Non-degenerate Priors for Arbitrarily Deep Networks

David Duvenaud
University of Cambridge
dkd23@cam.ac.uk

Oren Rippel
M.I.T., Harvard University
rippel@math.mit.edu

Ryan Adams
Harvard University
rpa@seas.harvard.edu

Zoubin Ghahramani
University of Cambridge
zoubin@eng.cam.ac.uk

Abstract

Choosing appropriate architectures and initialization strategies is crucial to good performance of deep networks. To shed light on this problem, we analyze the analogous problem of constructing useful priors on deep networks. We show that for typical deep architectures, the representational capacity of the network tends to capture fewer degrees of freedom as the number of layers increases, retaining only a single degree of freedom in the limit. We propose alternate priors on network structure which do not suffer from these pathologies. We also derive novel kernels obtained by composing infinitely-many feature transforms.

1 Introduction

Much recent work on deep networks has focused on weight initialization [1], regularization [2] and network architecture [3]. The interactions between these different design decisions can be complex and difficult to characterize. We propose to approach the design of deep architectures by examining the problem of creating priors on functions with desirable properties. Although inference in fully Bayesian models is generally difficult, well-defined priors allow us to characterize models in a data-independent way. Once we identify classes of priors with useful properties, these models may suggest regularization, initialization, and architecture choices with similar properties.

In this paper, we examine a simple and flexible class of priors on functions, Gaussian processes (GPs), and their extension into a deep architecture: Deep GPs [4]. Deep GPs are simply a prior on compositions of functions, each of which is distributed independently according to a GP prior:

$$f^{1:L}(\mathbf{x}) = f^L(f^{L-1}(\dots f^3(f^2(f^1(\mathbf{x})))\dots)) \qquad \text{where each } f^\ell \overset{\text{iid}}{\sim} \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}')) \tag{1}$$

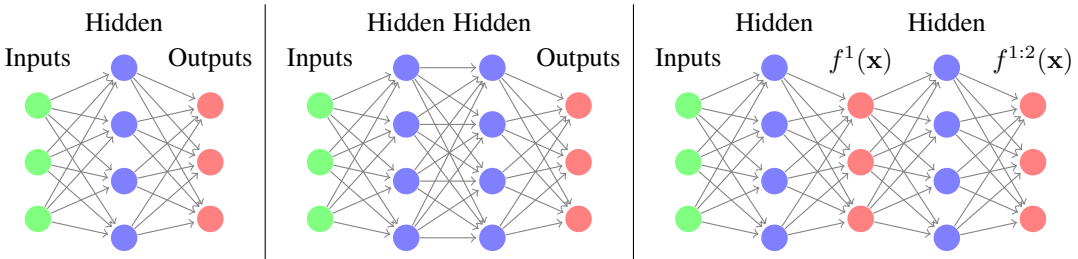
Although inference in these models is non-trivial, they can be derived as a special case of multi-layer perceptrons (MLPs), and so are a good candidate for generative models of functions with similar properties to existing neural networks.

2 Relating deep nets and deep Gaussian processes

2.1 A neural net with one hidden layer

In the typical definition of a multi-layer perception, the hidden units of the first layer are defined as:

$$\Phi(\mathbf{x}) = h^{(1)}(\mathbf{x}) = \sigma\left(b^{(1)} + W^{(1)}\mathbf{x}\right) \tag{2}$$



a) Single-layer MLP b) 2-Hidden layer MLP c) Two-layer GP: $\mathbf{y} = f^2(f^1(\mathbf{x}))$

Figure 1: Comparing architectures. In the deep GP models, there are two possible meanings for the hidden units. We can either consider every other layer to be a linear combination of an infinite number of parametric hidden units, or we can integrate out the hidden layers, and consider the deep GP to be a neural network with a finite number of hidden units, each with a different non-parametric activation function.

where h are the hidden unit activations, b is a bias vector, W is a weight matrix and σ is a sigmoidal function. The output vector $f(\mathbf{x})$ is simply a weighted sum of these hidden unit activations:

$$f(\mathbf{x}) = V\sigma\left(b^{(n)} + W^{(1)}h(\mathbf{x})\right) = Vh(\mathbf{x}) \tag{3}$$

where V is another weight matrix.

There exists a simple correspondence between one-layer MLPs and GPs [5]. GP priors can be viewed as a prior on neural networks with infinitely many hidden units. More precisely, for any model of the form

$$f(\mathbf{x}) = \frac{1}{K}\alpha^{\mathsf{T}}\Phi(\mathbf{x}) = \frac{1}{K}\sum_{i=i}^{K}\alpha_i\phi_i(\mathbf{x}), \tag{4}$$

with fixed feature vector $\Phi(x)$ and i.i.d. α's with zero mean and finite variance σ^2, the central limit theorem implies that as $K \to \infty$, any two function values $f(\mathbf{x}), f(\mathbf{x}')$ have a joint distribution approaching $\mathcal{N}\left(0, \frac{\sigma^2}{K}\sum_{i=1}^{K}\phi_i(\mathbf{x})\phi_i(\mathbf{x}')\right)$. The result is surprisingly general: It doesn't put any constraints on what the features are (other than having bounded activation), nor does it require that the feature weights α be Gaussian distributed.

We can also work backwards to derive a one-layer MLP from a GP: Mercer's theorem implies that any positive-definite kernel function corresponds to an inner product of features: $k(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^{\mathsf{T}}\Phi(\mathbf{x}')$. Thus in the one layer case, the correspondence between MLPs and GPs is simple: The features $\phi(\mathbf{x})$ of the GP correspond to the hidden units of the MLP.

2.2 Multiple hidden layers

In a neural net with multiple hidden layers, the correspondence is a little more complicated. In a MLP, the n^{th} layer units are given by the recurrence:

$$h^{(n)}(\mathbf{x}) = \sigma\left(b^{(n)} + W^{(n)}h^{(n-1)}(\mathbf{x})\right) \tag{5}$$

As shown in figure 1b. In this model, each hidden layer's output feeds directly into the next layer's input, weighted by the corresponding element of W.

However, in a deep GP, the D outputs $f^n(\mathbf{x})$ in between each layer are weighted sums of the hidden units of the layer below, and the next layer's hidden units depend only on these D outputs. Thus deep GPs have an extra set of layers that a MLP doesn't have, shown in figure 1c.

There are two ways to directly relate deep GPs to MLPs. First, we can note that, if the hidden units in a deep GP implied by Mercer's theorem $\phi^{(n)}(\mathbf{x})$ depend only on a linear projection of their inputs, as in the sigmoidal activation function $\phi(\mathbf{x}) = \sigma\left(b + W^{(n)}f^{n-1}(\mathbf{x})\right)$, then we can simply substitute for $f^{n-1} = V^{(n-1)}\phi^{(n-1)}(\mathbf{x})$ to recover $\phi(\mathbf{x}) = \sigma(b + W^{(n)}V^{(n-1)}\phi^{(n-1)}(\mathbf{x}))$. Thus, we can

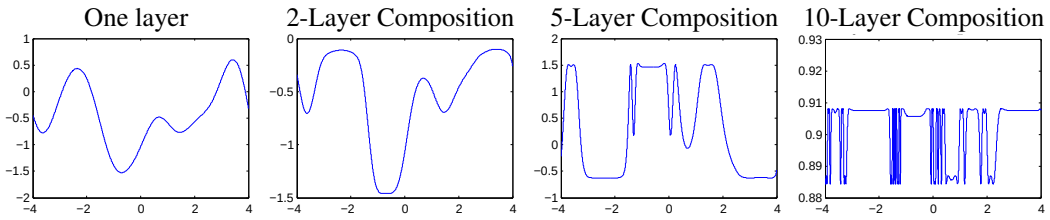


Figure 2: One-dimensional draws from a deep GP prior. After a few layers, the functions begin to be either nearly flat, or highly varying, everywhere. This is a consequence of the distribution on derivatives becoming heavy-tailed.

ignore the intermediate outputs $f(\mathbf{x})$, and exactly recover an MLP with activation functions given by Mercer's theorem, but with rank-D weight matrices between layers!

The second way, more general way we can relate the two model classes is to integrate out the basis functions, and view deep GP models as a neural network with a finite number of nonparametric basis functions, where the D outputs of $f^{1:\ell}(\mathbf{x})$ represent the output of the hidden nodes at the ℓ^{th} layer. This second view lets us compare GP models to multilayer perceptrons more directly, examining the activations and shapes of the finite number of basis functions at each layer.

3 One-dimensional Asymptotics

In this section, we characterize the limiting distribution of an arbitrarily deep, one-dimensional gp. Doing so will help characterize the ways in which a deep GP is non-Gaussian, and the ways in which lower layers affect the computation performed by higher layers. Figure 2 shows a draw from a 1D deep GP prior, at varying depths.

The derivative of a one-dimensional deep GP is simply a product of i.i.d Normal R.V.'s, each having zero mean and variance σ_f^2/ℓ^2. The distribution of the absolute value of this derivative is a product of half-normals, with mean $\sqrt{2\sigma_f^2/\pi\ell^2}$. Thus, if $\sigma_f^2/\ell_{d_1}^2 = \pi/2$, then $\mathbb{E}\left[\left|\partial f(x)/\partial x\right|\right] = 1$, and so $\mathbb{E}\left[\left|\partial f^{1:L}(x)/\partial x\right|\right] = 1$. If σ_f^2/ℓ^2 is less than $\pi/2$, the expected derivative magnitude goes to zero, and if it's greater, then the expected magnitude goes to infinity as a function of L.

The log of this variable has moments:

$$m_{\log} = \mathbb{E}\left[\log\left|\frac{\partial f(x)}{\partial x}\right|\right] = 2\log\left(\frac{\sigma_f}{\ell}\right) - \log 2 - \gamma \tag{6}$$

$$v_{\log} = \mathbb{V}\left[\log\left|\frac{\partial f(x)}{\partial x}\right|\right] = \frac{\pi^2}{4} + \frac{\log^2 2}{2} - \gamma^2 - \gamma\log 4 + 2\log\left(\frac{\sigma_f}{\ell}\right)\left[\gamma + \log 2 - \log\left(\frac{\sigma_f}{\ell}\right)\right]$$

where $\gamma \approxeq 0.5772$ is Euler's constant. Since the second moment is finite, by the central limit theorem, the limiting distribution of the size of the gradient is log-normal:

$$\log\left|\frac{\partial f^{1:L}(x)}{\partial x}\right| = \sum_{i=1}^{L}\log\left|\frac{\partial f^{i}(x)}{\partial x}\right| \implies \log\left|\frac{\partial f^{1:L}(x)}{\partial x}\right| \overset{L\to\infty}{\sim} \mathcal{N}\left(Lm_{\log}, L^2 v_{\log}\right) \tag{7}$$

Even if the expected magnitude of the derivative remains constant, the variance of the distribution grows without bound, and so will almost surely be very small almost everywhere, with rare but very large jumps.

4 The Jacobian of Deep GPs is a Product of i.i.d. Normal Matrices

In this section, we derive a result which will let us characterize the Jacobians of arbitrarily deep networks drawn from a deep GP prior.

Lemma 4.1. *The partial derivatives of a function drawn from a* GP *prior with a product kernel are i.i.d. Gaussian distributed.*

Proof. Because differentiation is a linear operator, the derivatives of a function drawn from a GP prior are also jointly Gaussian distributed. The covariance between partial derivatives w.r.t. input dimensions d_1 and d_2 of vector $\mathbf{x}$ are given by [6]:

$$\text{cov}\left(\frac{\partial f(\mathbf{x})}{\partial x_{d_1}}, \frac{\partial f(\mathbf{x})}{\partial x_{d_2}}\right) = \frac{\partial^2 k(\mathbf{x}, \mathbf{x}')}{\partial x_{d_1} \partial x'_{d_2}}\bigg|_{\mathbf{x}=\mathbf{x}'} \tag{8}$$

If our kernel is a product over individual dimensions $k(\mathbf{x}, \mathbf{x}') = \prod_d^D k_d(x_d, x'_d)$, as in the case of the squared-exp kernel, then the off-diagonal entries are zero, implying that all elements are independent and identically distributed. □

In the case of the multivariate squared-exp kernel, the covariance between derivatives has the form:

$$k_{SE}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \prod_{d=1}^{D} \left(-\frac{1}{2}\frac{(x_d - x'_d)^2}{\ell_d^2}\right) \implies \text{cov}\left(\frac{\partial f(\mathbf{x})}{\partial x_{d_1}}, \frac{\partial f(\mathbf{x})}{\partial x_{d_2}}\right) = \begin{cases} \frac{\sigma_f^2}{\ell_{d_1}^2} & \text{if } d_1 = d_2 \\ 0 & \text{if } d_1 \neq d_2 \end{cases} \tag{9}$$

Lemma 4.2. *The Jacobian of a* GP *with a product kernel is a matrix of i.i.d. Gaussian R.V.'s*

Proof. The Jacobian of the vector-valued function $f(\mathbf{x})$ is a matrix J whose $(i,j)^{\text{th}}$ element is $J_{ij} = \frac{\partial f_i(\mathbf{x})}{\partial x_j}$. When composing L different functions, we'll denote the *immediate* Jacobian of the function mapping from layer $\ell - 1$ to layer ℓ as $J^\ell(\mathbf{x})$, and the Jacobian of the entire composition of L functions by $J^{1:L}(\mathbf{x})$.

Because we've assumed that the GP on each output dimension $f_d(\mathbf{x}) \sim \mathcal{GP}$ are i.i.d. , it follows that for a given $\mathbf{x}$, each row of J is i.i.d. Lemma 4.1 shows that the elements of each row are i.i.d. Thus all entries in the Jacobian of a GP-distributed transform are i.i.d. Normal. □

Theorem 4.3. *The Jacobian of a deep* GP *with a product kernel is a product of i.i.d. Gaussian matrices.*

Proof. By the multivariate chain rule, the Jacobian of a composition of functions is simply the product of the Jacobian matrices of each function. Thus the Jacobian of the composed (deep) function $f^L(f^{L-1}(\dots f^3(f^2(f^1(\mathbf{x})))\dots))$ is:

$$J^{1:L}(x) = J^L J^{L-1} \dots J^3 J^2 J^1 \tag{10}$$

By Lemma 4.2, each $J_{i,j}^\ell \overset{\text{iid}}{\sim} \mathcal{N}\left(0, \frac{\sigma_f^2}{\ell^2}\right)$, so the complete Jacobian is a product of i.i.d. Gaussian matrices. □

Theorem 4.3 allows us to analyze the representational properties of a deep Gaussian process by simply examining the properties of products of i.i.d. Gaussian matrices.

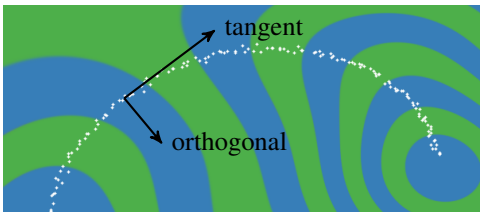
5 Formalizing the pathology

[7] argue that a good latent representation is invariant in directions orthogonal to the manifold on which the data lie. Conversely, a good latent representation must also change in directions tangent to the data manifold, in order to preserve relevant information. Figure 3 visualizes this idea. We follow [8] in characterizing the representational properties of a function by the singular value spectrum of the Jacobian[1].

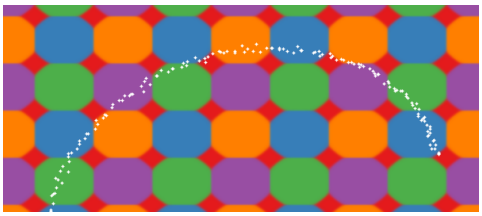
Figure 4 shows the spectrum for 5-dimensional deep GPs of different depths. As the net gets deeper, the largest singular value dominate, implying there is usually only one effective degree of freedom in representation being computed. As well, the distribution on singular vlaues becomes heavy-tailed.

Figure 5 demonstrates a related pathology that arises when composing functions to produce a deep density model. The density in observed space eventually becomes locally concentrated onto one-dimensional manifolds, or *filaments*.

[1] [8] examine the Jacobian at the training points, but the models we are examining are stationary, so it doesn't matter where we examine the function.



A noise-tolerant representation (blue & green)
of a one-dimensional manifold (white)



A naïve representation (colors)
of a one-dimensional manifold (white)

Figure 3: Comparing different representations of data on a 1-D manifold. Colors illustrate contours of the representation. A good representation is invariant in directions orthogonal to the data manifold, making the representation robust to noise in those directions. The representation must also change in directions tangent to the manifold, in order to preserve information for later layers.

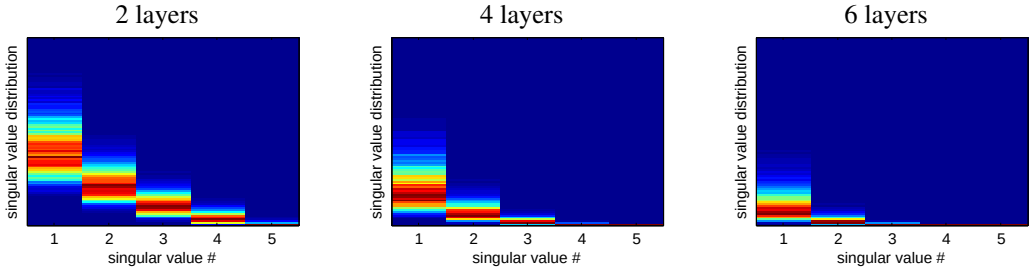
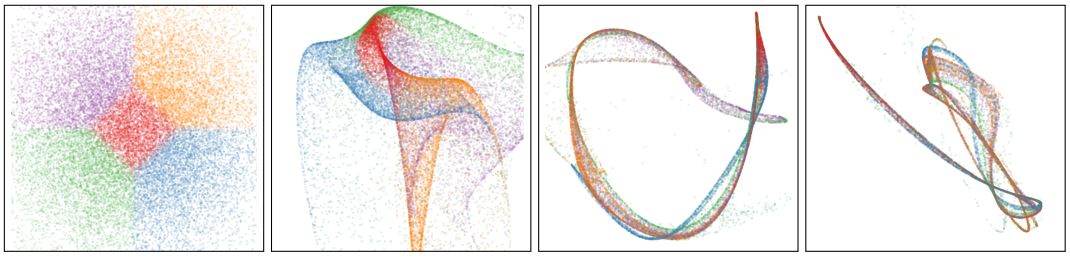


Figure 4: Normalized singular value spectrum of the Jacobian of a deep GP. As the net gets deeper, the largest singular value dominates. This implies that with high probability, there is only one effective degree of freedom in the representation being computed.



$p(\mathbf{x})$ $p(f^1(\mathbf{x}))$ $p(f^{1:4}(\mathbf{x}))$ $p(f^{1:6}(\mathbf{x}))$

Figure 5: Draws from a deep GP. A distribution is warped by successive functions drawn from a GP prior. As the number of layers increases, the density concentrates along one-dimensional filaments.

To visualize this pathology in another way, figure 6 illustrates the value that at each point in the input space is mapped to, after successive warpings. After 40 warpings, we can see that locally, there is usually only one direction that one can move in $\mathbf{x}$-space in order to change the value of the function.

To what extent are these pathologies present in nets being used today? In simulations, we found that for deep functions with a fixed latent dimension D, the singular value spectrum remained relatively flat for hundreds of layers as long as $D > 100$. Thus, these pathologies are unlikely to severely affect relatively shallow, wide networks.

6 Fixing the pathology

Follow a suggestion from [5], we can fix the pathologies exhibited in figures 5 and 6 by simply making each layer of depend not only on the output of the previous layer, but also on the original input $\mathbf{x}$. That is, $\forall L, f^{1:L}(\mathbf{x}) = f^L(f^{1:L-1}(\mathbf{x}), \mathbf{x})$. Draws from the resulting priors are shown in figures 7, 8 and 9.

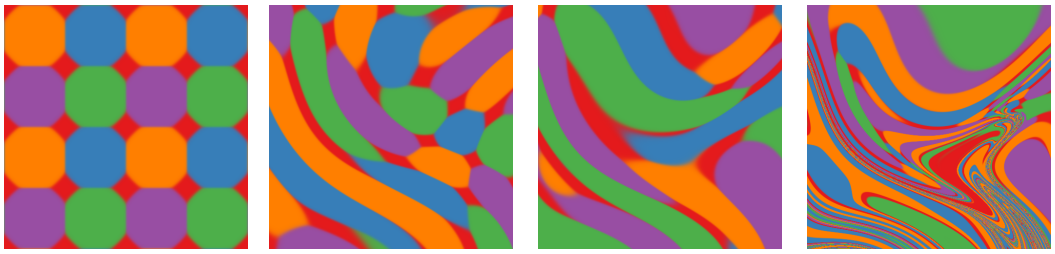


Figure 6: Feature Mapping of a deep GP. Shown here are the colors corresponding to the location $\mathbf{y} = f(\mathbf{x})$ that each point is mapped to after being warped by a deep GP. Just as the densities in 5 became locally one-dimensional, there is locally only one direction that one can move $\mathbf{x}$ in to change $\mathbf{y}$. Regions where the colors change rapidly indicate that the Jacobian is large in that area.

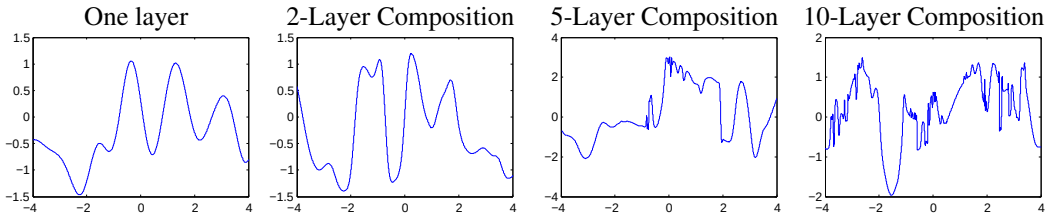


Figure 7: One-dimensional draws from a deep GP prior with eeach layer connected to the input. Even after many layers, the functions remain smooth in some regions, while varying rapidly in other regions.

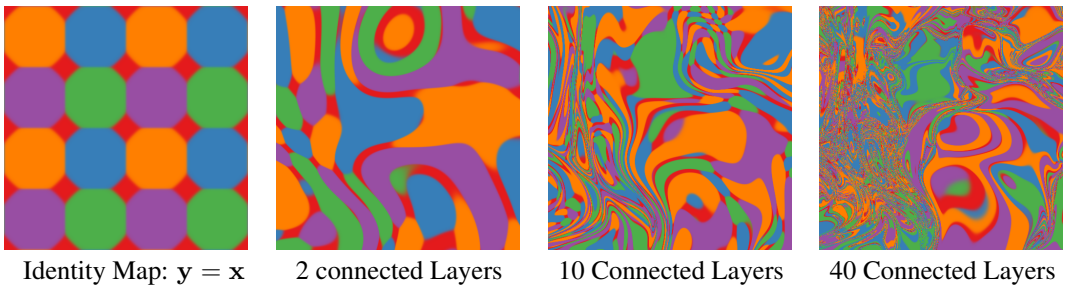


Figure 8: Feature Mapping of a deep GP with each layer connected to the input $\mathbf{x}$. Just as the densities in 9 became remain locally two-dimensional even after many transformations, in this mapping there are locally usually two directions that one can move in $\mathbf{x}$ to change $\mathbf{y}$.

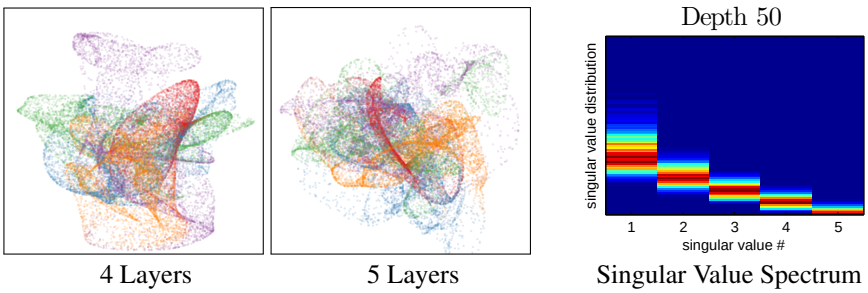


Figure 9: Left: Densities defined by a draw from a deep GP, with each layer connected to the input $\mathbf{x}$. As depth increases, the density becomes more complex without concentrating along filaments. Right: The singular value spectrum of a 5-dimensional deep GP prior 50 layers deep. The singular values remain roughly the same scale.

Jacobians of connected deep networks We can similarily examine the Jacobians of the new connected architecture. The Jacobian of a composite function which has had the original inputs appended $f_{\text{aug}}(\mathbf{x}) = [f(\mathbf{x}), \mathbf{x}]$, is $\begin{bmatrix} J_f \\ I_D \end{bmatrix}$ and the Jacobian of all one-layer transformations of these augmented functions are $D \times 2D$ matrices. The Jacobian of the composed, connected deep function is defined by the recurrence: $J^{1:L}(\mathbf{x}) = J^L \begin{bmatrix} J^{1:L-1} \\ I_D \end{bmatrix}$. Figure 9 shows that with this architecture, even 50-layer deep GPs have well-behaved singular value spectra.

7 Arbitarily Deep Kernels

[9] investigated kernels constructed by applying multiple layers of feature mappings. That is to say, if a kernel has the form $k_1(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^\mathsf{T}\Phi(\mathbf{x}')$, then we can consider constructing a kernel based on the repeated feature mapping: $k_2(\mathbf{x}, \mathbf{x}') = k_2(\Phi(\mathbf{x}), \Phi(\mathbf{x}')) = \Phi(\Phi(\mathbf{x}))^\mathsf{T}\Phi(\Phi(\mathbf{x}'))$.

For the squared-exp kernel, this composition operation has a closed form:

$$k_{n+1}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}||\Phi_n(\mathbf{x}) - \Phi_n(\mathbf{x}')||_2^2\right) \tag{11}$$

$$= \exp\left(-\frac{1}{2}\sum_i \left[\phi_n^{(i)}(\mathbf{x}) - \phi_n^{(i)}(\mathbf{x}')\right]^2\right) \tag{12}$$

$$= \exp\left(-\frac{1}{2}\left[k_n(\mathbf{x}, \mathbf{x}) - 2k_n(\mathbf{x}, \mathbf{x}') + k_n(\mathbf{x}', \mathbf{x}')\right]\right) \tag{13}$$

$$= \exp\left(k_n(\mathbf{x}, \mathbf{x}') - 1\right) \qquad (\text{if } k_n(\mathbf{x}, \mathbf{x}) = 1) \tag{14}$$

Note that this result holds for any base kernel k_n, as long as $k_n(\mathbf{x}, \mathbf{x}) = 1$.

Infinitely deep kernels What happens when we apply this compsition many times, starting with the squared-exp kernel? In the infinite limit, this recursion converges to $k(\mathbf{x}, \mathbf{x}') = 1$ for all pairs of inputs. Figure 10 shows this kernel at different depths, including the degenerate limit. One interpretation of why repeated feature transforms lead to this degenerate prior is that each layer can only lose information about the previous set of features. In the limit, the transformed features contain no information about the original input $\mathbf{x}$. Since the function doesn't depend on its input, it must be the same everywhere.

A non-degenerate construction Again, following a suggestion from [5], we connect the inputs $\mathbf{x}$ to each layer. To do so, we simply augment the feature vector $\Phi_n(\mathbf{x})$ with the $\mathbf{x}$ at each layer:

$$k_{n+1}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}\left|\left|\begin{bmatrix} \Phi_n(\mathbf{x}) \\ \mathbf{x} \end{bmatrix} - \begin{bmatrix} \Phi_n(\mathbf{x}') \\ \mathbf{x}' \end{bmatrix}\right|\right|_2^2\right) \tag{15}$$

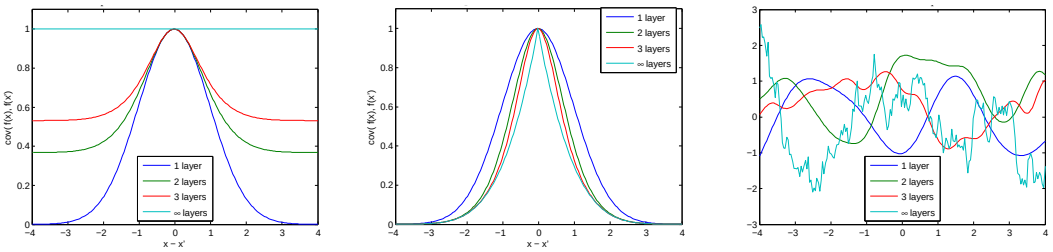
$$= \exp\left(-\frac{1}{2}\sum_i \left[\phi_i(\mathbf{x}) - \phi_i(\mathbf{x}')\right]^2 - \frac{1}{2}||\mathbf{x} - \mathbf{x}'||_2^2\right) \tag{16}$$

$$= \exp\left(k_n(\mathbf{x}, \mathbf{x}') - 1 - \frac{1}{2}||\mathbf{x} - \mathbf{x}'||_2^2\right) \tag{17}$$

This kernel satisfies the recurrence $k - \log(k) = 1 + \frac{1}{2}||\mathbf{x} - \mathbf{x}'||_2^2$, a non-degenerate limit. The solution to this recurrence has no closed form, but it is continuous and differentiable everywhere except at $\mathbf{x} = \mathbf{x}'$. Samples from a GP with this prior are not differentiable, and locally resemble brownian motion, having a sort of fractal behavior.

7.1 Can deep kernels be useful models?

[10] showed that kernel machines, such as GPs, have limited generalization ability when they use a local kernel such as the squared-exp. However, many interesting non-local kernels can be constructed which allow non-trivial extrapolation, for example, periodic kernels. A periodic kernel can



Iterated transform kernel | Connected transform kernel | Connected GP draws

Figure 10: Left: A kernel corresponding to composing features. In the limit, it becomes degenerate. Center: A non-degenerate version of the infinitely deep feature transform kernel. By connecting the inputs $\mathbf{x}$ to each layer, the function can still depend on its input even after arbitrarily many layers of computation. Right: Draws from this kernel.

be viewed as a 2-layer-deep kernel, in which the first layer maps the input $x \to [\sin(x), \cos(x)]$, and the second layer maps through a set of radial-basis functions.

Kernels can capture many other types of generalization, such as translation and rotation invariance in images [11]. [12] even uses a deep neural network to learn feature transforms for kernels, which learn invariances in an unsupervised manner. In contrast, the relatively uninteresting properties of the kernels derived in this section suggest that an arbitrary deep computation not necessarily powerful unless combined with learning. For any specific problem, an arbitrary set of computations are unlikely to lead to useful generalization.

8 Related Work

Variational inference in deep Gausian processes was developed by [4], who also analyzed the effect of ARD in these models. Besides deep GPs, other variants of deep models have been proposed, such as Bayesian Deep Networks [13].These models have no connections except between adjacent layers, and may also be expected to have similar pathologies to deep GP as the number of layers increases. Deep Density Networks [14] were constructed with invertibility in mind, with penalty terms encouraging the preservation of information about lower layers.

[15] perform a layer-wise analysis of deep networks, and note that the performance of MLPs degrades as the number of layers with random weights increases.

9 Conclusions

In this paper, we have shown the following propositions:

- Deep GPs can be viewed as MLPs with a finite number of nonparametric hidden units.
- Deep GPs can be chracterized using random matrix theory.
- Representations based on repeated composition of independently-initialized functions exhibit a pathology where the representation becomes invariant to all but one direction of variation.
- Connecting the input to each layer of a deep representation allows us to construct priors on deep functions without this pathology.

This analysis is simply a first step in constructing useful priors over deep functions. Future work could also include relating automatic relevance determination to sparse regularization methods, or variational inference in deep GPs to dropout-related regularization.

Acknowledgments

We thank Carl Rasmussen, Andrew McHutchon, Neil Lawrence, Andreas Diamanadou, James Lloyd, Creighton Heaukulani, Dan Roy and Mark van der Wilk for helpful discussions.

References

[1] James Martens. Deep learning via hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 735–742, 2010.

[2] Honglak Lee, Chaitanya Ekanadham, and Andrew Ng. Sparse deep belief net model for visual area v2. In *Advances in neural information processing systems*, pages 873–880, 2007.

[3] Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 689–690. IEEE, 2011.

[4] Andreas C Damianou and Neil D Lawrence. Deep gaussian processes. *arXiv preprint arXiv:1211.0358*, 2012.

[5] Radford M Neal. *Bayesian learning for neural networks*. PhD thesis, University of Toronto, 1995.

[6] Ercan Solak, Roderick Murray-Smith, E. Solak, William Leithead, Carl Rasmussen, and Douglas Leith. Derivative observations in gaussian process models of dynamic systems, 2003.

[7] Salah Rifai, Grégoire Mesnil, Pascal Vincent, Xavier Muller, Yoshua Bengio, Yann Dauphin, and Xavier Glorot. Higher order contractive auto-encoder. In *Machine Learning and Knowledge Discovery in Databases*, pages 645–660. Springer, 2011.

[8] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 833–840, 2011.

[9] Youngmin Cho. *Kernel methods for deep learning*. PhD thesis, University of California, San Diego, 2012.

[10] Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. The curse of highly variable functions for local kernel machines. pages 107–114, 2006.

[11] Imre Risi Kondor. *Group theoretical methods in machine learning*. 2008.

[12] Ruslan Salakhutdinov and Geoffrey Hinton. Using deep belief nets to learn covariance kernels for Gaussian processes. In *Advances in Neural Information Processing Systems*, volume 20, 2008.

[13] Ryan P Adams, Hanna M Wallach, and Zoubin Ghahramani. Learning the structure of deep sparse graphical models. In *International Conference on Artificial Intelligence and Statistics*, pages 1–8, 2010.

[14] Oren Rippel and Ryan Prescott Adams. High-dimensional probability estimation with deep density models. *arXiv preprint arXiv:1302.5125*, 2013.

[15] Grégoire Montavon, Dr Braun, Klaus-Robert Müller, et al. Layer-wise analysis of deep networks with gaussian kernels. *Advances in Neural Information Processing Systems (NIPS)*, 23:1678–1686, 2010.