arXiv:1905.10963v1 [cs.LG] 27 May 2019

Interpretable deep Gaussian processes

Chi-Ken Lu
Math and CS
Rutgers University Newark

Scott Cheng-Hsin Yang
Math and CS
Rutgers University Newark

Xiaoran Hao
Math and CS
Rutgers University Newark

Patrick Shafto
Math and CS
Rutgers University Newark

Abstract

We propose interpretable deep Gaussian Processes (GPs) that combine the expressiveness of deep Neural Networks (NNs) with quantified uncertainty of deep GPs. Our approach is based on approximating deep GP as a GP, which allows explicit, analytic forms for compositions of a wide variety of kernels. Consequently, our approach admits interpretation as both NNs with specified activation functions and as a variational approximation to deep GPs. We provide general recipes for deriving the effective kernels for deep GPs of two, three, or infinitely many layers, composed of homogeneous or heterogeneous kernels. Results illustrate the expressiveness of our effective kernels through samples from the prior and inference on simulated data and demonstrate advantages of interpretability by analysis of analytic forms, drawing relations and equivalences across kernels, and a priori identification of non-pathological regimes of hyperparameter space.

1 Introduction

The success of deep learning models in numerous domains is generally perceived as stemming from greater expressivity which results in powerful generalization [1]. However, deep learning models are still considered black box as their complexity arises from the enormous number of parameters and possible choices for different structures and activation units. Understanding these models remains an open and challenging problem [2–6]. Williams [7] demonstrated that the characteristics of single-layer neural networks (NNs) can be understood from its effective kernel. It is thus appealing to create more interpretable methods through the correspondence between deep learning and kernel-based methods [8–15] which have the advantage of an explicit mathematical formalization.

Quantifying uncertainty of inferences is critical issue for deep learning models as they are deployed in a broader array of settings [16, 17]. Gaussian processes (GPs) [18], the infinitely wide limit of single-layer neural network with random weight parameters [19], are attractive alternative models capable of quantifying uncertainty through Bayesian inference. Moreover, GPs can be composed into deep GPs [20], with inference via variational approximations [21–23], to achieve expressiveness that is comparable to deep NNs. However, as for NNs, with this expressivity comes difficulty in interpretability. It is desirable to leverage interpretability of explicit mathematical formalizations associated with kernel-based methods, while preserving experessivity and uncertainty quantification.

We introduce a new variational approximation to Bayesian inference in deep GPs that effectively represents any depth as single layer. By approximating deep GP as a GP, we gain the ability to analytically integrate over multiple *heterogeneous* layers, allowing a great variety of effectively deep, yet interpretable kernels. Section 2 reviews recent and relevant papers to highlight important prior works. Section 4 introduces our technical approach with recipes for integrating three types of

Preprint. Under review.

Feature					Reference
Deep	Bayesian	composition (homogeneous)	composition (heterogeneous)	analytic kernel	
F	T	N/A	N/A	T	[7]
T	F	T	F	T	[10]
T	F	T	F	T	[24, 25]
T	F	T	T	T	[27]
T	T	T	N/A	T	[26]

Table 1: Table summary of related works. Symbols T and F stand for true and false while N/A denotes not applicable.

compositions along with representative results for compositions of two layers (Table 2), three layers, and infinitely deep GPs. Section 5 addresses interpretability of our effective kernels. Section 6 is a detailed case study of squared exponential kernels. Section 7 presents conclusions.

2 Related works

Our contribution is the development of Bayesian deep GPs with analytic forms for compositional heterogeneous kernels. We focus on the most closely related previous works, highlighting the differences (see Table 1). Because activation functions of deep neural networks correspond with kernels in deep GP, our results shall also help interpret the heterogeneous networks using different activation units.

Williams [7] was first to derive analytic kernels of one-layer neural networks, using sigmoidal and Gaussian activation functions. Later, Cho and Saul [10] extended analytic results to polynomial activation functions and further made extension to deep models. Daniely *et al.* [24] extended these results for homogeneous deep networks with 2nd hermite, step, and exponential activation functions and Poole *et al.* [25] to $\tanh$ activation functions. Our approach differs by focusing on deep GPs for Bayesian inference, and allowing heterogeneous kernels.

Lee *et al.* [26] extended the correspondence between NNs and GPs [19] to deep NNs and GP, via the central limit theorem, focusing on homogeneous kernels.

Duvenaud *et al.* [27] studied pathologies in deep GPs, primarily focusing on squared exponential kernels and sampling functions from the prior. We study pathologies using our analytic approach identifying regimes of non-pathology in hyperparameter space, and we show results for prediction for a variety of kernels while controlling complexity through marginalization, which interestingly yields a different analytic form for deep squared exponential kernels.

3 Brief review of Gaussian Processes

A Gaussian process (GP) is a prior distribution $p(f)$ over continuous function f of which any subset of points $\{f_i\}$ follows the joint multivariate Gaussian distribution specified by mean function $\mu(\mathrm{x}_i)$ and covariance matrix $K_{ij} = k(\mathrm{x}_i, \mathrm{x}_j)$. The properties of function, such as smoothness, are encoded in the covariance function $k(\mathrm{x}_i, \mathrm{x}_j) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$. For example, functions that are sampled from the squared exponential kernel are infinitely differentiable, while the non-stationary neural network kernel [7] may generate functions that have discontinuity. Under the Bayesian framework, the posterior distribution $p(f|\mathcal{D}, \theta)$ is used to make predictions. The hyperparameters, denoted by θ, are determined by the maximizing the marginal distribution $p(\mathcal{D}|\theta)$. The benefit of Bayesian learning of θ is that over fitting is naturally avoided by the two competing terms, data-fit and complexity penalty, in the marginal likelihood.

4 Approximate deep GP with moment matching

Deep GP serves as a prior distribution over both homogeneous and heterogeneous compositions of functions, where the resultant function f from L layers of warping is given by

$$f \sim \mathcal{GP}(0, k^{(L)}(h_i^{(L-1)}, h_j^{(L-1)})) \tag{1}$$

with the hidden functions $h^{m=1:(L-1)}$

$$h^{(m)} \sim \mathcal{GP}(0, k^{(m)}(h_i^{(m-1)}, h_j^{(m-1)})) \,. \tag{2}$$

Homogeneous deep GPs compose functions with the same kernel, $k^{(i)} = k^{(j)},\ \forall i, j$, whereas heterogeneous deep GPs compose functions with different kernels $\exists i, j$ s.t. $k^{(i)} \neq k^{(j)}$. By convention, $h^{(0)}$ refers to the input $\mathrm{x} \in \mathbb{R}^D$. For a finite set of $\{f_i\}_{i=1}^N$ associated with the input $\{\mathrm{x}_i\}_{i=1}^N$, the joint distribution $p(\mathbf{f}, \mathbf{h}^{(1:L-1)}|\mathrm{X})$ can be expressed as

$$p(\mathbf{f}|\mathbf{h}^{(L-1)})p(\mathbf{h}^{(L-1)}|\mathbf{h}^{(L-2)}) \dots p(\mathbf{h}^{(1)}|\mathrm{X}) \,, \tag{3}$$

from which it is seen that the marginal distribution $p(\mathbf{f}|\mathrm{X}) = \int p(\mathbf{f}, \mathbf{h}|\mathrm{X}) d\mathbf{h}$ is intractable because the hidden function vectors $\mathbf{h}$ appear in the inverse of covariance matrix. In addition, the marginal distribution for f is generally non-Gaussian implying that a deep GP is not a GP.

To illustrate, we consider a deep GP with only one hidden layer, i.e. $L = 1$. The exact marginal distribution associated with the set of points $\mathbf{f}$ is

$$p(\mathbf{f}|\mathrm{X}) = \int dh_1 \cdots dh_N p(\mathbf{f}|\mathbf{h}) p(\mathbf{h}|\mathrm{X}) \,, \tag{4}$$

where the hidden function values $h_{1:N}$ are marginalized out. Because it is intractable, we proceed by approximating $p(\mathbf{f}|\mathrm{X})$ with a joint Gaussian so that only the mean $\mathbf{v}$ and the covariance matrix $\mathbf{K}_{\text{eff}}$ are needed, i.e.

$$p(\mathbf{f}|\mathrm{X}) \approx \mathcal{N}(\mathbf{f}; \mathbf{v}, \mathbf{K}_{\text{eff}}) \,. \tag{5}$$

Here we note that the matrix elements of the effective kernel matrix $[\mathbf{K}_{\text{eff}}]_{ij} = k_{\text{eff}}(\mathrm{x}_i, \mathrm{x}_j)$ is a function encoding the correlation propagating through the hidden layers in deep GP.

The mean $\mathbf{v}$ is obtained by taking the expectation, $v_i = \mathscr{E}[f_i] = \int f_i p(\mathbf{f}, \mathbf{h}|\mathrm{X}) d\mathbf{f} d\mathbf{h}$, and it follows from the zero-mean GP that $\mathbf{v}$ is zero. The covariance matrix is obtained by taking the second moment $[\mathbf{K}_{\text{eff}}]_{ij} = \mathscr{E}[f_i f_j]$. The effective covariance function becomes

$$k_{\text{eff}}(\mathrm{x}_i, \mathrm{x}_j) = \int k^{(1)}(h_i, h_j) p(\mathbf{h}|\mathrm{X}) dh_1 \cdots dh_N \,, \tag{6}$$

from which one may note that taking the second moment has removed the most formidable part, the inverse of covariance matrix $\mathbf{K}^{(1)}$, appearing in $p(\mathbf{f}|\mathbf{h})$. By employing the marginal property of multivariate Gaussian distribution, we arrive at the following integral,

$$k_{\text{eff}}(\mathrm{x}_i, \mathrm{x}_j) \propto \int k^{(1)}(h_i, h_j) \exp\left[-\frac{\mathrm{h}^T \tilde{\mathbf{K}}^{-1} \mathrm{h}}{2} \right] dh_i dh_j \,, \tag{7}$$

where h is the column vector with entries of h_i and h_j. The two-by-two matrix $\tilde{\mathbf{K}}$ is the submatrix of full covariance matrix $\mathbf{K}^{(0)}$ associated with the first-layer GP.

Now it can be seen that the moment matching method simplifies the difficult problem into the two-variable integral in Equation (7). For general covariance functions, the two-by-two matrix $\tilde{\mathbf{K}}$ is expressed as

$$\tilde{\mathbf{K}} = \begin{pmatrix} \alpha & \gamma \\ \gamma & \beta \end{pmatrix} \,, \tag{8}$$

in which the matrix elements $\alpha = k^{(0)}(\mathrm{x}_1, \mathrm{x}_1)$, $\beta = k^{(0)}(\mathrm{x}_2, \mathrm{x}_2)$, and $\gamma = k^{(0)}(\mathrm{x}_1, \mathrm{x}_2)$. For stationary covariance function such as SE, the diagonal elements are equal $\alpha = \beta$.

We demonstrate three kinds of covariance functions for $k^{(1)}$ for which closed form effective kernel can be obtained.

Case 1. The simplest case is the linear kernel $k^{(1)}(h_1, h_2) = \sigma_1^2 h_1 h_2$. We can show that the corresponding effect is to multiply a constant, namely

$$\mathrm{Lin}[k^{(0)}(\mathrm{x}_1, \mathrm{x}_2)] = \sigma_1^2 k^{(0)}(\mathrm{x}_1, \mathrm{x}_2)\,. \tag{9}$$

The proof is straightforward by observing that the integral of $k^{(1)}/\sigma_1^2$ against the Gaussian measure is by definition the covariance matrix $\mathscr{E}[\mathbf{h}\mathbf{h}^T]$.

Case 2. The general Gaussian kernel $k^{(1)}(h_1, h_2) = \sigma_1^2 \exp[-(uh_1^2 - 2vh_1h_2 + uh_2^2)/2]$ with the parameters $u > v > 0$ corresponds to the non-stationary kernel [7] derived from neural network with Gaussian activation function. The effective kernel is shown to be

$$\mathrm{NN}[k^{(0)}(\mathrm{x}_1, \mathrm{x}_2)] = \sigma_1^2 \left[(u^2 - v^2)D + u(\alpha + \beta) - 2v\gamma + 1\right]^{-\frac{1}{2}}\,, \tag{10}$$

with $D = \alpha\beta - \gamma^2$. The general Gaussian kernel reduces to the squared exponential (SE) one if $u = v = \ell_1^{-2}$, which leads to the special case,

$$\mathrm{SE}[\cdot] = \sigma_1^2 \left(1 + \frac{\alpha + \beta - 2\gamma}{\ell_1^2}\right)^{-\frac{1}{2}}\,. \tag{11}$$

The proof starts with observing that one can rewrite the covariance function as exponential quadratic form, $k^{(1)} = \sigma_1^2 \exp[-\frac{1}{2}\mathrm{h}^T M \mathrm{h}]$. It follows that the integral in Equation 7 turns into another Gaussian integral so the effective covariance function is given by the square root ratio of $D' = \det[M + \tilde{\mathbf{K}}^{-1}]^{-1}$ to $D = \det[\tilde{\mathbf{K}}]$, and

$$k_{\mathrm{eff}} = \sigma_1^2 \sqrt{\frac{D'}{D}}\,. \tag{12}$$

Case 3. To capture periodic pattern in data, we are interested in squared cosine kernel (SC), $k^{(1)}(h_1, h_2) = \sigma_1^2 \cos^2 \frac{h_1 - h_2}{\ell_1}$. It can be shown that the effective kernel reads,

$$\mathrm{SC}[\cdot] = \frac{\sigma_1^2}{2}\left[1 + \exp\frac{2\gamma - \alpha - \beta}{\ell_1^2}\right]\,. \tag{13}$$

One can first employ the equality $\cos^2 z = [2 + \exp(i2z) + \exp(-i2z)]/4$, followed by rewriting $\exp[i(h_1 - h_2)] = \exp[i\mathrm{a}^T\mathrm{h}]$ with $\mathrm{a}^T = (1, -1)/\ell_1$. By completing the square in the exponent, one can in the end get $k_{\mathrm{eff}} \propto \exp[-2\mathrm{a}^T\tilde{\mathbf{K}}\mathrm{a}]$.

Case 4. The above can be generalized to recursive relations applicable to the case where depth > 2. For *homogeneous* composition, e.g. three-layer deep GP with SE kernel, one can obtain the effective SE[SE[SE]] kernel by plugging the two-layer effective kernel in Eq. (11) back into the γ term in the same equation. Doing so results in the following recursive relation,

$$k_{\mathrm{eff}}^{(L+1)}(\mathrm{x}_i, \mathrm{x}_j) = \frac{\sigma_{L+1}^2}{\sqrt{1 + 2(\ell_{L+1}^{-2})[\sigma_L^2 - k_{\mathrm{eff}}^{(L)}(\mathrm{x}_i, \mathrm{x}_j)]}}\,, \tag{14}$$

which relates the current covariance function $k^{(L)}$ to the covariance $k^{(L+1)}$ when the SE kernel with hyperparameters σ_{L+1} and ℓ_{L+1} is employed. Similar approach can be taken for obtaining other homogeneous composition kernels SC[SC[SC[...]]] (Eq. 13) and the non-stationary Gaussian kernels NN[NN[NN[...]]] (Eq. 10).

Duvenaud *et al.* [27] suggest connecting the input layer (i.e. the data) to each of the hidden layer to resolve pathologies of deep neural networks. Here one also has the freedom to do so. Because the input variables are not random variables, one can simply multiply the exponential factor $\exp(-||x_1 - x_2||^2)$ to the effective kernel in all cases. In later sections, we exploit our analytic form to identify regions of the parameter space that are non-pathological, thus obviating this modification.

Case 5. For *heterogeneous* composition of, for example SE, SC, and NN, three-layer deep GP, one can obtain the effective kernel by recursively plugging the effective previous kernels into the α, β, and γ term of subsequent kernels. For example, the SE[SC[NN]] kernel is

$$\mathrm{SE[SC[NN]]}(\mathrm{x_i}, \mathrm{x_j}) = \frac{\sigma_2^2}{\sqrt{1 + (\sigma_1/\ell_2)^{-2}[1 - \mathrm{e}^{\ell_1^{-2}(2\gamma - \alpha - \beta)}]}}, \tag{15}$$

with $\alpha = k(x_i, x_i)$, $\beta = k(x_j, x_j)$ and $\gamma = k(x_i, x_j)$ from the general Gaussian kernel k or from another neural network kernel (e.g. equation (4.29) in [18]).

Notation	Effective kernel
SE[SE]	$a[1 + b\, G(\Delta x^2, c)]^{-\frac{1}{2}}$
SC[SE]	$a\{1 + \exp[-bG(\Delta x^2, c)]\}$
SE[SC]	$a[1 + b \sin^2(\Delta x/c)]^{-\frac{1}{2}}$
SC[SC]	$a\{1 + \exp[-b \sin^2(\Delta x/c)]\}$
SE[Lin]	$\mathrm{RQ}\frac{1}{2}$
SC[Lin]	SE + Const.
SE[Lin+SE]	$a\{1 + bG(\Delta x^2, c) + d\Delta x^2\}^{-\frac{1}{2}}$
NN[SE]	$a[1 + f + bG(\Delta x^2, c/2) + dG(\Delta x^2, c)]^{-\frac{1}{2}}$

Table 2: List of compositional kernels by stacking two GPs with respective kernels denoted by SE (squared exponential), SC (squared cosine), Lin (linear), and NN (neural network with Gaussian activation function). The function symbol $G(x^2, c) = 1 - \exp(-x^2/c)$ appears with SE composition. The sequence can be understood from the notation, for example, SE[SC] means input data is directed to the first GP with SC kernels, followed by sending its output to the second GP with SE kernel which produces the final output. The hyperparameters are represented by the symbols a, b, c, d, and f, all of which are positive.

5 Interpretation of effectively deep GP

Variational inference, where one approximates the true distribution p with a simpler distribution q by minimizing the KL divergence, $\mathrm{KL}(p||q)$, provides an alternative view of our approximation. We approximate deep GP, p, as a GP, q, and moment matching then ensures that the KL divergence is minimized [18].

Our approximation based on GPs yields benefits in interpretability for both deep GPs and deep NNs. Prior results establish that deep NNs can be viewed as GPs [7, 10, 26]. In both the deep GP and deep NNs paradigms, people tend toward homogeneous kernels / activation functions, in part because there are not effective tools for predicting how compositions will behave a priori. In Table 2, we list a few representative analytic kernels from stacking two GPs. The analysis above allows iterative generation of heterogeneous kernels such as SE[SC[NN]] and homogeneous one SE[SE[SE[...]]] of any depth. Moreover, because our approach sits between these two paradigms, we can view our compositions as arising from activation functions [24], kernels [8], or compositions of both.

Unlike the neural network framework which generates only non-stationary kernels, such as arc-cosine kernel [10, 7], the present kernel composition in deep GP does not have such constraint. If stationary kernels are employed in first and second layers, such as SE[SE], then the resultant kernel is also stationary. In fact, one can directly generalize the present composition to any stationary kernel in the first GP. For example, one may construct SE[Matern], SE[RQ], SE[Exp], etc, and similarly for SC[$\cdot$]. Interestingly, our approach reproduces some well known kernels. Based on Equation (13), the periodic kernel proposed by MacKay [28] is equivalent to SC[SC]; following Equation (11) one can see that the rational quadratic kernel (RQ) is equivalent to SE[Lin]. Most strikingly, the fundamental SE kernel can be obtained by SC[Lin].

In principle, the composition is also applicable to any kernel, e.g. Matern[SE], RQ[SC], Brownian[NN], are also possible but one needs to solve the integral in Eq. 7 numerically.

It is worthwhile to note the order of composition matters from the perspective of our approximation. Consider the composition, SC[Lin[Lin]]. There are two ways to interpret. First, from inside out, SC[Lin[Lin]]$\rightarrow$ SC[Lin] $\rightarrow$ SE plus constant kernel. Second, from outside in, SC[Lin[Lin]]$\rightarrow$ SE[Lin] $\rightarrow$ $\mathrm{RQ}\frac{1}{2}$. The former interpretation is correct in the sense that the approximation is taken from inside to outside.

We may leverage interpretability to analyze differences in expressivity of SE[SE] and SE. First, SE[SE] approaches $\mathrm{RQ}\frac{1}{2}$ in small Δx limit. Because the rational quadratic kernel is known to arise from summing over infinitely many SE kernels with Gamma distribution over the inverse of length scale [18], the effective SE[SE] kernel also possesses the multi-length scale feature. Second, the function composition $y = f(h(x))$ leads to the long-range correlation, i.e. when $\Delta x \rightarrow \infty$, SE[SE] kernel approaches some nonzero constant while SE becomes vanishingly small. It can be seen by noting that the first-layer outputs $h(x_1)$ and $h(x_2)$ are nearly independent if x_1 and x_2 are distantly apart. Nevertheless, the distance $h(x_1) - h(x_2)$, generally falling within $[0, \sigma_0]$ with high probability,

fed into the second layer is greatly reduced if the signal magnitude σ_0 associated with h is small, which leads to the relevant correlation between $f(h(x_1))$ and $f(h(x_2))$.

The composition using non-stationary kernel is also interesting. SE and NN kernels are both Gaussian type. What can be observed in SE[] is the appearance of this linear combination $k(x,x)+k(x',x')-2k(x,x')$ (Eq. 11) from previous layers, which is also the case for SC[]. More importantly, the nonlinear combination $k(x,x)k(x',x')-k(x,x')^2$ appears because of the non-stationariness ($u \neq v$ in Eq. 10), and such was not found in the literature. Such term makes both $e^{-2(x-x')^2/c}$ and $e^{-(x-x')^2/c}$ appear in NN[SE] but does nothing to the linear kernel, i.e. NN[Lin] = SE[Lin].

5.1 Comparing with deep kernels of neural networks

Duvenaud et al. [27] studied pathologies of deep kernels in a deep neural network. They suggest that each layer of neural network serves as feature maps $\mathbf{h}(\mathrm{x})$ for the input x and the resultant covariance function $k(\mathrm{x},\mathrm{x}') \propto \mathbf{h}^T(\mathrm{x}')\mathbf{h}(\mathrm{x})$ following the Mercer's theorem. If one stacks a GP with squared exponential kernel on top of this neural network, the effective kernel function is proportional to $\exp\{-[k_L(x,x)+k_L(x',x')-2k_L(x,x')]\}$, whic can be compared with SE[NN] in our composition. This result apparently makes sense and the dependence on the kernel from previous layer is the same as Equation 11. Yet, our approach yields a rather different functional form. We attribute the difference to the deterministic nature of neural networks mapping in [27]. Here the hidden functions are marginalized out in the integral in Equation 7. It is also intriguing to notice that the effective kernel in Equation 13 obtained by placing the squared cosine GP on top of another GP is very similar with the deep kernel in [27] up to addition of constant kernel.

5.2 Characteristics of effective kernel

We consider the distribution over derivatives of the functions sampled from GP. By definition of Gaussian process, any two values f_1 and f_2 from a function $f(x)$ must follow the joint Gaussian distribution $\mathcal{N}(f_1,f_2|0,\Sigma)$ with the two-by-two covariance matrix Σ. In order to shed light on the expressiveness of our effective kernels, we calculate the following expectation,

$$\mathscr{E}[(f_1-f_2)^2] = \int df_1 df_2 (f_1-f_2)^2 \mathcal{N}(f_1,f_2|0,\Sigma)\,, \tag{16}$$

which shall asymptotically approach $\mathscr{E}\{[f'(x)]^2\}(x_1-x_2)^2$ as their the inputs x_1 and x_2 are close to each other. Consequently, we obtain the expectation value of squared derivative,

$$\mathscr{E}\{[f'(x)]^2\} = 2\lim_{x_1\to x_2}\frac{\sigma_1^2 - k_{\text{eff}}(x_1,x_2)}{(x_1-x_2)^2}\,. \tag{17}$$

When applying the above to the effective kernel in Equation 11, one may show that the effective kernel SE[SE] has

$$\mathscr{E}\{[f'(x)]^2\} = \frac{\sigma_0^2\sigma_1^2}{\ell_0^2\ell_1^2}\,. \tag{18}$$

SC[SC] results in the same conclusion. This characteristics of the derivative of our effective kernels are consistent with observed by [27].

6 Case study: empirical behavior of SE compositions

6.1 SE vs. SE[SE]

Figure 1(a)–(d) shows functions generated from SE and SE[SE]. The functions generated by SE[SE] seem to possess two length scales in variations: the rapidly varying blue line in Figure 1(d) seems to have slowly varying underlying trend that is similar to a shifted version of the black dash line. This is broadly consistent with the discussion in Section 5 of the additional expressivity of multi-length scale behavior in deeper structure.

To further demonstrate the expressivity of SE[SE] kernel over the SE kernel, we use them to fit a dataset with two length scales [1]. Figure 1(e) shows that the smooth SE kernel does not fit the small

[1] The data is from the example of FITRGP MATLAB. See `https://www.mathworks.com/help/stats/fitrgp.html`

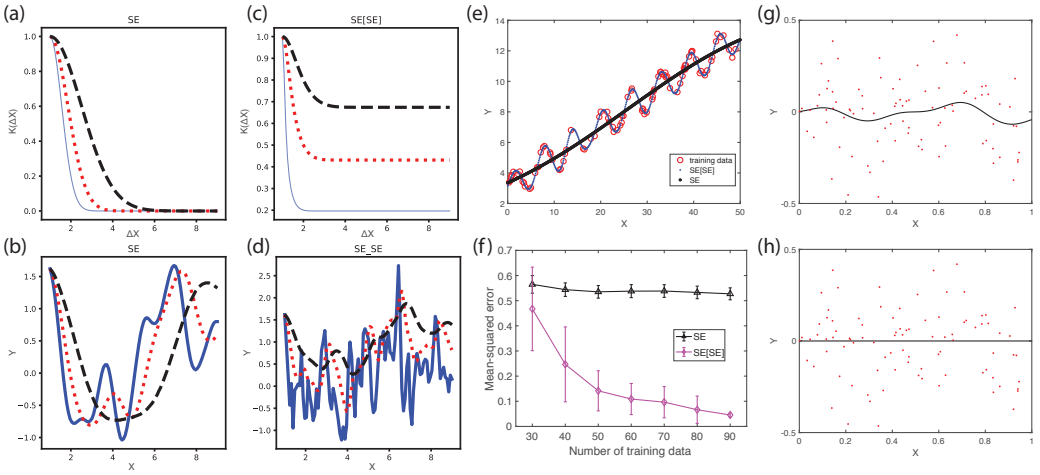

Figure 1: **(a)** Three different SE kernels and **(b)** functions sampled from these kernels. **(c)** Three different SE[SE] kernels and **(d)** functions sampled from these kernels. For (a) and (c), the kernel black dash, red dotted line, and blue solid line are 0.8, 0.5, and 0.2 at $\Delta x = 1$, respectively. All functions in (b) and (d) are generated using the same noise vector. **(e)-(f)** Regression on multiscale data with SE and SE[SE] kernels. **(g)-(h)** Regression with SE (g) and SE[SE] (h) kernels on pure noise data sampled from normal distribution with $\sigma = 0.2$. Red dots are data points, and black line is the predictive mean.

variation, while the SE[SE] kernel is able to capture these fast variation on top of the slowly varying one. Figure 1(f) shows that the SE[SE] kernel is better than SE kernel at prediction not only in the range of larger amount of data, but also in the limited data regime. We also remark in passing that the prediction accuracy for SE[SE] kernel is comparable with the RQ kernel, which is known to be multi-length scale, but with more stable hyperparameter optimization.

Given the ability of the SE[SE] kernel to capture fast variation, one might have the concern that it will fit to noise. To explore this possibility, we trained both SE and SE[SE] kernels on pure noise data. In Figure 1(g) and (h), we show the prediction mean (black) from training on 90 noise data points sampled from the normal distribution with $\sigma = 0.2$. For this particular noise data, the SE kernel in (g) generates a non-zero prediction mean, while the SE[SE] kernel in (f) nicely predicts the underlying zero function.

We also apply the SE and SE[SE] kernels to two UCI regression data sets, the House Price dataset and the Abalone dataset. We investigated the test error as a function of the fraction of training number. For the House Price dataset, the SE[SE] kernel obtain lower test error than the SE kernel does when the fraction of training data is larger than 30% (see Supplementary Material Figure 1(a)). For the Abalone dataset, the two kernels have similar test error (see Supplementary Material Figure 1(b)).

6.2 Deep SE compositions

Figure 2 shows kernels from different depth of SE compositions with randomly sampled hyperparameters. The first observation is that as the number of layers increase, the more likely an "L-shaped" kernel is sampled. For very deep constructions, the L-shape. kernels—a delta-function plus a constant, equivalent to a white noise kernel plus a constant kernel—pervade most of the hyperparameter space. This observation is consistent with an analysis that considers infinitely deep GP without the marginalization over the latent functions in each layer [27]. In that analysis, a constant function emerges from stacking infinitely many SE kernels one on top of the other, while a delta function emerges when every SE layer is connected to all subsequent layers in the infinitely deep structure.

The L-shaped kernel is often referred to as a pathology because functions sampled from such kernels are either constants or white noise are not very useful. [27] has proposed using an input-connected architecture to avoid this pathology. In Case 4 under Section 4, we mention the equivalent effective kernel of this architecture, which indeed produces fewer L-shaped kernels (rightmost panel of Figure 2).

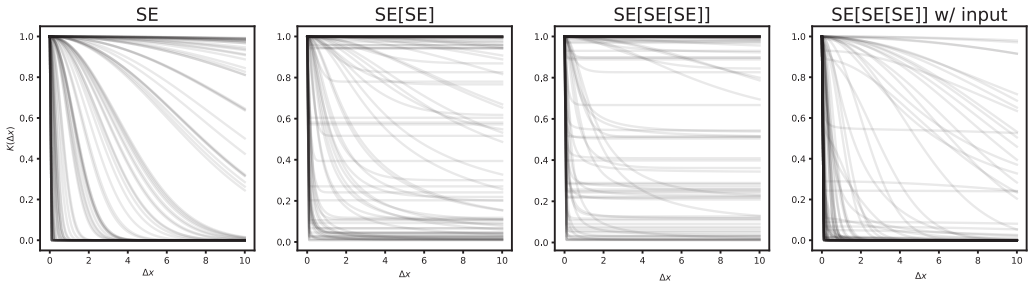

Figure 2: Kernels from different depth of SE compositions with randomly sampled hyperparameters. All hyperparameters are sampled uniformly from (-10,10) in log space, except for the parameter a (c.f., Table 2), which is set to $\exp(0) = 1$. The 'SE[SE[SE]] w/ input' kernel has input-connected layer as suggested by [27]. Plots for other kernels in Table 2 can be found in Supplementary Material Figure 2.

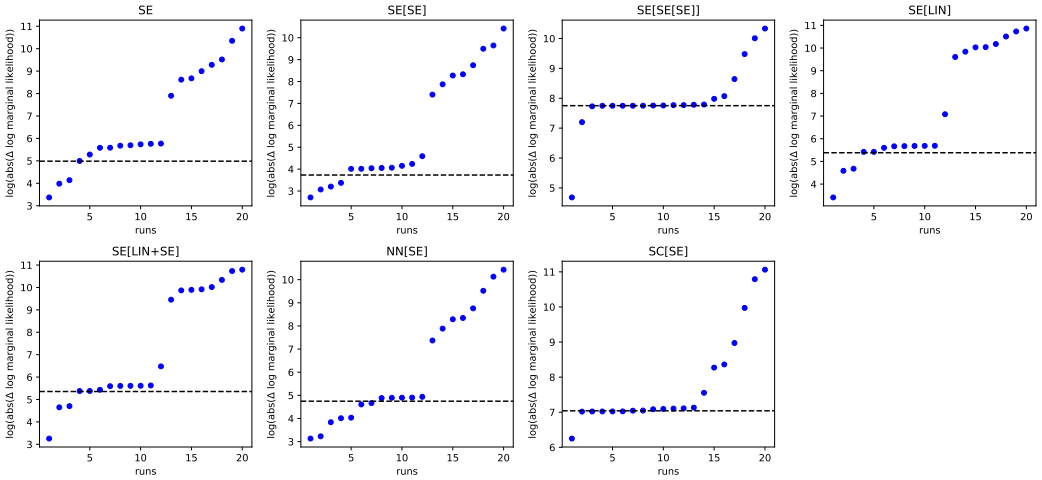

Figure 3: Effect of initialization on hyperparameter optimization. The title of each panel denotes the data-generating kernel. The hyperparameters that are optimized are those of the SE[SE[SE]]] kernel. The x-axis corresponds to different random initialization of the hyperparameters, sorted according to its y value. The y-axis is log(abs(Δ log marginal likelihood)), where Δ log marginal likelihood is defined to be the log marginal likelihood of the data under the data-generating kernel and hyperparameters minus that of the data under the optimized kernel and hyperparmaters. Smaller y values indicate that the optimized marginal likelihood is closer to the data's true marginal likelihood. The black dotted line corresponds to an initialization where the hyperparameters are chosen to avoid an L-shaped kernel. Ten different datasets are generated from the same data-generating kernel and hyperparameters. The y values plotted are the average over these 10 datasets.

On the other hand, one may ask whether, given the effectiveness of deeper models, pathology is the best interpretation. L-shaped kernels simply take the data at face value, and in this way may be interpreted as an inductive bias toward avoiding hasty generalization. Instead of viewing simplicity as something akin to smoothness in function space, which effectively asserts very strong a priori knowledge about the true function, deeper models may view no generalization beyond the data as simplest. Informally, we note that we have indeed observed cases where, for shallow models, optimization converges to settings that generalize from limited data, while for deeper models optimization converges to parameterizations that are L-shaped and do not generalize beyond the observed data.

6.3 Hyperparameter optimization

We expect that hyperparameter optimization be difficult because of the pervasiveness of the L-shaped kernels in deeper compositions. To test this, we generate data from the non-periodic kernels in Table 2. The hyperparameters of the data-generating kernel are chosen to satisfy $K(\Delta x = 0) = 1$ and $K(\Delta x = 1) = 0.8$ (e.g., Figure 1 (a) and (c) black dash lines), and we generate data from these

kernels (e.g., Figure 1 (b) and (d) black dash lines). We then optimize the hyperparameters of the SE[SE[SE]] kernel to fit the data by maximum marginal likelihood. We run the optimization 20 times with 20 random initialization where the hyperparameters are sampled uniformly from (-10,10) in log space. Figure 3 shows that different initialization can lead to different optimized values (blue dots), confirming that hyperparameter optimization can be difficult and is sensitive to initial condition.

The interpretability of our analytic forms allows us to efficiently identify settings of hyperparameters that are free from the L-shaped pathology. For example, constrained optimization can be used to find hyperparameters such that $K(\Delta = 0) > K(\Delta = a > 0)$ and $K(\Delta = a > 0) = b > 0$ efficiently. Marginal likelihood optimization under such initialization (Figure 3 black dotted line) is better (closer to the true marginal likelihood) than 75% to 95% of the randomly initialized optimization (Figure 3 blue dots). Furthermore, this initialization often causes the optimization to reach a more stable local minimum, as indicated by the black line being near a cluster of blue dots that have the same optimized marginal likelihood.

7 Conclusions

We have presented interpretable deep Gaussian Processes that combine increased expressiveness associated with deep NNs with uncertainty quantification of GPs. Our approach is based on approximating deep GPs as GPs, which enables one to analytically integrate yielding effectively deep, single layer kernels. We have provided a recipe for constructing effective kernels for cases including homogeneous and heterogeneous kernels (equivalently, activation functions), derived a variety of such kernels, analyzed their behavior, confirmed behavior by prior and posterior predictive simulation, and identified non-pathological regimes of hyperparameter behavior for both the prior and posterior predictive distributions. Simpler than alternative approaches to variational inference, our approach yields strong benefits in interpretability while retaining remarkable expressivity.

References

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[2] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[3] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. *arXiv preprint arXiv:1802.01396*, 2018.

[4] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[5] Madhu Advani, Subhaneil Lahiri, and Surya Ganguli. Statistical mechanics of complex neural systems and high dimensional data. *Journal of Statistical Mechanics: Theory and Experiment*, 2013(03):P03014, 2013.

[6] Zhenyu Liao and Romain Couillet. The dynamics of learning: a random matrix approach. *arXiv preprint arXiv:1805.11917*, 2018.

[7] Christopher KI Williams. Computing with infinite networks. In *Advances in neural information processing systems*, pages 295–301, 1997.

[8] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[9] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[10] Youngmin Cho and Lawrence K Saul. Kernel methods for deep learning. In *Advances in neural information processing systems*, pages 342–350, 2009.

[11] Julien Mairal, Piotr Koniusz, Zaid Harchaoui, and Cordelia Schmid. Convolutional kernel networks. In *Advances in neural information processing systems*, pages 2627–2635, 2014.

[12] Andrew G Wilson, Elad Gilboa, Arye Nehorai, and John P Cunningham. Fast kernel learning for multidimensional pattern extrapolation. In *Advances in Neural Information Processing Systems*, pages 3626–3634, 2014.

[13] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016.

[14] Mark Van der Wilk, Carl Edward Rasmussen, and James Hensman. Convolutional gaussian processes. In *Advances in Neural Information Processing Systems*, pages 2849–2858, 2017.

[15] Shengyang Sun, Guodong Zhang, Chaoqi Wang, Wenyuan Zeng, Jiaman Li, and Roger Grosse. Differentiable compositional kernel learning for gaussian processes. *arXiv preprint arXiv:1806.04326*, 2018.

[16] Sida Wang and Christopher Manning. Fast dropout training. In *international conference on machine learning*, pages 118–126, 2013.

[17] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[18] C. E. Rasmussen and C. K. I. Williams. *Gaussian Process for Machine Learning*. MIT press, Cambridge, MA, 2006.

[19] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[20] Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215, 2013.

[21] M. Titsias and N. Lawrence. Bayesian gaussian process latent variable model. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 844–851, 2010.

[22] 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*, pages 1472–1481, 2016.

[23] Hugh Salimbeni, Vincent Dutordoir, James Hensman, and Marc Peter Deisenroth. Deep gaussian processes with importance-weighted variational inference. *arXiv preprint arXiv:1905.05435*, 2019.

[24] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *NIPS*, 2016.

[25] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *NIPS*, 2016.

[26] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

[27] David Duvenaud, Oren Rippel, Ryan Adams, and Zoubin Ghahramani. Avoiding pathologies in very deep networks. In *Artificial Intelligence and Statistics*, pages 202–210, 2014.

[28] David JC MacKay. Introduction to gaussian processes. *NATO ASI Series F Computer and Systems Sciences*, 168:133–166, 1998.