

# Approximate Inference Turns Deep Networks into Gaussian Processes

**Mohammad Emtiyaz Khan**  
RIKEN Center for AI Project  
Tokyo, Japan  
emtiyaz.khan@riken.jp

**Alexander Immer**<sup>\*†</sup>  
EPFL  
Lausanne, Switzerland  
alexander.immer@epfl.ch

**Ehsan Abedi**<sup>\*†</sup>  
EPFL  
Lausanne, Switzerland  
ehsan.abedi@epfl.ch

**Maciej Korzepa**<sup>\*</sup>  
Technical University of Denmark  
Kgs. Lyngby, Denmark  
mjko@dtu.dk

## Abstract

Deep neural networks (DNN) and Gaussian processes (GP) are two powerful models with several theoretical connections relating them, but the relationship between their training methods is not well understood. In this paper, we show that certain Gaussian posterior approximations for Bayesian DNNs are equivalent to GP posteriors. As a result, we can obtain a GP kernel and a nonlinear feature map simply by training the DNN. Surprisingly, the resulting kernel is the neural tangent kernel which has desirable theoretical properties for infinitely-wide DNNs. We show feature maps obtained on real datasets and demonstrate the use of the GP marginal likelihood to tune hyperparameters of DNNs. Our work aims to facilitate further research on combining DNNs and GPs in practical settings.

## 1 Introduction

Deep neural networks (DNN) and Gaussian processes (GP) are both powerful models with complementary strengths and weaknesses. DNNs achieve state-of-the-art results on many real-world problems providing scalable end-to-end learning, but they can overfit on small datasets and be overconfident. In contrast, GPs are suitable for small datasets and compute confidence estimates, but they are not scalable and choosing a good kernel in practice is challenging [2]. Combining their strengths to solve real-world problems is an important problem.

Theoretically, the two models are very closely related to each other. Previous work has shown that as the width of a DNN increases to infinity, the DNN converges to a GP [15, 19, 3, 5, 12]. This relationship is surprising and gives us hope that a practical combination could be possible. Unfortunately, it is not clear how one can use such connections in practice, for example, to perform fast inference in GPs by using training methods of DNNs, or to reduce overfitting in DNNs by using GP inference. We argue that, to solve such practical problems, we need the relationship not only between the models but also between their training procedures. The purpose of this paper is to provide such a theoretical relationship.

We present theoretical results aimed at connecting training methods of deep learning to GP inference. We show that the Gaussian posterior approximations for Bayesian DNNs, such as those obtained by Laplace approximation and variational inference, are posterior distributions of GP regression models. As a result, a GP kernel and a nonlinear feature map are obtained when DNNs are trained using

<sup>†</sup>Equal contribution. <sup>\*</sup>This work is conducted during an internship at the RIKEN Center for AI project.

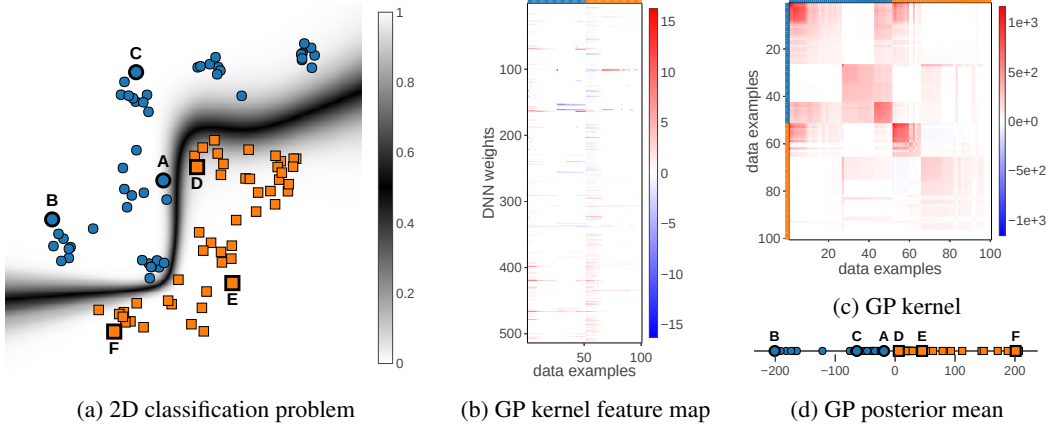


Figure 1: This figure illustrates how variational inference (VI) turns a DNN into a GP. Figure (a) shows 100 two-dimensional inputs  $\mathbf{x}_i$  with two classes, along with the predictions of DNN using 513 parameters. We use a variational Gaussian approximation. Our theoretical results show that the approximation is equivalent to the posterior of a GP regression model with kernel  $k(\mathbf{x}_i, \mathbf{x}_j) := \mathbf{J}_*(\mathbf{x}_i)\mathbf{J}_*(\mathbf{x}_j)^\top$  where  $\mathbf{J}_*(\mathbf{x}) \in \mathbb{R}^{513}$  is the Jacobian of the DNN (see Theorem 2). Figure (b) shows the feature matrix whose columns are the Jacobian  $\mathbf{J}_*(\mathbf{x}_i)$ , and Figure (c) shows the GP kernel (classes are grouped and marked with different colors along the axes). In Figure (d), we show the posterior mean of the GP where we see a clear separation between the two classes. Surprisingly, the border points A and D in (a) are also at the boundary in (d).

approximate inference methods. See Fig. 1 for an illustrative example. Surprisingly, the GP kernel we derive is equivalent to the recently proposed neural tangent kernel (NTK) [8] which has desirable theoretical properties for infinitely-wide DNNs. We present empirical results where we visualize the feature-map obtained on benchmark datasets such as MNIST and CIFAR, and demonstrate their use for tuning the hyperparameters of DNNs. The work presented in this paper aims to facilitate further research on combining the strengths of DNNs and GPs in practical settings.

## 1.1 Related Work

The equivalence between infinitely-wide neural networks and GPs was originally discussed by Neal [15]. Subsequently, many works derived explicit expressions for the GP kernel corresponding to neural networks [3, 7, 15] and their deep variants [5, 6, 12, 17]. These works use a prior distribution on weights and derive kernels by averaging over the prior. Our work differs from these works in the fact that we use the *posterior* approximations to relate DNNs to GPs. Unlike these previous results, our results hold for DNNs of finite width.

A GP kernel we derive is equivalent to the recently proposed neural Tangent Kernel (NTK) [8], which is obtained by using the Jacobian of the DNN outputs. For randomly initialized trajectories, as the DNN width goes to infinity, the NTK converges in probability to a deterministic kernel and remains asymptotically constant during training. Jacot et al. [8] motivate the NTK by using kernel gradient descent, and surprisingly the NTK appears in our work with an entirely different approach. Due to connections to the NTK, we expect similar properties for our kernel. Our approach additionally shows that by using different approximate inference methods we can obtain other types of kernels.

In a recent work, Lee et al. [13] derive the mean and covariance function corresponding to the GP induced by the NTK. Unfortunately, the model does not really correspond to inference in a GP model (see Section 2.3.1 in their paper). Our approach does not have this issue and we can express Gaussian approximations on DNN as inference in a GP regression model.

## 2 Deep Neural Networks (DNNs) and Gaussian Processes (GPs)

The goal of this paper is to present a theoretical relationship between training methods of DNNs and GPs. DNNs are typically trained by minimizing an empirical loss between the data and the

predictions. For example, in supervised learning with a dataset  $\mathcal{D} := \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$  of  $N$  examples of input  $\mathbf{x}_i \in \mathbb{R}^D$  and output  $\mathbf{y}_i \in \mathbb{R}^K$ , we can minimize a loss of the following form:

$$\bar{\ell}(\mathcal{D}, \mathbf{w}) := \sum_{i=1}^N \ell_i(\mathbf{w}) + \frac{1}{2} \delta \mathbf{w}^\top \mathbf{w}, \quad \text{where } \ell_i(\mathbf{w}) := \ell(\mathbf{y}_i, \mathbf{f}_w(\mathbf{x}_i)), \quad (1)$$

where  $\mathbf{f}_w(\mathbf{x}) \in \mathbb{R}^K$  denotes the DNN outputs with weights  $\mathbf{w} \in \mathbb{R}^P$ ,  $\ell(\mathbf{y}, \hat{\mathbf{y}})$  denotes a loss function between an output  $\mathbf{y}$  and its prediction  $\hat{\mathbf{y}}$  which is twice differentiable and strictly convex in  $\hat{\mathbf{y}}$  (e.g., squared loss and cross-entropy loss), and  $\delta$  is a small  $L_2$  regularizer.<sup>2</sup> An attractive feature of DNNs is that they can be trained using stochastic-gradient (SG) methods, e.g., stochastic-gradient descent (SGD), RMSprop, and Adam [10]. These methods can train large networks on a large amount of data.

GP models use an entirely different modeling approach which is based on directly modeling the functions rather than the parameters. For example, for regression problems with scalar outputs  $y_i \in \mathbb{R}$ , consider the following linear *basis-function* model with a nonlinear feature-map  $\phi(\mathbf{x}) : \mathbb{R}^D \mapsto \mathbb{R}^P$ :

$$y_i = \phi(\mathbf{x}_i)^\top \mathbf{w} + \epsilon_i, \quad \text{with } \epsilon_i \sim \mathcal{N}(0, \sigma^2), \quad \text{and } \mathbf{w} \sim \mathcal{N}(0, \mathbf{I}_P), \quad (2)$$

where  $\mathbf{I}_P$  is a  $P \times P$  identity matrix and  $\sigma^2$  is the output noise variance. The predictive distribution of this model is equivalent to that of the following model directly defined with a GP prior on the function  $f(\mathbf{x}) := \phi(\mathbf{x})^\top \mathbf{w}$  [20]:

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \quad \text{with } f(\mathbf{x}) \sim \mathcal{GP}(0, \kappa(\mathbf{x}, \mathbf{x}')), \quad (3)$$

where  $\kappa(\mathbf{x}, \mathbf{x}') := \mathbb{E}[f(\mathbf{x})f(\mathbf{x}')] = \phi(\mathbf{x})^\top \phi(\mathbf{x}')$  is the *covariance function* or *kernel* of the GP. The function-space model is more general in the sense that it can also deal with infinite-dimensional vector feature maps  $\phi(\mathbf{x})$ , giving us a *nonparametric* model. This view has been used to show that as a DNN becomes infinitely wide it tends to a GP, by essentially showing that averaging over  $p(\mathbf{w})$  with the feature map induced by a DNN leads to a GP covariance function.

An attractive property of the *function-space* formulation as opposed to the *weight-space* formulation, such as (1), is that the posterior distribution has a closed-form expression. Therefore, unlike DNNs, once the feature map  $\phi(\mathbf{x})$  is fixed then the posterior distribution over  $f(\mathbf{x}_i)$  is simply a multivariate Gaussian. Another attractive property is that the posterior is usually unimodal, in contrast to the loss  $\bar{\ell}$  which is typically nonconvex. Unfortunately, this takes  $O(N^3)$  which is infeasible for large datasets, and also require choosing a good kernel. Unlike DNNs, inference in GPs is more difficult in general.

To summarize, the training methods for DNNs and GPs seem to be fundamentally different from each other. DNNs employ stochastic optimization, while GPs use closed-form updates. How can these two approaches be related? In this paper, we provide an answer to this question. Our key idea is to use approximate inference on Bayesian DNNs. Approximations such as the Laplace approximation can be obtained by using the solutions found by standard deep learning methods. By showing an equivalence between the DNN posterior approximation and a GP posterior, we can relate solutions and iterations of inference on DNNs to inference in GP. We start with the Laplace approximation in the next section.

### 3 Laplace Approximation as GP Inference

The loss (1) corresponds to a Bayesian model<sup>3</sup>  $p(\mathcal{D}, \mathbf{w}) := \prod_{i=1}^N e^{-\ell_i(\mathcal{D}, \mathbf{w})} p(\mathbf{w})$  with prior distribution  $p(\mathbf{w}) := \mathcal{N}(\mathbf{w} | 0, \delta^{-1} \mathbf{I}_P)$ . We can therefore compute a posterior distribution  $p(\mathbf{w} | \mathcal{D}) = p(\mathcal{D}, \mathbf{w}) / p(\mathcal{D})$ . This is computationally intractable due to the normalization constant  $p(\mathcal{D})$ . Approximate inference methods find an approximation to the posterior  $p(\mathbf{w} | \mathcal{D})$  by using a parametric model. For example, the Laplace approximation uses Gaussian distributions to approximate the posterior. Denoting a minimum of  $\bar{\ell}(\mathbf{w})$  by  $\mathbf{w}_*$ , the Laplace approximation is defined as follows:

$$p(\mathbf{w} | \mathcal{D}) \approx \mathcal{N}(\mathbf{w} | \boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad \text{where } \boldsymbol{\mu} = \mathbf{w}_* \text{ and } \boldsymbol{\Sigma}^{-1} = \sum_{i=1}^N \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}_*) + \delta \mathbf{I}_P, \quad (4)$$

<sup>2</sup>We can also assume that  $\delta$  is small enough that it does not affect generalization of DNN.

<sup>3</sup>We assume that the loss is such that the model has a well-defined posterior distribution.

where  $\mathbf{I}_P$  is a  $P \times P$  identity matrix. The posterior approximation is directly built using the minima found by a deep-learning optimizer.

Our goal is to relate the Laplace approximation to the posterior of a GP model. To do so, we employ a further approximation to the Hessian of the loss  $\ell_i(\mathbf{w}_*)$  by using a *Generalized-Gauss Newton (GGN) approximation* [16, 14, 1]. First, note the special structure of the gradient and Hessian:

$$\nabla_w \ell_i(\mathbf{w}) = \mathbf{J}_w(\mathbf{x}_i)^\top \mathbf{r}_i, \quad \nabla_{ww}^2 \ell_i(\mathbf{w}) = \mathbf{J}_w(\mathbf{x}_i)^\top \mathbf{\Lambda}_i \mathbf{J}_w(\mathbf{x}_i) + [\nabla_{ww}^2 \mathbf{f}_w(\mathbf{x}_i)] \mathbf{r}_i, \quad (5)$$

where  $\mathbf{J}_w(\mathbf{x}_i) := \nabla_w \mathbf{f}_w(\mathbf{x}_i)^\top$  is a  $K \times P$  Jacobian matrix,  $\mathbf{r}_i := \nabla_{\hat{\mathbf{y}}} \ell(\mathbf{y}_i, \hat{\mathbf{y}})$  is the residual vector evaluated at  $\hat{\mathbf{y}} = \mathbf{f}_w(\mathbf{x}_i)$ , and  $\mathbf{\Lambda}_i := \nabla_{\hat{\mathbf{y}}\hat{\mathbf{y}}}^2 \ell(\mathbf{y}_i, \hat{\mathbf{y}})$  is the  $K \times K$  Hessian of the loss evaluated at the same point as the residual vector. The GGN approximation ignores the second term in the Hessian:

$$\nabla_{ww}^2 \ell_i(\mathbf{w}) \approx \mathbf{J}_w(\mathbf{x}_i)^\top \mathbf{\Lambda}_i \mathbf{J}_w(\mathbf{x}_i). \quad (6)$$

This is justified when the DNN  $\mathbf{f}_w(\mathbf{x})$  is a powerful model and can fit the data very well, in which case the residuals  $\mathbf{r}_i$  are close to zero. For example, for the least-square loss  $\ell(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{2\sigma^2} \|\mathbf{y} - \hat{\mathbf{y}}\|^2$  with  $\sigma^2$  as the noise variance, the residuals are simply  $\mathbf{r}_i := \sigma^{-2}(\mathbf{f}_w(\mathbf{x}_i) - \mathbf{y}_i)$  and  $\mathbf{\Lambda}_i := \sigma^{-2} \mathbf{I}_K$ . As the fit becomes more accurate, the residuals go to zero, justifying the approximation.

Using the GGN approximation in (4), we get the following modified version of Laplace approximation which we refer to as the *Laplace-GGN approximation*:

$$p(\mathbf{w}|\mathcal{D}) \approx \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}), \text{ where } \boldsymbol{\mu} = \mathbf{w}_* \text{ and } \tilde{\boldsymbol{\Sigma}}^{-1} = \sum_{i=1}^N \mathbf{J}_*(\mathbf{x}_i)^\top \mathbf{\Lambda}_{i,*} \mathbf{J}_*(\mathbf{x}_i) + \delta \mathbf{I}_P, \quad (7)$$

where  $\mathbf{J}_*(\mathbf{x}_i)$  and  $\mathbf{\Lambda}_{i,*}$  are evaluated at  $\mathbf{w}_*$ . The following theorem states the equivalence between the Laplace-GGN approximation and the posterior distribution of a GP model.

**Theorem 1. (Laplace approximation as a GP):** *The Laplace-GGN approximation (7) is equivalent to the posterior distribution of a linear basis-function model with observations  $\tilde{\mathbf{y}}_i \in \mathbb{R}^K$ :*

$$\tilde{\mathbf{y}}_i = \mathbf{J}_*(\mathbf{x}_i) \mathbf{w} + \boldsymbol{\epsilon}_i, \text{ with } \boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \mathbf{\Lambda}_{i,*}^{-1}) \text{ and } \mathbf{w} \sim \mathcal{N}(0, \delta^{-1} \mathbf{I}_P), \quad (8)$$

where the observations are defined as  $\tilde{\mathbf{y}}_i := \mathbf{J}_*(\mathbf{x}_i) \mathbf{w}_* - \mathbf{\Lambda}_{i,*}^{-1} \mathbf{r}_{i,*}$  and the feature maps are defined as  $\mathbf{J}_*(\mathbf{x}_i)^\top$  with  $\mathbf{J}_*(\mathbf{x}_i)$ ,  $\mathbf{r}_{i,*}$ , and  $\mathbf{\Lambda}_{i,*}$  being the Jacobian, residual, and noise precision evaluated at  $\mathbf{w}_*$ . The predictive distribution of this model is equivalent to that of a GP regression model defined with a multi-dimensional  $K \times K$  kernel:

$$\tilde{\mathbf{y}}_i = \mathbf{f}(\mathbf{x}_i) + \boldsymbol{\epsilon}_i, \quad \text{with } \mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(0, \delta^{-1} \mathbf{J}_*(\mathbf{x}) \mathbf{J}_*(\mathbf{x}')^\top). \quad (9)$$

A proof is given in Appendix A.1. The above equivalence is similar to the equivalence between the weight-space view shown in (2) and the function-space view shown in (3). Our theorem converts a DNN defined in the weight-space to a GP defined in the function-space. The feature map used to do so is the Jacobian  $\mathbf{J}_*(\mathbf{x})^\top$  defined at the solution  $\mathbf{w}_*$  of the loss (1), and the precision matrix  $\mathbf{\Lambda}_i$  for the noise is the Hessian of the loss. Therefore, by using the Laplace-GGN approximation, we can turn a DNN into a GP where the two are related through the Jacobian feature map.

The GP kernel above is the Neural Tangent Kernel<sup>4</sup> (NTK) [8] which has desirable theoretical properties. As the width of the DNN is increasing to infinity, the kernel converges in probability to a deterministic kernel and also remains asymptotically constant during training. Our kernel is the NTK defined at  $\mathbf{w}_*$  and is expected to have similar properties. It is also likely that, as the DNN width is increased, the Laplace-GGN approximation behaves like a GP posterior, and can be potentially used to improve the performance of DNNs. For example, we can use GPs to tune hyperparameters of DNNs, as demonstrated in our experiments. Such applications would make approximate inference methods desirable for overparametrized DNNs.

An advantage of our approach is that we can derive kernels other than the NTK. One way to do so is to employ a Hessian approximation different from the GGN approximation. In general, analogous to the exact derivatives (5), any approximation of the following form will always result in a linear basis-function model similar to (8):  $\nabla_w \ell_i(\mathbf{w}) \approx \mathbf{U}(\mathbf{x}_i)^\top \mathbf{v}_i$  and  $\nabla_{ww}^2 \ell_i(\mathbf{w}) \approx \mathbf{U}(\mathbf{x}_i)^\top \mathbf{S}_i \mathbf{U}(\mathbf{x}_i)$ ,

<sup>4</sup>The NTK corresponds to  $\delta = 1$  which implies a standard normal prior on weights.

where  $\mathbf{U}_i$  is a  $Q \times P$  matrix,  $\mathbf{v}_i$  is a  $Q$  length vector, and  $\mathbf{S}_i$  is a  $Q \times Q$  symmetric positive-definite matrix. Other versions where Jacobians are replaced by gradients  $\mathbf{g}_i := \nabla_{\mathbf{w}} \ell_i(\mathbf{w})$  are also possible, e.g.,  $\nabla_{\mathbf{w}} \ell_i(\mathbf{w}) := \mathbf{g}_i$  and  $\nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}) \approx \mathbf{g}_i \mathbf{g}_i^\top$  in which case  $Q = 1$ . Such modifications can be useful to reduce the cost of computing the feature map and kernel, although this is at the cost of increasing the approximation error. Another way to derive other types of GP formulation is to change the posterior approximation. We will now show a similar result for variational inference.

#### 4 Variational Inference as GP Inference

In this section, we present a result similar to Theorem 1 but for variational inference (VI). In VI, we can estimate Gaussian approximation  $q(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$  to the posterior distribution  $p(\mathbf{w}|\mathcal{D})$  by minimizing the following variational objective:

$$\mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^N \mathbb{E}_q[\ell_i(\mathbf{w})] + \mathbb{D}_{KL}[q(\mathbf{w}) \| p(\mathbf{w})], \quad (10)$$

where  $\mathbb{D}_{KL}[\cdot \| \cdot]$  is the Kullback-Leibler (KL) divergence. Given the prior distribution  $p(\mathbf{w}) := \mathcal{N}(\mathbf{w}|0, \delta^{-1}\mathbf{I}_P)$ , a minimum of this objective, denoted by  $\boldsymbol{\mu}_*$  and  $\boldsymbol{\Sigma}_*$ , satisfies the following condition as shown by [18]:

$$0 = \sum_{i=1}^N \mathbb{E}_{q_*(\mathbf{w})}[\nabla_{\mathbf{w}} \ell_i(\mathbf{w})] + \delta \boldsymbol{\mu}_*, \quad \boldsymbol{\Sigma}_*^{-1} = \sum_{i=1}^N \mathbb{E}_{q_*(\mathbf{w})}[\nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w})] + \delta \mathbf{I}_P, \quad (11)$$

where  $q_*(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$  is the Gaussian approximation corresponding to the minimum. Similar to Laplace approximation, the variational approximation is also specified using the gradients and Hessian of the loss  $\ell_i(\mathbf{w})$ . We can therefore approximate the covariance by using the GGN approximation (6). We call this the *variational-GGN approximation*. Since there are no closed-form expressions for the expectations, it is a standard practice to use  $S$  number of samples  $\mathbf{w}^{(s)} \sim q_*(\mathbf{w})$  and use the Monte Carlo approximation. We refer to this approximation as the *variational-GGN-MC approximation*, denoted by  $\tilde{q}_*(\mathbf{w})$ . The following theorem shows the equivalence of the variational-GGN-MC and GP posterior distribution.

**Theorem 2. (Minima of the variational objectives as a GP) :** *The Variational-GGN-MC approximation  $\tilde{q}_*(\mathbf{w})$  as defined above is equivalent to the posterior distribution of a linear basis-function model with observations  $\tilde{\mathbf{y}}_{i,s} \in \mathbb{R}^K$ :*

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{J}_s(\mathbf{x}_i)\mathbf{w} + \boldsymbol{\epsilon}_{i,s}, \text{ with } \boldsymbol{\epsilon}_{i,s} \sim \mathcal{N}(0, S\boldsymbol{\Lambda}_{i,s}^{-1}) \text{ and } \mathbf{w} \sim \mathcal{N}(0, \delta^{-1}\mathbf{I}_P), \quad (12)$$

where the observations are defined as  $\tilde{\mathbf{y}}_{i,s} := \mathbf{J}_s(\mathbf{x}_i)\boldsymbol{\mu}_* - \boldsymbol{\Lambda}_{i,s}^{-1}\mathbf{r}_{i,s}$  and the feature maps are defined as  $\mathbf{J}_s(\mathbf{x}_i)^\top$  with  $\mathbf{J}_s(\mathbf{x}_i)$ ,  $\mathbf{r}_{i,s}$ , and  $\boldsymbol{\Lambda}_{i,s}$  denoting the Jacobian, residual, and noise precision evaluated at the sampled weight  $\mathbf{w}^{(s)}$  drawn from  $\tilde{q}_*(\mathbf{w})$ . The predictive distribution of this model is equivalent to that of the following GP regression model:

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{f}_s(\mathbf{x}_i) + \boldsymbol{\epsilon}_{i,s}, \quad \text{where } \mathbf{f}_s(\mathbf{x}) \sim \mathcal{GP}(0, \delta^{-1}\mathbf{J}_s(\mathbf{x})\mathbf{J}_s'(\mathbf{x}')^\top). \quad (13)$$

The proof appears in Appendix A.2. Similar to Theorem 1, we see that we can use VI to convert DNNs into GPs. The feature map now is the Jacobian  $\mathbf{J}_s(\mathbf{x})$  evaluated at the samples from  $q_*(\mathbf{w})$ , instead of being evaluated at the minimum  $\mathbf{w}_*$  of the loss. This result is a direct consequence of VI being a *global* approximation where the conditions of the Laplace approximation hold on *average* [18] which can also be observed by comparing (11) and (4). Samples from  $q_*(\mathbf{w})$  are used to generate additional data points and can improve the quality of the posterior approximation.

So far, we have only related minima of  $\bar{\ell}$  and  $\mathcal{L}$  to GP inference. However, we can further show that the iterations minimizing  $\mathcal{L}$  possess similar properties. To do so, we consider natural-gradient methods for VI. In particular, the Variational Online Newton (VON) updates proposed in [9] take the following form:

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \boldsymbol{\Sigma}_{t+1} \left[ \sum_{i=1}^N \mathbb{E}_{q_t(\mathbf{w})}[\nabla_{\mathbf{w}} \ell_i(\mathbf{w})] + \delta \boldsymbol{\mu}_t \right], \quad (14)$$

$$\boldsymbol{\Sigma}_{t+1}^{-1} = (1 - \beta_t)\boldsymbol{\Sigma}_t^{-1} + \beta_t \left[ \sum_{i=1}^N \mathbb{E}_{q_t(\mathbf{w})}[\nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w})] + \delta \mathbf{I}_P \right], \quad (15)$$

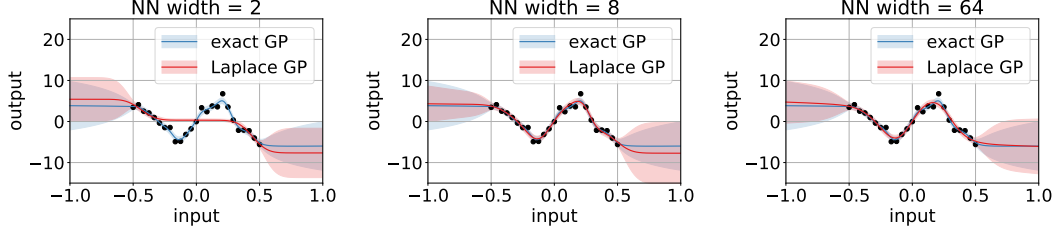


Figure 2: For a NN with sigmoidal transfer function [19], we compare the predictive distribution of the exact GP corresponding to the infinitely-wide NN to the GP obtained using Laplace-GGN on a finite-width NN. For larger widths, the two distributions match more.

where  $t$  denotes the iterations and  $\beta_t$  is the learning rate. Clearly, once converged, the updates satisfy the stationarity condition (11). For our purpose, this is useful because the updates are expressed using the gradient and Hessian of  $\ell_i(\mathbf{w})$ . Therefore, we can apply approximations to establish equivalence to GP inference. We refer to the update (14) and (15) with the GGN approximation (and a MC approximation) as the *VOGGN update*. Equivalence to GP inference is discussed below.

**Theorem 3. (Iterations of a VI algorithm as a GP) :** *At each iteration  $t$  of the VOGGN update, i.e. (14) and (15) but with the GGN approximation to the Hessian, the posterior approximation  $q_t(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$  is equivalent to the posterior distribution of the following linear basis-function model:*

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{J}_s(\mathbf{x}_i)\mathbf{w} + \epsilon_{i,s}, \text{ with } \epsilon_{i,s} \sim \mathcal{N}(0, S(\beta_t \boldsymbol{\Lambda}_{i,s})^{-1}) \text{ and } \mathbf{w} \sim \mathcal{N}(\mathbf{m}_t, \mathbf{S}_t) \quad (16)$$

where  $\mathbf{S}_t^{-1} := (1 - \beta_t)\boldsymbol{\Sigma}_t^{-1} + \beta_t\delta\mathbf{I}_P$  and  $\mathbf{m}_t := (1 - \beta_t)\mathbf{S}_t\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t$ . The observations are defined as  $\tilde{\mathbf{y}}_{i,s} := \mathbf{J}_s(\mathbf{x}_i)\boldsymbol{\mu}_t - \boldsymbol{\Lambda}_{i,s}^{-1}\mathbf{r}_{i,s}$  and the feature maps are defined as  $\mathbf{J}_s(\mathbf{x}_i)^\top$  with  $\mathbf{J}_s(\mathbf{x}_i)$ ,  $\mathbf{r}_{i,s}$ , and  $\boldsymbol{\Lambda}_{i,s}$  denoting the Jacobian, residual, and noise precision evaluated at the sampled weight  $\mathbf{w}^{(s)}$  drawn from  $q_t(\mathbf{w})$ . The predictive distribution of this model is equivalent to that of the following GP regression model:

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{f}_s(\mathbf{x}_i) + \epsilon_{i,s}, \quad \text{where } \mathbf{f}_s(\mathbf{x}) \sim \mathcal{GP}(\mathbf{J}_s(\mathbf{x})\mathbf{m}_t, \mathbf{J}_s(\mathbf{x})\mathbf{S}_t\mathbf{J}_s'(\mathbf{x})^\top). \quad (17)$$

A proof is given in Appendix A.3. The above result differs from Theorem 2 in the fact that the prior over  $\mathbf{w}$  is now a correlated multivariate normal distribution as it is obtained using the Gaussian approximation at iteration  $t$ . Therefore, the mean and the covariance function of the GP are modified.

To conclude, we show that iterations and solutions of a VI algorithm on Bayesian DNNs correspond to inference in a GP model. The GP update can be seen as a mirror-descent step in the function space [4], but the kernel in our update changes at every time step. This is similar to the functional gradient-descent discussed in [8] but for infinitely-wide DNNs the kernel remains the same throughout training. It is possible that this is also the case for our update and such theoretical properties need to be studied in the future.

## 5 Experiments and Results

### 5.1 Comparison with the Exact GP corresponding for an Infinitely-Wide DNN

Several works have shown that as the width of a DNN is increased, it converges to a GP. We compare the predictions of our GP model to those of the exact GP corresponding to an infinitely-wide DNN. We consider a neural network with a sigmoidal transfer function. An exact expression for the GP is available when the width goes to infinity [19]. Using this, we compute predictive distribution of the exact GP. We compare it to the posterior predictive distribution of the GP obtained due to Theorem 1 (see details for the computation of the predictive distribution in Appendix B). We increase the width of the neural network from 2 to 64. Figure 2 shows the results on a synthetic dataset, where we see that the two predictive distributions match well for large width.

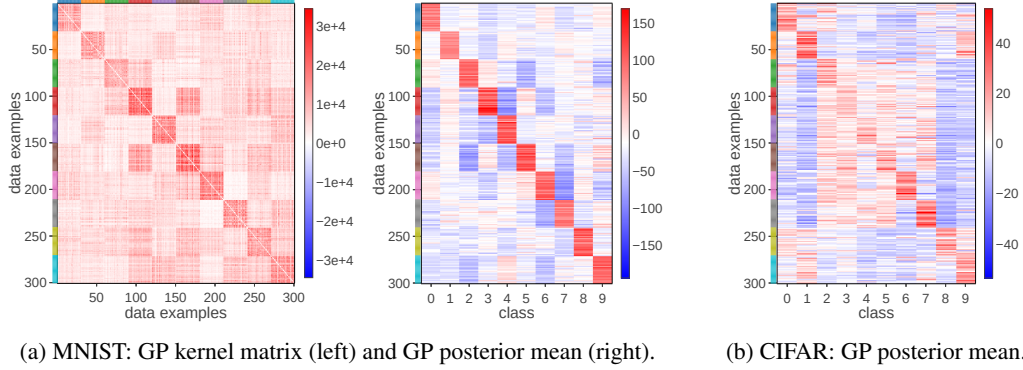


Figure 3: GP kernel and posterior mean corresponding to the Laplace approximation for LeNet5 trained on MNIST and CIFAR-10. The kernel matrix shows the correlations learned by the DNN (classes are grouped and marked with different colors along the axes). For MNIST, a higher posterior mean is assigned to the correct label most of the time (see rows in the middle figure), which reflects the good accuracy obtained by the DNN (99%). For CIFAR, the accuracy is only 68%, as a result, the pattern is a bit unclear reflecting the uncertainty in the predictions.

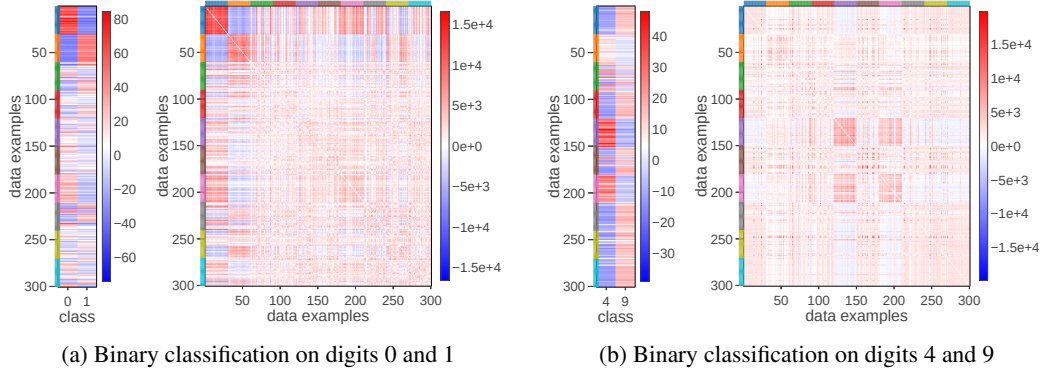


Figure 4: GP kernel and posterior mean corresponding to VI on 2 out of 10 MNIST classes. We clearly see that the *in-class* digits are assigned higher posterior mean and the correlation learned is also significant. There are plenty of other correlations learned by the network due to which we get many overconfident predictions for out-of-training classes, especially on the harder 4 vs. 9 task.

## 5.2 GP Kernel and Feature Map for Classification Datasets

In this section, we show the GP kernel and feature maps for DNNs trained on CIFAR-10 and MNIST. Our goal is to show that our GP kernel and its predictions enhance our understanding of DNN’s performance on classification tasks. We consider LeNet-5 [11] on both MNIST and CIFAR-10 and compute Gaussian approximations by using the Laplace approximation and VI methods discussed in Section 3 and 4. For Laplace, we obtain local minima using Adam [10]. For VI, we use a diagonal version of VOGGN called VOGN [9] and only compute Jacobians at the optimum. For both, we show the visualization at the posterior mean in Eq. (4) and (11).

We visualize the  $K \times K$  GP kernel  $\kappa_*(\mathbf{x}, \mathbf{x}') := \mathbf{J}_*(\mathbf{x})\mathbf{J}_*(\mathbf{x}')^\top$  for the *Laplace approximation*. The resulting kernel matrix is  $NK \times NK$  which makes it difficult to visualize for our datasets. To simplify, we sum the diagonal entries of  $\kappa_*(\mathbf{x}, \mathbf{x}')$  up and form an  $N \times N$  matrix which we refer to as the GP kernel matrix. We also visualize the GP posterior mean defined as follows:  $\mathbb{E}[f(\mathbf{x})|\mathcal{D}] = \mathbf{J}_*(\mathbf{x})\mathbf{w}_* \in \mathbb{R}^K$ . If the true label is  $k$ , then we want the  $k$ ’th entry of the posterior mean to be the highest. This is because we expect the GP function  $\mathbf{f}(\mathbf{x})$  to be related to the DNN function  $\mathbf{f}_w(\mathbf{x})$  at training data points.

Figure 3a shows the GP kernel matrix and the posterior mean for the Laplace approximation on 300 randomly sampled examples from MNIST. The rows and columns containing data examples are grouped according to the classes. The kernel matrix clearly shows the correlations learned by the DNN. As expected, each row in the posterior mean also reflects that the classes are correctly



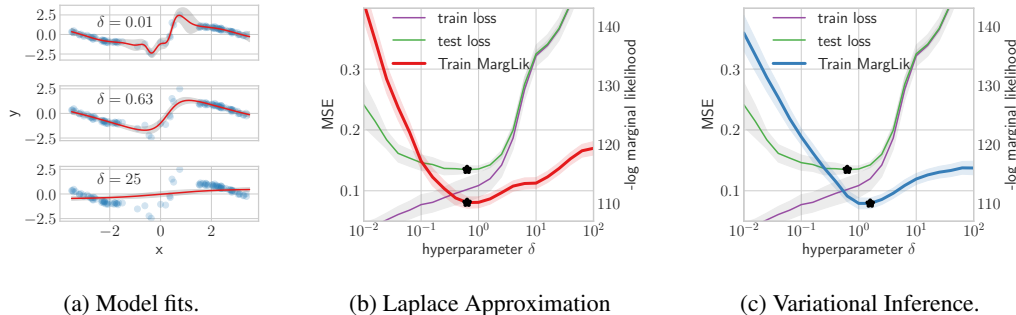


Figure 5: This figure demonstrates the use of the GP marginal likelihood to tune hyperparameters of a DNN. We tune the regularization parameter  $\delta$  on a synthetic dataset shown in (a). Fig. (b) and (c) show train and test MSE along with log of the marginal likelihoods on training data obtained with Laplace and VI respectively. We show the standard error over 10 runs. We clearly see that best hyperparameters (shown with black star) found by test loss and marginal-likelihood match well.

classified (DNN test accuracy is 99%). Figure 3b shows the GP posterior mean for CIFAR-10 where we see the same pattern but a bit unclear, which is due to a lower accuracy of around 68% on this task. Due to space constraints, the corresponding results for VI are shown in Appendix C.

In Fig. 4, we study the kernel for classes *outside* of the training data set using VI. We train LeNet-5 using VOGN on two binary-classification problems on MNIST. In both figures, the data examples are sorted according to the digits (0 is on top/left and 9 is at bottom/right).

Within the MNIST data set, the pair 4 and 9 is one of the hardest to distinguish while pair 0 and 1 forms a simple task. Fig. 4 shows that the kernel obtained for the simpler task leads to much less correlations with unseen classes and the posterior mean does not produce confident predictions on other classes. However, training on the harder task yields a potentially more complex feature map that leads to overconfident out-of-class predictions and high correlations. These observations are in line with confusion metrics typically obtained on the MNIST data set.<sup>5</sup>

### 5.3 Tuning the Hyperparameters of a DNN using the GP Marginal Likelihood

In this section, we demonstrate the tuning of a DNN hyperparameter by using the GP marginal likelihood on a synthetic dataset. In the DNN literature, this is usually done by analyzing loss on the validation set. Our goal is to demonstrate that, using our GP formulation, we can do so by simply computing marginal likelihoods on the *training* set alone.

We generate a synthetic regression dataset ( $N = 100$ ) with 1 dimensional inputs and a single output (see Fig. 5a) where there are a few data points around  $x = 0$  but plenty away from it. We fit the data by using a neural network with single hidden layer of 20 units and tanh-nonlinearity. Our goal is to tune the regularization parameter  $\delta$  to trade-off underfitting vs overfitting.

We compute the train and test mean-square error (MSE) obtained by using only the maximum-a-posterior (MAP) estimate, and compare it to the log marginal-likelihood obtained by our GP formulation. Fig. 5b and 5c show these for Laplace and VI respectively. Black stars show the hyperparameters chosen by using test loss and log marginal likelihood, respectively. We clearly see that the train marginal-likelihood chooses hyperparameters that generalize well. The train MSE on the other hand overfits as  $\delta$  is reduced.

## 6 Discussion and Future Work

In this paper, we present theoretical results connecting approximate inference on DNNs to GP posteriors. Our work enables the extraction of feature maps and GP kernels by simply training DNNs. It provides a natural way to combine the two different models.

<sup>5</sup>Exemplary MNIST confusion matrix: [https://ml4a.github.io/demos/confusion\\_mnist/](https://ml4a.github.io/demos/confusion_mnist/)



Our hope is that our theoretical results will facilitate further research on combining strengths of DNNs and GPs. A computational bottleneck is the Jacobian computation which prohibits application to large problems. There are several ways to reduce this computation, e.g., by choosing a different type of GGN approximation that uses gradients instead of the Jacobians. Exploration of such methods is a future direction that needs to be pursued.

Exact inference on the GP model we derive is still computationally infeasible for large problems. However, further approximations could enable inference on bigger datasets. Our theoretical connections do indicate this to certain degree. It is likely that, as the DNN width is increased, approximate inference behaves like the posterior of a GP which is an interesting future direction to pursue.

Finally, our work opens many other interesting avenues where a combination of GPs and DNNs can be useful. For example, hyperparameter tuning, Bandits, deep reinforcement learning, Bayesian optimization, active learning, interpretation, etc. We hope that our work enables the community to conduct further research on such problems.

### Acknowledgements

We would like to thank Kazuki Oosawa (Tokyo Institute of Technology), Anirudh Jain (RIKEN), and Runa Eschenhagen (RIKEN) for their help with the experiments. We are also thankful for the RAIDEN computing system and its support team at the RIKEN Center for Advanced Intelligence Project which we used extensively for our experiments.

### References

- [1] L. Bottou, F. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018. doi: 10.1137/16M1080173.
- [2] John Bradshaw, Alexander G de G Matthews, and Zoubin Ghahramani. Adversarial examples, uncertainty, and transfer testing robustness in Gaussian process hybrid deep networks. *arXiv preprint arXiv:1707.02476*, 2017.
- [3] Youngmin Cho and Lawrence K. Saul. Kernel methods for deep learning. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 342–350. Curran Associates, Inc., 2009.
- [4] Bo Dai, Niao He, Hanjun Dai, and Le Song. Provable bayesian inference via particle mirror descent. In *Artificial Intelligence and Statistics*, pages 985–994, 2016.
- [5] 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.
- [6] Adrià Garriga-Alonso, Carl Edward Rasmussen, and Laurence Aitchison. Deep convolutional networks as shallow Gaussian processes. In *International Conference on Learning Representations*, 2019.
- [7] Tamir Hazan and Tommi S. Jaakkola. Steps toward deep kernel methods from infinite neural networks. *CoRR*, abs/1508.05133, 2015.
- [8] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8571–8580. Curran Associates, Inc., 2018.
- [9] Mohammad Emtiyaz Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable Bayesian deep learning by weight-perturbation in Adam. In *International Conference on Machine Learning*, pages 2616–2625, 2018.
- [10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [11] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

- [12] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as Gaussian processes. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.
- [13] Jaehoon Lee, Lechao Xiao, Samuel S. Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. *arXiv e-prints*, art. arXiv:1902.06720, Feb 2019.
- [14] James Martens. New perspectives on the natural gradient method. *CoRR*, abs/1412.1193, 2014.
- [15] Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, Berlin, Heidelberg, 1996. ISBN 0387947248.
- [16] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.
- [17] Roman Novak, Lechao Xiao, Yasaman Bahri, Jaehoon Lee, Greg Yang, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-dickstein. Bayesian deep convolutional networks with many channels are Gaussian processes. In *International Conference on Learning Representations*, 2019.
- [18] Manfred Opper and Cédric Archambeau. The variational Gaussian approximation revisited. *Neural computation*, 21(3):786–792, 2009.
- [19] Christopher KI Williams. Computing with infinite networks. In *Advances in neural information processing systems*, pages 295–301, 1997.
- [20] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT Press Cambridge, MA, 2006.

## A Proofs

This appendix is dedicated to proving the theorems in the main text.

### A.1 Proof of Theorem 1

**Theorem 1. (Laplace approximation as a GP):** *The Laplace-GGN approximation (7) is equivalent to the posterior distribution of a linear basis-function model with observations  $\tilde{\mathbf{y}}_i \in \mathbb{R}^K$ :*

$$\tilde{\mathbf{y}}_i = \mathbf{J}_*(\mathbf{x}_i)\mathbf{w} + \boldsymbol{\epsilon}_i, \quad \text{with } \boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \boldsymbol{\Lambda}_{i,*}^{-1}) \text{ and } \mathbf{w} \sim \mathcal{N}(0, \delta^{-1}\mathbf{I}_P), \quad (8)$$

where the observations are defined as  $\tilde{\mathbf{y}}_i := \mathbf{J}_*(\mathbf{x}_i)\mathbf{w}_* - \boldsymbol{\Lambda}_{i,*}^{-1}\mathbf{r}_{i,*}$  and the feature maps are defined as  $\mathbf{J}_*(\mathbf{x}_i)^\top$  with  $\mathbf{J}_*(\mathbf{x}_i)$ ,  $\mathbf{r}_{i,*}$ , and  $\boldsymbol{\Lambda}_{i,*}$  being the Jacobian, residual, and noise precision evaluated at  $\mathbf{w}_*$ . The predictive distribution of this model is equivalent to that of a GP regression model defined with a multi-dimensional  $K \times K$  kernel:

$$\tilde{\mathbf{y}}_i = \mathbf{f}(\mathbf{x}_i) + \boldsymbol{\epsilon}_i, \quad \text{with } \mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(0, \delta^{-1}\mathbf{J}_*(\mathbf{x})\mathbf{J}_*(\mathbf{x}')^\top). \quad (9)$$

*Proof.* We first find the natural-parameters of the Laplace approximation (4) which we denote by  $\boldsymbol{\eta} := \{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, -\frac{1}{2}\boldsymbol{\Sigma}^{-1}\}$ . For the second natural parameter, we can write the following:

$$-\frac{1}{2}\boldsymbol{\Sigma}^{-1} = -\frac{1}{2} \left[ \sum_{i=1}^N \nabla_{ww}^2 \ell_i(\mathbf{w}_*) + \delta \mathbf{I}_P \right]. \quad (18)$$

Similarly, for the first natural parameter, we can use the first-order stationary condition, that is,  $\nabla_w \bar{\ell}(\mathbf{w}_*) = 0$ , to get the following:

$$\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} = -\nabla_w \bar{\ell}(\mathbf{w}_*) + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \quad (19)$$

$$= -\sum_{i=1}^N \nabla_w \ell_i(\mathbf{w}_*) - \delta \mathbf{w}_* + \left[ \sum_{i=1}^N \nabla_{ww}^2 \ell_i(\mathbf{w}_*) + \delta \mathbf{I}_P \right] \mathbf{w}_* \quad (20)$$

$$= \sum_{i=1}^N \left[ -\nabla_w \ell_i(\mathbf{w}_*) + \nabla_{ww}^2 \ell_i(\mathbf{w}_*) \mathbf{w}_* \right]. \quad (21)$$

Using (18) and (21), we obtain the Laplace approximation, denoted by  $q_L(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ ,

$$q_L \propto \exp \left[ -\frac{1}{2} \mathbf{w}^\top \boldsymbol{\Sigma}^{-1} \mathbf{w} + \mathbf{w}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \right] \quad (22)$$

$$= \exp \left( \frac{-\delta \mathbf{w}^\top \mathbf{w}}{2} \right) \prod_{i=1}^N \exp \left[ -\frac{1}{2} \mathbf{w}^\top \nabla_{ww}^2 \ell_i(\mathbf{w}_*) \mathbf{w} + \mathbf{w}^\top \left\{ -\nabla_w \ell_i(\mathbf{w}_*) + \nabla_{ww}^2 \ell_i(\mathbf{w}_*) \mathbf{w}_* \right\} \right]. \quad (23)$$

The above result is exact for Laplace approximation corresponding to the loss function (1). Now we employ the GGN approximation (6) and find that the Laplace-GGN approximation, denoted by  $\tilde{q}_L(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}})$  with parameters (7), is proportional to

$$\exp \left( \frac{-\delta \mathbf{w}^\top \mathbf{w}}{2} \right) \prod_{i=1}^N \exp \left[ \frac{-1}{2} \mathbf{w}^\top \mathbf{J}_*(\mathbf{x}_i)^\top \boldsymbol{\Lambda}_{i,*} \mathbf{J}_*(\mathbf{x}_i) \mathbf{w} + \mathbf{w}^\top \mathbf{J}_*(\mathbf{x}_i)^\top \left\{ \boldsymbol{\Lambda}_{i,*} \mathbf{J}_*(\mathbf{x}_i) \mathbf{w}_* - \mathbf{r}_{i,*} \right\} \right]. \quad (24)$$

The crucial point here is that each term in the product (24) can be written as a Gaussian distribution, provided that  $\boldsymbol{\Lambda}_{i,*} \succ 0$ . Defining  $\tilde{\mathbf{y}}_i := \mathbf{J}_*(\mathbf{x}_i)\mathbf{w}_* - \boldsymbol{\Lambda}_{i,*}^{-1}\mathbf{r}_{i,*}$  leads to

$$\tilde{q}_L(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}) \propto \mathcal{N}(\mathbf{w}|0, \delta^{-1}\mathbf{I}_P) \prod_{i=1}^N \mathcal{N}(\tilde{\mathbf{y}}_i | \mathbf{J}_*(\mathbf{x}_i)\mathbf{w}, \boldsymbol{\Lambda}_{i,*}^{-1}). \quad (25)$$

The right hand side of the above equation is proportional to the posterior distribution of a linear basis-function model  $\tilde{\mathbf{y}}_i = \mathbf{J}_*(\mathbf{x}_i)\mathbf{w} + \boldsymbol{\epsilon}_i$  with Gaussian noise  $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \boldsymbol{\Lambda}_{i,*}^{-1})$  and prior distribution  $\mathbf{w} \sim \mathcal{N}(0, \delta^{-1}\mathbf{I}_P)$ . The predictive distribution of this model can be equivalently written in the function space, giving rise to a GP regression model with kernel  $\delta^{-1}\mathbf{J}_*(\mathbf{x})\mathbf{J}_*(\mathbf{x}')^\top$ .  $\square$

## A.2 Proof of Theorem 2

**Theorem 2. (Minima of the variational objectives as a GP) :** *The Variational-GGN-MC approximation  $\tilde{q}_*(\mathbf{w})$  as defined above is equivalent to the posterior distribution of a linear basis-function model with observations  $\tilde{\mathbf{y}}_{i,s} \in \mathbb{R}^K$ :*

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{J}_s(\mathbf{x}_i)\mathbf{w} + \boldsymbol{\epsilon}_{i,s}, \text{ with } \boldsymbol{\epsilon}_{i,s} \sim \mathcal{N}(0, S\boldsymbol{\Lambda}_{i,s}^{-1}) \text{ and } \mathbf{w} \sim \mathcal{N}(0, \delta^{-1}\mathbf{I}_P), \quad (12)$$

where the observations are defined as  $\tilde{\mathbf{y}}_{i,s} := \mathbf{J}_s(\mathbf{x}_i)\boldsymbol{\mu}_* - \boldsymbol{\Lambda}_{i,s}^{-1}\mathbf{r}_{i,s}$  and the feature maps are defined as  $\mathbf{J}_s(\mathbf{x}_i)^\top$  with  $\mathbf{J}_s(\mathbf{x}_i)$ ,  $\mathbf{r}_{i,s}$ , and  $\boldsymbol{\Lambda}_{i,s}$  denoting the Jacobian, residual, and noise precision evaluated at the sampled weight  $\mathbf{w}^{(s)}$  drawn from  $\tilde{q}_*(\mathbf{w})$ . The predictive distribution of this model is equivalent to that of the following GP regression model:

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{f}_s(\mathbf{x}_i) + \boldsymbol{\epsilon}_{i,s}, \quad \text{where } \mathbf{f}_s(\mathbf{x}) \sim \mathcal{GP}(0, \delta^{-1}\mathbf{J}_s(\mathbf{x})\mathbf{J}_{s'}(\mathbf{x}')^\top). \quad (13)$$

*Proof.* We prove this theorem in the same way as we did for the previous one. Let  $\boldsymbol{\eta}_* := \{\boldsymbol{\Sigma}_*^{-1}\boldsymbol{\mu}_*, -\frac{1}{2}\boldsymbol{\Sigma}_*^{-1}\}$  denote the natural-parameters of the Variational approximation  $q_*(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$  that satisfies the condition (11). We first find  $\boldsymbol{\eta}_*$  and then obtain the distribution. For the expectations in (11), we use Monte Carlo sampling. The second natural parameter then is

$$-\frac{1}{2}\boldsymbol{\Sigma}_*^{-1} = -\frac{1}{2} \left[ \sum_{i=1}^N \mathbb{E}_{q_*(\mathbf{w})} [\nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w})] + \delta \mathbf{I}_P \right] \approx -\frac{1}{2} \left[ \frac{1}{S} \sum_{i,s=1}^{N,S} \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) + \delta \mathbf{I}_P \right]. \quad (26)$$

Using the first condition of (11), the first natural parameter can be written as

$$\boldsymbol{\Sigma}_*^{-1}\boldsymbol{\mu}_* = - \sum_{i=1}^N \mathbb{E}_{q_*(\mathbf{w})} [\nabla_{\mathbf{w}} \ell_i(\mathbf{w})] - \delta \boldsymbol{\mu}_* + \boldsymbol{\Sigma}_*^{-1}\boldsymbol{\mu}_* \quad (27)$$

$$\approx -\frac{1}{S} \sum_{i,s=1}^{N,S} \nabla_{\mathbf{w}} \ell_i(\mathbf{w}^{(s)}) - \delta \boldsymbol{\mu}_* + \left[ \frac{1}{S} \sum_{i,s=1}^{N,S} \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) + \delta \mathbf{I}_P \right] \boldsymbol{\mu}_* \quad (28)$$

$$= \frac{1}{S} \sum_{i,s=1}^{N,S} \left[ -\nabla_{\mathbf{w}} \ell_i(\mathbf{w}^{(s)}) + \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) \boldsymbol{\mu}_* \right]. \quad (29)$$

The equations above are similar to the natural parameters of the Laplace approximation (18) and (21) except that we have another sum which is taken over the samples ( $s$ ). We can treat them as observations. Given (26) and (29), the Variational-MC approximation  $q_*(\mathbf{w})$  is thus proportional to

$$\exp\left(\frac{-\delta \mathbf{w}^\top \mathbf{w}}{2}\right) \prod_{i,s=1}^{N,S} \exp\left[-\frac{1}{2S} \mathbf{w}^\top \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) \mathbf{w} + \frac{\mathbf{w}^\top}{S} \left\{ -\nabla_{\mathbf{w}} \ell_i(\mathbf{w}^{(s)}) + \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) \boldsymbol{\mu}_* \right\}\right]. \quad (30)$$

Here again, the key step is that we can use the GGN approximation (6) to show that the Variational-GGN-MC approximation  $\tilde{q}_*(\mathbf{w})$  is proportional to

$$\exp\left(\frac{-\delta \mathbf{w}^\top \mathbf{w}}{2}\right) \prod_{i,s=1}^{N,S} \exp\left[-\frac{1}{2} \mathbf{w}^\top \mathbf{J}_s(\mathbf{x}_i)^\top \frac{\boldsymbol{\Lambda}_{i,s}}{S} \mathbf{J}_s(\mathbf{x}_i) \mathbf{w} + \frac{\mathbf{w}^\top \mathbf{J}_s(\mathbf{x}_i)^\top}{S} \{\boldsymbol{\Lambda}_{i,s} \mathbf{J}_s(\mathbf{x}_i) \boldsymbol{\mu}_* - \mathbf{r}_{i,s}\}\right], \quad (31)$$

which can be expressed as a product of Gaussian distributions, provided that  $\boldsymbol{\Lambda}_{i,s} \succ 0$ . Here we define  $N \times S$  observations as  $\tilde{\mathbf{y}}_{i,s} := \mathbf{J}_s(\mathbf{x}_i)\boldsymbol{\mu}_* - \boldsymbol{\Lambda}_{i,s}^{-1}\mathbf{r}_{i,s}$  and get

$$\tilde{q}_*(\mathbf{w}) \propto \mathcal{N}(\mathbf{w}|0, \delta^{-1}\mathbf{I}_P) \prod_{i=1}^N \prod_{s=1}^S \mathcal{N}(\tilde{\mathbf{y}}_{i,s} | \mathbf{J}_s(\mathbf{x}_i)\mathbf{w}, S\boldsymbol{\Lambda}_{i,s}^{-1}). \quad (32)$$

As discussed in the proof of Theorem 1, the right hand side of the above equation is proportional to the posterior distribution of a linear basis-function model  $\tilde{\mathbf{y}}_{i,s} = \mathbf{J}_s(\mathbf{x}_i)\mathbf{w} + \boldsymbol{\epsilon}_{i,s}$  with Gaussian noise  $\boldsymbol{\epsilon}_{i,s} \sim \mathcal{N}(0, S\boldsymbol{\Lambda}_{i,s}^{-1})$  and prior distribution  $\mathbf{w} \sim \mathcal{N}(0, \delta^{-1}\mathbf{I}_P)$ . The predictive distribution of this model can be equivalently written in the function space, giving rise to a GP regression model with kernel  $\delta^{-1}\mathbf{J}_s(\mathbf{x})\mathbf{J}_{s'}(\mathbf{x}')^\top$ .  $\square$

### A.3 Proof of Theorem 3

**Theorem 3. (Iterations of a VI algorithm as a GP) :** At each iteration  $t$  of the VOGGN update, i.e. (14) and (15) but with the GGN approximation to the Hessian, the posterior approximation  $q_t(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$  is equivalent to the posterior distribution of the following linear basis-function model:

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{J}_s(\mathbf{x}_i)\mathbf{w} + \epsilon_{i,s}, \text{ with } \epsilon_{i,s} \sim \mathcal{N}(0, S(\beta_t \boldsymbol{\Lambda}_{i,s})^{-1}) \text{ and } \mathbf{w} \sim \mathcal{N}(\mathbf{m}_t, \mathbf{S}_t) \quad (16)$$

where  $\mathbf{S}_t^{-1} := (1 - \beta_t)\boldsymbol{\Sigma}_t^{-1} + \beta_t\delta\mathbf{I}_P$  and  $\mathbf{m}_t := (1 - \beta_t)\mathbf{S}_t\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t$ . The observations are defined as  $\tilde{\mathbf{y}}_{i,s} := \mathbf{J}_s(\mathbf{x}_i)\boldsymbol{\mu}_t - \boldsymbol{\Lambda}_{i,s}^{-1}\mathbf{r}_{i,s}$  and the feature maps are defined as  $\mathbf{J}_s(\mathbf{x}_i)^\top$  with  $\mathbf{J}_s(\mathbf{x}_i)$ ,  $\mathbf{r}_{i,s}$ , and  $\boldsymbol{\Lambda}_{i,s}$  denoting the Jacobian, residual, and noise precision evaluated at the sampled weight  $\mathbf{w}^{(s)}$  drawn from  $q_t(\mathbf{w})$ . The predictive distribution of this model is equivalent to that of the following GP regression model:

$$\tilde{\mathbf{y}}_{i,s} = \mathbf{f}_s(\mathbf{x}_i) + \epsilon_{i,s}, \quad \text{where } \mathbf{f}_s(\mathbf{x}) \sim \mathcal{GP}(\mathbf{J}_s(\mathbf{x})\mathbf{m}_t, \mathbf{J}_s(\mathbf{x})\mathbf{S}_t\mathbf{J}_s'(\mathbf{x}')^\top). \quad (17)$$

*Proof.* To prove this theorem, we first need to derive an update formula for the natural-parameters of the approximation  $q_t(\mathbf{w}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ , denoted by  $\boldsymbol{\eta}_t := \{\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t, -\frac{1}{2}\boldsymbol{\Sigma}_t^{-1}\}$ , based on equations (14) and (15) which provides an update formula for the  $\boldsymbol{\mu}_t$  and  $\boldsymbol{\Sigma}_t^{-1}$ , respectively. As before, we use Monte Carlo sampling for the expectations. Given (15), the update corresponding to the second natural-parameter is obvious

$$-\frac{1}{2}\boldsymbol{\Sigma}_{t+1}^{-1} = (1 - \beta_t) \left[-\frac{1}{2}\boldsymbol{\Sigma}_t^{-1}\right] - \frac{1}{2}\beta_t \left[ \sum_{i=1}^N \mathbb{E}_{q_t(\mathbf{w})} [\nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w})] + \delta\mathbf{I}_P \right] \quad (33)$$

$$\approx (1 - \beta_t) \left[-\frac{1}{2}\boldsymbol{\Sigma}_t^{-1}\right] - \frac{1}{2}\beta_t \left[ \frac{1}{S} \sum_{i,s=1}^{N,S} \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) + \delta\mathbf{I}_P \right]. \quad (34)$$

For the first natural-parameter update, we multiply (14) by  $\boldsymbol{\Sigma}_{t+1}^{-1}$  and get

$$\boldsymbol{\Sigma}_{t+1}^{-1}\boldsymbol{\mu}_{t+1} = \boldsymbol{\Sigma}_{t+1}^{-1}\boldsymbol{\mu}_t - \beta_t \left[ \sum_{i=1}^N \mathbb{E}_{q_t(\mathbf{w})} [\nabla_{\mathbf{w}} \ell_i(\mathbf{w})] + \delta\boldsymbol{\mu}_t \right] \quad (35)$$

$$= (1 - \beta_t) [\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t] + \beta_t \sum_{i=1}^N [-\mathbb{E}_{q_t(\mathbf{w})} [\nabla_{\mathbf{w}} \ell_i(\mathbf{w})] + \mathbb{E}_{q_t(\mathbf{w})} [\nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w})] \boldsymbol{\mu}_t] \quad (36)$$

$$\approx (1 - \beta_t) [\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t] + \frac{\beta_t}{S} \sum_{i,s=1}^{N,S} [-\nabla_{\mathbf{w}} \ell_i(\mathbf{w}^{(s)}) + \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) \boldsymbol{\mu}_t], \quad (37)$$

where in the second step, we replaced  $\boldsymbol{\Sigma}_{t+1}^{-1}$  in the first term by (15). Note that at convergence (34) and (37) correspond to (26) and (29), respectively. The posterior approximation  $q_{t+1}(\mathbf{w})$  at time  $t+1$  is hence proportional to

$$p^{\beta_t} q_t^{1-\beta_t} \prod_{i,s=1}^{N,S} \exp \left[ -\frac{\beta_t}{2S} \mathbf{w}^\top \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) \mathbf{w} + \frac{\beta_t \mathbf{w}^\top}{S} \left\{ -\nabla_{\mathbf{w}} \ell_i(\mathbf{w}^{(s)}) + \nabla_{\mathbf{w}\mathbf{w}}^2 \ell_i(\mathbf{w}^{(s)}) \boldsymbol{\mu}_t \right\} \right], \quad (38)$$

where  $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|0, \delta^{-1}\mathbf{I}_P)$  is the prior distribution (the dependence of  $p(\mathbf{w})$  and  $q_t(\mathbf{w})$  on  $\mathbf{w}$  have been dropped in equation above). For the product of posterior approximation at time  $t$  and prior in (38), we obtain the following unnormalized Gaussian:

$$p(\mathbf{w})^{\beta_t} q_t^{1-\beta_t} = \mathcal{N}(\mathbf{w}|0, \delta^{-1}\mathbf{I}_P)^{\beta_t} \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)^{1-\beta_t} \quad (39)$$

$$\propto \mathcal{N}(\mathbf{w} | ((1 - \beta_t)\boldsymbol{\Sigma}_t^{-1} + \beta_t\delta\mathbf{I}_P)^{-1} ((1 - \beta_t)\boldsymbol{\Sigma}_t^{-1}\boldsymbol{\mu}_t), \quad (40)$$

$$((1 - \beta_t)\boldsymbol{\Sigma}_t^{-1} + \beta_t\delta\mathbf{I}_P)^{-1}) := \mathcal{N}(\mathbf{w}|\mathbf{m}_t, \mathbf{S}_t). \quad (41)$$

Next, for the product over  $i$  and  $s$  in (38), we employ the GGN approximation (6), as we did before in proofs, to obtain the following

$$\prod_{i,s=1}^{N,S} \exp \left[ -\mathbf{w}^\top \mathbf{J}_s(\mathbf{x}_i)^\top \frac{\beta_t \mathbf{\Lambda}_{i,s}}{2S} \mathbf{J}_s(\mathbf{x}_i) \mathbf{w} + \frac{\beta_t \mathbf{w}^\top \mathbf{J}_s(\mathbf{x}_i)^\top}{S} \{ \mathbf{\Lambda}_{i,s} \mathbf{J}_s(\mathbf{x}_i) \boldsymbol{\mu}_t - \mathbf{r}_{i,s} \} \right]. \quad (42)$$

We use  $\tilde{q}_t(\mathbf{w})$  to denote the resulting distribution. We complete the proof by expressing each term in the product above as a Gaussian distribution, provided that  $\mathbf{\Lambda}_{i,s} \succ 0$ , and defining  $N \times S$  observations as  $\tilde{\mathbf{y}}_{i,s} := \mathbf{J}_s(\mathbf{x}_i) \boldsymbol{\mu}_t - \mathbf{\Lambda}_{i,s}^{-1} \mathbf{r}_{i,s}$  to get

$$\tilde{q}_t(\mathbf{w}) \propto \mathcal{N}(\mathbf{w} | \mathbf{m}_t, \mathbf{S}_t) \prod_{i,s=1}^{N,S} \mathcal{N}(\tilde{\mathbf{y}}_{i,s} | \mathbf{J}_s(\mathbf{x}_i) \mathbf{w}, S(\beta_t \mathbf{\Lambda}_{i,s})^{-1}), \quad (43)$$

Note that the right hand side of the above equation is proportional to the posterior distribution of a linear basis-function model  $\tilde{\mathbf{y}}_{i,s} = \mathbf{J}_s(\mathbf{x}_i) \mathbf{w} + \epsilon_{i,s}$  with Gaussian noise  $\epsilon_{i,s} \sim \mathcal{N}(0, S(\beta_t \mathbf{\Lambda}_{i,s})^{-1})$  and prior distribution  $\mathcal{N}(\mathbf{w} | \mathbf{m}_t, \mathbf{S}_t)$ . The predictive distribution of this model can be equivalently written in the function space with a GP kernel. To sum up, we have shown that a step in the VOGGN update (14) and (15) with GGN approximations and MC sampling is equivalent to inference in a GP regression model.  $\square$

## B Computation of Posterior Predictive using GP regression

Typically, we can always predict using Monte Carlo sampling from the Gaussian approximation, however this might be too noisy sometimes. In this section, we show how we can directly use the GP regression model to *approximate* the posterior predictive distribution. We only describe the method for Laplace approximation. This can be generalized to VI.

Given a test input, denoted by  $\mathbf{x}_{\text{test}}$ , we first compute the feature map  $\mathbf{J}_*(\mathbf{x}_{\text{test}})^\top$ . Using the linear model, we can compute the posterior predictive mean of the output which we denote by  $\tilde{\mathbf{y}}_{\text{test}}$ . However, to be able to compute the predictive distribution of the actual output  $\mathbf{y}_{\text{test}}$ , we need to invert the map from  $\mathbf{y}_{\text{test}}$  to  $\tilde{\mathbf{y}}_{\text{test}}$ . The expressions for this map are illustrated below for the squared loss.

**Example 1 (Squared loss).** Consider the least-squares regression,  $\ell(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{2\sigma^2} \|\mathbf{y} - \hat{\mathbf{y}}\|^2$  with  $\sigma^2$  as the noise variance. In this case, we have  $\mathbf{r}_{i,*} := \sigma^{-2}(\mathbf{f}_*(\mathbf{x}_i) - \mathbf{y}_i)$  and  $\mathbf{\Lambda}_{i,*} := \sigma^{-2} \mathbf{I}_K$ . Theorem 1 gives rise to a GP model with following observations and noise

$$\tilde{\mathbf{y}}_i := \mathbf{y}_i - \mathbf{f}_*(\mathbf{x}_i) + \mathbf{J}_*(\mathbf{x}_i) \mathbf{w}_*, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2 \mathbf{I}) \quad (44)$$

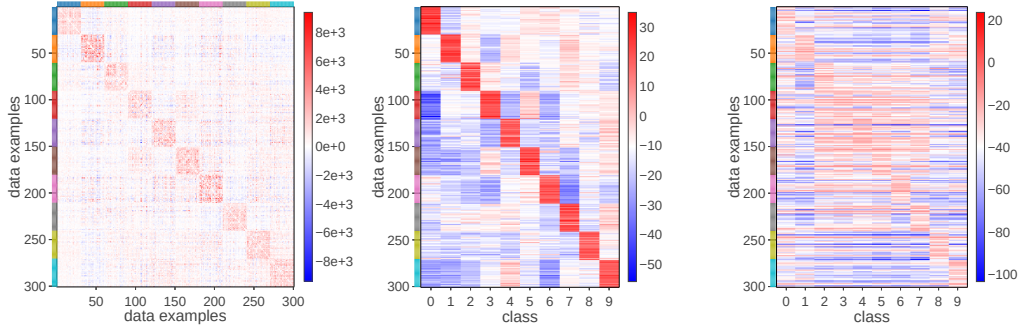
Using the expression above in (44), we can compute the predictive mean and variance of  $\mathbf{y}_{\text{test}}$  as it is displayed in Fig. 2.

## C Additional Results

In this appendix, we provide additional figures to the ones presented in Sec. 5.2.

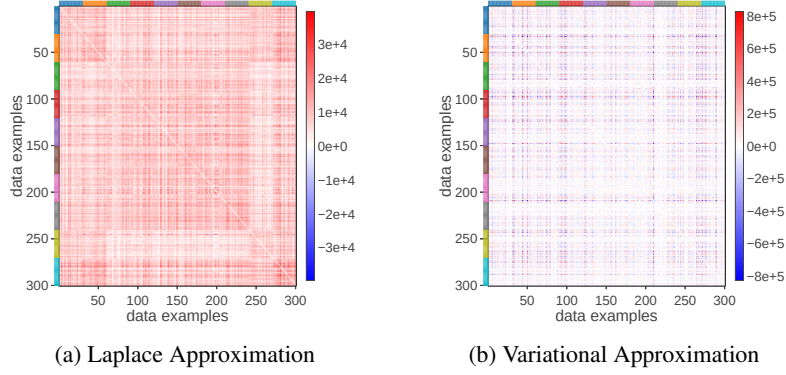
Fig. 6 is parallel to Fig. 3 but uses the variational approximation instead of a Laplace approximation. While the posterior mean on MNIST shows very similar structure for both approximations, the kernel shows some interesting differences. There are many more negative correlations between examples from different classes in the kernel corresponding to the variational approximation. The posterior mean on CIFAR-10 has similar structure yet it appears to exhibit higher uncertainty. In Fig. 7, we show the kernel matrix on 300 data points of CIFAR-10 with the respective class labels. The kernel is computed for both the Laplace and variational approximation but shows less structure than that of the MNIST dataset.

To analyze the structure of the posterior mean  $\mathbb{E}(f(\mathbf{x}) | \mathcal{D})$ , in Fig. 8 we visualize the first 4 principal components of the posterior mean obtained with both variational and Laplace approximation on MNIST and CIFAR-10. For MNIST, the classes are clearly separable corresponding to the high accuracy attained. On CIFAR-10 the representation does not allow to identify simple classification boundaries.



(a) MNIST: GP kernel matrix (left) and GP posterior mean (right). (b) CIFAR-10: GP posterior mean.

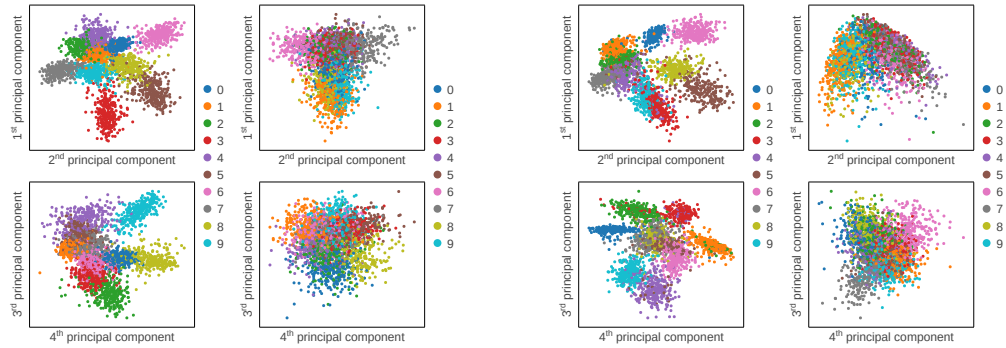
Figure 6: This figure visualizes the GP kernel matrix and posterior mean for LeNet5 trained with VOGN on MNIST (left) and CIFAR-10 (right). The kernel matrix clearly shows the correlations learned by the DNN. A higher posterior mean is assigned to the correct label which reflects the accuracy obtained by the DNN.



(a) Laplace Approximation

(b) Variational Approximation

Figure 7: GP kernels due to Laplace and variational approximation for neural networks on CIFAR-10. The kernels show slight traces of structure but are not as significant as the ones presented on MNIST in Sec. 5.



(a) Laplace Approximation: MNIST (left) and CIFAR-10 (right)

(b) Variational Approximation: MNIST (left) and CIFAR-10 (right)

Figure 8: Scatter plot of data points of MNIST and CIFAR-10 projected onto the first four principal components of the posterior mean  $\mathbf{m}(\mathbf{x}|\mathcal{D}) \in \mathbb{R}^{10}$ . For MNIST, both methods provide well separable clusters. On CIFAR-10 this is not possible which is also suggested by low achieved accuracy.