# Global inducing point variational posteriors for Bayesian neural networks and deep Gaussian processes

**Sebastian W. Ober** [1] **Laurence Aitchison** [2]

## Abstract

We consider the optimal approximate posterior over the top-layer weights in a Bayesian neural network for regression, and show that it exhibits strong dependencies on the lower-layer weights. We adapt this result to develop a correlated approximate posterior over the weights at all layers in a Bayesian neural network. We extend this approach to deep Gaussian processes, unifying inference in the two model classes. Our approximate posterior uses learned "global" inducing points, which are defined only at the input layer and propagated through the network to obtain inducing inputs at subsequent layers. By contrast, standard, "local", inducing point methods from the deep Gaussian process literature optimise a separate set of inducing inputs at every layer, and thus do not model correlations across layers. Our method gives state-of-the-art performance for a variational Bayesian method, without data augmentation or tempering, on CIFAR-10 of 86.7%, which is comparable to SGMCMC without tempering but with data augmentation (88% in Wenzel et al. 2020).[1]

## 1. Introduction

Deep models, formed by stacking together many simple layers, give rise to extremely powerful machine learning algorithms, from deep neural networks (DNNs) to deep Gaussian processes (DGPs) (Damianou & Lawrence, 2013). One approach to reason about uncertainty in these models is to use variational inference (VI) (Jordan et al., 1999). VI in Bayesian neural networks (BNNs) requires the user to specify a family of approximate posteriors over the weights, with the classical approach being independent Gaussian distributions over each individual weight (Hinton & Van Camp, 1993; Graves, 2011; Blundell et al., 2015). Later work has considered more complex approximate posteriors, for instance using a Matrix-Normal distribution as the approximate posterior for a full weight-matrix (Louizos & Welling, 2016; Ritter et al., 2018) and hierarchical variational inference (Louizos & Welling, 2017; Dusenberry et al., 2020). By contrast, DGPs use an approximate posterior defined over functions – the standard approach is to specify the inputs and outputs at a finite number of "inducing" points (Damianou & Lawrence, 2013; Salimbeni & Deisenroth, 2017).

Critically, these classical BNN and DGP approaches define approximate posteriors over functions that are independent across layers. An approximate posterior that factorises across layers is problematic, because what matters for a deep model is the overall input-output transformation for the full model, not the input-output transformation for individual layers. This raises the question of what family of approximate posteriors should be used to capture correlations across layers. One approach for BNNs would be to introduce a flexible "hypernetwork", used to generate the weights (Krueger et al., 2017; Pawlowski et al., 2017). However, this approach is likely to be suboptimal as it does not sufficiently exploit the rich structure in the underlying neural network. For guidance, we consider the optimal approximate posterior over the top-layer units in a deep network for regression. Remarkably, the optimal approximate posterior for the last-layer weights given the earlier weights can be obtained in closed form without choosing a restrictive family of distributions. In particular, the optimal approximate posterior is given by propagating the training inputs through lower layers to compute the top-layer representation, then using Bayesian linear regression to map from the top-layer representation to the outputs.

Inspired by this result, we use Bayesian linear regression to define a generic family of approximate posteriors for BNNs. In particular, we introduce learned "pseudo-outputs" at every layer, and compute the posterior over the weights by performing linear regression from the inputs (propagated

[1] Department of Engineering, University of Cambridge, Cambridge, UK [2] Department of Computer Science, University of Bristol, Bristol, UK. Correspondence to: Laurence Aitchison <laurence.aitchison@gmail.com>.

*Proceedings of the $38^{th}$ International Conference on Machine Learning*, PMLR 139, 2021. Copyright 2021 by the author(s).

[1] Reference implementation at: `github.com/LaurenceA/bayesfunc`

from lower layers) onto the pseudo-outputs. We reduce the burden of working with many training inputs by summarising the posterior using a small number of "inducing" points. We find that these approximate posteriors give excellent performance in the non-tempered, no-data-augmentation regime, with performance on datasets such as CIFAR-10 reaching 86.7%, comparable to SGMCMC witout tempering but with data augmentation (88%) (Wenzel et al., 2020). Our approach can be extended to DGPs, and we explore connections to the inducing point GP literature, showing that inference in the two classes of models can be unified.

Concretely, our contributions are:

- We propose an approximate posterior for BNNs based on Bayesian linear regression that naturally induces correlations between layers (Sec. 2.1.
- We provide an efficient implementation of this posterior for convolutional layers (Sec. 2.2.
- We introduce new BNN priors that allow for more flexibility with inferred hyperparameters (Sec. 2.3).
- We show how our approximate posterior can be naturally extended to DGPs, resulting in a unified approach for inference in BNNs and DGPs (Sec. 2.4).

## 2. Methods

To motivate our approximate posterior, we first consider the optimal top-layer posterior for a Bayesian neural network in the regression case. We consider neural networks with lower-layer weights $\{\mathbf{W}_\ell\}_{\ell=1}^{L}$, $\mathbf{W}_\ell \in \mathbb{R}^{N_{\ell-1} \times N_\ell}$, and output weights, $\mathbf{W}_{L+1} \in \mathbb{R}^{N_L \times N_{L+1}}$, where the activity, $\mathbf{F}_\ell$, at layer $\ell$ is given by,

$$\begin{aligned} \mathbf{F}_1 &= \mathbf{X}\mathbf{W}_1, \\ \mathbf{F}_\ell &= \phi\left(\mathbf{F}_{\ell-1}\right)\mathbf{W}_\ell \quad \text{ for } \ell \in \{2, \dots, L+1\}, \end{aligned} \tag{1}$$

where $\phi(\cdot)$ is an elementwise nonlinearity. The targets, $\mathbf{Y}$, depend on the neural-network outputs, $\mathbf{F}_{L+1} \in \mathbb{R}^{P \times N_{L+1}}$, according to a likelihood, $\mathrm{P}\left(\mathbf{Y}|\mathbf{F}_{L+1}\right)$. In the following derivations, we will focus on $\ell > 1$; corresponding expressions for the input layer can be obtained by replacing $\phi(\mathbf{F}_0)$ with the inputs, $\mathbf{X} \in \mathbb{R}^{P \times N_0}$. The prior over weights is independent across layers and output units (see Sec. 2.3 for the form of $\mathbf{S}_\ell$),

$$\begin{aligned} \mathrm{P}\left(\mathbf{W}_\ell\right) &= \textstyle\prod_{\lambda=1}^{N_\ell} \mathrm{P}\left(\mathbf{w}_\lambda^\ell\right), \\ \mathrm{P}\left(\mathbf{w}_\lambda^\ell\right) &= \mathcal{N}\left(\mathbf{w}_\lambda^\ell \middle| \mathbf{0}, \tfrac{1}{N_{\ell-1}}\mathbf{S}_\ell\right), \end{aligned} \tag{2}$$

where $\mathbf{w}_\lambda^\ell$ is a column of $\mathbf{W}_\ell$ representing all the input weights to unit $\lambda$ in layer $\ell$. To fit the parameters of the approximate posterior, $\mathrm{Q}\left(\{\mathbf{W}\}_{\ell=1}^{L+1}\right)$, we maximise the evidence lower bound (ELBO),

$$\begin{aligned} \mathcal{L} = \mathbb{E}_{\mathrm{Q}\left(\{\mathbf{W}\}_{\ell=1}^{L+1}\right)}\big[ &\log \mathrm{P}\left(\mathbf{Y}|\mathbf{X}, \{\mathbf{W}\}_{\ell=1}^{L+1}\right) \\ &+ \log \mathrm{P}\left(\{\mathbf{W}_\ell\}_{\ell=1}^{L+1}\right) - \log \mathrm{Q}\left(\{\mathbf{W}_\ell\}_{\ell=1}^{L+1}\right)\big] \end{aligned} \tag{3}$$

To build intuition about how to parameterise $\mathrm{Q}\left(\{\mathbf{W}\}_{\ell=1}^{L+1}\right)$, we consider the optimal $\mathrm{Q}\left(\mathbf{W}_{L+1}|\{\mathbf{W}_\ell\}_{\ell=1}^{L}\right)$ for any given $\mathrm{Q}\left(\{\mathbf{W}_\ell\}_{\ell=1}^{L}\right)$, i.e. the optimal top-layer posterior conditioned on the lower layers. We begin by simplifying the ELBO by incorporating terms that do not depend on $\mathbf{W}_{L+1}$ into $c(\{\mathbf{W}_\ell\}_{\ell=1}^{L})$,

$$\begin{aligned} \mathcal{L} = \mathbb{E}_{\mathrm{Q}\left(\{\mathbf{W}_\ell\}_{\ell=1}^{L+1}\right)}\big[ &\log \mathrm{P}\left(\mathbf{Y}, \mathbf{W}_{L+1}|\, \mathbf{X}, \{\mathbf{W}_\ell\}_{\ell=1}^{L}\right) \\ &- \log \mathrm{Q}\left(\mathbf{W}_{L+1}|\, \{\mathbf{W}_\ell\}_{\ell=1}^{L}\right) + c(\{\mathbf{W}_\ell\}_{\ell=1}^{L})\big]. \end{aligned} \tag{4}$$

Rearranging these terms (App. A), we find that all $\mathbf{W}_{L+1}$ dependence can be written in terms of the KL divergence between the approximate posterior of interest and the true posterior,

$$\begin{aligned} \mathcal{L} = \mathbb{E}_{\mathrm{Q}\left(\{\mathbf{W}_\ell\}_{\ell=1}^{L}\right)}\big[ \\ - \mathrm{D}_{\mathrm{KL}}\big(\mathrm{Q}(\mathbf{W}_{L+1}|\{\mathbf{W}_\ell\}_{\ell=1}^{L})||\mathrm{P}(\mathbf{W}_{L+1}|\mathbf{Y},\mathbf{X},\{\mathbf{W}_\ell\}_{\ell=1}^{L})\big) \\ + c(\{\mathbf{W}\}_{\ell=1}^{L})\big]. \end{aligned} \tag{5}$$

Thus, the optimal approximate posterior is the true last-layer posterior conditioned on the previous layers' weights,

$$\begin{aligned} \mathrm{Q}\left(\mathbf{W}_{L+1}|\{\mathbf{W}_\ell\}_{\ell=1}^{L}\right) &= \mathrm{P}\left(\mathbf{W}_{L+1}|\, \mathbf{Y}, \mathbf{X}, \{\mathbf{W}_\ell\}_{\ell=1}^{L}\right) \\ &\propto \mathrm{P}\left(\mathbf{Y}|\mathbf{W}_{L+1}, \mathbf{F}_L\right) \mathrm{P}\left(\mathbf{W}_{L+1}\right), \end{aligned} \tag{6}$$

where the final proportionality comes by applying Bayes theorem and exploiting the model's conditional independencies. For regression, the likelihood is Gaussian,

$$\begin{aligned} &\mathrm{P}\left(\mathbf{Y}|\mathbf{W}_{L+1}, \mathbf{F}_L\right) = \\ &\qquad \textstyle\prod_{\lambda=1}^{N_{L+1}} \mathcal{N}\left(\mathbf{y}_\lambda | \phi\left(\mathbf{F}_L\right)\mathbf{w}_\lambda^{L+1}, \mathbf{\Lambda}_{L+1}^{-1}\right), \end{aligned} \tag{7}$$

where $\mathbf{y}_\lambda$ is the value of a single output channel for all training inputs, and $\mathbf{\Lambda}_{L+1}$ is a precision matrix. Thus, the posterior is given in closed form by Bayesian linear regression (Rasmussen & Williams, 2006):

$$\begin{aligned} &\mathrm{Q}\left(\mathbf{W}_{L+1}|\{\mathbf{W}_\ell\}_{\ell=1}^{L}\right) = \\ &\qquad \textstyle\prod_{\lambda=1}^{N_{L+1}} \mathcal{N}\left(\mathbf{w}_\lambda^{L+1} \middle| \mathbf{\Sigma}\phi\left(\mathbf{F}_L\right)^T \mathbf{\Lambda}_{L+1}\mathbf{y}_\lambda, \mathbf{\Sigma}\right), \end{aligned} \tag{8}$$

where

$$\mathbf{\Sigma} = (N_L \mathbf{S}_{L+1}^{-1} + \phi\left(\mathbf{F}_L\right)^T \mathbf{\Lambda}_{L+1}\phi(\mathbf{F}_L))^{-1}. \tag{9}$$

While this result may be neither particularly surprising nor novel, it neatly highlights our motivation for the rest of the paper. In particular, it shows that for regression, we can always obtain the optimal conditional top-layer posterior for regression, which to the best of our knowledge has not been used before in BNN inference. Moreover, doing top-layer Bayesian linear regression based on the propagated features from the previous layers naturally introduces correlations between layers.

### 2.1. Defining the full approximate posterior with global inducing points and pseudo-outputs

We adapt the optimal top-layer approximate posterior above to give a scalable approximate posterior over the weights at all layers. To avoid propagating all training inputs through the network, which is intractable for large datasets, we instead propagate $M$ *global* inducing locations, $\mathbf{U}_0 \in \mathbb{R}^{M \times N_0}$,

$$\begin{aligned} &\mathbf{U}_1 = \mathbf{U}_0 \mathbf{W}_1, \\ &\mathbf{U}_\ell = \phi\left(\mathbf{U}_{\ell-1}\right) \mathbf{W}_\ell \quad \text{for } \ell = 2, \dots, L+1. \end{aligned} \tag{10}$$

Next, the optimal posterior requires outputs, $\mathbf{Y}$. However, no outputs are available at inducing locations for the output layer, let alone for intermediate layers. We thus introduce learned variational parameters to mimic the form of the optimal posterior. In particular, we use the product of the prior over weights and an "inducing-likelihood", $\mathcal{N}\left(\mathbf{v}_\lambda^\ell; \mathbf{u}_\lambda^\ell, \mathbf{\Lambda}_\ell^{-1}\right)$, representing noisy "pseudo-outputs" of the outputs of the linear layer at the inducing locations, $\mathbf{u}_\lambda^\ell = \phi\left(\mathbf{U}_{\ell-1}\right) \mathbf{w}_\lambda^\ell$. Substituting $\mathbf{u}_\lambda^\ell$ into the inducing-likelihood the approximate posterior becomes,

$$\begin{aligned} &\mathrm{Q}\left(\mathbf{W}_\ell \middle| \left\{\mathbf{W}_{\ell'}\right\}_{\ell'=1}^{\ell-1}\right) \\ &\qquad \propto \textstyle\prod_{\lambda=1}^{N_\ell} \mathcal{N}\left(\mathbf{v}_\lambda^\ell; \phi\left(\mathbf{U}_{\ell-1}\right) \mathbf{w}_\lambda^\ell, \mathbf{\Lambda}_\ell^{-1}\right) \mathrm{P}\left(\mathbf{w}_\lambda^\ell\right), \\ &\mathrm{Q}\left(\mathbf{W}_\ell \middle| \left\{\mathbf{W}_{\ell'}\right\}_{\ell'=1}^{\ell-1}\right) \\ &\qquad = \textstyle\prod_{\lambda=1}^{N_\ell} \mathcal{N}\left(\mathbf{w}_\lambda^\ell \middle| \mathbf{\Sigma}_\ell^{\mathbf{w}} \phi\left(\mathbf{U}_{\ell-1}\right)^T \mathbf{\Lambda}_\ell \mathbf{v}_\lambda^\ell, \mathbf{\Sigma}_\ell^{\mathbf{w}}\right), \\ &\mathbf{\Sigma}_\ell^{\mathbf{w}} = \left(N_{\ell-1} \mathbf{S}_\ell^{-1} + \phi\left(\mathbf{U}_{\ell-1}\right)^T \mathbf{\Lambda}_\ell \phi\left(\mathbf{U}_{\ell-1}\right)\right)^{-1}. \end{aligned} \tag{11}$$

where $\mathbf{v}_\lambda^\ell$ and $\mathbf{\Lambda}_\ell$ are variational parameters. For clarity, we have: $\mathbf{u}_\lambda^\ell, \mathbf{v}_\lambda^\ell \in \mathbb{R}^M$, so that $\mathbf{U}_{\ell-1} \in \mathbb{R}^{M \times N_{\ell-1}}$ and $\mathbf{V}_\ell \in \mathbb{R}^{M \times N_\ell}$ are formed by stacking these vectors, and $\mathbf{w}_\lambda^\ell \in \mathbb{R}^{N_{\ell-1}}$, with $\mathbf{S}_\ell, \mathbf{\Sigma}_\ell^{\mathbf{w}} \in \mathbb{R}^{N_{\ell-1} \times N_{\ell-1}}$ and $\mathbf{\Lambda}_\ell \in \mathbb{R}^{M \times M}$. Therefore, our full approximate posterior factorises as

$$\mathrm{Q}\left(\left\{\mathbf{W}_\ell\right\}_{\ell=1}^{L+1}\right) = \prod_{\ell=1}^{L+1} \mathrm{Q}\left(\mathbf{W}_\ell \middle| \left\{\mathbf{W}_{\ell'}\right\}_{\ell'=1}^{\ell-1}\right). \tag{12}$$

**Algorithm 1** Global inducing points for neural networks

```
**Parameters:** $\mathbf{U}_0$, $\{\mathbf{V}_\ell, \mathbf{\Lambda}_\ell\}_{\ell=1}^{L}$.
**Neural network inputs:** $\mathbf{F}_0$
**Neural network outputs:** $\mathbf{F}_{L+1}$
$\mathcal{L} \leftarrow 0$
**for** $\ell$ **in** $\{1, \dots, L+1\}$ **do**
    Compute the mean and cov. for weights at this layer
    $\mathbf{\Sigma}_\ell^{\mathbf{w}} = \left(N_{\ell-1} \mathbf{S}_\ell^{-1} + \phi\left(\mathbf{U}_{\ell-1}\right)^T \mathbf{\Lambda}_\ell \phi\left(\mathbf{U}_{\ell-1}\right)\right)^{-1}$
    $\mathbf{M}_\ell = \mathbf{\Sigma}_\ell^{\mathbf{w}} \phi\left(\mathbf{U}_{\ell-1}\right)^T \mathbf{\Lambda}_\ell \mathbf{V}_\ell$
    Sample the weights and compute the ELBO
    $\mathbf{W}_\ell \sim \mathcal{N}\left(\mathbf{M}_\ell, \mathbf{\Sigma}_\ell^{\mathbf{w}}\right) = \mathrm{Q}\left(\mathbf{W}_\ell \middle| \{\mathbf{W}_{\ell'}\}_{\ell'=1}^{\ell-1}\right)$
    $\mathcal{L} \leftarrow \mathcal{L} + \log \mathrm{P}\left(\mathbf{W}_\ell\right) - \log \mathcal{N}\left(\mathbf{W}_\ell | \mathbf{M}_\ell, \mathbf{\Sigma}_\ell^{\mathbf{w}}\right)$
    Propagate the inputs and inducing points using sampled weights,
    $\mathbf{U}_\ell = \phi\left(\mathbf{U}_{\ell-1}\right) \mathbf{W}_\ell$
    $\mathbf{F}_\ell = \phi\left(\mathbf{F}_{\ell-1}\right) \mathbf{W}_\ell$
**end for**
$\mathcal{L} \leftarrow \mathcal{L} + \log \mathrm{P}\left(\mathbf{Y} | \mathbf{F}_{L+1}\right)$
```

Substituting this approximate posterior and the factorised prior into the ELBO (Eq. 3), the full ELBO can be written,

$$\begin{aligned} \mathcal{L} = \mathbb{E}_{\mathrm{Q}\left(\{\mathbf{W}\}_{\ell=1}^{L+1}\right)} \Bigg[ &\log \mathrm{P}\left(\mathbf{Y}, | \mathbf{X}, \{\mathbf{W}\}_{\ell=1}^{L+1}\right) \\ &+ \sum_{\ell=1}^{L+1} \log \frac{\mathrm{P}\left(\mathbf{W}_\ell\right)}{\mathrm{Q}\left(\mathbf{W}_\ell | \left\{\mathbf{W}_\ell\right\}_{\ell=1}^{\ell-1}\right)} \Bigg]. \end{aligned} \tag{13}$$

where $\mathrm{P}\left(\mathbf{W}_\ell\right)$ is given by Eq. (2) and $\mathrm{Q}\left(\mathbf{W}_\ell | \left\{\mathbf{W}_\ell\right\}_{\ell=1}^{\ell-1}\right)$ is given by Eq. (11). The forms of the ELBO and approximate posterior suggest a sequential procedure to evaluate and subsequently optimise it: we alternate between sampling the weights using Eq. (11) and propagating the data and inducing points (Eq. 1 and Eq. 10; see Alg. 1). In summary, the parameters of the approximate posterior are the global inducing inputs, $\mathbf{U}_0$, and the pseudo-outputs and precisions at all layers, $\{\mathbf{V}_\ell, \mathbf{\Lambda}_\ell\}_{\ell=1}^{L+1}$. As each factor $\mathrm{Q}\left(\mathbf{W}_\ell \middle| \{\mathbf{W}_{\ell'}\}_{\ell'=1}^{\ell-1}\right)$ is Gaussian, these parameters can be optimised using standard reparameterised variational inference (Kingma & Welling, 2013; Rezende et al., 2014) in combination with the Adam optimiser (Kingma & Ba, 2014) (Appendix B). Importantly, by placing inducing inputs on the training data (i.e. $\mathbf{U}_0 = \mathbf{X}$), and setting $\mathbf{v}_\lambda^\ell = \mathbf{y}_\lambda$ this approximate posterior matches the optimal top-layer posterior (Eq. 6). Finally, we note that while this posterior is conditionally Gaussian, the full posterior over all $\{\mathbf{W}_\ell\}_{\ell=1}^{L+1}$ is non-Gaussian, and is thus potentially more flexible than a full-covariance Gaussian defined jointly over all weights at all layers.

### 2.2. Efficient convolutional Bayesian linear regression

The previous sections were valid for a fully connected network. The extension to convolutional networks is straightforward in principle: we transform the convolution into a matrix multiplication by treating each patch as a separate input feature-vector, flattening the spatial and channel dimensions together into a single vector. Thus, the feature-vectors have length `in_channels` × `kernel_width` × `kernel_height`, and the matrix $\mathbf{U}_\ell$ contains `patches_per_image` × `minibatch` patches. Likewise, we now have inducing outputs, $\mathbf{v}_\lambda^\ell$, at each location in all the inducing images, so this again has length `patches_per_image` × `minibatch`. After explicitly extracting the patches, we can straightforwardly apply standard Bayesian linear regression.

However, explicitly extracting image patches is very memory intensive in a DNN. If we consider a standard convolution with a $3 \times 3$ convolutional kernel, then there is a $3 \times 3$ patch centred at each pixel in the input image, meaning a factor of $9$ increase in memory consumption. Instead, we note that computing the matrices required for linear regression, $\phi\left(\mathbf{U}_{\ell-1}\right)^T \mathbf{\Lambda}_\ell \phi\left(\mathbf{U}_{\ell-1}\right)$ and $\phi\left(\mathbf{U}_\ell\right)^T \mathbf{\Lambda}_\ell \mathbf{V}_\ell$, does not require explicit extraction of image-patches. Instead, these matrices can be computed by taking the autocorrelation of the image/feature map, i.e. a convolution operation where we treat the image/feature map, as *both* the inputs and the weights (Appendix C for details).

### 2.3. Priors

We consider four priors in this work, which we refer to using the class names in the bayesfunc library published alongside this paper. We are careful to ensure that all parameters in the model have a prior and approximate posterior, which is necessary to ensure that ELBOs are comparable across models.

First, we consider a Gaussian prior with fixed scale, NealPrior, so named because it is necessary to obtain meaningful results when considering infinite networks (Neal, 1996),

$$\mathbf{S}_\ell = \mathbf{I}, \tag{14}$$

though it bears strong similarities to the "He" initialisation (He et al., 2015). NealPrior is defined so as to ensure that the activations retain a sensible scale as they propagate through the network. We compare this to the standard $\mathcal{N}(0,1)$ (StandardPrior), which causes the activations to increase exponentially as they propagate through network layers (see Eq. 2):

$$\mathbf{S}_\ell = N_{\ell-1}\mathbf{I}. \tag{15}$$

Next, we consider ScalePrior, which defines a prior and approximate posterior over the scale,

$$\mathbf{S}_\ell = \tfrac{1}{s_\ell}\mathbf{I} \tag{16}$$

$$\mathrm{P}\left(s_\ell\right) = \text{Gamma}\left(s_\ell; 2, 2\right) \tag{17}$$

$$\mathrm{Q}\left(s_\ell\right) = \text{Gamma}\left(s_\ell; 2+\alpha_\ell, 2+\beta_\ell\right) \tag{18}$$

where here we parameterise the Gamma distribution with the shape and rate parameters, and $\alpha_\ell$ and $\beta_\ell$ are non-negative learned parameters of the approximate posterior over $s_\ell$. Finally, we consider SpatialIWPrior, which allows for spatial correlations in the weights (e.g. see Fortuin et al., 2021, for a more restrictive spatial prior over weights). In particular, we take the covariance to be the Kronecker product of an identity matrix over channel dimensions, and a Wishart-distributed matrix, $\mathbf{L}_\ell^{-1}$, over the spatial dimensions,

$$\begin{aligned}
\mathbf{S}_\ell &= \mathbf{I} \otimes \mathbf{L}_\ell^{-1} \\
\mathrm{P}\left(\mathbf{L}_\ell\right) &= \mathcal{W}^{-1}\left(\mathbf{L}_\ell; (N_{\ell-1}{+}1)\,\mathbf{I} \qquad, N_{\ell-1}{+}1\right) \\
\mathrm{Q}\left(\mathbf{L}_\ell\right) &= \mathcal{W}^{-1}\left(\mathbf{L}_\ell; (N_{\ell-1}{+}1)\,\mathbf{I} + \mathbf{\Psi}, N_{\ell-1}{+}1{+}\nu\right)
\end{aligned} \tag{19}$$

where $\mathcal{W}^{-1}$ is the inverse-Wishart distribution, and the non-negative real number, $\nu$, and the positive definite matrix, $\mathbf{\Psi}$, are learned parameters of the approximate posterior (see Appendix D).

### 2.4. Extension to DGPs

It is a remarkable but underappreciated fact that BNNs are special cases of DGPs, with a particular choice of kernel (Louizos & Welling, 2016; Aitchison, 2019). Combining Eqs. (1) and (2),

$$\begin{aligned}
\mathrm{P}\left(\mathbf{F}_\ell | \mathbf{F}_{\ell-1}\right) &= \textstyle\prod_{\lambda=1}^{N_\ell} \mathcal{N}\left(\mathbf{f}_\lambda^\ell \middle| \mathbf{0}, \mathbf{K}\left(\mathbf{F}_{\ell-1}\right)\right) \\
\mathbf{K}\left(\mathbf{F}_{\ell-1}\right) &= \frac{1}{N_{\ell-1}} \phi\left(\mathbf{F}_{\ell-1}\right) \mathbf{S}_\ell \phi\left(\mathbf{F}_{\ell-1}\right)^T.
\end{aligned} \tag{20}$$

Here, we generalise our approximate posterior to the DGP case and link to the DGP literature. In a DGP there are no weights; instead we work directly with inducing outputs $\{\mathbf{U}_\ell\}_{\ell=1}^{L+1}$,

$$\mathrm{P}\left(\mathbf{U}_\ell | \mathbf{U}_{\ell-1}\right) = \textstyle\prod_{\lambda=1}^{N_\ell} \mathcal{N}\left(\mathbf{u}_\lambda^\ell \middle| \mathbf{0}, \mathbf{K}\left(\mathbf{U}_{\ell-1}\right)\right), \tag{21}$$

Note that here, we take the "global" inducing approach of using the inducing outputs from the previous layer, $\mathbf{U}_{\ell-1}$ as the inducing inputs for the next layer. In this case, we need only learn the original inducing inputs, $\mathbf{U}_0$. This contrasts with the standard "local" inducing formulation, (as in Salimbeni & Deisenroth, 2017), which learns separate inducing inputs at every layer, $\mathbf{Z}_{\ell-1}$, giving $\mathrm{P}\left(\mathbf{U}_\ell | \mathbf{Z}_{\ell-1}\right) = \prod_{\lambda=1}^{N_\ell} \mathcal{N}\left(\mathbf{u}_\lambda^\ell \middle| \mathbf{0}, \mathbf{K}\left(\mathbf{Z}_{\ell-1}\right)\right)$.

As usual in DGPs (Salimbeni & Deisenroth, 2017), the approximate posterior over $\mathbf{U}_\ell$ induces an approximate posterior on $\mathbf{F}_\ell$ through the prior correlations. However, it is

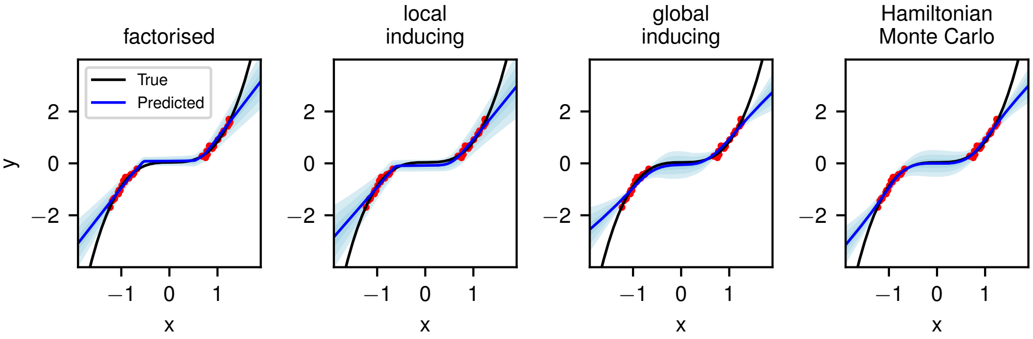

*Figure 1.* Predictive distributions on the toy dataset. Shaded regions represent one standard deviation.

important to remember that underneath the tractable distributions in Eqs. (20) and (21), there is an infinite dimensional GP-distributed function, $\mathcal{F}_\ell$, such that $\mathbf{F}_\ell = \mathcal{F}_\ell(\mathbf{F}_{\ell-1})$. Standard local inducing point methods specify a factorised approximate posterior over $\mathcal{F}_\ell$ by specifying the function's inducing outputs, $\mathbf{U}_\ell = \mathcal{F}_\ell\left(\mathbf{Z}_{\ell-1}\right)$, at a finite number of inducing input locations, $\mathbf{Z}_{\ell-1}$. Importantly, the approximate posterior over a function, $\mathcal{F}_\ell$, depends only on $\mathbf{Z}_{\ell-1}$, and $\mathbf{U}_\ell$. Thus, standard, local inducing, DGP approaches (e.g. Salimbeni & Deisenroth, 2017), give a layerwise-independent approximate posterior over $\{\mathcal{F}_\ell\}_{\ell=1}^{L+1}$, as they treat the inducing inputs, $\{\mathbf{Z}_{\ell-1}\}_{\ell=1}^{L+1}$, as fixed, learned parameters and use a layerwise-independent approximate posterior over $\{\mathbf{U}_\ell\}_{\ell=1}^{L+1}$ (Appendix G).

Next, we need to choose the approximate posterior on $\{\mathbf{U}_\ell\}_{\ell=1}^{L+1}$. However, if our goal is to introduce dependence across layers, it seems inappropriate to use the standard layerwise-independent approximate posterior over $\{\mathbf{U}_\ell\}_{\ell=1}^{L+1}$. Indeed, in Appendix G, we show that such a posterior implies functions in non-adjacent layers (e.g. $\mathcal{F}_\ell$ and $\mathcal{F}_{\ell+2}$) are marginally independent, even with global inducing points.

To obtain more appropriate approximate posteriors, we derive the optimal top-layer posterior for DGPs, which involves GP regression from activations propagated from lower layers onto the output data (Appendix F). Inspired by the form of the optimal posterior we again define an approximate posterior by taking the product of the prior and a "inducing-likelihood",

$$\begin{aligned}
\mathrm{Q}\left(\mathbf{U}_\ell|\mathbf{U}_{\ell-1}\right) &\propto \textstyle\prod_{\lambda=1}^{N_\ell} \mathrm{Q}\left(\mathbf{u}_\lambda^\ell\right) \mathcal{N}\left(\mathbf{v}_\lambda^\ell | \mathbf{u}_\lambda^\ell, \mathbf{\Lambda}_\ell^{-1}\right) \\
\mathrm{Q}\left(\mathbf{U}_\ell|\mathbf{U}_{\ell-1}\right) &= \textstyle\prod_{\lambda=1}^{N_\ell} \mathcal{N}\left(\mathbf{u}_\lambda^\ell | \mathbf{\Sigma}_\ell^{\mathbf{u}} \mathbf{\Lambda}_\ell \mathbf{v}_\lambda^\ell, \mathbf{\Sigma}_\ell^{\mathbf{u}}\right), \\
\mathbf{\Sigma}_\ell^{\mathbf{u}} &= \left(\mathbf{K}^{-1}\left(\mathbf{U}_{\ell-1}\right) + \mathbf{\Lambda}_\ell\right)^{-1},
\end{aligned} \tag{22}$$

where $\mathbf{v}_\lambda^\ell$ and $\mathbf{\Lambda}_\ell^{-1}$ are learned parameters, and in our global inducing method, the inducing inputs, $\mathbf{U}_{\ell-1}$, are propagated from lower layers (Eq. 21). Importantly, setting the inducing inputs to the training data and $\mathbf{v}_\lambda^{L+1} = \mathbf{y}_\lambda$, the approximate posterior captures the optimal top-layer posterior for regression (Appendix F). Under this approximate posterior, dependencies in $\mathbf{U}_\ell$ naturally arise across all layers, and hence there are dependencies between functions $\mathcal{F}_\ell$ at all layers (Appendix G).

In summary, we propose an approximate posterior over inducing outputs that takes the form

$$\mathrm{Q}\left(\{\mathbf{U}_\ell\}_{\ell=1}^{L+1}\right) = \prod_{l=1}^{L+1} \mathrm{Q}\left(\mathbf{U}_\ell|\mathbf{U}_{\ell-1}\right). \tag{23}$$

As before, the parameters of this approximate posterior are the global inducing inputs, $\mathbf{U}_0$, and the pseudo-outputs and precisions at all layers, $\{\mathbf{V}_\ell, \mathbf{\Lambda}_\ell\}_{\ell=1}^{L+1}$. The full ELBO, (see App.E for further details), takes the form

$$\begin{aligned}
\mathcal{L} = \mathbb{E}_{\mathrm{Q}\left(\{\mathbf{F}_\ell,\mathbf{U}_\ell\}_{\ell=1}^{L+1}|\mathbf{X},\mathbf{U}_0\right)}\Bigg[&\log \mathrm{P}\left(\mathbf{Y}|\mathbf{F}_{L+1}\right) \\
&+ \sum_{\ell=1}^{L+1} \log \frac{\mathrm{P}\left(\mathbf{U}_\ell|\mathbf{U}_{\ell-1}\right)}{\mathrm{Q}\left(\mathbf{U}_\ell|\mathbf{U}_{\ell-1}\right)}\Bigg].
\end{aligned} \tag{24}$$

where $\mathrm{P}\left(\mathbf{U}_\ell|\mathbf{U}_{\ell-1}\right)$ is given by Eq. (21) and $\mathrm{Q}\left(\mathbf{U}_\ell|\mathbf{U}_{\ell-1}\right)$ is given by Eq. (22).

We provide a full description of our method as applied to DGPs in App. E.

### 2.5. Asymptotic complexity

In the deep GP case, the complexity for global inducing is exactly that of standard inducing point Gaussian processes, i.e. $\mathcal{O}(M^3 + PM^2)$ where $M$ is the number of inducing points, and $P$ can be taken to be the number of training inputs, or the size of a minibatch, as appropriate. The first term, $M^3$, comes from computing and sampling the posterior over $\mathbf{U}_\ell$ based on the inducing points (e.g. inverting the covariance). The second term, and $PM^2$ comes from computing the implied distribution over $\mathbf{F}_\ell$.

In the fully-connected BNN case, we have three terms, $\mathcal{O}(N^3 + MN^2 + PN^2)$. The first term, $N^3$, where $N$

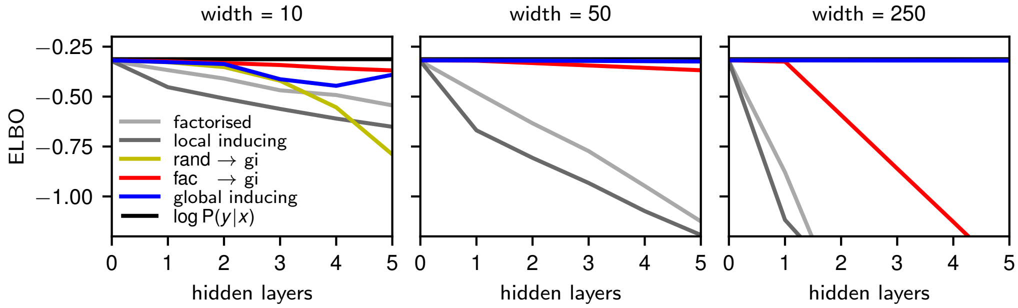

*Figure 2.* ELBO for different approximate posteriors as we change network depth/width on a dataset generated using a linear Gaussian model. The rand → gi line lies behind the global inducing line in width $= 50$ and width $= 250$.

corresponds to the width of the network, arises from taking the inverse of the covariance matrix in Eq. (11), but is also the complexity e.g. for propagating the inducing points from layer to the next (Eq. 10). The second term, $MN^2$, comes from computing that covariance in Eq. (11), by taking the product of input features with themselves. The third term $PN^2$ comes from multiplying the training inputs/minibatch by the sampled inputs (Eq. 1).

## 3. Results

We describe our experiments and results to assess the performance of global inducing points ('gi') against local inducing points ('li') and the fully factorised ('fac') approximation family. We additionally consider models where we use one method up to the last layer and another for the last layer, which may have computational advantages; we denote such models 'method1 → method2'.

### 3.1. Uncertainty in 1D regression

We demonstrate the use of local and global inducing point methods in a toy 1-D regression problem, comparing it with fully factorised VI and Hamiltonian Monte Carlo (HMC; (Neal et al., 2011)). Following Hernández-Lobato & Adams (2015), we generate 40 input-output pairs $(x, y)$ with the inputs $x$ sampled i.i.d. from $\mathcal{U}([-4, -2] \cup [2, 4])$ and the outputs generated by $y = x^3 + \epsilon$, where $\epsilon \sim \mathcal{N}(0, 3^2)$. We then normalised the inputs and outputs. Note that we have introduced a 'gap' in the inputs, following recent work (Foong et al., 2019b; Yao et al., 2019; Foong et al., 2019a) that identifies the ability to express 'in-between' uncertainty as an important quality of approximate inference algorithms. We evaluated the inference algorithms using fully-connected BNNs with 2 hidden layers of 50 ReLU hidden units, using the NealPrior. For the inducing point methods, we used 100 inducing points per layer.

The predictive distributions for the toy experiment can be seen in Fig. 1. We observe that of the variational methods, the global inducing method produces predictive distributions closest to HMC, with good uncertainty in the gap. Meanwhile, factorised and local inducing fit the training data, but do not produce reasonable error bars, demonstrating an important limitation of methods lacking correlation structure between layers.

We provide additional experiments looking at the effect of the number of inducing points in Appendix I, and experiments looking at compositional uncertainty (Ustyuzhaninov et al., 2020) in both BNNs and DGPs for 1D regression in Appendix J.

### 3.2. Depth-dependence in deep linear networks

The lack of correlations between layers might be expected to become more problematic in deeper networks. To isolate the effect of depth on different approximate posteriors, we considered deep linear networks trained on data generated from a toy linear model: 5 input features were mapped to 1 output feature, where the 1000 training and 100 test inputs are drawn IID from a standard Gaussian, and the true outputs are drawn using a weight-vector drawn IID from a Gaussian with variance $1/5$, and with noise variance of $0.1$. We could evaluate the model evidence under the true data generating process which forms an upper bound (in expectation) on the model evidence and ELBO for all models.

We found that the ELBO for methods that factorise across layers — factorised and local inducing — drops rapidly as networks get deeper and wider (Fig. 2). This is undesirable behaviour, as we know that wide, deep networks are necessary for good performance on difficult machine learning tasks. In contrast, we found that methods with global inducing points at the last layer decay much more slowly with depth, and perform better as networks get wider. Remarkably, global-inducing points gave good performance even with lower-layer weights drawn at random from the prior, which is not possible for any method that factorises across layers. We believe that fac → gi performed poorly at width $= 250$ due to optimisation issues as rand → gi performs better yet is a special case of fac → gi.

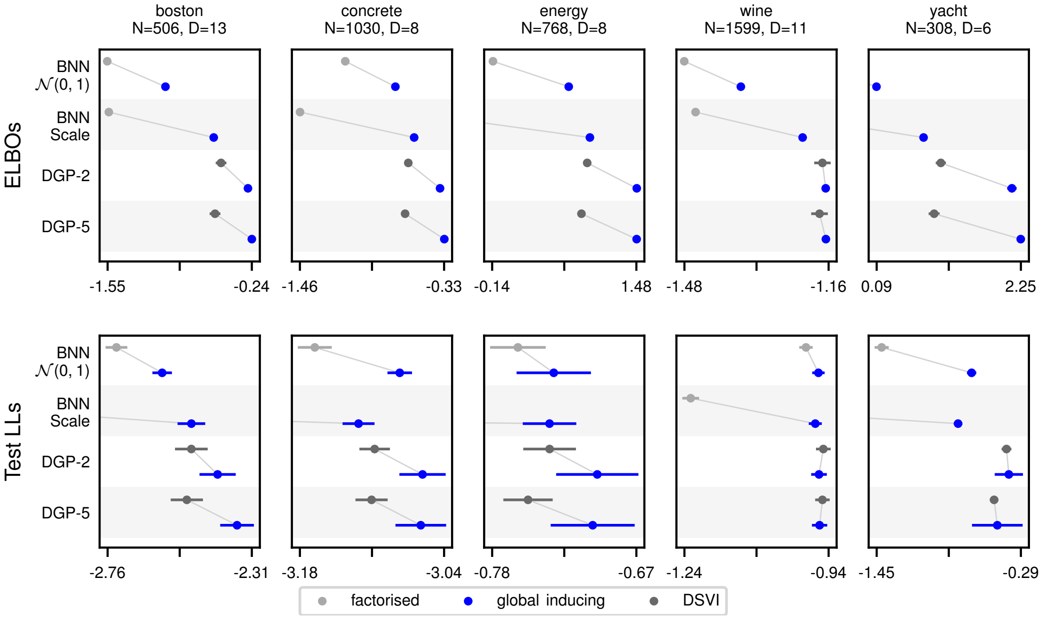

*Figure 3.* Average test log likelihoods for BNNs on the UCI datasets (in nats). Error bars represent one standard error. Shading represents different priors. We connect the factorised models with the fac → gi models with a thin grey line as an aid for easier comparison. Further to the right is better.

### 3.3. Regression benchmark: UCI

We benchmark our methods on the UCI datasets in Hernández-Lobato & Adams (2015), popular benchmark regression datasets for BNNs and DGPs. Following the standard approach (Gal & Ghahramani, 2015), each dataset uses 20 train-test 'splits' (except for protein with 5 splits) and the inputs and outputs are normalised to have zero mean and unit standard deviation. We focus on the five smallest datasets, as we expect Bayesian methods to be most relevant in small-data settings (see App. K andM for all datasets). We consider two-layer fully-connected ReLU networks, using fully factorised and global inducing approximating families, as well as two- and five-layer DGPs with doubly-stochastic variational inference (DSVI) (Salimbeni & Deisenroth, 2017) and global inducing. For the BNNs, we consider the standard $\mathcal{N}(0, 1)$ prior and ScalePrior.

We display ELBOs and average test log likelihoods for the un-normalised data in Fig. 3, where the dots and error bars represent the means and standard errors over the test splits, respectively. We observe that global inducing obtains better ELBOs than factorised and DSVI in almost every case, indicating that it does indeed approximate the true posterior better (since the ELBO is the marginal likelihood minus the KL to the posterior). While this is the case for the ELBOs, this does not always translate to a better test log likelihood due to model misspecification, as we see that occasionally DSVI outperforms global inducing by a very small margin. The very poor results for factorised on ScalePrior indicate that it has difficulty learning useful prior hyperparameters for prediction, which is due to the looseness of its bound to the marginal likelihood. We provide experimental details, as well as additional results with additional architectures, priors, datasets, and RMSEs, in Appendices K and M, for BNNs and DGPs, respectively.

### 3.4. Convolutional benchmark: CIFAR-10

For CIFAR-10, we considered a ResNet-inspired model consisting of conv2d-relu-block-avgpool2-block-avgpool2-block-avgpool-linear, where the ResNet blocks consisted of a shortcut connection in parallel with conv2d-relu-conv2d-relu, using 32 channels in all layers. In all our experiments, we used no data augmentation and 500 inducing points. Our training scheme (see App. O) ensured that our results did not reflect a 'cold posterior' (Wenzel et al., 2020). Our results are shown in Table 1. We achieved remarkable performance of 86.7% predictive accuracy, with global inducing points used for all layers, and with a spatial inverse Wishart prior on the weights. These results compare very favourably with comparable Bayesian approaches, i.e. those without data augmentation or posterior sharpening: past work with deep GPs obtained 80.3% (Shi et al., 2019), and work using infinite-width neural networks to define a GP obtained 81.4% accuracy (Li et al., 2019). Remarkably, with only 500 inducing points we are approaching the accuracy of sampling-based methods (Wenzel et al., 2020), which are in principle able to more closely approximate the true posterior. Furthermore, we see that global inducing performs the best

*Table 1.* CIFAR-10 classification accuracy. The first block shows our main results without data augmentation or tempering with SpatialIWPrior, (with ScalePrior in brackets). The next block shows our results with data augmentation and tempering on with a larger ResNet18 with SpatialIWPrior. The next block shows comparable past results, from GPs and BNNs. The final block show non-comparable (sampling-based) methods. Dashes indicate that the figures were either not reported, are not applicable. The time is reported per epoch with ScalePrior and for MNIST, rather than CIFAR-10 because of a known performance bug in the convolutions required in Sec. 2.2 with 32 × 32 (and above) images `https://github.com/pytorch/pytorch/issues/35603`.

| | | test log like. | accuracy (%) | ELBO | time |
|---|---|---|---|---|---|
| no tempering or data augmentation | factorised | -0.58 (-0.66) | 80.27 (77.65) | -1.06 (-1.12) | 19 s |
| | local inducing | -0.62 (-0.60) | 78.96 (79.46) | -0.84 (-0.88) | 33 s |
| | fac → gi | -0.49 (-0.56) | 83.33 (81.72) | -0.91 (-0.96) | 25 s |
| | **global inducing** | **-0.40 (-0.43)** | **86.70 (85.73)** | **-0.68 (-0.75)** | 65 s |
| with tempering and data augmentation | factorised | -0.39 | 87.52 | — | — |
| | **fac → gi** | **-0.24** | **92.41** | — | — |
| VI prior work | Shi et al. (2019) | — | 80.30% | — | |
| | Li et al. (2019) | — | 81.40% | — | |
| | Shridhar et al. (2019) | — | 73% | — | |
| sampling prior work | Wenzel et al. (2020) | $-0.35$ | 88.50% | — | |

in terms of ELBO (per datapoint) by a wide margin, demonstrating that it gets far closer to the true posterior than the other methods. We provide additional results on uncertainty calibration and out-of-distribution detection in Appendix L.

Finally, while we have focused this work on achieving good results with a fully principled, Bayesian approach, we briefly consider training a full ResNet-18 (He et al., 2016) using more popular techniques such as data augmentation and tempering. While data augmentation and tempering are typically viewed as clouding the Bayesian perspective, there is work attempting to formalise both within the context of modified probabilistic generative models (Aitchison, 2020; Nabarro et al., 2021). Using a cold posterior with a factor of 20 reduction on the KL term, with horizontal flipping and random cropping, we obtained a test accuracy of $87.52\%$ and a test log likelihood of $-0.39$ nats with a standard fully factorised Gaussian posterior using SpatialIWPrior. Using fac → gi, we obtain a significant improvement of $92.41\%$ test accuracy and a test log likelihood of $-0.24$ nats. We attempted to train a full global inducing model; however, we encountered difficulties in scaling the method to the large widths of ResNet-18 layers. We believe this presents an avenue for fruitful future work in scaling global inducing point posteriors.

## 4. Related work

Louizos & Welling (2016) attempted to use pseudo-data along with matrix variate Gaussians to form an approximate posterior for BNNs; however, they restricted their analysis to BNNs, and it is not clear how their method can be applied to DGPs. Their approach factorises across layers, thus missing the important layerwise correlations that we obtain. Moreover, they encountered an important limitation: the BNN prior implies that $\mathbf{U}_\ell$ is low-rank and it is difficult to design an approximate posterior capturing this constraint. As such, they were forced to use $M < N_\ell$ inducing points, which is particularly problematic in the convolutional, global-inducing case where there are many patches (points) in each inducing image input.

Note that some work on BNNs reports better performance on datasets such as CIFAR-10. However, to the best of our knowledge, no variational Bayesian method outperforms ours without modifying the BNN model or some form of posterior tempering (Wenzel et al., 2020), where the KL term in the ELBO is down-weighted relative to the likelihood (Zhang et al., 2017; Bae et al., 2018; Osawa et al., 2019; Ashukha et al., 2020), which often increases the test accuracy. However, tempering clouds the Bayesian perspective, as the KL to the posterior is no longer minimised and the resulting objective is no longer a lower bound on the marginal likelihood. By contrast, we use the untempered ELBO, thereby retaining the full Bayesian perspective. Dusenberry et al. (2020) report better performance on CIFAR-10 without tempering, but only perform variational inference over a rank-1 perturbation to the weights, and maximise over all the other parameters, which may risk overfitting (Ober et al., 2021). Our approach retains a full-rank parameterisation of the weight matrices.

Ustyuzhaninov et al. (2020) attempted to introduce dependence across layers in a deep GP by coupling inducing inputs to pseudo-outputs, which they term “inducing points as inducing locations”. However, as described above, the choice of approximate posterior over $\mathbf{U}_\ell$ is also critical.

They used the standard approximate posterior that is independent across layers, meaning that while functions in adjacent layers were marginally dependent, the functions for non-adjacent layers were independent (Appendix G). By contrast, our approximate posteriors have marginal dependencies across $\mathbf{U}_\ell$ and functions at all layers, and are capable of capturing the optimal top-layer posterior.

Contemporaneous work has a similar motivation but is ultimately very different (Lindinger et al., 2020). They use a joint multivariate Gaussian (with structured covariance) for the approximate posterior for all inducing outputs at all layers (with local inducing inputs). In contrast, our approximate posterior is only Gaussian at a single layer (conditioned on the input to that layer). The approximate posterior over all layers is *not* Gaussian (see Eq. 22, where the covariance at layer $\ell$ depends on $\mathbf{U}_{\ell-1}$), which allows us to capture the optimal top-layer posterior. Indeed, we have linear computational complexity in depth, whereas their approach is cubic (their Sec 3.2).

Karaletsos & Bui (2020) propose a very different hybrid GP-BNN which uses an RBF-GP prior over *weights*, whereas we use standard DGP models (which do not have any weights) and BNN models with IID weight priors. Our BNN and DGP map from global-inducing inputs defined only at the input layer onto *activations*, while their GPs map from inducing inputs in a "neuron-embedding" space onto *weights*. Unlike Karaletsos & Bui (2020), our approach scales directly to large ResNets, and allows us to capture the optimal top-layer posterior. Note they use "global" vs. "local" for different *priors*, while we use it for different *approximate posteriors*. While their models allow for the possibility of correlations between layers, these arise from modifying the prior structure, which are then reflected in the approximate posterior. In contrast, we use standard Gaussian priors over weights that are uncorrelated across layers, and our across-layer dependencies arise only in the approximate posterior.

Alternative approaches to introducing more flexibility and dependence in approximate posteriors are to use hierarchical variational inference (HVI), i.e. to introduce auxiliary random variables which can be integrated out to give a richer variational posterior (Ranganath et al., 2016; Louizos & Welling, 2017; Sobolev & Vetrov, 2019). This has been used in a number of prior works to introduce correlations between parameters (Dusenberry et al., 2020; Louizos & Welling, 2017; Ghosh & Doshi-Velez, 2017). As we do not introduce auxiliary variables, our approach is not HVI. However, our approximate posterior does include dependencies across latent variables and is thus an instance of structured stochastic variational inference (Hoffman & Blei, 2015).

The general idea of marginalising over the Gaussian process latent function in order to compute the marginal likelihood is common in the Gaussian process literature (e.g. Heinonen et al., 2016), and is related to, but very different from our approach of using the top-layer optimal posterior to inspire an approximate posterior for the weights of a Bayesian neural network, or the Gaussian process function.

Finally, very recent work has suggested an alternative approach to unifying inference in deep NN and DGP models, by noting that these models can be equivalently written in terms of distributions over positive-definite Gram matrices, rather than working with features or weights like here (Aitchison et al., 2020). Their motivation was explicitly to account for rotation/permutation symmetries in the posterior over neural network weights and over DGP features. Interestingly, our global inducing approximations partially account for these symmetries: the posteriors over features at the next layer: assuming isotropic kernels that depend only on distance, $\mathbf{U}_\ell$ are invariant to rotations in the input, $\mathbf{U}_{\ell-1}$, and this carries over to rotations of the inputs for BNNs. However, they are as of yet unable to scale to large convolutional architectures.

## 5. Conclusions

We derived optimal top-layer variational approximate posteriors for BNNs and deep GPs, and used them to develop generic, scalable approximate posteriors. These posteriors make use of *global* inducing points, which are learned only at the bottom layer and are propagated through the network. This leads to extremely flexible posteriors, which even allow the lower-layer weights to be drawn from the prior. We showed that these global inducing variational posteriors lead to improved performance with better ELBOs, and state-of-the-art performance for variational BNNs on CIFAR-10.

### Acknowledgments

We would like to thank the anonymous reviewers for their valuable feedback which greatly improved the manuscript. We thank David R. Burt, Andrew Y. K. Foong, Siddharth Swaroop, Adrià Garriga-Alonso, Vidhi Lalchand, and Carl E. Rasmussen for valuable discussions and feedback. SWO acknowledges the support of the Gates Cambridge Trust in funding his doctoral studies. LA would like to thank the University of Bristol's Advanced Computing Research Centre (ACRC) for computational resources.

## References

Aitchison, L. Why bigger is not always better: on finite and infinite neural networks. *arXiv preprint arXiv:1910.08013*, 2019.

Aitchison, L. A statistical theory of cold posteriors in deep neural networks. *arXiv preprint arXiv:2008.05912*, 2020.

Aitchison, L., Yang, A. X., and Ober, S. W. Deep kernel processes. *arXiv preprint arXiv:2010.01590*, 2020.

Ashukha, A., Lyzhov, A., Molchanov, D., and Vetrov, D. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. *arXiv preprint arXiv:2002.06470*, 2020.

Bae, J., Zhang, G., and Grosse, R. Eigenvalue corrected noisy natural gradient. *arXiv preprint arXiv:1811.12565*, 2018.

Bartlett, M. S. On the theory of statistical regression. *Proceedings of the Royal Society of Edinburgh*, 53:260–283, 1933.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Burt, D. R., Rasmussen, C. E., and van der Wilk, M. Convergence of sparse variational inference in Gaussian processes regression. *Journal of Machine Learning Research*, 21:1–63, 2020.

Chafaï, D. Bartlett decomposition and other factorizations, 2015. URL http://djalil.chafai.net/blog/2015/10/20/bartlett-decomposition-and-other-factorizations/.

Damianou, A. and Lawrence, N. Deep Gaussian processes. In *Artificial Intelligence and Statistics*, pp. 207–215, 2013.

Dusenberry, M. W., Jerfel, G., Wen, Y., Ma, Y.-a., Snoek, J., Heller, K., Lakshminarayanan, B., and Tran, D. Efficient and scalable Bayesian neural nets with rank-1 factors. *arXiv preprint arXiv:2005.07186*, 2020.

Dutordoir, V., van der Wilk, M., Artemev, A., Tomczak, M., and Hensman, J. Translation insensitivity for deep convolutional Gaussian processes. *arXiv preprint arXiv:1902.05888*, 2019.

Farquhar, S., Smith, L., and Gal, Y. Try depth instead of weight correlations: Mean-field is a less restrictive assumption for deeper networks. *arXiv preprint arXiv:2002.03704*, 2020.

Foong, A. Y., Burt, D. R., Li, Y., and Turner, R. E. Pathologies of factorised Gaussian and MC dropout posteriors in Bayesian neural networks. *arXiv preprint arXiv:1909.00719*, 2019a.

Foong, A. Y., Li, Y., Hernández-Lobato, J. M., and Turner, R. E. 'in-between' uncertainty in Bayesian neural networks. *arXiv preprint arXiv:1906.11537*, 2019b.

Fortuin, V., Garriga-Alonso, A., Wenzel, F., Rätsch, G., Turner, R., van der Wilk, M., and Aitchison, L. Bayesian neural network priors revisited. *arXiv preprint arXiv:2102.06571*, 2021.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *arXiv preprint arXiv:1506.02142*, 2015.

Ghosh, S. and Doshi-Velez, F. Model selection in Bayesian neural networks via horseshoe priors. *arXiv preprint arXiv:1705.10388*, 2017.

Graves, A. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pp. 2348–2356, 2011.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1321–1330. JMLR. org, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Heinonen, M., Mannerström, H., Rousu, J., Kaski, S., and Lähdesmäki, H. Non-stationary Gaussian process regression with Hamiltonian Monte Carlo. In *Artificial Intelligence and Statistics*, pp. 732–740. PMLR, 2016.

Hernández-Lobato, J. M. and Adams, R. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pp. 1861–1869, 2015.

Hinton, G. E. and Van Camp, D. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pp. 5–13, 1993.

Hoffman, M. D. and Blei, D. M. Structured stochastic variational inference. In *Artificial Intelligence and Statistics*, pp. 361–369, 2015.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456. PMLR, 2015.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Karaletsos, T. and Bui, T. D. Hierarchical Gaussian process priors for Bayesian neural network weights. *arXiv preprint arXiv:2002.04033*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, pp. 2575–2583, 2015.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Krueger, D., Huang, C.-W., Islam, R., Turner, R., Lacoste, A., and Courville, A. Bayesian hypernetworks. *arXiv preprint arXiv:1710.04759*, 2017.

Li, Z., Wang, R., Yu, D., Du, S. S., Hu, W., Salakhutdinov, R., and Arora, S. Enhanced convolutional neural tangent kernels. *arXiv preprint arXiv:1911.00809*, 2019.

Lindinger, J., Reeb, D., Lippert, C., and Rakitsch, B. Beyond the mean-field: Structured deep Gaussian processes improve the predictive uncertainties. *arXiv preprint arXiv:2005.11110*, 2020.

Louizos, C. and Welling, M. Structured and efficient variational deep learning with matrix Gaussian posteriors. In *International Conference on Machine Learning*, pp. 1708–1716, 2016.

Louizos, C. and Welling, M. Multiplicative normalizing flows for variational Bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2218–2227. JMLR. org, 2017.

Nabarro, S., Ganev, S., Garriga-Alonso, A., Fortuin, V., van der Wilk, M., and Aitchison, L. Data augmentation in Bayesian neural networks and the cold posterior effect. *arXiv preprint*, 2021.

Naeini, M. P., Cooper, G., and Hauskrecht, M. Obtaining well calibrated probabilities using Bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Neal, R. M. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pp. 29–53. Springer, 1996.

Neal, R. M. et al. MCMC using Hamiltonian dynamics. *Handbook of Markov chain Monte Carlo*, 2(11):2, 2011.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

Ober, S. W., Rasmussen, C. E., and van der Wilk, M. The promises and pitfalls of deep kernel learning. *arXiv preprint arXiv:2102.12108*, 2021.

Osawa, K., Swaroop, S., Khan, M. E. E., Jain, A., Eschenhagen, R., Turner, R. E., and Yokota, R. Practical deep learning with Bayesian principles. In *Advances in Neural Information Processing Systems*, pp. 4289–4301, 2019.

Pawlowski, N., Brock, A., Lee, M. C., Rajchl, M., and Glocker, B. Implicit weight uncertainty in neural networks. *arXiv preprint arXiv:1711.01297*, 2017.

Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Ranganath, R., Tran, D., and Blei, D. Hierarchical variational models. In *International Conference on Machine Learning*, pp. 324–333. PMLR, 2016.

Rasmussen, C. E. and Williams, C. K. Gaussian Processes for Machine Learning. *ISBN-13 978-0-262-18253-9*, 2006.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Ritter, H., Botev, A., and Barber, D. A scalable Laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

Salimbeni, H. and Deisenroth, M. Doubly stochastic variational inference for deep Gaussian processes. In *Advances in Neural Information Processing Systems*, pp. 4588–4599, 2017.

Shi, J., Titsias, M. K., and Mnih, A. Sparse orthogonal variational inference for Gaussian processes. *arXiv preprint arXiv:1910.10596*, 2019.

Shridhar, K., Laumann, F., and Liwicki, M. A comprehensive guide to Bayesian convolutional neural network with variational inference. *arXiv preprint arXiv:1901.02731*, 2019.

Sobolev, A. and Vetrov, D. Importance weighted hierarchical variational inference. *arXiv preprint arXiv:1905.03290*, 2019.

Tieleman, T. and Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4 (2):26–31, 2012.

Trippe, B. and Turner, R. Overpruning in variational Bayesian neural networks. *arXiv preprint arXiv:1801.06230*, 2018.

Ustyuzhaninov, I., Kazlauskaite, I., Kaiser, M., Bodin, E., Campbell, N. D., and Ek, C. H. Compositional uncertainty in deep Gaussian processes. *UAI*, 2020.

Wenzel, F., Roth, K., Veeling, B. S., Świątkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. How good is the Bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.

Yao, J., Pan, W., Ghosh, S., and Doshi-Velez, F. Quality of uncertainty quantification for Bayesian neural network inference. *arXiv preprint arXiv:1906.09686*, 2019.

Zhang, G., Sun, S., Duvenaud, D., and Grosse, R. Noisy natural gradient as variational inference. *arXiv preprint arXiv:1712.02390*, 2017.