Sparse Spectrum Warped Input Measures for Nonstationary Kernel Learning

Anthony Tompkins[1,*] **Rafael Oliveira**[1,2,*] **Fabio Ramos**[1,3]

[1] School of Computer Science, the University of Sydney, Australia
[2] ARC Centre for Data Analytics for Resources and Environments, Australia
[3] NVIDIA, USA

Abstract

We establish a general form of explicit, input-dependent, measure-valued warpings for learning nonstationary kernels. While stationary kernels are ubiquitous and simple to use, they struggle to adapt to functions that vary in smoothness with respect to the input. The proposed learning algorithm warps inputs as conditional Gaussian measures that control the smoothness of a standard stationary kernel. This construction allows us to capture non-stationary patterns in the data and provides intuitive inductive bias. The resulting method is based on sparse spectrum Gaussian processes, enabling closed-form solutions, and is extensible to a stacked construction to capture more complex patterns. The method is extensively validated alongside related algorithms on synthetic and real world datasets. We demonstrate a remarkable efficiency in the number of parameters of the warping functions in learning problems with both small and large data regimes.

1 Introduction

Many interesting real world phenomena exhibit varying characteristics, such as smoothness, across their domain. Simpler phenomena that *do not* exhibit such variation may be called *stationary*. The typical kernel based learner canonically relies on a *stationary* kernel function, a measure of "similarity", to define the prior beliefs over the function space. Such a kernel, however, cannot represent desirable *nonstationary* nuances, like varying spatial smoothness and sudden discontinuities. Restrictive stationary assumptions do not generally hold and limit applicability to interesting problems, such as robotic control and reinforcement learning [1], spatial mapping [2], genetics [3], and Bayesian optimisation [4]. One obvious way to alleviate the problem of finding the appropriate kernel function given one's data is hyperparameter optimisation. However for a GP with stationary kernel, even if the *optimal* set of hyperparameters were found, it would be insufficient if our underlying response were nonstationary with respect to the observed inputs.

In this paper we propose a method for nonstationary kernel learning, based on sparse spectral kernel representations. Our method is linear in complexity with respect to the number of data points and is simultaneously able to extract nonstationary patterns. In our setup, we consider the problem of learning a function $f : \mathcal{X} \to \mathbb{R}$ as a nonstationary Gaussian process. We decompose f as:

$$f(\mathbf{x}) = \mathbb{E}[u \circ \mathbf{m}(\mathbf{x}) | u] \,, \quad \mathbf{x} \in \mathcal{X} \,, \tag{1}$$

where $\circ$ denotes function composition, $u : \mathcal{Q} \to \mathbb{R}$ is a function over a latent space $\mathcal{Q}$, and $\mathbf{m}(\mathbf{x})$ represents the warped input. If u has covariance function k_u, the resulting f follows a GP with covariance $k_f(\mathbf{x}, \mathbf{x}') = \mathbb{E}[k_u(\mathbf{m}(\mathbf{x}), \mathbf{m}(\mathbf{x}'))]$. The latter constitutes a nonstationary kernel.

[*] Equal contribution

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

To model u as a stationary GP on $\mathcal{Q}$, we propose a formulation for $\mathbf{m} : \mathcal{X} \to \mathcal{Q}$, which is based on a locally affine stochastic transform:

$$\mathbf{m}(\mathbf{x}) = \mathbf{G}(\mathbf{x})\mathbf{x} + \mathbf{h}(\mathbf{x}) \, , \tag{2}$$

where $\mathbf{G}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ are Gaussian processes. Intuitively, the matrix $\mathbf{G}$ scales the inputs, with a similar effect to what length-scales have on stationary kernels [5], but which now varies across the space, while $\mathbf{h}$ allows for arbitrary shifts.

The conditional expectation (1) also corresponds to the composition of a function on $\mathcal{Q}$ with a measure [6] on $\mathcal{Q}$, which is actually a function of $\mathbf{x} \in \mathcal{X}$. In our case, the measure-valued warpings are Gaussian probability measures, which we parametrise as Gaussian process conditioned on *pseudo-training points*. In particular, we use sparse spectrum Gaussian processes [7] due to their scalability and availability of closed-form results for Gaussian inputs [8].

1.1 Contributions

- We propose a new method to learn nonstationary Gaussian process models via input warping. We introduce the use of a measure-valued, self-supervised and input-dependent warping function as a natural improvement for sparse spectrum Gaussian processes. We term this *sparse spectrum warped input measures* (SSWIM);
- We propose a self-supervised training scheme for representing the warping function allowing us to cleanly represent the latent measure valued warping; and
- We propose a simple extension to multiple levels of warping by propagating moments.

1.2 Related work

Foundational work [9, 10] on kernel based nonstationarity necessitated manipulation of the kernel function with expensive inference procedures. Recent spectral representation of kernel functions have emerged with Bochner's theorem [11]. In this paradigm, one constructs kernels in the Fourier domain via *random Fourier features* (RFFs) [12, 13] and extensions for nonstationarity via the generalised Fourier inverse transform [14, 15, 2, 16]. While general, these methods suffer from various drawbacks such as expensive computations and overfitting due to over-parameterised models [2].

More expressive modelling frameworks [17, 18, 19, 20] have played a major role in expanding the efficacy of kernel based learning. Perhaps the most well known in the recent literature is Deep Kernel Learning Wilson et al. [21] and the *deep Gaussian process* [22] and heretofore its various extensions [23, 24, 25]. While functionally elegant, methods like DKL and DGP often rely on increasing the complexity of the composition to produce expressiveness and are often unsuitable or unwieldy in practice occasionally resulting in performance worse than stationary inducing point GPs [23]. We remark a notable difference between DGP and SSWIM is one should interpret our pseudo-training points as hyperparameters of the kernel as opposed to parameters of a variational approximation.

Simple bijective input warpings were considered in Snoek et al. [26] for transforming nonstationary functions into more well behaved functions. In Heinonen et al. [27] the authors augment the standard GP model by learning nonstationary data-dependent functions for the *hyperparameters* of a nonstationary squared-exponential kernel [28]. Their method, however, is limited to low dimensions. More recently, the work of Hegde et al. [29] has explored a dynamical systems view of input warpings by processing the inputs through time-dependent differential fields. Less related models presented in Wang and Neal [30], Dutordoir et al. [31], Snelson et al. [32] involve *output warping* non-Gaussian likelihoods and heteroscedastic noise. For the curious reader we examine contrasting properties of output and input warping in the supplementary material.

2 Sparse spectrum Gaussian processes

We start by reviewing relevant background with regards to kernel methods for Gaussian process regression. In particular, we focus on the sparse spectrum approximation to GPs [7], which we use to formulate nonstationary kernels.

Gaussian processes. Suppose our goal is to learn a function $f : \mathbb{R}^D \to \mathbb{R}$ given IID data $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, with each data pair related through

$$y = f(\mathbf{x}) + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma_n^2), \tag{3}$$

where ϵ is IID additive Gaussian noise. A Gaussian process is a distribution on functions f over an input space $\mathcal{X} \subseteq \mathbb{R}^D$ such that any finite set of inputs $\mathbf{x}_1, ..., \mathbf{x}_N \in \mathcal{X}$ produces a multivariate normal distribution of response variables $\mathbf{f}_N := [f(\mathbf{x}_1), ..., f(\mathbf{x}_N)]^\mathsf{T}$:

$$\mathbf{f}_N \sim \mathcal{N}(\mathbf{m}_N, \mathbf{K}_N)\ , \tag{4}$$

where $\mathbf{m}_N = m(\mathbf{x}_1, ..., \mathbf{x}_N)$ is the mean vector, and $\mathbf{K}_N = \{k(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j}$ with kernel k.

Approximate GP in feature space. Full GP inference is a challenging problem naively occurring in $\mathcal{O}(N^3)$ complexity as a consequence of having to invert an (N, N) Gram matrix. An alternative perspective on approximate GP inference is to consider the *feature space* view of the kernel function using Bochner's theorem [11]. Under this view, *random Fourier features* [12, 13] decompose the kernel function in terms of Fourier features based on a finite approximation to the kernel's spectrum.

As presented by Rahimi and Recht [12], the Fourier transform of any shift-invariant positive-definite kernel $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ yields a valid probability distribution p_k, so that k is approximately:

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}_{\boldsymbol{\omega} \sim p_k}[\cos(\boldsymbol{\omega}^\mathsf{T}(\mathbf{x} - \mathbf{x}'))] \approx \boldsymbol{\phi}(\mathbf{x})^\mathsf{T}\boldsymbol{\phi}(\mathbf{x}')\,, \tag{5}$$

where $\boldsymbol{\phi}$ corresponds to the approximate feature map:

$$\boldsymbol{\phi}(\mathbf{x}) = \sqrt{\frac{2}{M}}\left[\cos\left(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}\right), \ldots, \cos\left(\boldsymbol{\omega}_M^\mathsf{T}\mathbf{x}\right), \sin\left(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}\right), \ldots, \sin\left(\boldsymbol{\omega}_M^\mathsf{T}\mathbf{x}\right)\right] \in \mathbb{R}^{2M}. \tag{6}$$

Using the feature map above we are able to define a GP termed the Sparse Spectrum Gaussian Process (SSGP) [7]. Due to feature map (6), the SSGP is a Gaussian distribution over the feature weights $\mathbf{w} \in \mathbb{R}^{2M}$. If we assume the weight prior is $\mathcal{N}(\mathbf{0}, \mathbf{I})$, after conditioning on data $\mathcal{D}$ the posterior distribution of $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\alpha},\, \sigma_n^2\mathbf{A}^{-1})$, where:

$$\boldsymbol{\alpha} = \mathbf{A}^{-1}\boldsymbol{\Phi}\mathbf{y}, \tag{7}$$

$$\mathbf{A} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\mathsf{T} + \sigma_n^2\mathbf{I}, \tag{8}$$

following from Bayesian Linear Regression [33]. The design matrix $\boldsymbol{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1), ..., \boldsymbol{\phi}(\mathbf{x}_N)]$ and column vector $\mathbf{y} = [y_1, ..., y_N]^\mathsf{T}$ are given directly by the data $\mathcal{D}$. The posterior distribution over the response y given an $\mathbf{x}$ is exactly Gaussian:

$$p(f(\mathbf{x})|\mathbf{x}) = \mathcal{N}\left(\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{\phi}(x), \sigma_n^2\|\boldsymbol{\phi}(x)\|_{\mathbf{A}^{-1}}^2\right), \tag{9}$$

where we define $\|\mathbf{v}\|_{\boldsymbol{\Sigma}}^2 := \mathbf{v}^\mathsf{T}\boldsymbol{\Sigma}\mathbf{v}$. Multivariate outputs can be modelled as conditionally independent GPs for each output dimension or jointly by encoding the covariance between the outputs as a vector-valued GP [34].

Kernels with Gaussian inputs. In our formulation of non-stationary kernel, we take form the kernel based on expectations with respect to distributions conditioned on the inputs. In the sparse-spectrum formulation, the expected kernel is simply the result of the inner product between the expected feature map of each input, due to the linearity of expectations. For the case of Gaussian inputs, results by Pan et al. [8] then allow us to compute the expected feature map in closed form. For a Gaussian input $\tilde{\mathbf{x}} \sim \mathcal{N}(\hat{\mathbf{x}}, \boldsymbol{\Sigma})$, we have:

$$\mathbb{E}[\cos(\boldsymbol{\omega}^\mathsf{T}\tilde{\mathbf{x}})] = \exp\left(-\frac{1}{2}\|\boldsymbol{\omega}\|_{\boldsymbol{\Sigma}}^2\right)\cos(\boldsymbol{\omega}^\mathsf{T}\hat{\mathbf{x}})\ , \tag{10}$$

$$\mathbb{E}[\sin(\boldsymbol{\omega}^\mathsf{T}\tilde{\mathbf{x}})] = \exp\left(-\frac{1}{2}\|\boldsymbol{\omega}\|_{\boldsymbol{\Sigma}}^2\right)\sin(\boldsymbol{\omega}^\mathsf{T}\hat{\mathbf{x}}). \tag{11}$$

What is important to note here is the exponential constant which scales the standard feature by a value proportional to the uncertainty in the warped input. That is to say, expectations that take on larger (predictive) uncertainties will be *smaller* than if we did not take this uncertainty into account.

Algorithm 1 Sparse Spectrum Warped Input Measures

```
Input: ${\mathbf{X}, \mathbf{y}}$
Output: $\boldsymbol{\theta} = \{\boldsymbol{\theta}_u, \boldsymbol{\theta}_{\mathbf{g}}, \boldsymbol{\theta}_{\mathbf{h}}, \mathbf{X}_{\mathbf{g}}, \mathbf{Y}_{\mathbf{g}}, \mathbf{X}_{\mathbf{h}}, \mathbf{Y}_{\mathbf{h}}\}$
Initialize pseudo-training points $\{\mathbf{X}_{\mathbf{g}}, \mathbf{Y}_{\mathbf{g}}\}$, $\{\mathbf{X}_{\mathbf{h}}, \mathbf{Y}_{\mathbf{h}}\}$
for $t \in \{1, \ldots, T\}$ do
    Fit $\mathbf{g}$ and $\mathbf{h}$ to $\{\mathbf{X}_{\mathbf{g}}, \mathbf{Y}_{\mathbf{g}}\}$, $\{\mathbf{X}_{\mathbf{h}}, \mathbf{Y}_{\mathbf{h}}\}$
    Compute $\hat{\mathbf{m}}$ and $\boldsymbol{\Sigma}_{\mathbf{m}}$ for $\mathbf{X}$
    Fit $u$ using expected feature map
    Calculate $\log p(\mathbf{y}|\boldsymbol{\theta})$
    Update gradients and take new step.
end for
```

3 Sparse spectrum warped input measures

In this section we introduce the main contribution of the paper: *sparse spectrum warped input measures* (SSWIM). The key idea in our work is based on two crucial steps. First, we construct a stochastic vector-valued mapping modelling the input warping $\mathbf{m} : \mathcal{X} \to \mathcal{Q}$, where $\mathcal{X} \subseteq \mathbb{R}^D$ represents the raw input space and $\mathcal{Q}$ is the resulting warped space. A top-level GP modelling $u : \mathcal{Q} \to \mathbb{R}$ then estimates the output of the regression function $f : \mathcal{X} \to \mathbb{R}$. To learn the warping, each lower-level SSGP is provided with *pseudo-training* points, which are learned jointly with the remaining hyper-parameters of both GP models.

It is important to note that the pseudo-training points are *free parameters* of the latent warping function and therefore *hyperparameters of the top-level function*. Furthermore, while our construction and implementation assumes a pseudo-training dimensionality equal to that of the original data $\dim \mathcal{X} = \dim \mathcal{Q}$, nothing preventing us from embedding the input warping into a lower $\dim \mathcal{Q} \ll \dim \mathcal{X}$ or higher $\dim \mathcal{Q} \gg \dim \mathcal{X}$ dimensional manifold.

3.1 Warped input measures

To model and propagate the uncertainty on the warping operator $\mathbf{G}$ through the predictions, we start by modelling $\mathbf{G} : \mathcal{X} \to \mathcal{L}(\mathcal{X})$ as a Gaussian process. Then every linear operation on $\mathbf{G}$ results in another GP [35], so that $\mathbf{m}(\mathbf{x}) = \mathbf{G}(\mathbf{x})\mathbf{x} + \mathbf{h}(\mathbf{x})$, for a deterministic $\mathbf{x} \in \mathcal{X}$, is Gaussian. Similarly, as expectations constitute linear operations, the *expected value* of the GP u under the random input given by the warping is also Gaussian [36]. Marginalising $\mathbf{m}$ out of the predictions, i.e. inferring the expected value of f under the distribution of $\mathbf{m}$, $\hat{f}(\mathbf{x}) = \mathbb{E}[u \circ \mathbf{m}(\mathbf{x})|u]$, we end up with a final GP, which has analytic solutions.

From Section 2, the uncertain-inputs predictions from $\hat{u} = \mathbb{E}[u(\tilde{\mathbf{x}})|u]$ for $\mathbf{m}(\mathbf{x}) \sim \mathcal{N}(\hat{\mathbf{m}}(\mathbf{x}), \boldsymbol{\Sigma}_{\mathbf{m}}(\mathbf{x}))$ are given by the SSGP predictive equations in (9) using the expected feature map for $\mathbb{E}[\boldsymbol{\phi}(\mathbf{m}(\mathbf{x}))]$. Equation 11 then allows us to compute $\mathbb{E}[\boldsymbol{\phi}(\mathbf{m}(\mathbf{x}))]$ in closed form for a given mean $\hat{\mathbf{m}}(\mathbf{x})$ and covariance matrix $\boldsymbol{\Sigma}_{\mathbf{m}}(\mathbf{x})$. The general form of the covariance matrix $\boldsymbol{\Sigma}_{\mathbf{m}}(\mathbf{x})$ for $\mathbf{m}(\mathbf{x})$ involves dealing with a fourth order tensor describing the second moment of $\mathbf{G}$. For this paper, however, we consider a particular case with a more elegant formulation and yet flexible enough to accommodate for a large variety of warpings.

Let $\mathbf{G}(\mathbf{x})\mathbf{x} := \mathbf{g}(\mathbf{x}) \odot \mathbf{x}$, where $\odot$ denotes the element-wise product and $\mathbf{g}$ is a vector-valued Gaussian process. This type of warping is equivalent to $\mathbf{G}(\mathbf{x})$ map to a diagonal matrix. The mean and covariance matrix of the warped input $\mathbf{m}(\mathbf{x})$, can be derived as (see Appendix for details):

$$\hat{\mathbf{m}}(\mathbf{x}) = \hat{\mathbf{g}}(\mathbf{x}) \odot \mathbf{x} + \hat{\mathbf{h}}(\mathbf{x}) \tag{12}$$

$$\boldsymbol{\Sigma}_{\mathbf{m}}(\mathbf{x}) = \mathbf{x}\mathbf{1}^{\mathsf{T}} \odot \boldsymbol{\Sigma}_{\mathbf{g}}(\mathbf{x}) \odot \mathbf{1}\mathbf{x}^{\mathsf{T}} + \boldsymbol{\Sigma}_{\mathbf{h}}(\mathbf{x}) \;, \tag{13}$$

where $\hat{\mathbf{g}}(\mathbf{x})$ and $\boldsymbol{\Sigma}_{\mathbf{g}}(\mathbf{x})$ are the predictive mean and covariance, respectively, of the GP defining $\mathbf{g}$, while $\hat{\mathbf{h}}(\mathbf{x})$ and $\boldsymbol{\Sigma}_{\mathbf{h}}(\mathbf{x})$ are the same for the GP on $\mathbf{h}$.

3.2 Latent self-supervision with pseudo-training

In order to fully specify our latent function, we utilise *pseudo-training* pairs $\{\mathbf{X}_{\mathbf{g}}, \mathbf{Y}_{\mathbf{g}}\}$ and $\{\mathbf{X}_{\mathbf{h}}, \mathbf{Y}_{\mathbf{h}}\}$, somewhat analogous to the well known *inducing-points* framework for sparse Gaussian processes

[37]. Conditioning on these virtual observations allows us to implicitly control the Gaussian measure determined by the warping SSGP.

We model the multiplicative warping $\mathbf{g} : \mathbb{R}^D \to \mathbb{R}^D$ using a standard, multi-output, SSGP that is analytically fit on virtual *pseudo-training* points $\{\mathbf{X_g}, \mathbf{Y_g}\}$. Assuming coordinate-wise output independence, we model $\mathbf{g}$ as $\mathbf{g}(\mathbf{x}) \sim \mathcal{N}(\hat{\mathbf{g}}(\mathbf{x}), \mathbf{\Sigma_g}(\mathbf{x}))$, where:

$$\hat{\mathbf{g}}(\mathbf{x}) = \boldsymbol{\phi}_{\mathbf{g}}(\mathbf{x})^\mathsf{T} \mathbf{A}_{\mathbf{g}}^{-1} \mathbf{\Phi_g} \mathbf{Y_g} \tag{14}$$

$$\mathbf{\Sigma_g}(\mathbf{x}) = \sigma_{n,g}^2 \boldsymbol{\phi}_{\mathbf{g}}(\mathbf{x})^\mathsf{T} \mathbf{A}_{\mathbf{g}}^{-1} \boldsymbol{\phi}_{\mathbf{g}}(\mathbf{x}) \mathbf{I} \,, \tag{15}$$

with $\mathbf{\Phi_g} := \mathbf{\Phi_g}(\mathbf{X_g})$ as the matrix of Fourier features for the pseudo-inputs $\mathbf{X_g}$, and $\mathbf{A_g} = \mathbf{\Phi_g}\mathbf{\Phi_g}^\mathsf{T} + \sigma_{n,g}^2 \mathbf{I}$. The pseudo-inputs $\mathbf{X_g}$ are initially sampled uniformly across the data domain, $\mathbf{X_g} \sim \mathcal{U}(\min(\mathbf{X}), \max(\mathbf{X}))$. The pseudo-training targets $\mathbf{Y_g}$ are initialised $[\mathbf{Y_g}]_{i,j} \sim \mathcal{N}(1, \sigma_\gamma^2)$ where σ_γ^2 mimics a prior variance for the latent warping function. The mean at 1 keeps the initial warping close to identity.

We adopt a similar construction for the GP on the additive component of the warping $\mathbf{h}$. However, we initialise the pseudo-training targets $\mathbf{Y_h}$ with zero-mean values $[\mathbf{Y_h}]_{i,j} \sim \mathcal{N}(0, \sigma_\gamma^2)$, so that we favour a null effect initially. In summary, the complete expected kernel is thus given as:

$$k_f(\mathbf{x}, \mathbf{x}') := \mathbb{E}[\boldsymbol{\phi}(\mathbf{m}(\mathbf{x})]^\mathsf{T} \mathbb{E}[\boldsymbol{\phi}(\mathbf{m}(\mathbf{x}'))] \,, \tag{16}$$

where the expectation is taken over $\mathbf{m}$, whose distribution is recursively defined by equations 12 to 15.

3.3 A layered warping

We have thus far considered a single warping $\mathbf{m}$ of the input $\mathbf{x}$. It is natural to ask: *can we warp the warpings?* A simple way to answer this is to revisit how we interpret a single warping: we are transforming the original input space, with which our response varies in a non-stationary way, to a new space a GP with a stationary kernel can easily represent. We could thus intuit a warping of a warping to mean that we are transforming the first level of warping to a second one to which our response variable is simply *more stationary* than if we had just relied on the first warping alone. We present now an extension to SSWIM which lets us perform this measure value warping of a measure valued warping. Let us begin by defining the J^{th} warping as:

$$\mathbf{m}^{(J)}(\mathbf{x}^{(J-1)}) = \mathbf{g}^{(J)}(\mathbf{x}^{(J-1)}) \odot \mathbf{x}^{(J-1)} + \mathbf{h}^{(J)}(\mathbf{x}^{(J-1)}), \tag{17}$$

where:

$$\mathbf{x}^{(J-1)} = \mathbf{m}^{(J-1)}(\mathbf{x}^{(J-2)}) \quad , \; J \geq 2 \tag{18}$$

While multiplication of a known vector by a Gaussian random matrix keeps Gaussianity, after the first warping layer, the product of two Gaussians is no longer Gaussian in (17). For the layered formulation, we therefore apply moment matching to approximate each layer's warped input as a Gaussian $\mathbf{x}^{(J)} \sim \mathcal{N}(\hat{\mathbf{x}}^{(J)}, \mathbf{\Sigma}_{\mathbf{x}}^{(J)})$. Making independence assumptions on (17) and applying known results for the Hadamard product of independent random variables [38], we have:

$$\hat{\mathbf{x}}^{(J)} = \hat{\mathbf{g}}^{(J)} \odot \hat{\mathbf{x}}^{(J-1)} + \hat{\mathbf{h}}^{(J)} \tag{19}$$

$$\mathbf{\Sigma}_{\mathbf{x}}^{(J)} = \mathbf{\Sigma}_{\mathbf{x}}^{(J-1)} \odot \mathbf{\Sigma}_{\mathbf{g}}^{(J)} + \mathbf{\Sigma}_{\mathbf{x}}^{(J-1)} \odot \hat{\mathbf{g}}^{(J)} \hat{\mathbf{g}}^{(J)\mathsf{T}} + \mathbf{\Sigma}_{\mathbf{g}}^{(J)} \odot \hat{\mathbf{x}}^{(J-1)} \hat{\mathbf{x}}^{(J-1)\mathsf{T}} + \mathbf{\Sigma}_{\mathbf{h}}^{(J)} \,, \tag{20}$$

where $\mathbf{g}^{(J)} \sim \mathcal{N}(\hat{\mathbf{g}}^{(J)}, \mathbf{\Sigma}_{\mathbf{g}}^{(J)})$ and $\mathbf{h}^{(J)} \sim \mathcal{N}(\hat{\mathbf{h}}^{(J)}, \mathbf{\Sigma}_{\mathbf{h}}^{(J)})$ are the SSGP predictions using the expected feature map (Equation 11) of the previous layer's output $\mathbf{x}^{(J-1)}$.

The layered warping allows for more complex input transformations. The drawback, however, is an increased computational cost due to the additional hyper-parameters, i.e. the pseudo-training points. In addition, we are taking a non-linearly transformed Gaussian input, which leads to a non-Gaussian result, and moment-matching it with a Gaussian. This distribution mismatch leads to compounding effects across several layers which could make the top-level Gaussian tend to a high-variance flat distribution. However, the training process should compensate for this increase in variance by adjusting the pseudo-training points according to a loss that takes the data into account, e.g. the GP marginal likelihood.

3.4 Joint training

The goal of optimization in learning our warping with uncertainty is to quickly discover hyper-parameters whose models explain the variation in the data. We also want to avoid pathologies that may manifest with an arbitrarily complex warping function. We do this by minimising the model's negative log-marginal likelihood. Given a set of observations $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, we learn the hyper-parameters $\boldsymbol{\theta}$ by minimising the negative log-marginal likelihood:

$$-\log p(\mathbf{y}|\boldsymbol{\theta}) = \frac{1}{2\sigma_n^2}(\mathbf{y}^\mathsf{T}\mathbf{y} - \mathbf{y}^\mathsf{T}\hat{\boldsymbol{\Phi}}_F^\mathsf{T}\mathbf{A}_F^{-1}\hat{\boldsymbol{\Phi}}_F\mathbf{y}) + \frac{1}{2}\log|\mathbf{A}_F| - \frac{D}{2}\log\sigma_n^2 + \frac{M}{2}\log 2\pi\sigma_n^2 \,, \quad (21)$$

where $\hat{\boldsymbol{\Phi}}_F$ is the matrix whose rows to expected feature maps for the top-level SSGP, i.e. $[\hat{\boldsymbol{\Phi}}_F]_i = \mathbb{E}_\mathbf{m}[\boldsymbol{\phi}_F(\mathbf{m}(\mathbf{x}_i))]^\mathsf{T}$, and $|\mathbf{A}_F|$ denotes the determinant of $\mathbf{A}_F$. The expectation is taken under the warping $\mathbf{m}$, whose parameters are computed from the predictive mean and covariance functions of the lower-level GPs (cf. (12) and (13)), and available in closed form via Equation 11.

3.5 Computational complexity

The top-level function u and two warping functions $\mathbf{G}$ and $\mathbf{h}$ all inherit the complexity of SSGP with and without predictions under uncertainty [8] which is $\mathcal{O}(nm^2 + m^3)$ for n samples and m features. For multiple warping levels this cost is multiplied by the number of levels J therefore the overall complexity remains $\mathcal{O}(nm^2 + m^3)$. In practice m is very small with $m < 1000$. For SSWIM, a single pseudo-training point has dimensionality D which is the same as the raw input $\mathbf{x}$. Therefore $\mathbf{G}$ and $\mathbf{h}$ consist of $D \times N_\mathbf{G}$ and $D \times N_\mathbf{h}$ pseudo-training points respectively. The functions u, $\mathbf{G}$ and $\mathbf{h}$ contain model and kernel hyperparameters of size $|\boldsymbol{\theta}_u|$, $|\boldsymbol{\theta}_\mathbf{G}|$ and $|\boldsymbol{\theta}_\mathbf{h}|$ respectively which should each not exceed much more than D for conventional stationary kernels.

3.6 On function classes of the warping

It has been remarked in prior work on deep GP models that degenerate covariance matrices may arise after consecutive compositions [39, 29]. Recent works, such as [29, 40], that employ a dynamical systems based formulation can demonstrate improved uncertainty propagation under an *injective* warping (by maintaining monotonic constructions) as opposed to conventional deep GP models [22]. Indeed our proposed SSWIM is not guaranteed to be injective and falls into the most general class of functions. An interesting consequence of this is that one could argue injectivity is *not necessarily ideal* for learning latent mappings and furthermore it certainly is not a necessary condition for preventing collapse of uncertainty although such phenomena may be correlated. To comment further, by relaxing from certain function types it is plausible for multiple different input values in a prior warping layer to *warp into the same input location* in the next layer; i.e. in a surjective function. This may ultimately be a desirable property – it suggests compressiblity of the input domain – in that there might be an underlying non-monotonic, non-stationary covariance function at play in the latent representation. Such expressiveness would not be able to be directly captured by a purely injective mapping. Injectivity and even bijectivity could be enforced as an additional constraint and this perspective undoubtedly deserves future investigation.

3.7 On kernel priors

One may enquire about the choice of kernels for latent and top level functions. Our methodology is, generally speaking, "kernel prior agnostic" in the sense that the nonstationarity is accomplished through the affine warpings. We remark that kernel choice undoubtedly may play a role in performance. One could indeed use extremely expressive kernels, like the stationary spectral mixture [41] or quantile kernel representations [42]. However, to restrict the space of analysis to measure the effect of the warping construction, we aimed to forgo kernel discovery.

4 EXPERIMENTS

We experimentally validate SSWIM alongside various state of the art methods in both small and large data regimes as well as expand upon the intuition in Section 4.1.1 by examining specific aspects of the model. Section 4.1.2 analyses computational complexity and model performance with respect to

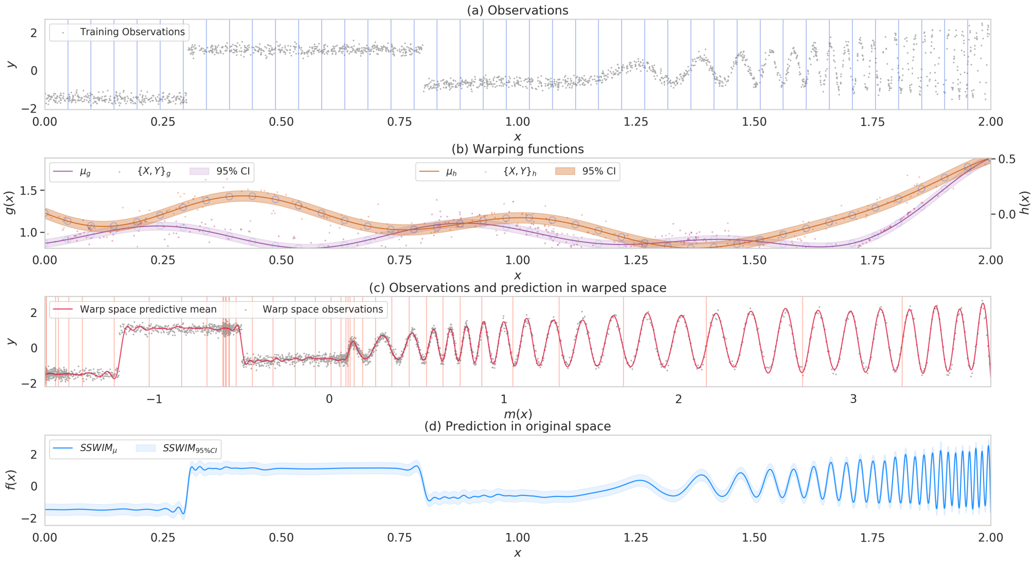

Figure 1: Visualisation of SSWIM learning an input warping. (a) Noisy training data. Going left to right, the signal observations exhibit abrupt steps, periodic, and spatial frequency nonstationarity. (b) The learned warping functions. (c) The training data after input warping, and (d) Final prediction with respect to the warped inputs. The key observation here is how the spatially varying frequencies and steps in the original training data from (a) have been transformed to (c) where the warped data varies in a more uniform (stationary) manner.

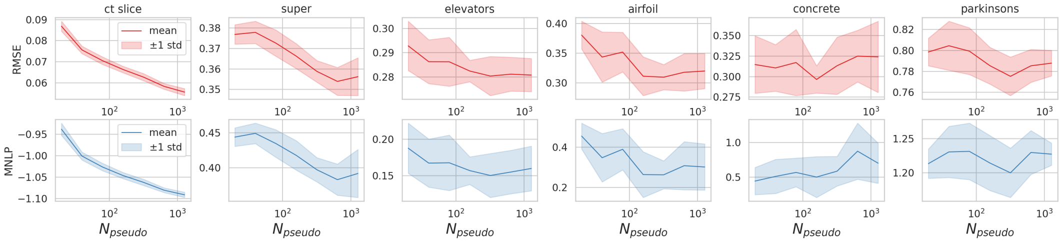

Figure 2: Performance in RMSE and MNLP as the number of pseudo-training points increases.

the pseudo-training points. We investigate increasing the number of warping levels in Section 4.1.3. Large scale comparison alongside various algorithms is presented in Section 4.2.

For every quantitative experiment, we report the mean and standard deviation over ten repeats. Metrics are presented in the standardized data scale. In all experiments the Matern $\frac{3}{2}$ is used as the base kernel. For performance evaluation we use the test Root Mean Square Error (RMSE) and test Mean Negative Log Probability (MNLP). These are defined as RMSE = $\sqrt{\langle (y_{*j} - \mu_{*j})^2 \rangle}$ and MNLP = $\frac{1}{2}\langle (\frac{y_{*j} - \mu_{*j}}{\sigma_{*j}})^2 + \log \sigma_{*j}^2 + \log 2\pi \rangle$ where y_{*j} is the true test value, and μ_{*j} and σ_{*j}^2 are the predictive mean and variance respectively for the j^{th} test observation. Mean is denoted as $\langle \cdot \rangle$.

4.1 Analysis

In this section we comment on the geometric interpretation of the warping in the sense of conditional affine maps as well as analyse key hyperparameters.

4.1.1 Inductive bias and a geometric interpretation

An intuitive interpretation of SSWIM is by imagining it as learning a conditional affine transformation. The quintessential affine transformation of some vector x is described as $\mathbf{A}x + b$ for some multiplica-

Table 1: RMSE and MNLP metrics for various real world datasets.

	(D, N) Method	$(8, 1030)$ concrete	$(16, 5875)$ parkinsons	$(15, 17379)$ bikeshare	$(379, 53500)$ ct slice	$(81, 21263)$ supercond	$(9, 45730)$ protein	$(77, 583250)$ buzz	$(90, 515345)$ song
RMSE $(\times 10^{-1})$	SSWIM$_1$	3.05 ± 0.26	7.63 ± 0.20	0.13 ± 0.04	0.46 ± 0.02	3.44 ± 0.14	5.91 ± 0.07	2.98 ± 0.04	8.12 ± 0.05
	SSWIM$_2$	**3.01** ± **0.31**	**7.55** ± **0.15**	0.11 ± 0.03	**0.23** ± **0.01**	**3.02** ± **0.04**	**5.80** ± **0.08**	**2.40** ± **0.01**	**7.97** ± **0.03**
	DSDGP	5.88 ± 1.24	7.94 ± 0.20	0.33 ± 0.55	4.81 ± 1.18	5.10 ± 0.84	5.96 ± 0.06	3.65 ± 0.75	8.46 ± 0.03
	DKL	3.18 ± 0.38	8.84 ± 0.74	0.24 ± 0.03	0.52 ± 0.08	3.46 ± 0.18	7.15 ± 1.10	4.11 ± 3.33	16.66 ± 8.14
	RFFNS	3.46 ± 0.24	8.15 ± 0.15	0.05 ± 0.01	4.39 ± 0.27	3.85 ± 0.05	6.87 ± 0.06	5.70 ± 0.84	8.35 ± 0.03
	SVGP	3.32 ± 0.26	8.14 ± 0.12	0.06 ± 0.03	1.16 ± 0.02	4.06 ± 0.05	7.32 ± 0.08	9.98 ± 0.02	12.19 ± 0.18
	SGPR	5.55 ± 0.58	7.86 ± 0.22	0.67 ± 0.18	1.79 ± 0.04	4.27 ± 0.06	6.45 ± 0.07	2.89 ± 0.02	8.40 ± 0.04
	RFFS	3.33 ± 0.30	8.24 ± 0.17	**0.03** ± **0.00**	2.34 ± 0.05	3.89 ± 0.06	6.91 ± 0.07	3.78 ± 0.14	8.36 ± 0.04
MNLP $(\times 10^{-1})$	SSWIM$_1$	10.22 ± 4.15	11.95 ± 0.47	-11.89 ± 0.15	-11.24 ± 0.05	3.55 ± 0.32	8.95 ± 0.12	2.03 ± 0.13	12.08 ± 0.05
	SSWIM$_2$	5.19 ± 2.59	12.50 ± 0.44	-11.78 ± 0.07	**-11.79** ± **0.02**	**2.82** ± **0.29**	**8.82** ± **0.15**	**-0.09** ± **0.04**	**11.93** ± **0.04**
	DSDGP	11.02 ± 1.06	**11.91** ± **0.24**	-23.28 ± 8.29	6.62 ± 2.61	7.36 ± 1.62	9.04 ± 0.10	3.80 ± 2.02	12.52 ± 0.04
	DKL	7.69 ± 0.20	13.17 ± 1.12	6.82 ± 0.01	6.83 ± 0.01	7.76 ± 0.10	11.02 ± 1.53	9.01 ± 4.65	42.64 ± 44.77
	RFFNS	3.31 ± 0.45	12.18 ± 0.18	-11.97 ± 0.00	5.95 ± 0.66	4.66 ± 0.12	10.39 ± 0.08	8.78 ± 1.87	12.39 ± 0.04
	SVGP	**2.83** ± **0.56**	12.21 ± 0.14	**-27.70** ± **1.24**	-5.98 ± 0.13	5.32 ± 0.12	11.10 ± 0.09	63.31 ± 3.44	18.02 ± 0.09
	SGPR	8.48 ± 2.10	12.39 ± 0.30	-13.67 ± 0.98	-3.14 ± 0.26	5.58 ± 0.10	10.05 ± 0.13	1.11 ± 0.12	11.97 ± 0.07
	RFFS	3.05 ± 0.96	12.29 ± 0.20	-11.98 ± 0.00	-0.33 ± 0.22	4.79 ± 0.13	10.45 ± 0.09	4.41 ± 0.37	12.40 ± 0.05

tion matrix $\mathbf{A}$ and addition vector b. Such transformations are typically interpreted geometrically [43] as *translation*, *rotation*, *reflection*, *scale* and *shear*. SSWIM learns a *conditional* affine map that *depends on the input*. I.e. $\mathbf{A}$ and b become maps $\mathbf{A}(x)$ and $b(x)$. By directly applying a learned warping to the original input data we transform the inputs into a locally Euclidean manifold which ultimately preserves any structure with respect to the input resulting in a convenient inductive bias. Observe in Figure 1 (c) how we have non-uniformly "stretched out" out the left and rightmost parts of the original data in (a) to produce a new warped dataset. What was original spatially nonstationary becomes spatially homogeneous resulting excellent prediction as in Figure 1 (d).

4.1.2 How many pseudo-training points?

To understand the overall sensitivity of our method we visualise the predictive performance as a function of the number of pseudo-training points. Figure 2 shows performance, in log scale, with respect to the number of pseudo-training points on real world datasets. We observe trending improvement however very few pseudo-targets are required to get excellent performance, even in much higher dimensional problems like *superconductivity* ($D = 81$) and *ct slice* ($D = 379$), suggesting that there is significant expressiveness in the underlying warping function.

We remark that a possible drawback of pseudo-training points and fitting a stochastic model over those points is the question of how to set the prior of their locations. Furthermore, how do we initialise them? To answer this, it is natural to set $\mathbf{G}$ and $\mathbf{h}$ to be fit to noisy pseudo-targets with mean $\mathbf{I}$ and $\mathbf{0}$ respectively. This has a nice interpretation as the matrices corresponding to the identity operations of an affine transformation.

4.1.3 Increased warping depth

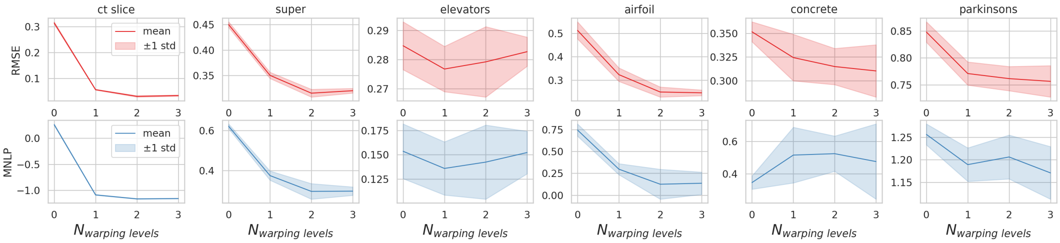

Figure 3: Performance in RMSE and MNLP as the number of warping levels increases.

In this experiment we evaluate the predictive performance of SSWIM as we increase the number of levels of consecutive input warping from 0 to 3. A depth of 0 simply corresponds to the *stationary* SSGPR specification. Figure 3 summarises the results. For all the datasets we can see that adding just a single level of input warping increases predictive performance. Adding additional levels of warping seems to consistently improve performance however it adds additional variance to all results which could be explained by the additional complexity required for optimization.

4.2 Real datasets

We compare our model on various real-world datasets including multiple regression tasks [44, 45, 46]. All datasets are standardised using the train set. We use $\frac{2}{3}$ of the samples for training and the remaining $\frac{1}{3}$ for testing. We compare multiple related algorithms alongside our proposed method SSWIM using both one level of warping (SSWIM$_1$) and two levels of warping (SSWIM$_2$), Deep Kernel Learning [21] (DKL), SSGP with stationary random fourier features kernel (RFFS), SSGP with nonstationary kernel features (RFFNS) with freely variable mean and width for the Matern $\frac{3}{2}$ spectral frequencies [15, 2], Sparse Gaussian Process Regression (SGPR) [37], Sparse Variational Gaussian Process (SVGP) [47], and Doubly Stochastic Deep GP with 2 layers (DSDGP) [23]. All experiments were performed on a Linux machine with a single Titan V GPU. We ran all methods for 150 iterations with stochastic gradient descent and the library GPyTorch was used for DKL, DSDGP, SGPR, and SVGP. We have provided implementations for RFFS, RFFNS, and SSWIM. Full experimental details are provided in the supplementary material with additional commentary. PyTorch Code is provided to reproduce the experimental results.

In the main experimental results given in Table 1 we can observe a consistent high performance across all datasets for SSWIM in all tasks for the RMSE metric. For *concrete, parkinsons* and *bikeshare* SSWIM is outperformed in MNLP by DSDGP, SVGP and RFFS suggesting they were more capable of representing the predictive distribution rather than the mean. For the remaining datasets SSWIM has performed extremely well, most notably on the high dimensional problem *ct slice*. SSWIM$_2$ with two levels of warping comprehensively outperforms other methods as well as SSWIM$_1$ which also performs competitively. These results further corroborate the analysis given in Figure 2 and Figure 3.

5 Conclusion

We have proposed a crucial advance to the sparse spectrum Gaussian process framework to account for nonstationarity through a novel input warping formulation. Our model analytically incorporates complete Gaussian measures in the functional input warping with the concept of *pseudo-training* data and latent *self-supervision*. We have further extended this core contribution with the necessary results to extend the warping to multiple levels resulting in higher levels of model expressiveness.

Experimentally, the methodology we propose has demonstrated excellent results in the total number of hyperparameters for various low and high dimensional real world datasets when compared to deterministic and neural network based approaches but also performing exceptionally well in contrast to deep Gaussian processes. Our model suggests an interesting and effective inductive bias when interpreted as a learned conditional affine transformation. This perspective invites a fresh take on how to discover more effective representations of nonstationary data.

Acknowledgements

We are grateful to and thank the anonymous reviewers for their thoughtful comments.

Broader Impact

The problem we address in this paper of efficient modelling of nonstationary stochastic processes is fundamental in geostatistics, time-series analysis, and the study of dynamical systems. To this end, our technique is directly applicable to spatial-temporal problems such as air pollution monitoring, the spread of diseases, and the study of natural resources such as underground water. In all of these problems, the strength of the spatial relationships between inputs varies with respect to the location. For example, during the current pandemic, nearby cities might exhibit different levels of infection rates within their boundaries but still being related due to infected people travelling between them Senanayake et al. [48]. Our approach is directly applicable to these cases and can be incorporated within epidemiological models that typically aggregate populations in large regions for a more refined prediction and study of intervention policies such as social distancing.

References

[1] Andrew Y Ng, Adam Coates, Mark Diel, Varun Ganapathi, Jamie Schulte, Ben Tse, Eric Berger, and Eric Liang. Autonomous inverted helicopter flight via reinforcement learning. In *Experimental robotics IX*. Springer, 2006.

[2] Jean-Francois Ton, Seth Flaxman, Dino Sejdinovic, and Samir Bhatt. Spatial mapping with Gaussian processes and nonstationary fourier features. *Spatial statistics*, 2018.

[3] Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using Bayesian networks to analyze expression data. *Journal of computational biology*, 2000.

[4] Ruben Martinez-Cantin. Bayesian optimization with adaptive kernels for robot control. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[5] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*. Springer, 2004.

[6] H. Bauer. *Probability theory and elements of measure theory*. Probability and mathematical statistics. Academic Press, 1981.

[7] Miguel Lázaro-Gredilla, Joaquin Quiñonero-Candela, Carl Edward Rasmussen, and Aníbal R Figueiras-Vidal. Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research (JMLR)*, 2010.

[8] Yunpeng Pan, Xinyan Yan, Evangelos A. Theodorou, and Byron Boots. Prediction under uncertainty in sparse spectrum Gaussian processes with applications to filtering and control. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, 2017.

[9] Dave Higdon, Jenise Swall, and J Kern. Non-stationary spatial modeling. *Bayesian statistics*, 1999.

[10] Christopher J Paciorek and Mark J Schervish. Nonstationary covariance functions for Gaussian process regression. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

[11] Salomon Bochner. *Vorlesungen über Fouriersche Integrale: von S. Bochner*. Akad. Verl.-Ges., 1932.

[12] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Neural Information Processing Systems (NIPS)*, 2007.

[13] A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomisation in learning. In *Neural Information Processing Systems (NIPS)*, 2008.

[14] Yves-Laurent Kom Samo and Stephen Roberts. Generalized spectral kernels. *arXiv preprint arXiv:1506.02236*, 2015.

[15] Sami Remes, Markus Heinonen, and Samuel Kaski. Non-stationary spectral kernels. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[16] Shengyang Sun, Guodong Zhang, Chaoqi Wang, Wenyuan Zeng, Jiaman Li, and Roger Grosse. Differentiable compositional kernel learning for Gaussian processes. In *International Conference on Machine Learning (ICML)*, 2018.

[17] Roberto Calandra, Jan Peters, Carl Edward Rasmussen, and Marc Peter Deisenroth. Manifold Gaussian processes for regression. In *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016.

[18] Andrew Gordon Wilson, David A Knowles, and Zoubin Ghahramani. Gaussian process regression networks. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1139–1146, 2012.

[19] Paul D Sampson and Peter Guttorp. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association*, 1992.

[20] Ethan B Anderes, Michael L Stein, et al. Estimating deformations of isotropic Gaussian random fields on the plane. *The Annals of Statistics*, 2008.

[21] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

[22] Andreas Damianou and Neil Lawrence. Deep Gaussian processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 207–215, 2013.

[23] Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep Gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[24] Kurt Cutajar, Edwin V Bonilla, Pietro Michiardi, and Maurizio Filippone. Random feature expansions for deep Gaussian processes. In *International Conference on Machine Learning (ICML)*, 2017.

[25] Thang Bui, Daniel Hernández-Lobato, Jose Hernandez-Lobato, Yingzhen Li, and Richard Turner. Deep Gaussian processes for regression using approximate expectation propagation. In *International Conference on Machine Learning (ICML)*, 2016.

[26] Jasper Snoek, Kevin Swersky, Rich Zemel, and Ryan Adams. Input warping for Bayesian optimization of non-stationary functions. In *International Conference on Machine Learning (ICML)*, 2014.

[27] Markus Heinonen, Henrik Mannerström, Juho Rousu, Samuel Kaski, and Harri Lähdesmäki. Non-stationary Gaussian process regression with hamiltonian monte carlo. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

[28] M. N. Gibbs. Bayesian Gaussian processes for regression and classification. *Ph. D. Thesis, Department of Physics, University of Cambridge*, 1997.

[29] Pashupati Hegde, Markus Heinonen, Harri Lähdesmäki, and Samuel Kaski. Deep learning with differential Gaussian process flows. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[30] Chunyi Wang and Radford M. Neal. Gaussian Process Regression with Heteroscedastic or Non-Gaussian Residuals. Technical report, University of Toronto, Toronto, Canada, 2012. URL `http://arxiv.org/abs/1212.6246`.

[31] Vincent Dutordoir, Hugh Salimbeni, James Hensman, and Marc Deisenroth. Gaussian Process Conditional Density Estimation. In S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2385–2395. Curran Associates, Inc., 2018.

[32] Edward Snelson, Zoubin Ghahramani, and Carl E Rasmussen. Warped Gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

[33] C Bishop. Pattern recognition and machine learning (information science and statistics), 1st edn. 2006. corr. 2nd printing edn. *Springer, New York*, 2007.

[34] Mauricio A Alvarez, Lorenzo Rosasco, and Neil D Lawrence. Kernels for Vector-Valued Functions: a Review. Technical report, MIT - Computer Science and Artificial Intelligence Laboratory, 2011.

[35] Carl Jidling, Niklas Wahlström, Adrian Wills, and Thomas B. Schön. Linearly constrained Gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[36] Rafael Oliveira, Lionel Ott, and Fabio Ramos. Bayesian optimisation under uncertain inputs. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, Naha, Okinawa, Japan, 2019.

[37] Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.

[38] H. Neudecker, S. Liu, and W. Polasek. The Hadamard product and some of its applications in statistics. *Statistics*, 26(4):365–373, 1995.

[39] David Duvenaud, Oren Rippel, Ryan Adams, and Zoubin Ghahramani. Avoiding pathologies in very deep networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2014.

[40] Ivan Ustyuzhaninov, Ieva Kazlauskaite, Carl Henrik Ek, and Neill Campbell. Monotonic gaussian process flows. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020.

[41] Andrew Wilson and Ryan Adams. Gaussian process kernels for pattern discovery and extrapolation. In *International Conference on Machine Learning (ICML)*, 2013.

[42] Anthony Tompkins, Ransalu Senanayake, Philippe Morere, and Fabio Ramos. Black box quantiles for kernel learning. In *International Conference on Artificial Intelligence and Statistics*, 2019.

[43] Rafael González and Richard Woods. Digital image processing. isbn: 9780131687288. *Prentice Hall*, 2008.

[44] Dheeru Dua and Casey Graff. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2017.

[45] Luís Torgo. Regression datasets. `https://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html`, 2019.

[46] D Cole, C Martin-Moran, AG Sheard, HKDH Bhadeshia, and DJC MacKay. Modelling creep rupture strength of ferritic steel welds. *Science and Technology of Welding and Joining*, 2000.

[47] James Hensman, Alexander G. de G. Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[48] Ransalu Senanayake, Simon O'Callaghan, and Fabio Ramos. Predicting spatio-temporal propagation of seasonal influenza using variational gaussian process regression. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.