The Sensitivity of Variational Bayesian Neural Network Performance to Hyperparameters

Scott Koermer *1 and Natalie Klein¹

¹Statistical Sciences Group, Los Alamos National Laboratory, Los Alamos, NM, USA

September 26, 2025

Abstract

In scientific applications, predictive modeling is often of limited use without accurate uncertainty quantification (UQ) to indicate when a model may be extrapolating or when more data needs to be collected. Bayesian Neural Networks (BNNs) produce predictive uncertainty by propagating uncertainty in neural network (NN) weights and offer the promise of obtaining not only an accurate predictive model but also accurate UQ. However, in practice, obtaining accurate UQ with BNNs is difficult due in part to the approximations used for practical model training and in part to the need to choose a suitable set of hyperparameters; these hyperparameters outnumber those needed for traditional NNs and often have opaque effects on the results. We aim to shed light on the effects of hyperparameter choices for BNNs by performing a global sensitivity analysis of BNN performance under varying hyperparameter settings. Our results indicate that many of the hyperparameters interact with each other to affect both predictive accuracy and UQ. For improved usage of BNNs in real-world applications, we suggest that global sensitivity analysis, or related methods such as Bayesian optimization, should be used to aid in dimensionality reduction and selection of hyperparameters to ensure accurate UQ in BNNs.

1 Introduction

Robust uncertainty quantification (UQ) is essential when applying machine learning models in high-stakes applications. For instance, in scientific applications, training a neural network (NN) that can predict well on data similar to that seen during training may not be sufficient, as we would like to know regions of input space where uncertainty increases to potentially collect more data or refine hypotheses. For instance, UQ can guide experimental design in materials science by identifying high-uncertainty regions in the search space [1], can help manage risk in statistical downscaling in climate modeling [2], and can identify when NNs for turbulence modeling [3] or geological analysis [4] begin to extrapolate to new regimes. However, while UQ has been essential for application of NNs in scientific areas, poorly calibrated UQ can lead to overconfident predictions and erroneous scientific conclusions. For true integration of machine learning methods such as NNs into science, reliable UQ essential [5].

Bayesian neural networks (BNNs) [6, 7] offer a promising approach that combines the empirical success of NN models across a plethora of prediction tasks with the principled UQ that arises from Bayesian modeling [8]. However, in practice, BNNs present unique challenges, in part because exact Bayesian inference is not possible (necessitating approximate inference methods) and in part because the complexity of the models and the number of hyperparameters increases relative to standard neural networks (NNs). In particular, when training BNNs, one must consider prior distribution hyperparameters and hyperparameters specific to the approximate inference method. In practice, tuning hyperparameters for BNNs appears to be more challenging than for non-Bayesian NNs; what is done in practice often appears closer to an art than a science [9]. This calls into question the reliability of UQ from BNN approaches.

^{*}Corresponding author, skoermer@lanl.gov

In this paper, we investigate how the performance of BNNs is influenced by hyperparameter choices in two synthetic data sets and seek to offer general guidance for hyperparameter tuning for successful BNN inference. We focus on the use of variational inference (VI) [10], a tractable approximate inference method frequently chosen for computationally tractable BNNs [11, 12], though we acknowledge that there are alternative inference schemes including Monte Carlo dropout [13], deep ensembles [14], Laplace approximations [15], and low-rank Gaussian approximations [16]. Our methodology for evaluating model performance and investigating sensitivity to hyperparameters could extend to such alternate inference approaches, but in this paper, we focus on general principles and procedures for obtaining good results with VI.

Good results are of course subjective, but we focus on quantifying two key aspects of a BNN fit: predictive accuracy of the average prediction and quality of predictive uncertainty intervals. In contrast, previous work on methodology for inference for BNNs typically focuses only on error rate or average likelihood. For instance, the introduction of stochastic variational inference for BNNs focused on phoneme error rate for the TIMIT speech corpus [11], while the expectation of the log likelihood of a test set over the posterior distribution of BNN weights is a popular modern measure of uncertainty for testing novel variational methodologies [17, 18, 19]. However, utilizing expected log-likelihood on a test set is only sensible in model comparison contexts and for methodological development, but does not inform a practitioner about whether their predictive uncertainty intervals are reasonable and likely to be useful in a specific application. In contrast, we assess how hyperparameters affect not only predictive accuracy (via root mean squared error, or RMSE) but also uncertainty intervals (via interval score, or IS, a metric that measures both coverage and interval width), with a focus on extracting information on the role that hyperparameters play in BNN inference.

Here, we utilize global sensitivity analysis [20] to work towards a more principled process for hyperparameter selection in BNNs. Global sensitivity analysis evaluates the impact of hyperparameter choices across an entire range of options and is particularly valuable when the response of interest (in this case, BNN performance) is nonlinear with respect to the hyperparameters. Using global sensitivity analysis, we can rank the influence of different hyperparameters and understand how they interact. In this paper, we evaluate the global sensitivity of two different VI algorithm variants to hyperparameters on two different synthetic data sets. Our results illustrate the complexity of BNN inference, underscore the necessity of tuning hyperparameters carefully rather than relying on heuristic default values, and open up the possibility for future research to better understand how BNN hyperparameters interact in complex ways to impact the quality of UQ. We hope our results are useful to those seeking to use BNNs in areas where calibrated UQ is critical, such as in scientific fields as disparate as climate modeling, medicine, physics simulations, and materials design.

In Sec. 2, we give background on BNNs and VI, detail the performance metrics we use, give an overview of global sensitivity analysis, and provide specific experimentation details. Sec. 3 summarizes the results of the global sensitivity analysis, including both main effects plots and tabulated sensitivity indices. Finally, in Sec. 4, we discuss the findings, propose future research related to additional sensitivity analysis investigations, and offer suggestions for practical hyperparameter selection in BNNs.

2 Materials and Methods

In this section, we first give background information on BNNs and VI (Secs. 2.1 and 2.2), followed by an exposition of performance metrics we will use to evaluate hyperparameter choices (Sec. 2.3). Sec. 2.4 explains the global sensitivity analysis we use to determine the impact of hyperparameters on each metric. Finally, we discuss our specific experimental procedures in Sec. 2.5.

2.1 Bayesian Neural Networks (BNNs)

Traditional NNs propose a function $g(x, \boldsymbol{\theta})$ that depends on parameters $\boldsymbol{\theta}$ (typically comprised of weights and biases for the neural network layers) and learns to predict outputs y by optimizing over $\boldsymbol{\theta}$ using training pairs $\{(x_i, y_i)\}_{i=1}^n$ and a loss function that seeks to align g with the observed y values. In contrast, Bayes' rule posits a posterior distribution over parameters rather than a single setting of parameters:

$$p(\boldsymbol{\theta}|x,y) = \frac{p(y|x,\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(y|x)} \propto p(y|x,\boldsymbol{\theta})p(\boldsymbol{\theta}).$$

In Bayesian inference, the goal is to infer the posterior distribution $p(\theta|x,y)$ of our model parameters θ given our data observations $\{(x_i,y_i)\}_{i=1}^n$ using a likelihood function $p(y|x,\theta)$ and prior distribution $p(\theta)$ [6, 7, 21]. In BNNs, the likelihood is typically specified by the NN function f as part of some chosen conditional probability distribution; for instance, for regression tasks, a common likelihood is Gaussian: $p(y|x,\theta) \sim \mathcal{N}(g(x,\theta),\sigma^2)$ where σ^2 is a noise variance that can be fixed or learned. Prior distributions are often specified with simple distributions such as independent standard Gaussians for each weight [22], though recent works have explored alternate choices [23, 24]. Given the prior and likelihood, exact evaluation of the posterior distribution is typically not possible for complicated models such as neural networks. Markov Chain Monte Carlo (MCMC, [25]) is a common method to produce samples from the posterior distribution, but presents challenges in neural networks due to high dimensionality and multimodality of the parameter space [26]. Therefore, most existing applications of Bayesian neural networks rely on approximate inference schemes such as variational inference, discussed in the next section.

Given an approximation of the posterior distribution, predictive uncertainty in BNNs is obtained via the posterior predictive distribution

$$p(y^* | x^*, x, y) = \int p(y^* | x^*, \boldsymbol{\theta}, \sigma^2) p(\theta | x, y) d\theta d\sigma^2.$$

That is, for a new input x^* , we can compute the probability of the output y^* conditional on x^* by integrating over parameters sampled from the posterior. Note that in this formulation we have treated σ^2 as a known and fixed noise variance as part of the likelihood. In practice, we will approximate this integral with samples, so a typical procedure would be to repeatedly sample θ from the approximate posterior, then sample y^* from the likelihood given both θ and the fixed noise variance σ^2 .

2.2 Variational Inference (VI)

Because exact posterior inference for BNNs is difficult, typical BNN methods approximate the posterior distribution. One approximation method, variational inference (VI), aims to minimize a statistical distance from a variational approximate posterior $q(\theta)$ chosen to lie in some distributional family to the true posterior distribution $p(\theta|x,y)$ [10]. In this work, we compare results using two different measures of statistical distance. The first, Kullback Leibler (KL) divergence [27], is more commonly used in BNNS [11, 28] and its use can be considered analogous to viewing Bayesian inference as an optimization problem [29] when q() is the same distributional form as the posterior. We utilize the equality

$$\mathrm{KL}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|x,y)] = -\mathbb{E}_{q(\boldsymbol{\theta})}\left[p(y|x,\boldsymbol{\theta})\right] + \mathrm{KL}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})] + \log(p(y|x))$$

in the estimation; because for a fixed set of training data and prior, $\log(p(y|x))$ is constant and positive, we have that

$$KL[q(\boldsymbol{\theta})||p(\boldsymbol{\theta}|x,y)] \le -\mathbb{E}_{q(\boldsymbol{\theta})}[p(y|x,\boldsymbol{\theta})] + KL[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})]. \tag{1}$$

The right hand side forms and objective function that we can minimize in an effort to minimize the KL divergence on the left hand side by bounding it from above. For typical choices of prior and q, the KL divergence in Eq. 1 is available in closed form, but the expectation must generally be estimated via Monte Carlo sampling. In practice, the objective function that is utilized is often written

$$\mathcal{L}_{\mathrm{KL}}(q) = -\mathbb{E}_{q(\boldsymbol{\theta})}\left[p(y|x,\boldsymbol{\theta})\right] + \gamma \mathrm{KL}[q(\boldsymbol{\theta})||p(\boldsymbol{\theta})]. \tag{2}$$

The parameter γ on the right hand side of Equation 2 is naturally set equal to one for the inequality to hold. However, in practice, values of γ are chosen to downweight or upweight the influence of the prior distribution on inference [30]. In addition to γ , other hyperparameters for variational inference with \mathcal{L}_{KL} include the prior hyperparameters, the number of Monte Carlo samples used for estimating the expectation, the learning rate for optimization, and the number of steps in the optimization.

Another choice of divergence that has been recently explored is α -Renyi divergence [12, 29]. The divergence is given by

$$\mathcal{L}_{AR}(q) = D_{AR}^{\alpha}[q(\theta) || p(\theta|y)] = \frac{1}{\alpha - 1} \log \left(\int p(\theta|y)^{\alpha} q(\theta)^{1 - \alpha} d\theta \right). \tag{3}$$

We note that for $\alpha \to 1$, one can show that this divergence becomes the KL divergence, so it can be seen as a generalization of KL divergence. The hyperparameters are similar to those for \mathcal{L}_{KL} with the addition of the α parameter, but there is no γ weighting term. Similar to KL divergence, the integral is evaluated via Monte Carlo sampling.

2.3 BNN Performance Metrics

We investigate the sensitivity two performance metrics for quantifying the goodness of model fit to settings of our VI hyperparameters. Often, the performance of NNs and BNNs is judged using a measure of accuracy such as root mean squared error (RMSE; Eq. 4) comparing the values of a testing set to the expected function value $\hat{g}(x_i)$ predicted from the fitted model. For BNNs, it is common to use the mean of the posterior predictive distribution to make predictions \hat{g} for accuracy computations. The ideal RMSE value for a model is a function of the true noise of the data generating mechanism as well as the number of test points and the uniformity of the test points in the input space X, but a lower RMSE is typically considered a better fit. RMSE is defined as

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N_{\text{test}}} [y_{\text{test}}(x_i) - \hat{g}(x_i)]^2}.$$
 (4)

However, RMSE does not directly inform about predictive UQ, so we additionally compute the interval score (IS; Eq. 5) [31, 32] on our testing set based on samples from the posterior predictive distribution. Intuitively, the IS measures the accuracy of a prediction interval produced at the $\alpha_{\rm IS}$ level with an upper bound of $u(x_i^{\rm IS})$ and a lower bound of $l(x_i^{\rm test})$ for given testing input location $x_i^{\rm test}$ and is defined as

$$IS = \sum_{i=1}^{N_{\text{test}}} (u(x_i^{\text{test}}) - l(x_i^{\text{test}})) + \frac{2}{\alpha_{\text{IS}}} (l(x_i^{\text{test}}) - y_{\text{test}}(x_i^{\text{test}})) 1(y_{\text{test}}(x_i^{\text{test}}) < l(x_i^{\text{test}}))$$

$$+ \frac{2}{\alpha_{\text{IS}}} (y_{\text{test}}(x_i^{\text{test}}) - u(x_i^{\text{test}})) 1(y_{\text{test}}(x_i^{\text{test}}) > u(x_i^{\text{test}})).$$
(5)

The function 1(a < b) is equal to 1 when the equality contained within the parenthesis is true, and equal to 0 when the equality is false. A lower IS is considered to indicate a better fit. The intuition is that we want predictive intervals that contain the true values, but that are not overly wide. The first term in Eq. 5 penalizes intervals which are too wide. The second and third terms penalize testing data points which lie outside of the prediction interval.

2.4 Global Sensitivity Analysis

Our goal is to investigate how varying hyperparameter settings, and combinations of hyperparameter settings, affects RMSE and IS in BNNs. Global sensitivity analysis provides such results [20]. Specifically, we will investigate how much a single factor can independently affect RMSE and IS through main effects and first order sensitivity indices, as well as how the interaction of a parameter with other parameters affects results through total sensitivity indices. We choose global sensitivity analysis for our investigation because of the limitations of local derivative-based sensitivity for functions of uncertain form (see Chapter 1 of [20]). Essentially, when a derivative is evaluated under function uncertainty and the function is not first-order linear, interpretation becomes difficult both at the location of evaluation as well as when extrapolating to a larger domain. Global sensitivity analysis instead considers a set of parameter ranges and weights them under the consideration of a measure on their probability of occurrence. This allows us to examine how factors influence the outcome both individually and in combination, crucial for nonlinear responses with correlated inputs. In this case, we consider marginally uniform distributions via Latin hypercube sampling (LHS), but other prior distributional forms of the parameters can be considered [33, 34].

In global sensitivity analysis, we typically sample hyperparameters ω and evaluate the response property of interest (for instance, the BNN RMSE) for each hyperparameter setting. Because this process is time-consuming (requiring fitting a BNN for each selection of hyperparameters), we utilize a surrogate model $f(\omega)$ that approximates the BNN performance for a new setting of hyperparameters (Sec. 2.4.1). Based on this

surrogate model, we can investigate a number of metrics of sensitivity, including main effects (Sec. 2.4.2), first order sensitivity indices (Sec. 2.4.3), and total sensitivity indices (Sec. 2.4.4) while incorporating uncertainty in the surrogate model into the analysis.

2.4.1 Treed Gaussian Process Surrogate Model

While a variety of surrogate models could be utilized for global sensitivity analysis, here we use a fully Bayesian surrogate model to incorporate uncertainty in the surrogate model into our analysis. Because we do not expect the response to be smooth with respect to the hyperparameters, we use a treed Gaussian process (TGP) [34] for each response of interest (RMSE and IS). TGPs are flexible models frequently used to emulate responses and partition the input space with separately parameterized Gaussian process (GP) models in order to allow for further flexibility. Specifically, we propose a surrogate

$$f(\omega) \sim GP_r(\mu_r(\omega), k_r(\omega, \omega'))$$
 for ω in region r ,

where μ_r is the local Gaussian process mean function for region r, k_r is the covariance function for region r, and the regions are determined using a tree structure that splits the input space. We utilize the tgp package [35] in R, which provides a fully Bayesian implementation of the model using Markov Chain Monte Carlo (MCMC) to sample from the posterior distribution. The tgp package additionally provides functionality for global sensitivity analysis [20], leveraging the TGP model fit to a provided set of data to approximate the integrals required for global sensitivity analysis. Further details of implementation can be found in Sec. 2.5.4 and [35].

2.4.2 Main Effects

Main effects can be considered as the effect of a single parameter of interest, if we average, or take the expectation, over the other parameters:

$$\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j] = \int f(\omega) \ p(\omega_{-j}|\omega_j) d\omega_{-j}.$$

Here, ω_j is our j^{th} parameter of interest for the main effect, and ω_{-j} is the set of parameters excluding parameter ω_j , where we assume a sufficiently large LHS sample such that $p(\omega_{-j}|\omega_j) \approx p(\omega_{-j})$. The result of the numerical evaluation of this integral is a function which can be evaluated for a value of ω_j when averaging over possible function evaluations of the remaining ω_{-j} parameters. That is, the main effect gives us the expected value of f for a given setting of ω_j while averaging across all other hyperparameters. With a fully Bayesian surrogate model such as a TGP, we also obtain uncertainty of the main effect. Uncertainty quantification of the main effect is related to GP model uncertainty [35]. The main effect is numerically approximated conditioned on a set of sampled GP parameters, and the process is repeated within each iteration of an MCMC algorithm (drawing from the posterior distribution over f) for generating posterior draws of model parameters.

2.4.3 First Order Sensitivity Indices

The first order sensitivity index is a measurement of how a single factor ω_j contributes to the variance of the outcome of a function, under the assumption that the remaining factors are held constant. If we think of the main effect $\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j]$ as a function of variable ω_j , then the first order sensitivity index is related to the variance, with respect to ω_j , of the main effect function as $\int (\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j] - \mathbb{E}_{\omega}[f(\omega)])^2 p(\omega_j)d\omega_j$.

The sensitivity analysis functions in the tgp R package only consider the variance of the surrogate function, and ignores the noise of the response for sensitivity analysis [36]. Therefore, the first order sensitivity index is the fraction of the total variance of our surrogate function which is explained by input ω_i :

$$S_j = \frac{\mathbb{V}\mathrm{ar}_{\omega_j}(\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j])}{\mathbb{V}\mathrm{ar}(f(\omega))}.$$
 (6)

Another way to consider the interpretation of the first order sensitivity analysis relies on the law of total variance decomposition $\mathbb{V}\operatorname{ar}(f(\omega)) = \mathbb{E}_{\omega_j}[\mathbb{V}\operatorname{ar}_{\omega_{-j}}(f(\omega)|\omega_j)] + \mathbb{V}\operatorname{ar}_{\omega_j}(\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j])$. Dividing both sides by $\mathbb{V}\operatorname{ar}(f(\omega))$ leads to

$$\frac{\mathbb{V}\mathrm{ar}(f(\omega))}{\mathbb{V}\mathrm{ar}(f(\omega))} = \frac{\mathbb{E}_{\omega_j}[\mathbb{V}\mathrm{ar}_{\omega_{-j}}(f(\omega)|\omega_j)]}{\mathbb{V}\mathrm{ar}(f(\omega))} + \frac{\mathbb{V}\mathrm{ar}_{\omega_j}(\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j])}{\mathbb{V}\mathrm{ar}(f(\omega))}.$$

In the right hand summand, $\mathbb{V}ar_{\omega_j}(\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j])$ is the variance of our function $f(\omega)$ if we only chose to model ω_j , the variance of our main effect, while $\mathbb{E}_{\omega_j}[\mathbb{V}ar_{\omega_{-j}}(f(\omega)|\omega_j)]$ is the residual variation in $f(\omega)$ that can be attributed to the remaining ω_{-j} parameters and interactions between ω_j and ω_{-j} . Dividing each term by $\mathbb{V}ar(f(\omega))$ produces the fraction of the variation of $f(\omega)$. Because $\mathbb{E}_{\omega_j}[\mathbb{V}ar_{\omega_{-j}}(f(\omega)|\omega_j)]$ is the residual variation this term contains the first order sensitivity of each ω_{-j} , as well as terms related to the total effect of ω_j . Next, we will show how to decouple the total effect terms for an ω_j from the residual variance $\mathbb{E}_{\omega_j}[\mathbb{V}ar_{\omega_{-j}}(f(\omega)|\omega_j)]$.

2.4.4 Total Sensitivity Indices

A total sensitivity index, T_i of factor ω_j is the sum of all first order and higher sensitivity index terms related to factor ω_j . The law of total variance allows for the decomposition

$$\mathbb{V}\mathrm{ar}_{\omega_{i}}(\mathbb{E}_{\omega_{-i}}[f(\omega)|\omega_{j}]) + \mathbb{E}_{\omega_{i}}[\mathbb{V}\mathrm{ar}_{\omega_{-i}}(f(\omega)|\omega_{j})] = \mathbb{V}\mathrm{ar}_{\omega_{-i}}(\mathbb{E}_{\omega_{i}}[f(\omega)|\omega_{-j}]) + \mathbb{E}_{\omega_{-i}}[\mathbb{V}\mathrm{ar}_{\omega_{i}}(f(\omega)|\omega_{-j})]. \tag{7}$$

On the left hand side of Eq. 7, we have terms related to the first order sensitivity $(\mathbb{V}\mathrm{ar}_{\omega_j}(\mathbb{E}_{\omega_{-j}}[f(\omega)|\omega_j]))$ and the remainder of the total variance $(\mathbb{E}_{\omega_j}[\mathbb{V}\mathrm{ar}_{\omega_{-j}}(f(\omega)|\omega_j)])$. On the right hand side, we have $\mathbb{V}\mathrm{ar}_{\omega_{-j}}(\mathbb{E}_{\omega_j}[f(\omega)|\omega_{-j}])$, which is related to the all first order and higher sensitivity indices for all ω_{-j} factors [20]. The term $\mathbb{E}_{\omega_{-j}}[\mathbb{V}\mathrm{ar}_{\omega_j}(f(\omega)|\omega_{-j})]$ includes variability related to ω_j and interactions between ω_j and ω_{-j} . We can therefore find the total sensitivity indices as

$$T_{j} = S_{j} + \frac{\mathbb{E}_{\omega_{j}}[\mathbb{V}\operatorname{ar}_{\omega_{-j}}(f(\omega)|\omega_{j})] - \mathbb{V}\operatorname{ar}_{\omega_{-j}}(\mathbb{E}_{\omega_{j}}[f(\omega)|\omega_{-j}])}{\mathbb{V}\operatorname{ar}(f(\omega))} = 1 - \frac{\mathbb{V}\operatorname{ar}_{\omega_{-j}}(\mathbb{E}_{\omega_{j}}[f(\omega)|\omega_{-j}])}{\mathbb{V}\operatorname{ar}(f(\omega))}.$$
 (8)

The term $\mathbb{E}_{\omega_j}[\mathbb{V}ar_{\omega_{-j}}(f(\omega)|\omega_j)] - \mathbb{V}ar_{\omega_{-j}}(\mathbb{E}_{\omega_j}[f(\omega)|\omega_{-j}])$ removes explanations of variance irrelevant to ω_j from the residual variance term found in Eq. 6 and summing with S_j produces the total sensitivity index. If we considered ω_{-j} to be a vector instead of a scalar parameter, we could say $\mathbb{V}ar_{\omega_{-j}}(\mathbb{E}_{\omega_j}[f(\omega)|\omega_{-j}])$ is the equivalent to the *first order index* of vector parameter ω_{-j} . The right most side of the equality in Eq. 8 shows removal of the proportion of variance explained by ω_{-j} from the total variance of $f(\omega)$. In interpreting the total sensitivity indices, it is often useful to compare them to the first order sensitivity indices; the difference between the two gives information on how much interactions with other parameters drive the total effect of a given parameter.

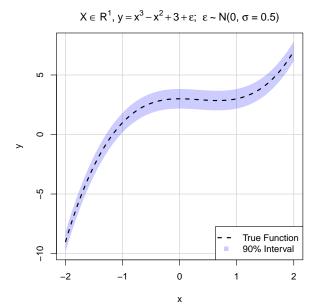
2.5 Experimental Methods

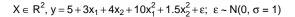
In this section, we first describe synthetic data generating mechanisms we used as test cases for fitting BNNs (Sec. 2.5.1); we then give details on our BNN implementations (Sec. 2.5.2), our experimental setup (Sec. 2.5.3), and the code to reproduce our results (Sec. 2.5.4).

2.5.1 Data Generating Mechanisms

For the global sensitivity analysis, we chose two simple data generating mechanisms that differ in the number of inputs to the underlying function; both data generating mechanisms have continuous inputs and outputs and homoskedastic noise. We avoided difficult test functions to ensure the difficulty of the problem does not confound the results (that is, ensuring the BNNS can fit the underlying function reasonably well without needing thorough architecture search). The first data generating mechanism is a polynomial function of one input:

$$y = x^3 - x^2 + 3 + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma = 0.5). \tag{9}$$





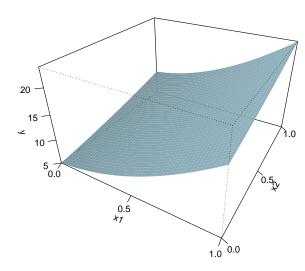


Figure 1: The data generating mechanisms used for testing the variational BNN approximations. For the $X \in \mathbb{R}^1$ data generating mechanism (left) the expected value of the function is plotted along with the 90% conditional distribution interval arising from additive homoskedastic noise. For the $X \in \mathbb{R}^2$ data generating mechanism (right), we show the expected value of the function with respect to the two input dimensions.

This data generating mechanism is similar to what is used as a demo function for the torchbnn package [37]. Because the input dimension of Eq. 9 is d=1, we refer to this data generating mechanism as $X \in \mathbb{R}^1$ in shorthand. Training and testing data were obtained as uniformly random on $X \in [-2, 2]$.

The second data generating mechanism we investigate is

$$y = 5 + 3x_1 + 4x_2 + 10x_1^2 + 1.5x_2^2 + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma = 1). \tag{10}$$

We refer to the data generating mechanism in Eq. 10 as $X \in \mathbb{R}^2$ because the dimensionality of the input space is d = 2. For $X \in \mathbb{R}^2$ training and testing data was uniformly sampled on $[0,1]^2$. Plots of the data generating mechanisms are shown in Fig. 1.

2.5.2 BNN Implementation

For our BNNs, we assume each observation is independently and identically distributed, given the known noise variance σ^2 , a set of neural network weights and biases $\boldsymbol{\theta}$, and location x_i . We choose a univariate Gaussian likelihood function, with the mean specified as the output of the NN $g(x, \boldsymbol{\theta})$ and known noise variance σ^2 . Each neural network weight and bias in the set of NN parameters $\boldsymbol{\theta}$ has its own independent univariate normal prior distribution with identical prior means μ_0 and prior variances σ_0^2 for each parameter. We note that both μ_0 and σ_0^2 are varied as part of our sensitivity analysis.

The NN architectures we use consist of two layers, with the output of the first layer passed through a $\tanh(x)$ activation function. The number of features passed between the NN layers is varied in our experiments and ranges from 2 to 100. A bias vector is added to the output of the first layer and a bias scalar value is added to the output of the second layer. For the $X \in \mathbb{R}^1$ mechanism, the inputs for both testing and training data are scaled to [0,1]; rescaling inputs is not necessary for the $X \in \mathbb{R}^2$ mechanism. We assume in our setup that the noise variance is known so that our investigation focuses on the variance of the BNN functional output. For both the $X \in \mathbb{R}^1$ and $X \in \mathbb{R}^2$ mechanisms we rescale training responses y_{train} and testing responses y_{test} as

$$\frac{y - \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} h(x_i^{\text{train}})}{\sqrt{\sigma^2 + \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \left(h(x_i^{\text{train}}) - \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} h(x_i^{\text{train}})\right)^2}.$$

Where, $h(x_i^{\text{train}})$ is the data generating mechanism evaluated at the i^{th} training data location x_i^{train} before noise is added, N_{train} is the number of training data points, and σ^2 is the known noise variance.

For our variational approximations, we choose independent Gaussian distributions $q(\theta_i|\hat{\mu}_i, \hat{\sigma}_i^2)$ for weight or bias θ_i which has a uniquely estimated mean $\hat{\mu}_i$ and variance $\hat{\sigma}_i^2$. Therefore, the number of parameters required for estimation through optimization is twice the number of non-Bayesian NN weights and biases. For fitting, we use stochastic gradient descent and estimate any expectations in the loss functions via Monte Carlo sampling with the reparameterization trick (Bayes by Backprop; see [28]). The learning rate, number of optimizer steps, and number of Monte Carlo samples are all varied in the sensitivity analysis.

2.5.3 Experimental Setup

We collected data to analyze how the hyperparameter choices required for variational approximations affect the RMSE and IS metrics. The parameters we vary are the KL reweighting γ (Eq. 2), the α parameter utilized in calculating α -Renyi divergence (Eq. 3), the prior mean μ_0 , the prior standard deviation σ_0 , the number of optimizer steps, the number of NN features, the number of samples drawn from the variational posterior in order to approximate the integrals required for KL divergence or α -Renyi divergence, and the learning rate (LR). Instead of uniformly sampling over the natural values of γ , σ_0 , and LR, we uniformly sample over \log_{10} of these values in order to better space out the smaller values.

To obtain a well spaced set of data for our parameters of interest, we generated a 750-point 7-dimensional Latin hypercube sample (LHS) [38] across the hyperparameters in Table 1 for each of the four combinations of data generating mechanism (\mathbb{R}^1 or \mathbb{R}^2) and statistical distance (KL or α -Renyi). The points in an LHS are uniformly distributed marginally over each of the 7 dimensions, but have constraints which ensure the points are well spaced within the full 7 dimensional space. Our LHS was bounded by a hyper-rectangle with bounds shown in Table 1. The bounds are the same for both data generating mechanisms and choices of statistical distance. For each hyperparameter setting, we fit a BNN to generated data and evaluated the RMSE and IS on a held-out test set for sensitivity analysis.

Table 1: Bounds for hyperparameters explored with global sensitivity analysis. For a given statistical distance (KL or α -Renyi), the bounds shown are used with both the $X \in \mathbb{R}^1$ and $X \in \mathbb{R}^2$ data generating mechanisms.

	$\log_{10}(\gamma)$	α	$\log_{10}(\sigma_0)$	Optimizer Steps	NN Features	Samples for Integration	$\log_{10}(LR)$	μ_0
KL Parameters								
Upper Bound	1	NA	0.5	20000	100	25	-0.3	2
Lower Bound	-1	NA	-0.5	2000	2	1	-3.3	-2
α -Renyi Paran	neters							
Upper Bound	NA	1	0.5	20000	100	25	-0.3	2
Lower Bound	NA	0	-0.5	2000	2	1	-3.3	-2

2.5.4 Code and Software

Our code for generating data and analysis can be found at the GitHub repository https://github.com/lanl/multiverse/tree/main [39] in the sensitivity_bnns folder. Within the src folder the functions kl_div.py and alpha_renyi.py define functions which leverage the torchbnn [37] and PyTorch [40] packages to fit variational BNNs with KL divergence and α -Renyi divergence respectively. evaluate.py provides functions for obtaining predictions and uncertainty quantification required for evaluating RMSE and IS. prepare_experiment.py generates the testing and training data used for each BNN fit as well as the LHS sample specifying tuning parameter choices varied for the experiment. main.py is run for each row of the LHS defining the experiment to fit a BNN, quantify its performance, and save the results. merge_results.py helps to merge results obtained in parallel into one *.csv file for sensitivity analysis using R code. Details on additional script functionality are provided within the sensitivity_bnns folder specific README.md file.

Table 2: First order and total sensitivity indices, displayed as first order (total), for each tuning parameter when KL divergence is used for variational inference; top section corresponds to the RMSE metric sensitivity while bottom section corresponds to the interval score (IS) metric sensitivity, with results shown for both data generating mechanisms ($X \in \mathbb{R}^1$ and $X \in \mathbb{R}^2$). It appears learning rate (LR) has the largest first order and total effects. First order effects are generally small for the other hyperparameters, but each has much larger total effects, indicating there are interactions between parameters affecting the outcome.

	$\log_{10}(\gamma)$	$\log_{10}(\sigma_0)$	Optimizer Steps	NN Features	Samples for Integration	$\log_{10}(LR)$	μ_0
KL, RMS	\mathbf{E}						
$X \in \mathbb{R}^1$	0.09(0.39)	0.01(0.36)	0.08(0.42)	0.04(0.36)	0.03(0.30)	0.23(0.69)	0.01(0.26)
$X \in \mathbb{R}^2$	0.01(0.46)	0.05(0.61)	0.04(0.56)	0.03(0.51)	0.02(0.46)	0.13(0.81)	0.00(0.42)
KL, IS							
$X \in \mathbb{R}^1$	0.04(0.29)	0.03(0.47)	0.06(0.45)	0.04(0.35)	0.02(0.27)	0.19(0.74)	0.01(0.25)
$X \in \mathbb{R}^2$	0.01(0.18)	0.04(0.40)	0.05(0.41)	0.04(0.44)	0.01(0.23)	0.12(0.76)	0.01(0.37)

Our sensitivity analysis utilized the sens() function within the tgp package. We chose to model RMSE and IS with a Bayesian treed Gaussian process, setting the argument model = btgp. A first order linear model is specified as the prior mean of the GP. The sens() function utilizes MCMC algorithms to obtain posterior draws of the model parameters which are then iteratively utilized with numerical methods to calculate main effects, first order sensitivity indices, and total sensitivity indices[35]. For MCMC sampling of posterior parameters we specify the argument BTE = c(5000, 150000,10) which stipulates that a total of 150,000 posterior samples are obtained, the first 5000 are discarded under the assumption that the algorithm did not yet converge on the target distribution, and only every 10th sample was utilized to calculate sensitivity statistics to reduce effects of autocorrelation [25]. We set the argument nn.lhs = 7000 for our numerical methods approximating the integrals over the input space required to obtain the desired sensitivity statistics. This setting results in 63,000 predictive location evaluations for the Monte Carlo integration scheme for each GP posterior parameter sample considered. The predictive locations were bounded by the parameter ranges which can be found in Table 1. The remainder of the function settings were the default settings outlined in the package documentation [36].

3 Results

In this section, we first give high-level sensitivity analysis results based on first order and total sensitivity indices, then proceed to discuss the main effects of each hyperparameter in detail. In Fig. 2, we show examples of the best and worst fits obtained for KL divergence with respect to both interval score and RMSE for the $X \in \mathbb{R}^1$ data generating mechanism. Qualitatively, the best fits are similar and do a good job capturing the conditional distribution of the data, indicating that well-fitting BNNs can achieve good performance with respect to both metrics. The worst-case fits, on the other hand, show evidence of both bias and high variance, leading to poor RMSE and interval scores. The hyperparameters corresponding to each fit are given in Table A1.

Table 2 gives the first order and total sensitivity indices for BNNs when KL divergence is used as the loss function, while Table 3 gives the indices for BNNs when α -Renyi divergence is used as the loss function. In both cases and across both data generating mechanisms, the learning rate (LR) has the largest first order and total effects, indicating that selection of a good learning rate is essential in BNN fitting. For KL divergence, the first order sensitivity indices of $\log_{10}(\gamma)$ and optimizer steps are larger for $X \in \mathbb{R}^1$ compared to $X \in \mathbb{R}^2$, suggesting that these hyperparameters need to be tuned for each specific data set. First order sensitivity indices for α -Renyi divergence are relatively lower than for KL divergence, suggesting that changing specific hyperparameters in isolation may have less effect on VI with α -Renyi divergence than with KL divergence.

However, the bulk of the variability is explained in the total sensitivity indices, as most of the hyperparameters tend to have small main effects. This indicates that in isolation they do not influence the outcome

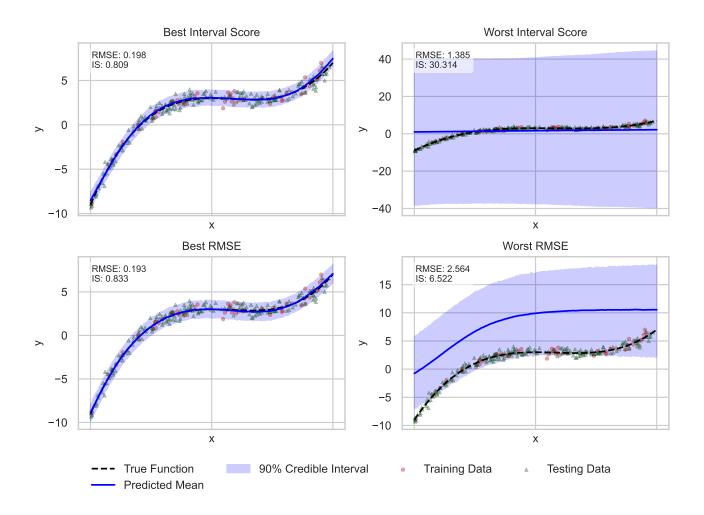


Figure 2: BNN variational approximations through minimizing KL divergence. The best and worst resulting fits from the 750 different initializations used to collect data are shown. The best fits were selected as either the minimum observed RMSE or IS, and the worst were selected as either the largest observed RMSE or IS. The best fits according to interval score and RMSE are qualitatively similar and appear to describe the data distribution accurately. The worst fit according to interval score appears flat with very high variance, while the worst fit according to RMSE incurs high bias and moderately high variance.

Table 3: First order and total sensitivity indices, displayed as first order (total), for each tuning parameter when α -Renyi divergence is used for variational inference; same format as Table 2 which gave results for KL divergence. Similar to the KL divergence results, learning rate (LR) appears to have the highest first order and total sensitivity indices, with other hyperparameters having small first order but in some cases fairly large total sensitivity, indicating interactions between variables affecting the outcome.

	α	$\log_{10}(\sigma_0)$	Optimizer Steps	NN Features	Samples for Integration	$\log_{10}(LR)$	μ_0
α -Renyi, RN	ASE						
$X \in \mathbb{R}^1$ 0.0	00(0.30)	0.00(0.30)	0.01(0.50)	0.01(0.53)	0.03(0.51)	0.12(0.94)	0.01(0.32)
$X \in \mathbb{R}^2$ 0.0	01(0.58)	0.01(0.57)	0.01(0.71)	0.03(0.78)	0.02(0.69)	0.09(0.88)	0.01(0.62)
α -Renyi, IS							
$X \in \mathbb{R}^1$ 0.0	00(0.65)	0.01(0.67)	0.00(0.65)	0.02(0.69)	0.03(0.79)	0.13(0.93)	0.01(0.69)
$X \in \mathbb{R}^2$ 0.0	01(0.67)	0.01(0.47)	0.00(0.47)	0.01(0.54)	0.03(0.76)	0.12(0.89)	0.01(0.60)

significantly, but they do influence the outcome in combination with the other hyperparameters through interaction effects. For a single hyperparameter, we note that total sensitivity is often significantly different for RMSE and interval score. For example, for KL divergence (Table 2), the total sensitivity of RMSE to $\log_{10}(\sigma_0)$ for $X \in \mathbb{R}^1$ is 0.36 and $X \in \mathbb{R}^2$ is 0.61, while the total sensitivity of interval score is 0.47 for $X \in \mathbb{R}^1$ and 0.4 for $X \in \mathbb{R}^2$. However, there do not appear to be consistent effects when comparing across data generating mechanisms or performance metrics. Finally, there are some trends in magnitude across the tuning parameters. Total sensitivity indices for μ_0 when using KL divergence to estimate the variational fit are lower then their α -Renyi counterparts, as well as lower than the total sensitivity of $\log_{10}(\sigma_0)$ for both variational inference methods. Optimizer steps and interactions explains more of the variance when using α -Renyi divergence than when using KL divergence. The same relationship can be seen for neural network weights and samples obtained to evaluate any Monte Carlo integral within the loss function.

In the remainder of this section, we discuss each hyperparameter in turn; detailed analysis of the main effects of each hyperparameter is based on Figs. 3 and 4 (for KL divergence) and Figs. 5 and 6 (for α -Renyi divergence). In these figures, solid lines represent the expected main effect, and dashed lines represent the resulting 5% and 95% quantiles for the main effect, with the hyperparameter value achieving the best metric value shown with a vertical red dashed line. The first two columns of the matrix of plots are results for the $X \in \mathbb{R}^1$ data generating mechanism and the second two columns correspond to the $X \in \mathbb{R}^2$ data generating mechanism. Columns 1 and 3 plot the main effect on RMSE, while columns 2 and 4 plot the main effect on interval score.

Learning rate As discussed above, learning rate appears to have the largest main and total effects on performance across data generating mechanisms and choice of loss function. Fig. 3.A shows that for KL divergence, there is a preferred learning rate which varies slightly by data generating mechanism; using learning rates that are too small or too large results in clearly worse performance according to both metrics. For α -Renyi divergence (Fig. 5.A), there does appear to be a preferred learning rate with larger learning rates leading to worse performance, but there is perhaps more tolerance for smaller learning rates (in that they can still result in very similar performance).

KL multiplier The main effect of the KL multiplier γ (used only with the KL divergence loss function) is shown in Fig. 3.B; universally across data generating mechanisms and metrics, a smaller KL multiplier results in better performance. This is interesting because there is little theoretical backing for using $\gamma < 1$, which has the effect of downweighting the KL divergence between the prior and the variational posterior in Eq. 2 relative to the expected likelihood term.

 α -Renyi α value Fig. 5.B shows the main effect of α in the α -Renyi divergence. There is generally high uncertainty, and in the $X \in \mathbb{R}^1$ data generating mechanism, it is not clear that any particular value results in better performance. Interestingly though in $X \in \mathbb{R}^2$, there does appear to be a differential effect in which α near 1 (which is most similar to KL divergence) results in the lowest RMSE while α near 0 results in the lowest interval score. These results indicate that different choices of α could in a sense prioritize different aspects of the resulting fit (matching the mean function value well versus characterizing the uncertainty).

Prior variance For KL divergence (Fig. 3.C), there is a differential effect of prior variance σ_0 across data generating mechanisms, with σ_0 near 1 ideal for $X \in \mathbb{R}^1$ and σ_0 near the minimum value preferred for $X \in \mathbb{R}^2$, indicating that the choice of σ_0 that yields the best performance for KL divergence inference may depend heavily on the particular data set. For α -Renyi divergence (Fig. 5.C), it appears that smaller values are preferred across the board, though there is higher uncertainty on the main effects, particularly for RMSE. These results could indicate that perhaps α -Renyi is somewhat less sensitive to the choice of σ_0 compared to KL divergence, though further study is required to draw this conclusion in general across different data sets.

Optimizer steps Increasing the number of optimizer steps is clearly beneficial for KL divergence (Fig. 3.D). However, the effect is less clear for α -Renyi divergence (Fig. 5.D); for $X \in \mathbb{R}^1$, a larger number of steps does appear beneficial, but for $X \in \mathbb{R}^2$, this is not the case (though there is also higher uncertainty).

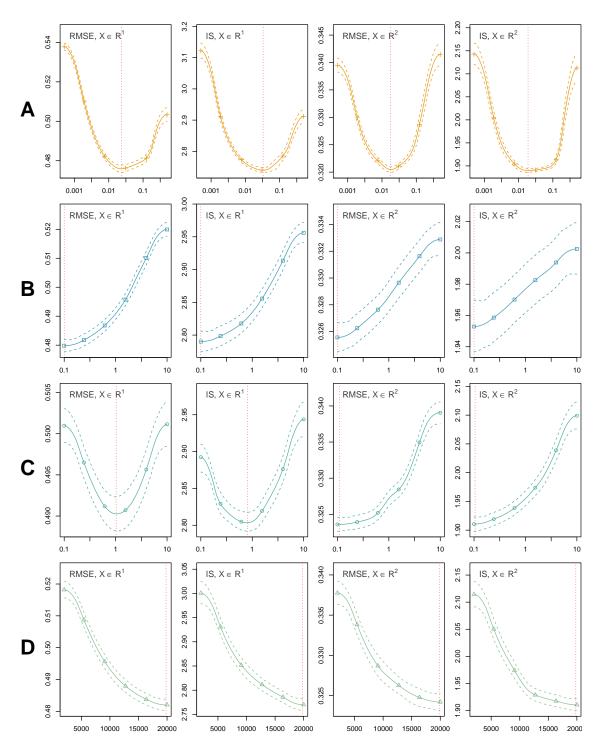


Figure 3: Main effects on RMSE and IS when KL divergence is used for variational inference; each row contains results for each combination of metric (RMSE and IS) and data generating mechanism $(X \in \mathbb{R}^1)$ or $X \in \mathbb{R}^2$). Each row corresponds to a different hyperparameter setting: A) $\log_{10}(\text{Learning Rate})$, B) KL multiplier γ , C) prior variance σ_0^2 , and D) optimizer steps. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color. The minimum of the main effect is shown as a vertical red dashed line. A) indicates that there is an ideal learning rate and that both RMSE and IS increase for smaller or larger learning rates. B) indicates that small KL multipliers result in the lowest RMSE and IS. C) shows that for $X \in \mathbb{R}^1$, prior variance near 1 is ideal, while for $X \in \mathbb{R}^2$, a smaller prior variance is preferred. D) shows that generally, large numbers of optimizer steps yield better results.

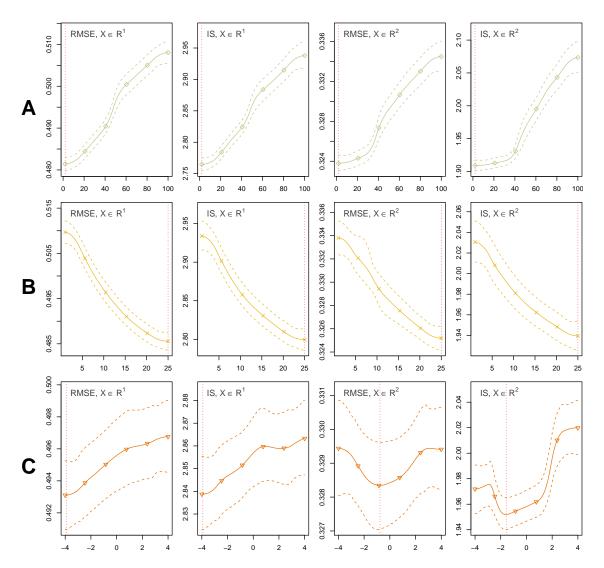


Figure 4: Similar to Fig. 3 but for hyperparameters A) neural network features, B) samples for integration, and C) prior mean μ_0 when KL divergence is used for variational inference. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that smaller numbers of neural network features were preferred in these analyses, while B) indicates that more samples for integration were beneficial. C) shows a differential effect across data generating mechanisms, with different prior means preferred in each case, though the uncertainty is relatively high.

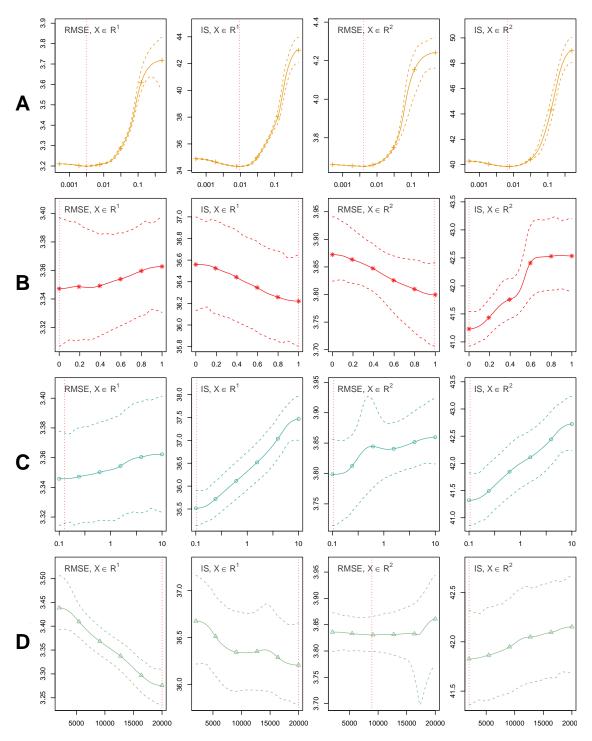


Figure 5: Similar to Fig. 3, but for α -Renyi divergence. The rows correspond to hyperparameters A) $\log_{10}(\text{Learning Rate})$, B) α parameter, C) prior variance σ_0^2 , and D) optimizer steps. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that too large a learning rate leads to worse performance, but the metrics are less sensitive to a range of smaller learning rates. B) shows that the α parameter generally has weak effects with high uncertainty, though it does appear that for $X \in \mathbb{R}^2$, α near 1 is preferred for RMSE while α near 0 is preferred for IS. C) shows that while in some casts, uncertainty is high, generally smaller prior variances are preferable. D) shows that for $X \in \mathbb{R}^1$, more optimizer steps are better, but it seems performance decreases with more optimizer steps for $X \in \mathbb{R}^2$.

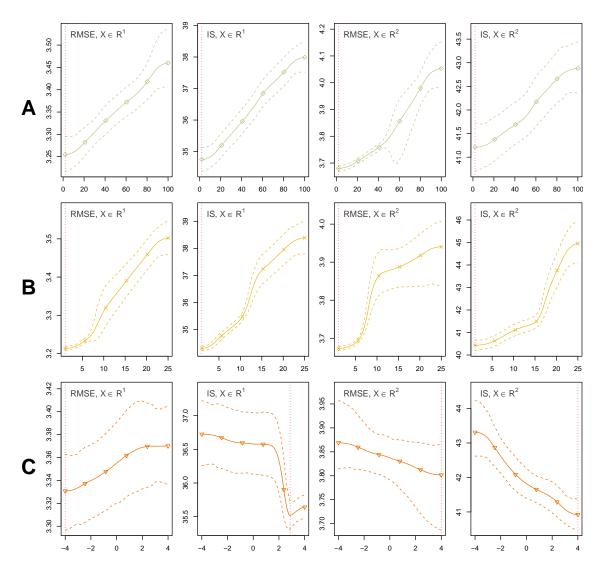


Figure 6: Similar to Fig. 3, but for α -Renyi divergence. The rows correspond to hyperparameters A) neural network features, B) samples for integration, and C) prior mean μ_0 . The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that smaller numbers of neural network features are preferred while B) indicates that smaller numbers of samples for integration are preferred. Similar to KL, the results for prior mean shown in C) are less clear, but with some evidence that results differ for the two data generating mechanisms despite higher uncertainty.

This could potentially indicate overfitting with too many optimizer steps, as it appears that RMSE and IS both increase somewhat with more iterations, suggesting that as is usual practice for neural network fitting, assessing convergence by monitoring training and validation loss curves is essential.

Neural network features In our experiments, smaller numbers of neural network features were preferred across both metrics, both data generating mechanisms, and both loss functions (KL divergence, Fig. 4.A; α -Renyi, Fig. 6.A). This could be due to the relatively simple data generating mechanisms we used, suggesting that large numbers of features are not necessary and could lead to overfitting.

Samples for integration Both loss functions depend on sampling to compute expected values that are not analytically tractable; generally speaking, increasing the number of samples used to compute these values should lower the variance of the estimate and thereby introduce less noise into the stochastic gradient descent algorithm. Generally, a larger number of samples is not used due to diminishing returns for increased computational cost, but our results do show that KL divergence does benefit from a larger number of samples (Fig. 4.B). However, this is decidedly not the case for α -Renyi divergence (Fig. 6.B), suggesting that perhaps this loss function benefits from extra stochasticity during training to locate good local optima.

Prior mean Typically, neural network weights are initialized using distributions symmetric around 0, and it is not common to see BNN implementations use anything but a zero-mean prior on the weights. However, our study does find that nonzero prior mean could be beneficial; for KL divergence (Fig. 4.C), nonzero values do appear to work well in general, though there is a fair amount of uncertainty in the effect. Similar results are observed for α -Renyi (Fig. 6.C). It is worth noting that our data is standardized to be positive (in the range [0,1]) and we use tanh activation functions, suggesting that negative weights may be particularly useful for attaining outputs in the full range of the activation function. This could potentially explain why in KL divergence, negative prior means are generally preferred, though this pattern does not hold for α -Renyi. In general, it is difficult to interpret the sign of the weights and we expect that results could differ significantly for different data generating mechanisms or neural network architectures, but it is worth nothing that perhaps the default prior mean of zero is not always ideal.

4 Discussion

Our sensitivity analysis results quantify what we have experienced, what peers have expressed, and what can be found in the literature: successful VI for BNNs is not as straightforward as gradient-based inference for standard NNs, and common practice is more like an art than a science [9]. In standard non-Bayesian NN training, the primary hyperparameters are neural network architecture (here parameterized by number of features), number of optimizer steps, and learning rate, and our results suggest that all of these are still very important for BNNs. However, there are many additional choices that must be made for BNNs (choice of statistical divergence, divergence-specific hyperparameters such as γ or α , prior hyperparameters, and number of samples for Monte Carlo integration). Furthermore, BNNs must be assessed by not only their predictive accuracy (e.g., RMSE), but also by the quality of their uncertainty intervals (which we measure here using IS).

Our results show that most of the variability in RMSE and IS is explained through interactions between the various hyperparameters, making it difficult to give simple prescriptions for BNN success. While our current results cannot elucidate the exact nature of the second- and higher-order interactions, we have some intuition about possible pairwise interactions to consider in future work. For example, in KL divergence, we expect an interaction between γ and the prior variance σ_0^2 , as the KL divergence between the prior and the variational posterior will be affected by σ_0 and this term in the loss function is then weighted by γ . We also expect there to be an interaction between learning rate and number of optimizer steps needed for convergence, as in standard NN training. In addition, the prior variance could interact with the complexity of the network architecture; for a given Gaussian prior with variance σ_0^2 on each weight and an input comprised of all ones, the output of the layer (pre-activation) will have variance $n\sigma_0^2$ where n is the number of weights, so variance downscaling may be beneficial as the number of weights increases. We plan to further probe these and other possible interactions in future work.

In terms of first-order effects, KL divergence consistently had the largest sum of first order indices for both data generating mechanisms and performance metrics when compared to using α -Renyi divergence. Results will vary depending on the data generating mechanism, training data, and the uncertainty distributions used in the sensitivity analysis, but these results indicate that a *BNN artisan* can effect change on the quality of their BNN fit (for better or worse) when adjusting one parameter at a time more readily when using KL divergence than when using α -Renyi divergence. When interpreting any values from the main effects, we have to consider the fact that the hyperparameters corresponding to minima are the computed when averaging over all other hyperparameters, so the minima identified across the set of hyperparameters may not actually correspond to the best performing model. For example, there will be specific tuning parameter settings where the minimum main effect learning rate does not produce the minimum RMSE or IS. However, these values, their order of magnitude, and their consistency (or lack there of) between data generating mechanisms and performance metrics can provide a starting point for fine tuning.

When comparing the tuning parameters individually, $\log_{10}(LR)$ stands out as having the largest influence on all results with the highest first order and total sensitivity indices. This is a quantification of what is anecdotally known: "Learning rate is important.". The first order sensitivity index for $\log_{10}(LR)$ makes up roughly half of the sum of the first order sensitivity indices over all parameters for a given data generating mechanism and performance metric. The main effect plots (Figs. 3 and 5) show relatively low uncertainty of the main effect, which could be attributed to the more clear and independent mapping of learning rate to an observed RMSE and IS. The minimum main effect of $\log_{10}(LR)$ appears to be similar when comparing all data generating mechanisms and performance metrics, given a statistical distance. When comparing the main effect function between RMSE and IS, the trend appears to be similar, with slight differences around the minimum value and between the boundaries of an individual plot. These results suggest that even if other hyperparameters are set using heuristics, tuning the learning rate is essential for attaining good BNN performance.

When using KL divergence for variational inference, our main effect results show the average RMSE or IS is minimized for the smallest choice of γ we considered, equal to 0.1. When evaluating our loss function in Eq. 2, γ downweights the influence of the prior density for the neural network weights. Downweighting the prior allows the expectation of the likelihood with respect to the variational posterior to have a larger influence, and allows the variational posterior to move further away from an almost certainly misspecified prior. Further research could investigate if there is similar consistency for optimal values of γ for more complicated data generating mechanisms, as well as if even smaller values of γ are on average optimal. We were surprised to see little clear trend for the main effect of choosing α when using α -Renyi divergence for estimating the variational posterior. The literature points to a relation between specific values of α and differing statistical distances, including KL divergence for $\alpha \to 1$ and Hellinger distance for $\alpha = 0.5$ [12]. The expected main effects have little variation, reflected in the first order indices, and relatively large uncertainties. We do not claim that the choice of α is irrelevant, as these results are affected by the hyperrectangle of parameter choices considered for the analysis, but there is no clear choice of α from these results. Interestingly, in the case of the $X \in \mathbb{R}^2$ data generating mechanism, the two metrics (RMSE and IS) appeared to prefer very different α values, suggesting that some choices of α may be better for predictive accuracy while others may be better for uncertainty quantification. As has been noted in previous works, α -Renyi divergence may be more robust to prior misspecification when $\alpha \neq 1$ [29]. Indeed, prior hyperparameters are difficult to select using a priori knowledge [23], and our results show that performance across divergences and data generating mechanisms is affected by prior variance (and to a lesser degree prior mean), so other hyperparameters that minimize the effect of a misspecified prior may be important to tune. Given the results presented here, we make the following suggestions for users of BNNs.

- Systematic tuning of the hyperparameters for BNNs is essential, regardless of the form of the divergence used in the loss function. While grid search is simple to implement, more advanced methods such as randomized search or Bayesian optimization [41, 42] can reduce the computational cost of tuning.
- With a very large computational budget, producing main effect plots as part of fitting a BNN can help diagnose which tuning parameters matter, and which ranges of tuning parameters to consider. Sensitivity analysis has previously been used to reduce the dimensionality of an optimization problem [43]. However, incorporating a sensitivity analysis into every BNN fit is computationally expensive, especially for more complex and higher dimensional data.

- Users should keep in mind that multiple metrics (including RMSE and IS, among others) may be helpful for tuning as they may not always agree on the best hyperparameter settings.
- Standard hyperparameters for neural networks, including learning rate, number of optimizer steps, and architecture are important to tune for BNNs, with learning rate showing the largest first order and total effects, so these should be prioritized if there is limited time to explore all hyperparameters, but the user should be aware that BNN performance is also sensitive to complex interactions between other settings (including prior hyperparameters).
- Prior hyperparameters are difficult to specify in a principled manner, so instead focusing on tuning other hyperparameters that may minimize the impact of a misspecified prior (such as γ or α) may be a suitable approach.

In summary, our global sensitivity analysis reveals that careful and targeted hyperparameter tuning is crucial for optimizing BNN performance. While some hyperparameters like the learning rate consistently exhibit strong influence on RMSE and interval score, others interact in subtle ways that can significantly affect predictive accuracy and UQ. Therefore, leveraging sensitivity analysis or related surrogate-based methods such as Bayesian optimization to guide the tuning process can offer valuable insights into model behavior. Ultimately, balancing principled tuning with practical constraints may enable more robust and valuable usage BNNs in applications where predictive uncertainty is key.

5 Acknowledgements

This manuscript has been authored with number LA-UR-25-26090 by Triad National Security under Contract with the U.S. Department of Energy. Research presented in this article was supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20230469ECR.

Code utilized in this research was developed with the help of chatGPT GPT-40 to improve productivity and aid in debugging.

6 Declaration of Interest Statement

There are no competing interests to declare.

7 References

- [1] Kevin Tran, Willie Neiswanger, Junwoong Yoon, Qingyang Zhang, Eric Xing, and Zachary W Ulissi. Methods for comparing uncertainty quantifications for material property predictions. *Machine Learning: Science and Technology*, 1(2):025006, 2020.
- [2] Thomas Vandal, Evan Kodra, Jennifer Dy, Sangram Ganguly, Ramakrishna Nemani, and Auroop R Ganguly. Quantifying uncertainty in discrete-continuous and skewed data with bayesian deep learning. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2377–2386, 2018.
- [3] Nicholas Geneva and Nicholas Zabaras. Quantifying model form uncertainty in reynolds-averaged turbulence models with bayesian deep neural networks. *Journal of Computational Physics*, 383:125–147, 2019.
- [4] Natalie Klein, Mark Hinds, Scott Koermer, and Michael Geyer. Beyond accuracy: evaluating Bayes neural networks in a real-world application. 2025.

- [5] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.
- [6] David JC MacKay. A practical Bayesian framework for backpropagation networks. Neural Computation, 4(3):448–472, 1992.
- [7] Radford M Neal. Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical report, Citeseer, 1992.
- [8] Peter D Hoff. A first course in Bayesian statistical methods, volume 580. Springer, 2009.
- [9] Varun Godbole, George E. Dahl, Justin Gilmer, Christopher J. Shallue, and Zachary Nado. Deep learning tuning playbook, 2023. Version 1.
- [10] 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.
- [11] Alex Graves. Practical variational inference for neural networks. Advances in Neural Information Processing Systems, 24, 2011.
- [12] Yingzhen Li and Richard E Turner. Rényi divergence variational inference. Advances in Neural Information Processing Systems, 29, 2016.
- [13] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.
- [14] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. Advances in Neural Information Processing Systems, 30, 2017.
- [15] Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux-effortless Bayesian deep learning. Advances in Neural Information Processing Systems, 34:20089–20103, 2021.
- [16] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. Advances in Neural Information Processing Systems, 32, 2019.
- [17] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869. PMLR, 2015.
- [18] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In *International Conference on Machine Learning*, pages 2218–2227. PMLR, 2017.
- [19] Christos Louizos and Max Welling. Structured and efficient variational deep learning with matrix Gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716. PMLR, 2016.
- [20] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano T. Global sensitivity analysis: the primer. *Ist ed., John Wiley & Sons, The Atrium, Southern Gate, Chichester, England*, 2008.
- [21] Ethan Goan and Clinton Fookes. Bayesian neural networks: An introduction and survey. Case Studies in Applied Bayesian Data Science: CIRM Jean-Morlet Chair, Fall 2018, pages 45–87, 2020.
- [22] Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-on Bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2):29–48, 2022.

- [23] Vincent Fortuin, Adrià Garriga-Alonso, Florian Wenzel, Gunnar Ratsch, Richard E Turner, Mark van der Wilk, and Laurence Aitchison. Bayesian neural network priors revisited. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2021.
- [24] Ba-Hien Tran, Simone Rossi, Dimitrios Milios, and Maurizio Filippone. All you need is a good functional prior for Bayesian deep learning. *Journal of Machine Learning Research*, 23(74):1–56, 2022.
- [25] Christian P. Robert. *Monte Carlo statistical methods*. Springer texts in statistics. Springer, New York, 2004.
- [26] Theodore Papamarkou, Jacob Hinkle, M Todd Young, and David Womble. Challenges in Markov chain Monte Carlo for Bayesian neural networks. *Statistical Science*, 37(3):425–442, 2022.
- [27] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.
- [28] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.
- [29] Jeremias Knoblauch, Jack Jewson, and Theodoros Damoulas. An optimization-centric view on Bayes' rule: Reviewing and generalizing variational inference. *Journal of Machine Learning Research*, 23(132):1–109, 2022.
- [30] Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the Bayes posterior in deep neural networks really? In Proceedings of the 37th International Conference on Machine Learning, pages 10248–10259, 2020.
- [31] Rob J Hyndman. Computing and graphing highest density regions. The American Statistician, 50(2):120–126, 1996.
- [32] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. Journal of the American statistical Association, 102(477):359–378, 2007.
- [33] Andrea Saltelli and Stefano Tarantola. On the relative importance of input factors in mathematical models: safety assessment for nuclear waste disposal. *Journal of the American Statistical Association*, 97(459):702–709, 2002.
- [34] Robert B Gramacy and Herbert K H Lee. Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, 2008.
- [35] Robert B Gramacy and Matthew Alan Taddy. Categorical inputs, sensitivity analysis, optimization and importance tempering with tgp version 2, an R package for treed Gaussian process models. *Journal of Statistical Software*, 33:1–48, 2010.
- [36] Robert B. Gramacy and Matthew Taddy. Categorical inputs, sensitivity analysis, optimization and importance tempering with tgp version 2, an R package for treed gaussian process models. *Journal of Statistical Software*, 33(6):1–48, 2010.
- [37] Sungyoon Lee, Hoki Kim, and Jaewook Lee. Graddiv: Adversarial robustness of randomized neural networks via gradient diversity regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- [38] Michael D McKay, Richard J Beckman, and William J Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000.
- [39] Natalie Klein, Giosue Migliorini, Thomas Winckelman, and Scott Koermer. Multiverse: Bayesian neural networks. https://github.com/lanl/multiverse, September 2023.

- [40] A Paszke. Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703, 2019.
- [41] Matthias Schonlau. Computer experiments and global optimization. 1997.
- [42] Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast bayesian optimization of machine learning hyperparameters on large datasets. In Artificial Intelligence and Statistics, pages 528–536. PMLR, 2017.
- [43] Matthew A Taddy, Herbert KH Lee, Genetha A Gray, and Joshua D Griffin. Bayesian guided pattern search for robust local optimization. *Technometrics*, 51(4):389–401, 2009.

A Parameters of Best and Worst BNN Fits

Table A1: Parameter settings for the best and worst observed fits from the $X \in \mathbb{R}^1$ data generating mechanism. All numeric values are displayed on the natural scale to make reading easier.

Method	Category	RMSE	IS	γ	α	σ_0	Optimizer Steps	NN Features	Samples for Integration	LR	μ_0
KL	Best IS	0.20	0.81	0.10	NA	0.71	10821	6	5	0.0226	-0.97
KL	Worst IS	1.39	30.31	4.36	NA	1.70	2611	97	6	0.0006	-1.25
KL	Best RMSE	0.19	0.83	0.16	NA	1.67	17030	61	6	0.0075	-0.08
KL	Worst RMSE	2.56	6.52	6.65	NA	1.27	13869	99	13	0.2489	0.57
α -Renyi	Best IS	0.22	0.88	NA	0.87	1.22	16744	4	16	0.0075	-0.13
α -Renyi	Worst IS	26.72	1570.32	NA	0.30	1.10	3778	65	21	0.4787	-1.39
α -Renyi	Best RMSE	0.20	1.12	NA	0.39	2.33	16455	29	2	0.1559	-0.36
α -Renyi	Worst RMSE	136.61	612.38	NA	0.31	0.77	2748	42	15	0.4925	1.10

B Tables with captions on individual pages

Table 1: Bounds for resulting goodness of fit data collection as well as integration for global sensitivity analysis. For a given statistical distance (KL or α -Renyi), the bounds shown are used with both the $X \in \mathbb{R}^1$ and $X \in \mathbb{R}^2$ data generating mechanisms.

	$\log_{10}(\gamma)$	α	$\log_{10}(\sigma_0)$	Optimizer Steps	NN Features	Samples for Integration	$\log_{10}(\mathrm{LR})$	μ_0
KL Parameters	8							
Upper Bound	1	NA	0.5	20000	100	25	-0.3	2
Lower Bound	-1	NA	-0.5	2000	2	1	-3.3	-2
α -Renyi Paran	neters							
Upper Bound	NA	1	0.5	20000	100	25	-0.3	2
Lower Bound	NA	0	-0.5	2000	2	1	-3.3	-2

Table 2: First order and total sensitivity indices, displayed as first order (total), for each tuning parameter when KL divergence is used for variational inference; top section corresponds to the RMSE metric sensitivity while bottom section corresponds to the interval score (IS) metric sensitivity, with results shown for both data generating mechanisms ($X \in \mathbb{R}^1$ and $X \in \mathbb{R}^2$). It appears learning rate (LR) has the largest first order and total effects. First order effects are generally small for the other hyperparameters, but each has much larger total effects, indicating there are interactions between parameters affecting the outcome.

	$\log_{10}(\gamma)$	$\log_{10}(\sigma_0)$	Optimizer Steps	NN Features	Samples for Integration	$\log_{10}(LR)$	μ_0
KL, RMS	${f E}$						
$X \in \mathbb{R}^1$	0.09(0.39)	0.01(0.36)	0.08(0.42)	0.04(0.36)	0.03(0.30)	0.23(0.69)	0.01(0.26)
$X \in \mathbb{R}^2$	0.01(0.46)	0.05(0.61)	0.04(0.56)	0.03(0.51)	0.02(0.46)	0.13(0.81)	0.00(0.42)
KL, IS							
$X \in \mathbb{R}^1$	0.04(0.29)	0.03(0.47)	0.06(0.45)	0.04(0.35)	0.02(0.27)	0.19(0.74)	0.01(0.25)
$X \in \mathbb{R}^2$	0.01(0.18)	0.04(0.40)	0.05(0.41)	0.04(0.44)	0.01(0.23)	0.12(0.76)	0.01(0.37)

Table 3: First order and total sensitivity indices, displayed as first order (total), for each tuning parameter when α -Renyi divergence is used for variational inference; same format as Table 2 which gave results for KL divergence. Similar to the KL divergence results, learning rate (LR) appears to have the highest first order and total sensitivity indices, with other hyperparameters having small first order but in some cases fairly large total sensitivity, indicating interactions between variables affecting the outcome.

α	$\log_{10}(\sigma_0)$	Optimizer Steps	NN Features	Samples for Integration	$\log_{10}(LR)$	μ_0
α -Renyi, RMSE						
$X \in \mathbb{R}^1 0.00(0.30)$	0.00(0.30)	0.01(0.50)	0.01(0.53)	0.03(0.51)	0.12(0.94)	0.01(0.32)
$X \in \mathbb{R}^2 0.01(0.58)$	0.01(0.57)	0.01(0.71)	0.03(0.78)	0.02(0.69)	0.09(0.88)	0.01(0.62)
α -Renyi, IS						
$X \in \mathbb{R}^1$ 0.00(0.65)	0.01(0.67)	0.00(0.65)	0.02(0.69)	0.03(0.79)	0.13(0.93)	0.01(0.69)
$X \in \mathbb{R}^2 0.01(0.67)$	0.01(0.47)	0.00(0.47)	0.01(0.54)	0.03(0.76)	0.12(0.89)	0.01(0.60)

C Figures

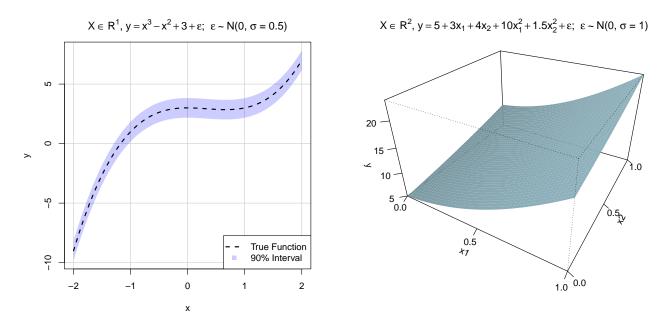


Figure 1: The data generating mechanisms used for testing the variational BNN approximations. For the $X \in \mathbb{R}^1$ data generating mechanism (left) the expected value of the function is plotted along with the 90% conditional distribution interval arising from additive homoskedastic noise. For the $X \in \mathbb{R}^2$ data generating mechanism (right), we show the expected value of the function with respect to the two input dimensions.

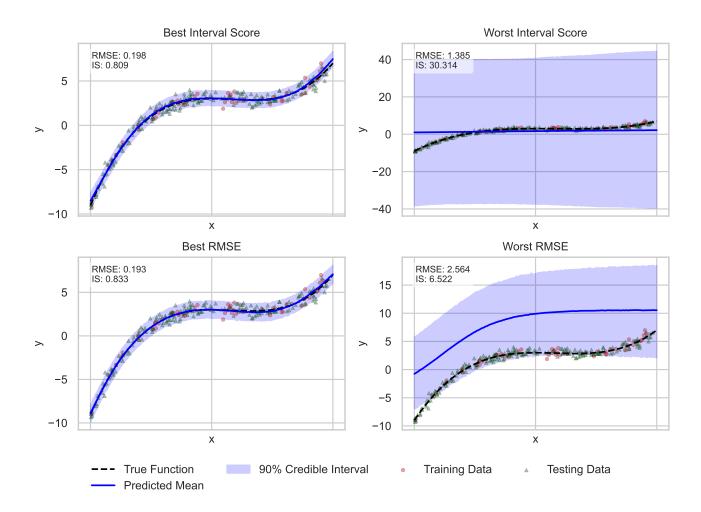


Figure 2: BNN variational approximations through minimizing KL divergence. The best and worst resulting fits from the 750 different initializations used to collect data are shown. The best fits were selected as either the minimum observed RMSE or IS, and the worst were selected as either the largest observed RMSE or IS. The best fits according to interval score and RMSE are qualitatively similar and appear to describe the data distribution accurately. The worst fit according to interval score appears flat with very high variance, while the worst fit according to RMSE incurs high bias and moderately high variance.

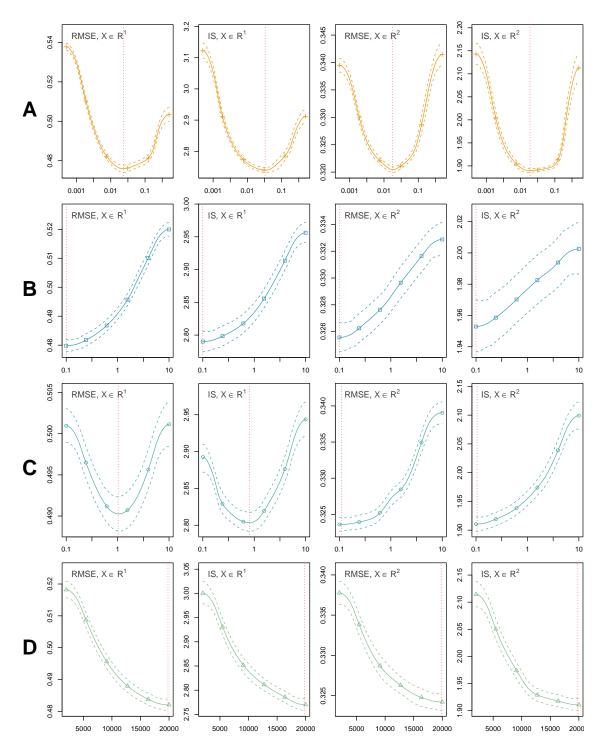


Figure 3: Main effects on RMSE and IS when KL divergence is used for variational inference; each row contains results for each combination of metric (RMSE and IS) and data generating mechanism $(X \in \mathbb{R}^1)$ or $X \in \mathbb{R}^2$). Each row corresponds to a different hyperparameter setting: A) $\log_{10}(\text{Learning Rate})$, B) KL multiplier γ , C) prior variance σ_0^2 , and D) optimizer steps. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color. The minimum of the main effect is shown as a vertical red dashed line. A) indicates that there is an ideal learning rate and that both RMSE and IS increase for smaller or larger learning rates. B) indicates that small KL multipliers result in the lowest RMSE and IS. C) shows that for $X \in \mathbb{R}^1$, prior variance near 1 is ideal, while for $X \in \mathbb{R}^2$, a smaller prior variance is preferred. D) shows that generally, large numbers of optimizer steps yield better results.

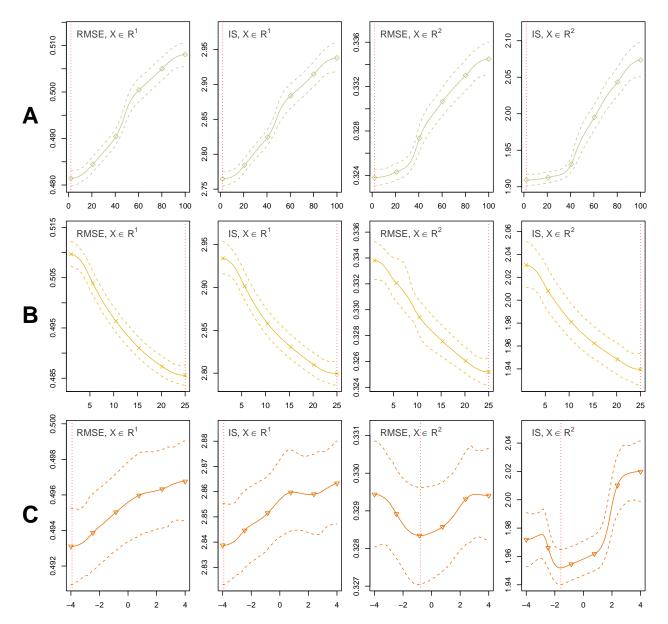


Figure 4: Similar to Fig. 3 but for hyperparameters A) neural network features, B) samples for integration, and C) prior mean μ_0 when KL divergence is used for variational inference. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that smaller numbers of neural network features were preferred in these analyses, while B) indicates that more samples for integration were beneficial. C) shows a differential effect across data generating mechanisms, with different prior means preferred in each case, though the uncertainty is relatively high.

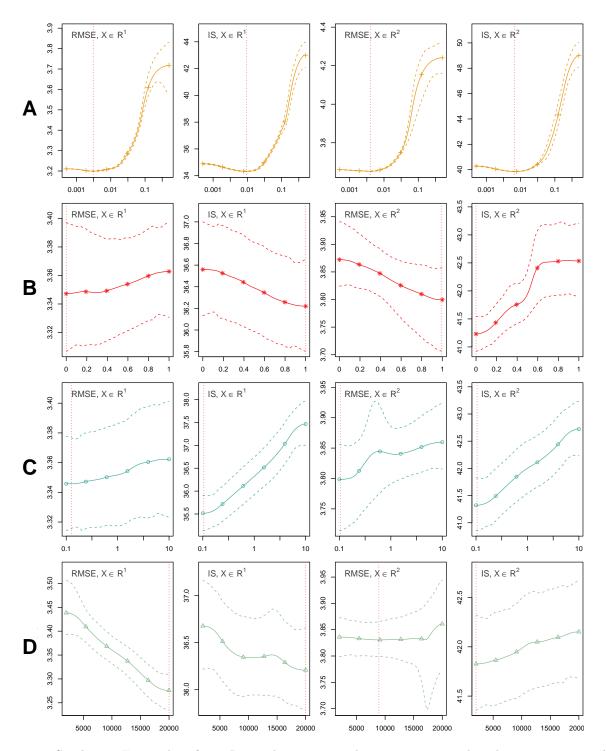


Figure 5: Similar to Fig. 3, but for α -Renyi divergence. The rows correspond to hyperparameters A) $\log_{10}(\text{Learning Rate})$, B) α parameter, C) prior variance σ_0^2 , and D) optimizer steps. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that too large a learning rate leads to worse performance, but the metrics are less sensitive to a range of smaller learning rates. B) shows that the α parameter generally has weak effects with high uncertainty, though it does appear that for $X \in \mathbb{R}^2$, α near 1 is preferred for RMSE while α near 0 is preferred for IS. C) shows that while in some casts, uncertainty is high, generally smaller prior variances are preferable. D) shows that for $X \in \mathbb{R}^1$, more optimizer steps are better, but it seems performance decreases with more optimizer steps for $X \in \mathbb{R}^2$.

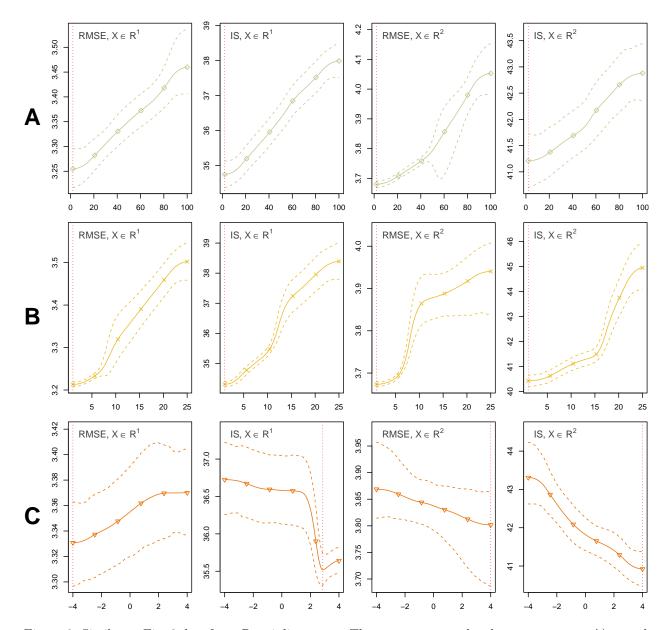


Figure 6: Similar to Fig. 3, but for α -Renyi divergence. The rows correspond to hyperparameters A) neural network features, B) samples for integration, and C) prior mean μ_0 . The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that smaller numbers of neural network features are preferred while B) indicates that smaller numbers of samples for integration are preferred. Similar to KL, the results for prior mean shown in C) are less clear, but with some evidence that results differ for the two data generating mechanisms despite higher uncertainty.

D Figure Captions as a list

List of figure captions:

- 1. The data generating mechanisms used for testing the variational BNN approximations. For the $X \in \mathbb{R}^1$ data generating mechanism (left) the expected value of the function is plotted along with the 90% conditional distribution interval arising from additive homoskedastic noise. For the $X \in \mathbb{R}^2$ data generating mechanism (right), we show the expected value of the function with respect to the two input dimensions.
- 2. BNN variational approximations through minimizing KL divergence. The best and worst resulting fits from the 750 different initializations used to collect data are shown. The best fits were selected as either the minimum observed RMSE or IS, and the worst were selected as either the largest observed RMSE or IS. The best fits according to interval score and RMSE are qualitatively similar and appear to describe the data distribution accurately. The worst fit according to interval score appears flat with very high variance, while the worst fit according to RMSE incurs high bias and moderately high variance.
- 3. Main effects on RMSE and IS when KL divergence is used for variational inference; each row contains results for each combination of metric (RMSE and IS) and data generating mechanism $(X \in \mathbb{R}^1)$ or $X \in \mathbb{R}^2$. Each row corresponds to a different hyperparameter setting: A) $\log_{10}(\text{Learning Rate})$, B) KL multiplier γ , C) prior variance σ_0^2 , and D) optimizer steps. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color. The minimum of the main effect is shown as a vertical red dashed line. A) indicates that there is an ideal learning rate and that both RMSE and IS increase for smaller or larger learning rates. B) indicates that small KL multipliers result in the lowest RMSE and IS. C) shows that for $X \in \mathbb{R}^1$, prior variance near 1 is ideal, while for $X \in \mathbb{R}^2$, a smaller prior variance is preferred. D) shows that generally, large numbers of optimizer steps yield better results.
- 4. Similar to Fig. 3 but for hyperparameters A) neural network features, B) samples for integration, and C) prior mean μ_0 when KL divergence is used for variational inference. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that smaller numbers of neural network features were preferred in these analyses, while B) indicates that more samples for integration were beneficial. C) shows a differential effect across data generating mechanisms, with different prior means preferred in each case, though the uncertainty is relatively high.
- 5. Similar to Fig. 3, but for α -Renyi divergence. The rows correspond to hyperparameters A) $\log_{10}(\text{Learning Rate})$, B) α parameter, C) prior variance σ_0^2 , and D) optimizer steps. The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that too large a learning rate leads to worse performance, but the metrics are less sensitive to a range of smaller learning rates. B) shows that the α parameter generally has weak effects with high uncertainty, though it does appear that for $X \in \mathbb{R}^2$, α near 1 is preferred for RMSE while α near 0 is preferred for IS. C) shows that while in some casts, uncertainty is high, generally smaller prior variances are preferable. D) shows that for $X \in \mathbb{R}^1$, more optimizer steps are better, but it seems performance decreases with more optimizer steps for $X \in \mathbb{R}^2$.
- 6. Similar to Fig. 3, but for α -Renyi divergence. The rows correspond to hyperparameters A) neural network features, B) samples for integration, and C) prior mean μ_0 . The expected main effect is shown as the solid line, and main effect uncertainty related to the surrogate model is shown as dashed lines of

the same color with the minimum of the main effect is shown as a vertical red dashed line. A) indicates that smaller numbers of neural network features are preferred while B) indicates that smaller numbers of samples for integration are preferred. Similar to KL, the results for prior mean shown in C) are less clear, but with some evidence that results differ for the two data generating mechanisms despite higher uncertainty.