trained for 5000 epochs.

For baseline comparison, we implemented a standard Stochastic Variational Inference (SVI) method using only a multivariate Gaussian family to approximate the posterior. The results of these models are summarised in Figure 6.3. The accuracies reported were calculated after taking 4000 samples of logistic regression parameter distributions obtained via RVI (or SVI) and averaging the predictions.



(a) Comparison of RVI vs SVI in Breast Cancer Wisconsin dataset

(b) Comparison of RVI vs SVI in Pima Indian Diabetes

(c) Comparison of RVI vs SVI in UCI Heart Disease

(d) Comparison of RVI vs SVI in Australian Credit Approval dataset

(e) Comparison of RVI vs SVI in German Credit dataset

Figure 6.3: Results of Bayesian logistic regression across 5 different benchmark datasets.

Analysis of results

We observe that Robust Variational Inference (RVI) does not consistently outperform and occasionally underperforms standard Stochastic Variational Inference (SVI) using a simple Gaussian posterior in Bayesian Logistic Regression, particularly on datasets averaging around 800 samples. This may seem counter-intuitive, as heavy-tailed distributions such as the Student-t are widely used to enhance robustness to outliers and model misspecification ([11]).

In our RVI framework, we approximate the posterior over model weights using a mixture of a Gaussian and a Student-t distribution. Importantly, the degrees of freedom (ν) of the Student-t are learnt from data and typically fall between 4 and 8. These values yield moderately heavy tails and, in theory, should offer a balance between robustness and tractability ([26], [11]).

Surprisingly, we found that during model combination via methods like Stacking or BB-pseudo-BMA, the Student-t component often received disproportionately high weights. This skewed ensemble weighting frequently degraded the overall predictive performance compared to a simpler Gaussian-only posterior.

A likely explanation lies in the interplay between flexibility and generalisation. The heavier tails of the Student-t allow it to assign non-negligible density to outliers, enhancing robustness, but this same flexibility may lead to overfitting in small datasets. The Student-t posterior may become overly diffuse, increasing predictive variance and reducing calibration. In contrast, the Gaussian posterior,though less flexible, tends to be more stable and regularized in low-data regimes.

Moreover, model averaging schemes like Stacking and BB-pseudo-BMA rely on validation-based metrics such as leave-one-out cross-validation ([55]), which tend to reward sharper predictions. The Student-t model, due to its higher expressiveness, may fit the training data more tightly and appear more accurate during validation, even if it overfits. This can cause it to dominate the ensemble weights, reducing the regularizing influence of the Gaussian component.

This issue is compounded by the fact that the Gaussian and Student-t posteriors are trained independently and only combined post hoc through ensemble techniques. Without joint optimisation, the two components cannot influence each other's learning dynamics. This makes it more difficult to achieve a balanced representation. Consequently, when the more flexible Student-t model receives higher weighting, it can disproportionately influence the final predictions, sometimes to the model's detriment.

In summary, the occasional underperformance of RVI in Bayesian Logistic Regression with small datasets stems from a combination of overfitting risk in heavy-tailed models, validation-driven ensemble weighting that favors overconfident predictions, and the lack of joint training between the mixture components ([56], [11]).

6.2.2 Image Reconstruction

In this section, we compare the performance of the RVI-based Variational Autoencoder (RVI-VAE) with a traditional VAE that employs a standard Amortized Variational Inference (A-VI) network. The A-VI network is trained to predict the parameters of a Gaussian posterior approximation, specifically the mean μ_G and standard deviation σ_G. In contrast, the RVI-VAE's inference network is designed to learn three additional parameters: the degrees of freedom (ν), the mean (μ_T), and the scale (σ_T) of a Student-t distribution. The overall loss function for RVI-VAE is constructed as the average of the losses computed from both the Gaussian and Student-t inference networks, encouraging a shared representation that leverages the strengths of both distributions.

Set up

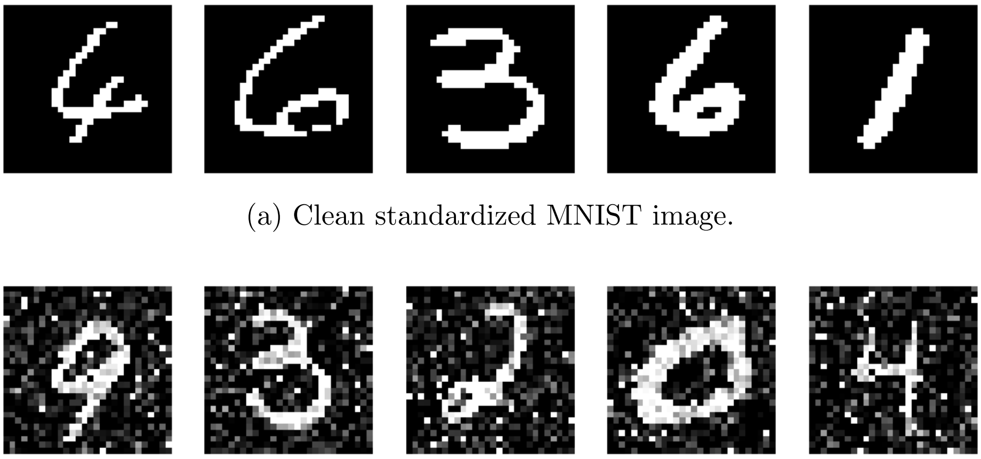
For both the encoder and decoder networks, we use the ReLU activation function defined as $\text{ReLU}(x) = \max(0, x)$. Additional architectural and training details for the RVI-VAE are summarised in the table below. The latent space dimension ($\text{latent}_{\text{dim}}$) was fixed at 2 for all experiments.

Layer	**# Layers**	**Neuron Units, Filters**	**Kernel Size**
Convolutional (Encoder)	2	32, 64 filters	3×3
Dense (Encoder Output)	1	$2\times$ $\text{latent}_{\text{dim}} + 3\times$ $\text{latent}_{\text{dim}}$	–
Dense (Decoder Input)	1	$7 \times 7 \times 32$	–
Transposed Convolution (Decoder)	3	64, 32, 1 filters	3×3

Table 6.2: Key network features of the RVI-VAE architecture

As in typical Variational Inference, we maximise the Evidence Lower Bound (ELBO); equivalently, we minimise the negative ELBO to make the objective compatible with standard gradient-based optimisers such as Adam and SGD. In our experiments, we used the Adam optimiser with a learning rate $\eta = 10^{-4}$, a batch size of 32, and trained the models for 10 epochs.

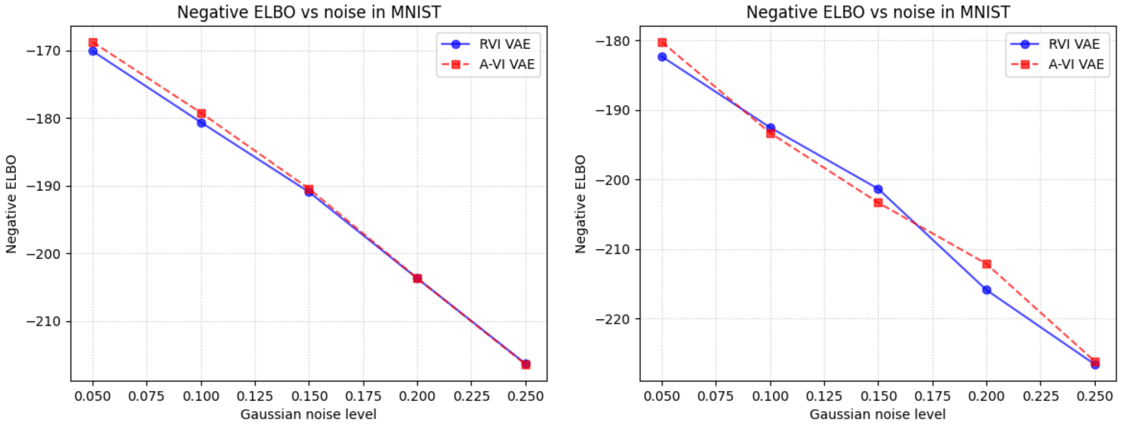
Given that MNIST is substantially larger than the datasets used in our Bayesian Logistic Regression experiments (N = 60,000), we additionally varied the fraction of outliers in the training data, setting it to either 0.01 or 0.05. For each selected outlier fraction, we also introduced varying levels of Gaussian noise into the training images, with $\sigma \in 0.05, 0.10, 0.15, 0.20, 0.25$. We observed that increasing the noise level beyond 0.30 leads to severe distortion of digit images; hence, we capped the maximum noise level at 0.25. The test dataset, in all cases, was kept clean i.e. free of noise and outliers, to provide a fair evaluation of reconstruction quality. The final test negative ELBO was reported after running and averaging the model 10 times on training dataset.



(a) Clean standardized MNIST image.

(b) Noisy MNIST image with 0.25 gaussian noise level.

Figure 6.4: MNIST image quality before and after adding level 0.25 gaussian noise.



(a) Outlier fraction 0.01

(b) Outlier fraction 0.05

Figure 6.5: Comparison of VAE and RVI-VAE on MNIST. Final negative ELBO loss plotted against varying Gaussian noise levels. Two scenarios are presented—one with an outlier fraction of 0.01 and another with an outlier fraction of 0.05.

Analysis of results

We observe that the RVI-based Variational Autoencoder (VAE) achieves comparable or often superior reconstruction performance in the MNIST dataset relative to the standard

Gaussian-VAE, as measured by the lower negative ELBO. This supports the theoretical motivation for incorporating heavier-tailed distributions, such as the Student-t, in the variational family to better approximate complex posterior geometries ([24]) along with the results we presented in chapter 5 (section 5.1).

Unlike standard VAEs that rely solely on a multivariate Gaussian based posterior inference network, our RVI-VAE uses a mixture of a multivariate Gaussian and a multivariate Student-t based inference network. The Student-t component introduces heavier tails, enabling the posterior to adapt more flexibly to non-Gaussian structure in the latent space. This is particularly beneficial in image datasets like MNIST, where even subtle variations in digit shapes and strokes can introduce multimodal or skewed posteriors ([3]).

Importantly, the large size of MNIST (60,000 training examples) acts as an implicit regularizer, reducing the risk of overfitting. This was a concern that often arose with more expressive posterior families in the smaller datasets for Bayesian Logistic Regression. As a result, the RVI-VAE can exploit the added flexibility without sacrificing generalisation. The ELBO, which balances reconstruction accuracy and KL divergence, naturally favours such expressive posteriors when they yield tighter approximations to the true posterior while maintaining compactness.

In the instances of underperformance, it is likely due to optimisation challenges introduced by the increased complexity of the variational family, such as increase in the number of variational parameters to learn, sensitivity to initialisation, or local minima during early training.

In summary, the RVI-VAE's performance gains stem from its ability to capture richer posterior structure through the Gaussian–Student-t mixture, which proves especially effective in large-sample settings like MNIST.

Chapter 7

Conclusion and Future Work

7.1 Overview of achievements

In this work, we introduced a novel approach to variational inference by leveraging the theoretical foundations of Imprecise Probability theory (sections 4.1, 4.2). Our proposed method, Robust Variational Inference (RVI), constructs a mixture of variational families to capture distributional ambiguity and improve robustness.

We provided theoretical justifications for RVI and demonstrated how the weights of each component distribution in the mixture can be learnt using optimisation techniques such as Sequential Least Squares Programming (SLSQP) and Stacking. In the univariate setting, we showed that RVI performs significantly better in the presence of heavy outliers, outperforming established techniques like Automatic Differentiation Variational Inference (ADVI).

We further evaluated RVI on image reconstruction tasks using a noisy MNIST dataset, where it achieved slightly better performance compared to standard Variational Autoencoders (VAEs) trained with Amortized Variational Inference (A-VI). We also applied RVI to Bayesian Logistic Regression for binary classification and observed competitive results compared to Stochastic Variational Inference (SVI), especially in terms of calibration and robustness.

To the best of our knowledge, this is the first work to propose a Credal Variational Inference framework, bridging classical precise probabilistic machine learning with the principles of Imprecise Probability Theory.

7.2 Critical reflection

The core motivation behind Randomised Variational Inference (RVI) lies in its ability to capture ambiguity by leveraging multiple distributional families, a departure from the single distributional assumption in traditional Variational Inference (VI). Our theoretical analysis, supported by empirical evidence in certain scenarios, indicates that this approach can yield superior results compared to standard VI. However, this improvement comes at the cost of a substantial increase in computational time. Future research could fruitfully explore the development of efficient training strategies aimed at mitigating this computational overhead.

Our evaluation aimed to showcase the full potential of RVI by benchmarking it against established variational inference methods across three diverse tasks, each featuring distinct datasets and varying levels of noise. While our experiments included MNIST, the relatively small size of the other datasets presented a challenge. We observed that increasing the number of variational parameters led to overfitting, particularly with the Student t component inside RVI. To address this, future work could also investigate the design of regularization techniques specifically tailored to RVI, potentially enhancing its generalisation performance on smaller datasets. Ultimately, to gain a more robust understanding of RVI's true performance potential, replicating these experiments on significantly larger datasets (with sample sizes of $N \geq 100,000$) would provide a more reliable and generalisable assessment.

7.3 Open problems

There are many open avenues for research in Imprecise Probability based variational inference. We discuss two of the most critical problems.

Both RVI and standard Variational Inference (VI) struggle to accurately model data with multiple modes, as shown in Figure 7.1. When the true distribution has distinct modes, like the mixture of two Gaussians $\mathcal{N}(x; 4, 1)$ and $\mathcal{N}(x; 9, 1)$ presented, this limitation arises for two main reasons. Variational Inference (VI) often struggles with multimodal posteriors due to the asymmetric nature of KL divergence, which heavily penalises assigning mass to low-probability regions while being lenient toward missing high-probability ones. As a result, unimodal approximations tend to average over modes rather than capture them distinctly. In our current RVI framework, this issue is compounded by the use of unimodal base families (e.g., Gaussian, Student t) optimised independently and combined only post hoc, leading the final mixture to centre around shared regions instead of resolving separate modes.

A potential solution to this issue is to employ a multi-start optimisation strategy. This involves running the ELBO optimisation multiple times, each starting from different random initial values for the variational parameters. By exploring a wider range of possibilities, this approach can yield a set of different RVI posterior approximations. The final RVI posterior could then be formed by combining these individual RVI posteriors (e.g., as a mixture), which may allow the resulting distribution to better represent the multiple modes of the true posterior.

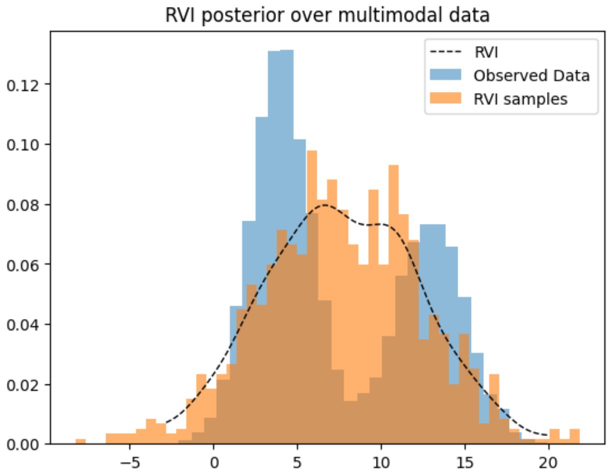


Figure 7.1: RVI fails to capture both modes in multimodal data.

In our current formulation of RVI, we use optimisation techniques (SLSQP and stacking) to determine the weights of the component distributions in the mixture. An alternative approach is to draw from the theory of imprecise probabilities by defining *lower* and *upper* probabilities. Specifically, we can define the lower probability as $\underline{q}^{VI}(\theta) = \min_j \hat{q}_j^{VI}(\theta)$ and the upper probability as $\overline{q}^{VI}(\theta) = \max_j \hat{q}_j^{VI}(\theta)$, where each $\hat{q}_j^{VI}(\theta)$ is a posterior approximation from a different variational family. We can then study the range $[\underline{q}^{VI}(\theta), \overline{q}^{VI}(\theta)]$ and in which conditions $p^{\text{true}}(\theta|D)$ will lie almost surely inside this range.

Acknowledgements

The successful completion of this project owes much to the invaluable guidance and unwavering support of my supervisor Professor Michele Caprio, for which I am deeply grateful. I also wish to acknowledge the love and encouragement of my family and friends. Finally, I offer my heartfelt thanks to God for granting me the strength and perseverance to bring this research to fruition.

References

[1] Daniel Andrade. Stable training of normalizing flows for high-dimensional variational inference. *arXiv preprint arXiv:2402.16408*, 2024.

[2] Thomas Augustin, Frank PA Coolen, Gert De Cooman, and Matthias CM Troffaes. *Introduction to imprecise probabilities*. John Wiley & Sons, 2014.

[3] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[5] Michael Braun and Jon McAuliffe. Variational inference for large-scale models of discrete choice. *Journal of the American Statistical Association*, 105(489):324–335, 2010.

[6] Michele Caprio, Souradeep Dutta, Kuk Jin Jang, Vivian Lin, Radoslav Ivanov, Oleg Sokolsky, and Insup Lee. Credal bayesian deep learning. *arXiv preprint arXiv:2302.09656*, 2023.

[7] Peter Carbonetto and Matthew Stephens. Scalable variational inference for bayesian variable selection in regression, and its accuracy in genetic association studies. 2012.

[8] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[9] Jean Daunizeau, Vincent Adam, and Lionel Rigoux. Vba: a probabilistic treatment of nonlinear models for neurobiological and behavioural data. *PLoS computational biology*, 10(1):e1003441, 2014.

[10] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[11] S Edition. Bayesian data analysis, 2013.

[12] Elena A Erosheva, Stephen E Fienberg, and Cyrille Joutard. Describing disability through individual-level mixture models for multivariate binary data. *The annals of applied statistics*, 1(2):346, 2007.

[13] Thomas Furmston and David Barber. Variational methods for reinforcement learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 241–248. JMLR Workshop and Conference Proceedings, 2010.

[14] Samuel Gershman, Matt Hoffman, and David Blei. Nonparametric variational inference. *arXiv preprint arXiv:1206.4665*, 2012.

[15] Walter R Gilks, Sylvia Richardson, and David Spiegelhalter. *Markov chain Monte Carlo in practice*. CRC press, 1995.

[16] Justin Grimmer. An introduction to bayesian inference via variational approximations. *Political Analysis*, 19(1):32–47, 2011.

[17] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

[18] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *the Journal of machine Learning research*, 14(1):1303–1347, 2013.

[19] Ferenc Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

[20] Tommi S Jaakkola and Michael I Jordan. Improving the mean field approximation via the use of mixture distributions. In *Learning in graphical models*, pages 163–173. Springer, 1998.

[21] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. In *Learning in graphical models*, pages 105–161. Springer, 1998.

[22] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

[23] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[24] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.

[25] Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *Journal of machine learning research*, 18(14):1–45, 2017.

[26] Kenneth L Lange, Roderick JA Little, and Jeremy MG Taylor. Robust statistical modeling using the t distribution. *Journal of the American Statistical Association*, 84(408):881–896, 1989.

[27] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[28] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in neural information processing systems*, 29, 2016.

[29] Benjamin A Logsdon, Gabriel E Hoffman, and Jason G Mezey. A variational bayes algorithm for fast and accurate multiple locus genome-wide association analysis. *BMC bioinformatics*, 11:1–13, 2010.

[30] Pengyuan Lu, Michele Caprio, Eric Eaton, and Insup Lee. Ibcl: Zero-shot model generation for task trade-offs in continual learning. 2024.

[31] Yingjie Ma, Xi Gao, Chao Liu, and Jie Li. Improved sqp and slsqp algorithms for feasible path-based process optimisation. *Computers & Chemical Engineering*, 188:108751, 2024.

[32] Charles C Margossian and David M Blei. Amortized variational inference: when and why? *arXiv preprint arXiv:2307.11018*, 2023.

[33] Kevin P Murphy. *Probabilistic machine learning: Advanced topics.* MIT press, 2023.

[34] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[35] Anil Raj, Matthew Stephens, and Jonathan K Pritchard. faststructure: variational inference of population structure in large snp data sets. *Genetics*, 197(2):573–589, 2014.

[36] Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial intelligence and statistics*, pages 814–822. PMLR, 2014.

[37] Jeffrey Regier, Andrew Miller, Jon McAuliffe, Ryan Adams, Matt Hoffman, Dustin Lang, David Schlegel, and Mr Prabhat. Celeste: Variational inference for a generative model of astronomical images. In *International Conference on Machine Learning*, pages 2095–2103. PMLR, 2015.

[38] Manuel J Reyes-Gomez, Daniel PW Ellis, and Nebojsa Jojic. Multiband audio modeling for single-channel acoustic source separation. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages V–641. IEEE, 2004.

[39] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[40] Joseph D Romano, Trang T Le, William La Cava, John T Gregg, Daniel J Goldberg, Praneel Chakraborty, Natasha L Ray, Daniel Himmelstein, Weixuan Fu, and Jason H Moore. Pmlb v1. 0: an open-source dataset collection for benchmarking machine learning methods. *Bioinformatics*, 38(3):878–880, 2022.

[41] Arianna W Rosenbluth, Augusta H Teller, E Teller, N Metropolis, and A Rosenbluth. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[42] Donald B Rubin. The bayesian bootstrap. *The annals of statistics*, pages 130–134, 1981.

[43] Tim Salimans, Diederik Kingma, and Max Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *International conference on machine learning*, pages 1218–1226. PMLR, 2015.

[44] Joshua S Speagle. A conceptual introduction to markov chain monte carlo methods. *arXiv preprint arXiv:1909.12313*, 2019.

[45] Oliver Stegle, Leopold Parts, Richard Durbin, and John Winn. A bayesian framework to account for complex non-genetic factors in gene expression levels greatly increases power in eqtl studies. *PLoS computational biology*, 6(5):e1000770, 2010.

[46] Peter Sykacek, Stephen J Roberts, and Maria Stokes. Adaptive bci based on variational bayesian kalman filtering: an empirical evaluation. *IEEE Transactions on biomedical engineering*, 51(5):719–727, 2004.

[47] Minh-Ngoc Tran, Trong-Nghia Nguyen, and Viet-Hung Dao. A practical tutorial on variational bayes. *arXiv preprint arXiv:2103.01327*, 2021.

[48] Bart Van Den Broek, Wim Wiegerinck, and Bert Kappen. Graphical model inference in optimal control of stochastic multi-agent systems. *Journal of Artificial Intelligence Research*, 32:95–122, 2008.

[49] Aki Vehtari and Jouko Lampinen. Bayesian model assessment and comparison using cross-validation predictive densities. *Neural computation*, 14(10):2439–2468, 2002.

[50] Aki Vehtari, Daniel Simpson, Andrew Gelman, Yuling Yao, and Jonah Gabry. Pareto smoothed importance sampling. *Journal of Machine Learning Research*, 25(72):1–58, 2024.

[51] Martin J Wainwright, Michael I Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

[52] Pengyu Wang and Phil Blunsom. Collapsed variational bayesian inference for hidden markov models. In *Artificial Intelligence and Statistics*, pages 599–607. PMLR, 2013.

[53] Lilian Weng. From autoencoder to beta-vae, 2018. Accessed: 26-Apr-2025.

[54] Christopher Wolf, Maximilian Karl, and Patrick van der Smagt. Variational inference with hamiltonian monte carlo. *arXiv preprint arXiv:1609.08203*, 2016.

[55] Yuling Yao, Aki Vehtari, Daniel Simpson, and Andrew Gelman. Using stacking to average bayesian predictive distributions (with discussion). 2018.

[56] Yuling Yao, Aki Vehtari, Daniel Simpson, and Andrew Gelman. Yes, but did it work?: Evaluating variational inference. In *International Conference on Machine Learning*, pages 5581–5590. PMLR, 2018.

[57] Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.