AMORTISING INFERENCE AND META-LEARNING PRIORS IN NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

One of the core facets of Bayesianism is in the updating of prior beliefs in light of new evidence—so how can we maintain a Bayesian approach if we have no prior beliefs in the first place? This is one of the central challenges in the field of Bayesian deep learning, where there is no clear way to translate beliefs about a prediction task into prior distributions over model parameters. Bridging the fields of Bayesian deep learning and neural processes, we propose to *meta-learn* our parametric prior from data by introducing a way to perform per-dataset amortised variational inference. The model we develop can be viewed as a neural process whose latent variable is the set of weights of a BNN and whose decoder is the neural network parameterised by a sample of the latent variable itself. This unique model allows us to study the behaviour of Bayesian neural networks under well-specified priors, use Bayesian neural networks as flexible generative models, and perform desirable but previously elusive feats in neural processes such as within-task minibatching or meta-learning under extreme data-starvation.

1 INTRODUCTION

While the ability to learn hierarchical representations of data has allowed neural networks to boast phenomenal predictive performance across many domains, enabling them to accurately estimate their predictive uncertainty remains generally unsolved. Bayesian deep learning (MacKay, 1992) promises a theoretically sound approach for endowing representation learners with uncertainty quantification, but there are many difficulties with the approach in practice. Of all of them, one of the most sleep-depriving is the question of how to choose appropriate priors. Neural network weights lack interpretability, meaning it is fiendishly difficult to elicit priors over them that are sensible in prediction space (Fortuin, 2022). The convenience priors that we generally use are known to reduce large[1] Bayesian neural networks (BNNs) to the "simple smoothing devices" (MacKay, 1998) that are Gaussian processes (GPs; Neal, 1995; Matthews et al., 2018; Yang, 2019), meaning that in trying to achieve uncertainty quantification we inadvertently destroy the ability to learn hierarchical representations (Aitchison, 2020). An increasingly popular approach to specifying priors in BNNs is to use function-space priors (Flam-Shepherd et al., 2017; Sun et al., 2019; Cinquin et al., 2024). However, these priors are most often chosen to be Gaussian process priors which, yet again, reduce Bayesian deep learning to approximate GP inference. Even when priors and architectures are chosen specifically to avoid GP behaviour, the resulting stochastic process prior is not just poorly understood, but generally a bad model of the real-world data-generating process.

On the other hand, neural processes (NPs; Garnelo et al., 2018a;b) use the shared structure between related tasks to meta-learn a free-form stochastic process prior. Unlike GPs and BNNs, they do not learn an explicit parametric prior that can be evaluated or sampled from, but, given some context observations, they learn to map directly to the stochastic process posterior corresponding to the learned implicit prior process. For a sufficiently flexible NP architecture and with enough data, they can model the ground-truth data-generating process (Foong et al., 2020). Given this remarkable ability, neural processes have enabled practitioners to *endow representation learners with uncertainty quantification* to great success across a range of domains including weather and climate applications (Allen et al., 2025; Ashman et al., 2024b; Andersson et al., 2023), causal machine learning (Dhir

[1] In the limit of infinite architecture width, depth, or size in some other sense.

et al., 2025), Bayesian optimisation (Maraval et al., 2023; Volpp et al., 2023), and cosmological applications (Park and Choi, 2021; Pondaven et al., 2022).

Drawing inspiration from neural processes, we are interested in meta-learning BNN priors that encode the shared structure across a related set of tasks. In other words, *can we use meta-learning to design well-specified priors in Bayesian deep learning?* To that end, **we devise a scheme for performing per-dataset amortised inference in BNNs**. This results in a neural process whose latent variable is the set of weights of a BNN, and whose decoder is the neural network parameterised by a particular latent variable posterior sample. We refer to our model as the Bayesian deep neural process (BDNP). The unique ability of the BDNP to amortise BNN inference and meta-learn BNN priors **enables us to investigate** such **previously unanswerable questions** as: "*under a well-specified prior, how important is the approximate inference method in Bayesian deep learning really?*". Furthermore, as a new member of the neural process family (Dubois et al., 2020), the BDNP introduces some completely novel capabilities. These include the ability to perform ***within-task*** **minibatching** for scalability to massive context sets, and the ability to **adjust the flexibility of the learned prior** so that overfitting can be avoided in settings for which only a few tasks have been observed.

2 The Bayesian Deep Neural Process

2.1 Layerwise Inference

We start by considering the conditional posterior over the last-layer weights $\mathbf{W}^L \in \mathbb{R}^{d_{L-1} \times d_L}$ in an L-layered multilayer perceptron (MLP), where d_l denotes the number of units in the l-th layer of the network. In this exposition we do not consider biases but they are straightforward to include in practice. Given some data $\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\}$ where $\mathbf{X} \in \mathbb{R}^{n \times d_0}$ and $\mathbf{Y} \in \mathbb{R}^{n \times d_L}$ as well as the weights of the previous layers $\mathbf{W}^{1:L-1} = \{\mathbf{W}^l\}_{l=1}^{L-1}$, and assuming a Gaussian likelihood, the conditional posterior over the last-layer's weights is of the form

$$p(\mathbf{W}^L | \mathbf{W}^{1:L-1}, \mathcal{D}) \propto p(\mathbf{W}^L) \prod_{d=1}^{d_L} \prod_{n=1}^{N} \mathcal{N}\left(y_{n,d}; \phi(\mathbf{x}_n^{L-1})^\top \mathbf{w}_d^L, \sigma_d^2\right) \tag{1}$$

where $y_{n,d}$ is the d-th dimension of the n-th target, $\phi(\cdot)$ is the MLP's elementwise nonlinearity, $\mathbf{x}_n^{L-1}$ denotes the activations of the penultimate layer for the n-th datapoint, $\mathbf{w}_d^L$ is the d-th column of $\mathbf{W}^L$ representing all input weights to unit d in the output layer, and σ_d is the observation noise level for the d-th dimension of the targets. Assuming that the prior $p(\mathbf{W}^L)$ is conjugate, this posterior is available in closed-form via Bayesian linear regression (BLR; Bishop, 2007). Inspired by this result, Ober and Aitchison (2021) propose a variational posterior for BNNs and deep GPs that decomposes into a product of layerwise conditionals

$$q(\mathbf{W}) = \prod_{l=1}^{L} q(\mathbf{W}^l | \mathbf{W}^{1:l-1}) \tag{2}$$

where the layerwise factors are computed via exact inference between the layerwise prior and a set of variationally-parameterised pseudo-likelihood terms. We will adopt a similar approach.

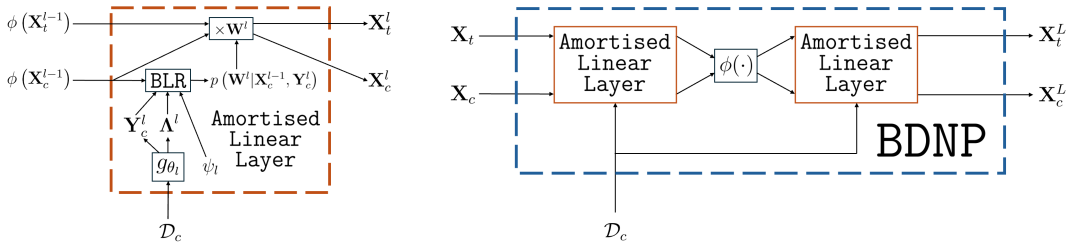
2.2 Amortising Layerwise Inference

We generalise the likelihood of the last-layer weights seen in Eq. (1) to the weights of the l-th layer as follows

$$p(\mathbf{Y}^l | \mathbf{X}^{l-1}, \mathbf{W}^l) = \prod_{d=1}^{d_l} \prod_{n=1}^{N} \underbrace{\mathcal{N}\left(y_{n,d}^l; \phi(\mathbf{x}_n^{l-1})^\top \mathbf{w}_d^l, {\sigma_{n,d}^l}^2\right)}_{t_n(\mathbf{w}_d^l)} \tag{3}$$

where $y_{n,d}^l$ is a pseudo-observation corresponding to the d-th activation of the l-th layer for the n-th datapoint, and $\sigma_{n,d}^l$ is the noise level for the corresponding pseudo-likelihood term. Note that $\mathbf{X}^0 \equiv \mathbf{X}$. These two parameters are obtained by passing the n-th input-output pair[2] through an

[2]Another option is to pass the concatenation of an input-output pair *as well as* the previous layer's activation $\mathbf{X}_n^{l-1}$ into each inference network. This would simplify the inference networks' prediction task, meaning they can be smaller.



(a) Amortised linear layer (b) BDNP

Figure 1: Computational diagrams of the amortised linear layer (a) and a two-layer BDNP (b). We use the context $\cdot_c$ and target $\cdot_t$ notation to distinguish between inputs with labels, on which we condition, and inputs without labels, at which we predict.

inference network g_{θ_l}:

$$\mathbf{y}_n^l,\ \log(\boldsymbol{\sigma}_n^l) = g_{\theta_l}(\mathbf{x}_n, \mathbf{y}_n) \tag{4}$$

where θ_l denotes the parameters of the l-th layer's inference network, which will be optimised during training. In the case of the final layer we can use the actual observations, that is, $\mathbf{Y}^L \equiv \mathbf{Y}$. With the pseudo-likelihood parameters, and assuming unitwise factorised Gaussian priors $p_{\psi_l}(\mathbf{W}^l) = \prod_{d=1}^{d_l} \mathcal{N}\left(\mathbf{w}_d^l; \boldsymbol{\mu}_d^l, \boldsymbol{\Sigma}_d^l\right)$ where $\psi_l = \{\boldsymbol{\mu}_d^l, \boldsymbol{\Sigma}_d^l\}_{d=1}^{d_l}$, the approximate layerwise posteriors are computed in closed form:

$$q(\mathbf{W}^l|\mathbf{W}^{1:l-1}, \mathcal{D}) = p(\mathbf{W}^l|\mathbf{X}^{l-1}, \mathbf{Y}^l) \propto \prod_{d=1}^{d_l} \mathcal{N}\left(\mathbf{w}_d^l; \boldsymbol{\mu}_d^l, \boldsymbol{\Sigma}_d^l\right) \mathcal{N}\left(\mathbf{y}_d^l; \phi(\mathbf{X}^{l-1})^\top \mathbf{w}_d^l, {\boldsymbol{\Lambda}_d^l}^{-1}\right)$$

$$= \prod_{d=1}^{d_l} \mathcal{N}\left(\mathbf{w}_d^l; \mathbf{m}_d^l, \mathbf{S}_d^l\right) \tag{5}$$

where $\boldsymbol{\Lambda}_d^l \in \mathbb{R}^{N\times N}$ is a diagonal precision matrix with the n-th diagonal element given by $\frac{1}{\left(\sigma_{n,d}^l\right)^2}$, and the posterior mean vectors $\mathbf{m}_d^l$ and covariance matrices $\mathbf{S}_d^l$ are given by

$${\mathbf{S}_d^l}^{-1} = {\boldsymbol{\Sigma}_d^l}^{-1} + \phi(\mathbf{X}^{l-1})^\top \boldsymbol{\Lambda}_d^l \phi(\mathbf{X}^{l-1}) \tag{6}$$

$$\boldsymbol{m}_d^l = \mathbf{S}_d^l \left({\boldsymbol{\Sigma}_d^l}^{-1} \boldsymbol{\mu}_d^l + \phi(\mathbf{X}^{l-1})^\top \boldsymbol{\Lambda}_d^l \mathbf{y}_d^l\right). \tag{7}$$

We consider inference under Gaussian priors with different factorisation structures in Appendix B.

This machinery, which we call the *amortised linear layer*, enables inference over the weights of a linear layer that is situated arbitrarily within a neural network architecture, conditional on the weights of the previous layers. By stacking these layers and sampling from each layer's conditional posterior before computing the next layer's posterior, we can perform amortised inference in a BNN via the variational posterior

$$q\left(\mathbf{W}|\mathcal{D}\right) = \prod_{l=1}^{L} q\left(\mathbf{W}^l|\mathbf{W}^{1:l-1}, \mathcal{D}\right) \tag{8}$$

$$= \prod_{l=1}^{L} p\left(\mathbf{W}^l|\mathbf{X}^{l-1}, \mathbf{Y}^l\right). \tag{9}$$

Since such a model may be interpreted as a latent-variable neural process (Garnelo et al., 2018b) in which the latent variable is the set of BNN weights and the decoder is the BNN itself, we refer to this model as the *Bayesian deep neural process* (BDNP). See Fig. 1 for computational diagrams of the amortised linear layer and a BDNP.

2.3 Training

We adopt a variational approach to training, where the parameters to be optimised are the inference network parameters $\Theta = \{\theta_l\}_{l=1}^L$ and the prior parameters $\Psi = \{\psi_l\}_{l=1}^L$. The former take the role of

variational parameters while the latter are model parameters. As is customary in the NP literature, we split datasets into disjoint context $\cdot_c$ and target $\cdot_t$ sets. While there are various training objectives that would be sensible, we motivate our choice by considering what behaviour we would like our training objective to encourage. We summarise these considerations into the following three desiderata:

(I) accurate approximate posteriors,
(II) a prior that faithfully encodes the data-generating process,
(III) and high quality predictions.

Assuming access to a meta-dataset $\Xi = \{\mathcal{D}^{(j)}\}_{j=1}^{|\Xi|}$ of tasks that are related to each other in that they share a data-generating process $\mathcal{D}^{(j)} \sim p(\mathcal{D})$, we propose a novel objective function to train the BDNP with. We call it the posterior-predictive amortised variational inference (PP-AVI) loss:

$$\mathcal{L}_{\text{PP-AVI}}(\Xi) := \frac{1}{|\Xi|} \sum_{j=1}^{|\Xi|} \log q\left(\mathbf{Y}_t^{(j)} | \mathcal{D}_c^{(j)}, \mathbf{X}_t^{(j)}\right) + \mathcal{L}_{\text{ELBO}}\left(\mathcal{D}_c^{(j)}\right) \tag{10}$$

where the first term is the log posterior-predictive density of the target set, and $\mathcal{L}_{\text{ELBO}}(\mathcal{D})$ denotes the usual evidence lower bound (ELBO) used for VI in BNNs; $\mathbb{E}_{q(\mathbf{W}|\mathcal{D})}\left[p(\mathbf{Y}|\mathbf{W}, \mathbf{X})\right] - \text{KL}\left[q(\mathbf{W}|\mathcal{D}) \parallel p(\mathbf{W})\right]$.

Proposition 1. *For $|\Xi| \to \infty$, maximisation of $\mathcal{L}_{\text{PP-AVI}}(\Xi)$ directly targets the three desiderata.*

A proof of Proposition 1, a practical guide to implementing the PP-AVI loss function, as well as a detailed discussion of alternative objective functions are all provided in Appendix A. We emphasise here, though, that $\mathcal{L}_{\text{PP-AVI}}(\Xi)$ can be unbiasedly estimated from a minibatch of tasks $\xi \subset \Xi$ via $\frac{1}{|\xi|} \sum_{j=1}^{|\xi|} \mathcal{L}_{\text{PP-AVI}}(\xi)$, enabling training on huge meta-datasets via stochastic optimisation.

2.4 Within-Task Minibatching via Sequential Bayesian Inference

When making predictions on a particular task, the full set of real and pseudo observations must be stored in memory in order to compute the posterior. So, while stochastic optimisation of the objective function allows us to scale to large *meta*-datasets, scalability to large *context* sets remains elusive. Fortunately however, we provide a solution to this problem too. The high memory requirements can be avoided by iteratively updating each layer's posterior with a minibatch of datapoints via sequential Bayesian inference (Bishop, 2007), temporarily discarding each minibatch after applying its update to a particular layer's conditional posterior.

We partition a task $\mathcal{D}$ into B minibatches $\{\mathcal{D}_b\}_{b=1}^{B}$. We use $\mathbf{W}_{q_b}^l$ to denote a sample from the l-th layer's posterior given only minibatch b, that is, $\mathbf{W}_{q_b}^l \sim q(\mathbf{W}^l | \mathbf{W}^{1:l-1}, \mathcal{D}_b)$. Since computation of each layer's posterior requires samples from the *full-batch* previous layer posteriors (i.e., $\mathbf{W}_{q_{1:B}}^{1:l-1}$), the minibatching procedure must be performed in full for each layer before sampling and proceeding to the next layer. Our minibatched approach to obtaining a BDNP posterior sample is detailed in Algorithm 1. The ability to minibatch a forward pass over a given context set is rare in the context of neural processes, and it is a valuable property as it allows us to scale to tasks with large and high-dimensional datasets without running into memory limitations. Crucially, we note that our minibatching procedure introduces no further approximation error—it results in the exact same approximate posterior as the full-batch procedure.

In addition to complexity analysis and further details regarding our minibatched forward pass algorithm found in Appendix C, in Appendix D we use a similar sequential Bayesian inference trick to devise an online-learning scheme for the BDNP through which predictions for a given task are updated in light of new data.

2.5 Adjusting the Flexibility of the Prior

One limitation shared amongst neural processes is their tendency to overfit when the number of observed tasks is limited (Rochussen and Fortuin, 2025). This happens because the model parameters are responsible for both amortising (predictive) inference *and* encoding a prior, such that there is no

way to regularise the learned prior without also affecting the model's ability to amortise prediction. In the BDNP however, these roles are disentangled into two distinct parameter groups; those of the weights prior, and those of the inference networks. This separation allows us to limit the flexibility of the learned prior *without* affecting the model's ability to perform inference. We do this by fixing the prior over a subset of the weights while optimising the remaining prior parameters. For example, we can fix the prior over the last layer's weights to a zero-centered and unit-variance diagonal Gaussian while optimising the prior parameters for the earlier-layer weights. By varying the number of weights whose prior is fixed, we introduce a new knob with which to tune the flexibility of the learnable prior. Such a scheme enables practitioners to balance between forcing a broad/misspecified prior and learning an overfit prior, leading to better generalisation in the small meta-dataset regime.

2.6 Attention

One way to extend the BDNP would be to go beyond MLPs and incorporate more sophisticated neural architectures, such as the attention mechanism (Vaswani et al., 2017). We outline two ways to do this. As is common in the NP literature, the encoder can be augmented with attention blocks (Kim et al., 2019; Nguyen and Grover, 2022). In the BDNP this means parameterising the inference networks $\{g_{\theta_l}\}_{l=1}^{L}$ as transformers rather than MLPs, and processing the full context set together at each layer. This could improve performance since modelling interactions *between* context points could lead to better pseudo-likelihood parameter estimates than when we process each point independently. However, an attentive encoder incurs an extra computational cost of $\mathcal{O}(n_c^2)$. Furthermore, application of our within-task minibatching scheme under an attentive encoder would lead to different posteriors to the full-batch forward pass due to the new dependencies between context points. Nonetheless, we refer to this variant of the model as the attentive BDNP (AttBDNP).

Alternatively, we can focus on the decoder. The core technology that we introduce is a way to amortise inference over the weights of a linear layer situated arbitrarily within a neural network. Since an attention block is just a composition of linear layers and various nonparametric operations[3], it is possible to amortise inference in an attention block by amortising the inference in each linear layer therein. If such amortised attention blocks are stacked, we end up with a transformer whose weights posterior is obtained in-context. A more sophisticated decoder could enable us to model more complex tasks. However, such a decoder processes the target locations together such that the prediction for a particular target depends on the other target locations. This property destroys the consistency of the model and therefore no longer results in a valid stochastic process. We therefore refer to this version of the model as the *Bayesian deep attentive machine* (BDAM), omitting any reference to stochastic processes from its name. Note that the BDAM also incurs an extra $\mathcal{O}(n_t^2)$ computational cost. For computational diagrams as well as further details on the lack of consistency of the BDAM, including a simple demo, see Appendix E.

3 Related Work

Neural Processes. The BDNP is a new member of the NP family (Garnelo et al., 2018a; Dubois et al., 2020), and in particular the latent-variable NP family (Garnelo et al., 2018b; Singh et al., 2019; Lee et al., 2020; Foong et al., 2020). The AttBDNP is also a member of the transformer NP family (Kim et al., 2019; Nguyen and Grover, 2022; Feng et al., 2023; Ashman et al., 2024a;c). Our model is different to existing latent-variable NPs in that our latent variable is the parameterisation of the decoder itself, and not an abstract representation of the context set that is passed to a globally parameterised decoder. While Rochussen and Fortuin (2025) adopt a similar approach, in our work the decoder is a BNN rather than a sparse GP. Inference in the BDNP can be viewed as an instantiation of Volpp et al. (2021)'s Bayesian context aggregation mechanism—in both settings we have an encoder that maps from individual context points to pseudo-likelihood terms with which the latent variable's posterior is obtained through exact inference. While we choose to exclude the BDAM from the NP family due to its lack of consistency, some existing NP variants also lack consistency; Bruinsma et al. (2023)'s NP variants and Nguyen and Grover (2022)'s TNP-A both fail to produce consistent predictive distributions due to autoregression amongst targets, while Nguyen and Grover (2022)'s

[3]Such as residual connections. We do not consider inference over the `layer-norm` parameters, justifying a deterministic treatment of them through their comparative inability to overfit.

TNP-ND is more similar to the BDAM, with attention performed between targets[4]. Our training and modelling setup is also closely related to that of Gordon et al. (2019), with the first term of our proposed training objective being exactly equivalent to theirs (see Appendix A).

Approximate Inference in BNNs. There have been considerable efforts to develop more accurate and scalable approximate inference methods for BNNs in recent decades. There are Markov-chain Monte-Carlo algorithms (Neal, 1992; Welling and Teh, 2011; Sommer et al., 2024), ensemble and particle approaches (Lakshminarayanan et al., 2017; D' Angelo and Fortuin, 2021; Liu and Wang, 2016), Laplace approximations (MacKay, 1992; Ritter et al., 2018; Immer et al., 2021), variational strategies (Hinton and van Camp, 1993; Graves, 2011; Louizos and Welling, 2017), as well as more bespoke solutions (Maddox et al., 2019; Gal and Ghahramani, 2016; Hernandez-Lobato and Adams, 2015). Our approach is most similar to Ober and Aitchison (2021)'s global inducing point variational posterior since we factorise the variational posterior into layerwise conditionals and compute each one through exact inference under pseudo-observations. Unlike their approach, in each layer we have a pseudo-likelihood corresponding to *every* datapoint rather than for a limited set of inducing locations; we parameterise the pseudo-likelihood terms via inference networks rather than directly; and we use a broader class of Gaussian priors. Another related method is Kurle et al. (2024)'s BALI, which adopts a non-variational approach to parameterising the pseudo-likelihood terms in each layer. Separately, our work shares the use of secondary (inference/"*hyper*") networks with normalising flow-based approaches (Louizos and Welling, 2017) and Bayesian hypernetworks (Krueger et al., 2018), but the difference is in what the secondary networks map from and to—for us it is from observations to pseudo-likelihood terms and for them it is from base distribution samples to posterior samples. Finally, our approximate inference scheme is an instance of structured variational inference (Hoffman and Blei, 2015).

Priors in BNNs. In general, there is no widely accepted way to select well-specified priors in BNNs (see Fortuin, 2022). While (meta-)learning priors is not a new concept (Rasmussen and Williams, 2006; Patacchiola et al., 2020; Fortuin et al., 2020; Rothfuss et al., 2021), it is a relatively under-explored approach in the context of Bayesian deep learning. To this end, Fortuin et al. (2022) analyse the empirical distributions of trained neural network weights to construct new priors, Shwartz-Ziv et al. (2022) use the approximate posterior from a large pre-training task as a learned prior for downstream fine-tuning tasks, and Villecroze et al. (2025) adopt an empirical Bayesian approach to learning a flexible prior over the last-layer weights of a BNN. We also adopt a variational approach, but we train across multiple tasks. Most similar to our approach is that of Rothfuss et al. (2021), who also meta-learn a BNN prior across related tasks. However, they adopt a PAC-Bayes framework (McAllester, 1998) involving naïvely chosen hyper-priors, and they restrict their priors to fully-factorised Gaussians. Furthermore, their posteriors are particle based, so it is costly to meta-train across arbitrarily many tasks (as we can) in order to learn a truly well-specified prior.

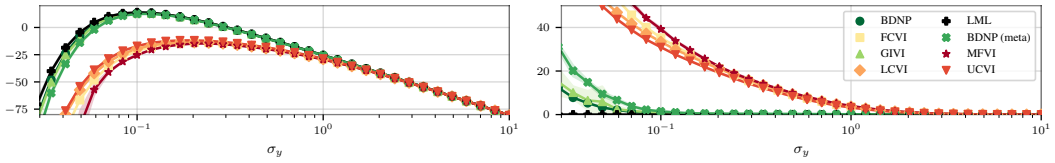
4 Empirical Investigation

4.1 How good is the BDNP's approximate posterior?

We begin by evaluating the quality of approximate inference in the BDNP. We adopt a similar setup to Bui (2021) and, under a common BNN architecture with fixed hyperparameters (including Ψ), we measure the gap between the log marginal likelihood (LML) and the ELBOs achieved by various VI techniques. The difference between these two quantities represents the KL divergence $\mathrm{KL}\big[q(\mathbf{W}|\mathcal{D}) \parallel p_\Psi(\mathbf{W}|\mathcal{D})\big]$, giving us a a clear metric of approximation quality in each case. To eliminate any bias in inference quality due to misspecified priors, we generate a dataset by 1.) sampling a function from the BNN's prior, 2.) uniformly sampling a set of inputs, and 3.) adding Gaussian noise with standard deviation 0.1 to the outputs. Since the data therefore has a high probability of being generated under the prior (because it was), we can estimate the log marginal likelihood via Monte Carlo integration. We compare a BDNP meta-trained across similarly generated datasets under $\mathcal{L}_{\text{PP-AVI}}$, a BDNP trained on just the evaluation task via the standard ELBO objective function, mean-field VI (MFVI; Blundell et al., 2015), Ober and Aitchison (2021)'s global inducing-point VI (GIVI), as well as a number of VI algorithms with increasingly high-rank Gaussian variational posteriors: unitwise correlated (UCVI), layerwise-correlated (LCVI), and fully-correlated

[4]In their case, attention is performed between the target tokens only for the predictive covariance module. The predictive means for each target location remain conditionally independent given the context set.

(FCVI). The results, which are collected across a range of likelihood noise settings and averaged over four repeat runs, are shown in Fig. 2. They demonstrate that the BDNP is capable of very high quality approximate inference. For all methods we see that approximate inference quality decreases with smaller likelihood noises, and since this effect is more pronounced for the unstructured variational approximations (MFVI, *CVI), with those methods we observe the same bias in model selection via the ELBO as in Bui (2021).



(a) $\mathcal{L}_{\text{ELBO}}(\mathcal{D})$ (b) $\text{KL}\big[q(\mathbf{W}|\mathcal{D}) \parallel p_\Psi(\mathbf{W}|\mathcal{D})\big]$

Figure 2: ELBO and KL divergence between approximate and true posteriors for different VI methods. The BDNPs' approximate posteriors are of very high quality.

4.2 Can We Learn Meaningful Priors with the BDNP?

Here we train BDNPs (including Ψ) on meta-datasets with different data-generating processes. We then qualitatively compare functions sampled from the BDNP's learned prior with those from the true data-generating process. We consider synthetic meta-regression datasets generated from random sawtooth functions, Heaviside (/step-)functions, functions from a standard BNN prior[5], as well as the MNIST dataset (LeCun et al., 1989) with random pixel masking cast as a pixelwise meta-regression dataset (Garnelo et al., 2018a), in which case we use the AttnBDNP. The results are visualised in Fig. 3 and Fig. 4.

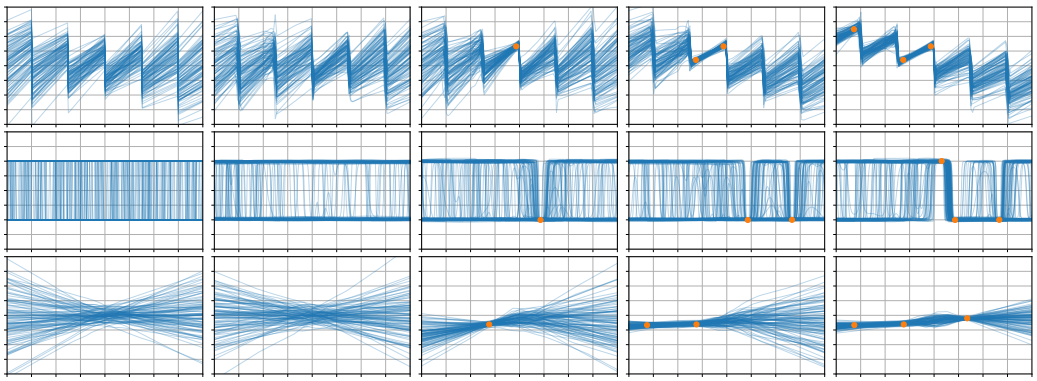


Figure 3: Function samples from the true data-generating process (first column), learned BDNP prior predictive samples (second column), BDNP posterior predictive function samples (last three columns, observations as orange dots). The learned prior predictives are very similar to the true generative processes.

We observe that the BDNP learns functional priors that are almost indistinguishable from the synthetic data-generating processes. The corresponding posterior predictive samples appear to be sensible as well, with increased uncertainty away from observations while maintaining the underlying functional structure in each case. While the AttnBDNP's ability to generate MNIST images might not represent the state-of-the-art, the fact that many of the samples are clearly recognisable digits further demonstrates this section's conclusion, which is that the (Attn)BDNP can model arbitrary stochastic processes by learning well-specified priors.

4.3 Does Approximate Inference Quality Matter Under a Good Prior?

To answer this question, we consider three synthetic and one real-world data-generating processes. In each case, we meta-train a BDNP such that it learns a well-specified prior for the problem. For each of the approximate inference methods that we consider, we then compare predictive performance

[5]Throughout our empirical investigation, we use *standard BNN prior* to refer to zero-centered fully-factorised Gaussian priors with variance scaled inversely proportional to layer width.

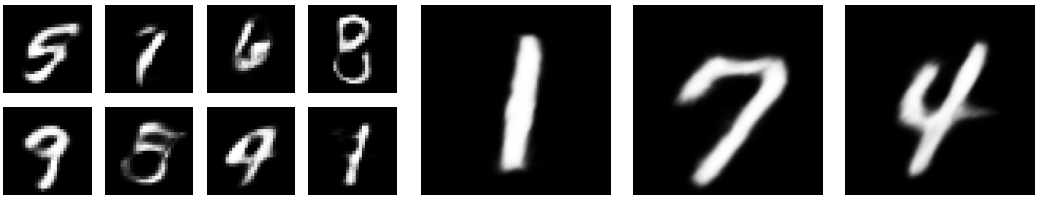
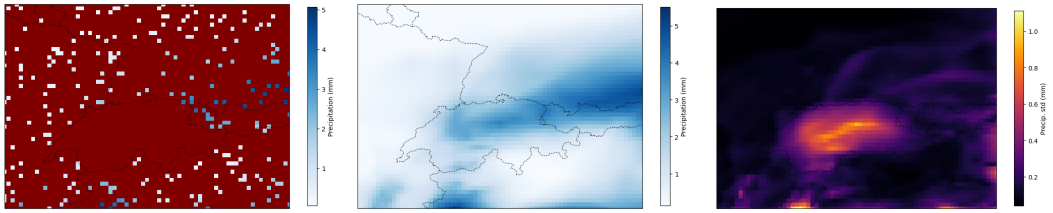


Figure 4: Generative modelling of MNIST digits with the AttnBDNP. Smaller images depict sampled prior functions evaluated at the same 28×28 grid of inputs that the training data lay on. Larger ones depict sampled functions queried at a 100×100 gid of inputs. The AttnBDNP's prior has encoded the functional behaviour of handwritten digits, so super-resolution is natively supported without needing further training.

when using a standard BNN prior versus the well-specified prior. This is repeated over 16 test datasets and we use per-datapoint log posterior predictive density (LPPD) and mean absolute error (MAE) as metrics. We consider SWAG (Maddox et al., 2019), MFVI, Langevin Monte Carlo (LMC; Rossky et al., 1978), GIVI, the BDNP, and Hamiltonian Monte Carlo (HMC; Neal, 1992). For our real-world setting, we substitute LMC and HMC for their stochastic-gradient counterparts (SGLD; Welling and Teh, 2011, SGHMC; Chen et al., 2014) due to larger context sets and architectures.



(a) Context set (b) Predictive mean (c) Predictive uncertainty

Figure 5: Demonstration of an ERA5 precipitation prediction test task with the BDNP. With no context points over Switzerland, the BDNP's predictive uncertainty is increased in that region.

For our synthetic data-generating processes, we consider squared-exponential GP, Heaviside, and sawtooth functions. In all three cases the inputs are sampled uniformly and the outputs are lightly corrupted with Gaussian noise. The real-world setting we consider is precipitation prediction over an area of Europe centered on Switzerland. We use the ERA5 Land dataset (Muñoz Sabater et al., 2021) and use longitude, latitude, and temperature as input variables. To make the prediction task even more challenging, we omit any context points from Switzerland in the test tasks, meaning predictions in this region are heavily influenced by the choice of prior. Fig. 5 demonstrates the ERA5 test task setup and Fig. 6 displays the results across the four settings. While it is clear that better priors boost predictive performance, we also see that there remains some considerable variation in performance amongst the methods when using a learned prior. In other words, a good prior is *not* all you need when working with BNNs; high quality approximate inference is still necessary.

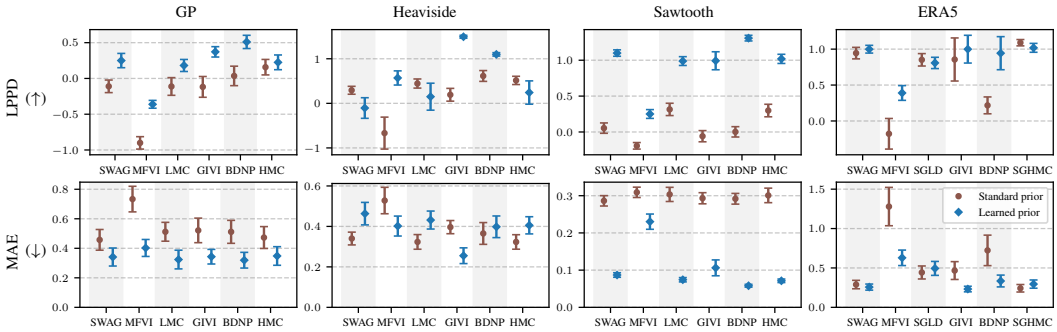
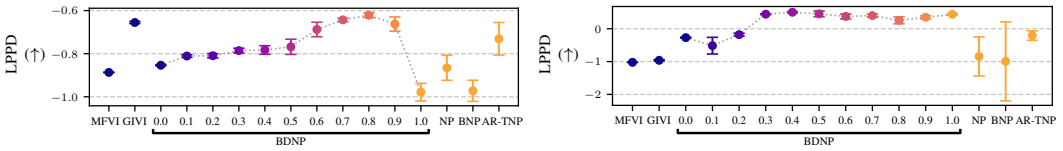


Figure 6: Target-set performance of approximate inference algorithms under a well-specified prior (blue) and a standard BNN prior (brown). The learned prior almost always leads to improved performance.

4.4 Can Restricting the Learned Prior Improve Meta-Level Data Efficiency?

We consider two problem settings for which the number of available datasets would traditionally be seen as too few for NPs to be applied. The first setting is a recasting of the Abalone age prediction task (Nash and Ford, 1994) as meta-regression. The three classes corresponding to a specimen's sex

(male/female/infant) are used to separate the data into three distinct datasets. The male and female datasets are used for meta-training and the infant dataset is reserved for testing. There are seven input features including various specimen size and weight measurements. The second setting is based on the Paul15 single-cell RNA sequencing dataset (Paul et al., 2015), and the task is to predict how specialised a cell is from 3451 gene expressions. We perform PCA on the data to obtain 100 information-rich abstract features, and split the dataset into 19 subsets according to cell clusterings that Paul et al. (2015) provide. We randomly select ten for meta-training and one for testing.



(a) Abalone (b) Paul15

Figure 7: Test-task target-set performance for two meta-learning problems with limited data. Decimals indicate the proportion of the BDNP's prior that is trained. BDNPs with partially trainable priors perform the best.

At the bottom end of the "prior-learnability" spectrum, we consider MFVI and GIVI baselines with standard priors (trained on test-task context set), while at the other, as comparable NPs which produce coherent function samples, we consider the original (latent-variable) NP (Garnelo et al., 2018b), the Bayesian NP (BNP; Volpp et al., 2021), and a transformer NP (Nguyen and Grover, 2022) under Bruinsma et al. (2023)'s state-of-the-art autoregressive sampling scheme (AR-TNP). We compare the baselines to the BDNP across its full range of prior-learnability where all non-learnable prior parameters are fixed to standard BNN prior settings, and prior parameters corresponding to earlier layer weights are made trainable first. The results are averaged over four trials. In Fig. 7 we see that in both problem settings, the best-performing model is a BDNP with a partially learnable prior (0.8 and 0.4 respectively). In the particularly data-scarce Abalone problem, we see clear evidence of overfitting in the fully-flexible NPs (including BDNP-1.0).

5 Discussion, Limitations, and the Bigger Picture

The BDNP's approximate posterior is a very good one. Each layer's approximate posterior is explicitly conditioned on the weights of the previous layers, enabling us to model weight correlations *between* layers. Since what matters in deep learning is the overall input-output transformation rather than individual layer transformations (Ober and Aitchison, 2021), the inter-layer correlations are likely to be somewhat responsible. Relatedly, our method effectively removes the redundant modes in the overall posterior caused by weight-space symmetries. This is because each layer is given access to the outputs of the previous layer as well as its own pseudo-outputs, meaning each layer is myopic in the sense that it only "sees" one mode of the posterior.

While it may be surprising that simple Gaussian priors can yield highly complex, even multimodal (Heaviside in Fig. 3, MNIST digits in Fig. 4) stochastic process priors, we note that the universal posterior predictive approximation result for mean-field Gaussian posteriors of Farquhar et al. (2020) likely also applies to Gaussian priors. We are encouraged by the demonstrated flexibility of Gaussian priors as it means finding good BNN priors is not as insurmountable a task as it might have been. The BDNP provides a solution to this problem for the case when multiple datasets are available, but the problem remains unsolved for the single-dataset case.

We highlight that our message is *not* that the BDNP is the one model to rule them all. When there are limited datasets, it is a very useful NP for practitioners to have in their inventory. Otherwise, our main focus in this work was in using it as a scientific tool with which to study BNNs. Inference in the BDNP scales unfavourably with architecture width (Appendix C), so we leave investigation of the BDNP's performance as a general-purpose NP to future work. Similarly, we introduce the BDAM for scientific interest but we do not include it in our experiments since it is of no use in answering our particular research questions.

Finally, we note that the impressive performance of our explicitly Bayesian meta-learning setup hints at an underlying lesson; that Bayesians should be using the abundant data of today's world to learn powerful *priors*, not to be forcing posteriors out of dubious priors. This message seems to be coming from an ever-growing chorus, with the posing of Bayesian inference as an in-context learning problem becoming increasingly popular (Dubois et al., 2020; Reuter et al., 2025; Chang et al., 2025).

REFERENCES

Laurence Aitchison. Why bigger is not always better: on finite and infinite neural networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 156–164. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/aitchison20a.html.

Anna Allen, Stratis Markou, Will Tebbutt, James Requeima, Wessel P Bruinsma, Tom R Andersson, Michael Herzog, Nicholas D Lane, Matthew Chantry, J Scott Hosking, et al. End-to-end data-driven weather prediction. *Nature*, pages 1–8, 2025.

Tom R. Andersson, Wessel P. Bruinsma, Stratis Markou, James Requeima, Alejandro Coca-Castro, Anna Vaughan, Anna-Louise Ellis, Matthew A. Lazzara, Dani Jones, Scott Hosking, and et al. Environmental sensor placement with convolutional Gaussian neural processes. *Environmental Data Science*, 2, 2023. doi: 10.1017/eds.2023.22.

Matthew Ashman, Tommy Rochussen, and Adrian Weller. Amortised inference in neural networks for small-scale probabilistic meta-learning, 2023. URL https://arxiv.org/abs/2310.15786.

Matthew Ashman, Cristiana Diaconu, Junhyuck Kim, Lakee Sivaraya, Stratis Markou, James Requeima, Wessel P Bruinsma, and Richard E. Turner. Translation equivariant transformer neural processes. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 1924–1944. PMLR, 21–27 Jul 2024a. URL https://proceedings.mlr.press/v235/ashman24a.html.

Matthew Ashman, Cristiana Diaconu, Eric Langezaal, Adrian Weller, and Richard E Turner. Gridded transformer neural processes for large unstructured spatio-temporal data. *arXiv preprint arXiv:2410.06731*, 2024b.

Matthew Ashman, Cristiana Diaconu, Adrian Weller, and Richard E. Turner. In-context in-context learning with transformer neural processes. In *Proceedings of the 6th Symposium on Advances in Approximate Bayesian Inference*, volume 253 of *Proceedings of Machine Learning Research*, pages 1–29. PMLR, 21 Jul 2024c. URL https://proceedings.mlr.press/v253/ashman24a.html.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2007. ISBN 0387310738. URL http://www.amazon.com/Pattern-Recognition-Learning-Information-Statistics/dp/0387310738%3FSubscriptionId%3D13CT5CVB80YFWJEPWS02%26tag%3Dws%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D0387310738.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1613–1622, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/blundell15.html.

Wessel Bruinsma, Stratis Markou, James Requeima, Andrew Y. K. Foong, Tom Andersson, Anna Vaughan, Anthony Buonomo, Scott Hosking, and Richard E Turner. Autoregressive conditional neural processes. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OAsXFPBfTBh.

T. Bui. Biases in variational Bayesian neural networks, Jan 2021.

Paul Edmund Chang, Nasrulloh Ratu Bagus Satrio Loka, Daolang Huang, Ulpu Remes, Samuel Kaski, and Luigi Acerbi. Amortized probabilistic conditioning for optimization, simulation and inference. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pages 703–711. PMLR, 03–05 May 2025. URL https://proceedings.mlr.press/v258/chang25a.html.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1683–1691, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/cheni14.html.

Tristan Cinquin, Marvin Pförtner, Vincent Fortuin, Philipp Hennig, and Robert Bamler. FSP-Laplace: Function-space priors for the Laplace approximation in Bayesian deep learning. In *Advances in Neural Information Processing Systems*, volume 37, pages 13897–13926. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/19774ce2d4b0d17a3a8aea26ad99fe8a-Paper-Conference.pdf.

Anish Dhir, Matthew Ashman, James Requeima, and Mark van der Wilk. A meta-learning approach to Bayesian causal discovery. In *The Thirteenth International Conference on Learning Representations*, 2025.

Francesco D' Angelo and Vincent Fortuin. Repulsive deep ensembles are Bayesian. In *Advances in Neural Information Processing Systems*, volume 34, pages 3451–3465. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/1c63926ebcabda26b5cdb31b5cc91efb-Paper.pdf.

Yann Dubois, Jonathan Gordon, and Andrew YK Foong. Neural process family. http://yanndubs.github.io/Neural-Process-Family/, September 2020.

Sebastian Farquhar, Lewis Smith, and Yarin Gal. Liberty or depth: Deep Bayesian neural nets do not need complex weight posterior approximations. In *Advances in Neural Information Processing Systems*, volume 33, pages 4346–4357. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/2dfe1946b3003933b7f8ddd71f24dbb1-Paper.pdf.

Leo Feng, Hossein Hajimirsadeghi, Yoshua Bengio, and Mohamed Osama Ahmed. Latent bottlenecked attentive neural processes. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=yIxtevizEA.

Daniel Flam-Shepherd, James Requeima, and David Duvenaud. Mapping Gaussian process priors to Bayesian neural networks. In *NIPS Bayesian deep learning workshop*, volume 3, 2017.

Andrew Foong, Wessel Bruinsma, Jonathan Gordon, Yann Dubois, James Requeima, and Richard Turner. Meta-learning stationary stochastic process prediction with convolutional neural processes. In *Advances in Neural Information Processing Systems*, volume 33, pages 8284–8295. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/5df0385cba256a135be596dbe28fa7aa-Paper.pdf.

Vincent Fortuin. Priors in Bayesian deep learning: A review. *International Statistical Review*, 90(3): 563–591, 2022.

Vincent Fortuin, Heiko Strathmann, and Gunnar Rätsch. Meta-learning mean functions for Gaussian processes, 2020. URL https://arxiv.org/abs/1901.08098.

Vincent Fortuin, Adrià Garriga-Alonso, Sebastian W. Ober, Florian Wenzel, Gunnar Ratsch, Richard E Turner, Mark van der Wilk, and Laurence Aitchison. Bayesian neural network priors revisited. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xkjqJYqRJy.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/gal16.html.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and S. M. Ali Eslami. Conditional neural processes. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1704–1713. PMLR, 10–15 Jul 2018a. URL https://proceedings.mlr.press/v80/garnelo18a.html.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural processes, 2018b. URL https://arxiv.org/abs/1807.01622.

Jonathan Gordon, John Bronskill, Matthias Bauer, Sebastian Nowozin, and Richard Turner. Meta-learning probabilistic inference for prediction. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HkxStoC5F7.

Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/file/7eb3c8be3d411e8ebfab08eba5f49632-Paper.pdf.

Jose Miguel Hernandez-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1861–1869, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/hernandez-lobatoc15.html.

Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, page 5–13, New York, NY, USA, 1993. Association for Computing Machinery. ISBN 0897916115. doi: 10.1145/168304.168306. URL https://doi.org/10.1145/168304.168306.

Matthew Hoffman and David Blei. Stochastic Structured Variational Inference. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 361–369, San Diego, California, USA, 09–12 May 2015. PMLR. URL https://proceedings.mlr.press/v38/hoffman15.html.

Alexander Immer, Maciej Korzepa, and Matthias Bauer. Improving predictions of Bayesian neural nets via local linearization. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 703–711. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/immer21a.html.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Teh. Attentive neural processes. In *International Conference on Learning Representations*, 01 2019. doi: 10.48550/arXiv.1901.05761.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. URL http://dblp.uni-trier.de/db/conf/iclr/iclr2014.html#KingmaW13.

David Krueger, Chin-Wei Huang, Riashat Islam, Ryan Turner, Alexandre Lacoste, and Aaron Courville. Bayesian hypernetworks, 2018. URL https://arxiv.org/abs/1710.04759.

Richard Kurle, Alexej Klushyn, and Ralf Herbrich. BALI: Learning neural networks via Bayesian layerwise inference, 2024. URL https://arxiv.org/abs/2411.12102.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf.

{Tuan Anh} Le, Hyunjik Kim, Marta Garnelo, Dan Rosenbaum, Jonathan Schwarz, and {Yee Whye} Teh. Empirical evaluation of neural process objectives. In *Bayesian Deep Learning workshop, Neural Information Processing Systems (NeurIPS)*, 2018.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. doi: 10.1162/neco.1989.1.4.541.

Juho Lee, Yoonho Lee, Jungtaek Kim, Eunho Yang, Sung Ju Hwang, and Yee Whye Teh. Bootstrapping neural processes. In *Advances in Neural Information Processing Systems*, volume 33, pages 6606–6615. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/492114f6915a69aa3dd005aa4233ef51-Paper.pdf.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/b3ba8f1bee1238a2f37603d90b58898d-Paper.pdf.

Christos Louizos and Max Welling. Structured and efficient variational deep learning with matrix Gaussian posteriors. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1708–1716, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/louizos16.html.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2218–2227. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/louizos17a.html.

David JC MacKay. A practical Bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

David JC MacKay. Introduction to Gaussian processes. *NATO ASI series F computer and systems sciences*, 168:133–166, 1998.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/118921efba23fc329e6560b27861f0c2-Paper.pdf.

Alexandre Maraval, Matthieu Zimmer, Antoine Grosnit, and Haitham Bou Ammar. End-to-end meta-Bayesian optimisation with transformer neural processes. In *Advances in Neural Information Processing Systems*, volume 36, pages 11246–11260. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/2561721d0ca69bab22b749cfc4f48f6c-Paper-Conference.pdf.

Alexander G de G Matthews, Jiri Hron, Mark Rowland, Richard E Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018.

David A. McAllester. Some PAC-Bayesian theorems. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, page 230–234, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 1581130570. doi: 10.1145/279943.279989. URL https://doi.org/10.1145/279943.279989.

J. Muñoz Sabater, E. Dutra, A. Agustí-Panareda, C. Albergel, G. Arduini, G. Balsamo, S. Boussetta, M. Choulga, S. Harrigan, H. Hersbach, B. Martens, D. G. Miralles, M. Piles, N. J. Rodríguez-Fernández, E. Zsoter, C. Buontempo, and J.-N. Thépaut. Era5-land: a state-of-the-art global reanalysis dataset for land applications. *Earth System Science Data*, 13(9):4349–4383, 2021. doi: 10.5194/essd-13-4349-2021. URL https://essd.copernicus.org/articles/13/4349/2021/.

Sellers Tracy Talbot Simon Cawthorn Andrew Nash, Warwick and Wes Ford. Abalone. UCI Machine Learning Repository, 1994. DOI: https://doi.org/10.24432/C55C7W.

Radford Neal. Bayesian learning via stochastic dynamics. In *Advances in Neural Information Processing Systems*, volume 5. Morgan-Kaufmann, 1992. URL https://proceedings.neurips.cc/paper_files/paper/1992/file/f29c21d4897f78948b91f03172341b7b-Paper.pdf.

Radford M Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

Tung Nguyen and Aditya Grover. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16569–16594. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/nguyen22b.html.

Sebastian W Ober and Laurence Aitchison. Global inducing point variational posteriors for bayesian neural networks and deep Gaussian processes. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8248–8259. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/ober21a.html.

Young-Jin Park and Han-Lim Choi. A neural process approach for probabilistic reconstruction of no-data gaps in lunar digital elevation maps. *Aerospace Science and Technology*, 113:106672, 04 2021. doi: 10.1016/j.ast.2021.106672.

Massimiliano Patacchiola, Jack Turner, Elliot J. Crowley, Michael O' Boyle, and Amos J Storkey. Bayesian meta-learning for the few-shot setting via deep kernels. In *Advances in Neural Information Processing Systems*, volume 33, pages 16108–16118. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/b9cfe8b6042cf759dc4c0cccb27a6737-Paper.pdf.

Franziska Paul, Ya'ara Arkin, Amir Giladi, Diego Adhemar Jaitin, Ephraim Kenigsberg, Hadas Keren-Shaul, Deborah Winter, David Lara-Astiaso, Meital Gury, Assaf Weiner, Eyal David, Nadav Cohen, Felicia Kathrine Bratt Lauridsen, Simon Haas, Andreas Schlitzer, Alexander Mildner, Florent Ginhoux, Steffen Jung, Andreas Trumpp, Bo Torben Porse, Amos Tanay, and Ido Amit. Transcriptional heterogeneity and lineage commitment in myeloid progenitors. *Cell*, 163(7): 1663–1677, 2015. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2015.11.013. URL https://www.sciencedirect.com/science/article/pii/S0092867415014932.

K. B. Petersen and M. S. Pedersen. The matrix cookbook, October 2008. URL http://www2.imm.dtu.dk/pubdb/p.php?3274. Version 20081110.

Alexander Pondaven, Märt Bakler, Donghu Guo, Hamzah Hashim, Martin Ignatov, and Harrison Zhu. Convolutional neural processes for inpainting satellite images, 2022. URL https://arxiv.org/abs/2205.12407.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Arik Reuter, Tim G. J. Rudner, Vincent Fortuin, and David Rügamer. Can transformers learn full Bayesian inference in context? In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=9Ip6fihKbc.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Skdvd2xAZ.

Tommy Rochussen and Vincent Fortuin. Sparse Gaussian neural processes. In *Proceedings of the 7th Symposium on Advances in Approximate Bayesian Inference*, volume 289 of *Proceedings of Machine Learning Research*, pages 194–219. PMLR, 29 Apr 2025. URL https://proceedings.mlr.press/v289/rochussen25a.html.

Peter J Rossky, Jimmie D Doll, and Harold L Friedman. Brownian dynamics as smart monte carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, 1978.

Jonas Rothfuss, Dominique Heyn, Andreas Krause, et al. Meta-learning reliable priors in the function space. *Advances in Neural Information Processing Systems*, 34:280–293, 2021.

Ravid Shwartz-Ziv, Micah Goldblum, Hossein Souri, Sanyam Kapoor, Chen Zhu, Yann LeCun, and Andrew Gordon Wilson. Pre-train your loss: Easy Bayesian transfer learning with informative priors. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=YCniF6_3Jb.

Gautam Singh, Jaesik Yoon, Youngsung Son, and Sungjin Ahn. Sequential neural processes. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/110209d8fae7417509ba71ad97c17639-Paper.pdf.

Emanuel Sommer, Lisa Wimmer, Theodore Papamarkou, Ludwig Bothmann, Bernd Bischl, and David Rügamer. Connecting the dots: Is mode-connectedness the key to feasible sample-based inference in Bayesian neural networks? In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 45988–46018. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/sommer24a.html.

Shengyang Sun, Guodong Zhang, Jiaxin Shi, and Roger Grosse. Functional variational bayesian neural networks. In *International Conference on Learning Representations*, 2019.

Terence Tao. *An introduction to measure theory*, volume 126. American Mathematical Soc., 2011.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Valentin Villecroze, Yixin Wang, and Gabriel Loaiza-Ganem. Last layer empirical bayes. In *I Can't Believe It's Not Better: Challenges in Applied Deep Learning*, 2025. URL https://openreview.net/forum?id=j53Vs162RA.

Michael Volpp, Fabian Flürenbrock, Lukas Grossberger, Christian Daniel, and Gerhard Neumann. Bayesian context aggregation for neural processes. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ufZN2-aehFa.

Michael Volpp, Philipp Dahlinger, Philipp Becker, Christian Daniel, and Gerhard Neumann. Accurate Bayesian meta-learning by accurate task posterior inference. In *The Eleventh International Conference on Learning Representations*, 2023.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Greg Yang. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*, 2019.

A Objective Functions

In this section we justify our choice of objective function, we discuss practical implementation of it, and we discuss alternatives. Throughout, we use q to denote distributions that depend on the approximate posterior $q(\mathbf{W}|\cdot)$ and p_Ψ to denote distributions that depend on the parameters of the prior *without* depending on $q(\mathbf{W}|\cdot)$.

A.1 PP-AVI

We begin by repeating Proposition 1 for the reader's benefit.

Proposition 1. *For $|\Xi| \to \infty$, maximisation of $\mathcal{L}_{\text{PP-AVI}}(\Xi)$ directly targets the three desiderata.*

In order to prove Proposition 1, we first provide formal definitions of the three desiderata. Throughout, we assume a true underlying data-generating process $\mathcal{D} \sim p(\mathcal{D})$.

Definition 1 (accurate approximate posteriors)**.** *A probabilistic meta-learner produces accurate approximate posteriors if the task-averaged KL divergence between approximate and true posteriors*

$$\mathbb{E}_{p(\mathcal{D}_c)}\Big[KL\left[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)\right]\Big] \tag{11}$$

is small.

Definition 2 (faithful prior)**.** *A probabilistic meta-learner has a prior that faithfully encodes the data-generating process if the task-averaged KL divergence between the true generative process $p(\mathbf{Y}_c|\mathbf{X}_c)$ and the model's marginal likelihood $p_\Psi(\mathbf{Y}_c|\mathbf{X}_c)$,*

$$\mathbb{E}_{p(\mathbf{X}_c)}\Big[KL\big[p(\mathbf{Y}_c|\mathbf{X}_c) \parallel p_\Psi(\mathbf{Y}_c|\mathbf{X}_c)\big]\Big] \tag{12}$$

is small.

Definition 3 (high quality predictions.)**.** *A probabilistic meta-learner produces high quality predictions if the task-averaged KL divergence between the true conditional generative process $p(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t)$ and the model's posterior predictive $q(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t)$,*

$$\mathbb{E}_{p(\mathcal{D}_c, \mathbf{X}_t)}\Big[KL\big[p(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t) \parallel q(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t)\big]\Big] \tag{13}$$

is small.

Proof. Recall the definition of $\mathcal{L}_{\text{PP-AVI}}(\Xi)$:

$$\mathcal{L}_{\text{PP-AVI}}(\Xi) := \frac{1}{|\Xi|}\sum_{j=1}^{|\Xi|} \log q\left(\mathbf{Y}_t^{(j)}|\mathcal{D}_c^{(j)}, \mathbf{X}_t^{(j)}\right) + \mathcal{L}_{\text{ELBO}}\left(\mathcal{D}_c^{(j)}\right). \tag{14}$$

Taking the limit of infinite datasets, we proceed as follows

$$\lim_{|\Xi|\to\infty} \mathcal{L}_{\text{PP-AVI}}(\Xi) = \mathbb{E}_{p(\mathcal{D})}\Big[\log q\left(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t\right) + \mathcal{L}_{\text{ELBO}}\left(\mathcal{D}_c\right)\Big] \tag{15}$$

$$= \mathbb{E}_{p(\mathcal{D})}\Big[\log q\left(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t\right) + \log p_\Psi(\mathbf{Y}_c|\mathbf{X}_c) - \mathrm{KL}[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)]\Big] \tag{16}$$

$$= \mathbb{E}_{p(\mathcal{D}_c,\mathbf{X}_t)}\Big[\mathbb{E}_{p(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)}\left[\log q(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)\right]\Big] + \mathbb{E}_{p(\mathbf{X}_c)}\Big[\mathbb{E}_{p(\mathbf{Y}_c|\mathbf{X}_c)}\left[\log p_\Psi(\mathbf{Y}_c|\mathbf{X}_c)\right]\Big] - \mathbb{E}_{p(\mathcal{D}_c)}\Big[\mathrm{KL}\left[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)\right]\Big] \tag{17}$$

$$= \mathbb{E}_{p(\mathcal{D}_c,\mathbf{X}_t)}\Big[\mathbb{E}_{p(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)}\left[\log \frac{q(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)p(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)}{p(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)}\right]\Big] + \mathbb{E}_{p(\mathbf{X}_c)}\Big[\mathbb{E}_{p(\mathbf{Y}_c|\mathbf{X}_c)}\left[\log \frac{p_\Psi(\mathbf{Y}_c|\mathbf{X}_c)p\mathbf{Y}_c|\mathbf{X}_c)}{p(\mathbf{Y}_c|\mathbf{X}_c)}\right]\Big] - \mathbb{E}_{p(\mathcal{D}_c)}\Big[\mathrm{KL}\left[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)\right]\Big] \tag{18}$$

$$= \underbrace{\mathbb{H}(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)}_{\text{const.}} - \mathbb{E}_{p(\mathcal{D}_c,\mathbf{X}_t)}\Big[\mathrm{KL}\Big[p(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t) \parallel q(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)\Big]\Big] + \underbrace{\mathbb{H}(\mathbf{Y}_c|\mathbf{X}_c)}_{\text{const.}} - \mathbb{E}_{p(\mathbf{X}_c)}\Big[\mathrm{KL}\Big[p(\mathbf{Y}_c|\mathbf{X}_c) \parallel p_\Psi(\mathbf{Y}_c|\mathbf{X}_c)\Big]\Big] - \mathbb{E}_{p(\mathcal{D}_c)}\Big[\mathrm{KL}\left[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)\right]\Big] \tag{19}$$

where $\mathbb{H}(\cdot|\cdot)$ denotes the Shannon conditional entropy. Since the entropy terms are constant with respect to the variational and model parameters $\{\Theta, \Psi\}$, we reach

$$\arg\max_{\{\Theta,\Psi\}} \lim_{|\Xi|\to\infty} \mathcal{L}_{\text{PP-AVI}}(\Xi) \equiv \arg\min_{\{\Theta,\Psi\}} \Bigg[\mathbb{E}_{p(\mathcal{D}_c,\mathbf{X}_t)}\Big[\mathrm{KL}\Big[p(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t) \parallel q(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t)\Big]\Big] + \mathbb{E}_{p(\mathbf{X}_c)}\Big[\mathrm{KL}\Big[p(\mathbf{Y}_c|\mathbf{X}_c) \parallel p_\Psi(\mathbf{Y}_c|\mathbf{X}_c)\Big]\Big] + \mathbb{E}_{p(\mathcal{D}_c)}\Big[\mathrm{KL}\left[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)\right]\Big]\Bigg]. \tag{20}$$

□

Practical Details. Unfortunately, both terms in $\mathcal{L}_{\text{PP-AVI}}$ are analytically intractable. As a first step, we decompose the ELBO into the usual *reconstruction error / complexity penalisation* form:

$$\mathcal{L}_{\text{PP-AVI}}(\Xi) = \frac{1}{|\Xi|}\sum_{j=1}^{|\Xi|} \log q\left(\mathbf{Y}_t^{(j)}|\mathcal{D}_c^{(j)}, \mathbf{X}_t^{(j)}\right) + \mathbb{E}_{q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)}\left[\log p\left(\mathbf{Y}_c^{(j)}|\mathbf{W}, \mathbf{X}_c^{(j)}\right)\right] - \mathrm{KL}\left[q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right) \parallel p_\Psi\left(\mathbf{W}\right)\right]. \tag{21}$$

We deal with these three terms in order of increasing difficulty. The final term becomes tractable by decomposing it into the sum of layerwise KL divergences

$$\mathrm{KL}\left[q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right) \parallel p_\Psi(\mathbf{W})\right] = \sum_{l=1}^{L} \mathrm{KL}\left[q\left(\mathbf{W}^l|\mathbf{W}^{1:l-1}, \mathcal{D}_c^{(j)}\right) \parallel p_{\psi_l}(\mathbf{W})\right] \tag{22}$$

where each term in the sum is available in closed form as the KL divergence between two multivariate Gaussians.

The middle term, the (context set) *expected log-likelihood*, is unbiasedly estimable via Monte Carlo integration with a finite number of samples K

$$\mathbb{E}_{q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)}\left[\log p\left(\mathbf{Y}_c^{(j)}|\mathbf{W},\mathbf{X}_c^{(j)}\right)\right] \approx \frac{1}{K}\sum_{k=1}^{K}\log p\left(\mathbf{Y}_c^{(j)}|\mathbf{W}^{(k)},\mathbf{X}_c^{(j)}\right) \tag{23}$$

where $\mathbf{W}^{(k)} \sim q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)$ and low-variance gradient estimates are available via application of the reparameterisation trick (Kingma and Welling, 2014).

The posterior predictive term, which can equivalently be referred to as the (target set) *log expected likelihood*, is defined as

$$\log q\left(\mathbf{Y}_t^{(j)}|\mathcal{D}_c^{(j)},\mathbf{X}_t^{(j)}\right) := \log\int p\left(\mathbf{Y}_t^{(j)}|\mathbf{W},\mathbf{X}_t^{(j)}\right) q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)\mathrm{d}\mathbf{W}. \tag{24}$$

Again turning to Monte Carlo integration with K samples, we estimate this term via

$$\log q\left(\mathbf{Y}_t^{(j)}|\mathcal{D}_c^{(j)},\mathbf{X}_t^{(j)}\right) \approx \log\frac{1}{K}\sum_{k=1}^{K} p\left(\mathbf{Y}_t^{(j)}|\mathbf{W}^{(k)},\mathbf{X}_t^{(j)}\right) \tag{25}$$

$$= \mathrm{LogSumExp}\left(\left\{\log p\left(\mathbf{Y}_t^{(j)}|\mathbf{W}^{(k)},\mathbf{X}_t^{(j)}\right)\right\}_{k=1}^{K}\right) - \log K \tag{26}$$

where $\mathbf{W}^{(k)} \sim q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)$ and we use the log-sum-exp trick to maintain stable computations in log-space. Unfortunately, this Monte Carlo estimate is biased and so we cannot get away with single-sample estimates as we might with standard variational inference. In our experiments we used 8 or 16 samples.

A.2 Standard AVI / Meta Empirical Bayes

Perhaps the most obvious alternative objective function to consider would be the standard AVI loss given by the task-averaged ELBO:

$$\mathcal{L}_{\text{AVI}}(\Xi) := \frac{1}{|\Xi|}\sum_{j=1}^{|\Xi|}\mathcal{L}_{\text{ELBO}}\left(\mathcal{D}_c^{(j)}\right). \tag{27}$$

In the infinite-dataset limit, this objective function corresponds to minimising two of the three desired expected KL divergences; $\mathbb{E}_{p(\mathbf{X}_c)}\Big[\mathrm{KL}\big[p(\mathbf{Y}_c|\mathbf{X}_c) \parallel p_\Psi(\mathbf{Y}_c|\mathbf{X}_c)\big]\Big]$, and $\mathbb{E}_{p(\mathcal{D}_c)}\Big[\mathrm{KL}\left[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)\right]\Big]$. Supposing we have globally optimised this objective function for an infinitely flexible model (e.g. not necessarily a BDNP) under infinite datasets, the trained model would have perfect approximate posteriors and a prior that models the true (context) data-generating process exactly. Perhaps the missing term would then be minimised for free—maybe we should expect extremely high quality posterior predictives from this resulting model. While this may be true (proper analysis left to future work), in practice we found $\mathcal{L}_{\text{AVI}}$ to be significantly harder to globally optimise than $\mathcal{L}_{\text{PP-AVI}}$. When training our BDNPs, we initialised the parameters of the prior to those of the standard isotropic Gaussian prior. For some tasks (particularly the heaviside and sawtooth function regression problems) it seems that this was a particularly adversarial initialisation in that maximisation of $\mathcal{L}_{\text{AVI}}$ seemed not to be able to "break free" of the predictive behaviour induced by the standard BNN prior. So, even if the extra expected KL term from the PP-AVI loss is not strictly necessary, it seems to simplify the loss-landscape by suppressing the non-globally optimal modes via penalising miscalibrated posterior predictives.

Since the ELBO is a lower-bound to the log marginal likelihood $\mathcal{L}_{\text{ELBO}}(\mathcal{D}_c) \leq \log p(\mathbf{Y}_c|\mathbf{X}_c)$, under infinite datasets we have that the AVI loss is a lower bound to the *expected* log marginal likelihood across tasks:

$$\lim_{|\Xi|\to\infty}\mathcal{L}_{\text{AVI}}(\Xi) \leq \mathbb{E}_{p(\mathbf{X}_c)}\big[\log p(\mathbf{Y}_c|\mathbf{X}_c)\big]. \tag{28}$$

This then corresponds to a meta-level type-II marginal likelihood scheme, or, meta-level empirical Bayes. Note that this is the objective function used in Rochussen and Fortuin (2025) and Ashman et al. (2023).

A.3 NP Maximum Likelihood / Posterior Predictive Maximisation / ML-PIP

Neural processes tend to be used as probabilistic predictors more than anything else (such as generative models), so the only distribution users really care about is the posterior predictive $q(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t)$. Note that while this distribution is referred to as the posterior predictive in the context of latent-variable NPs, for the conditional family of NPs it is referred to as the predictive likelihood. The obvious training scheme would then be to maximise the (log) posterior predictive likelihood of the target set over many tasks. While this is the de-facto approach in conditional family NPs, Gordon et al. (2019) demonstrated that it is sufficient if we wish to perform amortised variational inference in latent-variable meta-learners too. Gordon et al. refer to this scheme as *meta-learning probabilistic inference for prediction* (ML-PIP). We refer to this objective as the neural process maximum likelihood (NPML) objective, and it takes the form

$$\mathcal{L}_{\text{NPML}}(\Xi) := \frac{1}{|\Xi|}\sum_{j=1}^{|\Xi|} \log q(\mathbf{Y}_t^{(j)}|\mathcal{D}_c^{(j)}, \mathbf{X}_t^{(j)}). \tag{29}$$

In the infinite-dataset limit, maximising $\mathcal{L}_{\text{NPML}}$ corresponds to minimising the expected KL term that is missing from $\mathcal{L}_{\text{AVI}}$. In other words, we have that

$$\mathcal{L}_{\text{PP-AVI}}(\Xi) \equiv \mathcal{L}_{\text{NPML}}(\Xi) + \mathcal{L}_{\text{AVI}}(\Xi). \tag{30}$$

While NPML has been demonstrated to work well in NPs[6], with the BDNP we found it would lead either to solutions that modelled the data-generating process very badly, or to high numerical instability causing training runs to fail. As with $\mathcal{L}_{\text{AVI}}$, we suspect this is because of loss-landscape multimodality and difficulty ensuring global optimisation. In particular, we suspect that the globally optimal modes in $\mathcal{L}_{\text{AVI}}$ and $\mathcal{L}_{\text{NPML}}$ correspond to each other but that the secondary modes in each loss-landscape do not, meaning global numerical optimisation of their sum is more straightforward.

A.4 Neural Process Variational Inference

Another way to train latent-variable NPs is through a more conventional variational approach, where the objective function is given by a lower bound to the conditional marginal likelihood $p_\Psi(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t)$ averaged over tasks. We refer to this as the neural process variational inference (NPVI) objective, and it is defined as

$$\begin{aligned}\mathcal{L}_{\text{NPVI}}(\Xi) := \frac{1}{|\Xi|}\sum_{j=1}^{|\Xi|} &\mathbb{E}_{q(\mathbf{W}|\mathcal{D}^{(j)})}\left[\log p\left(\mathbf{Y}_t^{(j)}|\mathbf{W}, \mathbf{X}_t^{(j)}\right)\right] \\ &- \text{KL}\left[q\left(\mathbf{W}|\mathcal{D}^{(j)}\right) \parallel p_\Psi\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)\right].\end{aligned} \tag{31}$$

While NPVI is sensible in the infinite-dataset limit, being equivalent to minimising

$$\mathbb{E}_{p(\mathcal{D}_c, \mathbf{X}_t)}\left[\text{KL}\left[p(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t) \parallel p_\Psi(\mathbf{Y}_t|\mathcal{D}_c, \mathbf{X}_t)\right]\right] + \mathbb{E}_{p(\mathcal{D})}\left[\text{KL}\left[q(\mathbf{W}|\mathcal{D}) \parallel p_\Psi(\mathbf{W}|\mathcal{D})\right]\right], \tag{32}$$

it is intractable due to the presence of the true posterior $p_\Psi(\mathbf{W}|\mathcal{D}_c)$. While this is typically circumvented by approximating the true posterior with the approximate posterior $q(\mathbf{W}|\mathcal{D}_c)$, NPVI only targets Eq. (32) if the approximate posterior is exact. Under a poor approximate posterior, such as at the beginning of training, the loss landscape is then quite different to what it should be and it is unclear how close the practical version of $\mathcal{L}_{\text{NPVI}}$ ever becomes to the ideal one. Indeed, the NPVI approach is known to yield suboptimal predictive performance (Le et al., 2018). Furthermore, the KL term involves computing the two approximate posteriors $q(\mathbf{W}|\mathcal{D})$ and $q(\mathbf{W}|\mathcal{D}_c)$. Since the computational bottleneck associated with the BDNP is in computing the approximate posterior, the double approximate posterior computation renders $\mathcal{L}_{\text{NPVI}}$ even more inappropriate for the BDNP, and so we did not attempt to use it at all. Note that $\mathcal{L}_{\text{NPVI}}$ was the objective function used in the original (latent-variable) neural process paper (Garnelo et al., 2018b).

[6] in which case the latent variable is denoted by $\mathbf{z}$ rather than $\mathbf{W}$.

A.5 TELL-AVI

Motivated to find an objective function that can be ubiasedly estimated and which has similar infinite-dataset behaviour to $\mathcal{L}_{\text{PP-AVI}}$, we introduce the target-expected-log-likelihood (TELL) AVI loss. We define it as

$$\mathcal{L}_{\text{TELL-AVI}}(\Xi) := \frac{1}{|\Xi|}\sum_{j=1}^{|\Xi|}\mathbb{E}_{q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)}\left[\log p\left(\mathbf{Y}_t^{(j)}|\mathbf{W},\mathbf{X}_t^{(j)}\right)\right] + \mathcal{L}_{\text{ELBO}}\left(\mathcal{D}_c^{(j)}\right) \tag{33}$$

where we note the only difference between $\mathcal{L}_{\text{PP-AVI}}$ and $\mathcal{L}_{\text{TELL-AVI}}$ is in the ordering of the $\log$ and expectation $\mathbb{E}_{q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)}$ in the left-hand term[7]. The right-hand term is estimated exactly as in $\mathcal{L}_{\text{PP-AVI}}$, but the left-hand can be *unbiasedly* estimated by its Monte Carlo estimate from K samples, $\frac{1}{K}\sum_{k=1}^{K}\log p\left(\mathbf{Y}_t^{(j)}|\mathbf{W}^{(k)},\mathbf{X}_t^{(j)}\right)$ for $\mathbf{W}^{(k)} \sim q\left(\mathbf{W}|\mathcal{D}_c^{(j)}\right)$, enabling the use of very few samples for decreased computational cost.

Seeking to rewrite the expected log-likelihood term in terms of the log posterior predictive (log expected likelihood), we have

$$\mathbb{E}_{q(\mathbf{W}|\mathcal{D}_c)}\left[\log p\left(\mathbf{Y}_t|\mathbf{W},\mathbf{X}_t\right)\right] = \mathbb{E}_{q(\mathbf{W}|\mathcal{D}_c)}\left[\log \frac{p\left(\mathbf{Y}_t|\mathbf{W},\mathbf{X}_t\right)q(\mathbf{W}|\mathcal{D}_c)q\left(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t\right)}{q(\mathbf{W}|\mathcal{D}_c)q\left(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t\right)}\right] \tag{34}$$

$$= \mathbb{E}_{q(\mathbf{W}|\mathcal{D}_c)}\left[\log q\left(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t\right)\right] + \mathbb{E}_{q(\mathbf{W}|\mathcal{D}_c)}\left[\log \frac{q\left(\mathbf{Y}_t,\mathbf{W}|\mathcal{D}_c,\mathbf{X}_t\right)}{q(\mathbf{W}|\mathcal{D}_c)q\left(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t\right)}\right] \tag{35}$$

$$= \log q\left(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t\right) + \mathbb{E}_{q(\mathbf{W}|\mathcal{D}_c)}\left[\log \frac{q\left(\mathbf{W}|\mathbf{X}_t,\mathbf{Y}_t,\mathcal{D}_c\right)}{q(\mathbf{W}|\mathcal{D}_c)}\right] \tag{36}$$

$$= \log q\left(\mathbf{Y}_t|\mathcal{D}_c,\mathbf{X}_t\right) - \text{KL}\left[q(\mathbf{W}|\mathcal{D}_c) \parallel q(\mathbf{W}|\mathcal{D})\right], \tag{37}$$

which gives us the following relationship between $\mathcal{L}_{\text{TELL-AVI}}$ and $\mathcal{L}_{\text{PP-AVI}}$:

$$\lim_{|\Xi|\to\infty}\mathcal{L}_{\text{TELL-AVI}} = \lim_{|\Xi|\to\infty}\mathcal{L}_{\text{PP-AVI}} - \mathbb{E}_{p(\mathcal{D})}\Big[\text{KL}\big[q(\mathbf{W}|\mathcal{D}_c) \parallel q(\mathbf{W}|\mathcal{D})\big]\Big]. \tag{38}$$

Though this expected KL term is somewhat sensible when interpreted as encouraging approximate posteriors from partial datasets to be similar to their full-data counterparts, the reverse interpretation (that it encourages full-data approximate posteriors to be only as good as their partial-data counterparts) highlights its dubiousness. Furthermore, the extra term would serve as a distractor from the more important KL term that $\mathcal{L}_{\text{PP-AVI}}$ already includes; $\mathbb{E}_{q(\mathbf{W}|\mathcal{D}_c)}\Big[\text{KL}\big[q(\mathbf{W}|\mathcal{D}_c) \parallel p_\Psi(\mathbf{W}|\mathcal{D}_c)\big]\Big]$, and it is unclear what behaviour the sum of the two KL terms involving $q(\mathbf{W}|\mathcal{D}_c)$ leads to. In preliminary experiments we found that $\mathcal{L}_{\text{TELL-AVI}}$ led to reasonable performance in terms of predictions and the resulting learned prior, but it was never quite as good as $\mathcal{L}_{\text{PP-AVI}}$.

[7] i.e., $\mathcal{L}_{\text{PP-AVI}}$ might have equivalently been called the target-log-expected-likelihood (TLEL) AVI loss.

B Inference with Alternative Priors

Fully factorised. The simplest form of prior is a fully factorised one $p_{\psi_l}(\mathbf{W}^l) = \prod_{d=1}^{d_l}\prod_{d'=1}^{d_{l-1}}\mathcal{N}\left(w^l_{d,d'};\mu^l_{d,d'},{\sigma^l_{d,d'}}^2\right)$. Inference with this prior is performed similarly to the unitwise factorised prior, except that the unitwise covariance matrix is given by $\mathbf{\Sigma}^l_d = \text{diag}\left({\boldsymbol{\sigma}^l_d}^2\right)$ and can therefore be inverted by taking the reciprocal of the diagonal elements, taking $\mathcal{O}(d_{l-1})$ time rather than $\mathcal{O}({d_{l-1}}^3)$ time.

Layerwise matrix-Gaussian. While a seemingly obvious prior to consider is the matrix-Gaussian over layerwise weight matrices (Kurle et al., 2024; Louizos and Welling, 2016; Ritter et al., 2018), this prior turns out not to fit nicely into our amortisation framework. The pseudo-observation noise variances in each layer would have to become part of the prior rather than the pseudo likelihoods, meaning our inference networks would lose their ability to up- or down-weight pseudo-observations by predicting their corresponding noise level. In other words, using matrix-Gaussian priors would force us to destroy the Bayesian context aggregation behaviour of our method.

Layerwise full-rank Gaussian. A richer prior than the unitwise factorised one is a layerwise full-rank Gaussian defined over $\text{vec}(\mathbf{W}^l)$. To lighten the notation, we define $\boldsymbol{\omega}^l := \text{vec}(\mathbf{W}^l)$. The prior is then $p_{\psi_l}(\mathbf{W}^l) = p(\boldsymbol{\omega}^l) = \mathcal{N}\left(\boldsymbol{\omega}^l;\boldsymbol{\mu}^l,\mathbf{\Sigma}^l\right)$. In order to apply the standard (single-output) Bayesian linear regression results, we need to augment the data matrices $\mathbf{X}^{l-1},\mathbf{Y}^l$ to some matrix-vector pair $\boldsymbol{\chi}^{l-1}_\phi,\boldsymbol{y}^l$, where $\boldsymbol{y}^l := \text{vec}(\mathbf{Y}^l)$, so that the likelihood model can be written as $\boldsymbol{y}^l = \boldsymbol{\chi}^{l-1}_\phi\boldsymbol{\omega}^l$. It turns out that the "vec trick" (Petersen and Pedersen, 2008) of the Kronecker product $\otimes$ gives us what we need:

$$\mathbf{Y}^l = \phi(\mathbf{X}^{l-1})\mathbf{W}^l \tag{39}$$

$$\therefore \tag{40}$$

$$\boldsymbol{y}^l = \text{vec}\left(\phi(\mathbf{X}^{l-1})\mathbf{W}^l\right) \tag{41}$$

$$= \text{vec}\left(\phi(\mathbf{X}^{l-1})\mathbf{W}^l\mathbf{I}^\top\right) \tag{42}$$

$$= \left(\mathbf{I}\otimes\phi(\mathbf{X}^{l-1})\right)\boldsymbol{\omega}^l \tag{43}$$

giving us that $\boldsymbol{\chi}^{l-1}_\phi := \mathbf{I}\otimes\phi(\mathbf{X}^{l-1})$. Note that $\boldsymbol{\chi}^{l-1}_\phi \in \mathbb{R}^{d_l N\times d_l d_{l-1}}$. The posterior is therefore

$$p\left(\boldsymbol{\omega}^l|\mathbf{X}^{l-1},\mathbf{Y}^l\right) = \mathcal{N}\left(\boldsymbol{\omega}^l;\mathbf{m}^l,\mathbf{S}^l\right) \tag{44}$$

with mean vector and covariance matrix given by

$$\mathbf{S}^{l^{-1}} = \mathbf{\Sigma}^{l^{-1}} + {\boldsymbol{\chi}^{l-1}_\phi}^\top\mathbf{\Lambda}^l\boldsymbol{\chi}^{l-1}_\phi \tag{45}$$

$$\mathbf{m}^l = \mathbf{S}^l\left(\mathbf{\Sigma}^{l^{-1}}\boldsymbol{\mu}^l + {\boldsymbol{\chi}^{l-1}_\phi}^\top\mathbf{\Lambda}^l\boldsymbol{y}^l\right) \tag{46}$$

where $\mathbf{\Lambda}^l \in \mathbb{R}^{Nd_l\times Nd_l}$ is a diagonal precision matrix formed by stacking the $\mathbf{\Lambda}^l_d$ matrices along the leading diagonal of an $Nd_l\times Nd_l$ zeroes matrix. More specifically, $\mathbf{\Lambda}^l := \text{diag}\left(\text{concat}\left(\{\boldsymbol{\lambda}^l_d\}_{d=1}^{d_l}\right)\right)$ where the n-th element of $\boldsymbol{\lambda}^l_d \in \mathbb{R}^N$ is given by $\frac{1}{{\sigma^l_{n,d}}^2}$. To avoid direct matrix multiplications with the $d_lN\times d_ld_{l-1}$ matrix $\boldsymbol{\chi}^{l-1}_\phi$, Eq. (45) and Eq. (46) can be re-written as

$$\mathbf{S}^{l^{-1}} = \mathbf{\Sigma}^{l^{-1}} + \mathbf{\Phi}^l \tag{47}$$

$$\mathbf{m}^l = \mathbf{S}^l\left(\mathbf{\Sigma}^{l^{-1}}\boldsymbol{\mu}^l + \boldsymbol{\phi}^l\right) \tag{48}$$

where $\mathbf{\Phi}^l$ is constructed by stacking the $d_{l-1}\times d_{l-1}$ matrices $\{\phi(\mathbf{X}^{l-1})^\top\mathbf{\Lambda}^l_d\phi(\mathbf{X}^{l-1})\}_{d=1}^{d_l}$ along the leading diagonal of a $d_ld_{l-1}\times d_ld_{l-1}$ zeroes matrix, and $\boldsymbol{\phi}^l$ is given by $\text{vec}\left(\phi(\mathbf{X})^{l-1^\top}\tilde{\mathbf{Y}}^l\right)$ where the elements of $\tilde{\mathbf{Y}}^l$ are given by $\tilde{y}^l_{n,d} := \frac{y^l_{n,d}}{{\sigma^l_{n,d}}^2}$.

Note that inference with this type of prior is significantly more expensive than with the unitwise or weightwise factorised priors. This is because the computation is dominated by the inversion of $d_l d_{l-1} \times d_l d_{l-1}$ covariance matrices, requiring $\mathcal{O}\left(d_l^3 d_{l-1}^3\right)$ operations. For a network with uniform hidden layer width, the cost of inference therefore scales sextically $\left(d^6\right)$ with network width.

Full-rank Gaussian. The richest prior we consider is a full-rank Gaussian defined over all network weights $\boldsymbol{\omega} := \text{concat}\left(\{\text{vec}(\mathbf{W}^l)\}_{l=1}^L\right)$, i.e., $p(\mathbf{W}^{1:L}) = p(\boldsymbol{\omega}) = \mathcal{N}(\boldsymbol{\omega}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. Inference is then performed in the same way as with the layerwise full-rank Gaussian prior, save that the layerwise factors $\{p_{\psi_l}(\mathbf{W}^l)\}_{l=1}^L$ are replaced with the layerwise conditionals $\{p(\mathbf{W}^l|\mathbf{W}^{1:l-1})\}_{l=1}^L$. Each layerwise conditional takes the form

$$p(\mathbf{W}^l|\mathbf{W}^{1:l-1}) = \int p(\boldsymbol{\omega}^{l:L}|\boldsymbol{\omega}^{1:l-1}) \mathrm{d}\boldsymbol{\omega}^{l+1:L} = \mathcal{N}\left(\boldsymbol{\omega}^l; \boldsymbol{\mu}^{c|p}, \boldsymbol{\Sigma}^{c|p}\right) \tag{49}$$

where c and p denote the indices of the **c**urrent and **p**revious layer weights respectively, and the conditional mean and covariance are given by

$$\boldsymbol{\mu}^{c|p} = \boldsymbol{\mu}_c + \boldsymbol{\Sigma}_{cp}\boldsymbol{\Sigma}_{pp}^{-1}\left(\boldsymbol{\omega}^{1:l-1} - \boldsymbol{\mu}_p\right) \tag{50}$$

$$\boldsymbol{\Sigma}^{c|p} = \boldsymbol{\Sigma}_{cc} - \boldsymbol{\Sigma}_{cp}\boldsymbol{\Sigma}_{pp}^{-1}\boldsymbol{\Sigma}_{pc} \tag{51}$$

where $\boldsymbol{\omega}^a$ denotes the (vectorised) weights of layer a, $\boldsymbol{\mu}_a$ denotes the subvector of $\boldsymbol{\mu}$ indexed by a, and similarly $\boldsymbol{\Sigma}_{ab}$ denotes the submatrix of $\boldsymbol{\Sigma}$ indexed by a and b.

C MINIBATCHING

Our minibatched posterior sampling algorithm for scalable inference in the BDNP is detailed in Algorithm 1. The algorithm describes the usual full-batch posterior sampling procedure if the number of minibatches is set to $B = 1$. We use `partial_forward_pass`$(\mathbf{X}, \mathbf{W}^{1:l})$ to refer to the propagation of inputs $\mathbf{X}$ forward to the l-th layer of the network using weights $\mathbf{W}^{1:l}$, and `compute_posterior` to refer to computation of Eq. (5).

Algorithm 1 A minibatched BDNP posterior sampling algorithm.

Require: inference networks $\{g_{\theta_l}\}_{l=1}^{L}$ and priors $\{p_{\psi_l}(\mathbf{W}^l)\}_{l=1}^{L}$,
  for $l = 1, \ldots, L$ **do**
    for $b = 1, \ldots, B$ **do**
      Obtain $\mathcal{D}_b$
      if $l = 1$ **then**
        $\mathbf{X}_b^0 \leftarrow \mathbf{X}_b$
      else
        $\mathbf{X}_b^{l-1} \leftarrow$ `partial_forward_pass`$(\mathbf{X}_b, \mathbf{W}_{1:B}^{1:l-1})$
      end if
      if $b = 1$ **then**
        $q(\mathbf{W}^l|\mathbf{W}_{1:B}^{1:l-1}, \mathcal{D}_1) \leftarrow$ `compute_posterior`$\big(\mathbf{X}_1^{l-1}, \mathcal{D}_1, g_{\theta_l}, p_{\psi_l}(\mathbf{W}^l)\big)$
      else
        $q(\mathbf{W}^l|\mathbf{W}_{1:B}^{1:l-1}, \mathcal{D}_{1:b}) \leftarrow$ `compute_posterior`$\big(\mathbf{X}_b^{l-1}, \mathcal{D}_b, g_{\theta_l}, q(\mathbf{W}^l|\mathbf{W}_{1:B}^{1:l-1}, \mathcal{D}_{1:b-1})\big)$
      end if
      Discard $\mathcal{D}_b$
    end for
    Sample $\mathbf{W}_{1:B}^l \sim q(\mathbf{W}^l|\mathbf{W}_{1:B}^{1:l-1}, \mathcal{D}_{1:B})$
  end for
  return $\{\mathbf{W}_{1:B}^l\}_{l=1}^{L}$

The time complexity associated with a forward pass through the l-th layer is $\mathcal{O}(n_c {d_{l-1}}^3 d_l)$, and the corresponding space complexity is $\mathcal{O}(n_c {d_{l-1}}^2 d_l)$. The superlinear scaling with the number of input neurons d_{l-1} arises from inverting $d_{l-1} \times d_{l-1}$ matrices. The purpose of our minibatching algorithm is to reduce the linear scaling of the space complexity with context size to constant, i.e. to convert the space complexity to $\mathcal{O}(|b| {d_{l-1}}^2 d_l)$, where $|b|$ is the batch size.

While this is greatly beneficial at prediction time, during training it is necessary to store in memory the gradient information corresponding to all batches. This reverts the memory complexity to $\mathcal{O}(n_c {d_{l-1}}^2 d_l)$ in spite of the minibatching procedure. To remedy this, we propose to randomly select just one of the minibatches to compute gradients with respect to at every layer (i.e. the same minibatch for all layers), discarding gradient information for the other minibatches. While this leads to noisier gradient update steps, it reduces the memory complexity back down to $\mathcal{O}(|b| {d_{l-1}}^2 d_l)$ as desired. Although we leave a detailed analysis of any biases that this might introduce to future work, we emphasise here that this scheme maintains a distinct advantage over naive minibatching by simply restricting the context set size—our scheme maintains the ability for the BDNP to learn to generalise predictive inference *across various context set sizes* during training.

D ONLINE LEARNING VIA LAST-LAYER SEQUENTIAL BAYESIAN INFERENCE

As mentioned in Section 2.4, any form of posterior update due to new data must be applied in full to each layer's posterior before computing the next layer's conditional posterior. Unfortunately, this means that sequential Bayesian inference cannot be used naively for online learning in the BDNP. To see why, consider a partioning of a task's dataset into "original" and "update" subsets $\mathcal{D} = \mathcal{D}_o \cup \mathcal{D}_u$. Assume we have already computed the layerwise posteriors for the original data $\{q(\mathbf{W}^l|\mathbf{W}_o^{1:l-1}, \mathcal{D}_o)\}_{l=1}^L$, and have since discarded the original data. Without access to the original data we can only obtain the conditional posteriors $\{q(\mathbf{W}^l|\mathbf{W}_o^{1:l-1}, \mathcal{D})\}_{l=1}^L$, and not what we need, which is $\{q(\mathbf{W}^l|\mathbf{W}_{o,u}^{1:l-1}, \mathcal{D})\}_{l=1}^L$. This is because conditioning on samples $\mathbf{W}_{o,u}^{1:l-1}$ requires propagating the full collection of data.

However, we can approximate what we need by updating just the last layer's posterior while preserving the existing previous layer posterior samples, giving us

$$q(\mathbf{W}|\mathcal{D}) \approx q(\mathbf{W}^L|\mathbf{W}_o^{1:l-1}, \mathcal{D}) \prod_{l=1}^{L-1} q(\mathbf{W}^l|\mathbf{W}_o^{1:l-1}, \mathcal{D}_o) \tag{52}$$

$$\propto p\left(\mathbf{Y}_u^L|\mathbf{W}^L, \mathbf{X}_u^{L-1}\right) q(\mathbf{W}^L|\mathbf{W}_o^{1:L-1}, \mathcal{D}_o) \prod_{l=1}^{L-1} q(\mathbf{W}^l|\mathbf{W}_o^{1:l-1}, \mathcal{D}_o) \tag{53}$$

We can interpret this as the first $L-1$ layers being a feature selector whose weights have been inferred from just the original data, while the weights of the prediction head (last layer) are updated in light of the new data by sequential Bayesian inference. Although this might seem like a tenuous approximation, we find it works well in practice. We found that the approximation deteriorates for increasingly deep architectures—this is unsurprising given that an increasingly small proportion of weights' posteriors get updated.

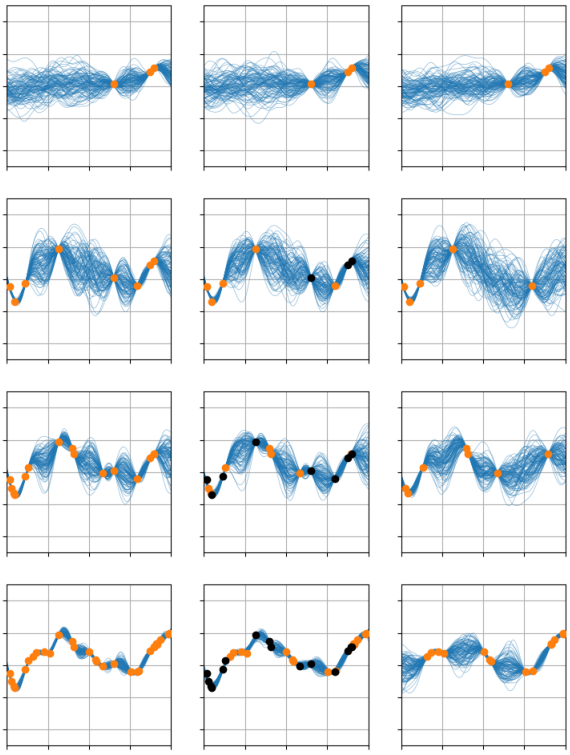
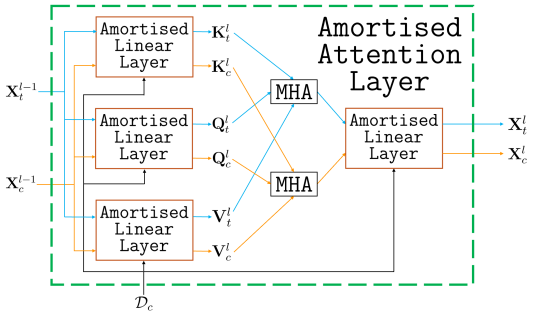


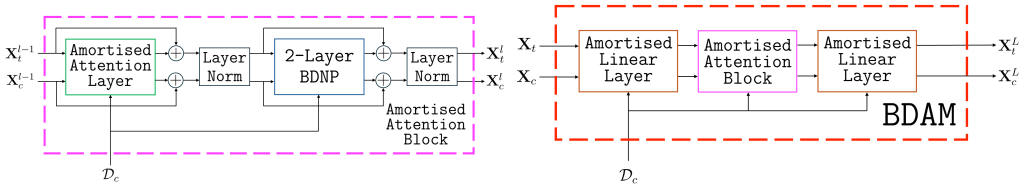
Figure 8: A demo of our online learning algorithm for the BDNP. Orange dots are context points, blue lines are posterior predictive samples, black dots are context points that we have lost access to. Each row incorporates more context data. The left-hand column corresponds to a BDNP given access to the *full* context set at each increment, the central column corresponds to our online learning algorithm, and the right-hand column corresponds to the BDNP given *only* the new context data.

E THE BAYESIAN DEEP ATTENTIVE MACHINE

E.1 ARCHITECTURE



(a) Amortised attention layer



(b) Amortised attention block

(c) BDAM

Figure 9: Computational diagrams of the amortised attention layer (a), amortised attention block (b), and BDAM (c). Due to the numerous crossing lines in (a), we colour code the context and target input data paths as orange and light blue respectively. Arbitrarily many amortised attention blocks can be stacked sequentially in the BDAM; our diagram shows the simplest possible BDAM architecture.

We see in Fig. 9(a) that amortised inference can be performed in an attention layer by using amortised linear layers in place of standard linear layers, where MHA is the usual multi-head dot-product attention mechanism acting on keys $\mathbf{K}$, queries $\mathbf{Q}$, and values $\mathbf{V}$. Similarly, in Fig. 9(b) we follow the standard approach (Vaswani et al., 2017) for constructing stackable attention *blocks* from attention layers, residual connections, layer norms, and 2-layer MLPs, but replacing each of the attention layer and MLP with their amortised counterparts. In Fig. 9(c) we show how amortised inference can be performed in a transformer by composing amortised linear layers and amortised attention blocks. We note that the resulting model can only be used in a somewhat unusual way for transformers; to map from test inputs $\mathbf{X}_t$ to predicted test outputs $\mathbf{Y}_t$ where attention is performed between the *test* inputs, and where the posterior over the transformer's weights is estimated from a context set.

E.2 LACK OF CONSISTENCY

As mentioned in the main text, the BDAM does not produce consistent predictive distributions over target outputs. As we shall see, it is the attention *between test inputs* that causes this. For any finite set of target locations $\mathbf{X}_{t_{1:n}}$, the joint distributions $p_{\mathbf{X}_{t_{1:n}}}(\mathbf{Y}_{t_{1:n}})$ and $p_{\mathbf{X}_{t_{1:m}}}(\mathbf{Y}_{t_{1:m}})$ for $m < n$ are *consistent* if

$$p_{\mathbf{X}_{t_{1:m}}}(\mathbf{Y}_{t_{1:m}}) = \int p_{\mathbf{X}_{t_{1:n}}}(\mathbf{Y}_{t_{1:n}}) \mathrm{d}\mathbf{Y}_{t_{m+1:n}}. \tag{54}$$

In the case of the BDAM, we are interested in joint distributions of the form

$$p\left(\mathbf{Y}_{t_{1:n}} | \mathcal{D}_c, \mathbf{X}_{t_{1:n}}\right) = \int p\left(\mathbf{Y}_{t_{1:n}} | \mathbf{W}, \mathbf{X}_{t_{1:n}}\right) p_\Psi\left(\mathbf{W} | \mathcal{D}_c\right) \mathrm{d}\mathbf{W} \tag{55}$$

where the weights posterior $p_\Psi(\mathbf{W}|\mathcal{D}_c)$ is approximated by the variational posterior $q(\mathbf{W}|\mathcal{D}_c) = \prod_{l=1}^{L} q(\mathbf{W}^l|\mathbf{W}^{1:l-1}, \mathcal{D}_c)$ that we developed in the paper and where the likelihood $p\left(\mathbf{Y}_{t_{1:n}} | \mathbf{W}, \mathbf{X}_{t_{1:n}}\right)$ is parameterised by a transformer $T_{\mathbf{W}}$ with weights $\mathbf{W}$. Plugging these terms in,

we verify that the BDAM does not produce consistent joint predictive distributions

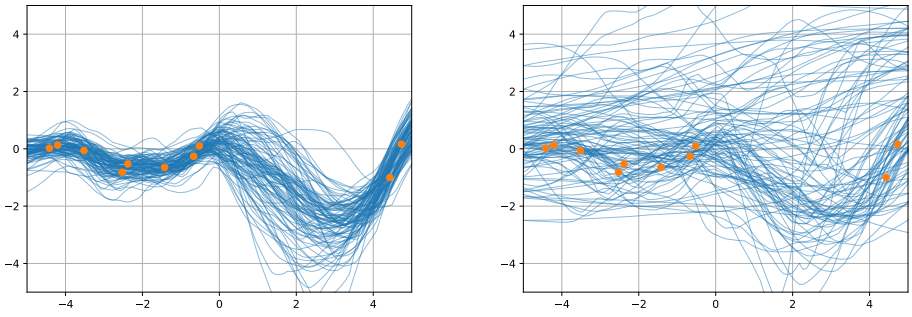
$$\int p\left(\mathbf{Y}_{t_{1:n}}|\mathcal{D}_c, \mathbf{X}_{t_{1:n}}\right) \mathrm{d}\mathbf{Y}_{t_{m+1:n}} = \int\int p\left(\mathbf{Y}_{t_{1:n}}|T_{\mathbf{W}}(\mathbf{X}_{t_{1:n}})\right) q\left(\mathbf{W}|\mathcal{D}_c\right) \mathrm{d}\mathbf{W} \mathrm{d}\mathbf{Y}_{t_{m+1:n}} \tag{56}$$

$$= \int p\left(\mathbf{Y}_{t_{1:m}}|T_{\mathbf{W}}(\mathbf{X}_{t_{1:n}})\right) q\left(\mathbf{W}|\mathcal{D}_c\right) \mathrm{d}\mathbf{W} \tag{57}$$

$$= p\left(\mathbf{Y}_{t_{1:m}}|\mathcal{D}_c, \mathbf{X}_{t_{1:n}}\right) \tag{58}$$

$$\neq p\left(\mathbf{Y}_{t_{1:m}}|\mathcal{D}_c, \mathbf{X}_{t_{1:m}}\right) \tag{59}$$

and visualise the consequences of this in Fig. 10. By contrast, the BDNP models target outputs as conditionally independent given a weights sample for the MLP $f_{\mathbf{W}}$. This means the BDNP's likelihood decomposes as $p\left(\mathbf{Y}_{t_{1:n}}|f_{\mathbf{W}}(\mathbf{X}_{t_{1:n}})\right) = \prod_{i=1}^{n} p\left(\mathbf{y}_{t_i}|f_{\mathbf{W}}(\mathbf{x}_{t_i})\right)$ which in turn ensures consistency of the model through the fact that $p\left(\mathbf{Y}_{t_{1:m}}|\mathcal{D}_c, \mathbf{X}_{t_{1:n}}\right) = p\left(\mathbf{Y}_{t_{1:m}}|\mathcal{D}_c, \mathbf{X}_{t_{1:m}}\right)$. By the Kolmogorov extension theorem (Tao, 2011), consistency of a collection of joint distributions is needed for them to define a valid stochastic process, and it is for this reason that the BDAM does *not* define a stochastic process. Note that the other condition required is exchangeability; both the BDNP and BDAM exhibit this through permutation invariance with respect to the context observations and permutation equivariance with respect to the target inputs. While the lack of consistency is generally a disadvantage, in settings for which the target inputs are always the same, this behaviour of the BDAM would not matter. An example of such a scenario is the common NP task of image completion via pixelwise meta-regression—in this case the target inputs are always the complete set of pixel coordinates.



(a) $p\left(\mathbf{Y}_{t_{1:m}}|\mathcal{D}_c, \mathbf{X}_{t_{1:m}}\right)$ (b) $p\left(\mathbf{Y}_{t_{1:m}}|\mathcal{D}_c, \mathbf{X}_{t_{1:n}}\right)$

Figure 10: Inconsistent predictive distributions of a BDAM trained on GP-prior generated data. The orange dots are context observations and the wiggly blue lines are predictive samples. $\mathbf{X}_{t_{1:m}}$ is generated as a uniform grid of 100 locations between -5 and 5 via `torch.linspace(-5.0, 5.0, 100)`, while $\mathbf{X}_{t_{1:n}}$ is generated by appending a value of 100.0 to $\mathbf{X}_{t_{1:m}}$ (i.e., $m = 100$ and $n = 101$). Observe that the joint distributions over $\mathbf{Y}_{t_{1:m}}$ are clearly very different. In other words, querying just a single extra target location can drastically change the BDAM's predictions, especially if the additional target location is OOD.