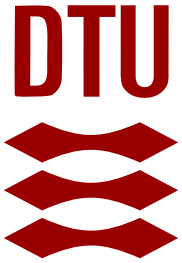


Technical University of Denmark

Assignment

Course:
Special Course: Gaussian Process

Author(s):
Anders Vestergaard (s154993)

22/2/2021

Contents

1 Introduction **2**
1.1 Outline of Report . . . 2
1.2 Gaussian Process . . . 2
1.3 Predictive Distribution . . . 5

2 Input Warped GP **8**
2.1 Intuition About the Transformation M . . . 9
2.2 Learning the Probabilistic Transformation M . . . 11
2.3 Predictive Distribution of The BNN-GP Model eq. (27) . . . 12

3 Sampling from Posterior **13**

4 Choosing Step-size, ϵ **14**

5 Predicting using HMCMC Samples **17**

6 Experimental Results **18**
6.1 Step Function . . . 19
6.2 Wall Pulse . . . 23

7 Conclusion and Discussion **25**

1 Introduction

Gaussian Processes (GP) models have, as any model do, pros and cons. One con is the challenge of discovering new kernels, which limit GPs ability to express data. One method that transforms stationary kernels into non-stationary kernels is the method of input warping. In [5] the warping, M, is modelled as a neural network. This report makes M a probabilistic mapping using a Bayesian Neural Network (BNN).

1.1 Outline of Report

First, Gaussian Processes is presented in their generality. Following is the introduction of the input warped GP model. Thirdly the probabilistic mapping, M, is learned using Hamiltonian Markov Chain Monte Carlo. Finally, the model is reviewed on a step function and a two-pulse function.

1.2 Gaussian Process

First, a general introduction to Gaussian Processes (GP). There are several approaches to explaining GP, the road chosen start with the generalised linear model eq. (1)

$$y = \phi(\mathbf{x})^T\mathbf{w} + \eta,\ \mathbf{x} \in \mathbb{R}^n, y \in \mathbb{R} \tag{1}$$
$$\eta \sim \mathcal{N}(0, \sigma_n^2) \tag{2}$$
$$\tag{3}$$

Let $\mathbf{w}$ be stochastic, and give it a normal prior $\mathbf{w} \sim \mathcal{N}(0, \Sigma)$.
Define $f(\mathbf{x}) := \phi(\mathbf{x})^T\mathbf{w}$. Suppose that we observe N points $\{y_i, \mathbf{x}_i\}_{i=1}^N$ and arrange the data in the vector y and matrix X:

$$\mathbf{y} = [y_1, ..., y_N]^T \tag{4}$$
$$X = \begin{bmatrix} \mathbf{x}_1^T \\ . \\ . \\ . \\ \mathbf{x}_N^T \end{bmatrix} \tag{5}$$

We wish to make predictions for a given input X_* using the data $(X, \mathbf{y})$. in the non-Bayesian approach, one would choose the weights $\mathbf{w}$ according to some criterion. In the Bayesian approach, we perform a weighted average over all the possible parameter values, $\mathbf{w}$, where the weights are the posterior $p(\mathbf{w}|X, \mathbf{y})$ eq. (6)

$$E[y(X*)|X\mathbf{y}] = \int \Phi^T\mathbf{w}p(\mathbf{w}|X, \mathbf{y})d\mathbf{w} \tag{6}$$

The posterior, $p(\mathbf{w}|X,\mathbf{y})$, is found using Bayes theorem eq. (7)

$$p(\mathbf{w}|X,\mathbf{y}) = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}} = \frac{p(\mathbf{y}|X,\mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|X)} \tag{7}$$

In this case the Evidence, $p(\mathbf{y}|X)$, is also called the Marginal Likelihood, since it is found by marginalising the likelihood over the weights, $\mathbf{w}$ eq. (8).

$$\text{Evidence} = p(\mathbf{y}|X) = \int \text{Likelihood} \times \text{Prior} = \int p(\mathbf{y}|X,\mathbf{w})p(\mathbf{w})d\mathbf{w} \tag{8}$$

In general the Evidence is difficult to evaluate and often intractable, therefor approximate solutions is often required for the posterior, $p(\mathbf{w}|X,\mathbf{y})$. But in the realm of normal likelihood and prior we can evaluate the exact posterior, $p(\mathbf{w}|X,\mathbf{y})$, as seen in eq. (9). Let $\phi(X) = \Phi$ and evaluate eq. (7):

$$p(\mathbf{w}|X,\mathbf{y}) \propto e^{-\frac{1}{2}\left(\frac{1}{\sigma_n^2}(\mathbf{y}-\Phi^T\mathbf{w})^T(\mathbf{y}-\Phi^T\mathbf{w})+\mathbf{w}^T\Sigma^{-1}\mathbf{w}\right)} \tag{9a}$$

$$\propto e^{-\frac{1}{2}\left(\frac{1}{\sigma_n^2}(\mathbf{y}^T\mathbf{y}-\mathbf{y}^T\Phi^T\mathbf{w}-\mathbf{w}^T\Phi\mathbf{y}+\mathbf{w}^T\Phi\Phi^T\mathbf{w})+\mathbf{w}^T\Sigma^{-1}\mathbf{w}\right)}, \quad \mathbf{y}^T\mathbf{y} \text{ is independent of } \mathbf{w} \tag{9b}$$

$$\propto e^{-\frac{1}{2}\left(\mathbf{w}^T(\frac{1}{\sigma_n^2}\Phi\Phi^T+\Sigma^{-1})\mathbf{w}-\frac{1}{\sigma_n^2}(\mathbf{w}^T\Phi\mathbf{y}+(\mathbf{w}^T\Phi\mathbf{y})^T)\right)} \tag{9c}$$

Matching $\mathcal{N}(\mu,S) \propto e^{\frac{-1}{2}(\mathbf{x}S^{-1}\mathbf{x}-\mathbf{x}^TS^{-1}\mu-\mu^TS^{-1}\mathbf{w})}$ with eq. (9c), we get that $\mathbf{w}|X,\mathbf{y} \sim \mathcal{N}(\bar{\mathbf{w}}, A^{-1})$ where:

$$A = (\frac{1}{\sigma_n^2}\Phi\Phi^T + \Sigma^{-1}) \tag{10}$$

$$\frac{1}{\sigma_n^2}\Phi\mathbf{y} = A\bar{\mathbf{w}} \iff \bar{\mathbf{w}} = \frac{1}{\sigma_n^2}A^{-1}\Phi\mathbf{y} \tag{11}$$

Note the similarity between $A^{-1}\Phi$ and the pseudo inverse used in Linear Regression.
Since the posterior, $p(\mathbf{w}|X,\mathbf{y})$, is normal the mode equals the mean and therefor $\bar{\mathbf{w}}$ is the Maximum a Posteriori (MAP) estimate, which in a non-Bayesian approach would correspond to a penalised Maximum Likelihood (ML) estimate. A normal prior on $\mathbf{w}$ corresponds to $L2$ regularisation. To formulate the GP we convert from a weight-space view ($\mathbf{w}$) to a function-space view by integrating out the weights, $\mathbf{w}$. First, introduce the latent function-space $\mathbf{f} = f(X) = \Phi^T\mathbf{w}$ and calculate the posterior, $p(\mathbf{f}|X,\mathbf{y})$, by marginalising over $\mathbf{w}$ using the posterior $p(\mathbf{w}|X,\mathbf{y})$ as weights.

$$cap(\mathbf{f}|X,\mathbf{y}) = \int p(\mathbf{f}|X,\mathbf{w})p(\mathbf{w}|X,\mathbf{y}) = \int p(\mathbf{f}|\mathbf{w})p(\mathbf{w}|X,\mathbf{y}) \tag{12}$$

We recognise that $p(\mathbf{f}|X,\mathbf{y})$ will be normal, and hence we calculate mean and variance directly using the mean and variance of $\mathbf{w}|X,\mathbf{y}$:

$$\mathbb{E}[\mathbf{f}|X,\mathbf{y}] = \mathbb{E}[\Phi^T\mathbf{w}|X,\mathbf{y}] = \Phi^T\mathbb{E}[\mathbf{w}|X,\mathbf{y}] = \frac{1}{\sigma_n^2}\Phi^T A^{-1}\Phi\mathbf{y} \tag{13a}$$

$$\mathbb{V}[\mathbf{f}|X,\mathbf{y}] = \Phi^T\mathbb{V}[\mathbf{w}|X,\mathbf{y}]\Phi = \Phi^T A^{-1}\Phi \tag{13b}$$

eq. (13) can be reformulated as eq. (14)

$$\mathbb{E}[\mathbf{f}|X,\mathbf{y}] = \Phi^T\Sigma\Phi(K+\sigma_n^2 I)^{-1}\mathbf{y} \tag{14a}$$

$$\mathbb{V}[\mathbf{f}|X,\mathbf{y}] = \Phi^T\Sigma\Phi - \Phi^T\Sigma\Phi(\Phi^T\Sigma\Phi+\sigma_n^2 I)^{-1}\Phi^T\Sigma\Phi \tag{14b}$$

Note that if $\sigma_n^2 = 0$ then $\mathbb{E}[\mathbf{f}|X,\mathbf{y}] = \mathbf{y}$ and $\mathbb{V}[\mathbf{f}|X,\mathbf{y}] = 0$ as expected. Define $k(\mathbf{x},\mathbf{x}') = \phi(\mathbf{x})^T\Sigma\phi(\mathbf{x}')$, and call k the **kernel** function. As Σ is positive definite, we can define $\Sigma^{\frac{1}{2}}$. The SVD of $\Sigma^{\frac{1}{2}}$ is $\Sigma^{\frac{1}{2}} = UD^{\frac{1}{2}}U^T$ since:

$$\Sigma = \Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}} = (UD^{\frac{1}{2}}U^T)(UD^{\frac{1}{2}}U^T) = UD^{\frac{1}{2}}ID^{\frac{1}{2}}U^T = UDU^T \tag{15}$$

Define the weighted feature space $\psi(\mathbf{x}) = \Sigma^{\frac{1}{2}}\phi(\mathbf{x})$ then $k(\mathbf{x},\mathbf{x}')$ can be written as a dot-product in ψ-space eq. (16)

$$k(\mathbf{x},\mathbf{x}') = \psi(\mathbf{x})^T\psi(\mathbf{x}') = (\Sigma^{\frac{1}{2}}\phi(\mathbf{x}))^T(\Sigma^{\frac{1}{2}}\phi(\mathbf{x}')) = \phi(\mathbf{x})^T\Sigma\phi(\mathbf{x}') \tag{16}$$

Using eq. (16) we see that the feature space Φ in eq. (14) is only defined by dot-products, and we can therefor use the kernel trick, and not worry about a parametric representation of ϕ, but instead construct a suitable kernel, k. We now use this formulation to construct a prior on $\mathbf{f} = \Phi^T\mathbf{w}$ using the prior $\mathbf{w} \sim \mathcal{N}(0,\Sigma)$. We find the mean and variance of the normal prior, $p(\mathbf{f}|X)$, in eq. (17) using properties of mean and variance.

$$\mathbb{E}[\mathbf{f}|X] = \Phi^T\mathbb{E}[\mathbf{w}|X] = \Phi^T\mathbb{E}[\mathbf{w}] = 0 \tag{17a}$$

$$\mathbb{V}[\mathbf{f}|X] = \Phi^T\mathbb{V}[\mathbf{x}|X]\Phi = \Phi^T\mathbb{V}[\mathbf{x}]\Phi = \Phi^T\Sigma\Phi = K \tag{17b}$$

where we have used $K_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$, for $i, j = 1, ..., N$. eq. (17) define a Gaussian process with zero mean and kernel function, k, which in short is written as:

$$f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}')) \tag{18}$$

In more general terms **Definition:** A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution [4].

Using the Squared Exponential (SE) kernel, given in eq. (22), with $l = 0.5, \sigma_f^2 = 1$ we generate 15 priors by drawing $f(X)$ from eq. (19), the samples are shown in fig. 1.

$$f(X) \sim \mathcal{N}(0, K) \tag{19}$$

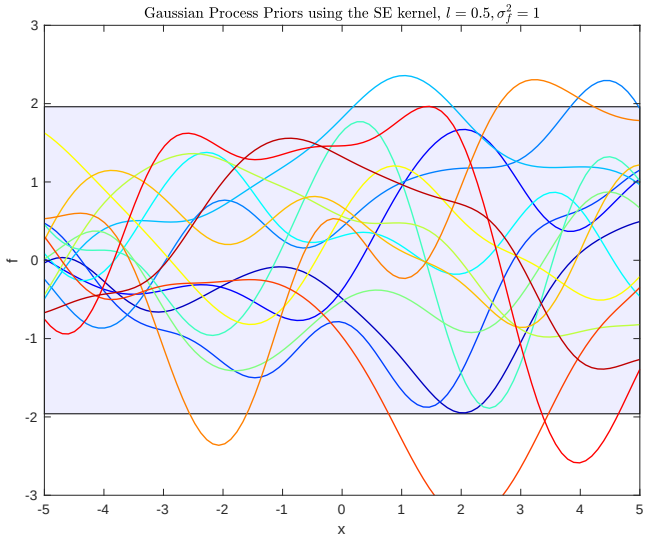


Figure 1: Gaussian Process priors, sampled from eq. (19) using the SE-kernel with $l = 0.5$, $\sigma_2^f = 1$, $n = 1$. The shaded area indicate the 95% confidence interval $= \pm\sqrt{\texttt{diag}(\mathbf{V}[\mathbf{f}|X])} = \pm\sqrt{\text{diag}(K)} = \pm 1.96\sigma_f^2 e$.

1.3 Predictive Distribution

Let X_* denote the input points at which we want to predict $\mathbf{y}_*$. The joint distribution of $[\mathbf{y}, \mathbf{f}_*]^T$ can the be written as:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + I\sigma_n^2 & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} = \begin{bmatrix} K + I\sigma_n^2 & K_*^T \\ K_* & K_{**} \end{bmatrix}\right) \tag{20}$$

We can then calculate the conditional distribution $p(\mathbf{f}_*|X_*, X, \mathbf{y})$ which is the predictive distribution in eq. (21)

$$\mathbf{f}_*|X_*, X, \mathbf{y} \sim \mathcal{N}(\bar{\mathbf{f}}_*, \text{cov}[\mathbf{f}_*]) \tag{21a}$$
$$\bar{\mathbf{f}}_* = \mathbb{E}[\mathbf{f}_*|X, \mathbf{y}, X_*] = K_*(K + \sigma_n^2 I)^{-1}\mathbf{y} \tag{21b}$$
$$\text{cov}[\mathbf{f}_*] = K_{**} - K_*(K + \sigma_n^2 I)^{-1}K_*^T \tag{21c}$$

In this report we choose the **stationary** Squared Exponential (SE) kernel:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 e^{-\frac{1}{2l}||\mathbf{x}-\mathbf{x}'||_2^2} \tag{22}$$

Parameter	**Description**
l	Length Scale
σ_n^2	Signal variance

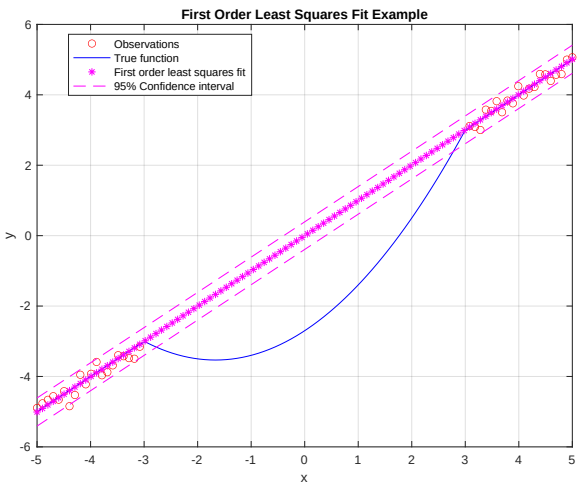
A nice feature about GP's is their ability to explain uncertainty, which is inherited from the probabilistic framework (Bayes) in which GP's are descried in. We show this by an example, consider the process in eq. (23):

$$y(x) = f(x) + \eta \tag{23}$$
$$f(x) = \begin{cases} x. & x < -3 \\ x,, & x > 3 \\ 0.3x^2 + x - 0.3 \cdot 9, & -3 \leq x \leq 3 \end{cases} \tag{24}$$
$$\eta \sim \mathcal{N}(0, 0.2^2) \tag{25}$$

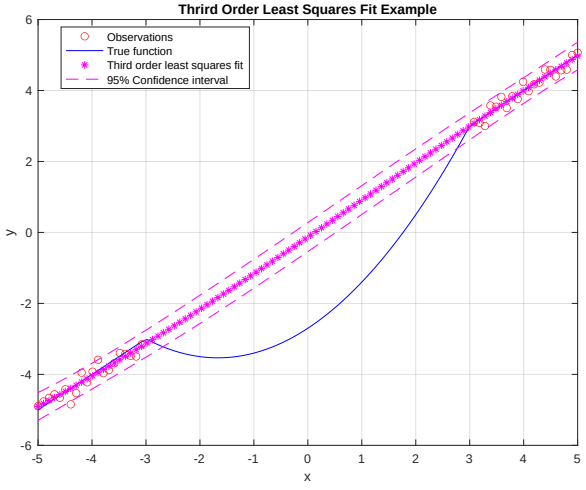
We sample $y(x)$ for $x \in \{\{x \in \mathbb{R}| -5 < x < -3\} \cup \{x \in \mathbb{R}|3 < x < 5\}\}$, and fit the data with the following 4 models:

1. Linear model, using least squares and plot the fit together with 95% confidence interval and $f(x)$ in fig. 2a.
2. Third order polynomial model, using least squares and plot the fit together with 95% confidence interval and $f(x)$ in fig. 2b.
3. Gaussian Process, using eq. (21) with a SE-Kernel and $l = 1.5, \sigma_f^2 = 1$ the the fit together with 95% confidence interval and $f(x)$ in fig. 2c.
4. Gaussian Process, using eq. (21) with a dot-product kernel, $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$. The fitted values together with 95% confidence interval and $f(x)$ is seen in fig. 2d.

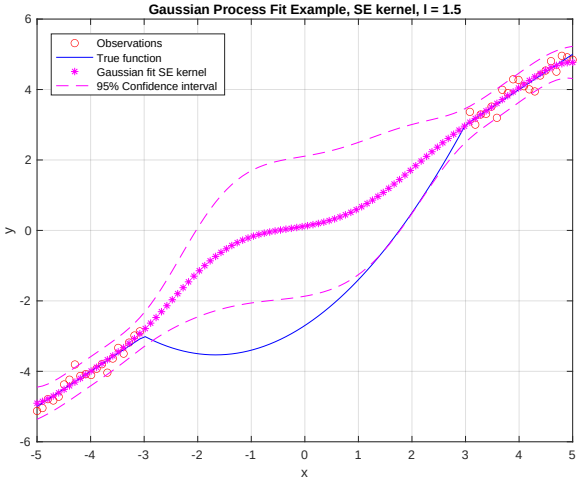
The purpose of the example in fig. 2 is to show that the Gaussian Process can explain uncertainty in areas where information is low due to the behaviour encoded in the prior, specified by the kernel. If one use the dot-product kernel $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$ the prior functions become linear, and the Gaussian process uncertainty (and mean) would be similar to the linear model as seen by comparing fig. 2a and fig. 2d. The ability to explain uncertainty in regions of low information is a nice and logical feature of the Gaussian Process framework and can enable more robust decisions. This example also visualises the importance of choosing a "proper" kernel to encode prior knowledge.



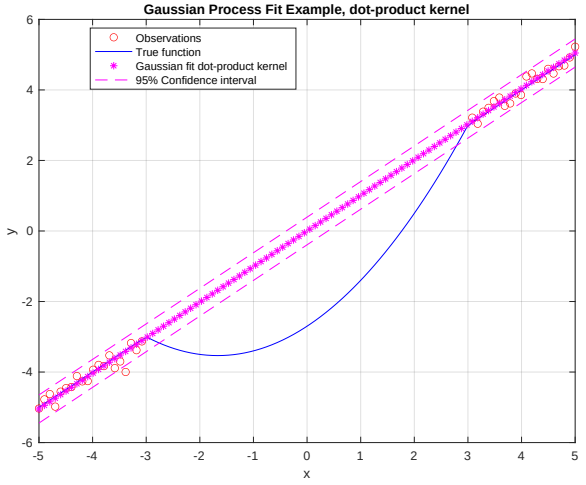
(a) First order least squares fit. Note that the uncertainty **does not increase** in areas where there is no/less observations/information



(b) Third order least squares fit. Note that the uncertainty **does not increase** in areas where there is no/less observations/information



(c) Gaussian Process uncertainty example using the SE kernel. Note that the uncertainty **increase** in areas where there is no/less observations/information



(d) Gaussian Process uncertainty example. Note that the uncertainty **does not increase** in areas where there is no/less observations/information, due to the dot-product kernel, $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$, encoding prior information about linearity

Figure 2: Example of uncertainty results, between linear/third order least squares and Gaussian Process

2 Input Warped GP

Consider the general result about kernels, given in eq. (26):

$$\text{Given a function: } M : \mathbb{R}^n \to \mathbb{R}^m, \text{ and a kernel } k_1(\cdot,\cdot) : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R} \tag{26a}$$
$$\text{then } k(\mathbf{x}, \mathbf{x}') = k_1(M(\mathbf{x}), M(\mathbf{x}')) \text{ is also a valid kernel [6]} \tag{26b}$$

As described in [4] one can introduce **non-stationarity** by introducing an arbitrary **non-linear** transformation $M : \mathbb{R}^n \to \mathbb{R}^m$ from input space. Suppose we observe a non-stationary process, we then wish to find a transformation, M, from the input space, such that the resulting process, $f(M(x))$, is stationary, which then enables us to use the SE kernel (or any other stationary kernel) in this new space. We formulate the model in eq. (27)

$$M(\mathbf{x}) = \mathbf{u}, \ \mathbf{x} \in \mathbb{R}^n, \mathbf{u} \in \mathbb{R}^m \tag{27a}$$
$$y = f(\mathbf{u}) + \eta \tag{27b}$$
$$f(\mathbf{u}) \sim \mathcal{GP}(0, k(\mathbf{u}, \mathbf{u}')) \tag{27c}$$
$$\eta \sim \mathcal{N}(0, \sigma_n^2) \tag{27d}$$

A graphical interpretation of eq. (27) is given in fig. 3.

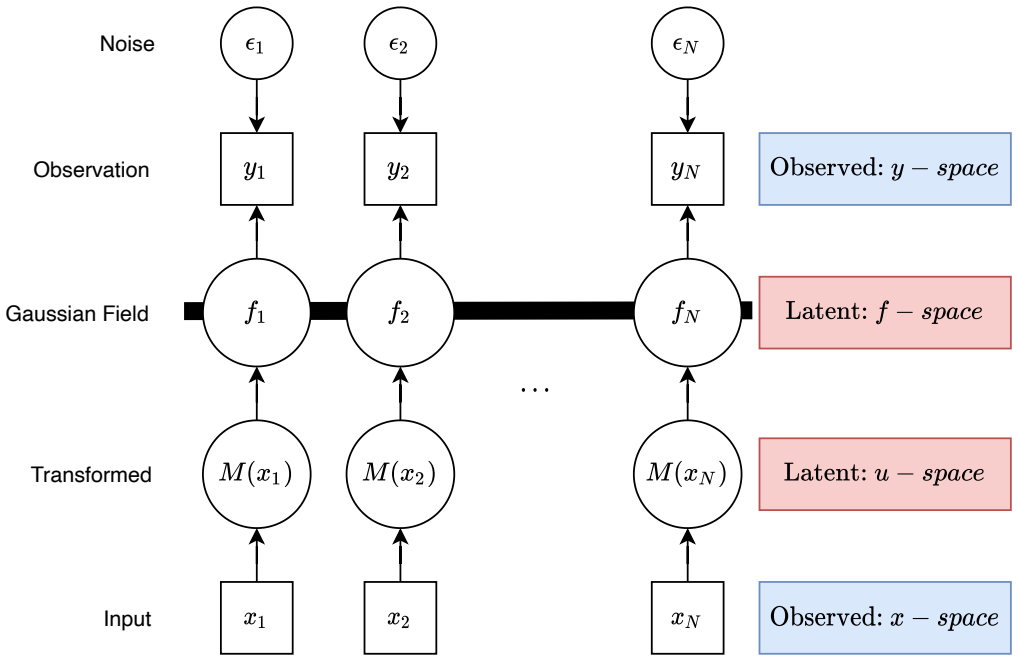


Figure 3: Graphical model of eq. (27). The horizontal bar represent a set of fully connected nodes. Circle is latent space (unobserved variables), and squares are observed variables.

2.1 Intuition About the Transformation M

Any stationary kernel, $k(\mathbf{x}, \mathbf{x}')$, can be rewritten as a function of $\mathbf{x} - \mathbf{x}'$. Consider the SE kernel applied to the transformed input $M(\mathbf{x}) = \mathbf{u}$:

$$k(\mathbf{u}, \mathbf{u}') = e^{-\frac{||\mathbf{u}-\mathbf{u}'||_2^2}{2l}} \tag{28}$$

Consider two input points, $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$, assume that $\mathbf{x} \neq \mathbf{x}'$ and $l < \infty$ then $k(\mathbf{x}, \mathbf{x}') < \sigma_f^2$. Suppose that M maps $\mathbf{x}$ and $\mathbf{x}'$ to the same point $\mathbf{u}$, then $k(\mathbf{u}, \mathbf{u}) = \sigma_f^2$ which corresponds to infinite length scale, $l = \infty$, in $x - space$. The below, loose, statements

- $||\mathbf{x} - \mathbf{x}'||_2 > ||\mathbf{u} - \mathbf{u}||_2$, corresponds to **increasing** the length-scale at the point $(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^n \times \mathbb{R}^n$
- $||\mathbf{x} - \mathbf{x}'||_2 < ||\mathbf{u} - \mathbf{u}||_2$, corresponds to **decreasing** the length-scale at the point $(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^n \times \mathbb{R}^n$
- $||\mathbf{x} - \mathbf{x}'||_2 = ||\mathbf{u} - \mathbf{u}||_2$, corresponds to **not changing** the length-scale at the point $(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^n \times \mathbb{R}^n$

suggest that there is an equivalence between M and the effective length-scale. Consider the kernel $k(\mathbf{x}, \mathbf{x}')$ in eq. (29) with a spatial variable length-scale $l(\mathbf{x}, \mathbf{x}')$:

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 e^{-\frac{||\mathbf{x}-\mathbf{x}'||_2^2}{2l(\mathbf{x},\mathbf{x}')}} \tag{29}$$

In general eq. (29) is not a valid kernel [2]. However consider the variable length-scale $l_M(\mathbf{x}, \mathbf{x}')$ calculated from the transformation M eq. (30).

$$\sigma_f^2 e^{-\frac{||\mathbf{x}-\mathbf{x}'||_2^2}{2l_M(\mathbf{x},\mathbf{x}')}} = \sigma_f^2 e^{-\frac{||\mathbf{u}-\mathbf{u}'||_2^2}{2l}} \iff l_M(\mathbf{x}, \mathbf{x}') = \begin{cases} \infty, & \mathbf{u} = \mathbf{u}' \\ l\frac{||\mathbf{x}-\mathbf{x}'||_2^2}{||\mathbf{u}-\mathbf{u}'||_2^2}, & \text{otherwise} \end{cases} \tag{30}$$

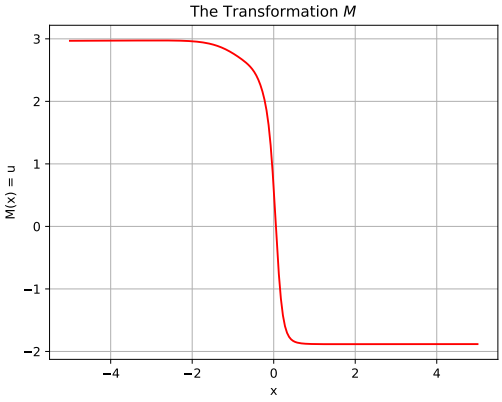
Since $0 \leq ||\mathbf{u} - \mathbf{u}||_2^2$, we have that $\lim_{\mathbf{u}\to\mathbf{u}'} ||\mathbf{u} - \mathbf{u}||_2^2 = 0^+$ will approach zero from the positive side, and therefore we get the limit:

$$\lim_{\mathbf{u}\to\mathbf{u}'} l\frac{||\mathbf{x} - \mathbf{x}'||_2^2}{||\mathbf{u} - \mathbf{u}'||_2^2} = l\frac{||\mathbf{x} - \mathbf{x}'||_2^2}{\lim_{\mathbf{u}\to\mathbf{u}'} ||\mathbf{u} - \mathbf{u}'||_2^2} = \infty \tag{31}$$

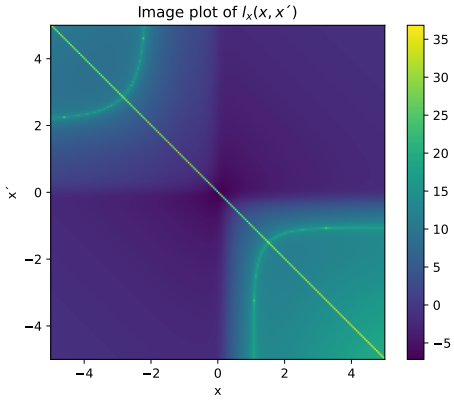
An example of eq. (30) for $n = m = 1$ is shown in fig. 4, where we consider the transformation shown in fig. 4a, we then calculate $l_M(x, x')$ using M and eq. (32) for $l = 0.1$. $l_M(x, x')$ is plotted in **log space** in fig. 4b. In fig. 4c we show the kernel matrix generated using in u-space, eq. (28).

In fig. 4d the kernel matrix generated in x-space using eq. (29) and $l_M(x, x')$, given in fig. 4b, is shown.

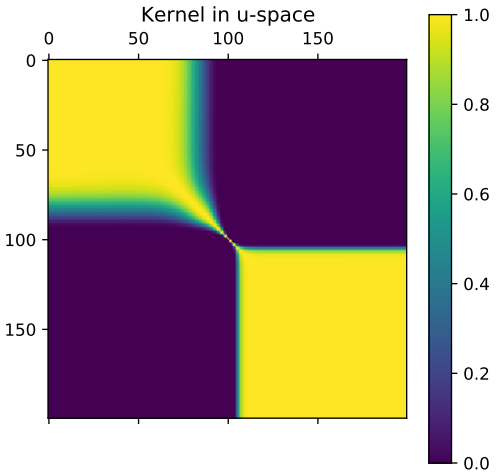
$$l_M(x, x') = \begin{cases} \infty, & u = u' \\ l\frac{(x-x')^2}{(u-u')^2}, & \text{otherwise} \end{cases} \tag{32}$$



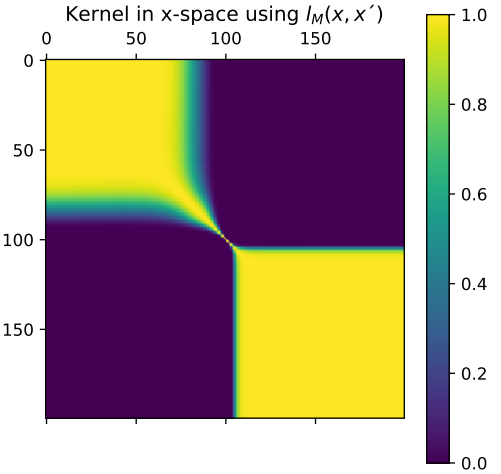
(a) The transformation M, used to show the equivalence between the transformation M, and the effective length scale.



(b) Variable length-scale matrix $L_{i,j} = l_M(x_i, x_j)$ calculated from eq. (32) using the transformation M shown in fig. 4a . The matrix is shown in **log space** i.e $\log L$



(c) Kernel matrix, $K_{i,j} = k(u_i, u_j) = \sigma_f^2 e^{-\frac{(u_i-u_j)^2}{2l}}$, calculated directly in u-space for $l = 0.1$, $\sigma_n^2 = 1$



(d) Kernel matrix, $K_{i,j} = k(x_i, x_j) = \sigma_f^2 e^{-\frac{(x_i-x_j)^2}{2l_M(x_i, x_j')}}$, $\sigma_f^2 = 1$, calculated in $x - space$ with the variable length-scale, $l_x(_i x, x_j)$, shown in fig. 4b

Figure 4: Example of the equivalence between the variable length scale and the transformation M as given in eq. (30). For $l = 0.1$, $\sigma_f^2 = 1$, $n = m = 1$ and M shown in fig. 4a

2.2 Learning the Probabilistic Transformation M

In this report, we wish to learn a probabilistic mapping, $p(M(X)|X, \mathbf{y})$, using a Bayesian Neural Network (BNN). The network's weights and biases are collected in the parameter vector θ, given a zero-mean multivariate normal distribution with variance σ_θ^2.

$$\theta \sim \mathcal{N}(0, \sigma_\theta^2 I) \tag{33}$$

The hyper-parameters of the model, θ_h, is given in eq. (34):

$$\theta_h = \{\sigma_f^2, l, \sigma_n^2, A\} \tag{34}$$

Parameter	**Description**
l	Length Scale of the kernel
σ_n^2	Signal variance of the kernel
σ_n^2	White noise variance
A	Architecture of the BNN

The hyper-parameters are considered **fixed** in this report. Suppose we observe the data $(\mathbf{y}, X)$, we then wish to sample from the posterior $p(\theta|X, \mathbf{y})$. From Bayes theorem we get eq. (35)

$$p(\theta|\mathbf{y}, X) \propto p(\mathbf{y}|U)p(U|\theta, X)p(\theta) \tag{35}$$

Parameter	**Description**
U	u-space. $U = M(X)$
$p(\theta) \sim \mathcal{N}\mathbf{0}, \sigma_\theta^2 I)$	M-Prior
$p(U\|\theta, X)$	M-Likelihood
$p(y\|U)$	Marginal Likelihood
$p(\theta\|\mathbf{y}, X)$	Posterior Distribution

where the Marginal Likelihood (ML), $p(\mathbf{y}|U)$, is found by marginalisation over each prior $\mathbf{f}$ weighted by its probability $p(\mathbf{f}|U)$:

$$\begin{aligned}\log p(\mathbf{y}|U) &= \log \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|U)d\mathbf{f} \\ &= -\mathbf{y}^T(K + \sigma_n^2 I)^{-1}\mathbf{y} - \log|K + \sigma_n^2 I| - \frac{N}{2}\log 2\pi\end{aligned}$$

The distribution $p(U|\theta, X)$ is a design choice. In this report, we choose the normal in eq. (37).

$$P(U|\theta, X) \sim \mathcal{N}(\mathbf{0}, \sigma_M^2 I), \text{ Biased towards: } U = \mathbf{0} \tag{37}$$
$$\tag{38}$$

We write up the log probabilities from eq. (35) in eq. (39). Constants will be called when sampling, and therefore they are removed from eq. (39)

$$\log p(\theta) \propto -\frac{\theta^T \theta}{\sigma_\theta^2} \tag{39a}$$
$$\log p(U|\theta, X) \propto -\frac{U^T U}{\sigma_M^2} \tag{39b}$$
$$\log p(\mathbf{y}|U) \propto -\mathbf{y}^T (K + \sigma_n^2 I)^{-1} \mathbf{y} - \log |K + \sigma_n^2 I| \tag{39c}$$

where $K = K(U, U)$ is the kernel (Gram) matrix in u-space.

2.3 Predictive Distribution of The BNN-GP Model eq. (27)

In this section we find the predictive distribution, $p(\mathbf{f}_*|X_*, X, \mathbf{y})$, of the model in eq. (27) with the probabilistic mapping M. Given training data, $(X, \mathbf{y})$ test data X_* and the parameters θ for M, we can predict $\mathbf{f}_*|X_*, X, \mathbf{y}, \theta$ by its expected value, $\bar{\mathbf{f}}_{*,\theta} = E[\mathbf{f}_*|X_*, X, \mathbf{y}, \theta]$ using eq. (21) in u-space:

$$\bar{\mathbf{f}}_{*,\theta} := \mathbb{E}[\mathbf{f}_*|X_*, X, \mathbf{y}, \theta] = K_*(K + \sigma_n^2 I)^{-1} \mathbf{y} \tag{40a}$$
$$\text{cov}[\mathbf{f}_*|X_*, X, \mathbf{y}, \theta] = K_{**} - K_*(K + \sigma_n^2)^{-1} K_*^T \tag{40b}$$

Parameter	**Description**
θ	Parameter of M.
X_*	Input points for which we want to predict
$M(X_*; \theta) = U_*$	Predictive u-space for parameter θ
K_*	SE kernel matrix for (U_*, U)
K_{**}	SE kernel matrix for (U_*, U_*)

A weighted average on $\bar{\mathbf{f}}_{*,\theta}$ over θ is performed using the posterior $p(\theta|X, \mathbf{y})$ as weights. As seen in eq. (41) this integral gives the predicted values $\mathbb{E}[\mathbf{f}_*|X_*, X, \mathbf{y}]$ of the model.

$$\int \bar{\mathbf{f}}_{*,\theta} p(\theta|X,\mathbf{y})d\theta = \int \mathbb{E}[\mathbf{f}_*|X_*,X,\mathbf{y},\theta]p(\theta|X,\mathbf{y})d\theta \tag{41a}$$

$$= \int \left(\int \mathbf{f}_* p(\mathbf{f}_*|X_*,X,\mathbf{y},\theta)d\mathbf{f}_* \right) p(\theta|X,\mathbf{y})d\theta \tag{41b}$$

$$= \int \mathbf{f}_* \left(\int p(\mathbf{f}_*|X_*,X,\mathbf{y},\theta)p(\theta|X,\mathbf{y})d\theta \right) d\mathbf{f}_* \tag{41c}$$

$$= \int \mathbf{f}_* p(\mathbf{f}_*|X_*,X,\mathbf{y})d\mathbf{f}_* = \mathbb{E}[\mathbf{f}_*|X_*,X,\mathbf{y}] \tag{41d}$$

Since the posterior, $p(\theta|X,\mathbf{y})$, is unknown the integral in eq. (41a) can not be evaluated. Even if the posterior was known, the integral would most likely be intractable. Instead the mean of $p(\mathbf{f}_*|X_*,X,\mathbf{y})$ is approximated by the arithmetic/sample method as described in section 5, this require samples from the posterior $p(\theta|X,\mathbf{y})$.

3 Sampling from Posterior

Hamiltonian Markov Chain Monte Carlo (HMCMC) is used to sample from the posterior eq. (35). The Hamiltonian function is defined as:

$$H(\theta,r) := E_U(\theta) + E_K(r) \tag{42}$$

Parameter	**Description**
E_U	Potential Energy
E_K	Kinetic Energy
θ	The weights and biases for M i.e the parameters
r	The momentum

In Hamiltonian dynamics, $H(\theta,r)$ is governed by the partial differential equation eq. (43)

$$\nabla_t r = -\nabla_\theta H \tag{43a}$$

$$\nabla_t \theta = \nabla_r H \tag{43b}$$

The potential energy, E_U, is specified as the negative log posterior (excluding constants):

$$-\log p(\theta|X,\mathbf{y}) \propto E_U(\theta) = \frac{\theta^T\theta}{\sigma_\theta^2} + \frac{U^TU}{\sigma_M^2} + \mathbf{y}^T(K+\sigma_n^2 I)^{-1}\mathbf{y} + \log|K+\sigma_n^2 I| \tag{44}$$

Often, in practice, the momentum, r, is chosen to follow a multivariate normal distribution, $r \sim \mathcal{N}(0, C)$ [7]. The kinetic energy, E_K, then become [7] [3]:

$$E_K(r) \propto \frac{1}{2} r^T C^{-1} r \tag{45}$$

we let $C = I$. The Leapfrog steps in eq. (46) are used to simulate the Hamiltonian dynamics.

$$\begin{aligned} r_{t+\epsilon/2} &= r_t + \frac{\epsilon}{2}\nabla_t r = r_t - \frac{\epsilon}{2}\nabla_\theta H(\theta_t, r_t) = r_t - \frac{\epsilon}{2}\nabla_\theta E_U(\theta_t) \\ \theta_{t+\epsilon} &= \theta_t + \nabla_t \theta = \theta_t + \epsilon \nabla_r H(\theta_t, r_{t+\epsilon/2}) = \theta_t + \epsilon r_{t+\epsilon/2} \\ r_{t+\epsilon} &= r_{t+\epsilon/2} + \frac{\epsilon}{2}\nabla_{t+\epsilon/2} r = r_{t+\epsilon/2} - \frac{\epsilon}{2}\nabla_\theta H(\theta_{t+\epsilon}, r_{t+\epsilon/2}) = r_{t+\epsilon/2} - \frac{\epsilon}{2}\nabla_\theta E_U(\theta_{t+\epsilon/2}) \end{aligned} \tag{46}$$

The leapfrog steps moves the parameters, θ, towards more probability dense locations, as a result increasing acceptance probability. The acceptance probability, α, is:

$$\alpha = e^{H(\theta_n, r_n) - H(\theta', r')} \tag{47}$$

where θ_n is the current parameters, and θ' is the proposed parameters. θ' is found by performing L leapfrog steps. We conclude this section with the pseudo-code for HMCMC and $\nabla E_U \theta$ which is seen in algorithm 1 and 2 respectively. **Note:** Solving eq. (50) is a relatively heavy computational task and does not scale well with N. Furthermore, θ' change at each leapfrog and hence the kernel matrix, K, also change. we, therefore, have to solve eq. (50) at each Leapfrog iteration, accounting to $T \cdot L$ times in total.

4 Choosing Step-size, ϵ

Given some position θ, where the probability density around θ is high, should we: stay near θ, or move further away, with the opportunity to find higher/lower probabilities. This introduces an exploration-exploitation trade-off when sampling. The parameters that control this trade-off is ϵ, L and C. We focus on the step-size ϵ. [1] propose a cyclic step-size following eq. (52). Example trajectories of eq. (52) is shown in fig. 5. [1] also propose to have $\beta \in (0, 1)$ proportion of each cycle dedicated only to exploration, and no sampling (exploitation), this give the condition in algorithm 3.

$$\epsilon_n = \frac{\epsilon_0}{2}\left(\cos\left(\frac{\pi \text{mod}(n-1, \lfloor T/P \rfloor)}{\lfloor T/P \rfloor}\right) + 1\right) \tag{52}$$

Algorithm 1: Pseudo Code: HMCMC sampling

Result: s samples from the posterior eq. (35)
Given, hyper parameters, θ_h, number of iteration, T, step-size ϵ, number of leapfrog steps L, initial sample θ_0.
for i in `range(T)` **do**
  Draw momentum $r_n \sim \mathcal{N}(0, I)$
  set $\theta' \leftarrow \theta_{n-1}$, $r' \leftarrow r_n$
  Leapfrog
  for i in `range(L)` **do**
    $r' = r' - \frac{\epsilon}{2}\nabla E_U(\theta')$
    $\theta' = \theta' + \epsilon r'$
    $r' = r' - \frac{\epsilon}{2}\nabla E_U(\theta')$
  end
  $K_n = \frac{||r_n||_2^2}{2}$, $K' = \frac{||r'||_2^2}{2}$
  $\log \alpha = -E_U(\theta') - K' + E_U(\theta_n) + K_n$
  Draw $u \sim \mathcal{U}(0, 1)$
  if $\texttt{log}(u) < \log \alpha$ **then**
    Sample: $\theta_n \leftarrow \theta'$, $E_U(\theta_n) \leftarrow E_U(\theta')$
  else
    $\theta_n \leftarrow \theta_{n-1}$
  end
end

Parameter	Description
ϵ_0	Initial step-size
T	Number of training iterations
P	Number of cycles

Algorithm 2:

Result: The potential energy $E_U(\theta)$ and the gradient $\nabla E_U(\theta)$
Given, θ, data $(\mathbf{y}, X)$ and hyper parameters θ_h.
Perform forward pass: $U \leftarrow M(X)$
Compute Log M-prior:

$$\log p(\theta) \propto l_1 = -\frac{\theta^T \theta}{\sigma_\theta^2} \tag{48}$$

Compute Log M-likelihood:

$$\log p(U|\theta) \propto l_2 = -\frac{U^T U}{\sigma_M^2} \tag{49}$$

Compute kernel matrix $K = K(U, U)$

$$\text{Solve: } (K + \sigma_n^2)\alpha = \mathbf{y}, \quad \text{using the Cholesky factorization of } K \tag{50}$$

Compute Log Marginal Likelihood:

$$\log p(\mathbf{y}|U) \propto l_3 = -\mathbf{y}^T \alpha - \log |K + \sigma_n^2 I| \tag{51}$$

Compute the potential energy: $E_U = -l_1 - l_2 - l_3$
Use `torch.autograd.grad` to compute $\nabla E_U(\theta)$

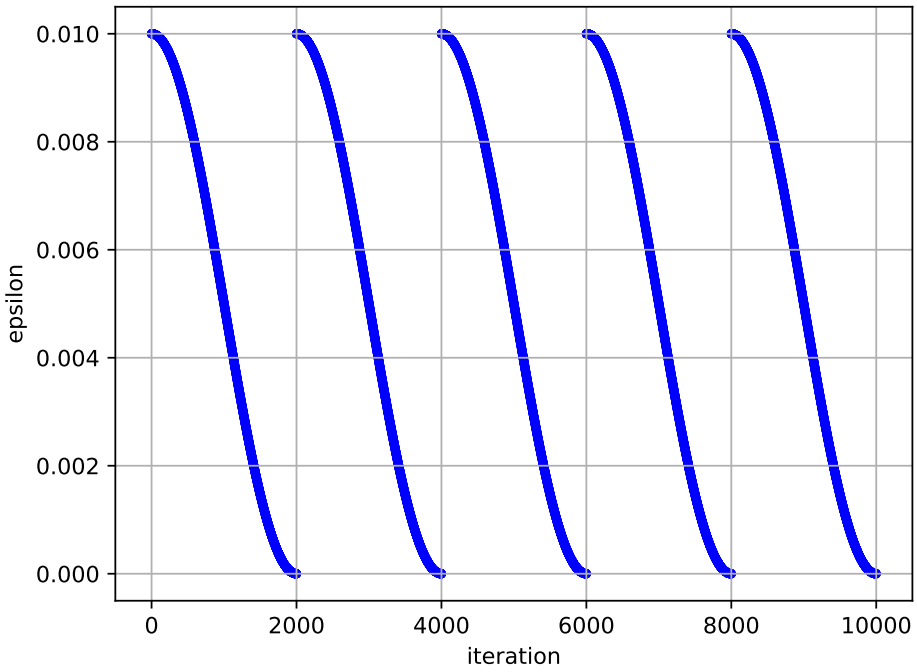


Figure 5: ϵ-trajectory example, generated using eq. (52) with $T = 10000$, $P = 5$, $\epsilon_0 = 0.01$

Algorithm 3: Deciding when to explore and when to exploit (sample)

Input Initial step-size ϵ_0, number of cycles P, training iterations T, proportion of exploration β and current iteration n.
Compute ϵ_n from eq. (52)
Perform, L, Leapfrog steps with $\epsilon \leftarrow \epsilon_n$
if $\underline{\texttt{mod}\big(n-1, \lceil T/P \rceil\big)/\lceil T/P \rceil < \beta}$ **then**
| Exploration Stage. Do not sample
else
| Sample Stage.
end

5 Predicting using HMCMC Samples

Suppose that S samples of θ is drawn from the posterior $p(\theta|X, \mathbf{y})$. The mean and variance of the predictive distribution eq. (21) for each sample, θ_s, are computed using eq. (53)

$$\bar{\mathbf{f}}_*^{(s)} := \mathbb{E}[\mathbf{f}_*|X_*, X, \mathbf{y}, \theta_s] = K_*^{(s)}(K^{(s)} + \sigma_n^2 I)^{-1}\mathbf{y} \tag{53a}$$

$$\text{cov}[\mathbf{f}_*^{(s)}|X_*, X, \mathbf{y}, \theta_s] = K_{**}^{(s)} - K_*^{(s)}(K^{(s)} + \sigma_n^2)^{-1}(K_*^{(s)})^T \tag{53b}$$

$$\omega[\mathbf{f}_*^{(s)}] := \texttt{diag}(\text{cov}[\mathbf{f}_*^{(s)}|X_*, X, \mathbf{y}, \theta_s]) \tag{53c}$$

Parameter	Description
θ_s	Each parameter sample
X_*	$N_* = 200$ equidistant point in the interval $\mathcal{I}_* = \{x \in \mathbb{R} \mid -5 \leq x \leq 5\}$
$M(X_*; \theta_s) = U_*^{(s)}$	Predictive u-space for sample θ_s
$K_*^{(s)}$	SE kernel matrix for $(U_*^{(s)}, U^{(s)})$
$K_{**}^{(s)}$	SE kernel matrix for $(U_*^{(s)}, U_*^{(s)})$

The approximation of $\mathbb{E}[\mathbf{f}_*|X_*, X, \mathbf{y}]$ is calculated as the arithmetic mean in eq. (54).

$$\mathbb{E}[\mathbf{f}_*|X_*, X, \mathbf{y}] \approx \frac{1}{S}\sum_{s=1}^{S}\mathbb{E}[\mathbf{f}_*|X_*, X, \mathbf{y}, \theta_s] = \sum_{s=1}^{S}\bar{\mathbf{f}}_*^{(s)} =: \bar{\mathbf{f}}_* \tag{54a}$$

$$\tag{54b}$$

From the samples $\{\mathbf{f}_*^{(s)}\}_{s=1}^S$ and mean $\bar{\mathbf{f}}_*$ we calculate the sample variance $V[\mathbf{f}_*]$:

$$V[\mathbf{f}_*] = \frac{1}{S}\sum_{s=1}^{S}(\bar{\mathbf{f}}_*^{(s)} - \bar{\mathbf{f}}_*)^2 \tag{55}$$

For large S we get the 95% Confidence Interval 1 (CI1), of $y(X_*)$:

$$\mathbb{E}[y(X_*)|X_*.X,\mathbf{y}|] \pm \sqrt{\mathbb{V}[y(X_*)|X_*,X,\mathbf{y}]} \approx \bar{\mathbf{f}}_* \pm 1.96\sqrt{V[\mathbf{f}_*] + \sigma_n^2)} =: \text{CI1} \tag{56}$$

The sample variance $V[\mathbf{f}_*]$ explains the uncertainty in the mapping M well. However it does not include the variance $\text{cov}[\mathbf{f}_*^{(s)}]$ which can be calculated exactly for each sample θ_s using eq. (53b). Exactly how to include $\text{cov}[\mathbf{f}_*^{(s)}]$ in the CI is yet to be decided, as of now we compare CI1 in eq. (56) with the confidence Interval 2 (CI2) in eq. (57).

$$\text{CI2} := \frac{1}{S}\sum_{s=1}^{S} \mathbf{f}_*^{(s)} \pm 1.96\sqrt{\omega[\mathbf{f}_*^{(s)}] \pm \sigma_n^2} \tag{57}$$

To consider the uncertainty in u-space, we can not use the transformation M. Since infinitely many transformations M can result in the same fit. As an example consider the translated transformation $M_1(x) = M(x) + a$, then M_1 will generate the same kernel as M, due to the stationarity of the SE kernel as shown in eq. (58).

$$k(M_1(x), M_1(x')) = e^{-\frac{1}{2l}||M_1(x) - M_(x')||_2^2} = e^{-\frac{1}{2l}||M(x)+a-(M(x')+a)||_2^2} = k(M(x), M(x)) \tag{58}$$

Instead we describe the uncertainty of the kernels generated from each sample $M_s(X_*, \theta_s) = U_*^{(s)}$, by computing the sample mean and -variance kernel, $\bar{K}_{**}$, $V[K_{**}]$ respectively eq. (59).

$$\bar{K}_{**} = \frac{1}{S}\sum_{s=1}^{S} K_{**}^{(s)} \tag{59a}$$

$$V[K_{**}] = \frac{1}{S}\sum_{s=1}^{S} (K_{**}^{(s)} - \bar{K}_{**})^2 \tag{59b}$$

Note: due to eq. (58) we do not use bias parameters in the last layer of M since they will be cancelled when calculating $||u - u'||_2^2$.

6 Experimental Results

This section evaluate the model (BNN-GP) on two function; section 6.1 a step function, section 6.2 a function with a pulse in each end.
Note: when initiating the HMCMC algorithm, we perform a warm start by letting θ_0 be the result of approximating M with a deterministic neural network minimising the Negative Marginal Likelihood, using the work described in [5].

6.1 Step Function

First, the model BNN-GP is evaluated on the step function $y : \mathbb{R} \to \mathbb{R}$ given in eq. (60).

$$y(x) = f(x) + \eta \tag{60}$$

$$f(x) = \begin{cases} 1, & x < t \\ -1, & x \geq t \end{cases} \tag{61}$$

$$\eta \sim \mathcal{N}(0, \sigma_n^2) \tag{62}$$

Parameter	Description
t	Threshold for the step function

In order to simulate uncertainty about the threshold, t, we generate an x-space which has no information around t. Let $t = 0$. Generate $N = 200$ equidistant points in the interval, $\mathcal{I}$ for $\delta = 1$ and store them in X. Sample $\mathbf{y} = y(X)$ for $\sigma_n^2 = 0.0005$. Generate $N_* = 200$ equidistant test points in the interval $\mathcal{I}_* = \{x \in \mathbb{R} | -5 \leq x \leq 5\}$ and store them in X_*.

$$\mathcal{I} = \{x \in \mathbb{R} | -5 \leq x \leq -\delta\} \cup \{x \in \mathbb{R} | \delta \leq x \leq 5\},\ \delta \in [0, 5] \tag{63a}$$

$$\Delta = 2\delta = \text{Length of zero-information interval} \tag{63b}$$

The hyper-parameters for the $BNN - GP$ model are given in eq. (64):

$$\theta_{H1} = \{\sigma_f^2 = 1,\ l = 1,\ \sigma_n^2 = 0.0005 \tag{64}$$
$$A = \text{one layer with 32 neurons + bias in hidden layer. } \tanh(.) \text{ as activation function}\}$$

We run the HMCMC algorithm 1 with the following parameters:

$$H1 = \{T = 20000, P = 400, \beta = 0.2, L = 10, \epsilon_0 = 0.0008\} \tag{65}$$

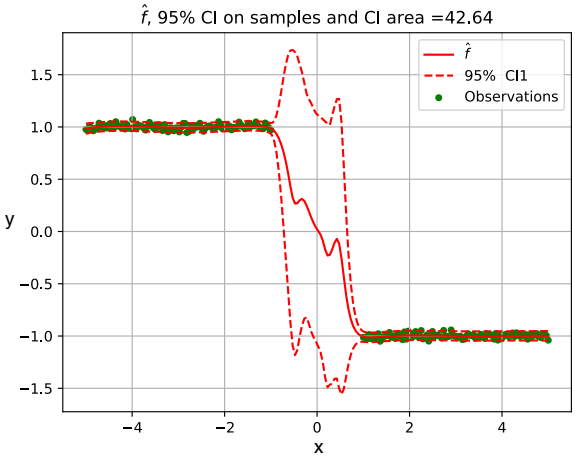
which results in 12711 samples from the posterior $p(\theta|X, \mathbf{y})$. In fig. 8b each $U_*^{(s)} = M(X_*; \theta_s)$ is shown. The samples $U_*^{(s)}$ does not look particular similar, one can obtain smoother/nicer samples by lowering ϵ_0, however the uncertainty will also decrease in $\mathbf{f}$-space (the sample parameters $H1$ might need extra tuning).
The estimate of the prediction, $\bar{\mathbf{f}}_*|X_*, X, \mathbf{y}$, is shown together with CI1 in fig. 8a and with CI2 in fig. 8c. Three observations is made when comparing CI1 with CI2 in fig. 8a and fig. 8c respectively:

1. CI1 increase the uncertainty faster, when we leave $\mathcal{I}$. CI2 maintain low uncertainty despite moving away from $\mathcal{I}$.

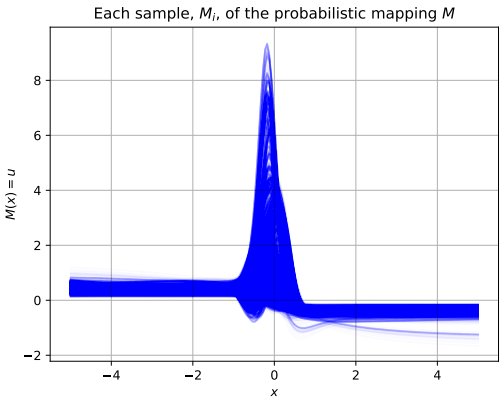
2. In the neighbourhood of $x = 0$, CI1 have a more jagged/uneven confidence interval, where CI2 is more smooth.

3. CI2 explain the variance more natural than CI1 in the region information $\mathcal{I}$.

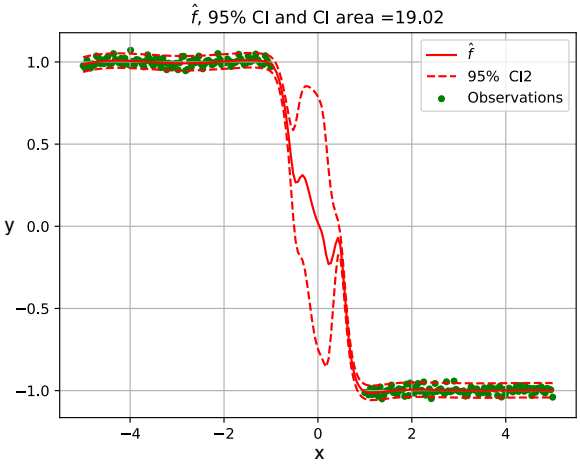
The mean and variance kernel $\bar{K}_{**}$, $V[K_*]$, is shown in fig. 8e and fig. 8d respectively. The "green plus" seen in fig. 8d show the uncertainty around $t = 0$.
We perform a similar fit for $\delta = 2$ and show the results in fig. 7. The model correctly increases the uncertainty around the threshold when increasing δ from 1 to 2. As seen from the increase in CI area between fig. 6 and fig. 7 similar, the "green plus" in the variance kernel also increases its width when δ increase, as expected.



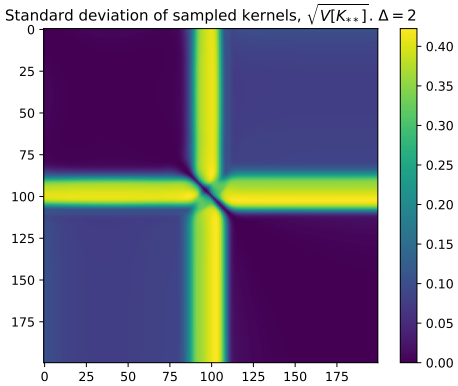
(a) BNN GP fit with Confidence Interval 1 calculated using eq. (56). CI area is the upper and lower CI subtracted and divided by $\Delta = 2$



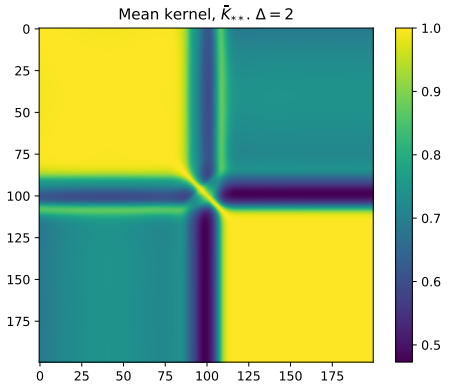
(b) All 12711 latent space samples, $U_*^{(s)} = M(X_*, \theta_s)$



(c) BNN GP fit with Confidence Interval 2 calculated using eq. (57). CI area is the upper and lower CI subtracted and divided by $\Delta = 2$

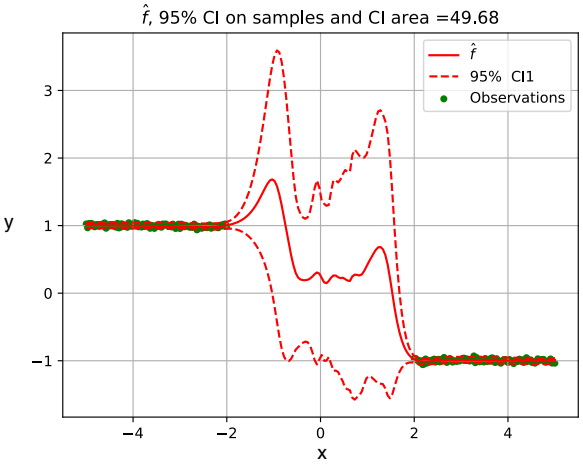


(d) Square root of the arithmetic variance kernel $\sqrt{V[K_{**}]}$ calculated using eq. (59).

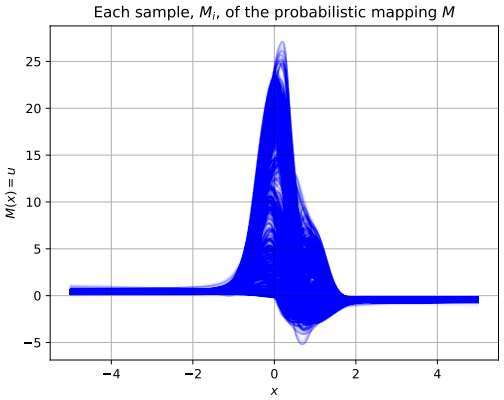


(e) Mean kernel $\bar{K}_{**}$ calculated using eq. (59).

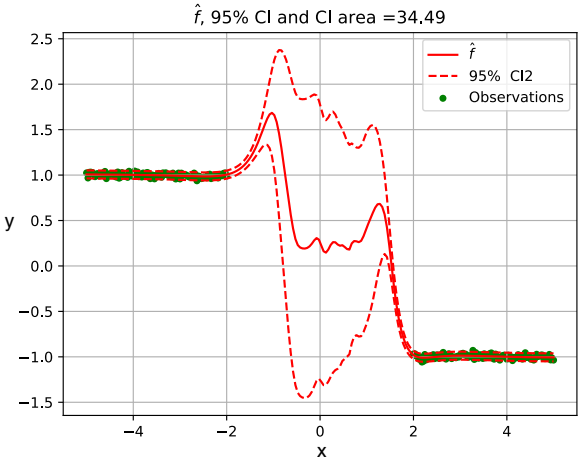
Figure 6: Fitting the model *eq.* (27) to data generated from the function eq. (60) with $\delta = 1$. Using hyper-parameters θ_{H1}. Sampling parameters $H1 = \{T = 20000, M = 400, \beta = 0.2, L = 10, \epsilon_0 = 0.0008\}$



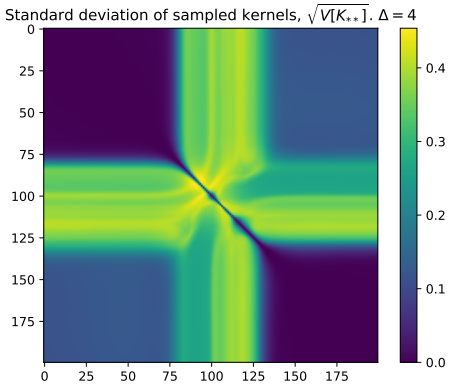
(a) BNN GP fit with Confidence Interval 1 calculated using eq. (56). CI area is the upper and lower CI subtracted and divided by $\Delta = 4$



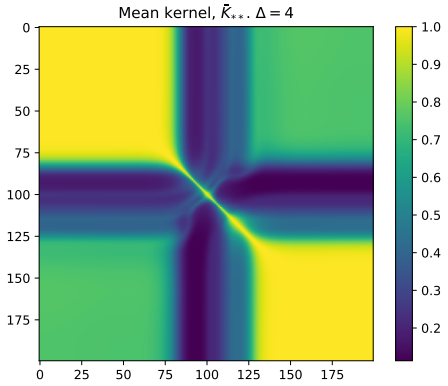
(b) Latent space samples, $U_*^{(s)} = M(X_*, \theta_s)$



(c) BNN GP fit with Confidence Interval 2 calculated using eq. (57). CI area is the upper and lower CI subtracted and divided by $\Delta = 4$



(d) Square root of the arithmetic variance kernel $\sqrt{V[K_{**}]}$ calculated using eq. (59).



(e) Mean kernel $\bar{K}_{**}$ calculated using eq. (59).

Figure 7: Fitting the model *eq.* (27) to data generated from the function eq. (60) with $\delta = 2$. Using hyper-parameters θ_{H1}. Sampling parameters $H1 = \{T = 20000, M = 400, \beta = 0.2, L = 10, \epsilon_0 = 0.0008\}$

6.2 Wall Pulse

Until now the M-likelihood $p(U|\theta, X)$ was chosen as the normal distribution:

$$p(U|\theta, X) \sim \mathcal{N}(\mathbf{0}, \sigma_M^2 I) \tag{66}$$

The zero-mean choice is arbitrary. In this section we experiment by assigning $\mathbb{E}[U|\theta, X]$ the warm start solution, $U_0 = M(\theta_0)$, (deterministic neural network). $\mathbb{E}[U|\theta, X] = U_0$. We test this on the function eq. (67).

$$\begin{aligned} &y(x) = f(x) + \eta \\ &f(x) = \begin{cases} \text{Re}(\text{e}^{-a\cdot(2x)^2} e^{2\pi f_c(2x)\cdot i}), & 0 \leq x \leq 5 \\ \text{Re}(\text{e}^{-a\cdot(2x)^2} e^{-2\pi f_c(2x)\cdot i}), & \text{-5} \leq x \leq 0 \end{cases} \\ &\eta \sim \mathcal{N}(0, \sigma_n^2) \end{aligned} \tag{67}$$

We generate $\mathbf{y} = y(X)$ where X is $N = 200$ equidistant points in the interval $\mathcal{I}$ for $\delta = 1$.

$$\begin{aligned} &\mathcal{I} = \{x \in \mathbb{R} | -5 \leq x \leq -\delta\} \cup \{x \in \mathbb{R} | \delta \leq x \leq 5\},\ \delta \in [0, 5] \\ &\Delta = 2\delta = \text{Length of zero-information interval} \end{aligned} \tag{68}$$

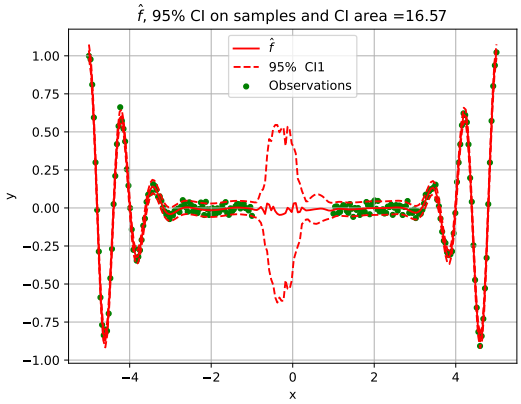
The hyper-parameters for the $BNN - GP$ model are given in

$$\theta_{H1} = \{\sigma_f^2 = 1,\ l = 1,\ \sigma_n^2 = 0.0005 \tag{69a}$$
$$A = \text{one layer with 32 neurons + bias in hidden layer. tanh(.) as activation function}\}$$

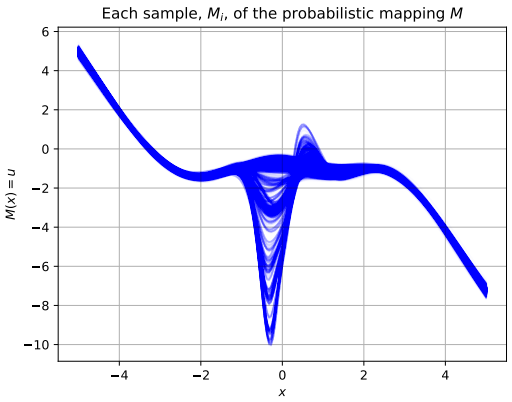
We run HMCMC algorithm 1 with the following parameters:

$$H2 = \{T = 10000, P = 200, \beta = 0.2, L = 10, \epsilon_0 = 0.001\} \tag{70}$$

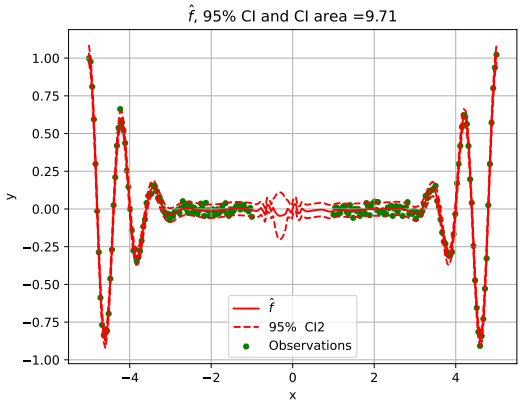
The results are shown in fig. 8



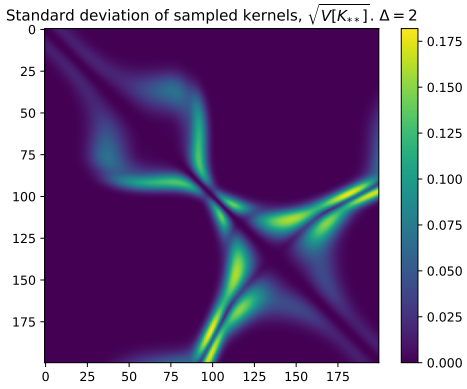
(a) BNN GP fit with Confidence Interval 1 calculated using eq. (56). CI area is the upper and lower CI subtracted and divided by $\Delta = 2$



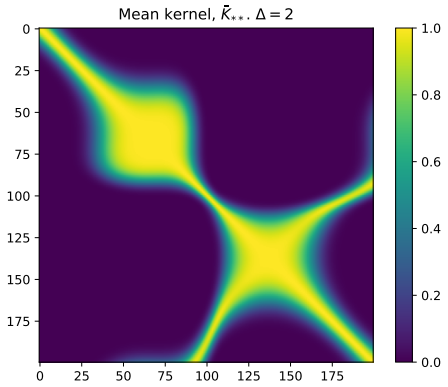
(b) All 12711 latent space samples, $U_*^{(s)} = M(X_*, \theta_s)$



(c) BNN GP fit with Confidence Interval 2 calculated using eq. (57). CI area is the upper and lower CI subtracted and divided by $\Delta = 2$



(d) Square root of the arithmetic variance kernel $\sqrt{V[K_{**}]}$ calculated using eq. (59).



(e) Mean kernel $\bar{K}_{**}$ calculated using eq. (59).

Figure 8: Fitting the model BNN-GP eq. (27) to data generated from the function eq. (67) with $\delta = 1$. Using hyper-parameters θ_{H2}. Sampling parameters $H2 = \{T = 10000, P = 200, \beta = 0.2, L = 10, \epsilon_0 = 0.001\}$

7 Conclusion and Discussion

In this report, a warped input GP with a **probabilistic mapping**, M, was fitted. A BNN was used to model M. The model was able to display uncertainty in regions of low/no information. Precisely calculating the confidence interval is yet to be determined. The confidence interval was **not symmetric** in the region of low/no information; this was unexpected and undesired. The reason might be the HMCMC parameters (step-size, leapfrog), and further tuning/investigation would be appropriate. The length-scale of the GP kernel was static. An idea could be to assign the length-scale a Gamma distribution with low variance as a prior and then include the length-scale in the warm start, and let this warm start be the mean of the Gamma distribution. Computation complexity has not been considered in this report, but it has been stated that the algorithms used do not scale well in their current form. For the Wall Pulse function in section 6.2 the mean of the M-Likelihood, $p(U|X,\theta)$, was assigned the warm start (deterministic neural network) $\mathbb{E}[U|X,\theta] = M(X;\theta_0)$, this M-Likelihood was also evaluated on the step-function, it showed promising results, and should be included in this report.

References

[1] Ruqi Zhang et al. "Cyclical Stochastic Gradient MCMC for Bayesian Deep Learning". In: ICLR (2019). URL: https://arxiv.org/pdf/1902.03932.pdf.

[2] Mark N. Gibbs. "Bayesian Gaussian Pro cesses for Regression and Classification". In: University of Cambridge (1997). URL: http://www.inference.org.uk/mng10/GP/thesis.ps.

[3] Neal M. Radford. MCMC using Hamiltonian dynamics. Chapman & Hall, 2010.

[4] Carl Edward Rasmussen and Christopher K. I. Williams. Gaussian Processes for Machine Learning. MIT Press, 2006.

[5] Carl E. Rasmussen Roberto Calandra Jan Peters and Marc P. Deisenroth. "Manifold Gaussian Processes". In: IEEE (2016). URL: https://arxiv.org/pdf/1402.5876.pdf.

[6] Sergios Theodoridis. Machine Learning A Bayesian and Optimization Perspective. Elsevier, 2015.

[7] David Frich Hansen Master Thesis. "Stochastic gradient Markov chain Monte Carlo". In: Danish Technical University (2020).