High-Fidelity Transfer of Functional Priors for Wide Bayesian Neural Networks by Learning Activations

M. Sendera, **T. Kuśmierczyk**

GMUM (UJ) tea seminar: 2024-11-08
seminar of Statistical Analysis and Modelling Group (IPI PAN): 2024-12-03

Background: Neural Networks

$$
\begin{aligned}
f_i^0(x) &= \sum_j^J w_{ij}^0 x_j + b_i^0, \quad i = 1, \ldots, H_0; \\
f^l(x) &= \sum_j^{H_0} w_j^l \phi(f_j^0(x)) + b^l \\
y &= \sigma(f^l(x))
\end{aligned}
$$

where x denotes the J-dimensional input, and $f^l(x)$ represents the output at the l-th layer; for regression: sample likelihood $\sigma = \delta$.

Background: Bayesian Neural Networks

$$f_i^0(x) = \sum_j^J w_{ij}^0 x_j + b_i^0, \quad i = 1, \ldots, H_0;$$

$$f^l(x) = \sum_j^{H_0} w_j^l \phi(f_j^0(x)) + b^l$$

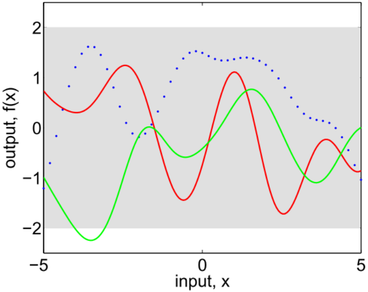
$$y = \sigma(f^l(x))$$

prior $p(w, b)$ + likelihood $p(y|w, b, x)$ + data $\{x, y\}$

$\rightarrow$ posterior $p(w, b|\{x, y\})$

Bayesian Neural Networks: a priori mapping

prior $p(w, b)$



$$f_i^0(x) = \sum_{j}^{J} w_{ij}^0 x_j + b_i^0$$

$$f^l(x) = \sum_{j}^{H_0} w_j^l \phi(f_j^0(x)) + b^l$$

Image from *GPs for ML (Carl Edward Rasmussen, Christopher K. I. Williams, 2006)*

Bayesian Neural Networks

$$\text{prior } p(w, b) + \text{likelihood } p(y|w, b, x) + \text{data } \{x, y\} \\ \rightarrow \text{posterior } p(w, b|\{x, y\})$$

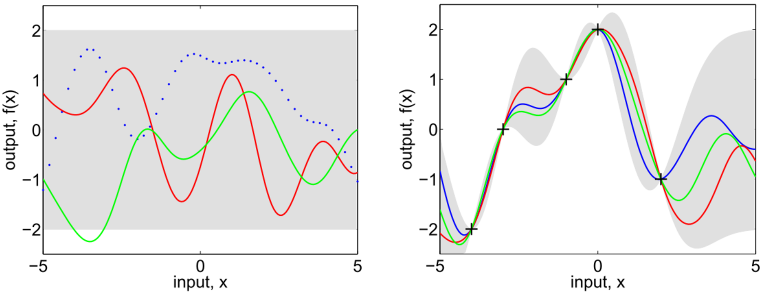
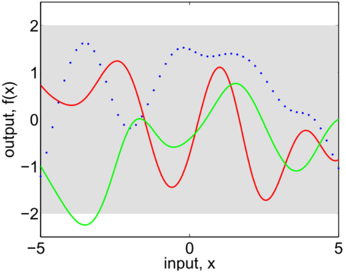
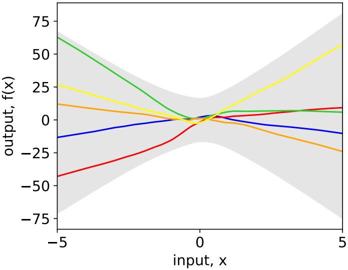


Image from *GPs for ML (Carl Edward Rasmussen, Christopher K. I. Williams, 2006)*

Bayesian Neural Networks: actual priors are not that nice

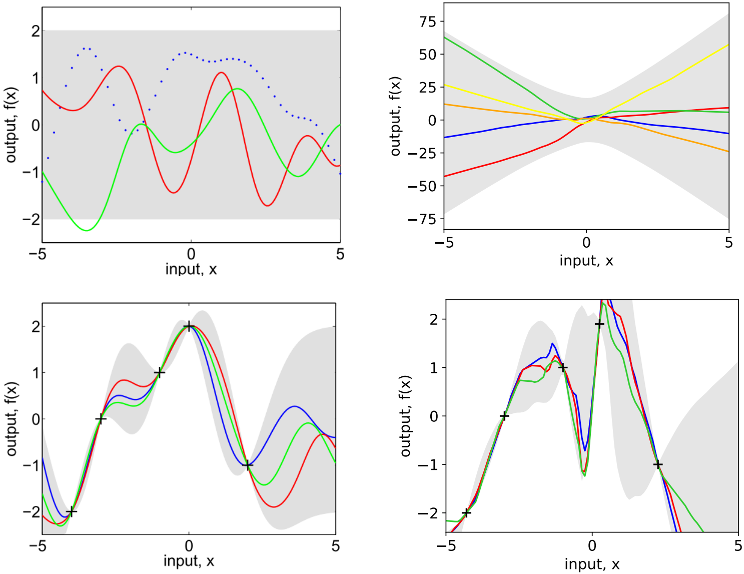
prior $p(w, b) = N(w|0, I)N(b|0, I)$





Bayesian Neural Networks: actual priors are not that nice

prior $p(w, b) = N(w|0, I)N(b|0, I)$



Bayesian Neural Networks: Function-space view

prior $p(f^l)$ + likelihood $p(y|f^l, x)$ + data $\{x, y\}$
$\rightarrow$ posterior $p(f^l|\{x, y\})$

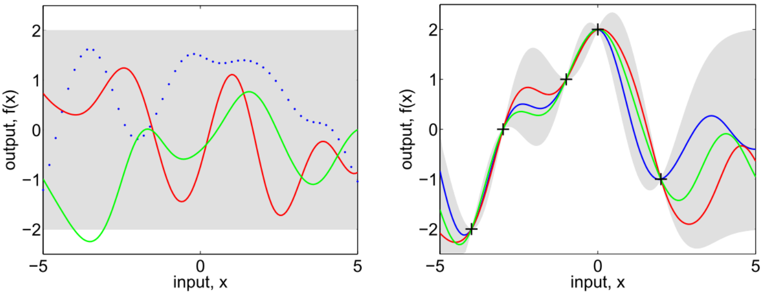


Image from *GPs for ML (Carl Edward Rasmussen, Christopher K. I. Williams, 2006)*

Weight-space vs Function-space

- BNNs rely on parameter-space priors (weights and biases).
- **Challenge**: Parameter-space priors don't directly constrain model outputs
  $\rightarrow$ reduced interpretability
- **Solution**: Function-space priors offer direct constraints on outputs:
  $\rightarrow$ smoothness and interpretability

GPs: Convenient Function-space Priors

$$f(x) \sim \mathrm{GP}(m(x), \kappa(x, x'))$$

where $m(x)$ is the mean function and $\kappa(x, x')$ is the kernel defining properties of $f(x)$.

GPs: Convenient Function-space Priors

- Gaussian Processes (GPs) provide a way to define priors over functions through **kernel specification**.

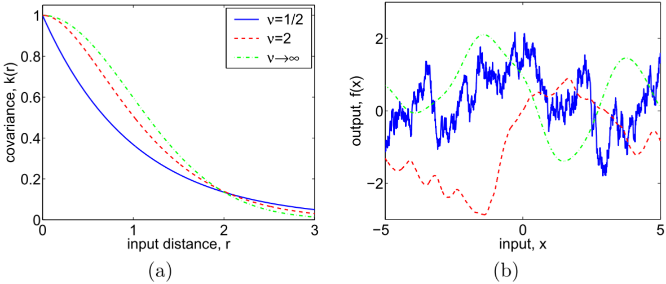


Figure 4.1: Panel (a): covariance functions, and (b): random functions drawn from Gaussian processes with Matérn covariance functions, eq. (4.14), for different values of ν, with $\ell = 1$. The sample functions on the right were obtained using a discretization of the x-axis of 2000 equally-spaced points.

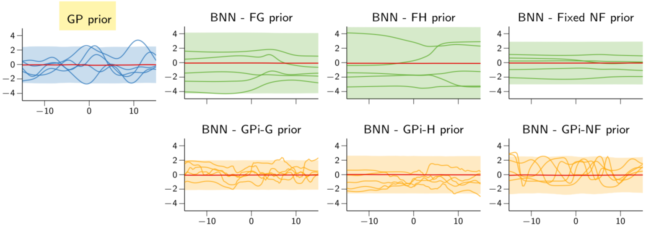
Image from *GPs for ML (Carl Edward Rasmussen, Christopher K. I. Williams, 2006)*

Enforcing Function-space Priors in BNNs: Method

- **Goal**: impose function-space GP priors in BNNs.
- **Method**:
  - Reparameterize $p(w, b|\lambda)$
  - $\lambda^* = \text{argmin}_\lambda \frac{1}{S} \sum_{X \sim p_X} D(p_{nn}(f^l(X|\lambda)), p_{gp}(f^l(X)))$

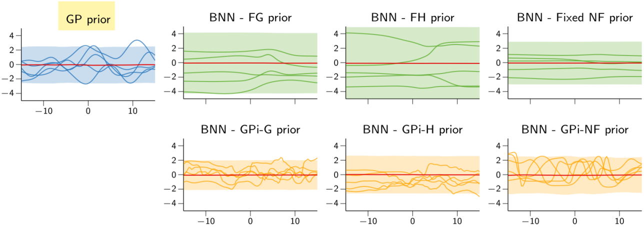
Enforcing Function-space Priors in BNNs: Method

- **Goal**: impose function-space GP priors in BNNs.
- **Method**:
  - Reparameterize $p(w, b|\lambda)$
  - $\lambda^* = \text{argmin}_\lambda \frac{1}{S} \sum_{X \sim p_X} D(p_{nn}(f^l(X|\lambda)), p_{gp}(f^l(X)))$



Tran et al. *All you need is a good functional prior for Bayesian deep learning.* JMLR 2022.

Enforcing Function-space Priors in BNNs: Challenges



Tran et al. *All you need is a good functional prior for Bayesian deep learning.* JMLR 2022.

- Complex models (e.g. NFs) for priors $p(w, b|\lambda)$ needed
- Tricky learning (=finding posteriors) for deep BNNs
- Little theoretical backing

Connecting BNNs and GPs: Wide BNNs are GPs

$$p(f_i^l(x)) \xrightarrow{H^l \to \infty} \mathcal{N}(\mu(x), \sigma^2(x)),$$

$$\mathrm{Cov}(f^l(x), f^l(x')) = {\sigma_b^l}^2 + {\sigma_w^l}^2 \mathbb{E}_{w_0, b}[\phi(w^0 x + b^0)\phi(w^0 x' + b^0)],$$

BNN corresponds to a GP$(\cdot,\ \kappa)$: $\kappa_f^l(x, x') = \mathrm{Cov}(f^l(x), f^l(x'))$

Connecting BNNs and GPs: Wide BNNs are GPs

$$p(f_i^l(x)) \xrightarrow{H^l \to \infty} \mathcal{N}(\mu(x), \sigma^2(x)),$$
$$\mathrm{Cov}(f^l(x), f^l(x')) = {\sigma_b^l}^2 + {\sigma_w^l}^2 \mathbb{E}_{w_0, b}[\phi(w^0 x + b^0)\phi(w^0 x' + b^0)],$$

BNN corresponds to a GP$(\cdot,\ \kappa)$: $\kappa_f^l(x, x') = \mathrm{Cov}(f^l(x), f^l(x'))$

- find κ_f^l given f^l: Easy

Connecting BNNs and GPs: Wide BNNs are GPs

$$p(f_i^l(x)) \xrightarrow{H^l \to \infty} \mathcal{N}(\mu(x), \sigma^2(x)),$$
$$\mathrm{Cov}(f^l(x), f^l(x')) = {\sigma_b^l}^2 + {\sigma_w^l}^2 \mathbb{E}_{w_0,b}[\phi(w^0 x + b^0)\phi(w^0 x' + b^0)],$$

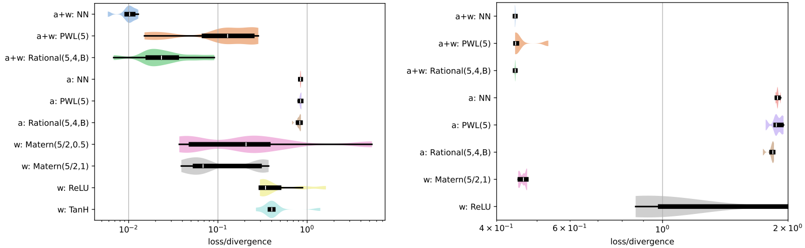
BNN corresponds to a GP($\cdot$, κ): $\kappa_f^l(x, x') = \mathrm{Cov}(f^l(x), f^l(x'))$

- find κ_f^l given f^l: Easy
- identify f^l given κ_f^l: Hard

Method

- Reparameterize priors and activation:
  $p(w, b|\lambda) = N(w|0, diag(\sigma_w))N(b|0, diag(\sigma_b))$, $\phi(\cdot|\eta)$
- $\lambda^* = \text{argmin}_\lambda \frac{1}{S} \sum_{X \sim p_X} D(p_{nn}(f^l(X|\lambda)), p_{gp}(f^l(X)))$, where $\lambda = \{\sigma_w, \sigma_b, \eta\}$
- Closed-form 2-Wasserstein divergence between two Gaussians:
  $D = ||\mu_1 - \mu_2||_2^2 + \text{Tr}\left(\Sigma_1 + \Sigma_2 - 2\sqrt{\sqrt{\Sigma_1}\Sigma_2\sqrt{\Sigma_1}}\right)$

Learning Activations



Fit quality for learned priors (w), activations (a) and both (a+w) for various activation models.

Input dimensionality =1D (left) / =16D (right).

Results for 1D Regression

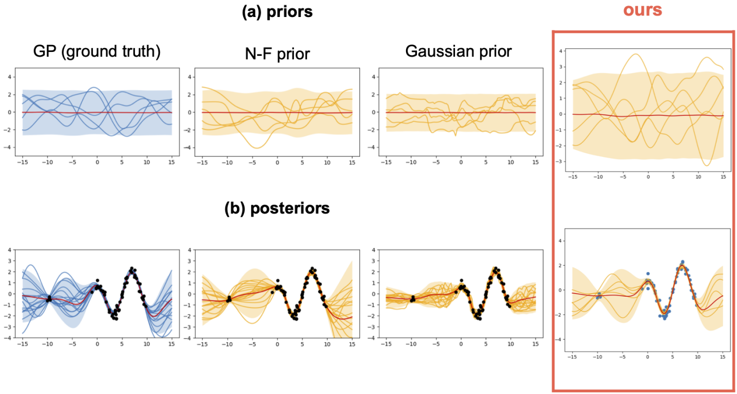


Figure: Prior **(a)** and posterior **(b)** predictive distributions for a BNN with trained parameters priors and activations (ours; 4th column), and for (Tran et al. 2022) approach with different prior realizations (Gaussian - 3rd column and Normalizing Flow - 2nd). The first column illustrates the ground truth (GP).

Results for 2D Classification

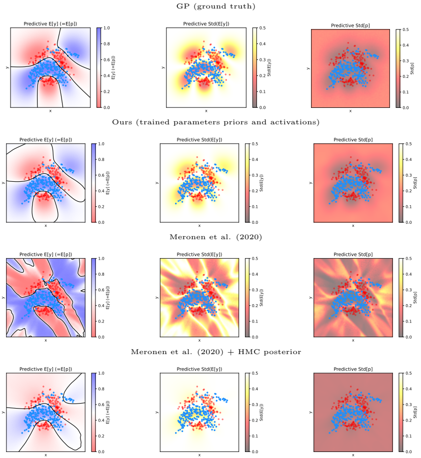
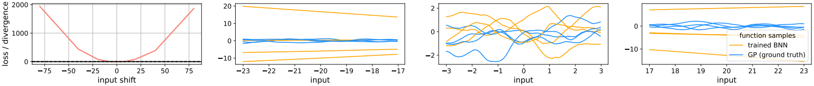


Figure: PPD for a BNN with trained parameters priors and activations (ours; 2nd row), and for a BNN with analytically derived activation (3rd and 4th row). 1st column = class probabilities, 2nd column = total variance in class predictions, 3rd column = epistemic uncertainty.

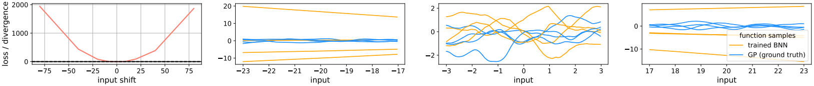
BNNs are (by default) non-stationary

BNN mimics stationary GP for $X \sim p_X$, but fails far from it:



BNNs are (by default) non-stationary

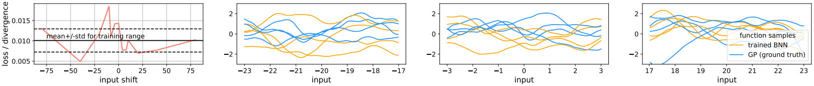
BNN mimics stationary GP for $X \sim p_X$, but fails far from it:



but periodic activations induce stationarity (Meronen et al. 2021):

$$\phi(x|\eta) = \sum_{i=1}^{K} A_i \cos(2\pi\psi_i x) + \sum_{j=1}^{K} A_j \sin(2\pi\psi_j x)$$

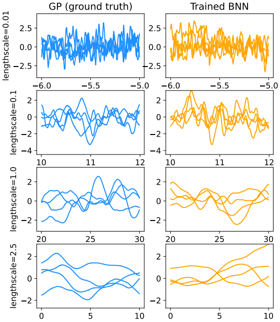
where $\eta = \{\psi_i, A_i, \psi_j, A_j\}$ and we used $K = 5$.



Conditioning

Idea: $p(w, b|\lambda(c))$, $\phi(\cdot|\eta(c))$, where c is a conditioning variable.

1. GPs' hyperparameters can be fit using marginal log-lik maximization, but BNNs are 'fixed':



$$c = \text{length scale}$$

2. Adopting to varying X-s (i.e. $c = X$; failing so far)

Acknowledgements

This research is part of the project No. 2022/45/P/ST6/02969 co-funded by the National Science Centre and the European Union Framework Programme for Research and Innovation Horizon 2020 under the Marie Skłodowska-Curie grant agreement No. 945339. For the purpose of Open Access, the author has applied a CC-BY public copyright licence.