Bayesian Convolutional Deep Sets with Task-Dependent Stationary Prior

Yohan Jung **Jinkyoo Park**
Korea Advanced Institute of Science and Technology

Abstract

Convolutional deep sets is a neural network architecture that can model stationary stochastic processes. This architecture uses the kernel smoother and deep convolutional neural network to construct translation equivariant functional representations. However, the non-parametric nature of the kernel smoother can produce ambiguous representations when the number of data points is not given sufficiently. To address this issue, we introduce Bayesian convolutional deep sets, which constructs random translation equivariant functional representations with a stationary prior. Furthermore, we present how to impose the task-dependent prior for each dataset because a wrongly imposed prior can result in an even worse representation than that of the kernel smoother. Empirically, we demonstrate that the proposed architecture alleviates the targeted issue in various experiments with time-series and image datasets.

1 Introduction

Neural process (NP) and Conditional neural process (CNP) [Garnelo et al., 2018a,b] are pioneering deep learning frameworks that can model stochastic processes, i.e., the functions over a distribution. That is, for any finite pairs of the input and output, referred to as context sets, these models output the predictive distribution on targeted inputs (target sets) by extracting the feature from context sets. Specifically, these models employ Deep sets [Zaheer et al., 2017], a specific structure of DNN, to reflect the exchangeability of the stochastic process into a predictive distribution of the NP. Many variants of NP [Singh et al., 2019, Yoon et al., 2020, Louizos et al., 2019, Gordon et al., 2019, Foong et al., 2020, Bruinsma et al., 2021, Kawano et al., 2020, Holderrieth et al., 2021] have been proposed to model stochastic processes elaborately.

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206. Copyright 2023 by the author(s).

Convolutional conditional neural process (ConvCNP) [Gordon et al., 2019] is a NP model that can model a stationary process in which statistical characteristics of the finite random variables, such as the mean and covariance, remain unchanged even when the time indexes of those random variables are shifted. ConvCNP employs a specific structure of DNN, called Convolutional Deep sets (ConvDeepsets), to construct the translation equivariant functional representation. This allows ConvCNP to incorporate the inductive bias of stationarity, which is a desirable property for modeling stationary processes.

ConvDeepsets employs the RBF kernel smoother [Nadaraya, 1964] to build a discretized functional representation of the context set. This discretized representation is then mapped to an abstract representation via a Convolutional neural network (CNN), which is used to make the predictive distribution on the target set. This abstract representation can be used to model stationary processes because the kernel smoother and convolution operation produce a consistent representation regardless of the translated inputs of the context sets. However, when the number of context data points is insufficient, the representation by the kernel smoother could be ambiguous because the kernel, a non-parametric model, has the expressive power depending on the amount of given context data points. This potentially degrades the CNN representation and the modeling performance of ConvCNP, which is analogous to the task ambiguity issue [Finn et al., 2018] in model-agnostic meta-learning (MAML) [Finn et al., 2017].

To mitigate the issue of task ambiguity, one intuitive approach is to incorporate the inductive bias on the representation for the kernel smoother. In fact, the Bayesian approach, which imposes prior distribution on the model parameters, has been effective in addressing task ambiguity in MAML [Finn et al., 2018]. However, using the prior distribution also raises the question of which prior distribution should be used because an incorrect choice for the prior distribution could degrade the ConvCNP representation and performance.

In this work, we propose Bayesian convolutional Deep sets that uses a task-dependent stationary prior to construct random functional representations. To this end, we first consider a set of stationary kernels, each characterized by its own distinct spectral density. Then, we construct a task-dependent prior using an amortized latent categorical variable modeled by a translate-invariant neural network. Thus, the latent variable assigns an appropriate kernel out of the candidate set depending on the task. Next, we generate Gaussian process (GP) posterior sample functions using the chosen kernel and forward those sample functions via CNN. We refer to this as the representation of Bayesian ConvDeepsets. Importantly, we prove that the representation of Bayesian ConvDeepsets still satisfies the translation equivalence necessary for modeling the stationary process.

For training, we use variational inference based on a meta-learning framework. Additionally, we consider an additional regularizer that allows the translate-invariant neural network to select the stationary prior appropriately depending on the task. Empirically, we demonstrate that the proposed method relaxes the task ambiguity issue in both time series and image datasets by utilizing a task-dependent prior. Our contributions can be summarised as follows:

- We propose Bayesian ConvDeepsets using a task-dependent stationary prior and ensure that it still satisfies translation equivariance.
- We demonstrate that Bayesian ConvDeepsets improves the performance of ConvCNP on various tasks of stationary process modeling, especially when the number of context data points is insufficiently given.

2 Preliminaries

Neural Process. NP uses Deepsets, a specific architecture of DNN [Zaheer et al., 2017], to reflect the exchangeability of the stochastic process into the predictive distribution of the NP and employs the meta-learning for training.

Let $D^c=\{(x^c_n, y^c_n)\}_{n=1}^{N^c}$ be a context set consisting of the context inputs $X^c=\{x^c_n\}_{n=1}^{N^c}$ and outputs $Y^c=\{y^c_n\}_{n=1}^{N^c}$. Similarly, let $D^t=\{(x^t_n, y^t_n)\}_{n=1}^{N^t}$ be a target set consisting of the target inputs $X^t=\{x^t_n\}_{n=1}^{N^t}$ and outputs $Y^t=\{y^t_n\}_{n=1}^{N^t}$. Let $\mathcal{T} = (D^c, D^t)$ be a task consisting of the context set and target set and $p(\mathcal{T})$ be a task distribution. Then, for any task $\mathcal{T} \sim p(\mathcal{T})$, NP employs a neural network g_{nn} to model the predictive distribution on target inputs X^t:

$$g_{\text{nn}} : \left(D^c, X^t\right) \longmapsto \left(\mu_{\text{nn}}(D^c, X^t),\ \sigma_{\text{nn}}(D^c, X^t)\right), \quad (1)$$

which maps the context set D^c and the target inputs X^t to the predictive mean $\mu_{\text{nn}}(D^c, X^t)$ and standard deviation $\sigma_{\text{nn}}(D^c, X^t)$. For training, NP is optimized with respect to model parameters θ_g with following objective:

$$\max_{\theta_g} \mathbb{E}_{D^c, D^t \sim p(\mathcal{T})} \left[\log p\left(Y^t | g_{\text{nn}}\left(X^t, D^c\right)\right)\right]. \quad (2)$$

Translation Equivariance. A stationary process is a stochastic process where the statistical characteristics of the finite random variable of the process remain unchanged even when there is a shift in time. To model a stationary process, functions should satisfy the special conditions referred to as Translation Equivariance (TE). Mathematically, TE can be defined as follows:

Definition 1 ([Gordon et al., 2019]). *Let $\mathcal{X} = R^d$ and $\mathcal{Y} \subset R^{d'}$ be space of the inputs and outputs, and let $\mathcal{D} = \cup_{m=1}^{\infty}(\mathcal{X} \times \mathcal{Y})^m$ be the joint space of the finite observations. Also, let $\mathcal{H}$ be a space of bounded continuous function on $\mathcal{X}$, and T and T^* be the mappings:*

$$\begin{aligned} T : \mathcal{X} \times \mathcal{D} \to \mathcal{D}, \qquad & T_\tau(D) = \{(x_n + \tau, y_n)\}_{n=1}^N \\ T^* : \mathcal{X} \times \mathcal{H} \to \mathcal{H}, \qquad & T^*_\tau(h(\cdot)) = h(\cdot - \tau) \end{aligned}$$

where $D = \{(x_n, y_n)\}_{n=1}^N \in \mathcal{D}$ denotes N pairs of the inputs and outputs, $\tau \in \mathcal{X}$ denotes translation variable for the inputs, and $h(\cdot) \in \mathcal{H}$ denote the bounded continuous function for any input $\cdot \in \mathcal{X}$. Then, a functional mapping Φ:$\mathcal{D} \to \mathcal{H}$, that maps finite data points $D \in \mathcal{D}$ to a function $h \in \mathcal{H}$, is translation equivariant if the following holds:

$$\Phi \circ (T_\tau(D)) = T^*_\tau \circ (\Phi(D)). \quad (3)$$

Roughly speaking, Definition 1 implies that the function satisfying the TE should produce a consistent functional representation up to the order of translation.

Convolutional Deep Sets. ConvDeepsets is a specific architecture of the neural network satisfying the TE in Eq. (3) and thus can be used to model stationary process. The following proposition introduces the ConvDeepsets structure.

Proposition 1 ([Gordon et al., 2019]). *The representation $\Phi(D)(\cdot)$ is translation equivariant if and only if $\Phi(D)(\cdot)$ is represented as:*

$$\begin{aligned} \underbrace{E(D)(\cdot)}_{\substack{\text{functional}\\ \text{representation}}} &= \Bigg[\underbrace{\sum_{n=1}^{N} k(\cdot - x_n)}_{\text{density}} , \underbrace{\sum_{n=1}^{N} \frac{y_n\, k(\cdot - x_n)}{\sum_{n=1}^{N} k(\cdot - x_n)}}_{\text{data representation}} \Bigg], \\ \underbrace{\Phi(D)(\cdot)}_{\substack{\text{ConvDeepsets}\\ \text{representation}}} &= \underbrace{\rho}_{\substack{\text{mapping via}\\ \text{CNN}}} \circ \underbrace{E(D)(\cdot)}_{\substack{\text{functional}\\ \text{representation}}} \end{aligned} \quad (4)$$

where $k(\cdot - x_n)$ denotes the stationary kernel centered at x_n, and $\rho(\cdot)$ is the continuous and translation equivariant mapping. Here, RBF kernel function is used for $k(\cdot)$, and $\rho(\cdot)$ is parameterized by Convolutional neural network (CNN).

Neural Process with Convolutional Deepsets. ConvCNP [Gordon et al., 2019] and ConvLNP [Foong et al., 2020] are well-known NP models for stationary process modeling by using ConvDeepsets as the main structure of the NP model.

To employ the functional representation of ConvDeepsets in practice, these models first consider M discretized inputs $\{t_m\}_{m=1}^M \subset [\min X, \max X]$ by spacing the range of inputs $X = X^c \cup X^t$ linearly. Then, these models construct the functional representation $\{\Phi(D^c)(t_m)\}_{m=1}^{M}$ on M discretized inputs $\{t_m\}_{m=1}^{M}$ as follows:

$$\Phi(D^c)(t_m) = (\rho \circ E(D^c))\,(t_m) \quad m = 1, .., M. \tag{5}$$

This discretized representation $\{\Phi(D^c)(t_m)\}_{m=1}^{M}$ is used to obtain the parameters of the predictive distribution $\mu_{\mathrm{nn}}(X^t)$ and $\sigma_{\mathrm{nn}}(X^t)$ as shown in Eq. (1). Specifically, the smoothed representation $\tilde{\Phi}(D^c)(x_n^t)$, represented as

$$\tilde{\Phi}(D^c)(x_n^t) = \sum_{m=1}^{M} \Phi(D^c)(t_m)\, k(x_n^t - t_m), \tag{6}$$

is used to model the predictive distribution on target inputs $x_n^t \in X^t$. For grid dataset, we can omit the discretization procedure and employ the CNN directly as described in [Gordon et al., 2019].

3 Methodology

In this section, we first interpret the representation of the ConvDeepsets and its motivation in Section 3.1. Then, we introduce the task-dependent stationary prior in Section 3.2, Bayesian ConvDeepsets in Section 3.3, and its application to stationary process modeling in Section 3.4. Fig. 2 outlines the prediction procedure via Bayesian ConvDeepsets that is described in Sections 3.2 to 3.4.

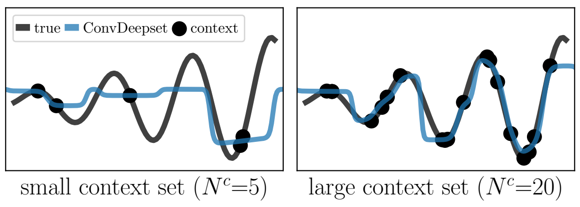
3.1 Interpretation of ConvDeepsets Representation

The data representation of $E(D^c)(\cdot)$ in Eq. (4) is constructed by the RBF kernel smoother known as the non-parametric model. The lengthscale of the RBF kernel controls the smoothness of the data representation; when the lengthscale is close to 0, the data representation is expressed as $\sum_{n=1}^{N^c} y_n^c\ \delta(\cdot - x_n^c)$ with a Dirac-delta function $\delta(\cdot)$. This implies that the expressive power of the kernel smoother is proportional to the number of context points N^c. If only a few context data points (small N^c) are available, the data representation may be ambiguous and thus negatively affect $\Phi(D^c)(\cdot)$ and the overall modeling performance of the NP models.

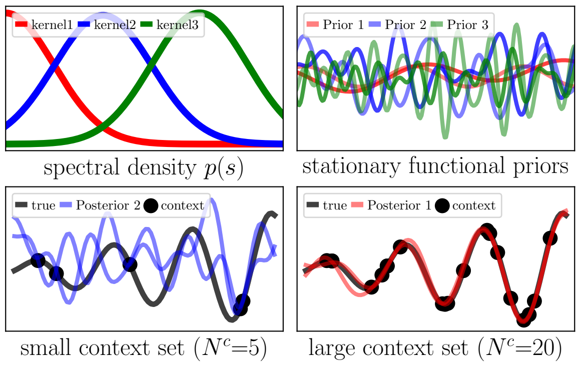
To address the issue, we note that the data representation of $E(D^c)(\cdot)$ in Eq. (4) can also be expressed as the rescaled predictive mean of GP posterior with the restricted covariance $\mathrm{Diag}(K(X^c, X^c))$, as follows:

$$\begin{aligned}&\sum_{n=1}^{N^c} \frac{y_n^c\ k(\cdot - x_n^c)}{\sum_{n=1}^{N^c} k(\cdot - x_n^c)} \\ &= \frac{1}{\sum_{n=1}^{N^c} k(\cdot - x_n^c)} \underbrace{K(\cdot, X^c)\mathrm{Diag}(K(X^c, X^c))^{-1} Y^c}_{\text{predictive mean of GP posterior}}\end{aligned} \tag{7}$$

where $[\mathrm{Diag}(K(X^c, X^c))]_{n,m} = 1_{n=m}$ and $K(\cdot, X^c) \in R^{1\times N^c}$ and the stacked context outputs $Y^c \in R^{N^c}$. Based on this observation, we hypothesis that (1) employing a suitable stationary GP prior depending on each task, and (2) using or adding GP posteriors sample functions to represent the data part in $E(D^c)(\cdot)$ could alleviate unclear ConvDeepsets representation when using only a small N^c context data points. Fig. 1 describes our concern for data representation and a intuitive approach to resolve this issue.



(a) Data representation of ConvDeepsets



(b) Data representation with stationary functional prior

Figure 1: Comparison of data representations over different N^c context points. Fig. 1a shows the data representation of $E(D^c)$ for ConvDeepsets; the black dots denotes the context set D^c, and black and blue line denote the true function and the data representation, respectively. Fig. 1b shows the stationary priors characterized by each spectral density $p_q(s)$ in Eq. (9), and the improved data representation due to the reasonably imposed prior, especially for the $N^c = 5$.

3.2 Amortized Task-Dependent Stationary Prior

To impose the appropriate prior depending on the task, we employ an amortized latent variable approach. To this end, we employ Bochner's theorem [Bochner, 1959] that explains how the stationary kernel can be constructed as the inverse Fourier transform of the spectral density. We first consider Q spectral densities $\{p_q(s)\}_{q=1}^{Q}$ where the q-th density $p_q(s)$ is modeled as Gaussian distribution with parameters $\{\mu_q, \sigma_q^2\}$:

$$p_q(s) = N\left(s; \mu_q, \mathrm{Diag}(\sigma_q^2)\right). \tag{8}$$

Then, we construct the q-th stationary kernel $k_q(\tau)$ by taking a inverse Fourier transform of $p_q(s)$ as follows:

$$\begin{aligned}k_q(\tau) &= \int e^{i2\pi s^T\tau} p_q(s)ds \\ &= \exp\left(-2\pi^2\tau^T\mathrm{Diag}(\sigma_q^2)\tau\right)\cos\left(2\pi\mu_q\tau\right).\end{aligned} \tag{9}$$

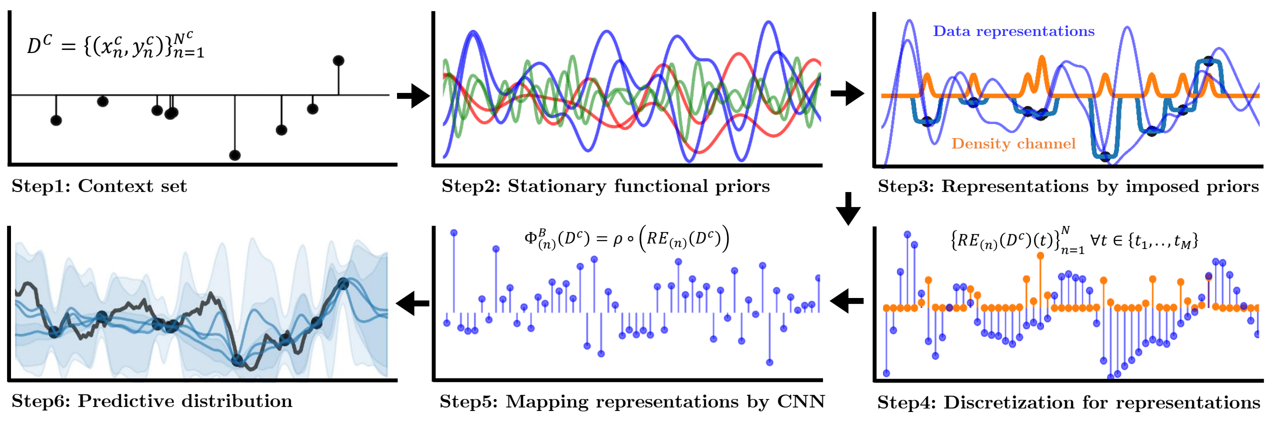

$D^C = \{(x_n^c, y_n^c)\}_{n=1}^{N^c}$
Step1: Context set
Step2: Stationary functional priors
Data representations
Density channel
Step3: Representations by imposed priors
$\{RE_{(n)}(D^c)(t)\}_{n=1}^N \ \forall t \in \{t_1, .., t_M\}$
Step4: Discretization for representations
$\Phi_{(n)}^B(D^c) = \rho \circ (RE_{(n)}(D^c))$
Step5: Mapping representations by CNN
Step6: Predictive distribution

Figure 2: Prediction procedure: **Step 1:** Given context set D^c. **Step 2:** Consider the random stationary functions $\{\phi_q\}_{q=1}^Q$ of Eq. (15). **Step 3:** Assign the task-dependent kernel by Eqs. (12) and (14) out of stationary priors in Step 2, and construct N random representations $\{RE_{(n)}(D^c)(\cdot)\}_{n=1}^N$ in Eq. (19). **Step 4:** Discretize the representation as explained in Eq. (5). **Step 5:** Map the discredited representation via CNN, i.e., $\Phi^{\mathcal{B}}_{(n)}(D^c)(\cdot)$. **Step 6:** Smooth the mapped representations by Eq. (6), and make prediction distribution by Eq. (22). For the Step 4 and 5, one random representation is depicted for clarity.

We order $\{k_q\}_{q=1}^Q$ by $\mu_1{=}0$ and $\mu_1 \leq .. \leq \mu_Q$ in element-wise sense. Then, we consider the categorical latent variable $Z_{\text{cat}} = (z_1, .., z_Q) \in \{0,1\}^Q$, with $\sum_{q=1}^Q z_q = 1$, of which distribution is defined as follows:

$$q(Z_{\text{cat}} \mid D^c) = \text{Cat}\left(Z_{\text{cat}}\ ;\ \text{p}_{\text{nn}}(D^c)\right), \qquad (10)$$

where $\text{p}_{\text{nn}} : D^c \longrightarrow \Delta^{Q-1}$, with Q-dimension simplex Δ^{Q-1}, denotes the translate-invariant NN satisfying

$$\text{p}_{\text{nn}}(T_\tau(D^c)) = \text{p}_{\text{nn}}(D^c). \qquad (11)$$

This is designed to make the derived functional representation satisfy the TE, which is stated in Proposition 2. Specific structure, used in this work, is described in Appendix A.2.1.

Using the kernels $\{k_q(\tau)\}_{q=1}^Q$ in Eq. (9) and the latent variable Z_{cat} in Eq. (10), we build the amortized latent stationary kernel depending on the context set D^c:

$$k(\tau) \mid Z_{\text{cat}}, D^c = z_1 k_1(\tau) \ + \ \cdots \ + \ z_Q k_Q(\tau). \quad (12)$$

Multi-channel Extension. The latent stationary kernel in Eq. (12) can be extended for multi-channel processes (K channels) by considering K-th channel categorical variables $\{Z^k_{\text{cat}}\}_{k=1}^K$ with $Z^k_{\text{cat}}|D^c \sim \text{Cat}(\text{p}^k_{\text{nn}}(D^c))$, where $\text{p}^k_{\text{nn}}(D^c)$ denotes an amortized parameter for k-th channel.

3.3 Bayesian Convolutional Deep sets.

With the latent stationary kernel in Eq. (12), we present the Bayesian ConvDeepsets using the task-dependent stationary prior. We employ the GP posterior path-wise sampling [Wilson et al., 2020] to build the random data representation efficiently.

Let $f(D^c)(\cdot)$ be the random data representation for the context set D^c. Then, $f(D^c)(\cdot)$ can be represented via terms of the prior function and the update function:

$$\underbrace{f(D^c)(\cdot)}_{\substack{\text{random data}\\ \text{representation}}} := \underbrace{\sum_{q=1}^{Q} \sqrt{z_q}\ \phi_q(\cdot)}_{\text{prior term}} + \underbrace{\sum_{n=1}^{N^c} v_n\ k(\cdot - x_n^c)}_{\text{update term}}. \quad (13)$$

For the prior function, we first construct the random samples $(z_1, .., z_Q)$ of the latent categorical variable of Eq. (10), i.e.,

$$(z_1, .., z_Q) \sim \text{Cat}\left(Z_{\text{cat}}\ ;\ \text{p}_{\text{nn}}(D^c)\right) \qquad (14)$$

by applying Gumbel-softmax trick [Jang et al., 2016] to the output of $\text{p}_{\text{nn}}(D^c)$. Then, based on random Fourier feature (RFF) [Rahimi and Recht, 2007, Jung et al., 2022], we construct the q-th random stationary function $\phi_q(\cdot) \sim GP(0, k_q)$ as follows:

$$\{w_{q,i}\}_{i=1}^l{\sim}N(0,I),\ \{s_{q,i}\}_{i=1}^l{\sim}p_q(s),\ \{b_{q,i}\}_{i=1}^l{\sim}U[0,2\pi],$$

$$\phi_q(\cdot) = \sqrt{\frac{2}{l}} \sum_{i=1}^{l} w_{q,i} \cos\left(2\pi\langle s_{q,i}, \cdot\rangle + b_{q,i}\right), \qquad (15)$$

where $w_{q,i}$ denotes the random weight samples, $s_{q,i}$ and $b_{q,i}$ denote the spectral points sampled from $p_q(s)$ and phase sampled from uniform distribution $U[0, 2\pi]$, respectively.

For the update function, we use the expected kernel, that is integrated over the latent categorical variable Z_{cat}:

$$k(\cdot - x_i^c) = \mathbb{E}_{q(Z_{\text{cat}}|D^c)}\left[k(\cdot - x_i^c) \mid Z_{\text{cat}}(D^c)\right]. \qquad (16)$$

We use smoothing weight $v_n \in R$ for n-th context input x_n^c:

$$v_n = \left[K(X^c, X^c){+}\sigma_\epsilon^2 I)^{-1} \left(Y^c - \Psi(X^c)\right)\right]_n, \qquad (17)$$

where $K(X^c, X^c) \in R^{N^c \times N^c}$ denotes Gram matrix of the expected kernel k in Eq. (16), and $Y^c \in R^{N^c}$ denotes the stacked vector of the context outputs $\{y_n^c\}_{n=1}^{N^c}$. $\Psi(X^c) \in R^{N^c}$ denotes the values of the random stationary function on X^c, which is computed by the prior term in Eq. (13).

Computational Efficiency. The random data representation in Eq. (13) can be constructed in a more computationally efficient way. This is because Eq. (13) requires Cholesky decomposition once for $\{v_n\}_{n=1}^{N^c}$ and thus takes computational cost $\mathcal{O}((N^c)^3 + M(Ql))$ with l random Fourier features and Q mixtures. On the other hand, the conventional GP sampling method requires $\mathcal{O}((N^c)^3 + M^3)$ computational cost due to two Cholesky decompositions to compute predictive parameters and sample the functions on grid. Hence, the proposed functional representation is more computationally efficient due to $Ql, N^c \ll M$ in the general setting. The details and evidence supporting this claim are further elaborated in Appendix A.3.2 and demonstrated in Appendix B.4.

Scalable Approximation for Grid inputs. For grid inputs, we present a scalable approximation of Eq. (13):

$$f(D^c)(\cdot) \approx \tag{18}$$
$$\alpha \underbrace{\sum_{q=1}^{Q} \sqrt{z_q}\phi_q(\cdot)}_{\text{prior term}} + \tilde{k} * \Big(\underbrace{\sum_{n=1}^{N^c} y_n^c \delta(\cdot - x_n^c)}_{\text{data term}} - \alpha \underbrace{\sum_{q=1}^{Q} \sqrt{z_q}\phi_q(\cdot)}_{\text{prior term}} \Big),$$

where $\alpha \in (0,1)$ denotes the hyperparameter that controls how much the stationary prior is reflected on $f(D^c)(\cdot)$. $\tilde{k}$ denotes the filter of convolution, and is set to be the truncated expected kernel of Eq. (16) with the finite length (filter window size). This approximation scheme is designed to reflect the stationary prior and the data term on the data representation as $f(D^c)(\cdot)$ constructs the representation in Eq. (13). We demonstrate that Eq. (18) alleviates the task ambiguity on the large-sized image completion task in Section 4.3.

Bayesian ConvDeepsets. With the random data representation $f(D^c)(\cdot)$ in Eqs. (13) and (18), we define a random functional representation $RE(D)(\cdot)$ and Bayesian ConvDeepsets $\Phi^{\mathcal{B}}(D^c)(\cdot)$ in the similar way with ConvDeepsets $\Phi(D^c)(\cdot)$ in Eq. (4) as follows:

$$\underbrace{RE(D^c)(\cdot)}_{\substack{\text{random functional}\\ \text{representation}}} = \Big[\underbrace{\sum_{n=1}^{N^c} k(\cdot - x_n^c)}_{\text{density}} \;,\; \underbrace{f(D^c)(\cdot)}_{\substack{\text{random data}\\ \text{representation}}} \Big],$$
$$\underbrace{\Phi^{\mathcal{B}}(D^c)(\cdot)}_{\substack{\text{Bayes ConvDeepsets}\\ \text{representation}}} = \underbrace{\rho}_{\substack{\text{mapping via}\\ \text{CNN}}} \circ \underbrace{RE(D^c)(\cdot)}_{\substack{\text{random functional}\\ \text{representation}}} \tag{19}$$

We prove that Bayesian ConvDeepsets still holds the translation equivariance (TE) in the following proposition.

Proposition 2. *If Bayesian ConvDeepsets $\Phi^{\mathcal{B}}(D)(\cdot)$ is defined on the finite grid points, $\Phi^{\mathcal{B}}(D)(\cdot)$ is still translation equivariant in distribution sense, i.e.,*

$$\Phi^{\mathcal{B}} \circ (T_\tau(D)) \overset{d}{=} T_\tau^* \circ (\Phi^{\mathcal{B}}(D)). \tag{20}$$

Proof. The proof can be checked in Appendix A.3.1. □

3.4 Prediction and Training

Prediction. NP model using the Bayesian ConvDeepsets builds the predictive distribution $p(Y^t|X^t, D^c)$ as follows:

$$\begin{aligned} &p(Y^t|X^t, D^c) \\ &= \int p(Y^t|X^t, \Phi^{\mathcal{B}}) p(\Phi^{\mathcal{B}}|Z_{\text{Cat}}, D^c)\, p(Z_{\text{Cat}}|D^c)\, d\Phi^{\mathcal{B}} dZ_{\text{Cat}} \\ &\approx \frac{1}{N}\sum_{n=1}^{N} p(Y^t|X^t, \Phi^{\mathcal{B}}_{(n)}(D^c)) \end{aligned} \tag{21}$$

where $\Phi^{\mathcal{B}}(D^c)$ denotes the random representation of the Bayesian ConvDeepsets on finite grid $\{t_m\}_{m=1}^{M}$, and $\Phi^{\mathcal{B}}_{(n)}(D^c)$ denotes the n-th instance. The n-th predictive distribution $p(Y^t|X^t, \Phi^{\mathcal{B}}_{(n)}(D^c))$ is modeled as follows:

$$\begin{aligned} p(Y^t|X^t, \Phi^{\mathcal{B}}_{(n)}(D^c)) &= \prod_{i=1}^{N^t} p\left(y_i^t \mid x_i^t, \Phi^{\mathcal{B}}_{(n)}(D^c)\right) \\ &= \prod_{i=1}^{N^t} N\left(y_i^t \mid \mu_{(n)}(x_i^t), \sigma^2_{(n)}(x_i^t)\right) \end{aligned} \tag{22}$$

where the n-th predictive mean $\mu_{(n)}(x_i^t)$ and standard deviation $\sigma^2_{(n)}(x_i^t)$ are obtained by forwarding the smoothed feature $\tilde{\Phi}^{\mathcal{B}}_{(n)}(D^c)(x_i^t)$, described in Eq. (6), by MLP layer.

Training. Let $\Theta = \{\theta_{\text{kernels}}, \theta_{\text{p}_{\text{nn}}}, \theta_\rho, \theta_{\text{mlp}}\}$ be the learnable parameters for kernels $\{k_q\}_{q=1}^{Q}$ in Eq. (9), the translate invariant network p_{nn} in Eq. (11), the CNN ρ in Eq. (19), and the MLP layer in Eq. (22). Then, based on meta-learning framework, the NP model is trained by maximizing the following objective with respect to the parameters Θ:

$$\mathbb{E}_{D^c, D^t \sim p(\mathcal{T})}\left[\mathcal{L}_{\text{ll}}(\Theta; D^c, D^t) - \beta \mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)\right] \tag{23}$$

where $\mathcal{L}_{\text{ll}}(\Theta; D^c, D^t)$ denotes the log likelihood, and $\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)$ denotes the regularizer that induces the network $\text{p}_{\text{nn}}(D^c)$ in Eq. (10). β denotes the hyperparameter that controls the effect of the regulaizer.

The log likelihood is expressed as follows:

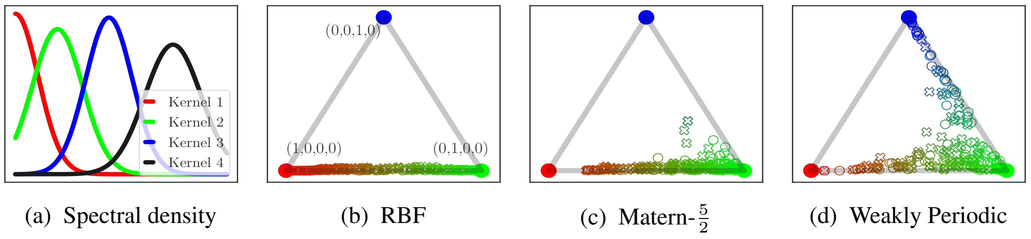
$$\log\left(\frac{1}{N}\sum_{n=1}^{N}\exp\left(\sum_{i=1}^{N^t}\log p\left(y_i^t \mid x_i^t, \Phi^{\mathcal{B}}_{(n)}(D^c)\right)\right)\right),$$

which can be optimized in end-to-end manner based on the automatic differentiation by [Jang et al., 2016, Jung et al., 2020].

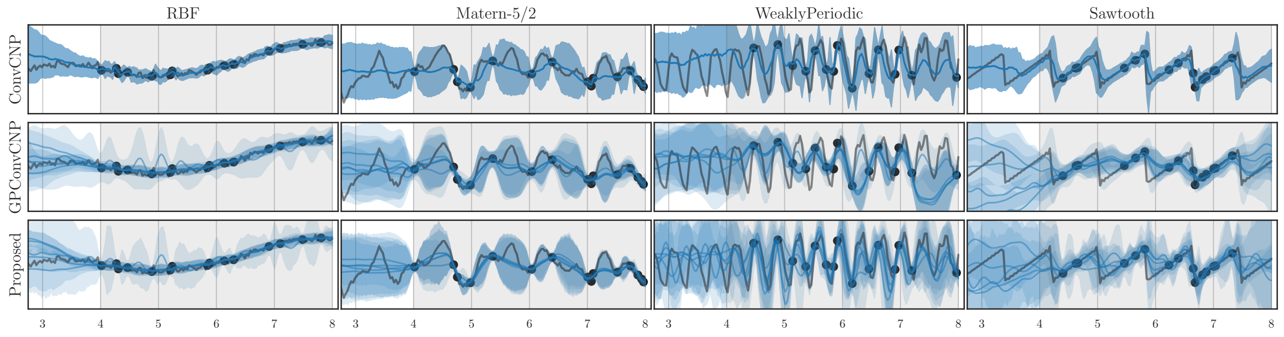
The regularizer $\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)$ is expressed as follows:

$$\text{KL}\left(q(Z_{\text{cat}} \mid D^c) \,||\, p(Z_{\text{cat}} \mid D^c, D^t)\right)$$

where $q(Z_{\text{cat}}|D^c)$ denotes the amortized categorical distribution in Eq. (10). $p(Z_{\text{cat}}|D^c, D^t)$ denotes the posterior distribution, which will be used as the prior distribution to train



(a) Spectral density (b) RBF (c) Matern-$\frac{5}{2}$ (d) Weakly Periodic (e) Sawtooth



(f) Predictive distribution of each baseline on each process ($N^c = 15$)

Figure 3: Stochastic processes modeling beyond training range (gray region $[4, 8]$): Fig. 3a denotes the trained spectral density of Eq. (9) with $Q = 4$ kernels. Figs. 3b to 3e denote the output of $\mathrm{p_{nn}}(D^c)$ for 128 context set $\{D^c_n\}_{n=1}^{128}$ per each process. Fig. 3f shows the predictive distributions of the baselines on each process.

network $p_{\text{nn}}(D^c)$ in $q(Z_{\text{cat}} \mid D^c)$. Note that $p(Z_{\text{cat}}|D^c, D^t)$ can be computed only during training phase.

The posterior $p(Z_{\text{cat}}|D^c, D^t)$ is modeled as a categorical distribution $\text{Cat}(Z_{\text{cat}}; \mathrm{p_{prior}})$ with the parameter $\mathrm{p_{prior}} \in \Delta^{Q-1}$, where the q-th element $(\mathrm{p_{prior}})_q$ is defined:

$$(\mathrm{p_{prior}})_q = \frac{\exp\left(\log p(Y^t|X^t, D^c, k_q) \;/\; \tau_0\right)}{\sum_{q=1}^{Q} \exp\left(\log p(Y^t|X^t, D^c, k_q) \;/\; \tau_0\right)} \quad (24)$$

where τ_0 denotes the temperature parameter. The marginal likelihood $p(Y^t|X^t, D^c, k_q)$, using the q-th stationary kernel k_q, is computed as

$$p(Y^t|X^t, D^c, k_q) = N\left(Y^t; \hat{\mu}_q(X^t), \text{Diag}(\hat{\sigma}^2_q(X^t))\right)$$

where $\hat{\mu}_q(X^t) \in R^{N_t}$ and $\text{Diag}(\hat{\sigma}^2_q(X^t)) \in R^{N_t \times N_t}$ denote the empirical predictive mean and diagonal covariance of the GP posterior distribution on target set X^t. For computational efficiency, we use GP posterior sample functions in Eq. (13). The derivation of the training objective and further details are described in Appendix A.3.3.

4 Experiments

In this section, we validate the following main question:

Does the Bayesian ConvDeepsets alleviate the task ambiguity arising from the small number of context data points?

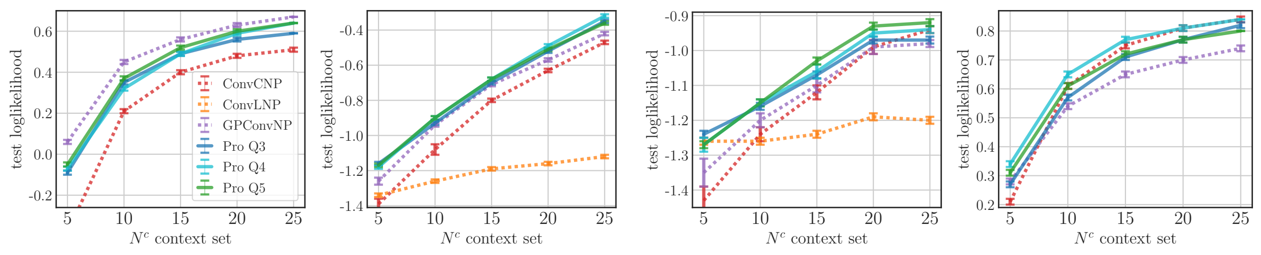
We validate this question on the 1-d regression in Section 4.1, multi-channel regression in Section 4.2, and large-sized image completion in Section 4.3. The detailed setting and additional results are given in Appendices B to D. Our implementation is available at https://github.com/becre2021/BayesConvdeepset.

4.1 Single-Channel Regression

We conduct the 1-d regression task based on meta learning framework; given each context set $D^c = \{(x^c_n, y^c_n)\}_{n=1}^{N^c}$, the model constructs the predictive distribution for target set $D^t = \{(x^t_n, y^t_n)\}_{n=1}^{N^t}$. Specially, we consider small N^c for the training context set D^c to investigate how N^c data points affects the ConvDeepsets representation and the modeling performance. For each task, we set the number of context points $N^c \sim U(5, 25)$ and target points $N^t \sim U(N^c, 50)$, using half the number of context data points compared to the setting of the ConvCNP [Gordon et al., 2019]. In addition, we consider different task diversity to investigate whether task-dependent prior can be effective even when N^c is insufficiently small to recognize each task.

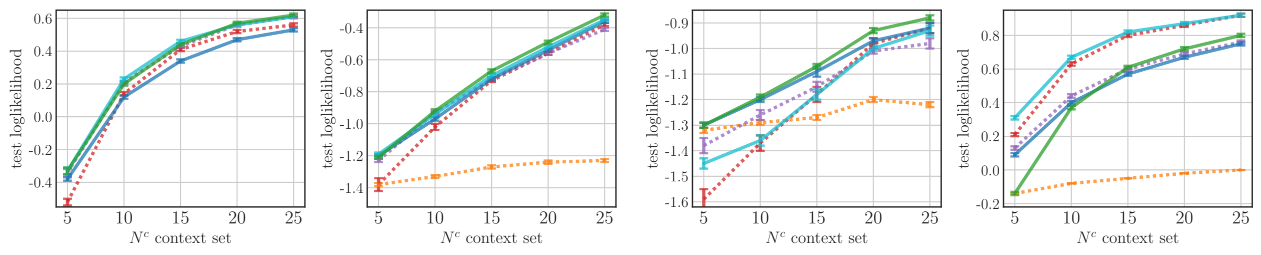
For training and validation, we use the tasks generated on $[0, 4]$ (training range). For test, we evaluate the trained models with the tasks on $[4, 8]$ to check the TE in Eq. (20).

Dataset. We use 4 different stochastic processes used in [Gordon et al., 2019]: GP using the RBF, Matern-$\frac{5}{2}$, and Weakly periodic kernel, and sawtooth process that is not GP. For task diversity, we randomly generate each task by choosing one of these processes and use it for training. This setting differs from that conducted in [Gordon et al., 2019, Foong et al., 2020], where NP models are trained and



(a) RBF (b) Matern-$\frac{5}{2}$ (c) Weakly periodic (d) Sawtooth

Figure 4: Results of the training with $N^c \sim U(5, 25)$ context points.



(a) RBF (b) Matern-$\frac{5}{2}$ (c) Weakly periodic (d) Sawtooth

Figure 5: Results of the training with $N^c \sim U(10, 50)$ context points.

evaluated independently on each stochastic process.

Baselines. We use the Attentive NP (ANP) [Kim et al., 2018], ConvCNP [Gordon et al., 2019], ConvLNP [Foong et al., 2020], GPConvCNP [Petersen et al., 2021] that uses the sample function of GP posterior with RBF kernel.

Results. Fig. 3 describes the imposed priors per each process and the corresponding prediction results. Fig. 3a displays the spectral densities $\{p_q(s)\}_{q=1}^{4}$ for the trained kernels ($Q = 4$). Figs. 3b to 3e show the output of $\mathrm{p_{nn}}(D^c)$, i.e. the parameters of Cat $(Z_{\text{cat}}; \mathrm{p_{nn}}(D^c))$ for 128 context sets $\{D^c_n\}_{n=1}^{128}$ per each process. Fig. 3f shows the prediction results for one task ($N^c = 15$), where each column and row shows the task of each process and the NP models.

The proposed model imposes the stationary prior differently depending on the tasks of each process, and the prediction on RBF and WeaklyPeriodic task is affected by the different stationary prior of kernels 1 and 3.

Fig. 4 describes the mean and one-standard error of test log likelihood for 1024 tasks (beyond training range); for evaluation, we set the varying number of context data points $N^c \in \{5, 10, 15, 20, 25\}$ and target points $N^t = 50$ to investigate the effectiveness of the proposed method. Fig. 5 denotes the corresponding result when the models are trained with the larger number of the context data points $N^c \sim U(10, 50)$. We obtain the following observations:

- When the small N^c context points are allowed for training, the NP model using Bayesian ConvDeepsets ($Q \in \{3, 4, 5\}$) predicts the target set most accurately (high log likelihood) for the tasks of each process; for small context points $N^c \in \{5, 10\}$, the proposed method outperforms ConvCNP with a large margin.
- GPConvCNP and ConvCNP show good prediction performance on specific process (GPConvCNP: RBF, ConvCNP: Sawtooth), whereas the proposed method show good prediction performance on all processes. ANP shows the bad generalization on tasks beyond the training range, and thus can not be reported together.
- When the large N^c context points are allowed for training, the NP model ($Q = 4$) still shows superior prediction performance. However, the performance gap between the proposed model and ConvCNP is reduced. This indicates that the proposed Bayesian Convdeepsets is significantly effective for the targeted issue arising from the small number of context data points.

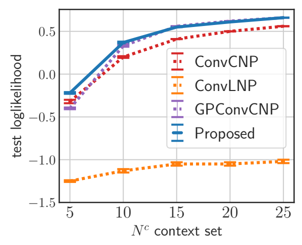
Additional studies for the computational efficiency of the data representation $f(D^c)(\cdot)$ in Eq. (13) and the effect of the size of amortized neural network $\mathrm{p_{nn}}(D^c)$ in Eq. (11) are investigated in Appendix B.4.

4.2 Multi-Channel Regression

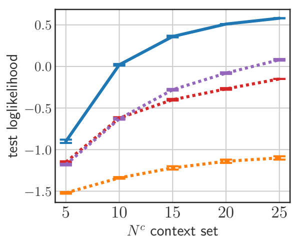
Experiment setting. We conduct the multi-channel regression task. We consider different task diversities to investigate the effectiveness of the task-dependent prior, especially for a small number of context points $N^c \sim \mathcal{U}([5, 25])$.

Dataset. We use 3 channels sinusoidal process where i-channel function f_i is represented as

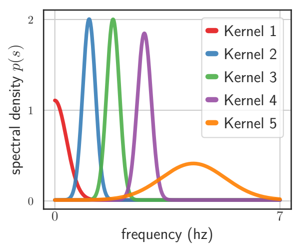
$$f_i(t) = A_i \sin\left(2\pi(w_i + \theta_i)(t - \phi_i)\right) + \epsilon, \quad \epsilon \sim N(0, 0.1^2),$$



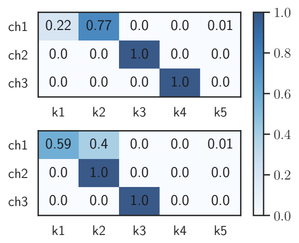
(a) Sinusoidal-phase



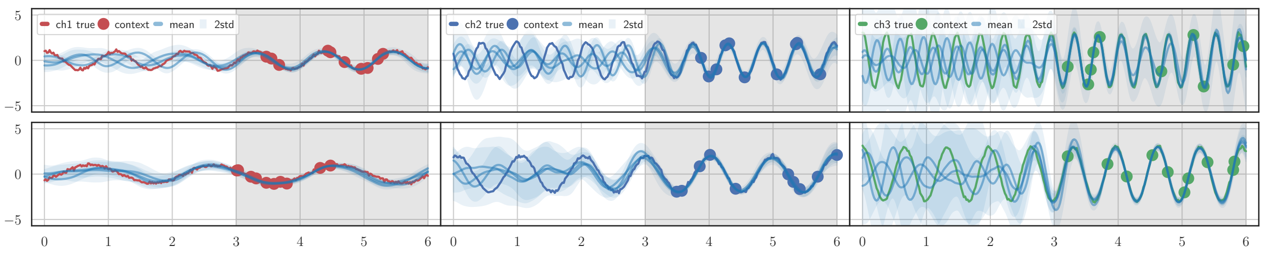
(b) Sinusoidal-all



(c) Spectral density



(d) Task-dependent prior



(e) Prediction for 2 different tasks ($N^c = 10$) of the sinusoidal-varying

Figure 6: 3 channel sinusoidal processes modeling: Figs. 6a and 6b denotes test likelihood on sinusoidal processes having different task diversity. Fig. 6c denote the trained spectral density of Eq. (9) for the Sinusoidal-all process, and Figs. 6d and 6e show the chosen prior and corresponding prediction for 2 tasks; as the context sets having small frequency characteristics are given (from first to second row in Fig. 6e), the stationary prior is imposed differently (from top to bottom in Fig. 6d)

where $\{A_i, w_i, \phi_i\}_{i=1}^{3}$ denotes the i-channel amplitude, frequency, and phase parameters. For task diversity, one process is generated from the fixed amplitude A_i, frequency w_i, and a varying phase parameter ϕ_i that is sampled differently for each task. Another process is generated from the varying amplitude, frequency, and phase, which is regarded to have high diversity. For the details, see Appendix C.1.

Results. Figs. 6a and 6b describe the mean and one-standard error of test log likelihood for 1024 tasks beyond training range. Fig. 6a shows the less diversity task (varying phase), and Fig. 6b corresponds to that of the high diversity task (varying amplitude, frequency, and phase). These figures show the proposed method can model all sinusoidal processes tasks well even with high diversity tasks. Fig. 6c shows the trained spectral densities $\{p_q(s)\}_{q=1}^{5}$ for the high diversity tasks. Fig. 6d denotes the parameters $\mathrm{p_{nn}}(D^c)$ for two tasks (top and bottom), and Fig. 6e shows the corresponding predictions. This demonstrates how task-dependent priors are imposed and affect prediction on target set. We confirm similar results on another dataset (GP with MOSM kernel [Parra and Tobar, 2017]) in Appendix C.4.

4.3 Image Completion

Experiment setting. We conduct the image completion task using a monthly land surface temperature set used in [Remes et al., 2017] because the temperature exhibits different local stationarity depending on the month. Each task is to predict the image of the temperature when only partial pixel values are given. The temperature of North America (53×115×3) and Europe (78×102×3) are used for the training (2018 - 2020), and for the test (2021).

For each task, we set N^c and N^t to be proportional to the size of the image by randomly choosing a low context rate $p \in \{.01, .05, .10, .20\}$ with probability $[4/10, 3/10, 2/10, 1/10]$. This setting is indented to investigate how NP models predict the image when trained with a small number of context points N^c in each context set D^c.

We compare the following baselines: ConvCNP, the proposed NP using the scalable approximation of Eq. (18) with Q=1 (task-independent RBF prior), and Q=4 (task-dependent prior). We do not compare GPConvCNP and the proposed NP using the random representation Eq. (13) because constructing random data representations with large-sized images (North America: 53×115×3 and Europe: 78×102×3) caused memory issue for conventional and path-wise GP posterior sampling.

For the prior coefficient α in Eq. (18), we consider $\alpha \in \{.05, .10\}$ and pick $\alpha = .10$ having the best performance on the validation set. Additionally, the effect of α is investigated in Appendix D.4.

Results. Fig. 7 shows the Europe images (June and December of 2021), context sets with $p \in \{.05, 0.10\}$, and the corresponding predictive mean of each model. The parameters, obtained by 10000 training tasks, are used for prediction results. Compared to the targeted image in the leftmost column, the proposed NP yields more accurate predictions than ConvCNP.

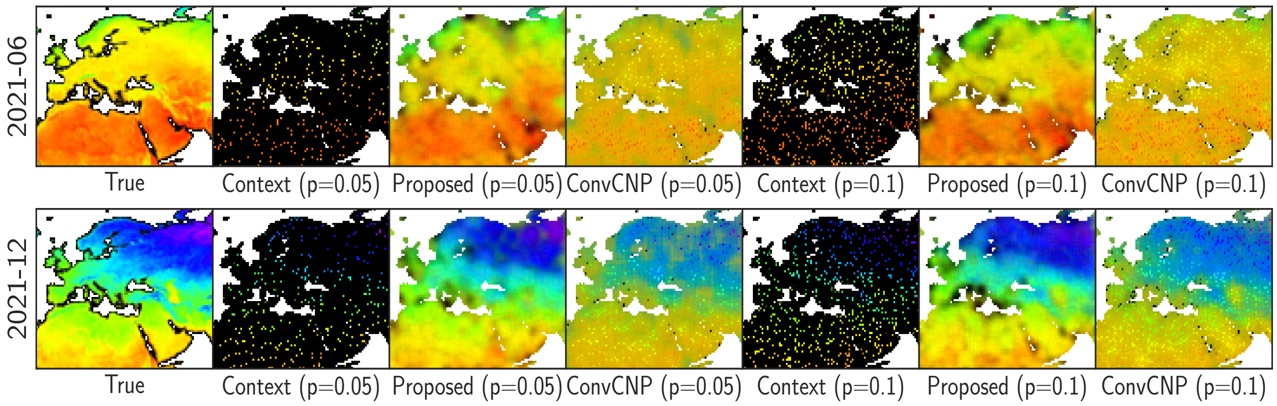


Figure 7: Prediction results for two Europe temperatures (June 2021 and December 2021) that is out of training sets.

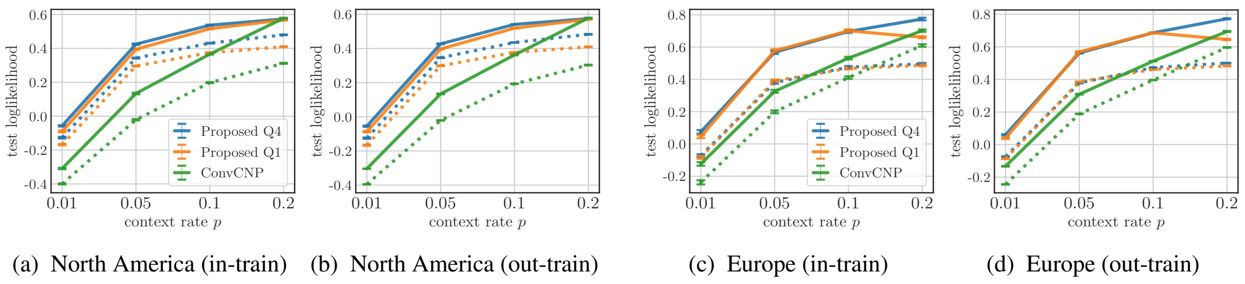


Figure 8: Prediction results of each 256 tasks over context rates $p \in \{.01, .05, .10, .20\}$ and target rate $p = .50$.

Figs. 8a and 8c describe the log likelihood for North America and Europe datasets (2018 - 2020), respectively; the dotted $(\cdot)$ and straight (-) line denote the results obtained from training on 5000 and 10000 tasks, respectively. Figs. 8b and 8d show the corresponding result of out-of-training set (2021). We compare ConvCNP and the proposed NP with $Q = 1$ (task-independent RBF prior) and $Q = 4$ (task-dependent prior) using approximation scheme in Eq. (18). Based on these results, we obtain the following conclusions:

- The proposed NP ($Q = 4$) outperforms ConvCNP on small context sets ($p \in .01, .05$) and 5000 training tasks. This indicates that task-dependent prior is effective when the small number of the context data points and the context sets are available during the training phase.

- The proposed NP ($Q = 4$) outperforms the proposed NP ($Q = 1$) in general. This indicates the effectiveness of task-dependent stationary prior because Q=1 (task-independent RBF prior) can be regarded as the scalable approximation of the GPConvCNP on grid inputs.

- As the number of training tasks and context data points increases, the performance gap between ConvCNP and the proposed NP becomes smaller. This indicates the task-dependent prior becomes less significant when a large amount of the training datasets is available.

5 Discussion

Comparison with ConvLNP. Convolutional latent neural process (ConvLNP) [Foong et al., 2020] considers the latent variable into ConvDeepsets to relax the conditional independence assumption of the ConvCNP. However, ConvLNP could have the same issue as ConvCNP because it uses ConvDeepsets. Also, training the ConvLNP is known to be difficult in Section 7, [Gordon, 2021]; we believe this difficulty arises because the latent variables are constructed from the ambiguous representation of ConvDeepsets, and the randomness of the latent variable prevents the NP model from fitting the target as well. On the other hand, the proposed method uses the multiple data representations via the task-dependent stationary prior and uses them.

Comparison with GPConvCNP. GPConvCNP [Petersen et al., 2021] uses the sample function of GP Posterior (RBF kernel) as the functional representation, which is similar to our work. However, our work generalizes the ConvDeepsets in a Bayesian way; that is, we propose how to impose the task-dependent prior. Also, we build the data representation in more computational efficient way and scalable way (grid inputs). Last, we prove that the Bayesian ConvDeepsets still holds the TE as shown in Proposition 2.

Limitation. Bayesian ConvDeepsets uses inductive bias of stationarity and its effectiveness can be limited when the targeted process cannot be modeled by stationary process.

Acknowledgements

This work is supported by the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (grant RS-2019-KA153277). Specially, we are grateful to Prof. Juho Lee for introducing us to Convolutional NP works and Prof. Kyungwoo Song for discussing the manuscript direction, respectively. We would also like to thank Federico Berto for helping us prepare the image datasets, Fergus Simpson for sharing the implementation about the Minecraft kernel, and Geoff Pleiss for advising on implementing the CG solver in an automatic differentiation way. Finally, we thank anonymous reviewers for their constructive feedback, which has improved the quality of our work.

References

S. Bochner. *Lectures on Fourier integrals*. Princeton University Press, 1959.

W. P. Bruinsma, J. Requeima, A. Y. Foong, J. Gordon, and R. E. Turner. The gaussian neural process. *arXiv preprint arXiv:2101.03606*, 2021.

Y. Burda, R. B. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations*, 2015.

T. de Wolff, A. Cuevas, and F. Tobar. MOGPTK: The Multi-Output Gaussian Process Toolkit. *Neurocomputing*, 2020. ISSN 0925-2312. doi:https://doi.org/10.1016/j.neucom.2020.09.085. URL `https://github.com/GAMES-UChile/mogptk`.

C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

C. Finn, K. Xu, and S. Levine. Probabilistic model-agnostic meta-learning. *Advances in neural information processing systems*, 31, 2018.

A. Foong, W. Bruinsma, J. Gordon, Y. Dubois, J. Requeima, and R. Turner. Meta-learning stationary stochastic process prediction with convolutional neural processes. *Advances in Neural Information Processing Systems*, 33, 2020.

M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. Rezende, and S. A. Eslami. Conditional neural processes. In *International Conference on Machine Learning*, pages 1704–1713. PMLR, 2018a.

M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. M. A. Eslami, and Y. W. Teh. Neural processes, 2018b.

J. Gordon. *Advances in Probabilistic Meta-Learning and the Neural Process Family*. PhD thesis, University of Cambridge, 2021.

J. Gordon, W. P. Bruinsma, A. Y. Foong, J. Requeima, Y. Dubois, and R. E. Turner. Convolutional conditional neural processes. In *International Conference on Learning Representations*, 2019.

P. Holderrieth, M. J. Hutchinson, and Y. W. Teh. Equivariant learning of stochastic fields: Gaussian processes and steerable conditional neural processes. In *International Conference on Machine Learning*, pages 4297–4307. PMLR, 2021.

E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2016.

Y. Jung, K. Song, and J. Park. Approximate inference for spectral mixture kernel. *arXiv preprint arXiv:2006.07036*, 2020.

Y. Jung, K. Song, and J. Park. Efficient approximate inference for stationary kernel on frequency domain. In *International Conference on Machine Learning*, pages 10502–10538. PMLR, 2022.

M. Kawano, W. Kumagai, A. Sannai, Y. Iwasawa, and Y. Matsuo. Group equivariant conditional neural processes. In *International Conference on Learning Representations*, 2020.

H. Kim, A. Mnih, J. Schwarz, M. Garnelo, A. Eslami, D. Rosenbaum, O. Vinyals, and Y. W. Teh. Attentive neural processes. In *International Conference on Learning Representations*, 2018.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

C. Louizos, X. Shi, K. Schutte, and M. Welling. The functional neural process, 2019.

E. A. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964.

G. Parra and F. Tobar. Spectral mixture kernels for multi-output gaussian processes. In *Advances in Neural Information Processing Systems*, pages 6681–6690, 2017.

J. Petersen, G. Köhler, D. Zimmerer, F. Isensee, P. F. Jäger, and K. H. Maier-Hein. Gp-convcnp: Better generalization for convolutional conditional neural processes on time series data. *arXiv preprint arXiv:2106.04967*, 2021.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

S. Remes, M. Heinonen, and S. Kaski. Non-stationary spectral kernels. *Advances in neural information processing systems*, 30, 2017.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation.

In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

G. Singh, J. Yoon, Y. Son, and S. Ahn. Sequential neural processes. *Advances in Neural Information Processing Systems*, 32, 2019.

J. Wilson, V. Borovitskiy, A. Terenin, P. Mostowsky, and M. Deisenroth. Efficiently sampling functions from gaussian process posteriors. In *International Conference on Machine Learning*, pages 10292–10302. PMLR, 2020.

J. Yoon, G. Singh, and S. Ahn. Robustifying sequential neural processes. In *International Conference on Machine Learning*, pages 10861–10870. PMLR, 2020.

M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. *Advances in Neural Information Processing Systems*, 30, 2017.

A Further Details for Proposed Methodology

A.1 Implementations

We explain how the proposed NP model outputs the predictive distributions on target sets in following algorithm.

Algorithm 1 Forward pass of the NP model via Bayesian ConvDeepsets (off-grid)

```
Require: Model parameters: $\Theta$
Require: The number of basis stationary kernels $Q$, the number of sampled spectral points $l$
Require: Context set $D^c = \{(x_n^c, y_n^c)\}_{n=1}^{N^c}$, target inputs $X^t = \{x_n^t\}_{n=1}^{N^t}$
1: Compute the discretized inputs $\{t_m\}_{m=1}^{M}$ by spacing inputs range $[\min\{X^c, X^t\}, \max\{X^c, X^t\}]$ linearly.
2: Select the task-dependent stationary prior as described in Eq. (14).
3: Construct $N$ random functional representation $\{f_{(n)}(D^c)(\cdot)\}_{n=1}^{N}$ on $\mathcal{I}$ as described in Eq. (13).
4: Construct $N$ random representations $\{\Phi_{(n)}^{\mathcal{B}}(D^c)(\cdot)\}_{n=1}^{N}$ on finite grid points $\{t_m\}_{m=1}^{M}$ as described in Eq. (19).
5: Smooth $N$ random representations $\{\Phi_{(n)}^{\mathcal{B}}(D^c)(\cdot)\}_{n=1}^{N}$ on target inputs $X^t$ by .
6: Generate $N$ predictive distributions $\{p\big(Y^t \mid X^t, \Phi_{(n)}^{\mathcal{B}}\big)\}_{n=1}^{N}$ on target inputs $X^t$ by Eq. (22).
```

Algorithm 2 Forward pass of the NP model via Bayesian ConvDeepsets (on-grid)

```
Require: Model parameters: $\Theta$
Require: The number of basis stationary kernels $Q$, the number of sampled spectral points $l$
Require: Context set $D^c = \{(x_n^c, y_n^c)\}_{n=1}^{N^c}$, target inputs $X^t = \{x_n^t\}_{n=1}^{N^t}$ on grid.
1: Extract the context inputs $\sum_{n=1}^{N^c} \delta(\cdot - x_n^c)$ and outputs $\sum_{n=1}^{N^c} y_n^c \delta(\cdot - x_n^c)$ on grid.
2: Construct $N$ random functional representation $\{f_{(n)}(D^c)(\cdot)\}_{n=1}^{N}$ on given grid as described in Eq. (18).
3: Map $N$ random representations on grid where the $n$-th representation $f_{(n)}^M$ is computed as
       $$f_{(n)}^M = \mathrm{CNN}\left(\left[\sum_{n=1}^{N^c} \delta(\cdot - x_n^c)\ ,\ \Phi_{(n)}^{\mathcal{B}}(D^c)(\cdot)\right]\right).$$
4: Generate $N$ predictive distributions for target set with the $n$-th predictive mean $\mu_{(n)}^t$ and standard deviation $\sigma_{(n)}^t$,
       $$\mu_{(n)}^t\ ,\ \sigma_{(n)}^t = f_{(n)}^M \circ \left(\sum_{n=1}^{N^t} \delta(\cdot - x_n^t)\right).$$
```

A.2 Structure of Neural Network

A.2.1 Structure of Translation Invariant NN in Eq. (11).

In this work, we consider the following structure for the translation invariant network $\mathrm{p_{nn}}(D^c)$.

Version 1 (off grid). With the q-th stationary kernel k_q, we construct the q-th representation $h_q(\cdot) = \sum_{n=1}^{N} y_n^c \, k_q(\cdot - x_n^c)$ for given context set context set $D^c = \{(x_n^c, y_n^c)\}_{n=1}^{N^c}$, and map the concatenated representations via MLPs as follows:

$$\mathrm{p_{nn}}(D^c) \coloneqq \Phi_{\text{MLP-2}}\Bigg(\sum_{(x^c, y^c) \in D^c} \Phi_{\text{MLP-1}} \bigg(\Big[h_1(x^c),\ h_2(x^c), .., \ h_Q(x^c),\ y^c \Big] \bigg) \Bigg). \tag{25}$$

This structure employs the structure of the kernel smoother and the Deepsets. Since $h_q(\cdot)$ function is invariant to the translation dataset $T_\tau(D^c)$, this structure satisfies $\mathrm{p}_{nn}(T_\tau(D^c)) = \mathrm{p}_{nn}(D^c)$ with $T_\tau(D^c) = \{(x_n^c + \tau, y_n^c)\}_{n=1}^{N^c}$.

Version 2 (off grid). With the RBF kernel function k_{RBF}, we use the data representation of ConvDeepsets on grid $\{t_m\}_{m=1}^{M}$, and map this representation to the parameters of the categorical distribution as follows:

$$\mathrm{p_{nn}}(D^c) \coloneqq \Phi_{\text{MLP}} \Bigg(\sum_{\cdot \in \{t_m\}_{m=1}^{M}} \mathrm{CNN_{1D}} \bigg(\sum_{n=1}^{N} \frac{y_n \, k_{\text{RBF}}(\cdot - x_n)}{\sum_{n=1}^{N} k_{\text{RBF}}(\cdot - x_n)} \bigg) \Bigg) \tag{26}$$

where $\mathrm{CNN_{1D}}$ denotes a stack of convolutional layer with ReLU activation. This structure employs the structure of the RBF kernel smoother used in ConvDeepsets and the Deepsets that ensures the set invariance. Since adding or averaging the representation of RBF smoother over the finite grid is invariant to the translation dataset $T_\tau(D^c)$, this structure satisfies $\mathrm{p_{nn}}(T_\tau(D^c)) = \mathrm{p_{nn}}(D^c)$ with $T_\tau(D^c) = \{(x_n^c + \tau, y_n^c)\}_{n=1}^{N^c}$.

Here, we use the representation of RBF smoother to obtain the parameter $\mathrm{p_{nn}}(D^c)$ of a latent Categorical distribution $\mathrm{Cat}(Z_{\text{Cat}} | \mathrm{p_{nn}}(D^c))$ so that $\mathrm{p_{nn}}(D^c)$ assigns a task-dependent stationary prior. This approach is different to the use of ConvDeepsets in [Gordon et al., 2019] that represents the data.

Version 3 (on grid). We extend **version 2** to obtain the parameter of Categorical distribution for the image dataset D^c as follows:

$$\mathrm{p_{nn}}(D^c) \coloneqq \Phi_{\text{MLP}} \Bigg(\text{Pooling} \bigg(\mathrm{CNN_{2D}} \Big(\sum_{n=1}^{N} y_n \, \delta(\cdot - x_n) \Big) \bigg) \Bigg) \tag{27}$$

where $\mathrm{CNN_{2D}}$ denotes a stack of residual block.

A.2.2 Structure of CNN in Eq. (19)

Shallow and Deep CNN (1d). For regression tasks with time-series dataset, we employ 5-layer CNN using ReLU activation function as a shallow network. All layers uses 16 channels except for the first layer (8 inchannels) and last layer (8 outchannels). We use kernels of size 5, stride 1, and zero.

For the structure of Deep CNN, we employ the 1D-Unet [Ronneberger et al., 2015], which is used in [Gordon et al., 2019]. 1D-Unet consists of 12-layer architecture with skip connections where the number of channels is doubled every layer for the first six layers, and halved every layer for the last six layers. The following describes which layers are concatenated, where $L_i \leftarrow [L_j, L_k]$ means that the input to layer i is the concatenation of the activations of layers j and k:

- $L_8 \leftarrow [L_5, L_7], \quad L_9 \leftarrow [L_4, L_8], \quad L_{10} \leftarrow [L_3, L_9], \quad L_{11} \leftarrow [L_2, L_{10}], \quad L_{12} \leftarrow [L_1, L_{11}].$

We use ReLU activation function, kernels of size 5, stride 1, and zero padding for two units on all layers.

Shallow and Deep CNN (2d). For image completion tasks, we use 3-layer of residual block as a shallow network and 6-layer of residual block as a deep network where each block uses 128 channel, and kernels of size 5, stride 1, and zero.

Further details of DNN structure used in this work can be checked in our implementation at the attached github link.

A.3 Mathematical Details

A.3.1 Proof of Proposition 2

Proposition 2. *If Bayesian ConvDeepsets $\Phi^{\mathcal{B}}(D)(\cdot)$ is defined on the finite grid points, $\Phi^{\mathcal{B}}(D)(\cdot)$ is still translation equivariant in distribution sense, i.e.,*

$$\Phi^{\mathcal{B}} \circ (T_\tau(D)) \overset{d}{=} T^*_\tau \circ (\Phi^{\mathcal{B}}(D)). \tag{28}$$

Proof. Let us first consider the left side of equation in Eq. (28). Then, for given dataset $D = \{(x_n, y_n)\}_{n=1}^N$, its τ-translated dataset $T_\tau(D) = \{(x_n + \tau, y_n)\}_{n=1}^N$, and finite grid $\{t_m\}_{m=1}^M$, we can compute $\Phi^{\mathcal{B}}(T_\tau(D))(\cdot)$ as follows:

$$\Phi^{\mathcal{B}}(T_\tau(D))(\cdot) = \rho \circ \Big[\underbrace{\sum_{n=1}^N k_{\mathrm{RBF}}(\cdot - (x_n + \tau))}_{\text{density}} \; , \; \underbrace{\sum_{q=1}^Q \sqrt{z_q^\tau}\, \phi_q^\tau(\cdot) + \sum_{n=1}^N v_n^\tau \, k(\cdot - x_n + \tau)}_{\text{data representation}} \Big]. \tag{29}$$

where z_q^τ, $\phi_q^\tau(\cdot)$, and v_n^τ denote the variation of z_q, $\phi_q(\cdot)$, and v_n due to τ-translated dataset $T_\tau(D)$. The kernel function $k(\cdot - x_n + \tau)$ denotes the expectation of the latent kernel $\mathrm{E}_{q(Z_{\mathrm{cat}}|D)}[k(\cdot - x_n) \mid Z_{\mathrm{cat}}]$ over $q(Z_{\mathrm{cat}}|D) = \mathrm{Cat}\,(Z_{\mathrm{cat}} \; ; \; \mathrm{p_{nn}}(D))$ with the translated invariant network $\mathrm{p_{nn}}(D)$ in Eq. (11). Since it has already been proven in [Gordon et al., 2019] that mapping of density channel by CNN holds the TE property, we thus focus on validating that mapping of data representation by CNN holds the TE as well.

Prior term. For the prior term, we check the following equalities:

$$(z_1^\tau, .., z_Q^\tau) \;\overset{d}{=}\; (z_1, .., z_Q) \tag{30}$$

where $(z_1^\tau, .., z_Q^\tau) \sim \mathrm{Cat}(Z; \mathrm{p_{nn}}(T_\tau(D)))$ and $(z_1, .., z_Q) \sim \mathrm{Cat}(Z; \mathrm{p_{nn}}(D)$. This equality holds because the output of translated invariant network, i.e., the parameter of categorical distribution satisfies $\mathrm{p_{nn}}(T_\tau(D)) = \mathrm{p_{nn}}(D)$.

For the stationary prior term, each q-th random stationary function $\phi_q^\tau(\cdot)$ denotes a sample function of $GP(f; 0, k_q)$ obtained by random Fourier Feature; the q-th kernel $k_q(\tau)$ is set to be $k_q(\tau) = \int e^{i2\pi s^T \tau} p_q(s) ds$. Since the covariance matrix generated by each stationary kernel function $k_q(\tau)$ is independent to the translated dataset $T_\tau(D)$, thus it is trivial that

$$\phi_q^\tau(\cdot) \overset{d}{=} \phi_q(\cdot). \tag{31}$$

These results of Eqs. (30) and (31) imply that the task-dependent prior function $\sum_{q=1}^Q \sqrt{z_q^\tau} \phi_q^\tau(\cdot)$ is set to be consistent up to $T_\tau(D) = \{(x_n + \tau, y_n)\}_{n=1}^N$.

Update term. For the data update term, we check that the expected kernel holds as follow:

$$\begin{aligned}
\mathrm{E}_{q(Z_{\mathrm{cat}}|T_\tau(D))}[\, k(\cdot) \mid Z_{\mathrm{cat}} \,] &= \mathrm{E}_{q(Z_{\mathrm{cat}}|T_\tau(D))}[z_1 k_1(\cdot) \;+\; \cdots \;+\; z_Q k_Q(\cdot)] \\
&= \mathrm{p_{nn}}(T_\tau(D))_{(1)} k_1(\cdot) \;+\; \cdots \;+ \mathrm{p_{nn}}(T_\tau(D))_{(Q)} k_Q(\cdot) && (32) \\
&= \mathrm{p_{nn}}(D)_{(1)} k_1(\cdot) \;+\; \cdots \;+ \mathrm{p_{nn}}(D)_{(Q)} k_Q(\cdot) && (33) \\
&= \mathrm{E}_{q(Z_{\mathrm{cat}}|D)}[\, k(\cdot) \mid Z_{\mathrm{cat}} \,], && (34)
\end{aligned}$$

where $\mathrm{p_{nn}}(D)_{(q)}$ denotes q-th element of $\mathrm{p_{nn}}(D)$. The third equality holds due to the property of the translated invariant network $\mathrm{p_{nn}}(T_\tau(D)) = \mathrm{p_{nn}}(D)$.

Also, we confirm that the smoothing weight v_n^τ holds as follows:

$$v_n^\tau := [\, K(X_\tau, X_\tau) + \sigma_\epsilon^2 I)^{-1} \, (Y^c - \Psi(X_\tau)) \,]_n = [\, K(X, X) + \sigma_\epsilon^2 I)^{-1} \, (Y^c - \Psi(X)) \,]_n = v_n, \tag{35}$$

where $X_\tau = \{x_n + \tau\}_{n=1}^N$ and $X = \{x_n\}_{n=1}^N$. The first equality holds because the expected kernel $k(\cdot)$ is set consistently up to the translated inputs as described in Eq. (34), the expected kernel $k(\cdot)$ is stationary kernel, and the random stationary function $\Psi(X_\tau)$ is set consistently up to the translated inputs as described in **Prior term**.

Prior term + Update term. Using the above results, we show that if $\Phi^{\mathcal{B}} \circ (T_\tau(D))$ and $T_\tau^* \circ (\Phi^{\mathcal{B}}(D))$ are evaluated on finite grid points $\{t_m\}_{m=1}^M$ respectively, then $\Phi^{\mathcal{B}} \circ (T_\tau(D)) \overset{d}{=} T_\tau^* \circ (\Phi^{\mathcal{B}}(D))$ holds as follows:

$$\Phi^{\mathcal{B}}(T_\tau(D))(\cdot) = \rho \circ \Big[\sum_{n=1}^{N} k_{\text{RBF}}(\cdot - (x_n + \tau)) \, , \, \sum_{q=1}^{Q} \sqrt{z_q^\tau} \, \phi_q^\tau(\cdot) + \sum_{n=1}^{N} v_n^\tau \, k(\cdot - (x_n + \tau)) \Big] \tag{36}$$

$$\overset{d}{=} \rho \circ \Big[\sum_{n=1}^{N} k_{\text{RBF}}(\cdot - (x_n + \tau)) \, , \, \sum_{q=1}^{Q} \sqrt{z_q} \, \phi_q(\cdot) + \sum_{n=1}^{N} v_n \, k(\cdot - (x_n + \tau)) \Big] \tag{37}$$

$$\overset{d}{=} \rho \circ \Big[\sum_{n=1}^{N} k_{\text{RBF}}(\cdot - (x_n + \tau)) \, , \, \sum_{q=1}^{Q} \sqrt{z_q} \, \phi_q(\cdot - \tau) + \sum_{n=1}^{N} v_n \, k(\cdot - (x_n + \tau)) \Big] \tag{38}$$

$$= \underbrace{\Bigg(\rho \circ \Big[\sum_{n=1}^{N} k_{\text{RBF}}(\cdot - x_n) \, , \, \sum_{q=1}^{Q} \sqrt{z_q} \, \phi_q(\cdot) + \sum_{n=1}^{N} v_n \, k(\cdot - x_n) \Big]}_{\Phi^{\mathcal{B}}(D)} \Bigg)(\cdot - \tau) = T_\tau^* \circ (\Phi^{\mathcal{B}}(D)). \tag{39}$$

The first equality in Eq. (37) holds in distribution sense because (1) Eqs. (30), (34) and (35) hold, and (2) ρ is a continuous operator and its push forward measure is defined as $\mu(\rho^{-1}(E))$ for a measurable set E in image space of ρ and Gaussian measure $\mu(\cdot)$ for $\sum_{q=1}^{Q} \sqrt{z_q^\tau} \, \phi_q^\tau(\cdot) + \sum_{n=1}^{N} v_n^\tau \, k(\cdot - (x_n + \tau))$ and $\sum_{q=1}^{Q} \sqrt{z_q} \, \phi_q(\cdot) + \sum_{n=1}^{N} v_n \, k(\cdot - (x_n + \tau))$ on finite grid points $\{t_m\}_{m=1}^M$. Therefore, the mapped random representations hold in distribution sense (each mapped representation has the same measure on the image space of ρ):

$$\rho \circ \left(\sum_{q=1}^{Q} \sqrt{z_q^\tau} \, \phi_q^\tau(\cdot) + \sum_{n=1}^{N} v_n^\tau \, k(\cdot - (x_n + \tau)) \right) \overset{d}{=} \rho \circ \left(\sum_{q=1}^{Q} \sqrt{z_q} \, \phi_q(\cdot) + \sum_{n=1}^{N} v_n \, k(\cdot - (x_n + \tau)) \right) \tag{40}$$

The equality in Eq. (38) holds because each q-th stationary random prior function $\phi_q(\cdot)$ evaluated on finite grid points $G := \{t_m\}_{m=1}^M$ follows the Gaussian distribution as

$$\phi_q(\cdot) \sim N(0; \Phi_q(G)\Phi_q(G)^T), \quad \Phi_q(G)\Phi_q(G)^T \approx K_q(G, G) \tag{41}$$

where $\Phi_q(G) = [\phi_q(t_1), .., \phi_q(t_M)] \in R^{M \times l}$, with the number of the sampled spectral points l, denotes the random feature matrix obtained by applying the random Fourier feature to q-th stationary kernel k_q. $\Phi_q(G)\Phi_q(G)^T$ approximates kernel Gram matrix $K_q(G, G) \in R^{M \times M}$. The τ-translated function $\phi_q(\cdot - \tau)$ follows the same distribution as well:

$$\phi_q(\cdot - \tau) \sim N(0; \Phi_q(G)\Phi_q(G)^T). \tag{42}$$

This is because $\phi_q(\cdot)$ is an instance of the stationary process evaluated on finite points G, and its translated function also follows the same stationary process.

Therefore, we prove the representation of Bayesian ConvDeepsets holds the TE in distribution sense as stated in Proposition 2.

□

A.3.2 Computational Efficiency for Eq. (13)

Construing the functional representation by Eq. (13) is computationally efficient compared to the conventional GP posterior sampling method. This is because for given M grid points $\mathcal{I} = \{t_m\}_{m=1}^{M}$ and N^c context data points $D^c = \{(x_n^c, y_n^c)\}_{n=1}^{N^c}$, the conventional sampling method first builds the predictive distribution of $f(\mathcal{I}) = [f(t_1), .., f(t_M)]$ on the discretized inputs $\mathcal{I}$ as follows:

$$\begin{aligned} p(f(\mathcal{I}); X^c, Y^c) &= N\left(f(\mathcal{I}); m(\mathcal{I}), \Sigma(\mathcal{I})\right) \\ m(\mathcal{I}) &= K(\mathcal{I}, X^c)(K(X^c, X^c) + \sigma_\epsilon^2 I)^{-1} Y^c \\ \Sigma(\mathcal{I}) &= K(\mathcal{I}, \mathcal{I}) - K(\mathcal{I}, X^c)(K(X^c, X^c) + \sigma_\epsilon^2 I)^{-1} K(X^c, \mathcal{I}) \end{aligned}$$

Then, it decomposes the predictive covariance matrix $\Sigma(\mathcal{I}) \in R^{M\times M}$ as $\Sigma(\mathcal{I}) = L(\mathcal{I})L(\mathcal{I})^T$ with the lower triangular matrix $L(\mathcal{I}) \in R^{M\times M}$ and then builds the sample function $f(\mathcal{I})$ as follows:

$$f(\mathcal{I}) = m(\mathcal{I}) + L(\mathcal{I})\mathcal{E} \tag{43}$$

where $\mathcal{E} = [\epsilon_1, .., \epsilon_M] \in R^M$ with $\epsilon_m \sim N(0, I)$. Since this sampling method requires two Cholesky decompositions related to $(K(X^c, X^c) + \sigma_\epsilon^2 I)^{-1}$ and $\Sigma(\mathcal{I})$, the computational cost is $\mathcal{O}((N^c)^3 + M^3)$.

On the other hand, the sampling method for the proposed random representation builds the sample function as follows:

$$f(D^c)(\cdot) = \underbrace{\sum_{q=1}^{Q} \sqrt{z_q}\, \phi_q(\cdot)}_{\text{prior term}} + \underbrace{\sum_{n=1}^{N^c} v_n\, k(\cdot - x_n^c)}_{\text{update term}} \quad \text{for} \cdot \in \mathcal{I}, \tag{44}$$

which is described in Eqs. (14) to (17). Since the proposed sampling method employs Cholesky decomposition once $\{v_n\}_{n=1}^{N^c}$ and takes computational cost $\mathcal{O}(M(Ql))$ for the prior term (l random Fourier features and Q mixtures), the computational cost is $\mathcal{O}((N^c)^3 + M(Ql))$.

As considering $Ql, N^c \ll M$ in general setting, the proposed functional representation is more computationally efficient.

A.3.3 Proof of Training objective in Eq. (23)

Let $\Theta = \{\theta_{\text{kernels}}, \theta_{\text{p}_{\text{nn}}}, \theta_{\rho}, \theta_{\text{pred}}\}$ be the learnable parameters for kernels, the translate invariant network, the CNN, and the network for prediction. To train the model parameters, we optimize following objective w.r.t Θ :

$$\mathbb{E}_{D^c, D^t \sim p(\mathcal{T})} \left[\mathcal{L}_{\text{ll}}(\Theta; D^c, D^t) - \beta \mathcal{L}_{\text{reg}}(\Theta; D^c, D^t) \right] \tag{45}$$

where $\mathcal{L}_{\text{ll}}(\Theta; D^c, D^t)$ denotes the log likelihood, and $\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)$ denotes the regularizer that induces the output of network $\text{p}_{\text{nn}}(D^c)$ to assign the reasonable stationary prior depending on context set D^c. In the following, we explain how the log likelihood $\mathcal{L}_{\text{ll}}(\Theta; D^c, D^t)$ and the regulaizer $\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)$ are derived.

Likelihood. For each task of the context set $D^c = \{(x_n^c, y_n^c)\}_{n=1}^{N^c}$ and target set $D^t = \{(x_n^t, y_n^t)\}_{n=1}^{N^t}$ with $D^c, D^t \sim p(\mathcal{T})$, the multi-sampled log likelihood estimator $\mathcal{L}_{\text{ll}}(\Theta; D^c, D^t)$ is derived as

$$\begin{aligned}
\log p(Y^t | X^t, D^c) &= \log \int p(Y^t | X^t, \Phi^{\mathcal{B}}) p(\Phi^{\mathcal{B}} | Z_{\text{Cat}}, D^c)\, p(Z_{\text{Cat}} | D^c)\, d\Phi^{\mathcal{B}} dZ_{\text{Cat}} &(46)\\
&\approx \log \left(\frac{1}{N} \sum_{n=1}^{N} p(Y^t \mid X^t, \Phi^{\mathcal{B}}_{(n)}) \right), \quad Z_{(n)} \sim q(Z_{\text{Cat}} | \text{p}_{\text{nn}}(D^c)),\ \Phi^{\mathcal{B}}_{(n)} \sim p(\Phi^{\text{B}}(D^c) \mid Z_{\text{Cat}} = Z_{(n)}) \\
&= \log \left(\frac{1}{N} \sum_{n=1}^{N} \left(\prod_{i=1}^{N^t} p\left(y_i^t | x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right) \right) \right) &(47)\\
&= \log \left(\frac{1}{N} \sum_{n=1}^{N} \exp \left(\sum_{i=1}^{N^t} \log p\left(y_i^t | x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right) \right) \right) := \mathcal{L}_{\text{ll}}(\Theta; D^c, D^t) &(48)
\end{aligned}$$

where $Z_{(n)}$ denotes the n-th random sample of $q(Z_{\text{Cat}} | D^c) = \text{Cat}(Z_{\text{Cat}}; \text{p}_{\text{nn}}(D^c))$, and $\Phi^{\mathcal{B}}_{(n)}$ denote the n-th sampled representation of Bayesian ConvDeepsets on finite grid $\{t_m\}_{m=1}^{M}$.

In Eq. (47), the conditional independence assumption for given $\Phi^{\mathcal{B}}_{(n)}$ is used for the n-th likelihood, i.e., $p(Y^t | X^t, \Phi^{\mathcal{B}}_{(n)}) = \prod_{i=1}^{N^t} p\left(y_i^t | x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right)$, and the likelihood of each observation $p\left(y_i^t | x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right)$ is modeled as Gaussian distribution, i.e.,

$$p\left(y_i^t \mid x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right) = N\left(y_i^t; \mu_{\text{nn}}\left(x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right), \sigma^2_{\text{nn}}\left(x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right) \right), \tag{49}$$

where $\mu_{\text{nn}}\left(x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right)$ and $\sigma_{\text{nn}}\left(x_i^t, \tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t) \right)$ denote the predictive mean and standard deviation on x_i^t, which are parameterized by neural network. In this procedure, we use the smoothed representation $\tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t)$ on target input x_i^t, as described in Eq. (6), and forward $\tilde{\Phi}^{\mathcal{B}}_{(n)}(x_i^t)$ by MLP layers.

Regularizer. We consider the regulaizer $\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)$ to allow the output of $\text{p}_{\text{nn}}(D^c)$ to assign the reasonable stationary prior for given task. To this end, we employ the result of variational inference for the regulaizer; the optimal distribution of Z_{Cat} is proportional to the posterior distribution of Z_{Cat}. That is, let $\tilde{q}\left(Z_{\text{Cat}}\right)$ be a variational distribution in a class of categorical distribution. Then, the lower bound of log marginal likelihood is represented as

$$\log p(Y^t | X^t, D^c) \geq \int \log \left(\frac{p(Y^t, Z_{\text{Cat}} | X^t, D^c)}{\tilde{q}\left(Z_{\text{Cat}}\right)} \right) \tilde{q}\left(Z_{\text{Cat}}\right) dZ_{\text{Cat}}. \tag{50}$$

The equality holds when the variational distribution $\tilde{q}\left(Z_{\text{Cat}}\right)$ is a posterior distribution of Z_{Cat} within a class of categorical distribution, i.e., $\tilde{q}\left(Z_{\text{Cat}}\right) = p(Z_{\text{Cat}} | D^c, D^t) = \text{Cat}(Z_{\text{cat}}; \text{p}_{\text{posterior}})$. Using the fact that the posterior distribution is proportional as follows:

$$
\begin{aligned}
p\Big(Z_{\text{cat}} = \underbrace{(0,..,1,...0)}_{\text{q-th indicator}} \mid D^c, D^t\Big) &= \frac{p(D^t, Z_{\text{cat}} = (0,..,1,...0) \mid D^c)}{p(D^t \mid D^c)} && (51)\\
&\propto p(D^t, Z_{\text{cat}} = (0,..,1,...0) \mid D^c) && (52)\\
&= p(D^t | Z_{\text{cat}} = (0,..,1,...0), D^c)\, p(Z_{\text{cat}} = (0,..,1,...0) \mid D^c) && (53)\\
&\propto p(Y^t | X^t, Z_{\text{cat}} = (0,..,1,...0), D^c)\, p(Y^c | X^c, Z_{\text{cat}} = (0,..,1,...0)) \\
&\propto p(Y^t | X^t, Z_{\text{cat}} = (0,..,1,...0), D^c), && (54)
\end{aligned}
$$

we set the parameter of posterior distribution $\mathrm{p}_{\text{posterior}}$ such that q-th element of $\mathrm{p}_{\text{posterior}}$ is represented as

$$
(\mathrm{p}_{\text{posterior}})_{(q)} = \frac{\exp\left(\log p(Y^t | X^t, D^c, k_q) \;/\; \tau_0\right)}{\sum_{q=1}^{Q} \exp\left(\log p(Y^t | X^t, D^c, k_q) \;/\; \tau_0\right)} \tag{55}
$$

where τ_0 denotes the temperature hyperparameter. For $\log p\left(Y^t | X^t, D^c, k_q\right)$, we compute the likelihood via empirical posterior distribution defined as:

$$
p\left(Y^t | X^t, D^c, k_q\right) = N\left(Y^t; \hat{\mu}_q(X^t)\ ,\ \mathrm{Diag}\left(\hat{\Sigma}_q(X^t)\right)\right), \tag{56}
$$

which can be computed efficiently. The empirical predictive mean $\hat{\mu}_q(X^t) \in R^{N_t}$ and diagonal covariance $\mathrm{Diag}(\hat{\Sigma}_q(X^t)) \in R^{N_t}$ are computed by using the sample functions of GP posterior obtained by Eq. (13).

As a result, we consider the regulaizer $\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)$ as the KL divergence between categorical distribution:

$$
\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t) := \mathrm{KL}\left(q(Z_{\text{cat}} \mid D^c) \;||\; p(Z_{\text{cat}} \mid D^c, D^t)\right). \tag{57}
$$

Comparison the proposed training objective with the conventional ELBO. As we consider the conventional ELBO estimator $\mathcal{L}_N$ of meta-learning framework, represented as,

$$
\begin{aligned}
&\mathbb{E}_{D^c, D^t \sim p(\mathcal{T})}\left[\int \log p\left(Y^t | X^t, D^c, \Phi^{\mathcal{B}}, Z_{\text{Cat}}\right) q(\Phi^{\mathcal{B}}, Z_{\text{Cat}})\, d\Phi^{\mathcal{B}}\, dZ_{\text{Cat}} - \mathrm{KL}(q(\Phi^{\mathcal{B}}, Z_{\text{Cat}}) || p(\Phi^{\mathcal{B}}, Z_{\text{Cat}}))\right] \\
&\approx \mathbb{E}_{D^c, D^t \sim p(\mathcal{T})}\Bigg[\underbrace{\frac{1}{N}\sum_{n=1}^{N} \log p\left(Y^t | X^t, D^c, \Phi^{\mathcal{B}}_{(n)}\right)}_{\text{likelihood}} - \underbrace{\Big(\mathrm{KL}(q(\Phi^{\mathcal{B}}) || p(\Phi^{\mathcal{B}})) + \mathrm{KL}(q(Z_{\text{Cat}}) || p(Z_{\text{Cat}}))\Big)}_{\text{regulaizer}}\Bigg] := \mathcal{L}_N \qquad (58)
\end{aligned}
$$

the proposed objective employs the multi-sampled log likelihood estimator, used in [Burda et al., 2015, Foong et al., 2020], which is known to be tighter than the likelihood term of $\mathcal{L}_N$. In addition, the proposed objective focuses on regularizing Z_{Cat}, and use the posterior distribution $p(Z_{\text{cat}} | D^t, D^c)$ in Eq. (57) for training $q(Z_{\text{Cat}})$ instead of the prior distribution $p(Z_{\text{cat}})$ as shown in Eq. (58).

B Further Details for 1d Regression Task

B.1 Details for Datasets

To train the NP models, we employ the meta-learning framework; for every task, when the given finite observations, referred as context sets and target sets are assumed to be finite samples of the function sampled from stochastic process, the NP models first construct the functional representation of the context sets and then generate the predictive distribution on the target sets.

We consider 4 stationary stochastic processes: RBF, Matern-$\frac{5}{2}$, Weakly periodic, and Sawtooth, which are used in [Gordon et al., 2019], to prepare the context and target sets. We describe the each process as follows:

- **RBF (large lengthscale)**: The context and target sets are constructed by randomly choosing N_c data points and N_t data points from the function sampled from Gaussian Process (GP) with the following kernel function k with the random lengthscale $l \sim \mathcal{U}([1.1, 2.1])$, that is randomly sampled per each task:

$$k(x, x') = e^{-\frac{1}{2}(\frac{x-x'}{l})^2}.$$

- **Matern-$\frac{5}{2}$ (small lengthscale)**: The context and target sets are constructed by randomly N_c data points and N_t data points from the function sampled from GP with the following kernel function k with $d = |\frac{x-x'}{l}|$ and random lengthscale $l \sim \mathcal{U}([0.19, 0.21])$, that is randomly sampled per each task:

$$k(x, x') = (1 + \sqrt{5}d + \frac{5}{3}d^2)e^{-\sqrt{5}d}.$$

- **Weakly periodic**: The context and target sets are constructed by randomly choosing N_c data points and N_t data points from the function sampled from GP with the following kernel function k with $g_1(x) = \cos(2\pi f x)$, $g_2(x) = \sin(2\pi f x)$, and random frequency parameter $f \sim \mathcal{U}([2.0, 3.0])$, that is randomly sampled per each task:

$$k(x, x') = e^{\left(-\frac{1}{2}(g_1(x)-g_1(x'))^2 - \frac{1}{2}(g_2(x)-g_2(x'))^2 - \frac{1}{32}(x-x')^2\right)}.$$

- **Sawtooth**: The context and target sets are constructed by randomly choosing N_c data points and N_t data points from the sampled function $y_{\text{sawtooth}}(t)$ represented as

$$y_{\text{sawtooth}}(t) = \frac{A}{2} - \frac{A}{\pi}\sum_{k=1}^{\infty}(-1)^k \frac{\sin(2\pi k f(t+\tau))}{k},$$

  where A denotes the amplitude, f denotes the frequency, τ denotes the shift, and t denotes time. We use truncate the series at an integer K. We consider the random amplitude $A \sim \mathcal{U}([0.8, 1.2])$, random frequency $f \sim \mathcal{U}([1, 2])$, random truncation integer $K \sim \mathcal{U}([10, 20])$, and random shift $\tau \sim \mathcal{U}([-1, 1])$, which are randomly sampled for each task.

B.2 Details for Tasks of Training, Validation, and Test

In this experiment, we set the training range $[0, 4]$ and test range $[4, 8]$ (outside of training range). For training, we construct the context sets and target sets by sampling the data points on training range. Then, we evaluate the trained models with context sets and target sets, that are sampled on test range. These test sets are used to check whether trained NP models holds translation equivariance.

We consider the following number of data points for training, validation, and test:

- **Small number of context data points:** For training, we randomly sample $N^c \sim \mathcal{U}([5, 25])$ as the number of context data points, and randomly sample $N^t \sim \mathcal{U}([N^c, 50])$ as the number of target data points for each task. We use $500 \times 50 \times 16$ tasks for training through 500 batches. For validation, we set N^c context data points and N^t target data points as done in training, and use 128×16 tasks for validation to choose the parameters of the trained models. For test, we also set N^c context data points and N^t target data points for each processes (RBF, Matern-$\frac{5}{2}$, Weakly periodic, and Sawtooth), as done in training, and use 128×16 tasks per the process to evaluate the trained models using the chosen parameters in validation.

- **Large number of context data points:** For training, we randomly sample $N^c \sim \mathcal{U}([10, 50])$ as the number of context data points, and randomly sample $N^t \sim \mathcal{U}([N^c, 50])$ as the number of target data points for each task. For validation and test, we use 128×16 tasks. For test set, we use 128×16 tasks per the process, as done in the case of **the small number of context data points**.

Additionally, we consider the same number of tasks per each stochastic process so that the NP models does not fit particular stochastic process only during training

B.3 Details for Hyperparameters.

Hyperparameters of Kernels. For the hyperparameter of RBF kernels used for ConvCNP and ConvLNP, we conduct the experiment with $l \in \{0, 01, 0.1, 0.5\}$, and set the lengthscale $l = 0.01$ for both models obtaining the best performance out of those candidates.

For the hyperparameter of GPConvCNP, we conduct the experiment with $l \in \{0.1, 0.5, 1.0\}$, and set the lengthscale $l = 1.0$ obtaining the best performance out of those candidates.

For the hyperparameter of the proposed method, we use $Q \in \{3, 4, 5\}$ basis stationary kernels, and set $\mathrm{HZ}_{\max} = Q$. Then, we space the frequency range $[0, \mathrm{HZ}_{\max}]$ linearly, set each centered value as μ_q with $\mu_1 = 0 \leq .. \leq \mu_Q$, and set $\sigma_q = 0.5(\mu_2 - \mu_1)$ for $q = 1, .., Q$. For the noise parameter σ_ϵ^2, we set $\sigma_\epsilon = 1e$-2.

For the number of spectral points, we use $l = 10$ in Eq. (14).

For the number of sample function, we use $N = 5$ in Eq. (19).

For training, we use ADAM optimizer [Kingma and Ba, 2014] with learning rate $5e$-4 and weight decay $1e$-4.

For the regularizer hyperparameter β in Eq. (21), we set $\beta = 0.1$ for the proposed method.

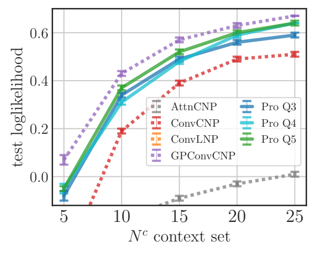
B.3.1 Baseline Implementations

For ANP, we employ the deterministic path of the model described by [Kim et al., 2018] for 1d regression experiment. For ConCNP, ConvLNP, and GPConvCNP, we employ the structure of Deep CNN 1d as described in Appendix A.2.2. For ANP and ConvCNP, we employ the implementation [1]. For ConvLNP, we refer to the implementation [2].

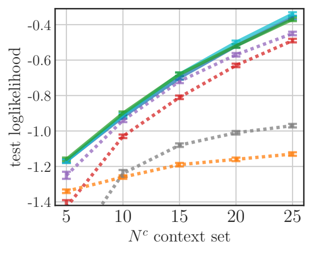
[1] https://github.com/cambridge-mlg/convcnp
[2] https://github.com/YannDubs/Neural-Process-Family

B.4 Additional Results

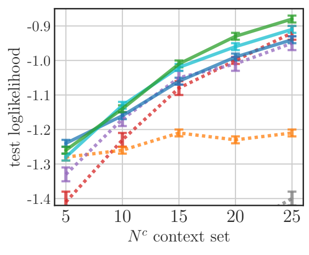
Training with a small number of context sets. For the completeness of the experiment results, we report additional results of the training with a small number of context set ($N^c \sim \mathcal{U}([5, 25])$). Figs. 9a to 9d describes the mean and one-standard error of the log likelihood for 1024 tasks (within training range). For evaluation, we set varying number of context data points $N^c \in \{5, 10, 15, 20, 25\}$ and target points $N^t = 50$. Figs. 9e to 9h corresponds to the results for tasks (beyond training range). We could not report the results of AttnCNP due to poor generalization performance.



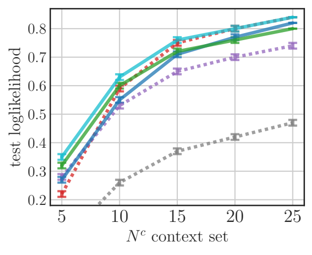
(a) RBF (in-range)



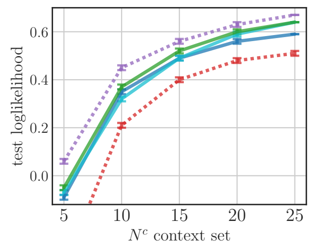
(b) Matern-$\frac{5}{2}$ (in-range)



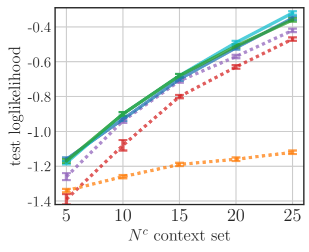
(c) Weakly periodic (in-range)



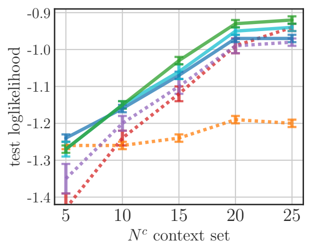
(d) Sawtooth (in-range)



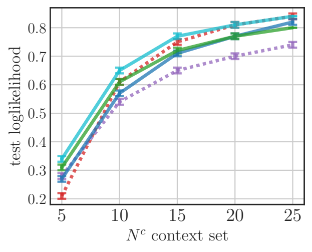
(e) RBF (out-range)



(f) Matern-$\frac{5}{2}$ (out-range)



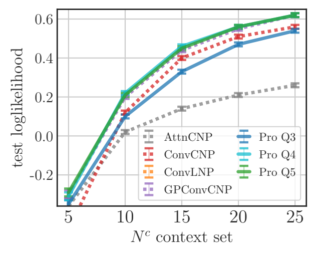
(g) Weakly periodic (out-range)



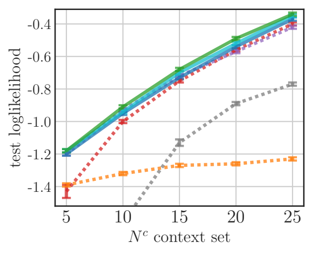
(h) Sawtooth (out-range)

Figure 9: Prediction result of each processes from training setting of small N^c context points.

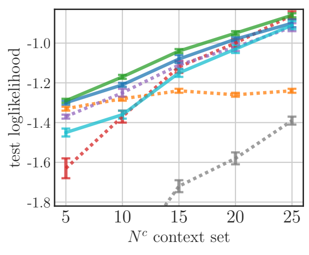
Training with a large number of context sets. Additionally, we report the results of the training with a large number of context set ($N^c \sim \mathcal{U}([10, 50])$), and apply the same evaluation procedure with setting of a small number of context set. Figs. 10a to 10d describes the mean and one-standard error of the log likelihood for 1024 tasks (within training range). Figs. 10e to 10h corresponds to the results for tasks (beyond training range).



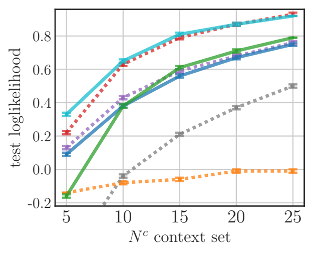
(a) RBF (in-range)



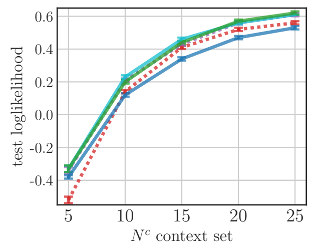
(b) Matern-$\frac{5}{2}$ (in-range)



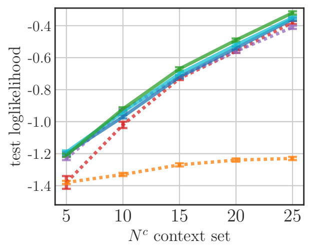
(c) Weakly periodic (in-range)



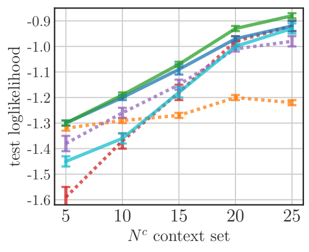
(d) Sawtooth (in-range)



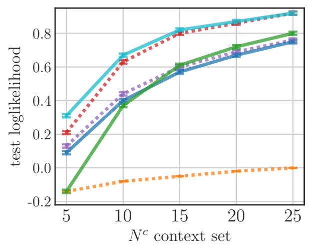
(e) RBF (out-range)



(f) Matern-$\frac{5}{2}$ (out-range)



(g) Weakly periodic (out-range)

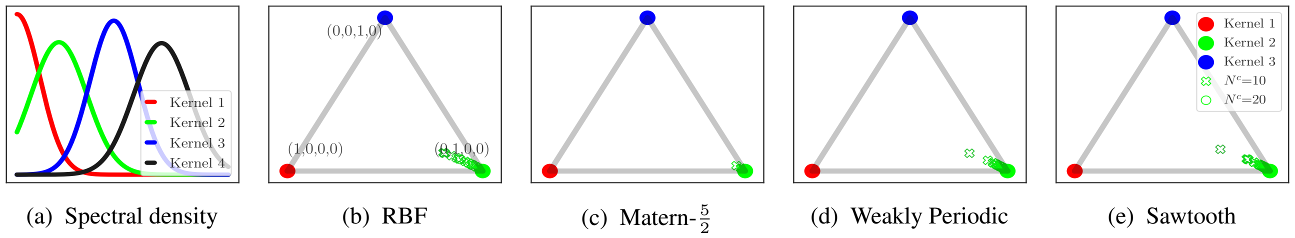


(h) Sawtooth (out-range)

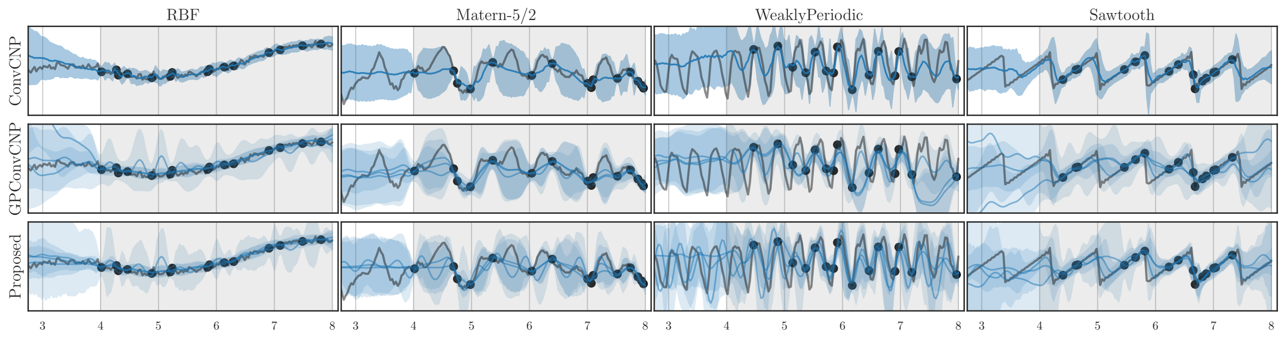
Figure 10: Prediction result of each processes from training setting of large N^c context points .

Conclusion. The proposed NP model ($Q = 3, 4, 5$) using a Bayesian ConvDeepsets has shown superior prediction performances especially when training with a small number of context data points. When a more number of context points is allowed for training, the performance gap between ConvCNP and proposed method decreases as shown in Figs. 9e and 10e and Figs. 9f and 10f.

Prediction results for training a large number of context sets. Also, we report prediction results in Fig. 11 and quantitative analysis of the trained parameters when the NP models are trained with the setting of a large number of context sets. We see that $\mathrm{p}_{\mathrm{nn}}(D^c)$ allocates the most of stationary priors as second spectral density (green) that overlaps most frequency region with the RBF prior (red). This results are quite different to the result of the setting of a small number of context sets (main). We believe that this implies why the RBF kernel smoother of ConvDeepsets could be effective when using a sufficient number of context set, and why the task-dependent stationary prior could be effective when using a small number of context set.



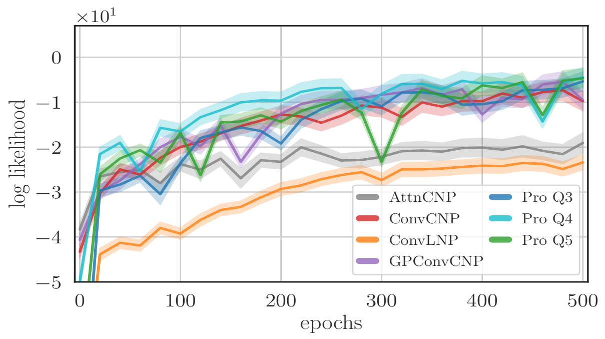
(a) Spectral density (b) RBF (c) Matern-$\frac{5}{2}$ (d) Weakly Periodic (e) Sawtooth



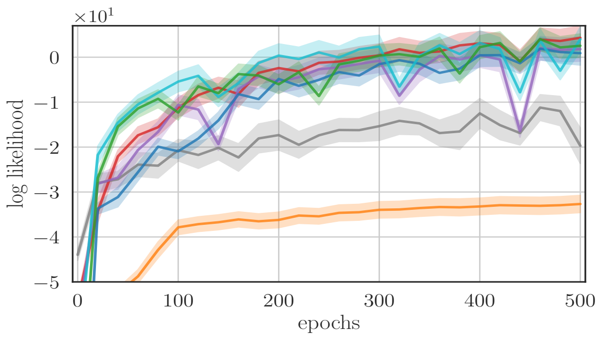
(f) Predictive distribution of each baseline on each process ($N^c = 15$)

Figure 11: Stochastic processes modeling beyond training range (gray region $[4, 8]$): Fig. 11a denotes the trained spectral density with $Q = 4$ kernels. Figs. 11b to 11e denote the output of $\mathrm{p}_{\mathrm{nn}}(D^c)$ for 128 context set $\{D_n^c\}_{n=1}^{128}$ of each process. Fig. 11f shows the predictive distributions of the baselines on each process.

Validation error during training phase. We report the averaged log likelihood evaluated on validation set during training; the validation set consists of 128×16 tasks that have the equal number of task for each process. Fig. 12a and Fig. 12b shows the validation metric over training epochs when using a small number of context set and large number of context set, respectively. In each epoch, we use 500×16 tasks for training. These figures imply that when the stationary priors are imposed well like $Q = 4$, the propose model can achieve good performance while using the less number of training tasks.



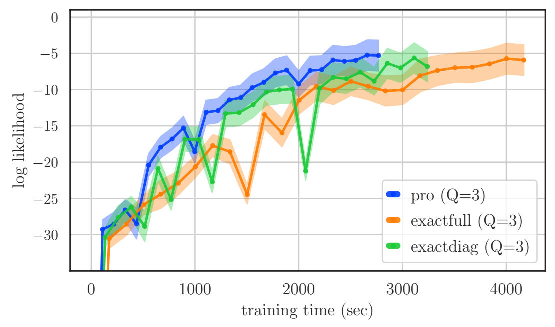
(a) Small context set $N^c \sim U(5, 25)$



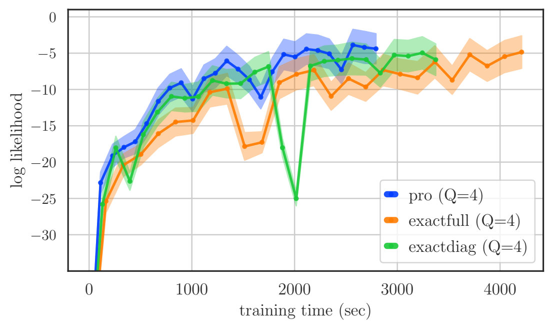
(b) Large context set $N^c \sim U(10, 50)$

Ablation study for computational efficiency. We investigate the computational efficiency of the proposed representation in the same experiment setting of Section 4.1 ($l = 10$ and $M = 384$). We compare the progress of the training time and the corresponding log likelihood when using the conventional GP posterior sampling method and path-wise sampling method to construct the random data representation of Eq. (13), respectively.

Figs. 13a and 13b compare the training time (x-axis) and the progress of the log likelihood (y-axis) on the validation set during the training phase when using $Q = 3$ and $Q = 4$, respectively; exactfull and exactdiag denote the sampling procedure that uses the predictive distribution and diagonal predictive distribution for the conventional GP posterior sampling, respectively. This indicates that the proposed representation using the path-wise GP posterior sampling reduces training time as explained in Appendix A.3.2 while maintaining the performance of the log likelihood.



(a) $Q = 3$



(b) $Q = 4$

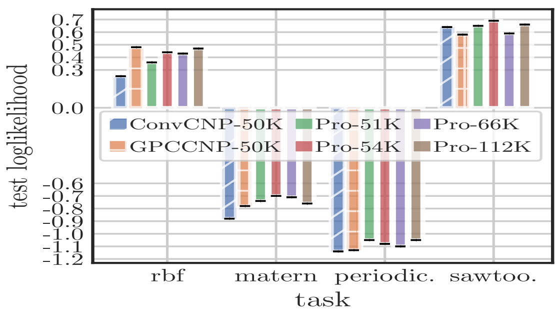
Figure 13: Computational efficiency: training time (seconds) VS progress of log likelihood on validation set.

Ablation study for the size of amortized neural network. We investigate how the size of amortized neural network $p_{\text{nn}}(D^c)$ affects the prediction performance in the same experiment setting of Section 4.1. For $p_{\text{nn}}(D^c)$ structure, we use 3-layer structure of the **Version 2** described in Eq. (26). We consider the different size of the convolution channels, which results in the different number of $p_{\text{nn}}(\cdot)$ parameters.

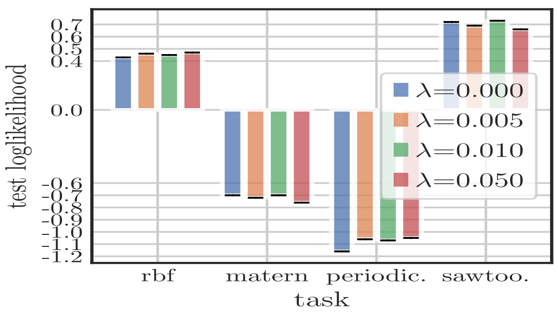
Fig. 14a compares the number of model parameters (K=10^3) and the averaged test likelihood (w.r.t context points N^c) for ConvCNP, GPConvCNP, and proposed methods (Q=4) using the different size of channels. This figure indicates using a few more parameters exploits merit of task-dependent prior.

Ablation study for regularization. We investigate how the hyperparameter β in Eq. (23) affects the trained NP model in the same experiment setting of Section 4.1. For the case of $Q = 4$, we consider various cases by setting different regularizer coefficient λ; $\lambda = 0.0$ means the regularizer is not used for training, and $\lambda = 0.1$ denotes $\beta = 0.1 \left| \frac{\mathcal{L}_{\text{ll}}(\Theta; D^c, D^t)}{\mathcal{L}_{\text{reg}}(\Theta; D^c, D^t)} \right|$ in Eq. (23), is used every iteration.

Fig. 14b shows the averaged test likelihood with respect to the N^c number of context data points. This indicates that the training without the regularizer ($\lambda = 0$) results in poor performance on the weakly-periodic task, whereas the training with regularizer ($\lambda \in \{.005, .01\}$) leads to good performance on all tasks.



(a) Effect of the size of $p_{\text{nn}}(D^c)$



(b) Effect of the regularizer coefficient β

Figure 14: Ablation studies for the size of amortized neural network and the regularizer coefficient λ

C Further Details for Multi-Channel Regression Task

C.1 Details for Datasets

In this section, we consider two types of the multi-channel stationary process: (1) sinusoidal process and (2) GP using the Multi-output SM kernel (MOSM) [Parra and Tobar, 2017], that constructs the cross covariance function of two different processes based on Crammer theorem, i.e., a multi-channel extension of Bochner's theorem. Additionally, we consider two types of tasks for each process: (1) a simple task and a difficult task having more diversity to study how the task-dependent prior on functional representation affects the performance of the corresponding NP model depending on task diversity. We hypothesis that when the small number of data points is given for the task having more diversity, assigning a reasonable prior on functional representation could be helpful to solve the targeted tasks. Each process is described as follows:

- **Sinusoidal-phase** : i-th channel context and target sets are constructed by randomly choosing $N_c^{(i)}$ data points and $N_t^{(i)}$ data points from the sampled function $f_i(t)$, that is represented as

$$f_i(t) = A_i \sin\left(2\pi w_i (t - \phi_i)\right) + \epsilon, \qquad i = 1, 2, 3 \quad ,$$

  where A_i denotes i-th channel amplitude parameter, w_i denotes i-th channel frequency parameter, ϕ_i denotes i-th channel phase parameters, and ϵ is random noise satisfying $\epsilon \sim N(0, 0.1^2)$. For frequency parameters, we set $w_1 = 2.1, w_2 = 4.1$, and $w_3 = 6.1$. For the amplitude, we set $A_1 = 1 + a, A_2 = 2 + a$, and $A_3 = 3 + a$ by using a random amplitude $a \sim \mathcal{U}([-25, 0.25])$. For the phase parameter, we consider random phase $\tau_1 \sim \mathcal{U}([-1, 1]), \tau_2 \sim \mathcal{U}([-1.5, 0.5])$, and $\tau_3 \sim \mathcal{U}([-2, 0])$. The random parameters for amplitude a and phases $\{\tau_1, \tau_2, \tau_3\}$ are randomly sampled for each task.

- **Sinusoidal-all** : i-th channel context and target sets are constructed by randomly choosing $N_c^{(i)}$ data points and $N_t^{(i)}$ data points from the sampled function $f_i(t)$, that is represented as that is represented as

$$f_i(t) = A_i \sin\left(2\pi (w_i + \theta_i)(t - \phi_i)\right) + \epsilon \qquad i = 1, 2, 3,$$

  where θ_i denotes i-th channel random frequency parameter. In this generation procedure, we consider random frequency parameters additionally; we first sample $\theta \sim \mathcal{U}([0, 5])$, and set $\theta_1 = \theta, \theta_2 = 2\theta$, and $\theta_3 = 3\theta$. This is intended to let the task of this process have more diversity. For other parameters $\{A_i, w_i, \phi_i\}_{i=1}^{3}$, we set the same setting as described in **Sinusoidal**.

- **MOSM**: : i-th channel context and target sets are constructed by randomly choosing $N_c^{(i)}$ data points and $N_t^{(i)}$ data points from i-th channel function of the multi-output function sampled from Gaussian Process (GP) with the following kernel function $\{k_{ij}\}_{i,j=1}^{3}$ of which the cross kernel k_{ij} between i-th channel and j-th channel is represented as

$$k_{ij}(x, x') = \exp\left(-\frac{1}{2}(x - x' + \theta_{ij})^\top \Sigma_{ij} (x - x' + \theta_{ij})\right) \cos\left(2\pi (x - x' + \theta_{ij})^\top \mu_{ij} + \phi_{ij}\right),$$

  where $\mu_{ij} = (\Sigma_i + \Sigma_j)^{-1}(\Sigma_i \mu_j + \Sigma_j \mu_i)$, $\Sigma_{ij} = 2\Sigma_i (\Sigma_i + \Sigma_j)^{-1} \Sigma_j$, $\phi_{ij} = \phi_i - \phi_j$, and $\phi_{ij} = \phi_i - \phi_j$ when μ_i denote i-th channel mean parameter, Σ_i denotes i-th channel covariance parameter, θ_i denotes i-th channel delay parameter, and ϕ_i denotes i-th channel phase parameter, respectively. In this experiment, we set $\mu_1 = 0.1$, $\mu_2 = 3.0$, $\mu_3 = 5.0$ for mean parameters and $\Sigma_1 = 0.1$, $\Sigma_2 = 0.1$, $\Sigma_3 = 0.1$ for covariance parameters. We set $\theta_i = 1$ with $i = 1, 2, 3$ for delay parameters and $\phi_i = 0$ with $i = 1, 2, 3$ for phase parameters. We employ the following implementation [de Wolff et al., 2020] [3].

- **MOSM-varying**: i-th channel context and target sets are constructed by randomly choosing $N_c^{(i)}$ data points and $N_t^{(i)}$ data points from i-th channel function of the multi-output function as described in **MOSM** with different hyperparameter setting. We consider random mean parameters to generate the diverse tasks. We consider the random mean parameters; we first sample perturb noises $\{n_j\}_{j=1}^{3} \sim N(0, .5^2 I)$ every task, and then set $\mu_1 = 0.1 + n_1$, $\mu_2 = 3.0 + n_2$, $\mu_3 = 5.0 + n_3$, which are reinitialized every task. Since this trick generates the data points observed from a new multi-output stationary process every task by considering the different cross correlation function between different two processes, the generated tasks are more diverse.

[3] https://github.com/GAMES-UChile/mogptk

C.2 Details for Tasks of Training, Validation, and Test

In this experiment, we set the training range $[0, 3]$ and test range $[3, 6]$ (outside of training range). For training, we construct the context sets and target sets by sampling the data points on training range. Then, we evaluate the trained models with context sets and target sets, that are sampled on test range as described in Appendix B.2.

We consider the following number of data points for training, validation, and test:

- For training, we randomly sample $N^c \sim \mathcal{U}([5, 25])$ as the number of context data points, and randomly sample $N^t \sim \mathcal{U}([N_c, 50])$ as the number of target data points for each task. We use $500\times50\times16$ tasks for training through 500 batches. For validation, we set N^c context data points and N^t target data points as done in training, and use 64×16 tasks for validation to choose the parameters of the trained models. For test, we consider the varying $N^c \in \{5, 10, 15, 20, 25, 30\}$ context data points and $N^t = 50$ target data points per a task, and use 64×16 tasks per given N^c context points to study how the number of context data points affects the predictive performance of the trained models using the parameters obtained in validation procedure.

C.3 Details for Hyperparameters.

For the hyperparameter of RBF kernels used for ConvCNP and ConvLNP, we conduct the experiment with $l \in \{0.1, 0.5, 1.0\}$, and set the lengthscale $l = 0.1$ for both models obtaining the best performance out of those candidates.

For the hyperparameter of GPConvCNP, we set $l = 1.0$ for each channel respectively.

For the hyperparameter of the proposed method, we use 5 basis stationary kernels ($Q = 5$), and set $\mathrm{HZ}_{\max} = Q$. Then, we space the frequency range $[0, \mathrm{HZ}_{\max}]$ linearly, and set each centered value as μ_q with $\mu_1 = 0 \leq .. \leq \mu_5$, and set $\sigma_q = 0.5(\mu_2 - \mu_1)$ for $q = 1, .., 5$. For the noise parameter σ_ϵ^2, we set $\sigma_\epsilon = 1e\text{-}2$.

For the number of spectral points, we use $l = 10$ in Eq. (14).

For the number of sample function, we use $N = 10$ in Eq. (19).

For training, we use ADAM optimizer [Kingma and Ba, 2014] with learning rate $5e\text{-}4$ and weight decay $1e\text{-}4$.

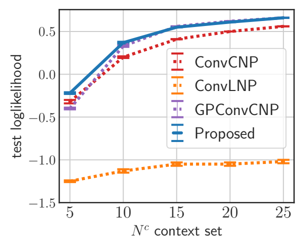
For the regularizer hyperparameter β in Eq. (21), we set $\beta = 0.1$ for the proposed method.

C.4 Additional Results

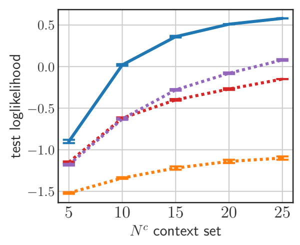
Results of Sinusoidal dataset. For completeness of results, we report additional prediction results of other baseline on theh tasks of Sinusoidal-all as shown in Figs. 15f and 15g.

Results of MOSM dataset. We report additional results on GP-MOSM dataset. Figs. 16a and 16b describes the mean and one-standard error of the log likelihood for 1024 tasks (beyond training range); Fig. 16a shows the result of the less diversity task (varying phase), and Fig. 16b corresponds to that of the high diversity (varying amplitude, frequency, and phase). These figures show that the proposed method could model processes well on both less and more diverse tasks.

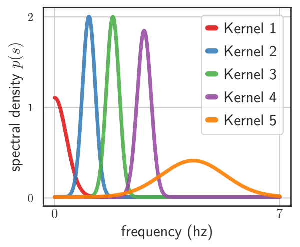
Fig. 16c shows the trained spectral density of $\{k_q\}_{q=1}^5$ on high diversity task. Figs. 16d and 16e denote the parameters $\mathrm{p_{nn}}(D^c)$ for two tasks (top and bottom), and THE corresponding predictions of the proposed method. Figs. 16f and 16g show the prediction results of ConvCNP and GPConvCNP, respectively.



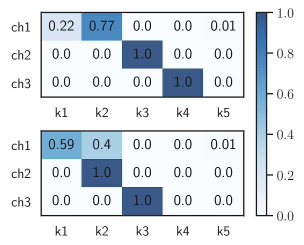
(a) Sinusoidal-phase



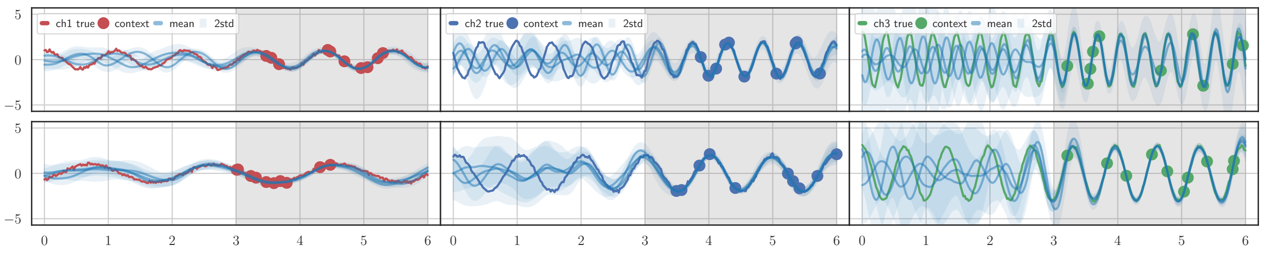
(b) Sinusoidal-all



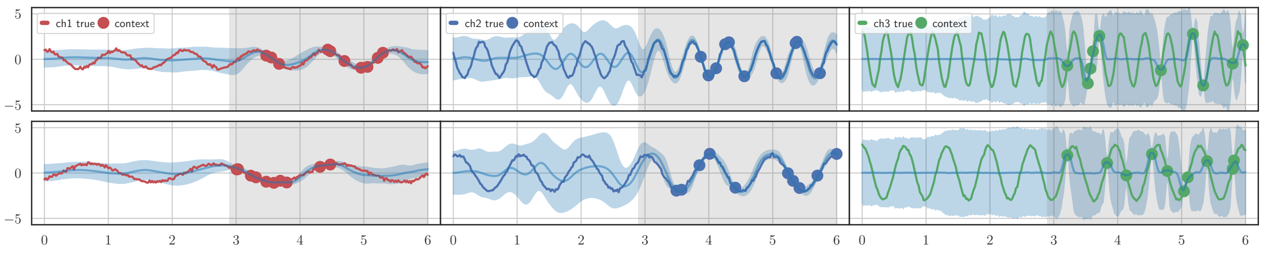
(c) Spectral density



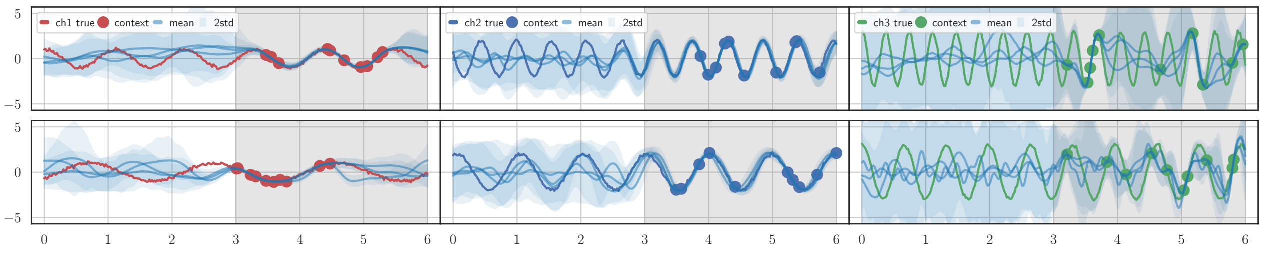
(d) Task-dependent prior



(e) Proposed: Prediction for 2 different tasks ($N^c = 10$) of the Sinusoidal-all

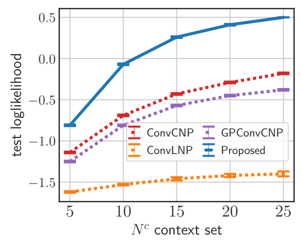


(f) ConvCNP: Prediction for 2 different tasks ($N^c = 10$) of the Sinusoidal-all

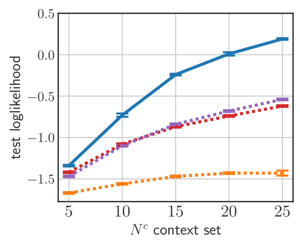


(g) GPConvCNP: Prediction for 2 different tasks ($N^c = 10$) of the Sinusoidal-all

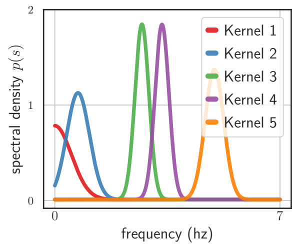
Figure 15: 3-channel sinusoidal processes modeling: Figs. 15a and 15b denotes test likelihood on sinusoidal processes having different task diversity. Fig. 15c denote the trained spectral density $\{p_q(s)\}_{q=1}^{5}$ for the Sinusoidal-all process, and Figs. 15d and 15e show the chosen prior and corresponding prediction for 2 tasks; as the context sets having small frequency characteristics are given (from first to second row in Fig. 15e), the stationary prior is imposed differently (from top to bottom in Fig. 15d). Figs. 15f and 15g show the prediction results using the same context set with Fig. 15e.



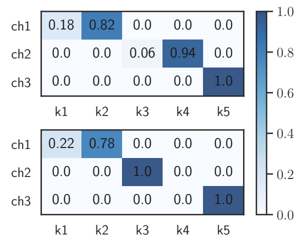
(a) MOSM



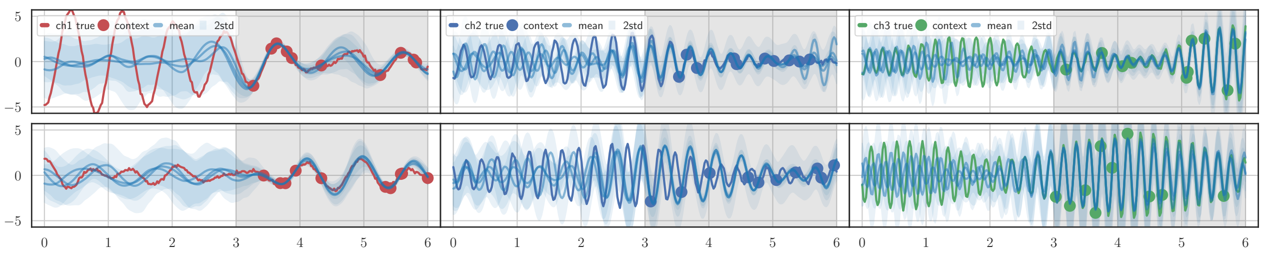
(b) MOSM-varying



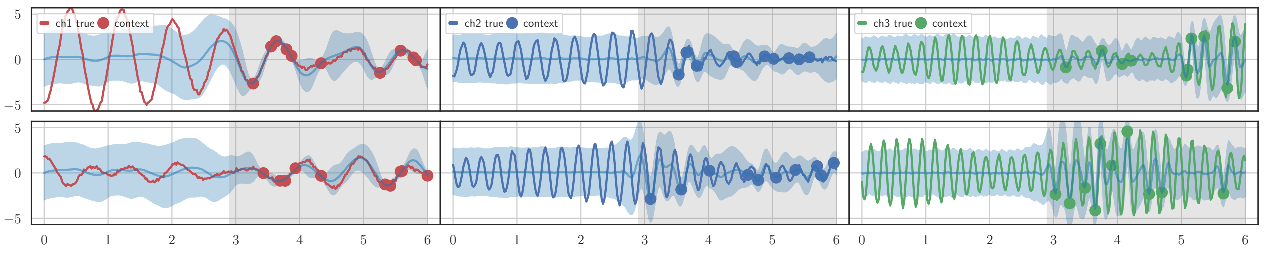
(c) Spectral density



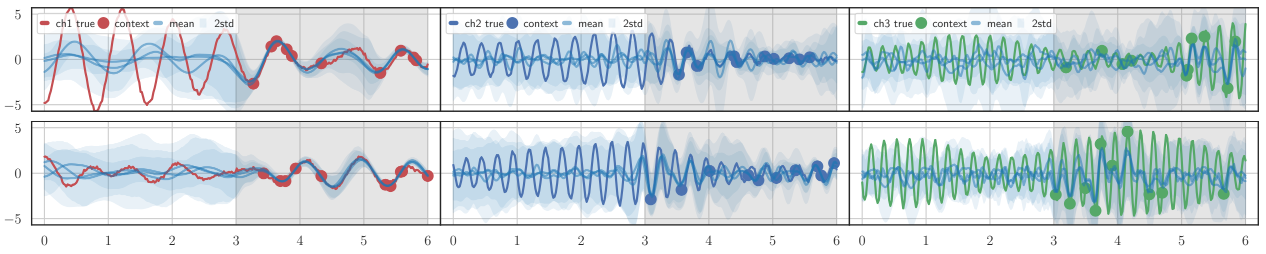
(d) Task-dependent prior



(e) Proposed: Prediction for 2 different tasks ($N^c = 10$) of the MOSM-varying



(f) ConvCNP: Prediction for 2 different tasks ($N^c = 10$) of the MOSM-varying



(g) GPConvCNP: Prediction for 2 different tasks ($N^c = 10$) of the MOSM-varying

Figure 16: 3-channel GP-MOSM processes modeling: Figs. 16a and 16b denotes test likelihood having different task diversity. Fig. 16c denote the trained spectral density for the MOSM-varying process, and Figs. 16d and 16e show the chosen prior and corresponding prediction for 2 tasks; check that the stationary prior is imposed differently (from top to bottom in Fig. 16d), when context are given differently (from first to second row in Fig. 16e). Figs. 16f and 16g show the prediction results using the same context set with Fig. 16e.

D Further Details for Image Completion Task

D.1 Details for Datasets

For image completion task, each task is defined to predict the randomly chosen pixel values of image when using given partial pixel values as context set. To do this, we use a subset of monthly land surface temperature set used in [Remes et al., 2017]. We use the surface temperature of North America (53×115×3) and Europe (78×102×3).

D.2 Details for Tasks of Training, Validation, and Test

We use the datasets (2018 - 2020) for the training, and datasets (2021) for the test.

For training, we randomly sample the context and target set by choosing a small context rate $p \in \{.01, .05, .10, .20\}$ randomly with probability $[4/10, 3/10, 2/10, 1/10]$. We use 2500×5 tasks for training, and 256×4 tasks as validation set.

For test, we set the varying context rate $p \in \{.01, .05, .10, .20\}$ as done in training phase, and set target set by choosing a target rate $p = .5$ We use each 256 tasks per context rate $p \in \{.01, .05, .10, .20\}$ to evaluate the trained models.

D.3 Details for Hyperparameters.

For stationary kernels, we set $\mu_1 = [0.0, 0.0]$ and $\sigma_1 = [0.5, 0.5]$ for $Q = 1$.

For $Q = 4$, we set $\mu_1 = [0.0, 0.0], \mu_2 = [2.0, 0.0], \mu_3 = [0.0, 2.0]$, and $\mu_4 = [2.0, 2.0]$ and $\sigma_q = [0.5, 0.5]$ for $q = 1, .., 4$.

For the number of spectral points, we use $l = 10$ in Eq. (14).

For the number of sample function, we use $N = 4$ in Eq. (19).

For the prior hyperparameter of approximate scheme α, we use $\alpha \in \{.05, 0.1\}$.

For training, we use ADAM optimizer [Kingma and Ba, 2014] with learning rate $5e$-4 and weight decay $1e$-4.

For the regularizer hyperparameter β in Eq. (21), we set $\beta = 0.1$ for the proposed method.

D.4 Additional Results

We report additional prediction results for image completion tasks of North America dataset in Fig. 17.

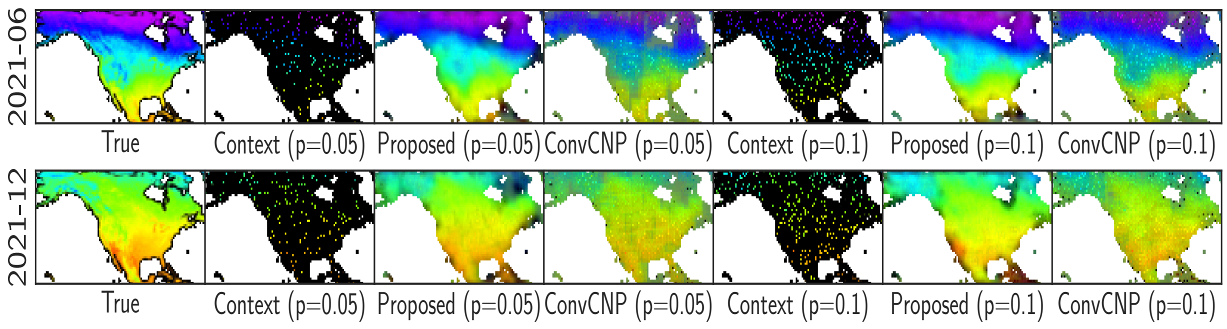
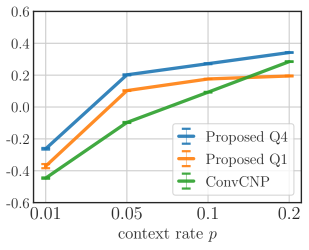


Figure 17: Prediction results for North America temperatures (June 2021 and December 2021) that are out of training sets.

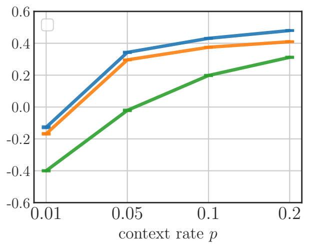
Additionally, we report the progress of the test log likelihood during training phase in Figs. 18 and 19. These figures compare the prediction improvement of the models depending on the number of used training tasks.

Figs. 18a to 18d shows the test log likelihood on the dataset of North America (2018-2020) over varying training tasks $\{2500, 5000, 7500, 10000\}$. Figs. 18e to 18h shows the corresponding results on the dataset of North America (2021).

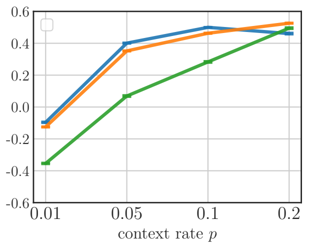
With the same protocol of North America experiment results, Figs. 19a to 19d shows the corresponding results on Europe temperature (2018-2020), and Figs. 19e to 19h shows the corresponding results on Europe temperature (2018-2020),



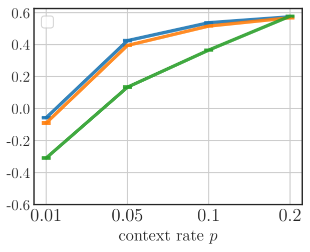
(a) 2500 tasks (in-train)



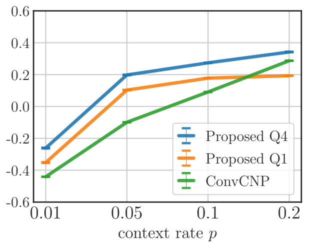
(b) 5000 tasks (in-train)



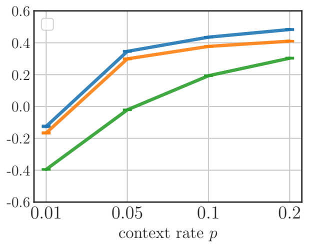
(c) 7500 tasks (in-train)



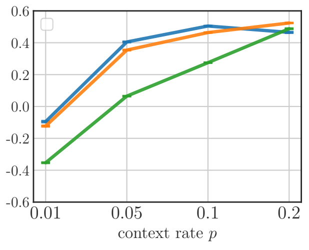
(d) 10000 tasks (in-train)



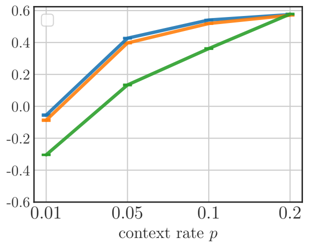
(e) 2500 tasks (out-train)



(f) 5000 tasks (out-train)

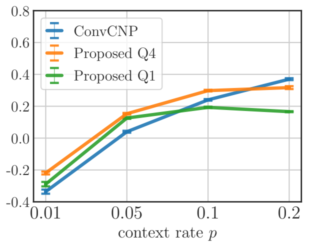


(g) 7500 tasks (out-train)

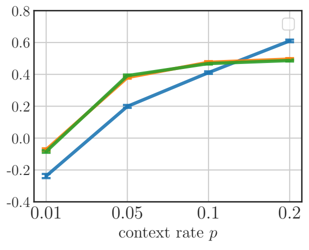


(h) 10000 tasks (out-train)

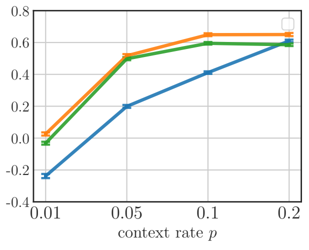
Figure 18: Test log likelihood of North America dataset over the varying number of training tasks.



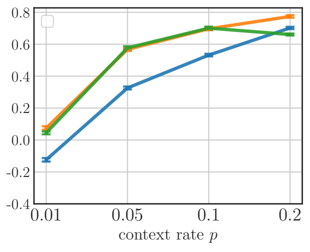
(a) 2500 tasks (in-train)



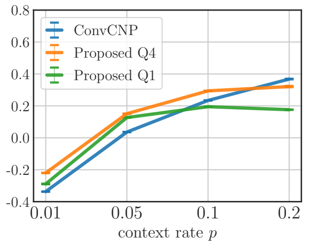
(b) 5000 tasks (in-train)



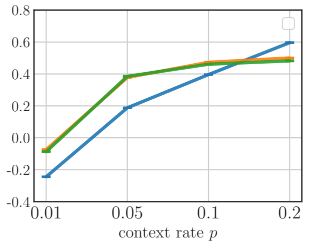
(c) 7500 tasks (in-train)



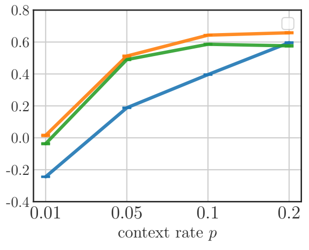
(d) 10000 tasks (in-train)



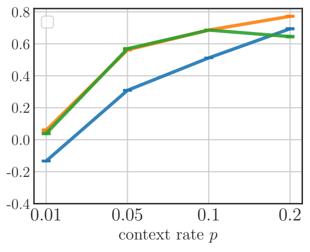
(e) 2500 tasks (out-train)



(f) 5000 tasks (out-train)



(g) 7500 tasks (out-train)



(h) 10000 tasks (out-train)

Figure 19: Test log likelihood of Europe dataset over the varying number of training tasks.

Ablation study for prior coefficient. We investigate the effect of the prior coefficient α in Eq. (18) that controls how much the random stationary prior is reflected on data representation in the same experiment setting of Section 4.3.

Fig. 20 compares the proposed method ($Q = 4$) with varying alpha $\alpha \in \{.00, .05, .10, .20\}$ on Europe set (out-train); $\alpha = 0$ denotes that the random stationary prior does not the data representation, which is similar to ConvCNP. This figure indicates that employing the proper stationary prior ($\alpha = .05, .10$) for training could improve the prediction performance on the image completion tasks.

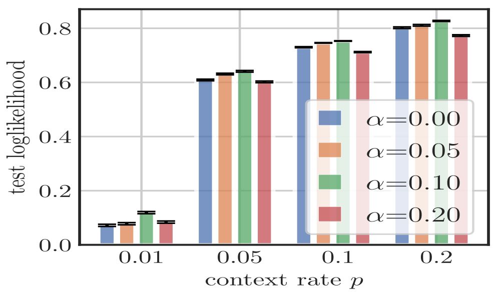


Figure 20: Effect of prior coefficient α.