On the Connection between Neural Processes and Gaussian Processes with Deep Kernels

Tim G. J. Rudner
OATML Group, Department of Computer Science
University of Oxford
tim.rudner@cs.ox.ac.uk

Vincent Fortuin
Department of Computer Science
ETH Zürich
fortuin@inf.ethz.ch

Yee Whye Teh
Department of Statistics
University of Oxford
y.w.teh@stats.ox.ac.uk

Yarin Gal
OATML Group, Department of Computer Science
University of Oxford
yarin@cs.ox.ac.uk

1 Introduction

Neural Processes (NPs) are a class of neural latent variable models that combine desirable properties of Gaussian Processes (GPs) and neural networks [5, 6]. Like GPs, NPs define distributions over functions and are able to estimate the uncertainty in their predictions. Like neural networks, NPs are computationally efficient during training and prediction time.

In this paper, we establish an explicit theoretical connection between NPs and GPs. In particular, we show that, under certain conditions, NPs are mathematically equivalent to GPs with deep kernels. This result further elucidates the relationship between GPs and NPs and makes previously derived theoretical insights about GPs applicable to NPs. Furthermore, it suggests a novel approach to learning expressive GP covariance functions applicable across different prediction tasks by training a deep kernel GP on a *set of datasets* [3, 7, 9, 10].

2 Background

2.1 Neural Processes

NPs are designed to learn distributions over functions from distributions over datasets. Consider a set of datasets, $\mathcal{D}$. For each dataset in $\mathcal{D}$ with input-output pairs $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$, we define a *context set*, $C = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{M}$, and a *target set*, $T = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$ with $C \not\subseteq T$ in general [6]. However, in practice we use $M \leq N$ so that $C \subseteq T$ [1, 5]. More compactly, for a given dataset in $\mathcal{D}$, we denote the context data by $\{\mathbf{X}_C, \mathbf{Y}_C\}$ and the target data by $\{\mathbf{X}_T, \mathbf{Y}_T\}$. For exposition, we first consider the case in which context and target data are identical, i.e., $M = N$, and simply denote inputs and outputs by $\mathbf{X}$ and $\mathbf{Y}$, respectively. To describe an NP, we define a Gaussian likelihood

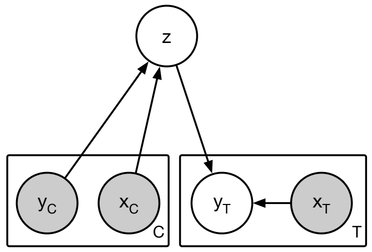
$$p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X}) = \mathcal{N}(\mathbf{Y}; g_\theta(\mathbf{Z}, \mathbf{X}), \tau^{-1}\mathbf{I}),$$

where $\mathbf{Z}$ is a latent variable and $g_\theta(\mathbf{Z}, \mathbf{X})$ is a *decoder* function parameterized by a deep neural network with parameters θ. For a standard Gaussian prior over $\mathbf{Z}$, $p(\mathbf{Z}) = \mathcal{N}(\mathbf{Z}; \mathbf{0}, \mathbf{I})$, the generative model of an NP is then given by

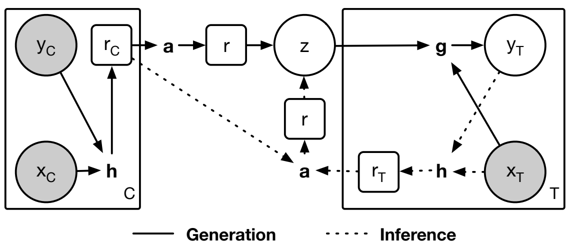
$$p(\mathbf{Y}, \mathbf{Z} \,|\, \mathbf{X}) = p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X}) p(\mathbf{Z}) = \mathcal{N}(\mathbf{Y}; g_\theta(\mathbf{Z}, \mathbf{X}), \tau^{-1}\mathbf{I})\, \mathcal{N}(\mathbf{Z}; \mathbf{0}, \mathbf{I}).$$

To perform approximate inference in NPs, a variational distribution is defined as

$$q(\mathbf{Z} \,|\, \mathbf{X}, \mathbf{Y}) = \mathcal{N}(\mathbf{Z}; \boldsymbol{\mu}_\omega(a(h_\psi(\mathbf{X}, \mathbf{Y}))), \boldsymbol{\Sigma}_\omega(a(h_\psi(\mathbf{X}, \mathbf{Y})))), \qquad (1)$$



(a) Graphical model of a Neural Process



(b) Computational diagram of a Neural Process

Figure 1: NPs include a global latent variable $\mathbf{Z}$ that captures information from the context data and informs predictions at the target test points (a). Computationally, the context data information is aggregated into a joint representation r, which influences the distribution over $\mathbf{Z}$ (b). Note that during training time (here denoted as *Inference*), the choice of $\mathbf{Z}$ is also informed by the target data. Figures taken from [6].

where $h_\psi(\cdot)$ is an *encoder* function parameterized by a deep neural network with parameters ψ, $a(\cdot)$ is an *aggregator* function, and $\mu_\omega(\cdot)$ and $\Sigma_\omega(\cdot)$ take aggregated and encoded input-output pairs as inputs and parameterize a normal distribution from which $\mathbf{Z}$ is sampled [6]. Intuitively, the latent variable $\mathbf{Z}$ is designed to capture all the information about the data-generating process needed to make predictions on the target inputs. Using the variational distribution in Eq. (1), we obtain an evidence lower bound (ELBO) on the log marginal likelihood given by

$$\log p(\mathbf{Y} \,|\, \mathbf{X}) \geq \mathbb{E}_{q(\mathbf{Z} \,|\, \mathbf{X})}[\log p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X})] - \mathrm{KL}(q(\mathbf{Z} \,|\, \mathbf{X}) \,||\, p(\mathbf{Z})),$$

In an alternative objective that better reflects the desired model behavior of an NP we assume $C \subseteq T$ and model the target points given the context data, which yields the ELBO

$$\log p(\mathbf{Y}_T \,|\, \mathbf{X}_T, \mathbf{X}_C, \mathbf{Y}_C) \geq \mathbb{E}_{q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T)}[\log p(\mathbf{Y}_T \,|\, \mathbf{Z}, \mathbf{X}_T)] - \mathrm{KL}(q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T) \,||\, p(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)),$$

where the prior $p(\mathbf{Z}) = \mathcal{N}(\mathbf{Z}; \mathbf{0}, \mathbf{I})$ is replaced by the conditional prior $p(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)$. Approximating the intractable conditional prior by $q(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)$, the NP objective becomes

$$\begin{aligned}\log p(\mathbf{Y}_T \,|\, \mathbf{X}_T, \mathbf{X}_C, \mathbf{Y}_C) \geq\ & \mathbb{E}_{q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T)}[\log p(\mathbf{Y}_T \,|\, \mathbf{Z}, \mathbf{X}_T)] \\ & - \mathrm{KL}(q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T) \,||\, q(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)).\end{aligned} \tag{2}$$

2.2 Gaussian Processes with Deep Kernels

We consider a set of N observations $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_N]^\top \in \mathbb{R}^{N \times D}$ at input locations $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times P}$, a function $f : \mathbb{R}^P \to \mathbb{R}^D$, and a likelihood $p(\mathbf{Y} \,|\, \mathbf{F}; \mathbf{X})$, with $\mathbf{F} = f(\mathbf{X})$ denoting the function values at the input locations. To infer f, a GP prior is placed on the function f, which models all function values as jointly Gaussian and has covariance function $K : \mathbb{R}^P \times \mathbb{R}^P \to \mathbb{R}^D$ and mean function $m : \mathbb{R}^P \mapsto \mathbb{R}^D$. The generative model of the corresponding GP is then given by

$$\begin{aligned} p(\mathbf{F} \,|\, \mathbf{X}) &= \mathcal{N}(\mathbf{F}; m(\mathbf{X}), K(\mathbf{X}, \mathbf{X})) \\ p(\mathbf{Y} \,|\, \mathbf{F}) &= \mathcal{N}(\mathbf{Y}; \mathbf{F}, \tau^{-1}\mathbf{I}), \end{aligned}$$

where τ^{-1} is the precision of the distribution.

We follow [8] and define a parametric covariance function between different input locations as the finite rank covariance function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{M} \sum_{m,m'=1}^{M} \sigma(\mathbf{w}_m^\top \mathbf{x}_i + b_m)\ \Sigma_{mm'}\ \sigma(\mathbf{w}_{m'}^\top \mathbf{x}_j + b_m), \tag{3}$$

with M denoting the number of hidden units in a single hidden layer neural network parameterized by the parameters $[\mathbf{w}_m]_{m=1}^M$ and $[b_m]_{m=1}^M$, $\sigma(\cdot)$ denoting some nonlinear function, and requiring $\mathbf{\Sigma} \in \mathbb{R}^{M \times M}$ to be positive semidefinite. We will only consider single-layer neural networks to maintain notational simplicity in the exposition, but it is straightforward to extend this covariance function to deep neural network architectures [4]. We will refer to GPs with deep parametric covariance functions as *deep kernel* GPs.

To write the covariance function in Eq. (3) as a product of matrices, we let

$$\phi(\mathbf{x}_i, \mathbf{W}, \mathbf{b}) = \sqrt{\frac{1}{M}}\sigma(\mathbf{W}^\top \mathbf{x}_i + \mathbf{b}),$$

for $i = 1, ..., N$, and $\mathbf{\Phi} = [\phi(\mathbf{x}_n, \mathbf{W}, \mathbf{b})]_{n=1}^N \in \mathbb{R}^{N \times M}$ to get $K(\mathbf{X}, \mathbf{X}) = \mathbf{\Phi\Sigma\Phi}^\top$ and define the mean function to be

$$m(\mathbf{X}) = \mathbf{\Phi}\boldsymbol{\mu},$$

for some $\boldsymbol{\mu} \in \mathbb{R}^{M \times D}$. We then analytically marginalize the latent function f out from the joint distribution $p(\mathbf{Y}, \mathbf{F} \,|\, \mathbf{X})$, which yields the marginal likelihood

$$p(\mathbf{y}_d \,|\, \mathbf{X}) = \int p(\mathbf{y}_d, \mathbf{f}_d \,|\, \mathbf{X})\, \mathrm{d}\mathbf{f}_d = \int p(\mathbf{y}_d \,|\, \mathbf{f}_d)\; p(\mathbf{f}_d \,|\, \mathbf{X})\, \mathrm{d}\mathbf{f}_d = \mathcal{N}(\mathbf{y}_d; \mathbf{\Phi}\boldsymbol{\mu}, \mathbf{\Phi\Sigma\Phi}^\top + \tau^{-1}\mathbf{I}_N). \tag{4}$$

To relate the generative model of deep kernel GPs to that of the NPs, we introduce an auxiliary (latent) random variable drawn from an M-dimensional Gaussian distribution, $\mathbf{z}_d \sim \mathcal{N}(\mathbf{z}_d; \boldsymbol{\mu}, \mathbf{\Sigma})$, with $\mathbf{Z} = [\mathbf{z}_d]_{d=1}^D \in \mathbb{R}^{M \times D}$. We can then use Gaussian conditioning to find the likelihood of $\mathbf{Y}$ given the latent variables, $p(\mathbf{y}_d \,|\, \mathbf{X}, \mathbf{z}_d)$, by writing the marginal distribution, $p(\mathbf{y}_d \,|\, \mathbf{X})$, as an integral of the joint distribution of $p(\mathbf{y}_d \,|\, \mathbf{X}, \mathbf{z}_d)$ and $p(\mathbf{z}_d)$ over $\mathbf{z}_d$, i.e.,

$$\begin{aligned} p(\mathbf{y}_d \,|\, \mathbf{X}) = \mathcal{N}(\mathbf{y}_d; \mathbf{\Phi}\boldsymbol{\mu}, \mathbf{\Phi\Sigma\Phi}^\top + \tau^{-1}\mathbf{I}_N) &= \int p(\mathbf{y}_d \,|\, \mathbf{X}, \mathbf{z}_d)\; p(\mathbf{z}_d)\, \mathrm{d}\mathbf{z}_d \\ &= \int \mathcal{N}(\mathbf{y}_d; \mathbf{\Phi}\, \mathbf{z}_d, \tau^{-1}\mathbf{I}_N)\; \mathcal{N}(\mathbf{z}_d; \boldsymbol{\mu}, \mathbf{\Sigma})\, \mathrm{d}\mathbf{z}_d, \end{aligned}$$

a short proof of which is given in the appendix. We note that if the distribution of the auxiliary variable were given by $\mathbf{z}_d \sim \mathcal{N}(\mathbf{z}_d; \mathbf{0}, \mathbf{I}_M)$, the mean and the variance of the GP prior would be $m(\mathbf{X}) = \mathbf{0}$ and $K(\mathbf{X}, \mathbf{X}) = \mathbf{\Phi\Phi}^\top$, and we would have $p(\mathbf{y}_d \,|\, \mathbf{X}) = \mathcal{N}(\mathbf{y}_d; \mathbf{0}, \mathbf{\Phi\Phi}^\top + \tau^{-1}\mathbf{I}_N)$. Unfortunately, computing the marginal likelihood in Eq. (4) requires inversion of the $N \times N$-dimensional covariance matrix $\mathbf{\Phi\Sigma\Phi}^\top + \tau^{-1}\mathbf{I}_N$ and thus scales cubically in the number of input points.

To avoid computing the marginal likelihood in Eq. (4) analytically and achieve better scalability, we can perform approximate inference in the above deep kernel GP model. To do so, we assume a mean-field variational distribution

$$q(\mathbf{Z} \,|\, \mathbf{X}) = \prod_{d=1}^D q(\mathbf{z}_d \,|\, \mathbf{X}),$$

and derive an evidence lower bound on the log marginal likelihood $\log p(\mathbf{Y} \,|\, \mathbf{X})$,

$$\log p(\mathbf{Y} \,|\, \mathbf{X}) \geq \mathbb{E}_{q(\mathbf{Z} \,|\, \mathbf{X})}[\log p(\mathbf{Y} \,|\, \mathbf{Z}, \mathbf{X})] - \mathrm{KL}(q(\mathbf{Z} \,|\, \mathbf{X}) \,||\, p(\mathbf{Z})).$$

3 Neural Processes as Gaussian Processes with Deep Kernels

In this section, we will establish an explicit connection between NPs and deep kernel GPs. To do so, we return to the marginal distribution of the full deep kernel GP given in Eq. (4),

$$p(\mathbf{y}_d \,|\, \mathbf{X}) = \int p(\mathbf{y}_d \,|\, \mathbf{z}_d, \mathbf{X})\; p(\mathbf{z}_d)\, \mathrm{d}\mathbf{z}_d = \int \mathcal{N}(\mathbf{y}_d; \mathbf{\Phi}\, \mathbf{z}_d, \tau^{-1}\mathbf{I})\; \mathcal{N}(\mathbf{z}_d; \boldsymbol{\mu}, \mathbf{\Sigma})\, \mathrm{d}\mathbf{z}_d\, .$$

Now, making the same distinction between context and target data as we did for NPs, conditioning on the context data, and performing variational inference over $\mathbf{z}_d$, the ELBO on the log marginal likelihood of the deep kernel GP becomes

$$\log p(\mathbf{Y}_T \,|\, \mathbf{X}_T, \mathbf{X}_C, \mathbf{Y}_C) \geq \mathbb{E}_{q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T)}[\log p(\mathbf{Y}_T \,|\, \mathbf{Z}, \mathbf{X}_T)] - \mathrm{KL}(q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T) \,||\, p(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)),$$

for which we define the conditional, "data-driven" prior to be the distribution of the auxiliary variable introduced in the previous section, with its mean and variance estimated from the context data, i.e.

$$p(\mathbf{z}_d \,|\, \mathbf{X}_C, \mathbf{Y}_C) = \mathcal{N}(\mathbf{z}_d; \boldsymbol{\mu}(\mathbf{X}_C, \mathbf{Y}_C), \mathbf{\Sigma}(\mathbf{X}_C, \mathbf{Y}_C)).$$

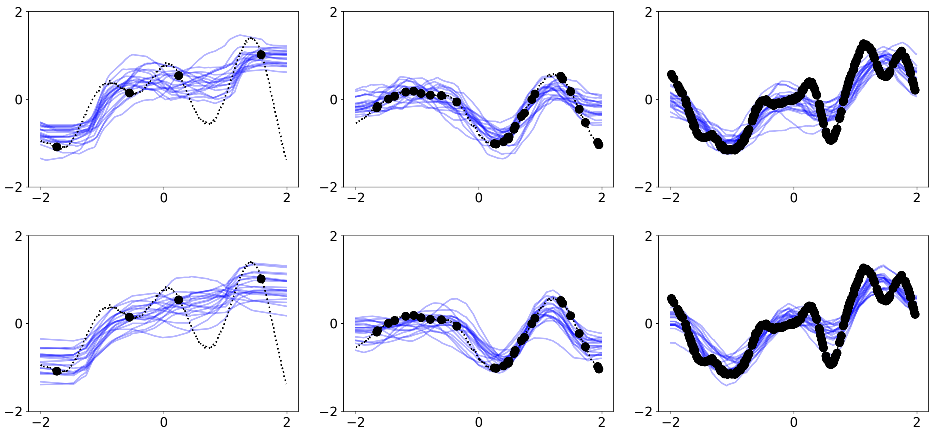


Figure 2: The plots show samples of curves drawn from NPs with a standard decoder (top row) and with an affine decoder (bottom row) conditioned on an increasing number of context points. Each figure shows 20 draws from the respective $\mathbf{z}$ distributions. Details about the experimental setup can be found in the appendix.

Approximating $p(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)$ the same way as for NPs, i.e. letting $p(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C) = q(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)$, the objective of the approximate deep kernel GP becomes

$$\log p(\mathbf{Y}_T \,|\, \mathbf{X}_T, \mathbf{X}_C, \mathbf{Y}_C) \geq \mathbb{E}_{q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T)}[\log p(\mathbf{Y}_T \,|\, \mathbf{Z}, \mathbf{X}_T)] - \mathrm{KL}(q(\mathbf{Z} \,|\, \mathbf{X}_T, \mathbf{Y}_T) \,||\, q(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)),$$

which is identical to the ELBO of an NP, given in Eq. (2), if the NP and GP generative models are identical. Comparing the two generative models described in the previous section, we see that they are in fact identical if the decoder function is defined to be an affine transformation of the form

$$g_\theta(\mathbf{Z}, \mathbf{X}_T) = \mathbf{\Phi}_\Theta(\mathbf{X}_T)\, \mathbf{Z},$$

where $\Theta = \{\mathbf{W}^\ell, \mathbf{b}^\ell\}_{\ell=1}^{L}$ parameterize an L-layer deep neural network $\mathbf{\Phi}_\Theta(\cdot)$.

In other words, GPs with (i) a covariance function parameterized by a deep neural network and (ii) a conditional prior $p(\mathbf{Z} \,|\, \mathbf{X}_C, \mathbf{Y}_C)$, trained in the same manner as NPs, are mathematically equivalent to neural processes with decoder functions of the form $g(\mathbf{Z}, \mathbf{X}_T) = \mathbf{\Phi}(\mathbf{X}_T)\, \mathbf{Z}$. In particular, exact inference in affine-decoder NPs is possible and identical to exact inference in deep kernel GPs. Furthermore, approximate inference in affine-decoder NPs is mathematically equivalent to approximate inference in deep kernel GPs.

To assess whether affine-decoder NPs exhibit model behavior similar to that of non-affine-decoder NPs, we consider a 1D-regression task reminiscent of the 1D-regression experiment presented in [6]. We trained affine- and non-affine-decoder NPs on a set of datasets, each containing function draws from different GP priors, and subsequently made predictions on unseen test curves. The results of the experiment are shown in Fig. 2.

On this simple regression task, we find that affine- and non-affine-decoder NPs qualitatively show the same model behavior in all three data regimes. The wiggliness of the generated functions and the predictive accuracy of the two models are comparable, and the predictive variance decreases in the number of context points for both of them. This result provides some evidence for the possibility that affine-decoder NPs may be applicable to the same problems as non-affine-decoder NPs, with the additional benefit of being equivalent to approximate deep kernel GPs.

4 Conclusion

We established an explicit connection between NPs and GPs with deep kernels. Given this connections, we speculate that GPs that satisfy the conditions above may share many of the properties of non-affine-decoder NPs. In particular, the relationship between NPs and GPs begs the question if an NP-style training procedure on a *set of datasets* would make it possible for approximate deep kernel GPs to effectively learn generalization across prediction tasks the same way NPs do. If so, this approach could help solve the longstanding problem of covariance function selection in GP models.

Acknowledgements

TGJR acknowledges financial support from the Rhodes Trust and the UK Engineering and Physical Sciences Research Council. VF is supported by ETH core funding. YWT's research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 617071. YG acknowledges support from Intel and Nvidia.

References

[1] Anonymous. Attentive neural processes. In *Submitted to International Conference on Learning Representations*, 2019. Under Review.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, Berlin, Heidelberg, 2006.

[3] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1166–1174, 2013.

[4] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*, 2016.

[5] Marta Garnelo, Dan Rosenbaum, Chris J Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo J Rezende, and SM Eslami. Conditional neural processes. *arXiv preprint arXiv:1807.01613*, 2018.

[6] Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

[7] Shengyang Sun, Guodong Zhang, Chaoqi Wang, Wenyuan Zeng, Jiaman Li, and Roger B. Grosse. Differentiable compositional kernel learning for gaussian processes. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, 2018.

[8] Koji Tsuda, Taishin Kin, and Kiyoshi Asai. Marginalized kernels for biological sequences. *Bioinformatics*, 2002.

[9] Andrew G Wilson, Zhiting Hu, Ruslan R Salakhutdinov, and Eric P Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pages 2586–2594, 2016.

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

Appendix A Experimental Details

For the 1D-regression experiment presented in Fig. 2, both affine and non-affine-decoder NPs are trained on functions sampled from GP priors with RBF kernels,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f \ \exp\left(\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2\ell^2}\right),$$

generated by varying lengthscales ℓ and function noise levels σ_f at random according to $\ell \sim \mathcal{U}(0.1, 0.6)$ and $\sigma_f \sim \mathcal{U}(0.1, 1.0)$. In other words, we sample pairs of kernel hyperparameters, use them to define GP priors, and then draw curves from each of them. This way, we create a set of datasets, with each dataset containing curves drawn from a different GP prior.

For both affine and non-affine-decoder NPs, we use a network with three layers and 128 units per layer to model the encoder function $h_\psi(\cdot, \cdot)$. To estimate the mean and variance of the variational distribution, i.e., $\boldsymbol{\mu}_\omega(\cdot)$ and $\boldsymbol{\Sigma}_\omega(\cdot)$, we use a two-layer network with 128 units per layer for each. To model the decoder function of the non-affine-decoder NP, we use a three-layer network with 128 units per layer, $g_\theta(\mathbf{Z}, \mathbf{X}_T)$, and to model the decoder function of the affine NP, we use a three-layer network with 128 units per layer, $\boldsymbol{\Phi}_\Theta(\mathbf{X})$, that only takes $\mathbf{X}$ as input.

After training each model for 200,000 iterations with a learning rate of 10^{-3}, we made predictions on unseen test curves by sampling 20 different curves for each.

Appendix B Gaussian Conditioning

Claim. The marginal distributions over $\mathbf{y}_d$ of the prior distribution

$$p(\mathbf{z}_d) = \mathcal{N}(\mathbf{z}_d; \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

and the conditional distribution

$$p(\mathbf{y}_d \,|\, \mathbf{X}, \mathbf{z}_d) = \mathcal{N}(\mathbf{y}_d; \boldsymbol{\Phi}\, \mathbf{z}_d, \tau^{-1}\mathbf{I}_N)$$

is given by

$$p(\mathbf{y}_d \,|\, \mathbf{X}) = \mathcal{N}(\mathbf{y}_d; \boldsymbol{\Phi}\boldsymbol{\mu}, \boldsymbol{\Phi}\boldsymbol{\Sigma}\boldsymbol{\Phi}^\top + \tau^{-1}\mathbf{I}_N).$$

Proof.

The claim follows from Gaussian conditioning [2]. In general, for random variables $\mathbf{z}$ and $\mathbf{y}$ with

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z} \,|\mathbf{m}, \boldsymbol{\Lambda}^{-1})$$

and

$$p(\mathbf{y} \,|\, \mathbf{z}) = \mathcal{N}(\mathbf{A}\mathbf{z} + \mathbf{b}, \mathbf{L}^{-1}),$$

the marginal distribution over $\mathbf{y}$ is given by

$$\begin{aligned} p(\mathbf{y}) &= \int p(\mathbf{y}, \mathbf{z})\, \mathrm{d}\mathbf{z} \\ &= \int p(\mathbf{y} \,|\, \mathbf{z}) p(\mathbf{z})\, \mathrm{d}\mathbf{z} \\ &= \int \mathcal{N}(\mathbf{y}; \mathbf{A}\mathbf{z} + \mathbf{b}, \mathbf{L}^{-1}) \mathcal{N}(\mathbf{z} \,|\mathbf{m}, \boldsymbol{\Lambda}^{-1})\, \mathrm{d}\mathbf{z} \\ &= \mathcal{N}(\mathbf{y}; \mathbf{A}\mathbf{m} + \mathbf{b}, \mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^\top + \mathbf{L}^{-1}) \end{aligned}$$

The claim follows directly for

$$\begin{aligned} \mathbf{A} &= \boldsymbol{\Phi} \\ \mathbf{b} &= \mathbf{0} \\ \mathbf{L}^{-1} &= \tau^{-1}\mathbf{I} \\ \mathbf{m} &= \boldsymbol{\mu} \\ \boldsymbol{\Lambda}^{-1} &= \boldsymbol{\Sigma}. \end{aligned}$$