

Flow-Induced Diagonal Gaussian Processes

Moule Lin¹, Andrea Patane¹, Weipeng Jing², Shuhao Guan³, Goetz Botterweck¹,

¹ Trinity College Dublin, Dublin, Ireland

² Northeast Forestry University, Heilongjiang, China

³ University College Dublin, Dublin, Ireland

{moulel, apatane, goetz.botterweck}@tcd.ie, jwp@nefu.edu.cn, shuhao.guan@ucdconnect.ie

Abstract

We present Flow-Induced Diagonal Gaussian Processes (**FiD-GP**), a compression framework that incorporates a compact inducing weight matrix to project a neural network’s weight uncertainty into a lower-dimensional subspace. Critically, **FiD-GP** relies on normalising-flow priors and spectral regularisations to augment its expressiveness and align the inducing subspace with feature-gradient geometry through a numerically stable projection mechanism objective. Furthermore, we demonstrate how the prediction framework in **FiD-GP** can help to design a single-pass projection for Out-of-Distribution (OoD) detection. Our analysis shows that **FiD-GP** improves uncertainty estimation ability on various tasks compared with SVGP-based baselines, satisfies tight spectral-residual bounds with theoretically guaranteed OoD detection, and significantly compresses the neural network’s storage requirements at the cost of increased inference computation dependent on the number of inducing weights employed. Specifically, in a comprehensive empirical study spanning regression, image classification, semantic segmentation, and out-of-distribution detection benchmarks, it cuts Bayesian training cost by several orders of magnitude, compresses parameters by roughly 51%, reduces model size by about 75%, and matches state-of-the-art accuracy and uncertainty estimation.

Code: <https://github.com/anonymousspaper987/FiD-GP.git>

1 Introduction

Reliable uncertainty estimates are especially crucial and sought after in safety-critical applications of neural networks, like autonomous driving (Hubmann et al. 2017), medical diagnosis (Chua et al. 2023), and many others (Blasco, Sánchez, and García 2024; Hernández-Lobato and Adams 2015; Zyphur and Oswald 2015). Research on predictive uncertainty has expanded in multiple directions. Bayesian Neural Networks (BNNs) (Kononenko 1989; MacKay 1995; Thodberg 1996), ensemble methods (Hoffmann, Fortmeier, and Elster 2021; Rahaman et al. 2021), and distance-aware frameworks (Mukhoti et al. 2023; Liu et al. 2020; Zhang, Das, and Kumar 2024) have emerged as prominent approaches that produce strong performance on estimating uncertainty and related benchmarks. In these settings, models are expected not only to deliver accurate predictions but also to gener-

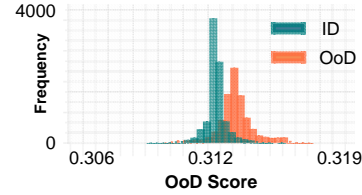


Figure 1: Distribution of predictive scores generated by the ResNet-18 model equipped with Sparse Variational Gaussian Processes (SVGP). In-distribution (ID) dataset is CIFAR-100, Out-of-Distribution (OoD) dataset is CIFAR-10.

ate reliable confidence estimates, particularly for identifying Out-of-Distribution (OoD) inputs.

Unfortunately, although progress has been made, the more significant computational overhead required compared to deterministic counterparts during training and inference still limits the widespread adoption of these approaches in practical industry settings. In turn, it has spurred research into more efficient and effective uncertainty estimation methods, such as Rank-1 BNN (Dusenberry et al. 2020), lower uncertainty space (Sparse Gaussian Processes or VAE) (Ritter et al. 2021; Franchi et al. 2023), distance-aware frameworks (Van Amersfoort et al. 2020; Liu et al. 2020; Mukhoti et al. 2023) as well as quantisation methods (Lin et al. 2023, 2025; Hubin and Storvik 2024; Ritter et al. 2021).

Meanwhile, Gaussian Processes (GP) (Seeger 2004; Williams and Rasmussen 1995; Hida and Hitsuda 1993; Opper and Winther 2000) can represent an infinitely wide network with a finite number of parameters. Sparse Gaussian Processes (SGPs) (Snelson and Ghahramani 2005; Kuss and Rasmussen 2005; Wei et al. 2024) project the full GP onto a small inducing matrix and thereby dramatically lower computational cost by restricting uncertainty to a low-dimensional subspace and making inference tractable. Indeed, research has recently looked at obtaining the best of both worlds: the high representation capacity of BNNs, combined with the mathematical simplicity of GPs (Seeger 2004; Damianou and Lawrence 2013; Hida and Hitsuda 1993), particularly in the form of SGPs applied to encode BNNs’ parametric distributions (Snelson and Ghahramani 2005; Titsias 2009; Leibfried et al. 2020). Unfortunately, despite their elegance, sparse-GP wrappers on BNNs still underperform on deep networks as

limited expressiveness, stemming from their simple Gaussian prior over a low-dimensional inducing subspace, restrictive stationary kernel assumptions, and many other factors, not only degrades predictive accuracy and uncertainty calibration (Swiler et al. 2020; Lawrence, Seeger, and Herbrich 2002) but also fails to separate In-distribution from Out-of-Distribution input.

As shown in Figure 1 for example, the predictive-score distributions produced by the ResNet-18+SVGP model on CIFAR-100 (ID) and CIFAR-10 (OoD) overlap substantially, indicating that the simple Gaussian inducing prior cannot adequately discriminate OoD samples. This overlap manifests as unreliable confidence estimates and motivates the current work, focused on a flow-induced prior to enhance feature correlation modeling in the BNN, latent, weight space.

In this paper, we present a framework that shapes the SGP inducing prior with a normalising flow equipped with spectral regularisation to model complex, multi-modal feature correlations. The SGP so designed, is used to express and compress the distributions of several BNN architectures, formally through employing a hierarchical prior over the latter’s parameter space. We adapted a Kronecker-structure to present the covariance of the inducing matrix and its associated weights for efficient training and inference. In the context of uncertainty’s practical applications, we further showcase how our modelling framework can be used to derive an Out-of-Distribution (OoD) detection system based on a process that projects the feature space into the inducing-matrix space, with theoretical guarantees for feature-inducing alignment provided by the normalising flow and spectral regularisation, which forms a **single-pass ID/OoD** detection mechanism.

We perform an extensive empirical investigation on the effectiveness of our method across regression, image classification, and semantic segmentation tasks. We utilise a synthetic 1-D function for evaluating regression performance, following (Miyato et al. 2018) and (Ritter et al. 2021), and conduct classification experiments on CIFAR-100 (Krizhevsky, Hinton et al. 2009) and ImageNet-1k (Deng et al. 2009) using ResNet-18 (He et al. 2016) as the base backbone. For semantic segmentation, we validate our method on widely adopted benchmarks, including CamVID (Brostow, Fauqueur, and Cipolla 2009) and CityScapes (Cordts et al. 2016), using FCN-ResNet50 (Long, Shelhamer, and Darrell 2015) and HRNet-W48 (Sun et al. 2019) as the backbone networks. Our approach consistently achieves state-of-the-art performance compared with several recent methods, including strong deterministic baselines, BatchEnsemble, FFG-U (Ritter et al. 2021), and F-SGVB-LRT (Nguyen et al. 2023).

Our main contributions:

- We introduce Flow-Induced Diagonal Gaussian Processes (**FiD-GP**), a GP-inspired uncertainty module that integrates with off-the-shelf BNN architectures and builds on sparse Gaussian processes, normalising-flow priors and Bayesian Kronecker covariance, while preserving a Gaussian-form posterior over inducing matrix.
- We develop a jittered-Cholesky projection objective that aligns inducing-point subspaces with feature-gradient geometry, producing a single-pass projection score with

tight spectral-residual bounds and theoretically guaranteed near-perfect OoD discrimination.

- We conducted comprehensive empirical experiments across regression, image classification, semantic segmentation, and Out-of-Distribution detection benchmarks, demonstrating significant reductions in training cost, approximately 51% parameter compression, and achieving state-of-the-art accuracy and uncertainty estimation without heavy post-hoc calibration.

2 Related Work

Uncertainty estimation for modern artificial neural networks remains a long-standing and significant challenge, particularly in safety-critical domains (Blei, Kucukelbir, and McAuliffe 2017; Snelson and Ghahramani 2007; Arhonditsis et al. 2017). Many studies have investigated the balance of efficiency and expressiveness in estimating uncertainty from different perspectives. In this work we look at combining GPs, in particular sparse GPs, together with BNNs.

Considering the inference efficiency, several works have looked at decomposing the inducing points or redesigning the inference process. Decoupled Gaussian Processes (DGPs) (Cheng and Huang 2015) push SVGPs, separating the bases for the mean and covariance, resulting in the mean growing linearly without enlarging the cubic bottleneck. Greater expressiveness can also be achieved through structured inducing domains. This was later extended to Convolution (Van der Wilk, Rasmussen, and Hensman 2017) through the construction of an inter-domain inducing matrix approximation that is well-tailored to the convolutional kernel. Recent research has proposed Gaussian posterior approximations for BNNs with efficient covariance structures (Ritter, Botev, and Barber 2018; Mishkin et al. 2018).

SVGPs scale exact GPs to $\mathcal{O}(M^3)$ via $M \ll N$ inducing points (Titsias 2009; Hensman et al. 2015), where N is the total number of parameters of the neural network and M denotes the dimensionality of the inducing matrix, but their Gaussian variational posterior is limited in expressivity (Titsias 2009). To enrich this, normalising flows, e.g. Real NVP (Dinh, Sohl-Dickstein, and Bengio 2017), MAF (Papamakarios, Pavlakou, and Murray 2017), FFGORD (Grathwohl et al. 2018) and Inverse Autoregressive Flow (Kingma et al. 2016) have been applied to the inducing points outputs to capture more flexible, non-Gaussian marginals and tighter ELBOs (Rezende and Mohamed 2015; Cutajar et al. 2016; Rezende and Mohamed 2015; Kingma et al. 2016).

Beyond modelling predictive uncertainty within the training distribution, several works seek to detect and properly score samples that fall outside it. The earliest attempts rely on likelihood-based generative models, where normalising flows or autoregressive densities assume higher log-likelihood on in-distribution data than OoD inputs (Dinh, Sohl-Dickstein, and Bengio 2017; Kingma and Dhariwal 2018; Nalisnick et al. 2019; Ren et al. 2019). This further motivates the reconstruction-error paradigm, which treats the difficulty of rebuilding an input via autoencoders or memory-augmented networks as an anomaly signal (An and Cho 2015; Gong et al. 2019). Recently, there has been increasing interest in

integrating the Gaussian Process (GP) perspective into deep architectures: distance-aware single-pass heads (SNGP) (Liu et al. 2020), kernel-density hybrids (DUQ/DUE) (Van Amersfoort et al. 2020), deterministic GP post-processing (DDU) (Mukhoti et al. 2023), logit-level GP unification (Chen et al. 2024), and sparse or geometric variants.

These works investigated the trade-off between efficiency and expressiveness; however, they still either sacrifice closed-form calibration, rely on multiple forward passes, or impose specialised architectural constraints. Instead, our approach shares parameters via the augmented prior with an efficient low-rank structure.

3 Preliminaries

3.1 Sparse Gaussian Process

We first reviews the core concepts of SGPs and their variational inference, then show how to integrate them with BNNs to enable end-to-end uncertainty estimation.

A Gaussian Process (GP) (Williams and Rasmussen 1995) defines a distribution over functions: $f(\cdot) \sim \mathcal{GP}(m(\cdot), \Sigma(\cdot, \cdot))$ where m is the mean and $\Sigma(\cdot, \cdot)$ the kernel.

Introduce $M \ll N$ inducing points \mathbf{U} with variables $\mathbf{u} = f(\mathbf{U})$. We place the prior

$$p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, \Sigma_{MM}) \quad (1)$$

and approximate the posterior by

$$\mathcal{L} = \sum_{n=1}^N \mathbb{E}_{q(f_n)} [\log p(y_n | f_n)] - \text{KL}[q(\mathbf{u}) \| p(\mathbf{u})] \quad (2)$$

The conditional prior over $\mathbf{f} = [f(\mathbf{x}_n)]_{n=1}^N$ is

$$p(\mathbf{w} | \mathbf{u}) = \mathcal{N}(\Sigma_{NM} \Sigma_{MM}^{-1} \mathbf{u}, \Sigma_{NN} - \Sigma_{NM} \Sigma_{MM}^{-1} \Sigma_{MN}) \quad (3)$$

where $\Sigma_{NM}[n, m] = K(\mathbf{x}_n, \mathbf{z}_m)$ and $\Sigma_{NN}[n, n'] = K(\mathbf{x}_n, \mathbf{x}_{n'})$. K is the kernel function, and here we identify $\mathbf{w} = \mathbf{f}$ as the vector of weights in our Bayesian Neural Network. This reduces inference to $\mathcal{O}(NM^2 + M^3)$ time and $\mathcal{O}(NM + M^2)$ memory.

(For more details see Appendix A.)

3.2 Kronecker-Structured Covariance

Using Kronecker identities, the transforms become

$$T_{\text{row}} = \Sigma_{W,U}^{(\text{row})} (\Sigma_U^{(\text{row})})^{-1}, \quad T_{\text{col}} = \Sigma_{W,U}^{(\text{col})} (\Sigma_U^{(\text{col})})^{-1} \quad (4)$$

Hence the conditional mean of W given U is simply

$$\mathbb{E}[W | U] = T_{\text{row}} U T_{\text{col}}^\top \quad (5)$$

and sampling follows Matheron’s rule:

$$W | U = W_{\text{prior}} + T_{\text{row}} (U - U_{\text{prior}}) T_{\text{col}}^\top \quad (6)$$

(The detailed derivation is provided in Appendix B.)

Whitened Representation: To improve numerical stability during sampling, we adopt a whitened parameterization. Let $K_U = LL^\top$ be the Cholesky decomposition of the prior covariance, and define the whitened variable $\mathbf{v} = L^{-1}\mathbf{u}$. The prior becomes: $p(\mathbf{v}) = \mathcal{N}(\mathbf{v} | \mathbf{0}, I_M)$ with variational distribution $q(\mathbf{v}) = \mathcal{N}(\mathbf{v} | \tilde{\mathbf{m}}, \tilde{\mathbf{S}})$. The Matheron sampling rule in whitened space is:

$$W | \mathbf{v} = W_{\text{prior}} + T_{\text{row}} L(\mathbf{v} - \mathbf{v}_{\text{prior}}) T_{\text{col}}^\top \quad (7)$$

3.3 Normalisation Flow with Spectral regularisation

Spectral normalisation (Miyato et al. 2018) enforces a 1-Lipschitz constraint by normalising each weight matrix:

$$\tilde{W} = \frac{W}{\sigma_{\max}(W)}, \quad \sigma_{\max}(W) = \sup_{\|x\|_2=1} \|Wx\|_2. \quad (8)$$

In a Normalising Flow g_ϕ , each layer’s Jacobian $J_l = D_l \tilde{W}_l$ satisfies $|\det \tilde{W}_l| \leq 1$, so

$$|\det J_g| = \prod_l |\det J_l| \leq \prod_{l,i} \phi'_i(h_{l,i}) \quad (9)$$

ensuring the overall Jacobian determinant remains tractable.

4 Methodology

In this section, we divide our approach into two parts. The overall framework is shown in Figure 2, which comprises the standard uncertainty-estimation model (Section 4.1) and a robust Out-of-Distribution detection module (Section 4.2).

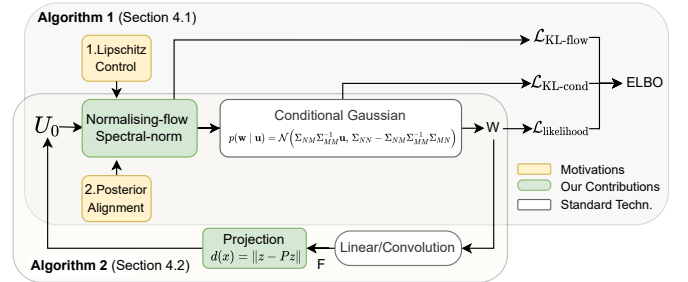


Figure 2: Overview of our approach (**FiD-GP**): Flow-based conditional Gaussian with spectral control and projection residual.

4.1 Flow-Based Priors for Uncertainty Estimation

We initially applied a Normalising Flow with Spectral Normalisation to our inducing-point matrix (see Definition 1), so that the prior is no longer a simple zero-mean Gaussian but instead follows a richer, non-Gaussian form. Surprisingly, however, by exploiting the conditional-Gaussian identity (Eq. 3), the resulting posterior over the inducing matrix remains Gaussian (See the detailed derivation in Appendix B.) and enforce a 1-Lipschitz mapping and ensuring tractable Jacobian determinants in the flow-based prior.

Definition 1 (Normalising Flow with Spectral Normalisation Prior). Let $p_{\mathbf{z}}(\mathbf{z}) = \mathcal{N}(\mathbf{z} | \mathbf{0}, \mathbf{I}_M)$ be a base Gaussian in \mathbb{R}^M , and let

$$g_\phi : \mathbb{R}^M \rightarrow \mathbb{R}^M \quad (10)$$

be a smooth, invertible mapping parameterised by ϕ . Each linear layer in the flow g_ϕ is equipped with spectral normalisation (Miyato et al. 2018), constraining its Lipschitz constant and stabilising training. This also ensures that the Jacobian determinant in

$$p(\mathbf{u}) = p_{\mathbf{z}}(g_\phi^{-1}(\mathbf{u})) |\det \nabla_{\mathbf{u}} g_\phi^{-1}(\mathbf{u})| \quad (11)$$

remains tractable.

Algorithm 1: Variational Training with Flow-based Prior

Require: Dataset \mathcal{D} , network f_θ , inducing params (m, S) , spectral-norm flow g_ϕ , hyper-params (λ, α, \dots)
Ensure: Learned θ, ϕ, m, S

- 1: **while** not converged **do**
- 2: **Sample base inducing** \triangleright Gaussian in Eq. (1)
- 3: $\mathbf{u}_0 \sim \mathcal{N}(0, IM)$
- 4: **Normalising-flow transform**
- 5: $(\mathbf{u}, \log |\det J_g|) \leftarrow g_\phi(\mathbf{u}_0)$ \triangleright Eq. (9)
- 6: **if** whitened_u **then**
- 7: $\mathbf{u} \leftarrow L_{\text{row}} \mathbf{u} L_{\text{col}}^\top$ \triangleright jittered-Cholesky scales
- 8: **end if**
- 9: **Kronecker weight draw**
- 10: $M_w \leftarrow T_{\text{row}} \mathbf{u} T_{\text{col}}^\top$ \triangleright Eq. (4)
- 11: $W \leftarrow \text{cg}(M_w)$ \triangleright cg is Eq. (3)
- 12: **Likelihood**
- 13: $\mathcal{L}_{\text{loglik}} \leftarrow \mathbb{E}[\log p(\mathcal{D} | W, b)]$
- 14: **KL-flow part**
- 15: $\mathcal{L}_{\text{KL-flow}} \leftarrow \log q_0(\mathbf{u}_0) - \log |\det J_g| - \log p(\mathbf{u})$
- 16: **KL-conditional part**
- 17: $\mathcal{L}_{\text{KL-cond}} \leftarrow \frac{D}{2} (\lambda^2 - 1 - 2 \log \lambda)$ \triangleright Eq. (12)
- 18: **ELBO & update**
- 19: $\text{ELBO} \leftarrow \mathcal{L}_{\text{likelihood}} - \mathcal{L}_{\text{KL-flow}} - \mathcal{L}_{\text{KL-cond}}$
- 20: $\nabla(-\text{ELBO}) \rightarrow \text{Adam}(\theta, \phi, m, S)$
- 21: **end while**

Substitute the $p(\mathbf{u})$ into standard SVGP Evidence Lower Bound (ELBO) function, and then the ELBO becomes:

$$\begin{aligned}
 \mathcal{L}_{\text{ELBO}} = & \underbrace{\mathbb{E}_{q_0, \epsilon} [\log p(\mathcal{D} | W)]}_{\text{Expected log-likelihood}} \\
 & - \underbrace{\mathbb{E}_{q_0} [\log q_0 - \log |\det J_g| - \log p(g(u_0))]}_{\text{KL divergence of flow-based inducing posterior}} \\
 & - \underbrace{\frac{D}{2} (\lambda^2 - 1 - 2 \log \lambda)}_{\text{Conditional Gaussian KL}}
 \end{aligned} \quad (12)$$

(For more details see Appendix C.)

where q_0 is the base distribution of latent variables u_0 , ϵ is Gaussian noise for reparameterisation, W denotes neural network weights sampled from u_0 and ϵ , \mathcal{D} is the dataset, g is a flow transformation with Jacobian determinant $|\det J_g|$, p denotes prior or likelihood densities, D is the weight dimension, and λ controls the conditional Gaussian variance. The pseudo-code of our approach is presented in Algorithm 1.

4.2 Efficient Single-Pass ID/OoD

The flow-transformed inducing variables \mathbf{u} align with In-distribution features and diverge from Out-of-Distribution ones, which provides training-free, projection-based ID/OoD separation.

For each layer ℓ , compute the feature-gradient vector:

$$\mathbf{z}_i^{(\ell)} = \text{vec}(h^{(\ell)}(x_i)) \odot \text{vec}(\nabla_{h^{(\ell)}} \ell(y_i, \hat{y}_i)) \in \mathbb{R}^{N_\ell} \quad (13)$$

Project $\mathbf{z}_i^{(\ell)}$ onto the row-space of $U^{(\ell)} \in \mathbb{R}^{M_\ell \times N_\ell}$ by solv-

Algorithm 2: One-Pass ID/OoD Scoring

Require: Trained model f_θ , key layer set \mathcal{L} , test batch $\{\mathbf{x}_i\}$, optional projector \mathcal{P}
Ensure: OoD scores $\{s_i\}$

Pre-compute: for each $\ell \in \mathcal{L}$ sample $\mathbf{U}^{(\ell)}$; build $P^{(\ell)}$ with Eq. 14.

- 1: **for all** \mathbf{x}_i **do**
- 2: Run one forward-backward pass to get $\mathbf{f}_i^{(\ell)}, \mathbf{g}_i^{(\ell)}$
- 3: **for all** $\ell \in \mathcal{L}$ **do**
- 4: $\mathbf{z}_i^{(\ell)} \leftarrow (\mathbf{f}_i^{(\ell)} \odot \mathbf{g}_i^{(\ell)})$
- 5: **if** \mathcal{P} exists **then**
- 6: $\mathbf{z}_i^{(\ell)} \leftarrow \mathcal{P}^{(\ell)}(\mathbf{z}_i^{(\ell)})$
- 7: **end if**
- 8: $\mathbf{r}_i^{(\ell)} \leftarrow (I - P^{(\ell)}) \mathbf{z}_i^{(\ell)}$
- 9: **end for**
- 10: $s_i \leftarrow \frac{1}{|\mathcal{L}|} \sum_\ell \|\mathbf{r}_i^{(\ell)}\|_2$ \triangleright Eq. (15)
- 11: **end for**

ing a regularised least-squares problem:

$$\begin{aligned}
 \tilde{\mathbf{z}}_i^{(\ell)} &= P^{(\ell)} \mathbf{z}_i^{(\ell)}, \\
 P^{(\ell)} &= \arg \min_P \|P U^{(\ell)} - I\|_F^2 + \lambda \|P\|_F^2
 \end{aligned} \quad (14)$$

Finally, define the OoD score as the average projection residual:

$$S = \inf_{\|x\|=1} \|(I - P) g(T_{\text{row}} U T_{\text{col}}^\top x)\|, \quad (15)$$

Lemma 1 (Spectral Residual Separation). *Let*

$$W = T_{\text{row}} U T_{\text{col}}^\top + E. \quad (16)$$

Define the projector

$$P = U^\top (U U^\top)^{-1} U, \quad (17)$$

and let g be a 1-Lipschitz flow. Set

$$S = \inf_{\|x\|=1} \|(I - P) g(T_{\text{row}} U T_{\text{col}}^\top x)\|, \quad (18)$$

$$d(x) = \|(I - P) g((T_{\text{row}} U T_{\text{col}}^\top + E) x)\|. \quad (19)$$

If during training

$$S > 2 \|E\|, \quad (20)$$

then

$$\sup_{x_{\text{ID}}} d(x_{\text{ID}}) < \inf_{x_{\text{OoD}}} d(x_{\text{OoD}}), \quad (21)$$

strictly separating ID and OoD samples.

(The detailed proof is provided in Appendix D.)

Hyperparameter Configuration: To satisfy Eq. (20), we constrain hyperparameters: We set $\lambda \leq 10^{-2}$ with L2 penalty $\beta \lambda^2$ ($\|E\| \approx 0.02 - 0.04$); set whitened_u=True, wrap all linear layers with spectral_norm, and initialize Σ_{NM}

Method	Time complexity	Storage complexity
Deterministic	$\mathcal{O}(Nd_{in}d_{out})$	$\mathcal{O}(d_{in}d_{out})$
BatchEnsemble	$\mathcal{O}(NKd_{in}d_{out})$	$\mathcal{O}(Kd_{in}d_{out})$
FFG-U (Ritter et al. 2021)	$\mathcal{O}(NKd_{in}d_{out} + 2M_{in}^3 + 2M_{out}^3 + K(d_{out}M_{out}M_{in} + M_{in}d_{out}d_{in}))$	$\mathcal{O}(d_{in}M_{in} + d_{out}M_{out} + 2M_{in}M_{out})$
FiD-GP (Reparam)	$\mathcal{O}(NK(d_{in}d_{out} + M_{in}) + 2M_{in}^3 + 2M_{out}^3)$	$\mathcal{O}(d_{in}M_{in} + d_{out}M_{out} + M_{in})$
FiD-GP (Matheron)	$\mathcal{O}(NK(d_{in}d_{out} + M_{in}) + 2M_{in}^3 + 2M_{out}^3 + K(d_{out}M_{out}M_{in} + M_{in}d_{out}d_{in}))$	$\mathcal{O}(d_{in}M_{in} + d_{out}M_{out} + KM_{in}M_{out} + M_{in})$

Table 1: Computational complexity per layer. We assume $\mathbf{W} \in \mathbb{R}^{d_{out} \times d_{in}}$, $\mathbf{U} \in \mathbb{R}^{M_{out} \times M_{in}}$, and K forward passes for each of the N inputs.

orthogonally (approximating $S \approx 0.15$). Thus, empirically,

$$S \approx 0.15 > 2 \times 0.03 = 0.06 \implies S > 2\|E\| \quad (22)$$

Since the OoD distribution is unknown during training, we cannot rigorously prove the bound $S > 2\|E\|$. However, based on our empirical observations, the value of $\|E\|$ consistently falls in the above-mentioned range, and the measured value of S_{OoD} significantly exceeds $2\|E\|$.

5 Experiments

We conducted comprehensive experiments on synthetic 1-D regression; image classification on ImageNet-1k and CIFAR-100, including Out-of-Distribution detection; and semantic segmentation on CamVID and CityScapes, likewise evaluating Out-of-Distribution detection. We compare our results against several state-of-the-art and relevant baselines.

Datasets. We conduct regression experiments on the Synthetic 1-D dataset, classification experiments on image datasets including CIFAR-100 and ImageNet, and semantic segmentation experiments on CamVID and CityScapes. For Out-of-Distribution (OoD) detection, we follow standard evaluation protocols using the following dataset pairs: CIFAR-100 vs. CIFAR-10, CIFAR-10 vs. ImageNet and SVHN, as well as CityScapes vs. CamVID and CamVID vs. CityScapes.

Model Complexity. We analyze per-layer computational and storage complexity under standard assumptions: weight matrix $\mathbf{W} \in \mathbb{R}^{d_{out} \times d_{in}}$, inducing matrix $\mathbf{U} \in \mathbb{R}^{M_{out} \times M_{in}}$, N inputs, and K forward passes (Table 1). The deterministic baseline shows minimal complexity, while BatchEnsemble scales linearly with K . Compared to FFG-U (Ritter et al. 2021), our reparameterisation method reduces time complexity by eliminating K -scaled matrix products. Our Matheron approach maintains similar time complexity to FFG-U but requires additional negligible storage for K low-rank components. Full complexity expressions are provided in Table 1.

5.1 Synthetic 1-D Regression

In synthetic 1-D regression task, we follow (Miyato et al. 2018) and (Ritter et al. 2021), who took 2 input clusters $x_1 \sim \mathcal{U}[0.5, 0.8]$, $x_2 \sim \mathcal{U}[1.2, 1.6]$, and targets $y \sim \mathcal{N}(\cos(4x + 0.8), 0.01)$. The deterministic backbone is a fully connected

network with 3 hidden layers of width 100. Each hidden layer consists of a linear transformation, batch normalisation, and a tanh activation to stabilise bounded outputs.

In Figure 3, we compare two different configurations that differ in the shape of their inducing matrix and the associated hyperparameters. The larger inducing matrix is equipped with higher tolerance (λ and σ) (left side of the figure) and, therefore, outperforms the alternative (right side of the figure) as it has greater posterior expressiveness.

5.2 Classification

We evaluate our proposed method on standard image classification benchmarks: CIFAR-100 and ImageNet-1k. Our goal is to assess not only predictive accuracy, but also the quality of uncertainty estimation under in-distribution (ID) and Out-of-Distribution (OoD) conditions.

Results. On ImageNet-1k, Matheron sampling approach applied to all layers achieves state-of-the-art accuracy (70.19%) while significantly reducing parameters to 5.62M (51.6% compression versus deterministic baseline). For uncertainty estimation, all our variants demonstrate exceptional OoD detection performance, see Figure 4, with Matheron sampling achieving near-perfect AUROC scores of 99.9% on both SVHN and CIFAR-10 OoD benchmarks, substantially outperforming BatchEnsemble (94.3%/89.9%) and FFG-U (83.7%/76.2%). On CIFAR-100, the 4-layer Matheron implementation establishes new benchmarks across multiple metrics: highest accuracy (76.27%), and most efficient parameter usage (5.51M).

5.3 Semantic Segmentation

For the CamVID semantic segmentation task, we evaluated **FiD-GP** using FCN-ResNet50 as the backbone. Our method delivered competitive results (Table 4): the 4 layers Matheron variant achieved an mIoU of 62.9%, which is very close to BatchEnsemble’s state-of-the-art performance of 63.1%. Moreover, all configurations of **FiD-GP** exhibited remarkable robustness against domain shifts, attaining a 99.9% AUROC for the CamVID→CityScapes generalization task. We also conducted experiments on CityScapes, using the HRNet-W48 as backbone. Table 5 shows that our method demonstrates superior generalisation ability. It achieved a high mIoU

Method	Accuracy(%) \uparrow	NLL \downarrow	ECE (%) \downarrow	AUROC SVHN (%) (\uparrow)	AUROC CIFAR-10(%) (\uparrow)	FLOPs	#Parameters(M) \downarrow
Deterministic ⁺	69.68	1.12	5.25	-	-	3.62G	11.6M
BatchEnsemble ⁺	70.01	1.06	3.91	94.3	89.9	7.81G	12.4M
FFG-U (Ritter et al. 2021) ⁺	68.31	0.98	4.17	83.7	76.2	11.4G	6.15M
F-SGVB-LRT (Nguyen et al. 2023)	68.42	2.71	5.91	-	-	-	13.1M
(Reparam, 4 layers)	70.00	1.32	5.91	99.8	98.9	8.08G	8.45M
FiD-GP (Matheron, 4 layers)	70.17	1.13	6.86	99.9	99.9	8.09G	8.48M
(Matheron, all layers)	70.19	1.06	4.81	99.9	99.9	14.7G	5.62M

Table 2: Comparison of **FiD-GP** and competitive techniques on the **ImageNet-1k** dataset (superscripts indicate our reproduced variants⁺, otherwise from original papers). The last three rows show our implementations using two sampling methods—*Reparam* and *Matheron*—applied to four convolutional layer pairs (*Reparam*, 2 pairs⁺, *Matheron*, 2 pairs⁺) and to all layers (*Matheron*, all layers⁺). Note: all methods are based on the **ResNet-18** architecture for fair comparison;

Method	Accuracy(%) \uparrow	NLL \downarrow	ECE (%) \downarrow	AUROC CIFAR-100 \rightarrow SVHN	AUROC CIFAR-100 \rightarrow CIFAR-10	FLOPs	#Parameters(M) \downarrow
Deterministic ⁺	75.61	0.93	4.31	-	-	1.1G	11.2M
BatchEnsemble ⁺	76.01	0.99	3.26	91.1	83.4	4.8G	11.9M
FFG-U (Ritter et al. 2021) ⁺	74.81	0.99	4.31	80.6	75.4	11.8G	5.85M
F-SGVB-LRT (Nguyen et al. 2023)	70.10	1.12	3.62	-	-	-	11.8M
(Reparam, 4 layers)	75.99	1.15	4.72	99.8	98.9	2.4G	8.01M
FiD-GP (Matheron, 4 layers)	76.27	1.08	3.69	99.9	99.9	5.4G	8.01M
(Matheron, all layers)	76.11	0.92	3.60	99.9	99.9	11.8G	5.51M

Table 3: Comparison of **FiD-GP** and competitive techniques on the **CIFAR-100** dataset (superscripts indicate our reproduced variants⁺, otherwise from original papers). Note: all methods are based on the **ResNet-18** architecture for fair comparison.

Method	mIoU(%) \uparrow	NLL \downarrow	MPA (%) \uparrow	ECE(%) \downarrow	AUROC CamVID \rightarrow CityScapes	FLOPs	#Parameters(M)
Deterministic ⁺	62.2	0.57	77.3	8.6	-	115.6G	35.3M
BatchEnsemble ⁺	63.1	0.36	80.1	5.2	84.4	126.2G	42.2M
FFG-U ⁺	60.1	0.37	77.1	8.3	80.9	149.1G	15.8M
FiD-GP (Reparam, 4 layers)	62.4	0.40	78.2	7.5	99.9	116.8G	32.7M
FiD-GP (Matheron, 4 layers)	62.8	0.31	78.9	7.4	99.9	119.5G	32.7M
FiD-GP (Matheron, all layers)	62.6	0.48	79.2	7.1	99.9	149.2G	16.2M

Table 4: Comparison of **FiD-GP** and competitive techniques on **CamVID**. Note: all methods are based on the **FCN-ResNet50** architecture for fair comparison.

of 80.9% and an AUROC of 99.9% for the CityScapes to CamVID domain shift task, along with competitive NLL (0.11) and ECE (1.2%). In both experiments, we not only presented the predicted segmentation results but also visualised the associated uncertainty distributions. These distributions clearly indicate increased uncertainty along object boundaries, for CamVID (Figure 5) and CityScapes (Figure 6).

5.4 Ablation Experiments

The shape of the inducing matrix in **FiD-GP** determines a trade-off between model compression and accuracy. To quan-

tify this effect, we conduct ablation studies varying the inducing matrix size on CIFAR-100 using ResNet-18. As shown in Table 6, larger matrices generally achieve higher accuracy at the cost of fewer parameter savings. The 256×256 configuration (256×256) achieves the highest accuracy (77.48%) without compression, while the smallest (32×32) achieves maximum compression (87.9%) with reduced accuracy (69.83%). The 128×128 setting provides a favourable balance, which we adopted throughout our experiments. We always set the linear layer to $128 \times \text{num_class}$.

Method	mIoU(%) \uparrow	NLL \downarrow	MPA (%) \uparrow	ECE(%) \downarrow	AUROC		FLOPs	#Parameters(M)
					CityScapes \rightarrow	CamVID		
Deterministic ⁺	81.0	0.18	96.4	1.9	-	-	373.9G	65.8M
BatchEnsemble ⁺	81.5	0.11	97.1	1.2	88.6	-	394.8G	70.2M
FFG-U ⁺	78.1	0.21	91.7	3.2	71.1	-	459.7G	55.7M
FiD-GP (Reparam, 4 layers)	80.4	0.15	96.1	2.6	99.9	-	374.3G	65.9M
FiD-GP (Matheron, 4 layers)	80.9	0.11	96.5	1.2	99.9	-	374.9G	65.9M
FiD-GP (Matheron, all layers)	80.7	0.18	95.2	1.8	99.9	-	460.3G	58.0M

Table 5: Evaluation of **FiD-GP** against other state-of-the-art methods on **CityScapes**, with all approaches employing the **HRNet-W48** backbone for a fair comparison; Note: we applied a pre-trained model here to reduce the computational cost.

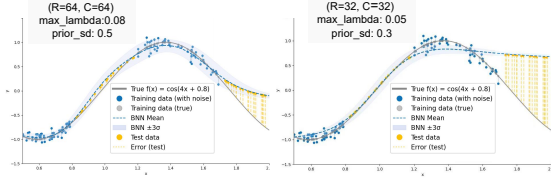


Figure 3: Synthetic 1-D regression: true $f(x) = \cos(4x+0.8)$ (black); noisy training (blue) with error bars; test (orange) with error lines; **FiD-GP** mean (dashed blue) and $\pm 3\sigma$ interval (shaded). **Left**: inducing grid $R \times C = 64 \times 64$ (rows \times columns), $\lambda_{max} = 0.08$, $prior_sd = 0.5$. **Right**: $R \times C = 32 \times 32$, $\lambda_{max} = 0.05$, $prior_sd = 0.3$.

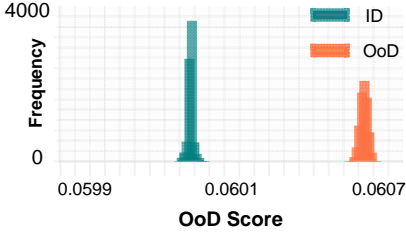


Figure 4: Distributions of predictive scores from ResNet-18 + **FiD-GP** (Reparam, 4 layers) on CIFAR-100 (ID) and CIFAR-10 (OoD).

Table 6: CIFAR-100 results for ResNet-18 modified with **FiD-GP**: accuracy, parameter count, and compression rate relative to the 11.2 M-parameter deterministic model, over various inducing-matrix sizes.

Type/Size	Accuracy(%) \uparrow	NLL \downarrow	ECE (%) \downarrow	Parameters Compression \uparrow	
Conv	32 \times 32	69.83	1.16	4.46	1.35M / 87.9%
	64 \times 64	73.26	1.14	3.88	2.66M / 76.2%
	128 \times 128	76.11	0.92	3.60	5.51M / 50.8%
	256 \times 256	77.48	1.00	4.55	12.1M / -
Linear setting		128 \times num_class			

6 Discussion

Layer Selection: In our experiments, we selected 2 pairs (4 layers) of consecutive convolution layers for replacement

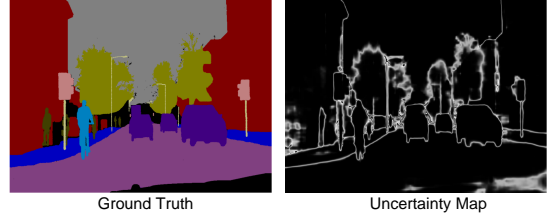


Figure 5: CamVID Ground Truth (left) and Uncertainty Map (right) generated by FCN-ResNet50 (Matheron, 4 layers).

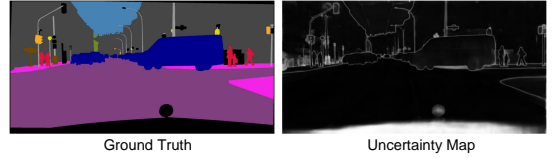


Figure 6: **CityScapes** Ground Truth (left) and Uncertainty Map (right) generated by HRNet-W48 (Matheron, 4 layers).

with **FiD-GP**. Furthermore, the second layer in each pair will be used to compute the predictive score to distinguish ID from OoD data, as the output of the first layer satisfies the 1-Lipschitz condition (see Appendix C for details). While these 2 pairs can be chosen randomly, selecting one layer from shallow layers and another one from deep layers can reduce the computational cost and increase the separation distance between ID and OoD. For example, in ResNet-18, we used the layers ["layer2.1.conv1", "layer2.1.conv2", "layer4.1.conv1", "layer4.1.conv2"], where the first two layers form the first pair and the last two layers form the second pair.

7 Conclusion

We have proposed a GP-inspired uncertainty approach, named **FiD-GP**, which can seamlessly be integrated into a deep neural network. **FiD-GP** incorporates a compact inducing weight matrix to project neural network weights' uncertainty into a lower-dimensional subspace, with expressiveness augmented through a normalizing-flow prior and spectral regularization, which aligns the inducing subspace with feature-gradient geometry through a numerically stable projection mechanism objective, and produces a single-pass projection for Out-of-Distribution (OoD) detection.

References

- An, J.; and Cho, S. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, 2(1): 1–18.
- Arhonditsis, G.; Kim, D.-K.; Kelly, N.; Neumann, A.; and Javed, A. 2017. Uncertainty analysis by Bayesian inference. In *Ecological Informatics: Data Management and Knowledge Discovery*, 215–249. Springer.
- Blasco, T.; Sánchez, J. S.; and García, V. 2024. A survey on uncertainty quantification in deep learning for financial time series prediction. *Neurocomputing*, 576: 127339.
- Blei, D. M.; Kucukelbir, A.; and McAuliffe, J. D. 2017. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518): 859–877.
- Brostow, G. J.; Fauqueur, J.; and Cipolla, R. 2009. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2): 88–97.
- Chen, Y.; Sung, C.-L.; Kusari, A.; Song, X.; and Sun, W. 2024. Uncertainty-aware out-of-distribution detection with gaussian processes. *arXiv preprint arXiv:2412.20918*.
- Cheng, C.-A.; and Huang, H.-P. 2015. Learn the Lagrangian: A vector-valued RKHS approach to identifying Lagrangian systems. *IEEE Transactions on Cybernetics*, 46(12): 3247–3258.
- Chua, M.; Kim, D.; Choi, J.; Lee, N. G.; Deshpande, V.; Schwab, J.; Lev, M. H.; Gonzalez, R. G.; Gee, M. S.; and Do, S. 2023. Tackling prediction uncertainty in machine learning for healthcare. *Nature Biomedical Engineering*, 7(6): 711–718.
- Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; and Schiele, B. 2016. The cityscapes dataset for semantic urban scene understanding. In *Conference on Computer Vision and Pattern Recognition*, 3213–3223.
- Cutajar, K.; Osborne, M.; Cunningham, J.; and Filippone, M. 2016. Preconditioning kernel matrices. In *International Conference on Machine Learning*, 2529–2538. PMLR.
- Damianou, A.; and Lawrence, N. D. 2013. Deep gaussian processes. In *Artificial Intelligence and Statistics*, 207–215. PMLR.
- Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 248–255. IEEE.
- Dinh, L.; Sohl-Dickstein, J.; and Bengio, S. 2017. Density estimation using Real NVP. In *International Conference on Learning Representations*.
- Dusenberry, M.; Jerfel, G.; Wen, Y.; Ma, Y.; Snoek, J.; Heller, K.; Lakshminarayanan, B.; and Tran, D. 2020. Efficient and scalable bayesian neural nets with rank-1 factors. In *International Conference on Machine Learning*, 2782–2792. PMLR.
- Franchi, G.; Bursuc, A.; Aldea, E.; Dubuisson, S.; and Bloch, I. 2023. Encoding the latent posterior of bayesian neural networks for uncertainty quantification. *IEEE TPAMI*, 46(4): 2027–2040.
- Gong, D.; Liu, L.; Le, V.; Saha, B.; Mansour, M. R.; Venkatesh, S.; and Hengel, A. v. d. 2019. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Conference on Computer Vision and Pattern Recognition*, 1705–1714.
- Grathwohl, W.; Chen, R. T.; Bettencourt, J.; Sutskever, I.; and Duvenaud, D. 2018. FFJORD: Free-Form Continuous Dynamics for Scalable Reversible Generative Models. In *International Conference on Learning Representations*.
- He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 770–778.
- Hensman, J.; Matthews, A. G.; Filippone, M.; and Ghahramani, Z. 2015. MCMC for variationally sparse Gaussian processes. *Advances in Neural Information Processing Systems*, 8: 1648–1656.
- Hernández-Lobato, J. M.; and Adams, R. 2015. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, 1861–1869. PMLR.
- Hida, T.; and Hitsuda, M. 1993. *Gaussian Processes*. American Mathematical Society.
- Hoffmann, L.; Fortmeier, I.; and Elster, C. 2021. Uncertainty quantification by ensemble learning for computational optical form measurements. *Machine Learning: Science and Technology*, 2(3): 035030.
- Hubin, A.; and Storvik, G. 2024. Sparse Bayesian neural networks: bridging model and parameter uncertainty through scalable variational inference. *Mathematics*, 12(6): 788.
- Hubmann, C.; Becker, M.; Althoff, D.; Lenz, D.; and Stiller, C. 2017. Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles. In *IEEE Intelligent Vehicles Symposium*, 1671–1678. IEEE.
- Kingma, D. P.; and Dhariwal, P. 2018. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31: 10215–10224.
- Kingma, D. P.; Salimans, T.; Jozefowicz, R.; Chen, X.; Sutskever, I.; and Welling, M. 2016. Improved variational inference with inverse autoregressive flow. *Advances in Neural Information Processing Systems*, 29: 4743–4751.
- Kononenko, I. 1989. Bayesian neural networks. *Biological Cybernetics*, 61(5): 361–370.
- Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images. Technical report, University of Toronto.
- Kuss, M.; and Rasmussen, C. 2005. Assessing approximations for Gaussian process classification. *Advances in Neural Information Processing Systems*, 18: 699–706.
- Lawrence, N.; Seeger, M.; and Herbrich, R. 2002. Fast sparse Gaussian process methods: The informative vector machine. *Advances in Neural Information Processing Systems*, 15: 625–632.
- Leibfried, F.; Dutordoir, V.; John, S.; and Durrande, N. 2020. A tutorial on sparse Gaussian processes and variational inference. *arXiv preprint arXiv:2012.13962*.

- Lin, J.-L.; Krishnan, R.; Ranipa, K. R.; Subedar, M.; Sanghavi, V.; Arunachalam, M.; Tickoo, O.; Iyer, R.; and Kandemir, M. T. 2023. Quantization for bayesian deep learning: Low-precision characterization and robustness. In *IEEE International Symposium on Workload Characterization*, 180–192. IEEE.
- Lin, M.; Guan, S.; Jing, W.; Botterweck, G.; and Patane, A. 2025. Stochastic Weight Sharing for Bayesian Neural Networks. In *International Conference on Artificial Intelligence and Statistics*, 4519–4527. PMLR.
- Liu, J.; Lin, Z.; Padhy, S.; Tran, D.; Bedrax Weiss, T.; and Lakshminarayanan, B. 2020. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in Neural Information Processing Systems*, 33: 7498–7512.
- Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3431–3440.
- MacKay, D. J. 1995. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 354(1): 73–80.
- Mishkin, A.; Kunstner, F.; Nielsen, D.; Schmidt, M.; and Khan, M. E. 2018. Slang: Fast structured covariance approximations for bayesian deep learning with natural gradient. *Advances in Neural Information Processing Systems*, 31: 6245–6255.
- Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral Normalization for Generative Adversarial Networks. In *International Conference on Learning Representations*.
- Mukhoti, J.; Kirsch, A.; Van Amersfoort, J.; Torr, P. H.; and Gal, Y. 2023. Deep deterministic uncertainty: A new simple baseline. In *Conference on Computer Vision and Pattern Recognition*, 24384–24394.
- Nalisnick, E. T.; Matsukawa, A.; Teh, Y. W.; Görür, D.; and Lakshminarayanan, B. 2019. Do Deep Generative Models Know What They Don’t Know? In *International Conference on Learning Representations*.
- Nguyen, V.-A.; Vuong, T.-L.; Phan, H.; Do, T.-T.; Phung, D.; and Le, T. 2023. Flat seeking bayesian neural networks. *Advances in Neural Information Processing Systems*, 36: 30807–30820.
- Opfer, M.; and Winther, O. 2000. Gaussian processes for classification: Mean-field algorithms. *Neural Computation*, 12(11): 2655–2684.
- Papamakarios, G.; Pavlakou, T.; and Murray, I. 2017. Masked autoregressive flow for density estimation. *Advances in Neural Information Processing Systems*, 30: 2338–2347.
- Rahaman, R.; et al. 2021. Uncertainty quantification and deep ensembles. *Advances in Neural Information Processing Systems*, 34: 20063–20075.
- Ren, J.; Liu, P. J.; Fertig, E.; Snoek, J.; Poplin, R.; Depristo, M.; Dillon, J.; and Lakshminarayanan, B. 2019. Likelihood ratios for out-of-distribution detection. *Advances in Neural Information Processing Systems*, 32: 14707–14718.
- Rezende, D.; and Mohamed, S. 2015. Variational inference with normalizing flows. In *International Conference on Machine Learning*, 1530–1538. PMLR.
- Ritter, H.; Botev, A.; and Barber, D. 2018. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*. International Conference on Representation Learning.
- Ritter, H.; Kukla, M.; Zhang, C.; and Li, Y. 2021. Sparse uncertainty representation in deep learning with inducing weights. *Advances in Neural Information Processing Systems*, 34: 6515–6528.
- Seeger, M. 2004. Gaussian processes for machine learning. *International Journal of Neural Systems*, 14(02): 69–106.
- Snelson, E.; and Ghahramani, Z. 2005. Sparse Gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems*, 18: 1257–1264.
- Snelson, E.; and Ghahramani, Z. 2007. Local and global sparse Gaussian process approximations. In *Artificial Intelligence and Statistics*, 524–531. PMLR.
- Sun, K.; Zhao, Y.; Jiang, B.; Cheng, T.; Xiao, B.; Liu, D.; Mu, Y.; Wang, X.; Liu, W.; and Wang, J. 2019. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*.
- Swiler, L. P.; Gulian, M.; Frankel, A. L.; Safta, C.; and Jake-man, J. D. 2020. A survey of constrained Gaussian process regression: Approaches and implementation challenges. *Journal of Machine Learning for Modeling and Computing*, 1(2): 119–156.
- Thodberg, H. H. 1996. A review of Bayesian neural networks with an application to near infrared spectroscopy. *IEEE transactions on Neural Networks*, 7(1): 56–72.
- Titsias, M. 2009. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, 567–574. PMLR.
- Van Amersfoort, J.; Smith, L.; Teh, Y. W.; and Gal, Y. 2020. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, 9690–9700. PMLR.
- Van der Wilk, M.; Rasmussen, C. E.; and Hensman, J. 2017. Convolutional gaussian processes. *Advances in Neural Information Processing Systems*, 30: 2849–2858.
- Wei, Y.; Zhuang, V.; Soedarmadji, S.; and Sui, Y. 2024. Scalable Bayesian optimization via focalized sparse Gaussian processes. *Advances in Neural Information Processing Systems*, 37: 120443–120467.
- Williams, C.; and Rasmussen, C. 1995. Gaussian processes for regression. *Advances in Neural Information Processing Systems*, 8: 514–520.
- Zhang, J.; Das, K.; and Kumar, S. 2024. Discriminant Distance-Aware Representation on Deterministic Uncertainty Quantification Methods. In *International Conference on Artificial Intelligence and Statistics*, 2917–2925. PMLR.
- Zyphur, M. J.; and Oswald, F. L. 2015. Bayesian estimation and inference: A user’s guide. *Journal of Management*, 41(2): 390–420.

Reproducibility Checklist

1. General Paper Structure

- 1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) **yes**
- 1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) **yes**
- 1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) **yes**

2. Theoretical Contributions

- 2.1. Does this paper make theoretical contributions? (yes/no) **yes**

If yes, please address the following points:

- 2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) **yes**
- 2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) **yes**
- 2.4. Proofs of all novel claims are included (yes/partial/no) **partial**
- 2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) **yes**
- 2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) **yes**
- 2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) **yes**
- 2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) **yes**

3. Dataset Usage

- 3.1. Does this paper rely on one or more datasets? (yes/no) **yes**

If yes, please address the following points:

- 3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) **yes**
- 3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) **yes**
- 3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) **yes**

- 3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) **yes**

- 3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) **yes**

- 3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisfying (yes/partial/no/NA) **yes**

4. Computational Experiments

- 4.1. Does this paper include computational experiments? (yes/no) **yes**

If yes, please address the following points:

- 4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) **yes**
- 4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) **yes**
- 4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) **yes**
- 4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) **yes**
- 4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) **yes**
- 4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) **yes**
- 4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) **yes**
- 4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) **yes**
- 4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) **yes**
- 4.11. Analysis of experiments goes beyond single-

dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) [yes](#)

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) [no](#)

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) [yes](#)

Appendix A: Derivation of Conditional Gaussian via Completing the Square

This appendix presents a rigorous derivation of the conditional Gaussian distribution using the method of completing the square in the exponent. We emphasize the geometric interpretation of covariance matrices and demonstrate how the Schur complement naturally emerges during the process. This approach reveals the deep connection between joint and conditional distributions in Gaussian systems.

Preliminary Setup Let $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{n+m}$ be jointly Gaussian random vectors with partitioned moments:

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_x^{(n \times 1)} \\ \boldsymbol{\mu}_y^{(m \times 1)} \end{bmatrix}, \quad (23)$$

$$\Sigma = \begin{bmatrix} \Sigma_{xx}^{(n \times n)} & \Sigma_{xy}^{(n \times m)} \\ \Sigma_{yx}^{(m \times n)} & \Sigma_{yy}^{(m \times m)} \end{bmatrix}$$

where superscripts in parentheses indicate matrix dimensions. The joint density is:

$$p(\mathbf{x}, \mathbf{y}) = (2\pi)^{-\frac{n+m}{2}} |\Sigma|^{-\frac{1}{2}} \times \exp \left(-\frac{1}{2} \begin{bmatrix} \mathbf{x} - \boldsymbol{\mu}_x \\ \mathbf{y} - \boldsymbol{\mu}_y \end{bmatrix}^\top \Sigma^{-1} \begin{bmatrix} \mathbf{x} - \boldsymbol{\mu}_x \\ \mathbf{y} - \boldsymbol{\mu}_y \end{bmatrix} \right) \quad (24)$$

Inverse Covariance Structure The key insight comes from the block matrix inversion formula:

$$\Sigma^{-1} = \begin{bmatrix} A & B \\ B^\top & D \end{bmatrix} = \begin{bmatrix} (\Sigma/\Sigma_{yy})^{-1} & -(\Sigma/\Sigma_{yy})^{-1} \Sigma_{xy} \Sigma_{yy}^{-1} \\ -\Sigma_{yy}^{-1} \Sigma_{yx} (\Sigma/\Sigma_{yy})^{-1} & (\Sigma_{yy} - \Sigma_{yx} \Sigma_{xx}^{-1} \Sigma_{xy})^{-1} \end{bmatrix} \quad (25)$$

where $\Sigma/\Sigma_{yy} := \Sigma_{xx} - \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{yx}$ denotes the Schur complement. This reveals the fundamental relationship:

$$A^{-1} = \Sigma_{xx} - \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{yx} \quad (26)$$

Expanding the Mahalanobis distance in the exponent:

$$\begin{aligned} \mathcal{Q} &:= \begin{bmatrix} \mathbf{x} - \boldsymbol{\mu}_x \\ \mathbf{y} - \boldsymbol{\mu}_y \end{bmatrix}^\top \begin{bmatrix} A & B \\ B^\top & D \end{bmatrix} \begin{bmatrix} \mathbf{x} - \boldsymbol{\mu}_x \\ \mathbf{y} - \boldsymbol{\mu}_y \end{bmatrix} \\ &= (\mathbf{x} - \boldsymbol{\mu}_x)^\top A (\mathbf{x} - \boldsymbol{\mu}_x) \\ &\quad + 2(\mathbf{x} - \boldsymbol{\mu}_x)^\top B (\mathbf{y} - \boldsymbol{\mu}_y) \\ &\quad + (\mathbf{y} - \boldsymbol{\mu}_y)^\top D (\mathbf{y} - \boldsymbol{\mu}_y) \end{aligned} \quad (27)$$

Conditional Density Derivation

$$p(\mathbf{x}|\mathbf{y}) \propto \exp \left(-\frac{1}{2} [(\mathbf{x} - \boldsymbol{\mu}_x)^\top A (\mathbf{x} - \boldsymbol{\mu}_x) + 2(\mathbf{x} - \boldsymbol{\mu}_x)^\top B (\mathbf{y} - \boldsymbol{\mu}_y)] \right) \quad (28)$$

To complete the square, we seek $\mathbf{m} \in \mathbb{R}^n$ and $\Phi \in \mathbb{R}^{n \times n}$ such that:

$$\begin{aligned} (\mathbf{x} - \mathbf{m})^\top \Phi (\mathbf{x} - \mathbf{m}) &= (\mathbf{x} - \boldsymbol{\mu}_x)^\top A (\mathbf{x} - \boldsymbol{\mu}_x) \\ &\quad + 2(\mathbf{x} - \boldsymbol{\mu}_x)^\top B (\mathbf{y} - \boldsymbol{\mu}_y) \end{aligned} \quad (29)$$

Matching coefficients yields:

$$\Phi = A \quad (30a)$$

$$\Phi(\mathbf{m} - \boldsymbol{\mu}_x) = -B(\mathbf{y} - \boldsymbol{\mu}_y) \quad (30b)$$

Solving this system gives the conditional parameters:

$$\mathbf{m} = \boldsymbol{\mu}_x - A^{-1} B (\mathbf{y} - \boldsymbol{\mu}_y), \quad \Phi = A \quad (31)$$

Canonical Form Conversion Using the matrix inversion lemma:

$$\begin{aligned} A^{-1} &= \Sigma_{xx} - \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{yx}, \\ A^{-1} B &= \Sigma_{xy} \Sigma_{yy}^{-1} \end{aligned} \quad (32)$$

Substituting these into the conditional parameters produces the standard form:

$$p(\mathbf{x} | \mathbf{y}) = \mathcal{N}(\mathbf{x} | \underbrace{\boldsymbol{\mu}_x + \Sigma_{xy} \Sigma_{yy}^{-1} (\mathbf{y} - \boldsymbol{\mu}_y)}_{\text{Conditional Mean}}, \underbrace{\Sigma_{xx} - \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{yx}}_{\text{Conditional Covariance}}) \quad (33)$$

$$\begin{aligned} p(\mathbf{x} | \mathbf{y}) &= \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{x|y}, \Sigma_{x|y}), \\ \boldsymbol{\mu}_{x|y} &= \boldsymbol{\mu}_x + \Sigma_{xy} \Sigma_{yy}^{-1} (\mathbf{y} - \boldsymbol{\mu}_y), \\ \Sigma_{x|y} &= \Sigma_{xx} - \Sigma_{xy} \Sigma_{yy}^{-1} \Sigma_{yx}. \end{aligned} \quad (34)$$

Appendix B: Derivation of Conditional Mean via Kronecker Factorization under Matrix Normal Distributions

In this section, we provide a detailed derivation of the conditional mean formula for jointly Gaussian-distributed matrix-valued random variables. Suppose we have random matrices $W \in \mathbb{R}^{m \times n}$ and $U \in \mathbb{R}^{p \times q}$, jointly Gaussian distributed with

$$\begin{aligned} W &\sim \mathcal{MN}(\mathbf{0}, \Sigma_{\text{row}}, \Sigma_{\text{col}}), \\ U &\sim \mathcal{MN}(\mathbf{0}, \Sigma_U^{(\text{row})}, \Sigma_U^{(\text{col})}) \end{aligned} \quad (35)$$

together with the cross-covariance

$$\text{Cov}(W, U) = \Sigma_{W,U} \quad (36)$$

Given the joint Gaussian distribution, the conditional expectation of W given U is expressed as

$$\mathbb{E}[W | U] = \Sigma_{W,U} \Sigma_U^{-1} U \quad (37)$$

where $\Sigma_{W,U}$ denotes the cross-covariance between the matrices W and U , and Σ_U is the covariance of U .

By vectorizing matrices in a column-wise fashion, we define $\text{vec}(W) \in \mathbb{R}^{mn \times 1}$ and $\text{vec}(U) \in \mathbb{R}^{pq \times 1}$. Consequently, the conditional expectation becomes

$$\begin{aligned} \mathbb{E}[\text{vec}(W) | U] &= K_{W,U} K_U^{-1} \text{vec}(U), \\ K_{W,U} &\in \mathbb{R}^{mn \times pq}, \quad K_U \in \mathbb{R}^{pq \times pq} \end{aligned} \quad (38)$$

This formulation clearly illustrates the fundamental linear mapping $K_{W,U} K_U^{-1}$ characteristic of conditional Gaussian distributions.

Kronecker-Structured Covariance Decomposition Further assuming a Kronecker separable covariance structure, we have

$$\begin{aligned} K_W &= \Sigma_{\text{col}} \otimes \Sigma_{\text{row}}, \\ K_U &= \Sigma_U^{(\text{col})} \otimes \Sigma_U^{(\text{row})}, \\ K_{W,U} &= K_{W,U}^{(\text{col})} \otimes K_{W,U}^{(\text{row})} \end{aligned} \quad (39)$$

Using properties of the Kronecker product, specifically

$$\begin{aligned} (A \otimes B)(C \otimes D) &= (AC) \otimes (BD), \\ (A \otimes B)^{-1} &= A^{-1} \otimes B^{-1} \end{aligned} \quad (40)$$

we obtain

$$K_{W,U} K_U^{-1} = (K_{W,U}^{(\text{col})} (\Sigma_U^{(\text{col})})^{-1}) \otimes (K_{W,U}^{(\text{row})} (\Sigma_U^{(\text{row})})^{-1}) \quad (41)$$

Thus, the conditional expectation in vectorized form is

$$\begin{aligned} \text{vec}(\mathbb{E}[W | U]) &= (K_{W,U}^{(\text{col})} (\Sigma_U^{(\text{col})})^{-1}) \otimes \\ &\quad (K_{W,U}^{(\text{row})} (\Sigma_U^{(\text{row})})^{-1}) \text{vec}(U) \end{aligned} \quad (42)$$

We leverage the vectorization identity

$$\text{vec}(AXB^\top) = (B \otimes A) \text{vec}(X) \quad (43)$$

To reconstruct the conditional expectation back to matrix form. Let us define the transformations

$$\begin{aligned} T_{\text{row}} &= K_{W,U}^{(\text{row})} (\Sigma_U^{(\text{row})})^{-1}, \\ T_{\text{col}} &= K_{W,U}^{(\text{col})} (\Sigma_U^{(\text{col})})^{-1} \end{aligned} \quad (44)$$

Then, the conditional expectation of W given U can be succinctly written in matrix form as

$$\mathbb{E}[W | U] = T_{\text{row}} U T_{\text{col}}^\top \quad (45)$$

To avoid direct computation of inverses, numerical implementations typically utilize the Cholesky decomposition of covariance matrices. Specifically, for covariance matrices:

$$\begin{aligned} \Sigma_U^{(\text{row})} &= L_{\text{row}} L_{\text{row}}^\top, \\ \Sigma_U^{(\text{col})} &= L_{\text{col}} L_{\text{col}}^\top \end{aligned} \quad (46)$$

the transformations can be computed efficiently by solving linear systems involving these Cholesky factors, thus improving numerical stability and computational efficiency.

Appendix C: Derivation of the Flow-based KL Divergence Term

Let $u_0 \in \mathbb{R}^{RC}$ be a *base* latent with density $q_0(u_0)$, and let $g : \mathbb{R}^{RC} \rightarrow \mathbb{R}^{RC}$ be a bijective, differentiable transformation with Jacobian $J_g(u_0) = \partial g(u_0) / \partial u_0$. Define $u = g(u_0)$ and the *pushforward* density $q(u) = g_\# q_0$. A prior density $p(u)$ is specified on the same space as u . We wish to compute

$$\text{KL}[q(u) \| p(u)] = \int q(u) [\log q(u) - \log p(u)] du \quad (47)$$

Because g is bijective we may write $u_0 = g^{-1}(u)$ and apply the change-of-variables formula:

$$\begin{aligned} q(u) &= q_0(g^{-1}(u)) \left| \det J_{g^{-1}}(u) \right| \\ &= q_0(u_0) \left| \det J_g(u_0) \right|^{-1} \end{aligned} \quad (48)$$

Taking logs gives

$$\log q(u) = \log q_0(u_0) - \log |\det J_g(u_0)| \quad (49)$$

Substitute this expression for $\log q(u)$ into the KL definition and simultaneously change integration variables from u to u_0 (with $du = |\det J_g(u_0)| du_0$, which cancels the reciprocal Jacobian factor already present in $q(u)$):

$$\begin{aligned} \text{KL}[q(u) \| p(u)] &= \int q_0(u_0) \left[\log q_0(u_0) - \right. \\ &\quad \left. \log |\det J_g(u_0)| - \log p(g(u_0)) \right] du_0 \end{aligned} \quad (50)$$

Recognizing the integral as an expectation over $u_0 \sim q_0$ yields the desired identity:

$$\begin{aligned} \text{KL}[q(u) \| p(u)] &= \mathbb{E}_{u_0 \sim q_0} \left[\log q_0(u_0) - \right. \\ &\quad \left. \log |\det J_g(u_0)| - \log p(g(u_0)) \right] \end{aligned} \quad (51)$$

Monte Carlo estimator. Drawing a reparameterized sample $u_0 \sim q_0$ and computing $u = g(u_0)$ and $\log |\det J_g(u_0)|$ from the flow, a single-sample stochastic estimator of the KL term in (51) is

$$\begin{aligned} \widehat{\text{KL}} &= \log q_0(u_0) - \log |\det J_g(u_0)| \\ &\quad - \log p(g(u_0)) \end{aligned} \quad (52)$$

optionally averaged over multiple samples to reduce variance.

Adding a conditional Gaussian factor (weights). Suppose model weights $W \in \mathbb{R}^D$ (flattened) are conditionally Gaussian given u :

$$W | u \sim \mathcal{N}(\mu(u), (\lambda \sigma_p)^2 I), \quad (53)$$

with Gaussian prior

$$W \sim \mathcal{N}(\mu(u), \sigma_p^2 I), \quad (54)$$

—note the same mean (the prior mean may be zero in practice; if means differ, include the usual quadratic term). Then the per- u conditional KL is the sum over D independent dimensions of the 1D Gaussian KL:

$$\text{KL}[\mathcal{N}(\mu, (\lambda \sigma_p)^2) \| \mathcal{N}(\mu, \sigma_p^2)] = \frac{1}{2} (\lambda^2 - 1 - 2 \log \lambda) \quad (55)$$

hence

$$\text{KL}[q(W | u) \| p(W | u)] = \frac{D}{2} (\lambda^2 - 1 - 2 \log \lambda) \quad (56)$$

Combining (51) and (56) yields the total variational KL used in the ELBO of the inducing-parameter model described in the main text.

Appendix D: Proof of Spectral Residual Separation

Lemma (Spectral Residual Separation) Let the weight matrix be decomposed as

$$w = T_{\text{row}} U T_{\text{col}}^\top + E \quad (57)$$

Define the separation margin and residual respectively as

$$S = \inf_{\|x\|=1} \|(I - P) g(T_{\text{row}} U T_{\text{col}}^\top x)\| \quad (58)$$

$$d(x) = \|(I - P) g((T_{\text{row}} U T_{\text{col}}^\top + E)x)\| \quad (59)$$

where the projection P is given by

$$P = U^\top (UU^\top + \lambda I)^{-1} U \quad (60)$$

with $\|I - P\|_\sigma = 1$, and $g : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is 1-Lipschitz.

For in-distribution (ID) samples, we have empirically:

$$\sup_{x_{\text{ID}}} d(x_{\text{ID}}) < \|E\| \quad (61)$$

If during training, we achieve (empirically observed) for Out-of-Distribution (OoD) samples:

$$S_{\text{OoD}} > 2 \|E\|, \quad (62)$$

then the following strict separation holds:

$$\sup_{x_{\text{ID}}} d(x_{\text{ID}}) < \inf_{x_{\text{OoD}}} d(x_{\text{OoD}}) \quad (63)$$

Proof. We divided the proof into 4 steps:

Step 1: Spectral norm of $I - P$ when U is square but rank-deficient (ignoring λI)

Suppose $U \in \mathbb{R}^{N \times N}$ is a square matrix with rank $r < N$, so it is not invertible. We consider the projection matrix defined by

$$P = U^\top (UU^\top)^\dagger U, \quad (64)$$

where $(\cdot)^\dagger$ denotes the Moore–Penrose pseudoinverse.

Using the singular value decomposition $U = V\Sigma W^\top$, with

$$\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r, 0, \dots, 0), \quad \text{where } \sigma_1 \geq \dots \geq \sigma_r > 0 \quad (65)$$

we can express P as

$$P = W \text{diag}(\underbrace{1, \dots, 1}_r, \underbrace{0, \dots, 0}_{N-r}) W^\top \quad (66)$$

Therefore,

$$I - P = W \text{diag}(\underbrace{0, \dots, 0}_r, \underbrace{1, \dots, 1}_{N-r}) W^\top \quad (67)$$

This shows that $I - P$ has eigenvalue 1 in the null space of U (which is of dimension $N - r > 0$) and eigenvalue 0 in the row space of U . Since $I - P$ is symmetric, its spectral norm equals its largest eigenvalue magnitude:

$$\|I - P\|_\sigma = \max_{\lambda \in \text{spec}(I - P)} |\lambda| = 1. \quad (68)$$

Remark: If U were full-rank and invertible, then

$$UU^\top = UU^\top = UU^\top \Rightarrow (UU^\top)^{-1} = U^{-\top} U^{-1} \quad (69)$$

so

$$P = U^\top (UU^\top)^{-1} U = U^\top U^{-\top} U^{-1} U = I \quad (70)$$

and thus

$$\|I - P\|_\sigma = \|0\|_\sigma = 0 \quad (71)$$

Therefore, $\|I - P\|_\sigma = 1$ if and only if U is not full rank.

Step 2: Upper bound for ID residuals: For any in-distribution sample x_{ID} , by construction:

$$(I - P) g(T_{\text{row}} U T_{\text{col}}^\top x_{\text{ID}}) = 0 \quad (72)$$

Thus,

$$\begin{aligned} d(x_{\text{ID}}) &= \|(I - P) [g((T_{\text{row}} U T_{\text{col}}^\top + E)x_{\text{ID}}) \\ &\quad - g(T_{\text{row}} U T_{\text{col}}^\top x_{\text{ID}})]\| \\ &\leq \|I - P\|_\sigma \|g((T_{\text{row}} U T_{\text{col}}^\top + E)x_{\text{ID}}) \\ &\quad - g(T_{\text{row}} U T_{\text{col}}^\top x_{\text{ID}})\| \\ &\leq \|E x_{\text{ID}}\| \quad (g \text{ is 1-Lipschitz}) \\ &\leq \|E\| \end{aligned} \quad (73)$$

Therefore,

$$\sup_{x_{\text{ID}}} d(x_{\text{ID}}) \leq \|E\| \quad (74)$$

Explanation of critical steps: The penultimate inequality holds because g being 1-Lipschitz implies:

$$\|g(\mathbf{a}) - g(\mathbf{b})\| \leq \|\mathbf{a} - \mathbf{b}\| \quad \forall \mathbf{a}, \mathbf{b} \quad (75)$$

Here we set $\mathbf{a} = (T_{\text{row}} U T_{\text{col}}^\top + E)x_{\text{ID}}$ and $\mathbf{b} = T_{\text{row}} U T_{\text{col}}^\top x_{\text{ID}}$, yielding:

$$\|\mathbf{a} - \mathbf{b}\| = \|E x_{\text{ID}}\| \quad (76)$$

The final step uses the Cauchy-Schwarz inequality $\|E x_{\text{ID}}\| \leq \|E\| \cdot \|x_{\text{ID}}\|$ combined with the unit norm constraint $\|x_{\text{ID}}\| = 1$ from the infimum definition in Lemma 1.

Therefore,

$$\sup_{x_{\text{ID}}} d(x_{\text{ID}}) \leq \|E\| \quad (77)$$

Step 3: Lower bound for OoD residuals: For any Out-of-Distribution vector x_{OoD} , by reverse triangle inequality,

$$\begin{aligned} d(x_{\text{OoD}}) &= \|(I - P) g((T_{\text{row}} U T_{\text{col}}^\top + E)x_{\text{OoD}})\| \\ &\geq \|(I - P) g(T_{\text{row}} U T_{\text{col}}^\top x_{\text{OoD}})\| \\ &\quad - \|(I - P) [g((T_{\text{row}} U T_{\text{col}}^\top + E)x_{\text{OoD}}) \\ &\quad - g(T_{\text{row}} U T_{\text{col}}^\top x_{\text{OoD}})]\| \\ &\geq S_{\text{OoD}} - \|I - P\|_\sigma \|E x_{\text{OoD}}\| \\ &\geq S_{\text{OoD}} - \|E\| \quad (\text{since } \|I - P\|_\sigma = 1) \end{aligned} \quad (78)$$

Thus, we have:

$$\inf_{x_{\text{OoD}}} d(x_{\text{OoD}}) \geq S_{\text{OoD}} - \|E\| \quad (79)$$

Step 4: Strict separation condition: Given the empirical condition:

$$S_{\text{OoD}} > 2 \|E\| \quad (80)$$

we directly obtain the strict separation:

$$\begin{aligned} \sup_{x_{\text{ID}}} d(x_{\text{ID}}) &\leq \|E\| < S_{\text{OoD}} - \|E\| \\ &\leq \inf_{x_{\text{OoD}}} d(x_{\text{OoD}}) \end{aligned} \quad (81)$$

This proves that ID residuals lie strictly below OoD residuals, establishing the empirical spectral residual separation.

Appendix E: Hyperparameter Settings

The training follows a variational Bayesian approach with inducing point approximations applied to both convolutional and linear layers. Inducing point sizes, prior configurations, and regularization parameters are carefully chosen to balance expressivity and tractability.

The model is trained for 200 epochs using the Adam or SGD optimizer with standard data augmentations and label smoothing. The variational inference is configured using a diagonal Gaussian for the posterior over inducing variables, and KL annealing is employed during early training. Table 7 lists all key hyperparameters.

Table 7: Training Hyperparameters for Bayesian ResNet

Parameter	Value
Dataset	CIFAR-10 / CIFAR-100
Input Size	32×32
Model	ResNet-18
Epochs	200
Train Samples	1
Test Samples	8
Batch Size (train/test)	100 / 200
Learning Rate	$\eta = 10^{-3}$
Optimizer	Adam / SGD
Momentum (SGD)	$\mu = 0.9$
Milestones	[100]
Learning Rate Decay	$\gamma = 0.1$
Seed	42
Data Augmentation	Crop + Flip
Label Smoothing	$\epsilon = 0.05$
Bayesian Inference (Inducing Points)	
Inference Type	Inducing Point
Conv2d Inducing Size	128×128
Linear Inducing Size	100×128
Whitened Inducing Variables	True
$q(\mathbf{u})$ Covariance	Diagonal
Learn λ	True
Initial λ	$\lambda_{\text{init}} = 0.001$
Max λ	$\lambda_{\text{max}} = 0.03$
Prior Standard Deviation	$\sigma_{\text{prior}} = 1.0$
Max Std. Dev. of \mathbf{u}	$\sigma_u^{\text{max}} = 0.1$
Cache Cholesky Decomposition	True
Sqrt Width Scaling	True
Key Layers	layer2.1.conv1&2, layer4.1.conv1&2

The choice of hyperparameters reflects a balance between modeling flexibility and computational efficiency. In the optimization configuration, we adopt standard values for learn-

ing rate and momentum following common deep learning practices, while also introducing a learning rate scheduler to facilitate convergence. Label smoothing is applied with a moderate coefficient $\epsilon = 0.05$ to mitigate overconfidence in classification outputs. For the Bayesian inference procedure, we apply variational approximations with inducing point parameterizations. The number of inducing points is selected based on empirical validation: 128×128 for convolutional layers and 100×128 for fully connected layers, which ensures a good trade-off between approximation accuracy and memory footprint. We employ diagonal covariance in the variational posterior over the inducing variables $q(\mathbf{u})$ to reduce computational complexity, along with whitening and Cholesky caching to stabilize training and accelerate matrix operations. The scale parameters λ and σ are initialized conservatively and gradually increased, with learned constraints to avoid numerical instability.

Appendix F: Extra Lemma

Lemma 2 (Gaussian-Form Preservation). *Under the SVGP variational approximation*

$$q(f, u) = p(f | u) \underbrace{\mathcal{N}(u; m, S)}_{q(u)} \quad (82)$$

with the normalising-flow prior $p(\mathbf{u})$ as defined above, the marginal

$$q(f) = \int p(f | \mathbf{u}) q(\mathbf{u}) d\mathbf{u} \quad (83)$$

remains a Gaussian posterior with closed-form mean and covariance functions.

Proof Sketch Since $p(f | \mathbf{u})$ is Gaussian and $q(\mathbf{u})$ is Gaussian, their convolution yields a GP. The flow-based prior modifies only the KL term in the ELBO; it does not alter the conditional $p(f | \mathbf{u})$.