SPECTRAL CONVOLUTIONAL CONDITIONAL NEURAL PROCESSES

Peiman Mohseni & Nick Duffield
Texas A&M University
{peiman.mohseni, duffieldng}@tamu.edu

ABSTRACT

Conditional Neural Processes (CNPs) constitute a family of probabilistic models that harness the flexibility of neural networks to parameterize stochastic processes. Their capability to furnish well-calibrated predictions, combined with simple maximum-likelihood training, has established them as appealing solutions for addressing various learning problems, with a particular emphasis on meta-learning. A prominent member of this family, Convolutional Conditional Neural Processes (ConvCNPs), utilizes convolution to explicitly introduce translation equivariance as an inductive bias. However, ConvCNP's reliance on local discrete kernels in its convolution layers can pose challenges in capturing long-range dependencies and complex patterns within the data, especially when dealing with limited and irregularly sampled observations from a new task. Building on the successes of Fourier neural operators (FNOs) for approximating the solution operators of parametric partial differential equations (PDEs), we propose Spectral Convolutional Conditional Neural Processes (SConvCNPs), a new addition to the NPs family that allows for more efficient representation of functions in the frequency domain.

1 INTRODUCTION

Stochastic processes are fundamental mathematical concepts extensively employed across diverse disciplines to characterize the inherent randomness observed in numerous natural phenomena. At its core, a stochastic process is defined as a typically infinite collection of random variables, indexed by a specific set. Particularly noteworthy is when these random variables pertain to elements within a function space, as exemplified by entities like Gaussian Processes (GPs). These processes find utility in representing real-valued signals that are indexed by either time or space, leading to a growing interest in the development of machine learning algorithms designed to handle data generated by such processes (Mathieu et al., 2021; Dupont et al., 2021).

Unfortunately, the conventional construction of stochastic processes like GPs, used in regression and classification tasks, can quickly become computationally challenging as the size of the dataset or dimensionality expands Garnelo et al. (2018a). The remarkable success of deep neural networks in approximating functions, especially in the presence of large datasets, has sparked a burgeoning line of research focused on using neural networks to directly parameterize stochastic processes. Neural Processes (NPs, Garnelo et al. (2018a;b)) are prominent outcomes of this perspective and have found successful applications in the realm of meta-learning (Jha et al., 2022). The use of neural networks is appealing since the majority of the computational burden is concentrated during the training process Dutordoir et al. (2023). The training regime of NPs which revolves around training over a large number of small datasets, in conjunction with their formulation, naturally fosters an exchange of information between different datasets while providing on-the-fly uncertainty estimates (Gordon et al., 2019).

Since their initial introduction, there have been various follow-up works aimed at addressing the limitations of vanilla CNPs, including issues such as underfitting (Kim et al., 2019; Nguyen & Grover, 2022; Kim et al., 2022), uncorrelated predictions (Louizos et al., 2019; Foong et al., 2020; Bruinsma et al., 2021; Markou et al., 2022; Bruinsma et al., 2023), and the use of a simple Gaussian predictive distribution (**?**). In this article, we focus specifically on Convolutional Conditional Neural Processes (ConvCNPs, Gordon et al. (2019)), which represent a variant designed to handle symmetries such as translation equivariance by integrating convolutional neural networks (CNNs). Although ConvCNPs exhibit impressive performance, they may encounter challenges when it comes to effectively integrating partial information scattered irregularly across the domain to infer the underlying pattern. This arises from the commonly used local discrete kernels, which often possess an effective memory horizon considerably shorter than the length of the input sequence (Romero et al., 2021). To circumvent this issue, we take a different approach based on the developments of neural network based surrogates for solving partial differential equations (PDEs, Raissi et al. (2019); Lu et al. (2019); Kovachki et al. (2023)). In particular, we leverage the formulation of Fourier Neural Operators (FNOs, Li et al. (2020a)) to perform global convolution. The resulting model, which we refer to as Spectral Convolution Conditional Neural Process (SConvCNP), is demonstrated to outperform baselines on challenging regression problems[1].

2 PRELIMINARIES

2.1 FOURIER NEURAL OPERATORS

Neural operators (Li et al., 2020b;a; Gupta et al., 2021; Tran et al., 2021; Kovachki et al., 2023; Liu et al., 2023; Helwig et al., 2023) are a class of models that leverage neural networks to construct a parametric approximation of mappings between Banach spaces. Much like the conventional design of a standard neural network layer, which relies on an affine transformation of the input followed by a pointwise nonlinearity, neural operators apply the linear integral operator $\mathcal{K}(v)(t) = \int \kappa(x,t)v(x)dx$ with kernel κ on the input function $v \in C_b(\mathbb{X}, \mathbb{Y})$, which is subsequently subjected to a nonlinear activation function (Kovachki et al., 2023). When κ is stationary, $\mathcal{K}(v)$ simplifies to $\kappa * v$, the convolution of κ and v. By the convolution theorem, $\mathcal{F}(\kappa * v)$, i.e. the Fourier transform of $\kappa * v$, coincides with the pointwise product of $\mathcal{F}(\kappa)$ and $\mathcal{F}(v)$. A particularly interesting scenario arises when κ is assumed to be periodic, leading to a Fourier transform that adopts the shape of a Dirac comb with discrete support. As suggested by Li et al. (2020a), both $\mathcal{F}(\kappa)$ and $\mathcal{F}(v)$ are truncated at k Fourier modes, thus allowing for direct parameterization of the kernel in the frequency domain with a finite number of complex-valued weights W_κ leading to the following formulation of the integral operator

$$\mathcal{K}(v) = \mathcal{F}^{-1}\left(W_k \,.\, \mathcal{F}\left(v\right)\right). \tag{1}$$

Augmented by the element-wise affine transformation $W_\alpha v + b_\alpha$ of v in the physical space, each operator layer is expressed as

$$\mathcal{L}(v) = \sigma(W_\alpha v + b_\alpha + \mathcal{K}(v)), \tag{2}$$

where σ is a pointwise non-linear function. In practice, when we have access to a finite set of samples of v, $\mathcal{F}(v)$ (and similarly its transform) is computed using the Fast Fourier Transform (FFT). Through the composition of operator layers as depicted in Equation (3), FNOs can effectively approximate highly intricate operators.

$$\mathcal{G}(v) = (\mathcal{Q} \circ \mathcal{L}^{(L)} \circ \cdots \circ \mathcal{L}^{(1)} \circ \mathcal{P})(v). \tag{3}$$

In this equation, L represents the number of layers, and $\mathcal{P}$ and $\mathcal{Q}$ correspond to element-wise mappings to and from a latent space.

3 METHOD

3.1 CONVOLUTIONAL CONDITIONAL NEURAL PROCESSES

Conditional Neural Processes (CNPs, Garnelo et al. (2018a;b)) are meta-learning methods that use neural networks to parameterize the predictive distribution $p(y \mid x, \mathcal{D})$ where (x, y) and $\mathcal{D} = \{(x_i, y_i) \in \mathbb{X} \times \mathbb{Y}\}_{i=1}^N$ correspond to a target input-output pair and a set of observations sampled from a task $\mathcal{T}$, respectively. In CNPs, first, the set of observations $\mathcal{D}$, also known as the context set, are embedded into a latent space $\mathbb{H}$ through an encoder $\varphi_e \colon \cup_{k=1}^{\infty} (\mathbb{X} \times \mathbb{Y})^k \to \mathbb{H}$. Next, a decoder $\varphi_d \colon \mathbb{X} \times \mathbb{H} \to \Theta$ maps target input x and the context embedding to the parameters $\theta \in \Theta$ of the conditional distribution $p_\theta(y \mid x, \mathcal{D})$. Unlike the majority of CNP variants where the latent space

[1] Code is available at `https://github.com/peiman-m/SpecConvCNP`

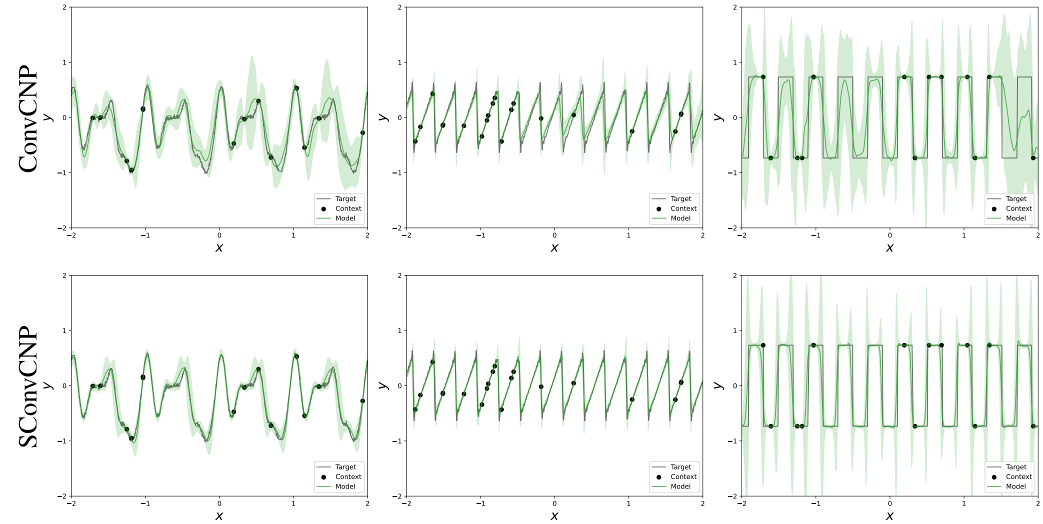

Figure 1: Illustrative functions acquired by ConvCNP (upper row) and SConvCNP (lower row) during training on a periodic kernel (first column), sawtooth function (second column), and square wave function (third column). The plots display means and two standard deviations.

$\mathbb{H}$ is chosen to be a finite-dimensional Euclidean space, ConvCNP maps the context points to an infinite-dimensional space of functions operating on $\mathbb{X}$. Consequently, the translation of inputs can be reflected in the context embedding meaningfully. More precisely, ConvCNP employs a kernel density estimator along with the Nadaraya-Watson estimator to form the functional representations $E(\mathcal{D})\colon \cup_{k=1}^{\infty} (\mathbb{X} \times \mathbb{Y})^k \to \mathbb{H}$ of the context set as follows

$$E(\mathcal{D})(t) = \left[\sum_{i=1}^{N} \psi_e(t, x_i) \, , \, \sum_{i=1}^{N} \frac{\psi_e(t, x_i) \, y_i}{\psi_e(t, x_i)} \right], \tag{4}$$

where ψ_e is a kernel function usually taken as the Gaussian kernel with a learnable length scale. This functional representation is then transformed through a translation-equivariant operator $\Phi\colon \mathbb{H} \to C_b(\mathbb{X}, \mathbb{Y})$ to form the final context representation as

$$\varphi_e(\mathcal{D}) = \Phi\left(E\left(\mathcal{D}\right)\right) \tag{5}$$

In practice, the operator Φ is implemented using a convolutional neural network (CNN) since any deep translation-equivariant model can be represented equivalently using CNNs (Kondor & Trivedi, 2018). Before applying the CNN, the range, which encompasses both the context and target inputs, is discretized uniformly at $(t_j)_{j=1}^{M} \subset \mathbb{X}$ with a predetermined density. This step, however, can be bypassed if the data is already organized on a grid that can serve as our discretization. In the end, the decoder φ_d uses the discretized context representation $(\varphi_e(\mathcal{D})(t_j))_{j=1}^{M}$ to regress the parameters of $p_\theta(y \,|\, x, \mathcal{D})$ as follows

$$\begin{aligned} \theta &= \varphi_d\left(x \, , \, \left(\varphi_e\left(\mathcal{D}\right)\left(t_j\right)\right)_{j=1}^{M}\right) \\ &= \sum_{j=1}^{M} \varphi_e(\mathcal{D})(t_j) \, \psi_d(x, \, t_j), \end{aligned} \tag{6}$$

where ψ_d is a kernel function. Additional constraints, such as ensuring the positivity of the scale parameter of the distribution, can be enforced by employing a suitable activation function to the output of the decoder. Much like the preceding step, this process can be streamlined for on-the-grid data by integrating it into the CNN (Gordon et al., 2019).

3.2 Spectral Convolutional Conditional Neural Processes

By a substantial margin, most convolutional kernels used in CNNs are designed as finite sequences of independent weights. The CNNs' effective memory horizon, which is determined by the kernel size, is generally much smaller than the input length and must be specified beforehand (Romero et al., 2021). As a result, the capability to capture dependencies that extend beyond the kernel's effective range is hindered. This limitation becomes even more pronounced when dealing with irregularly sampled or partially observed data, a situation frequently encountered in contexts where NPs are utilized. It's important to note that while ConvCNP assesses the functional representation as described in Equation (4) over a uniform grid to facilitate the use of discrete convolution kernels, the predictions derived from the kernel regression are primarily reliable in areas with a higher density of data points. Consequently, relying solely on local convolution might be insufficient for capturing the underlying patterns within the data by stitching together information from various regions. This problem, which is ubiquitous in meta-learning scenarios where limited observations from a new task are available, can be alleviated by performing global convolution. Building upon this, we suggest substituting the CNNs employed to implement the operator Φ in Equation (5) with FNOs.

4 EXPERIMENTS

We perform experiments on synthetic one-dimensional regression tasks to evaluate the efficacy of our proposed framework. The assessment involves comparing SConvCNP against three baseline models: Vanilla Conditional Neural Process (CNP), Attentive Conditional Neural Process (AttCNP, Kim et al. (2019)), and Convolutional Conditional Neural Process (ConvCNP). For the baseline models, we utilize the open-source implementation provided by Bruinsma et al. (2023), which is available at [2]. Default parameters, extensively tested and optimized across diverse tasks, are applied. Our implementation of SConvCNP closely adheres to the ConvCNP framework outlined in Bruinsma et al. (2023), with the primary distinction lying in the parameterization of the convolutional network, further detailed in Appendix A.2.

The summarized predictive log-likelihood of these methods on testing data is presented in Table 1. It is evident that SConvCNP either matches or exceeds the performance of the baseline models. Notably, the performance improvement is more pronounced in scenarios where the underlying function is periodic. This suggests that the use of global convolution may offer a more robust representation of underlying patterns by considering information collectively, as also demonstrated in the examples provided in Figure 1 where SConvCNP produces superior fits. Additional details on experimental setups can be found in Appendix A.1.

Table 1: Comparison of predictive log-likelihood obtained by different methods over synthetically generated tasks (5 Seeds).

	RBF	Matérn 5/2	Periodic	Sawtooth	Square Wave
CNP	$-0.010_{\pm 0.015}$	$-0.286_{\pm 0.010}$	$-0.935_{\pm 0.000}$	$-0.549_{\pm 0.000}$	$-1.112_{\pm 0.000}$
AttCNP	$0.171_{\pm 0.124}$	$-0.081_{\pm 0.081}$	$-0.947_{\pm 0.000}$	$-0.550_{\pm 0.000}$	$-0.624_{\pm 0.065}$
ConvCNP	$1.191_{\pm 0.069}$	$\mathbf{0.675}_{\pm 0.004}$	$0.883_{\pm 0.022}$	$1.725_{\pm 0.012}$	$0.928_{\pm 0.069}$
SConvCNP	$\mathbf{1.197}_{\pm 0.017}$	$0.635_{\pm 0.017}$	$\mathbf{1.349}_{\pm 0.018}$	$\mathbf{1.893}_{\pm 0.007}$	$\mathbf{1.446}_{\pm 0.093}$

5 CONCLUSION

In this work, we introduced Spectral Convolutional Conditional Neural Processes (SConvCNPs), a new addition to the CNP family that harnesses advancements in operator learning to enhance the expressive capabilities of Convolutional Conditional Neural Processes (ConvCNPs) when modeling stationary stochastic processes. Our experiments, conducted on synthetic datasets, demonstrated that SConvCNPs enhance the predictive performance of ConvCNPs in regression tasks, as evidenced by improvements in log-likelihood. Furthermore, they adeptly capture global symmetries, including the prevalent periodic patterns in the data.

[2] https://github.com/wesselb/neuralprocesses

ACKNOWLEDGMENTS

We would like to thank Arman Hasanzadeh and Jonathan W. Siegel for their valuable feedback and constructive suggestions. Additionally, we appreciate the support from Texas A&M High Performance Research Computing, which provided the computational resources necessary to conduct the experiments in this study.

REFERENCES

Lorenzo Bonito, James Requeima, Aliaksandra Shysheya, and Richard E Turner. Diffusion-augmented neural processes. *arXiv preprint arXiv:2311.09848*, 2023.

Wessel P Bruinsma, James Requeima, Andrew YK Foong, Jonathan Gordon, and Richard E Turner. The gaussian neural process. *arXiv preprint arXiv:2101.03606*, 2021.

Wessel P Bruinsma, Stratis Markou, James Requiema, Andrew YK Foong, Tom R Andersson, Anna Vaughan, Anthony Buonomo, J Scott Hosking, and Richard E Turner. Autoregressive conditional neural processes. *arXiv preprint arXiv:2303.14468*, 2023.

Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative models as distributions of functions. *arXiv preprint arXiv:2102.04776*, 2021.

Vincent Dutordoir, Alan Saul, Zoubin Ghahramani, and Fergus Simpson. Neural diffusion processes. In *International Conference on Machine Learning*, pp. 8990–9012. PMLR, 2023.

Andrew Foong, Wessel Bruinsma, Jonathan Gordon, Yann Dubois, James Requeima, and Richard Turner. Meta-learning stationary stochastic process prediction with convolutional neural processes. *Advances in Neural Information Processing Systems*, 33:8284–8295, 2020.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International conference on machine learning*, pp. 1704–1713. PMLR, 2018a.

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

Jonathan Gordon, Wessel P Bruinsma, Andrew YK Foong, James Requeima, Yann Dubois, and Richard E Turner. Convolutional conditional neural processes. *arXiv preprint arXiv:1910.13556*, 2019.

Gaurav Gupta, Xiongye Xiao, and Paul Bogdan. Multiwavelet-based operator learning for differential equations. *Advances in neural information processing systems*, 34:24048–24062, 2021.

Jacob Helwig, Xuan Zhang, Cong Fu, Jerry Kurtin, Stephan Wojtowytsch, and Shuiwang Ji. Group equivariant fourier neural operators for partial differential equations. *arXiv preprint arXiv:2306.05697*, 2023.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Saurav Jha, Dong Gong, Xuesong Wang, Richard E Turner, and Lina Yao. The neural process family: Survey, applications and perspectives. *arXiv preprint arXiv:2209.00517*, 2022.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019.

Mingyu Kim, Kyeongryeol Go, and Se-Young Yun. Neural processes with stochastic attention: Paying more attention to the context dataset. *arXiv preprint arXiv:2204.05449*, 2022.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *International Conference on Machine Learning*, pp. 2747–2755. PMLR, 2018.

Nikola B Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew M Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *J. Mach. Learn. Res.*, 24(89):1–97, 2023.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020a.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020b.

Ning Liu, Yue Yu, Huaiqian You, and Neeraj Tatikola. Ino: Invariant neural operators for learning complex physical systems with momentum conservation. In *International Conference on Artificial Intelligence and Statistics*, pp. 6822–6838. PMLR, 2023.

Christos Louizos, Xiahan Shi, Klamer Schutte, and Max Welling. The functional neural process. *Advances in Neural Information Processing Systems*, 32, 2019.

Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

Stratis Markou, James Requeima, Wessel P Bruinsma, Anna Vaughan, and Richard E Turner. Practical conditional neural processes via tractable dependent predictions. *arXiv preprint arXiv:2203.08775*, 2022.

Emile Mathieu, Adam Foster, and Yee Teh. On contrastive representations of stochastic processes. *Advances in Neural Information Processing Systems*, 34:28823–28835, 2021.

Tung Nguyen and Aditya Grover. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. *arXiv preprint arXiv:2207.04179*, 2022.

Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators. *arXiv preprint arXiv:2204.11127*, 2022.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

David W Romero, Anna Kuzina, Erik J Bekkers, Jakub M Tomczak, and Mark Hoogendoorn. Ckconv: Continuous kernel convolution for sequential data. *arXiv preprint arXiv:2102.02611*, 2021.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pp. 234–241. Springer, 2015.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. *arXiv preprint arXiv:2111.13802*, 2021.

A EXPERIMENTAL DETAILS

A.1 DATA GENERATION

Table 2: Synthetic processes used in 1D regression experiments. Here $\lfloor\cdot\rfloor$ indicates the floor function, and $[\![P]\!]$ takes on a value of 1 if the statement P is true, and 0 otherwise.

Process	$g(\cdot)$
GP (RBF)	$g \sim \mathcal{GP}(0, C)\,,\, C(x, x') = \sigma^2 \exp\left(-\frac{\|x-x'\|_2^2}{2\ell}\right) + \delta$
GP (Matérn 5/2)	$g \sim \mathcal{GP}(0, C)\,,\, C(x, x') = \sigma^2 (1 + \frac{\sqrt{5}d}{\ell} + \frac{5d^2}{3\ell^2}) + \delta,\, d = \|x - x'\|_2$
GP (Periodic)	$g \sim \mathcal{GP}(0, C)\,,\, C(x, x') = \sigma^2 \exp\left(-\frac{2\sin^2\left(\pi\|x-x'\|_1/p\right)}{\ell^2}\right) + \delta$
Sawtooth (Gordon et al., 2019)	$g(x) = \frac{\alpha}{2} - \frac{\alpha}{\pi}\sum_{k=1}^{K}(-1)^k \frac{\sin\left(2\pi k\omega(x+\delta)\right)}{k}$
Square Wave (Bonito et al., 2023)	$g(x) = [\![\lfloor\lfloor\omega x - \phi\rfloor\rfloor \bmod 2 = 0]\!]$

We consider data arising from the three stochastic processes described in table 2 with the following choice of parameters:

- RBF: $\ell = 0.25$, $\sigma = 0.75$, $\delta = 0.02$
- Matérn 5/2: $\ell = 0.25$, $\sigma = 0.75$, $\delta = 0.02$
- Periodic: $\ell = 0.6$, $\sigma = 0.75$, $p = 1$, $\delta = 0.02$
- Sawtooth: $\alpha \sim \mathcal{U}(1, 2)$, $\omega \sim (3, 5)$, $\delta \sim \mathcal{U}(-5, 5)$, $K \sim \mathcal{U}(10, 20)$
- Square Wave: $\alpha \sim \mathcal{U}(1, 2)$, $\omega \sim (2, 5)$, $\phi \sim \mathcal{U}(\frac{1}{\omega}, 1)$

For every functional form, the models undergo training and testing across 10^5 and 5×10^4 tasks. Each task involves samples from 16 functions drawn from a stochastic process. For each function, we restrict the inputs domain to $(-2,\, 2)$. In both the training and testing phases, the count of context points for each task is randomly chosen as any integer between 3 and 50. This selection is mirrored for the target points.

A.2 MODEL ARCHITECTURES

Consider a Fourier layer denoted by $\mathrm{F}(d_{\text{in}},\, d_{\text{out}},\, l_{\text{out}},\, m_{\text{f}})$ where d_{in}, d_{out}, l_{out} and m_{f} represent the input channels, output channels, output length, and the number of kept Fourier modes, respectively. Additionally, let $\mathrm{MLP}(d_{\text{in}},\, d_{\text{h}_1},\, \cdots,\, d_{\text{h}_\text{k}},\, d_{\text{out}})$ represent a fully connected network with Gaussian error linear unit (GeLU) activation (Hendrycks & Gimpel, 2016). This network has d_{in} input channels, d_{out} output channels, and comprises k hidden layers with sizes $d_{\text{h}_1},\, \cdots,\, d_{\text{h}_\text{k}}$. In contrast to the UNet architecture (Ronneberger et al., 2015) utilized in ConvCNP, we incorporate the following layers:

- $\mathrm{L}_0 = \mathrm{MLP}(d_{\text{in}} = 16,\, d_{\text{h}_1} = 32,\, d_{\text{out}} = 64)$
- $\mathrm{L}_1 = \mathrm{F}(d_{\text{in}} = 64,\, d_{\text{out}} = 64,\, l_{\text{out}} = \lfloor l_0/2 \rfloor,\, m_{\text{f}} = 32)$
- $\mathrm{L}_2 = \mathrm{F}(d_{\text{in}} = 64,\, d_{\text{out}} = 128,\, l_{\text{out}} = \lfloor l_0/4 \rfloor,\, m_{\text{f}} = 32)$
- $\mathrm{L}_3 = \mathrm{F}(d_{\text{in}} = 128,\, d_{\text{out}} = 256,\, l_{\text{out}} = \lfloor l_0/4 \rfloor,\, m_{\text{f}} = 32)$
- $\mathrm{L}_4 = \mathrm{F}(d_{\text{in}} = 256,\, d_{\text{out}} = 128,\, l_{\text{out}} = \lfloor l_0/2 \rfloor,\, m_{\text{f}} = 32)$
- $\mathrm{L}_5 = \mathrm{F}(d_{\text{in}} = 192,\, d_{\text{out}} = 64,\, l_{\text{out}} = l_0,\, m_{\text{f}} = 32)$
- $\mathrm{L}_6 = \mathrm{MLP}(d_{\text{in}} = 128,\, d_{\text{h}_1} = 32,\, d_{\text{out}} = 16)$

where l_0 is the length of L_0's input. The MLP layers serve the purpose of elevating the input channels to a higher dimension and projecting them back at the end of the process. During our experiments, we observed that introducing skip layers between coupled Fourier layers enhances performance, akin to the approach in UNOs (Rahman et al., 2022). The following outlines the concatenation of layers, denoted as $\mathrm{L_i} \leftarrow [\mathrm{L_j}, \mathrm{L_k}]$, signifying that the input to layer i is the concatenation of activations from layers j and k:

- $\mathrm{L_5} \leftarrow [\mathrm{L_4}, \mathrm{L_1}]$
- $\mathrm{L_6} \leftarrow [\mathrm{L_5}, \mathrm{L_0}]$

The training of all models was conducted with a batch size set to 16, utilizing the Adam optimizer with a learning rate of 0.0005.