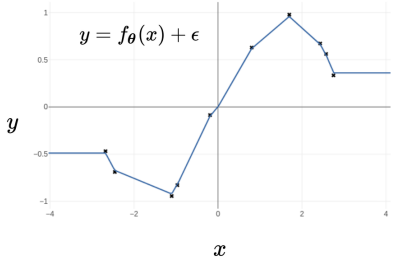
Towards Expressive Priors for BNNs: Poisson Process Radial Basis Function Networks

Melanie F. Pradier

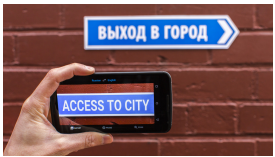
Joint work with Beau Coker and Finale Doshi-Velez
Harvard University

December 15th, 2019

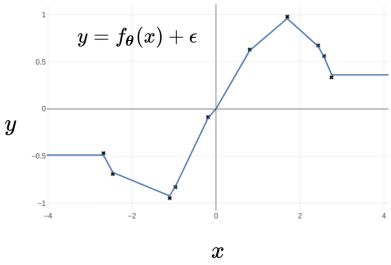
Neural networks (NNs) as universal approximators



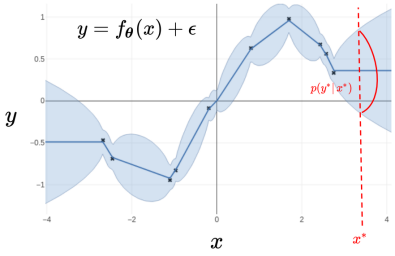
Several success stories...



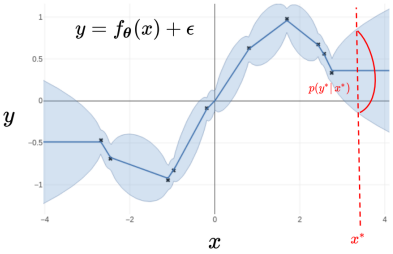
But what if stakes are high?



But what if stakes are high?

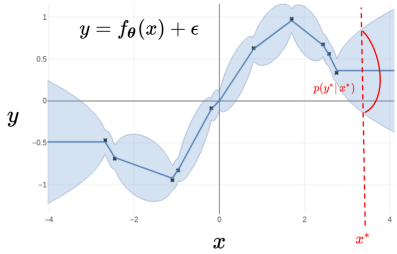


But what if stakes are high?



Uncertainty estimation becomes crucial!

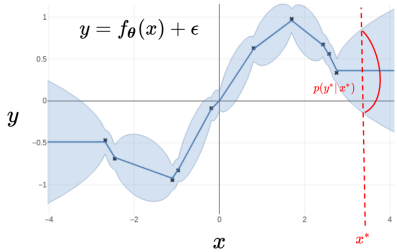
Sometimes we have a priori functional knowledge...



Some basic examples:

- Range of heart rate at rest between 60-100 bpm.
- Slow/fast variation of air pollutant
- Volatility of stock market

Sometimes we have a priori functional knowledge...



Some basic examples:

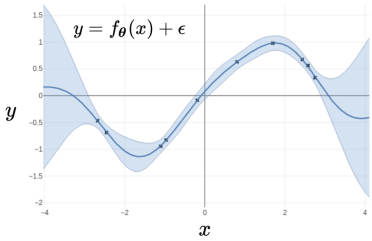
- Range of heart rate at rest between 60-100 bpm.
- Slow/fast variation of air pollutant
- Volatility of stock market

How can we incorporate such functional desiderata into the model?

An easy way to specify functional desiderata: Gaussian Processes (GPs)

Definition: *a Gaussian process is a collection of random variables, any finite number of which have (consistent) Gaussian distributions.*

$$f \sim \mathcal{N}\big(\mu(\cdot), k\,(\cdot,\cdot)\,\big)$$



Example: RBF kernel as covariance function:

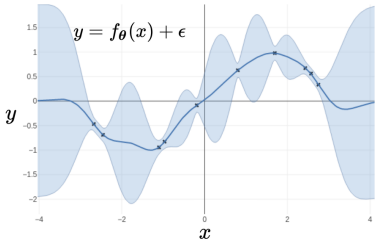
$$k(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2\gamma^2}\right)$$

- Stationarity
- Lengthscale
- Amplitude variance

An easy way to specify functional desiderata: Gaussian Processes (GPs)

Definition: *a Gaussian process is a collection of random variables, any finite number of which have (consistent) Gaussian distributions.*

$$f \sim \mathcal{N}\big(\mu(\cdot), k\,(\cdot,\cdot)\,\big)$$



Example: RBF kernel as covariance function:

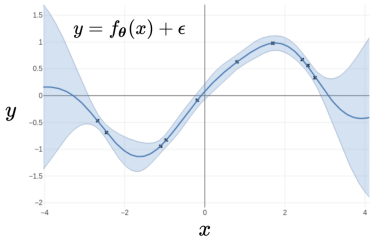
$$k(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2\gamma^2}\right)$$

- Stationarity
- Lengthscale
- Amplitude variance

An easy way to specify functional desiderata: Gaussian Processes (GPs)

Definition: *a Gaussian process is a collection of random variables, any finite number of which have (consistent) Gaussian distributions.*

$$f \sim \mathcal{N}\big(\mu(\cdot), k\,(\cdot,\cdot)\,\big)$$



Example: RBF kernel as covariance function:

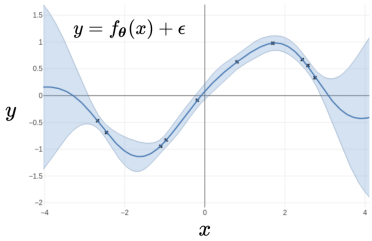
$$k(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2\gamma^2}\right)$$

- Stationarity
- Lengthscale
- Amplitude variance

An easy way to specify functional desiderata: Gaussian Processes (GPs)

Definition: *a Gaussian process is a collection of random variables, any finite number of which have (consistent) Gaussian distributions.*

$$f \sim \mathcal{N}\big(\mu(\cdot), k\,(\cdot,\cdot)\,\big)$$



Example: RBF kernel as covariance function:

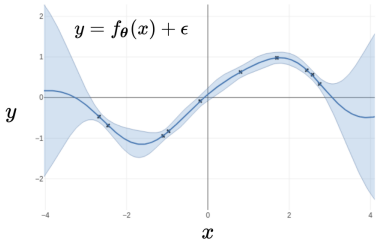
$$k(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2\gamma^2}\right)$$

- Stationarity
- Lengthscale
- Amplitude variance

An easy way to specify functional desiderata: Gaussian Processes (GPs)

Definition: *a Gaussian process is a collection of random variables, any finite number of which have (consistent) Gaussian distributions.*

$$f \sim \mathcal{N}\big(\mu(\cdot), k\,(\cdot,\cdot)\,\big)$$



Example: RBF kernel as covariance function:

$$k(x, x') = \sigma^2 \exp\left(-\frac{(x - x')^2}{2\gamma^2}\right)$$

- Stationarity
- Lengthscale
- Amplitude variance

GPs are great, but what if I still want a NN?

Benefits of NN approaches:

- widely used (many tools available)
- parametric expression
- fast at evaluation time

GPs are great, but what if I still want a NN?

Benefits of NN approaches:

- widely used (many tools available)
- parametric expression
- fast at evaluation time

Key Research Questions:

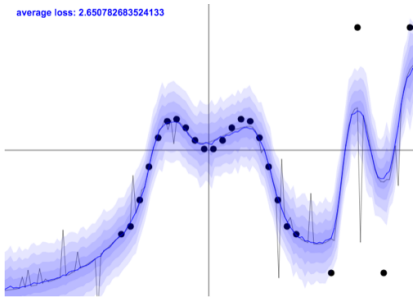
1. Can we design Bayesian NN priors that encode **stationarity properties** like a GP while retaining the benefits of neural networks?
2. Can we easily specify lengthscale and amplitude variance in a **decoupled** fashion?

Background

Bayesian Neural Networks

- Assume prior on network parameters
- Most common, i.i.d Gaussians

$$
\begin{aligned}
\mathbf{y} &= f_{\boldsymbol{\theta}}(\mathbf{x}) + \boldsymbol{\epsilon} \\
\epsilon &\sim \mathcal{N}(0, \sigma_y^2 I) \\
\theta_i &\sim \mathcal{N}(0, \sigma_\theta^2 I) \quad \forall i
\end{aligned}
$$



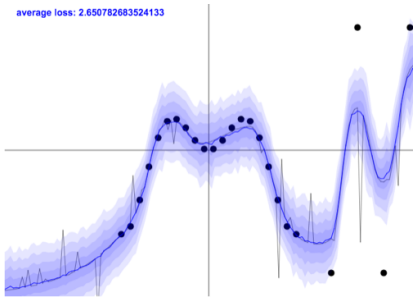
(Yarin Gal blog)

- $p(\boldsymbol{\theta}) \implies p(f)$

Bayesian Neural Networks

- Assume prior on network parameters
- Most common, i.i.d Gaussians

$$
\begin{aligned}
\mathbf{y} &= f_{\boldsymbol{\theta}}(\mathbf{x}) + \boldsymbol{\epsilon} \\
\epsilon &\sim \mathcal{N}(0, \sigma_y^2 I) \\
\theta_i &\sim \mathcal{N}(0, \sigma_\theta^2 I) \quad \forall i
\end{aligned}
$$



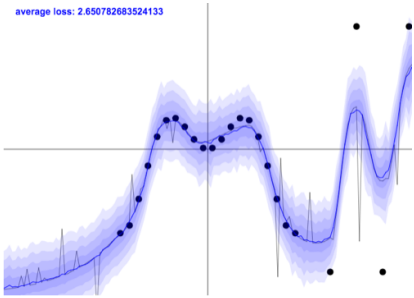
(Yarin Gal blog)

- $p(\boldsymbol{\theta}) \implies p(f)$
- But what does a prior over weights mean in function space?

Bayesian Neural Networks

- Assume prior on network parameters
- Most common, i.i.d Gaussians

$$
\begin{aligned}
\mathbf{y} &= f_{\boldsymbol{\theta}}(\mathbf{x}) + \boldsymbol{\epsilon} \\
\epsilon &\sim \mathcal{N}(0, \sigma_y^2 I) \\
\theta_i &\sim \mathcal{N}(0, \sigma_\theta^2 I) \quad \forall i
\end{aligned}
$$

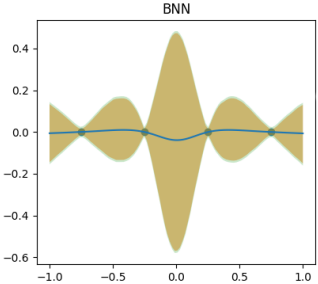


(Yarin Gal blog)

- $p(\boldsymbol{\theta}) \implies p(f)$
- But what does a prior over weights mean in function space? Hard to know!

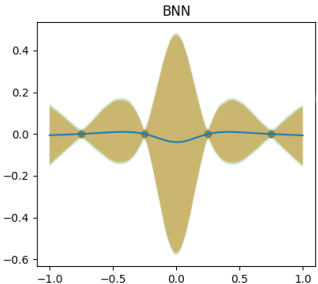
Not only hard to encode functional properties with BNNs; some properties are impossible to get

- For example, a BNN (with RBF activations) is nonstationary in amplitude variance (Williams, 1997)



Not only hard to encode functional properties with BNNs; some properties are impossible to get

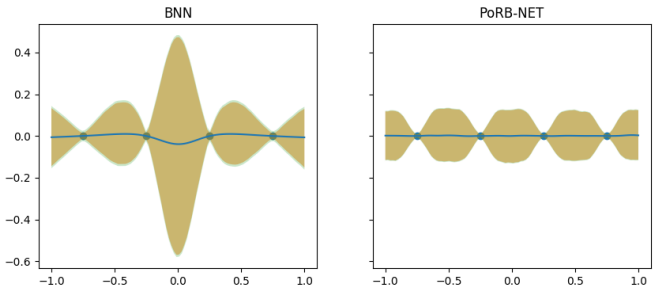
- For example, a BNN (with RBF activations) is nonstationary in amplitude variance (Williams, 1997)



Question: can we design a Bayesian NN that exhibits stationarity?

Not only hard to encode functional properties with BNNs; some properties are impossible to get

- For example, a BNN (with RBF activations) is nonstationary in amplitude variance (Williams, 1997)



Question: can we design a Bayesian NN that exhibits stationarity? Yes!

Related works

Expressive priors for Bayesian NNs

- Functional BNNs (Flam-Shepherd, et.al 2017; Sun et.al, 2019): sample-based optimization w.r.t. reference functional distribution
- Neural processes (Garnelo et al., 2018): meta-learning to identify functional properties based on many prior examples
- (Pearce et al., 2019) BNN architectures that recover equivalent GP kernel combinations in the infinite width limit

Related works

Expressive priors for Bayesian NNs

- Functional BNNs (Flam-Shepherd, et.al 2017; Sun et.al, 2019): sample-based optimization w.r.t. reference functional distribution
- Neural processes (Garnelo et al., 2018): meta-learning to identify functional properties based on many prior examples
- (Pearce et al., 2019) BNN architectures that recover equivalent GP kernel combinations in the infinite width limit

	user specs	optim. free	finite width	deep
Sun et.al, 2019	yes	no	yes	yes
Garnelo et.al, 2018	no	no	yes	yes
Pearce et.al, 2019	yes	yes	no	yes
PoRB-NET (this work)	yes	yes	yes	not yet

Radial Basis Function Networks (RBFNs)

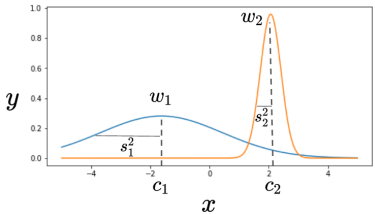
- Around since the 90s (Gyorfi et.al, 2002), recently renewed attention (Taghi et.al, 2004; Zadeh et.al, 2018)

Radial Basis Function Networks (RBFNs)

- Around since the 90s (Gyorfi et.al, 2002), recently renewed attention (Taghi et.al, 2004; Zadeh et.al, 2018)
- NN based on radial basis $\phi(\cdot)$, e.g., $\phi(x) = \exp(-x^2)$

$$f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \phi\big(s_k(x - c_k)\big),$$

- $s_k^2 \in \mathbb{R}$: scale
- $c_k \in \mathbb{R}$: center
- $w_k \in \mathbb{R}$: output weight
- $b \in \mathbb{R}$: output bias



Comparison RBFN versus BNN formulation (D=1)

$$f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \phi\big(s_k(x - c_k)\big)$$

- $s_k^2 \in \mathbb{R}$: scale
- $c_k \in \mathbb{R}$: center
- $w_k \in \mathbb{R}$: output weight
- $b \in \mathbb{R}$: output bias

$$f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \phi\big(v_k x + d_k\big)$$

- $v_k \in \mathbb{R}$: input weight
- $d_k \in \mathbb{R}$: input bias
- $w_k \in \mathbb{R}$: output weight
- $b \in \mathbb{R}$: output bias

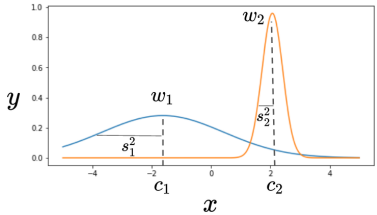
Take-away: priors on different random quantities, RBFN more intuitive

Bayesian RBFNs (Barber et.al, 1998)

$$
\begin{aligned}
c_k &\sim \mathcal{N}(0, \sigma_c^2) \\
s_k^2 &\sim \text{Gamma}(\alpha_s, \beta_s) \\
w_k &\sim \mathcal{N}\left(0, \sigma_w^2 I\right) \\
b &\sim \mathcal{N}(0, \sigma_b^2) \\
y_n \,|\, x_n, \boldsymbol{\theta} &\sim \mathcal{N}(f_{\boldsymbol{\theta}}(x_n), \sigma_y^2)
\end{aligned}
$$

where

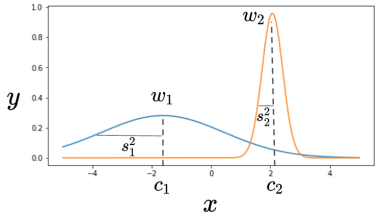
$$
f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \exp\left(-s_k^2 (x - c_k)^2\right)
$$



- $s_k^2 \in \mathbb{R}$: scale
- $c_k \in \mathbb{R}$: center
- $w_k \in \mathbb{R}$: output weight
- $b \in \mathbb{R}$: output bias

Bayesian RBFNs (Barber et.al, 1998)

$$
\begin{aligned}
c_k &\sim \mathcal{N}(0, \sigma_c^2) \\
s_k^2 &\sim \text{Gamma}(\alpha_s, \beta_s) \\
w_k &\sim \mathcal{N}\left(0, \sigma_w^2 I\right) \\
b &\sim \mathcal{N}(0, \sigma_b^2) \\
y_n \,|\, x_n, \boldsymbol{\theta} &\sim \mathcal{N}(f_{\boldsymbol{\theta}}(x_n), \sigma_y^2)
\end{aligned}
$$



where

$$
f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \exp\left(-s_k^2 (x - c_k)^2\right)
$$

- $s_k^2 \in \mathbb{R}$: scale
- $c_k \in \mathbb{R}$: center
- $w_k \in \mathbb{R}$: output weight
- $b \in \mathbb{R}$: output bias

Functional properties still hard or impossible to encode!

Functional properties still hard or impossible

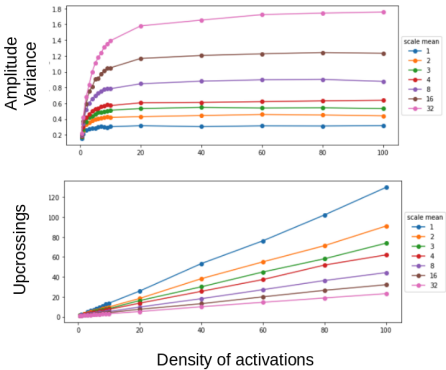
Issues:

- non-stationary covariance function (Williams, 1997)
- lengthscale and variance are **coupled**

Functional properties still hard or impossible

Issues:

- non-stationary covariance function (Williams, 1997)
- lengthscale and variance are **coupled**



- As RBFs concentrate in same region:
  - summation $\implies$ higher variance
  - increase in expressivity $\implies$ more upcrossings

Model

Poisson Process Radial Basis Function Networks (PoRB-NET)

$$
\begin{aligned}
c_k &\sim \mathcal{N}(0, \sigma_c^2) \\
s_k^2 &\sim \text{Gamma}(\alpha_s, \beta_s) \\
w_k &\sim \mathcal{N}\left(0, \sigma_w^2 I\right) \\
b &\sim \mathcal{N}(0, \sigma_b^2) \\
y_n \,|\, x_n, \boldsymbol{\theta} &\sim \mathcal{N}(f_{\boldsymbol{\theta}}(x_n), \sigma_y^2)
\end{aligned}
$$

where

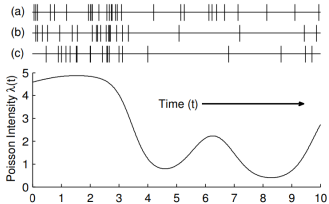
$$
f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \exp\left(-s_k^2 (x - c_k)^2\right)
$$

Poisson Process Radial Basis Function Networks (PoRB-NET)

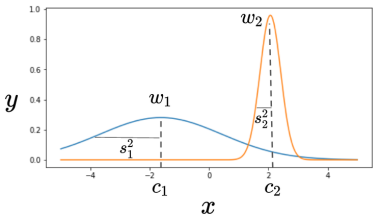
$$
\begin{aligned}
\mathbf{c} \,|\, \lambda &\sim \text{Poisson Process}\,(\lambda) \\
s_k^2 &\sim \text{Gamma}(\alpha_s, \beta_s) \\
w_k &\sim \mathcal{N}\left(0, \tilde{\sigma}_w^2 I\right) \\
b &\sim \mathcal{N}(0, \tilde{\sigma}_b^2) \\
y_n \,|\, x_n, \boldsymbol{\theta} &\sim \mathcal{N}(f_{\boldsymbol{\theta}}(x_n), \sigma_y^2)
\end{aligned}
$$

where

$$
f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \exp\left(-s_k^2 (x - c_k)^2\right)
$$



(Adams et.al, 2009)

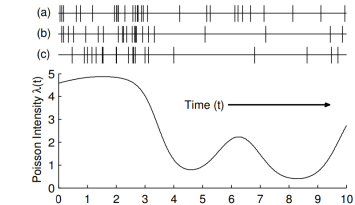


Poisson Process Radial Basis Function Networks (PoRB-NET)

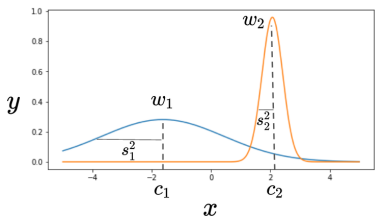
$$
\begin{aligned}
\mathbf{c} \,|\, \lambda &\sim \text{Poisson Process}\,(\lambda) \\
s_k^2 &= \lambda^2(c_k) \\
w_k &\sim \mathcal{N}\left(0, \tilde{\sigma}_w^2 I\right) \\
b &\sim \mathcal{N}(0, \tilde{\sigma}_b^2) \\
y_n \,|\, x_n, \boldsymbol{\theta} &\sim \mathcal{N}(f_{\boldsymbol{\theta}}(x_n), \sigma_y^2)
\end{aligned}
$$

where

$$
f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \exp\left(-s_k^2 (x - c_k)^2\right)
$$



(Adams et.al, 2009)



What if we don't know the intensity function?

What if we don't know the intensity function?

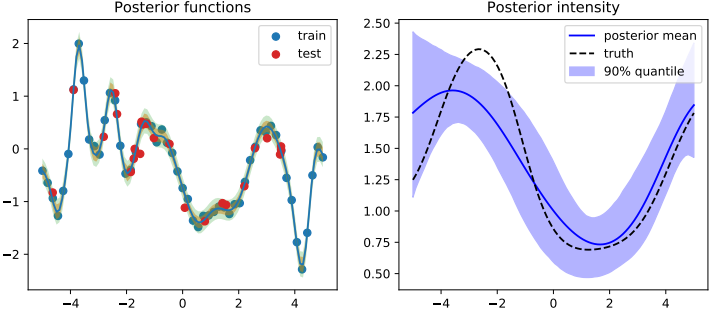
Prior on Intensity Function of Poisson Process

$$\begin{aligned} h &\sim \text{GP}(0, C(\cdot,\cdot)) \\ \lambda^* &\sim \text{Gamma}(\alpha_\lambda, \beta_\lambda) \\ \lambda(c) &= \lambda^* \text{sigmoid}(h(c)), \end{aligned}$$

What if we don't know the intensity function?

Prior on Intensity Function of Poisson Process

$$\begin{aligned} h &\sim \mathrm{GP}(0, C(\cdot,\cdot)) \\ \lambda^* &\sim \mathrm{Gamma}(\alpha_\lambda, \beta_\lambda) \\ \lambda(c) &= \lambda^* \mathrm{sigmoid}(h(c)), \end{aligned}$$



Inference

1. Update network parameters $\boldsymbol{\theta}$ given fixed nr. of hidden units K via Hamiltonian Monte Carlo (HMC)

$$p(\boldsymbol{\theta} \,|\, \mathbf{y}, \mathbf{x}, K, \lambda) \propto \left(\prod_{n=1}^{N} \mathcal{N}(y_n; f(x_n; \boldsymbol{\theta}) \right) \mathcal{N}(b; 0, \sigma_b^2) \left(\prod_{k=1}^{K} \mathcal{N}(w_k; 0, \sigma_w^2)\ \lambda(c_k) \right)$$

2. Update network width K via birth/death moves
3. Update point-estimate for Poisson process intensity λ

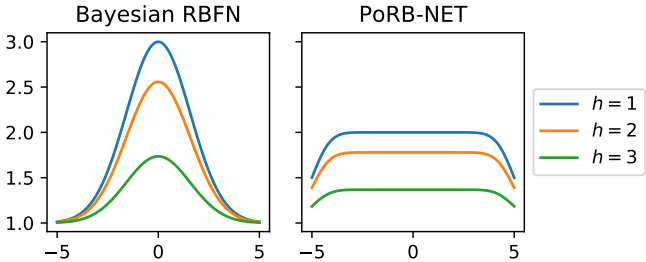
$$\hat{\lambda}(c) \approx \frac{1}{S} \sum \lambda^\star \phi(h^{(s)}(c)),$$

where $h^{(s)} \sim p(h|\mathbf{y}, \mathbf{x}, \boldsymbol{\theta})$.

Properties

Stationarity

$$\mathrm{Cov}(f(x), f(x+h)) = \sigma_b^2 + \sigma_w^2 \mathbb{E}\left[K\right] \underbrace{\mathbb{E}_\theta\left[\rho(x;\theta)\rho(x+h;\theta)\right]}_{:=U(x,x+h)}$$



$$U(x_1, x_2) \propto \underbrace{\exp\left(-\frac{(x_1 - x_2)^2}{2(2\sigma_s^2 + \sigma_s^4/\sigma_c^2)}\right)}_{Stationary} \underbrace{\exp\left(-\frac{x_1^2 + x_2^2}{2(2\sigma_c^2 + \sigma_s^2)}\right)}_{Nonstationary}$$

$$U(x_1, x_2) = \frac{\lambda}{\Lambda}\sqrt{\frac{\pi}{s^2}} \exp\left\{-s^2\left(\frac{x_1 - x_2}{2}\right)^2\right\} \left[\Phi((C_1 - x_m)\sqrt{2s^2}) - \Phi((C_0 - x_m)\sqrt{2s^2}\lambda)\right]$$

Decoupled lengthscale and amplitude variance

- **Homogeneous Poisson Process**
  - We derive closed-form expression for covariance function
  - Poisson process defined over finite region $\mathcal{C}$
  - As size of $\mathcal{C}$ tends to infinity,

$$\mathrm{Cov}\left(f(x_1), f(x_2)\right) \approx \sigma_b^2 + \tilde{\sigma}_w^2 \exp\left\{-\lambda^2 \left(\frac{x_1 - x_2}{2}\right)^2\right\}$$

- **Non-homogeneous Poisson Process**
  - Empirical stationarity

Consistency

- Estimator $\hat{g}_n(x)$ is said to be consistent with respect to the true regression function $g_0(x)$ if, as n tends to infinity:

$$\int (\hat{g}_n(x) - g_0(x))^2 \, dx \xrightarrow{p} 0.$$

- Posterior consistent over Hellinger neighborhoods if $\forall \epsilon > 0$,

$$p(\{f : D_H(f, f_0) \leq \epsilon\}) \xrightarrow{p} 1.$$

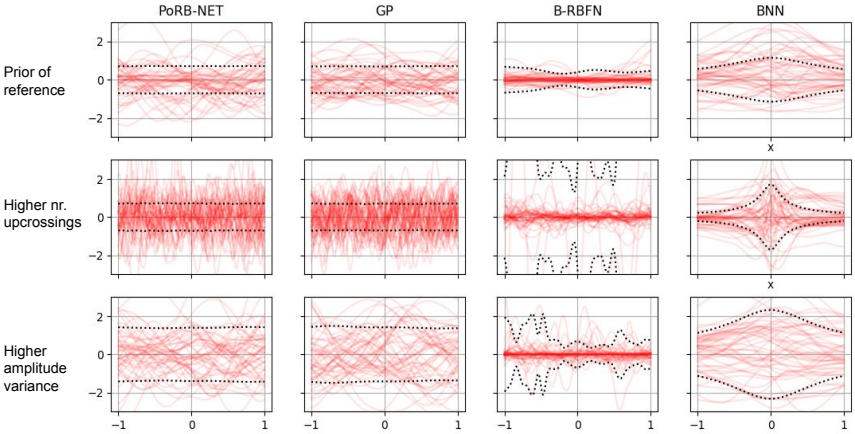
- (Lee,2000) shows that Hellinger consistency implies frequentist consistency.

Theorem (Consistency of PoRB-NETs)

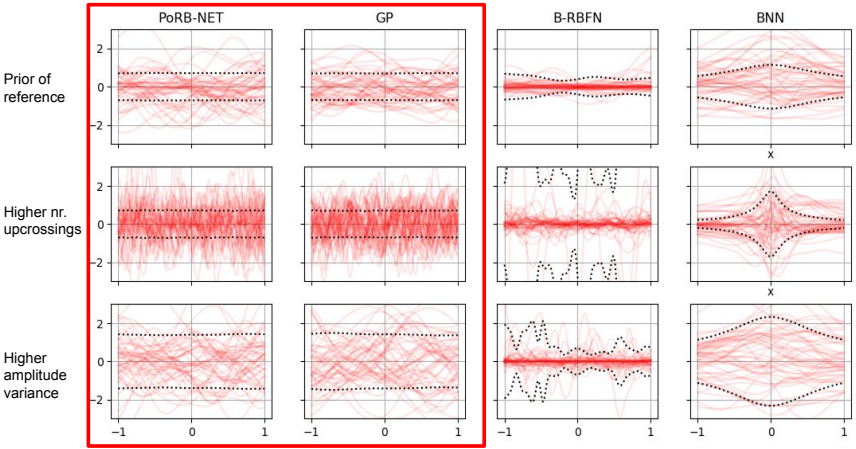
A PoRB-NET with uniform intensity function is Hellinger consistent as the number of observations goes to infinity.

Empirical Results

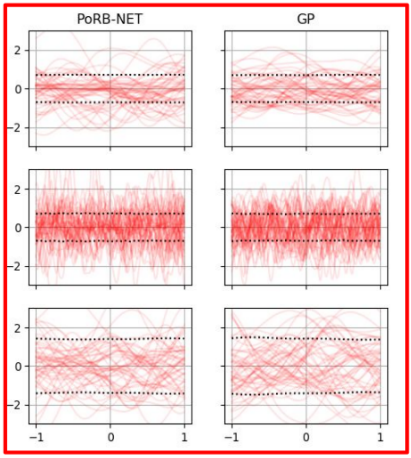
PoRB-NET allows for easy specification of lengthscale and signal variance like a GP



PoRB-NET allows for easy specification of lengthscale and signal variance like a GP

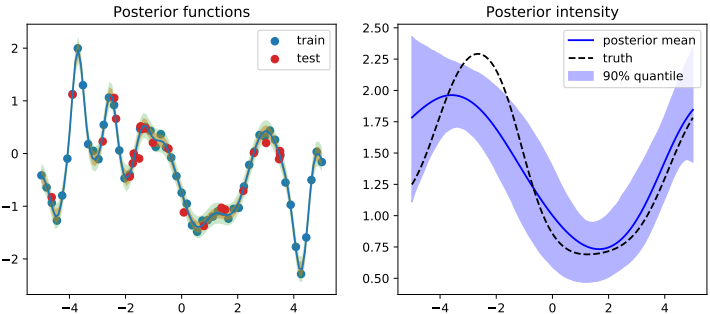


PoRB-NET allows for easy specification of lengthscale and signal variance like a GP



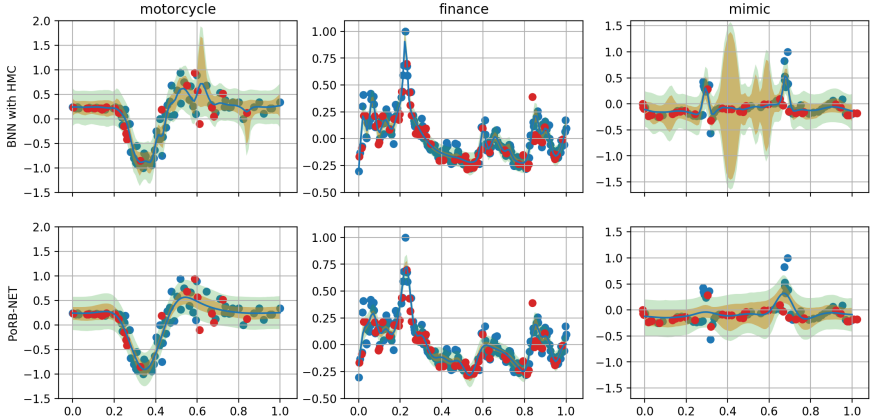
1. stationarity
2. easy specification in a decoupled manner

PoRB-NET is able to learn input-dependent lengthscale information



PoRB-NET adds more hidden units wherever needed, and adapts architecture width based on the data.

PoRB-NET is able to capture non-stationary patterns in real scenarios, adapting the lengthscale locally



Conclusion

Conclusion

In this talk, we have...

- highlighted incapacity of BNNs to express functional properties
- introduced PoRB-NET, a Bayesian NN prior to encode functional desiderata like a GP
- proposed an inference scheme to learn input-dependent lengthscale
- showed theoretical properties: (i) consistency, (ii) decoupling of amplitude and lengthscale
- validated empirically in synthetic and real datasets

All information online: https://arxiv.org/abs/1912.05779

Conclusion

In this talk, we have...

- highlighted incapacity of BNNs to express functional properties
- introduced PoRB-NET, a Bayesian NN prior to encode functional desiderata like a GP
- proposed an inference scheme to learn input-dependent lengthscale
- showed theoretical properties: (i) consistency, (ii) decoupling of amplitude and lengthscale
- validated empirically in synthetic and real datasets

All information online: https://arxiv.org/abs/1912.05779

As future work: deeper networks, higher dimensions.

Conclusion

In this talk, we have...

- highlighted incapacity of BNNs to express functional properties
- introduced PoRB-NET, a Bayesian NN prior to encode functional desiderata like a GP
- proposed an inference scheme to learn input-dependent lengthscale
- showed theoretical properties: (i) consistency, (ii) decoupling of amplitude and lengthscale
- validated empirically in synthetic and real datasets

All information online: https://arxiv.org/abs/1912.05779

As future work: deeper networks, higher dimensions.

Thank you for listening!

Comparison RBFN versus BNN formulation (D=1)

$$f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \phi\big(s_k(x - c_k)\big) \qquad f_{\boldsymbol{\theta}}(x) = b + \sum_{k=1}^{K} w_k \phi\big(v_k x + b_k\big)$$

$$\begin{aligned} s_k^2 &\sim \text{Gamma}(\alpha_s, \beta_s) \\ c_k &\sim \mathcal{N}(0, \sigma_c^2) \\ w_k &\sim \mathcal{N}(0, \sigma_w^2) \\ b &\sim \mathcal{N}(0, \sigma_0^2) \end{aligned} \qquad \begin{aligned} v_k^2 &\sim \mathcal{N}(0, \sigma_v^2) \\ b_k &\sim \mathcal{N}(0, \sigma_b^2) \\ w_k &\sim \mathcal{N}(0, \sigma_w^2) \\ b &\sim \mathcal{N}(0, \sigma_0^2) \end{aligned}$$

Take-away: priors on different random quantities, RBFN more intuitive