A Interpretation of lengthscale parameters

First, we start with the stationary case. There exist stationary kernels k which can be represented as:

$$k(\boldsymbol{a}, \boldsymbol{b}) = \pi_k\big((\boldsymbol{a} - \boldsymbol{b})^\intercal \boldsymbol{\Delta}^{-1} (\boldsymbol{a} - \boldsymbol{b})\big), \tag{10}$$

for a given scalar function π_k and lengthscale parameter $\boldsymbol{\Delta}$. In this case, the lengthscale controls the spatial variation of the Gaussian process with that kernel. More concretely, in the 1D case, using scalar lengthscale $\boldsymbol{\Delta} = \ell^2$ and the squared exponential kernel:

$$k_{\text{SE}}(a, b) = \exp\left[-\frac{1}{2}\frac{(a-b)^2}{\ell^2}\right], \tag{11}$$

we have that the corresponding marginals for $f \sim \mathcal{GP}(0(\cdot), k_{\text{SE}})$ are:

$$f(x) \sim \mathrm{N}(0, 1), \tag{12}$$

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) \sim \mathrm{N}\left(0, \frac{1}{\ell^2}\right). \tag{13}$$

So the lengthscale parameter directly controls the amplitude of the gradient's variance.

In general, non-stationary kernels do not have a corresponding concept, which attention is given to kernels that, in the neighborhood of a point $\boldsymbol{x}$, can be expressed in terms of a local lengthscale matrix $\boldsymbol{\Delta}(\boldsymbol{x})$. The lengthscale mixture kernels k_{lmx} described in Section 2 and our proposed kernel from Section 3, k_{TDGP} both have this local lengthscale property.

Again, assuming inputs are 1D and the base kernel is squared exponential, both kernels are:

$$k_{\text{lmx}}(a, b) = \ell(a)^{\frac{2}{4}} \ell(b)^{\frac{2}{4}} \left[\frac{\ell(a)^2 + \ell(b)^2}{2}\right]^{-\frac{1}{2}} \exp\left[-\frac{1}{2}\frac{(a-b)^2}{\frac{\ell(a)^2 + \ell(b)^2}{2}}\right],$$

$$k_{\text{TDGP}}(a, b) = \exp\left[-\frac{1}{2}\left(\frac{a}{\ell(a)} - \frac{b}{\ell(b)}\right)^2\right].$$

And in terms of derivatives:

$$\frac{\mathrm{d}}{\mathrm{d}x} f_{\text{lmx}}(x) \sim \mathrm{N}\left(0, \frac{2 + \frac{\mathrm{d}}{\mathrm{dx}}\ell(x)^2}{2\ell(x)^2}\right), \tag{14}$$

$$\frac{\mathrm{d}}{\mathrm{d}x} f_{\text{TDGP}}(x) \sim \mathrm{N}\left(0, \frac{\left(\ell(x) - x\frac{\mathrm{d}}{\mathrm{dx}}\ell(x)\right)^2}{\ell(x)^4}\right). \tag{15}$$

Both kernels generalize Eq. (13) and, as expected, recover the stationary case when $\frac{\mathrm{d}}{\mathrm{d}x}\ell(x) = 0$.

Note that the domain of the lengthscale function $\ell(x)$ is always the domain of the function f, meaning that as we consider deeper models, the lengthscale function is always a function of the original domain. This is unlike the general compositional case, e.g. $f(g(h(x)))$, where the domain of each individual function is the image of the previous function. Moreover, the relationship with the lengthscale parameter and derivative of the output function remains clear.

B TDGP and CDGP are limits of a generalized DGP

Theorem 3.1. Any L-layer CDGP prior over a function $f(\boldsymbol{x}) = h^L(\boldsymbol{h}^{L-1}(\cdots \boldsymbol{h}^1(\boldsymbol{x})\cdots))$ is a special case of a TDGP prior with equal depth defined over the augmented input-space $\tilde{\boldsymbol{x}} = [\boldsymbol{x},\ 1]^\intercal$. Since linear deformations $\tilde{\boldsymbol{W}}\tilde{\boldsymbol{x}}$ in the augmented space correspond to affine transformations $\boldsymbol{W}\boldsymbol{x} + \boldsymbol{d}$ in the original space, the special case of the CDGP model corresponds to a TDGP where the prior variance of the $\boldsymbol{W}^\ell$ approaches zero.

Proof. First, we append a bias to the input data: $\tilde{\boldsymbol{x}} = [\boldsymbol{x} \quad 1]^\intercal \in \mathbb{R}^{D+1}$. Therefore, the hidden-layer matrices need to be expanded so that $\tilde{\boldsymbol{W}}(\tilde{\boldsymbol{h}}) \in \mathbb{R}^{(Q+1)\times(D+1)}$. Then, choose the following form for

$\tilde{\boldsymbol{W}}$:

$$\tilde{\boldsymbol{W}}(\tilde{\boldsymbol{h}}) = \begin{bmatrix} \boldsymbol{W}(\boldsymbol{h}) & \boldsymbol{d}(\boldsymbol{h}) \\ \boldsymbol{0}_{1\times Q} & 1 \end{bmatrix}, \tag{16}$$

where,

$$\boldsymbol{W}(\cdot) \in \mathbb{R}^{Q\times D} \sim \prod_{q=1}^{Q}\prod_{d=1}^{D} \mathcal{GP}\big(w_{qd}(\cdot) \mid 0(\cdot), k_{w_{qd}}\big), \tag{17}$$

$$\boldsymbol{d}(\boldsymbol{x}) \in \mathbb{R}^{Q} \sim \prod_{q=1}^{Q} \mathcal{GP}\big(d_q(\cdot) \mid \mu_{d_q}, k_{d_q}\big). \tag{18}$$

Note that $\tilde{\boldsymbol{W}}(\cdot)$ still follows the TDGP prior because all of its entries are either GP distributed or limits of GP priors, like in the case of the lower row where the Dirac delta distributions can be obtained by taking the limit of the kernel variance parameter σ^2 to zero.

Then, the ℓ-th latent space of this model is:

$$\begin{aligned}
\tilde{\boldsymbol{h}}^{\ell} &= \tilde{\boldsymbol{W}}(\tilde{\boldsymbol{h}}^{\ell-1})\,\tilde{\boldsymbol{x}} && (19)\\
&= \begin{bmatrix} \boldsymbol{W}(\boldsymbol{h}^{\ell-1}) & \boldsymbol{d}(\boldsymbol{h}^{\ell-1}) \\ \boldsymbol{0}_{1\times Q} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ 1 \end{bmatrix} && (20)\\
&= \begin{bmatrix} \boldsymbol{W}(\boldsymbol{h}^{\ell-1})\,\boldsymbol{x} + \boldsymbol{d}(\boldsymbol{h}^{\ell-1}) \cdot 1 \\ 0 \cdot \boldsymbol{x} + 1 \end{bmatrix} && (21)\\
&= \begin{bmatrix} \boldsymbol{W}(\boldsymbol{h}^{\ell-1})\boldsymbol{x} + \boldsymbol{d}(\boldsymbol{h}^{\ell-1}) & 1 \end{bmatrix}^{\mathsf{T}} && (22)\\
&= \begin{bmatrix} \boldsymbol{h}^{\ell} & 1 \end{bmatrix}^{\mathsf{T}}. && (23)
\end{aligned}$$

By ignoring the bias dimension, we get a latent space $\boldsymbol{h}$ that includes a multiplicative component $\boldsymbol{W}(\cdot)$ and an additive component $\boldsymbol{d}(\cdot)$. If the prior variance of $\boldsymbol{W}(\cdot)$ goes to zero, which is controlled by the kernel variance hyperparameter, then $\boldsymbol{W}(\cdot) \to \boldsymbol{0}$, resulting in only the additive component remaining:

$$\boldsymbol{h}^{\ell} = \boldsymbol{d}(\boldsymbol{h}^{\ell-1}), \tag{24}$$

which recovers the traditional compositional deep GP. ☐

Note that if $\mu_{d_q}(\cdot)$ is also a zero function, then, if the prior variance of $\boldsymbol{d}(\cdot)$ tends to zero, then $\boldsymbol{d}(\cdot) \to \boldsymbol{0}$, meaning that $\boldsymbol{h} = \boldsymbol{W}(\boldsymbol{h}^{\ell-1})\boldsymbol{x}$, which recovers the original TDGP model.

C Variational inference

First, the prior L-layer TDGP model as defined in Section 3 is

$$p(f^L(\cdot) \mid \boldsymbol{h}^{L-1}(\cdot)) = \mathcal{GP}\big(f \mid 0, \pi_k(\|\boldsymbol{h}^{L-1}(\boldsymbol{a}) - \boldsymbol{h}^{L-1}(\boldsymbol{b})\|)\big), \tag{25}$$

where

$$\boldsymbol{h}^{\ell}(\boldsymbol{x}) = \boldsymbol{W}^{\ell}(\boldsymbol{h}^{\ell-1}(\boldsymbol{x}))\boldsymbol{x}, \qquad \boldsymbol{h}^{0}(\boldsymbol{x}) = \boldsymbol{x}, \tag{26}$$

$$p(\boldsymbol{W}^{\ell}(\cdot) \mid \boldsymbol{h}^{\ell-1}(\cdot)) = \prod_{d=1}^{D}\prod_{q=1}^{Q_\ell} \mathcal{GP}\Big(w^{\ell}_{qd} \mid \mu^{\ell}_{w_{qd}}, \pi^{\ell}_{k_{w_{qd}}}(\|\boldsymbol{h}^{\ell-1}(\boldsymbol{a}) - \boldsymbol{h}^{\ell-1}(\boldsymbol{b})\|)\Big). \tag{27}$$

We now introduce inducing points for each GP layer in this process. The last-layer process has inducing points $\boldsymbol{u}^L$ and each hidden layer has inducing points $\boldsymbol{V}^{\ell}$ defined as follows:

$$\boldsymbol{u}^L = f^L(\boldsymbol{Z}^L), \qquad \boldsymbol{V}^{\ell} = \boldsymbol{W}^{\ell}(\boldsymbol{Z}^{\ell}), \tag{28}$$

where we also introduce L sets of pseudo-inputs $\{\boldsymbol{Z}^{\ell} \in \mathbb{R}^{m_\ell \times Q_\ell} \mid \ell \in [1, L]\}$. Finally, we define the variational distribution as follows:

$$q\big(\boldsymbol{f}^L, \boldsymbol{u}, \{\boldsymbol{W}^{\ell}, \boldsymbol{V}^{\ell}\}\big) = p(f(\cdot) \mid \boldsymbol{u}^L)q(\boldsymbol{u}^L) \prod_{\ell=1}^{L-1}\prod_{d=1}^{D}\prod_{q=1}^{Q_\ell} p(\boldsymbol{w}^{\ell}_{qd}(\cdot) \mid \boldsymbol{v}^{\ell}_{qd})q(\boldsymbol{v}^{\ell}_{qd}), \tag{29}$$

where

$$q(\boldsymbol{u}^L) = \mathrm{N}\big(\boldsymbol{u}^L \mid \check{\boldsymbol{\mu}}_u^L, \check{\boldsymbol{\Sigma}}_u^L\big), \qquad q(\boldsymbol{v}_{qd}^\ell) = \mathrm{N}\Big(\boldsymbol{v}_{qd}^\ell \mid \check{\boldsymbol{\mu}}_{v_{qd}}^\ell, \check{\boldsymbol{\Sigma}}_{v_{qd}}^\ell\Big). \tag{30}$$

The main simplification of this ELBO is to make each layer conditionally independent of each other when conditioned on the set of inducing variables.

C.1 Simplification for efficiency

In order to simplify this model, first, we will make each row of $\boldsymbol{W}^\ell(\cdot)$ share the same kernel and kernel hyperparameters, this means that the variational posterior covariance of $\boldsymbol{V}^\ell$ are also shared, i.e., $\check{\boldsymbol{\Sigma}}_{v_{qd}}^\ell = \check{\boldsymbol{\Sigma}}_{v_{qd'}}^\ell$ for every d and d', so we will represent this as $\check{\boldsymbol{\Sigma}}_{v_q}^\ell$. Secondly, to compute some expectations in closed form, following Titsias and Lázaro-Gredilla [6], we will consider all kernels to be squared exponential kernels. So that $\pi_k(r) = \sigma_f^2 \exp\big[-0.5r^2\big]$ and $\pi_{k_{w_{qd}}}^\ell(r) = \sigma_q^2 \exp\big[-0.5r^2\big]$.

And then, the prior model and variational distributions become:

$$\begin{aligned}
p(f^L(\cdot) \mid \boldsymbol{h}^{L-1}(\cdot)) &= \mathcal{GP}\big(f \mid 0, \sigma_f^2 \exp\big[-0.5\|\boldsymbol{h}^{L-1}(\boldsymbol{a}) - \boldsymbol{h}^{L-1}(\boldsymbol{b})\|^2\big]\big),\\
p(w_{qd}^\ell(\cdot) \mid \boldsymbol{h}^{\ell-1}(\cdot)) &= \mathcal{GP}\Big(w_{qd}^\ell \mid \mu_{w_{qd}}^\ell, \sigma_{w_q}^2 \exp\big[-0.5\|\boldsymbol{h}^{\ell-1}(\boldsymbol{a}) - \boldsymbol{h}^{\ell-1}(\boldsymbol{b})\|^2\big]\Big),\\
q(\boldsymbol{u}^L) &= \mathrm{N}\big(\boldsymbol{u}^L \mid \check{\boldsymbol{\mu}}_u^L, \check{\boldsymbol{\Sigma}}_u^L\big),\\
q(\boldsymbol{v}_{qd}^\ell) &= \mathrm{N}\Big(\boldsymbol{v}_{qd}^\ell \mid \check{\boldsymbol{\mu}}_{v_{qd}}^\ell, \check{\boldsymbol{\Sigma}}_{v_q}^\ell\Big).
\end{aligned}$$

C.2 Two-layer model

Finally, we will work on the model with a single hidden layer $\boldsymbol{W}(\cdot)$ and one output layer $f(\cdot)$. Again, the prior model is simplified to:

$$p(f(\cdot) \mid \boldsymbol{h}(\cdot)) = \mathcal{GP}\big(f \mid 0, \sigma_f^2 \exp\big[-0.5\|\boldsymbol{h}(\boldsymbol{a}) - \boldsymbol{h}(\boldsymbol{b})\|^2\big]\big), \tag{31}$$

where

$$p(\boldsymbol{W}(\cdot) \mid \boldsymbol{h}(\cdot)) = \prod_{d=1}^{D}\prod_{q=1}^{Q} \mathcal{GP}\left(w_{qd} \mid \mu_{w_{qd}}^\ell, \sigma_{w_q}^2 \exp\left[-0.5\left\|\frac{\boldsymbol{a}}{\boldsymbol{l}} - \frac{\boldsymbol{b}}{\boldsymbol{l}}\right\|^2\right]\right), \tag{32}$$

$$\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{W}(\boldsymbol{x})\,\boldsymbol{x}. \tag{33}$$

Accordingly, the variational distribution becomes:

$$q(\boldsymbol{u}) = \mathrm{N}\big(\boldsymbol{u} \mid \check{\boldsymbol{\mu}}_u, \check{\boldsymbol{\Sigma}}_u\big), \qquad q(\boldsymbol{v}_{qd}) = \mathrm{N}\big(\boldsymbol{v}_{qd} \mid \check{\boldsymbol{\mu}}_{v_{qd}}, \check{\boldsymbol{\Sigma}}_{v_q}\big). \tag{34}$$

C.3 Evidence lower bound (ELBO)

As in Titsias and Lázaro-Gredilla [6] and Titsias [4], we will consider the marginals of $f(\cdot)$ and $\boldsymbol{W}(\cdot)$ evaluated at the training data $(\boldsymbol{X}, \boldsymbol{y})$, $\boldsymbol{f} \in \mathbb{R}^n = \boldsymbol{f}(\boldsymbol{X})$ and $\boldsymbol{W} \in \mathbb{R}^{n\times Q\times D} = \boldsymbol{W}(\boldsymbol{X})$. Following the definition of the ELBO, we have the following lower bound on the evidence:

$$p(\boldsymbol{y}) = \langle p(\boldsymbol{y} \mid \boldsymbol{f})\rangle_{p(\boldsymbol{f},\boldsymbol{W})} \tag{35}$$

$$\geq \boxed{\langle \log p(\boldsymbol{y} \mid \boldsymbol{f})\rangle_{q(\boldsymbol{f},\boldsymbol{u},\boldsymbol{W},\boldsymbol{v})} - \mathrm{KL}(q(\boldsymbol{u}) \,\|\, p(\boldsymbol{u}))} - \sum_{d=1}^{D}\sum_{q=1}^{Q} \mathrm{KL}(q(\boldsymbol{v}_{dq}) \,\|\, p(\boldsymbol{v}_{dq})). \tag{36}$$

Given our simplifying assumptions from before and choice of variational distribution, the terms inside the blue box have the same form as the ELBO of Titsias and Lázaro-Gredilla [6], therefore the value of the blue box with optimal $q(\boldsymbol{u})$ is:

$$\begin{aligned}
\blacksquare = &-\frac{1}{2\sigma^2}\big(\boldsymbol{y}^\intercal\boldsymbol{y} + \psi_0 - \mathrm{Tr}\left[\boldsymbol{K}_u^{-1}\boldsymbol{\Psi}_2\right]\big) - \frac{n}{2}\ln\big(2\pi\sigma^2\big)\\
&+ \frac{1}{2\sigma^2}\boldsymbol{y}^\intercal\boldsymbol{\Psi}_1\boldsymbol{\Sigma}^{-1}\boldsymbol{\Psi}_1^\intercal\boldsymbol{y} + \frac{m_u}{2}\ln\big(\sigma^2\big) - \frac{1}{2}\ln|\boldsymbol{\Sigma}| + \frac{1}{2}\ln|\boldsymbol{K}_u|,
\end{aligned} \tag{37}$$

where the Ψ-statistics [5, 6] are defined as:

$$\psi_0 = \langle \operatorname{Tr} \boldsymbol{K}_f \rangle_{q(\boldsymbol{W},\boldsymbol{V})} = n\sigma_f^2, \qquad \boldsymbol{\Psi}_1 = \langle \boldsymbol{K}_{fu} \rangle_{q(\boldsymbol{W},\boldsymbol{V})}, \qquad \boldsymbol{\Psi}_2 = \left\langle \boldsymbol{K}_{fu}^{\intercal} \boldsymbol{K}_{fu} \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})}, \tag{38}$$

and the optimal $q(\boldsymbol{u})$ is:

$$q(\boldsymbol{u}) = \mathrm{N}\Big(\boldsymbol{u} \mid \boldsymbol{K}_u \left[\sigma^2 \boldsymbol{K}_u + \boldsymbol{\Psi}_2\right]^{-1} \boldsymbol{\Psi}_1^{\intercal} \boldsymbol{y},\ \sigma^2 \boldsymbol{K}_u \big(\sigma^2 \boldsymbol{K}_u + \boldsymbol{\Psi}_2\big)^{-1} \boldsymbol{K}_u\Big) \tag{39}$$

C.4 Computing the Ψ-statistics

The trick for computing the Ψ statistics is to show that each entry of the matrices only depends on a specific $\boldsymbol{W}(\boldsymbol{x}_i)$ and can be expressed as a product in the rows q, therefore allowing us to marginalize $q(\boldsymbol{W}, \boldsymbol{V})$ to $\prod_{d=1}^{D} q(\boldsymbol{w}_{iqd}, \boldsymbol{v}_{iqd})$. So, starting with $\boldsymbol{\Psi}_1$:

$$[\boldsymbol{\Psi}_1]_{ij} = \Big\langle [\boldsymbol{K}_{fu}]_{ij} \Big\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \tag{40}$$

$$= \left\langle \sigma_f \exp\left[-\frac{1}{2}\Big(\boldsymbol{W}_i \boldsymbol{x}_i - \boldsymbol{z}_j^{(1)}\Big)\Big(\boldsymbol{W}_i \boldsymbol{x}_i - \boldsymbol{z}_j^{(1)}\Big)^{\intercal}\right]\right\rangle_{q(\boldsymbol{W})}, \tag{41}$$

so we can marginalize $q(\boldsymbol{W})$,

$$= \left\langle \sigma_f \exp\left[-\frac{1}{2}\Big(\boldsymbol{W}_i \boldsymbol{x}_i - \boldsymbol{z}_j^{(1)}\Big)\Big(\boldsymbol{W}_i \boldsymbol{x}_i - \boldsymbol{z}_j^{(1)}\Big)^{\intercal}\right]\right\rangle_{q(\boldsymbol{W}_i)} \tag{42}$$

$$= \left\langle \sigma_f \exp\left[-\frac{1}{2}\sum_{q=1}^{Q}\Big(\boldsymbol{w}_{iq}^{\intercal} \boldsymbol{x}_i - z_{jq}^{(1)}\Big)^2\right]\right\rangle_{q(\boldsymbol{W}_i)} \tag{43}$$

$$= \sigma_f \prod_{q=1}^{Q} \left\langle \exp\left[-\frac{1}{2}\Big(\boldsymbol{w}_{iq}^{\intercal} \boldsymbol{x}_i - z_{jq}^{(1)}\Big)^2\right]\right\rangle_{q(\boldsymbol{w}_{iq})}. \tag{44}$$

This is the same situation as in Appendix B.1 of [6], so:

$$[\boldsymbol{\Psi}_1]_{ij} = \sigma_f \prod_{q=1}^{Q} \Big(\boldsymbol{x}_i^{\intercal} \tilde{\boldsymbol{\Sigma}}_q \boldsymbol{x}_i + 1\Big)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}\frac{\Big(\tilde{\boldsymbol{\mu}}_{iq}^{\intercal} \boldsymbol{x}_i - z_{jq}^{(1)}\Big)^2}{\Big(\boldsymbol{x}_i^{\intercal} \tilde{\boldsymbol{\Sigma}}_{iq} \boldsymbol{x}_i + 1\Big)}\right], \tag{45}$$

where $\tilde{\boldsymbol{\mu}}_{iq}$ and $\tilde{\boldsymbol{\Sigma}}_{iq}$ are the mean and covariance of $\prod_{d=1}^{D} q(\boldsymbol{w}_{iqd})$. Now, for $\boldsymbol{\Psi}_2$:

$$[\boldsymbol{\Psi}_2]_{jk} = \left\langle \Big[\boldsymbol{K}_{fu}^{\intercal} \boldsymbol{K}_{fu}\Big]_{jk} \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \tag{46}$$

$$= \left\langle \sum_{i=1}^{n} [\boldsymbol{K}_{fu}]_{ij} [\boldsymbol{K}_{fu}]_{ik} \right\rangle_{q(\boldsymbol{W},\boldsymbol{V})} \tag{47}$$

$$= \sum_{i=1}^{n} \Big\langle [\boldsymbol{K}_{fu}]_{ij} [\boldsymbol{K}_{fu}]_{ik} \Big\rangle_{q(\boldsymbol{W}_i)}. \tag{48}$$

Again, following [6]:

$$\begin{aligned} [\boldsymbol{\Psi}_2]_{jk} = \ & \sigma_f^2 \exp\left[-\frac{1}{4}\sum_{q=1}^{Q}\Big(z_{jq}^{(1)} - z_{kq}^{(1)}\Big)^2\right] \\ & \times \sum_{i=1}^{n} \prod_{q=1}^{Q} \Big(2\boldsymbol{x}_i^{\intercal} \tilde{\boldsymbol{\Sigma}}_q \boldsymbol{x}_i + 1\Big)^{-\frac{1}{2}} \exp\left[-\frac{\big(\tilde{\boldsymbol{\mu}}_{iq}^{\intercal} \boldsymbol{x}_i - \tilde{z}_q\big)^2}{2\boldsymbol{x}_i^{\intercal} \tilde{\boldsymbol{\Sigma}}_{iq} \boldsymbol{x}_i + 1}\right], \end{aligned} \tag{49}$$

where $\tilde{z}_q = \frac{z_{jq}^{(1)} + z^{(1)}{}_{kq}}{2}$.

D Details on experiments

Setting. For each experiment, we performed cross-validation (five folds for bathymetry and 10 folds for UCI). We compare the generalization error using MRAE and compare the uncertainty quantification using NLPD. We used the Adam optimizer, and the following schedule (we varied Adam step size and likelihood variance):

- For 500 epochs: step size 0.1, and likelihood variance fixed to 0.01,
- For 1500 epochs: step size 0.01, and likelihood variance fixed to 0.01,
- For 5000 epochs: step size 0.001, and likelihood variance is trainable.

Architecture. The architectural details of each model are:

SGPR 50 inducing points for the output process and an ARD-squared exponential kernel. Inference is done by using the optimal $q(\boldsymbol{u})$ as described by Titsias [4].

DKL 50 inducing points for the output process and an ARD-squared exponential kernel. For the deep kernel, we use an MLP with architecture $[D, 500, 50, D]$, where D is the dimension of the inputs and a final BatchNorm layer. All hidden-layer activations are ReLU. Inference is done by using the optimal $q(\boldsymbol{u})$ as described by Titsias [4].

CDGP 50 inducing points for the output process and 25 for the latent space process. All layers use an ARD-squared exponential kernel. The dimension of the hidden layer is set to D. We use doubly stochastic inference [3] with whitened variables, i.e. we reparametrize $\boldsymbol{u}$ as $\boldsymbol{K}_u^{-\frac{1}{2}}\boldsymbol{u}$.

DNSGP 50 inducing points for the output process and 25 for the lengthscale matrix space process. All layers use an ARD-squared exponential kernel and the lengthscale matrix process is set to be diagonal with warping function $\exp(\boldsymbol{h} + s)$, where s is a learnable scalar. We use doubly stochastic inference [3] with whitened variables, i.e. we reparametrize $\boldsymbol{u}$ as $\boldsymbol{K}_u^{-\frac{1}{2}}\boldsymbol{u}$.

TDGP 50 inducing points for the output process and 25 for the inverse lengthscale matrix space process. The size of the inverse lengthscale matrix $\boldsymbol{W}$ is set to $Q \times D$, where $Q = D$ and each row q of shares the same kernel. Our variational posterior distribution in $q(\boldsymbol{V})$ is set to mean-field where $q(\boldsymbol{V}) = \prod_{i=1}^{n} \prod_{q=1}^{Q} \prod_{d=1}^{D} q(v_{iqd})$.

D.1 Synthetic experiment

Data. We generated a synthetic dataset by definition a composite function $f = g \circ h$, with $g : \mathbb{R} \to \mathbb{R}$ and $h : \mathbb{R}^2 \to \mathbb{R}$ are non-linear functions. In this context, h acts as a “funnel” inducing a 1D manifold. These functions are defined as:

$$h(\boldsymbol{x}) = 2x_0 \sin(x_0 \pi) + 2\cos(x_0 \pi) \tag{50}$$

$$g(z) = \frac{\sin(z)}{z} - z^2. \tag{51}$$

Then, we uniformly sample $\boldsymbol{x}$ in the interval $[-1, 1] \times [-1, 1]$ and split 50/50 for train and validation.

D.2 Bathymetry case study

Data. The selected subset of GEBCO data covers the Andes mountain range, ocean, and land as an example of a non-stationary task (longitude in the range from -80 to -60, and latitude in the range from -20 to -10. We randomly subsampled 1000 data points.

Computational resources. Running all the models took 2.5 hours using an NVIDIA A100 GPU. More fine-grained time measurements are presented in Table 3.

Table 3: Training and evaluation time (seconds) in GEBCO dataset (avg±std). Lower is better.

	Train	Evaluation
SGP	23.4 ± 1.5	0.03 ± 0.01
DKL	384.6 ± 8.5	0.76 ± 1.28
CDGP	570.8 ± 0.6	0.14 ± 0.17
DNSGP	596.4 ± 29.8	0.06 ± 0.01
TDGP	126.7 ± 0.8	0.12

Table 4: Training time (seconds) for the benchmark datasets (avg±std). Lower is better.

	housing	concrete	energy	wine_red
SGP	41.4 ± 0.2	41.0 ± 0.7	41.5 ± 1.2	42.6 ± 2.0
DKL	379.1 ± 3.7	379.0 ± 3.5	376.7 ± 2.3	386.1 ± 13.4
CDGP	601.5 ± 9.4	592.1 ± 4.1	592.9 ± 3.5	615.4 ± 23.5
DNSGP	668.0 ± 75.9	621.8 ± 2.2	620.8 ± 1.9	650.8 ± 24.0
TDGP	572.4 ± 12.4	563.7 ± 1.8	484.3 ± 1.4	928.6 ± 42.1

D.3 Benchmark datasets

Data. We used a subset of datasets from the UCI repository: housing[3], concrete[4], wine-red[5], and energy datasets.[6]

Housing dataset has 506 samples with 13 features; concrete dataset has 1030 samples with 8 features; wine-red has 1599 samples with 11 features; energy dataset has 768 samples with 8 features.

Computational resources. Training all models in all datasets for ten folds took 25.55 hours using an NVIDIA TITAN RTX GPU. Per dataset time measurements are presented in Table 4

E Computational and test performance as a function of width

In the experiments of Section 4, the width of the hidden layer Q for TDGP was always set to match the dimension of the input D. As seen in Fig. 10, after optimization of the hyperparameters, the effective width of the layer for all datasets was always much smaller than D. Therefore, it is reasonable to expect that a wider model wouldn't increase the model's performance.

Nevertheless, we conduct an additional experiment to explore the performance penalty of increasing Q up to D in terms of computational resources and test accuracy. In the chosen `housing` dataset, Fig. 10 shows that $Q \approx 2 < D$ is the effective width of an optimized network. Therefore, we re-run this experiment with values of Q ranging from 1 to D as shown in Fig. 11.

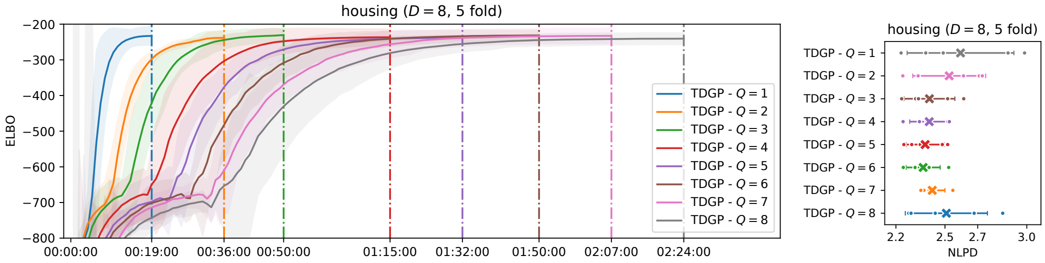

Figure 11: Training curves (left) and test metrics (right) for the housing dataset with a variable width Q

---

[3] `https://archive.ics.uci.edu/ml/machine-learning-databases/housing/`

[4] `https://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength`

[5] `https://archive.ics.uci.edu/ml/datasets/Wine+Quality`, red variant

[6] `https://archive.ics.uci.edu/ml/datasets/energy+efficiency`

As discussed, we observe a linear increase in training time as the model's width increases. In terms of predictive performance, the best widths are 3 to 6; in theory, we wouldn't expect a performance drop above a certain minimum width, as the effective width is a trained variable, however, as stated in our limitation, we expected the increased number of variables to optimize to add more complexity to the optimization landscape and, therefore, increase the difficulty in finding the best set of hyperparameters.

F Expressivity of the prior with increasing depth

The TDGP model as defined in Section 3 places a zero-mean prior on all the layers. This is in contrast with the standard CDGP model, which as shown in Duvenaud et al. [1], suffers prior collapse under this assumption. Figure 2 shows this effect by plotting different samples from CDGP and TDGP priors with zero mean as the model depth increases. Nonetheless, as shown in Salimbeni and Deisenroth [2], another way to visualize this pathology is to plot samples of the covariance matrix as the number of layers increase.

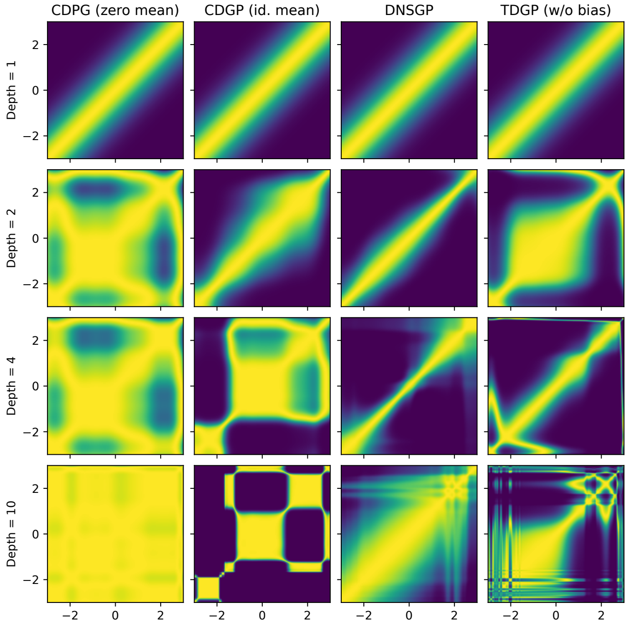

Figure 12: Samples from the prior covariance matrix for different layers and models

As shown in Fig. 12, as the model depth increses, the covariance matrix of CDGP with zero mean eventually saturates, i.e. all points are high correlated, which leads to the flat priors shown in Fig. 2. We can also see that, as discussed in Salimbeni and Deisenroth [3], changing the zero-mean prior to one with an linear mean function fixes this pathology, as well as using a zero-mean DNSGP or a zero-mean TDGP model. This is further evidence that our model

G Societal and broader impact

Gaussian processes are popular methods for spatiotemporal modeling in, e.g., climatology, geoscience, public health, and ecology. This work proposes TDGP, a novel formulation for deep GPs that preserves the performance of compositional deep GPs while significantly improving their interpretability. We believe the inherent interpretability of TDGP priors will make it easier for applied researchers to encode their subjective knowledge, consequently improving the data efficiency of their models and reducing predictive uncertainties. Additionally, we do not foresee any negative societal impact stemming directly from this work.

References

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

[2] Hugh Salimbeni and Marc Peter Deisenroth. "Deeply non-stationary Gaussian processes". In: *2nd Workshop on Bayesian Deep Learning (NeurIPS)*. 2017.

[3] Hugh Salimbeni and Marc Peter Deisenroth. "Doubly Stochastic Variational Inference for Deep Gaussian Processes". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017.

[4] Michalis K. Titsias. "Variational Learning of Inducing Variables in Sparse Gaussian Processes". In: *Artificial Intelligence and Statistics (AISTATS)*. 2009.

[5] Michalis K. Titsias and Neil D. Lawrence. "Bayesian Gaussian Process Latent Variable Model". In: *Artificial Intelligence and Statistics (AISTATS)*. 2010.

[6] Michalis K. Titsias and Miguel Lázaro-Gredilla. "Variational Inference for Mahalanobis Distance Metrics in Gaussian Process Regression". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2013.