MULTI-TASK PROCESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural Processes (NPs) consider a task as a function realized from a stochastic process and flexibly adapt to unseen tasks through inference on functions. However, naive NPs can model data from only a *single* stochastic process and are designed to infer each task independently. Since many real-world data represent a set of correlated tasks from multiple sources (*e.g.,* multiple attributes and multi-sensor data), it is beneficial to infer them jointly and exploit the underlying correlation to improve the predictive performance. To this end, we propose Multi-Task Processes (MTPs), an extension of NPs designed to jointly infer tasks realized from multiple stochastic processes. We build our MTPs in a hierarchical manner such that inter-task correlation is considered by conditioning all per-task latent variables on a single global latent variable. In addition, we further design our MTPs so that they can address multi-task settings with incomplete data (*i.e.,* not all tasks share the same set of input points), which has high practical demands in various applications. Experiments demonstrate that MTPs can successfully model multiple tasks jointly by discovering and exploiting their correlations in various real-world data such as time series of weather attributes and pixel-aligned visual modalities.

1 INTRODUCTION

Neural Processes (NPs) (Garnelo et al., 2018b) are a class of meta-learning methods that model a distribution of functions (*i.e.* a stochastic process). By considering a task as a function realized from the underlying stochastic process, they can flexibly adapt to various unseen tasks through inference on functions. The adaptation requires only one forward step of a trained neural network without any costly retraining or fine-tuning, and has linear complexity to the data size. NPs can also quantify their prediction uncertainty, which is essential in risk-sensitive applications (Gal & Ghahramani, 2016). Thanks to such appealing properties, there have been increasing attempts to improve NPs in various domains, such as image regression (Kim et al., 2019; Gordon et al., 2020), image classification (Requeima et al., 2019; Wang & Van Hoof, 2020), time series regression (Qin et al., 2019; Norcliffe et al., 2021), and spatio-temporal regression (Singh et al., 2019).

In this paper, we explore extending NPs to a multi-task setting where correlated tasks are realized simultaneously from multiple stochastic processes. Many real-world data represent multiple correlated functions, such as different attributes or modalities. For instance, medical data (Johnson et al., 2016; Harutyunyan et al., 2019) or climate data (Wang et al., 2016) contain various correlated attributes on a patient or a region that need to be inferred simultaneously. Similarly, in multi-task vision data (Lin et al., 2014; Zhou et al., 2017; Zamir et al., 2018), multiple labels of different visual modalities are associated with an image. In such scenarios, it is beneficial to exploit functional correlation by modeling the functions jointly rather than independently, in terms of performance and efficiency (Caruana, 1997). Unfortunately, naive NPs lack mechanisms to jointly handle a set of multiple functions and cannot capture their correlations either. This motivates us to extend NPs to model multiple tasks jointly by exploiting the inter-task correlation.

In addition to extending NPs to multi-task settings, we note that handling multi-task data often faces a practical challenge where observations can be incomplete (*i.e.* not all the functions share the common sample locations). For example, when we collect multi-modal signals from different sensors, the sensors may have asynchronous sampling rates, in which case we can observe signals from only an arbitrary subset of sensors at a time. To fully utilize such incomplete observations, the model should be able to associate functions observed in different inputs such that it can improve the predictive performance of all functions using their correlation. A multivariate extension of Gaussian Processes (GPs) (Álvarez et al., 2012) can handle incomplete observations to infer multiple functions jointly. However, the GP family suffers from cubic complexity to the data size and depends heavily on the kernel choice (Kim et al., 2019).

To address these challenges, we introduce Multi-Task Processes (MTPs), a new family of stochastic processes that jointly models multiple tasks given possibly incomplete data. We first design a combined space of multiple function spaces, which allows not only joint inference on multiple functions but also handling incomplete data. Then we define MTPs over the combined function space by extending the conditional Latent Variable Model (LVM) of NPs. To exploit the inter-task correlation, we introduce a hierarchical LVM consists of (1) a global latent variable that captures knowledge about all tasks and (2) task-specific latent variables that additionally capture knowledge specific to each task conditioned on the global latent variable. Inducing each task conditioned on the global latent, the hierarchical LVM allows MTP to effectively learn and exploit functional correlation in multi-task inference. MTP also inherits advantages of NPs, such as flexible adaptation, scalable inference, and uncertainty-aware prediction. Experiments in synthetic and real-world datasets show that MTPs effectively utilize incomplete observations from multiple tasks and outperform several NP variants in terms of accuracy, uncertainty estimation, and prediction coherency.

2 Preliminary

2.1 Background: Neural Processes

We consider a task $f^t : \mathcal{X} \to \mathcal{Y}^t$ as a realization of a stochastic process over a function space $(\mathcal{Y}^t)^{\mathcal{X}}$ that generates a data $D^t = (X_D, Y_D^t) = \{(x_i, y_i^t)\}_{i \in \mathcal{I}(D^t)}$, where $\mathcal{I}(D^t)$ denotes a set of data index. Neural Processes (NPs) use a conditional latent variable model to learn the stochastic process. Given a set of observations $C^t = (X_C, Y_C^t) = \{(x_i, y_i^t)\}_{i \in \mathcal{I}(C^t)}$, NP infers the target task f^t through a latent variable z and models the data D^t by a factorized conditional distribution $p(Y_D^t | X_D, z)$:

$$p(Y_D^t | X_D, C^t) = \int p(Y_D^t | X_D, z) p(z | C^t) dz = \int \prod_{i \in \mathcal{I}(D^t)} p(y_i^t | x_i, z) p(z | C^t) dz. \tag{1}$$

We refer to the set of observations C^t as a *context* data and the modeling data D^t as a *target* data.

NP approximates the conditional distribution $p(Y_D^t | X_D, z)$ and the posterior $p(z | C^t)$ by two neural networks, a decoder p_θ and an encoder q_ϕ, respectively. The networks are trained by amortized variational inference, whose objective is to maximize the following evidence lower-bound (ELBO).

$$\log p_\theta(Y_D^t | X_D, C^t) \geq \mathbb{E}_{q_\phi(z|D^t)}[\log p_\theta(Y_D^t | X_D, z)] - D_{KL}(q_\phi(z|D^t) || p_\theta(z|C^t)). \tag{2}$$

Since the prior $p_\theta(z|C^t)$ is usually intractable, it is common to approximate it by $q_\phi(z|C^t)$ (Garnelo et al., 2018b). Then the KL term acts as a regularizer for the encoder q_ϕ such that the summary of the context is close to the summary of the target. This reflects the assumption that the context and target are generated by the same underlying data-generating process and aids effective test-time adaptation. After training, NP infers the target function according to the latent variable model (Eq.1).

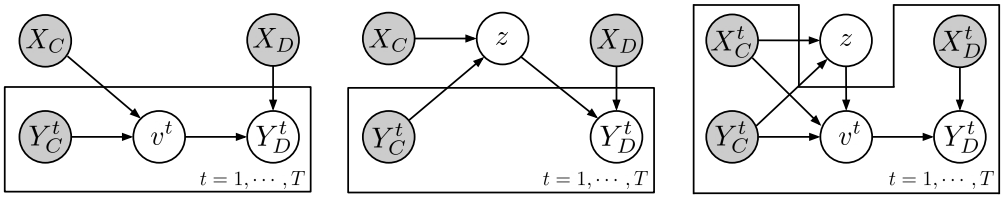
2.2 Extending to Multiple Target Functions

Now we extend the problem to consider multiple tasks $f^1, \cdots, f^T$ realized from T stochastic processes, each of which has its own function space $(\mathcal{Y}^t)^{\mathcal{X}}, \forall t \in \mathcal{T} = \{1, 2, \cdots, T\}$. Let $D = (X_D, Y_D^{1:T}) = \bigcup_{t \in \mathcal{T}} D^t$ be a multi-task target data, where each D^t corresponds to the data of task f^t. Then the multi-task objective is to model the conditional probability $p(Y_D^{1:T} | X_D, C)$ given the multi-task context $C = (X_C, Y_C^{1:T}) = \bigcup_{t \in \mathcal{T}} C^t$, where each C^t is a set of observations of task f^t.

However, assuming the complete context C for all tasks is often challenged by many practical issues, such as asynchronous sampling across multiple sensors or missing labels in multi-attribute data. To address such challenges, we relax the assumptions on context C and let $\mathcal{I}(C^t)$ be different across $t \in \mathcal{T}$. In this case, an input point x_i can be associated with a partial set of output values $\{y_i^t\}_{t \in \mathcal{T}_i}, \mathcal{T}_i \subsetneq \mathcal{T}$, which is referred *incomplete* observation. Next, we present two ways to use NPs to model the multi-task data and discuss their limitations.

Single-Task Processes (STPs) A straightforward application of NPs to the multi-task setting is assuming independence across tasks and define independent NPs over the function spaces $(\mathcal{Y}^1)^{\mathcal{X}}, \cdots, (\mathcal{Y}^T)^{\mathcal{X}}$. We refer to this approach as Single-Task Processes (STPs). Specifically, a STP has T independent latent variables $v^1, \cdots, v^T$, where each v^t implicitly represents a task f^t.

$$p(Y_D^{1:T} | X_D, C) = \prod_{t=1}^{T} \int p(Y_D^t | X_D, v^t) p(v^t | C^t) dv^t. \tag{3}$$



(a) STP (b) JTP (c) MTP

Figure 1: Graphical models of three different stochastic processes for multiple functions. Gray and white circles represent observable and latent variables, respectively.

Thanks to the independence assumption, STPs can handle incomplete context by conditioning on each task-specific data C^t independently. However, this approach can only model the marginal distributions for each task, ignoring complex inter-task correlation within the joint distribution of the tasks. Note that this is especially impractical for multi-task settings under the incomplete data since each task f^t can be learned only from C^t, ignoring rich contexts available in other data $C^{t'}$, $\forall t' \neq t$.

Joint-Task Process (JTP) An alternative approach is to combine output spaces to a product space $\mathcal{Y}^{1:T} = \prod_{t \in \mathcal{T}} \mathcal{Y}^t$ and define a single NP over the function space $(\mathcal{Y}^{1:T})^{\mathcal{X}}$. We refer to this approach as Joint-Task Process (JTP). In this case, a single latent variable z governs all tasks.

$$p(Y_D^{1:T}|X_D, C) = \int p(Y_D^{1:T}|X_D, z)p(z|C)dz. \tag{4}$$

JTPs are amenable to incorporate correlation across tasks through the shared variable z. However, by definition, they require complete context and target for both training and inference. This is because any incomplete set of output values $\{y_i^t\}_{t \in \mathcal{T}_i}$ for an input point x_i such that $\mathcal{T}_i \neq \mathcal{T}$ is not a valid element of the product space $\mathcal{Y}^{1:T}$. In addition, it relies solely on a single latent variable to explain all tasks, ignoring per-task stochastic factors in each function f^t.

In what follows, we propose an alternative formulation for jointly handling multiple tasks on incomplete data, which (1) enables a probabilistic inference on the incomplete data and (2) is more amenable for learning both task-specific and task-agnostic functional representations.

3 Multi-Task Processes

In this section, we describe *Multi-Task Processes* (MTPs), a family of stochastic processes to model multiple functions jointly and handle incomplete data. We first formulate MTPs using a hierarchical latent variable model. Then we propose the training objective and a neural network model.

3.1 Formulation

Our objective is to extend NPs to jointly infer multiple tasks from incomplete context. Discussions in Section 2.2 suggest that direct modeling of a distribution over functions of form $f : \mathcal{X} \to \prod_{t \in \mathcal{T}} \mathcal{Y}^t$ is achievable via JTP (Eq. 4), yet it requires complete data in both training and inference. To circumvent this problem, we reformulate the functional form by $h : \mathcal{X} \times \mathcal{T} \to \bigcup_{t \in \mathcal{T}} \mathcal{Y}^t$. Note that this functional form allows us to model the same set of functions as JTP by $f(x_i) = (h(x_i, 1), \cdots, h(x_i, T))$. However, by using the union form we can exploit incomplete data since any partial set of output values $\{y_i^t\}_{t \in \mathcal{T}_i}$ now becomes a set of valid output values at different input points $(x_i, t), t \in \mathcal{T}_i$. For notational convenience, we denote $x_i^t = (x_i, t)$ and assume input points in the context C and the target D are embedded by the task indices, *i.e.,* $C = (X_C^{1:T}, Y_C^{1:T}) = \bigcup_{t \in \mathcal{T}} C^t$ where $C^t = (X_C^t, Y_C^t) = \{(x_i^t, y_i^t)\}_{i \in \mathcal{I}(C^t)}$ and the same for D.

Next, we present a latent variable model that induces a stochastic process over functions of form h. To make use of both task-agnostic and task-specific knowledge, we define a hierarchical latent variable model (Figure 1(c)). In this model, the global latent variable z captures shared stochastic factors across tasks using the whole context C, while per-task stochastic factors are captured by the task-specific latent variable v^t using C^t and z. It induces the predictive distribution on the target by:

$$p(Y_D^{1:T}|X_D^{1:T}, C) = \int \int \left[\prod_{t=1}^{T} p(Y_D^t|X_D^t, v^t)p(v^t|z, C^t) \right] p(z|C)dv^{1:T}dz, \tag{5}$$

where $v^{1:T} := (v^1, \cdots, v^T)$. Similar to Eq. 1, we assume the conditional independence on $p(Y_D^t|X_D^t, v^t)$. Note that this hierarchical model can capture and leverage the inter-task correlation by sharing the same z across $v^{1:T}$. Also, it is amenable to fully utilize the incomplete data: since the global variable z is inferred from the entire context data $C = \bigcup_{t\in\mathcal{T}} C^t$ and is conditioned to infer task-specific latent variable v^t, each function f^t induced by v^t exploits the observations available for not only itself C^t, but also for other tasks $C^{t'}, \forall t' \neq t$. Next, we show that Eq. 5 induces a stochastic process over the functions of form $h : \mathcal{X} \times \mathcal{T} \to \bigcup_{t\in\mathcal{T}} \mathcal{Y}^t$.

Proposition 1. *Consider the following generative process on data D and context C, which is a generalized form of Eq. 5.*

$$z \sim p(z|C),\ v^t \sim p(v^t|z, t, C),\ y_i^t \sim p(y_i^t|x_i^t, v^t),\ \forall t \in \mathcal{T},\ \forall i \in \mathcal{I}(D). \tag{6}$$

Then under some mild assumptions, there exists a stochastic process over functions of form $h : \mathcal{X} \times \mathcal{T} \to \bigcup_{t\in\mathcal{T}} \mathcal{Y}^t$, where the data D is generated.

Proof. We leave the proof in Appendix A.2. □

We refer to the resulting stochastic processes as Multi-Task Processes (MTPs). In the perspective of stochastic process, Eq. 5 allows us to learn functional posterior not only on each task via v^t, but also across the tasks via z. Then optimizing Eq. 5 can be interpreted as learning to learn each task captured by v^t together with the functional correlation captured by z.

3.2 Learning and Inference

We use an encoder network q_ϕ and a decoder network p_θ to approximate hierarchical posterior and conditional likelihood in Eq. 5. Since the direct optimization of Eq. 5 is intracable, we train the networks via the following variational lower bound:

$$\begin{aligned}
&\log p_\theta(Y_D^{1:T}|X_D^{1:T}, C) \\
&\geq \mathbb{E}_{q_\phi(z|D)}\Big[\sum_{t=1}^{T} \mathbb{E}_{q_\phi(v^t|z,D^t)}[\log p_\theta(Y_D^t|X_D^t, v^t)]] - D_{\mathrm{KL}}\Big(q_\phi(v^t|z, D^t) \mid\mid q_\phi(v^t|z, C^t)\Big)\Big] \\
&\qquad - D_{KL}\Big(q_\phi(z|D) \mid\mid q_\phi(z|C)\Big),
\end{aligned} \tag{7}$$

We leave the derivation in Appendix A.3. The above objective reflects several desirable behaviors for our model. Similar to NP, the KL divergences encourage that both latent variables z and v^t inferred from the context data are consistent with those inferred from the entire target data. On the other hand, we observe that minimizing the KL divergence on task-specific variables forces the global latent z to be informative across all tasks, such that it can induce the task-specific factors v^t from the limited context C^t. This makes the model encode *correlated* information across tasks in z and use it for inferring each task with v^t, which is critically important for joint inference with incomplete context data. After training, MTP infers the target functions according to the latent variable model (Eq. 5).

3.3 Neural Network model for MTP

This section presents an implementation of MTPs composed of an encoder q_ϕ and a decoder p_θ (Eq. 7). While our MTP formulation is not restricted to a specific architecture, we adopt ANP (Kim et al., 2019) as our backbone, which implements the encoder by attention layers (Vaswani et al., 2017) and the decoder by a MLP. Figure 2 illustrates the overall architecture.

In the following, we denote a stacked multi-head attention block (Parmar et al., 2018) by $\mathit{Attn}(Q, K, V)$ and a MLP by $\psi(x)$. Also, we denote e^t by a learnable task embedding for $t \in \mathcal{T}$ which is used to condition on the task index t.

Latent Encoder The latent encoder samples global and per-task latent variables by aggregating the context C. For each context example $(x_i^t, y_i^t) \in C^t$, we first project it to a hidden representation $s_i^t = \psi_s(x_i, y_i^t) + e^t$. Then we aggregate them to a task-specific representation s^t via self-attention followed by a pooling operation, which is further aggregated to a global representation s.

$$s^t = \mathit{pool}(\mathit{Attn}(\{s_i^t\}_{i\in\mathcal{I}(C^t)}, \{s_i^t\}_{i\in\mathcal{I}(C^t)}, \{s_i^t\}_{i\in\mathcal{I}(C^t)})), \quad \forall t \in \mathcal{T}, \tag{8}$$

$$s = \mathit{pool}(\mathit{Attn}(\{s^t\}_{t\in\mathcal{T}}, \{s^t\}_{t\in\mathcal{T}}, \{s^t\}_{t\in\mathcal{T}})). \tag{9}$$

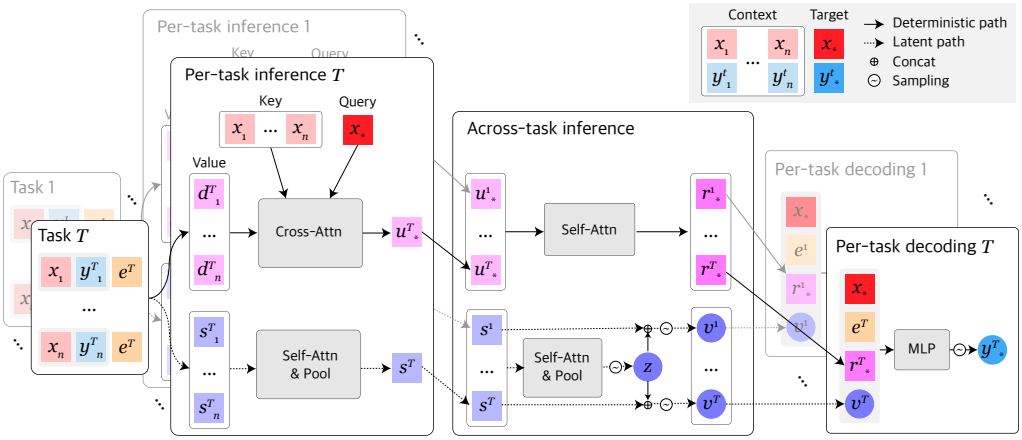


Figure 2: Architecture of the neural network model for MTP.

Note that the first attention is applied along the example axis (per-task) to encode information of each task, while the second one is applied along the task axis (across-task) to aggregate the information across the tasks. Then, we get the global and task-specific latent variables via ancestral sampling.

$$z \sim \mathcal{N}(\psi_{(z,1)}(s), \psi_{(z,2)}(s)), \tag{10}$$

$$v^t \sim \mathcal{N}(\psi_{(v^t,1)}(s^t, z), \psi_{(v^t,2)}(s^t, z)), \quad \forall t \in \mathcal{T}. \tag{11}$$

Deterministic Encoder To further improve the expressiveness of model, we extend the deterministic encoder of Kim et al. (2019) that produces local representation specific to both target example and task via attention mechanism. As in the latent encoder, we first project each context example $(x_i^t, y_i^t) \in C^t$ to a hidden representation $d_i^t = \psi_d(x_i, y_i^t) + e^t$ that serves as *value* embedding in cross-attention. Also, we use context and target input x_i^t as *key* and *query* embeddings for the cross-attention, respectively. Then we apply cross-attention along the example axis (per-task) followed by self-attention along the task axis (across-task).

$$\{u_i^t\}_{i \in \mathcal{I}(D^t)} = \mathit{Attn}(\{x_i^t\}_{i \in \mathcal{I}(C^t)}, \{x_i^t\}_{i \in \mathcal{I}(C^t)}, \{d_i^t\}_{i \in \mathcal{I}(D^t)}), \quad \forall t \in \mathcal{T}, \tag{12}$$

$$\{r_i^t\}_{t \in \mathcal{T}} = \mathit{Attn}(\{u_i^t\}_{t \in \mathcal{T}}, \{u_i^t\}_{t \in \mathcal{T}}, \{u_i^t\}_{t \in \mathcal{T}}), \quad \forall i \in \mathcal{I}(D). \tag{13}$$

Decoder Finally, the decoder produces predictive distributions for the target output $y_i^t \in Y_D^t$ for each target input x_i^t. We first project the input to $w_i^t = \psi_w(x_i) + e^t$, then concatenate it with the corresponding latent variable v^t and determinstic representation r_i^t. The output distribution is computed by MLPs, whose output depends on the type of the task.

$$y_i^t \sim \begin{cases} \mathcal{N}(\psi_{(y,1)}(w_i^t, v^t, r_i^t), \psi_{(y,2)}(w_i^t, v^t, r_i^t)), & \text{if } y_i^t \text{ is continuous,} \\ \text{Categorical}(\psi_{(y,1)}(w_i^t, v^t, r_i^t)), & \text{if } y_i^t \text{ is discrete,} \end{cases} \quad \forall i \in \mathcal{I}(D^t), \forall t \in \mathcal{T}. \tag{14}$$

4 Related Work

Stochastic Processes for Multi-Task Learning There exist several stochastic processes related to MTPs that consider learning multiple tasks. Multi-Output Gaussian Processes (MOGPs) (Álvarez et al., 2012) extends Gaussian Processes (GPs) to infer multiple tasks together, and can also handle incomplete data. However, MTPs and MOGPs have a major difference in training behaviors. MTPs learn inter-task correlation from multiple context and target pairs through meta-training, which enables accurate inference with a few observations. In contrast, MOGPs learn the correlation only from the context data given at inference time, thus require a lot of observations to produce accurate predictions. They also suffer from cubic computational cost and strong dependence on the kernel structure. Sequential Neural Processes (SNPs) (Singh et al., 2019; Yoon et al., 2020) treat a sequence of tasks using a sequence of NPs. While SNPs are designed to model temporal dynamics underlying a sequence of homogeneous tasks, MTPs can capture arbitrary correlation across a set of heterogeneous tasks. Finally, Conditional Neural Adaptive Processes (CNAPs) (Requeima et al., 2019; Bateni et al., 2020) consider a general classification model for heterogeneous label spaces (*i.e.,* different sets of classes). However, like NPs, they infer each task independently and do not explicitly consider inter-task correlation during inference. Also, CNAPs are designed specifically to classification tasks, while MTPs are generally applicable to various tasks including classification and regression.

Hierarchical Models in Neural Process Family Since the pioneering work by Garnelo et al. (2018b), several variants of NP introduced the concept of hierarchical modeling. Attentive Neural Processes (ANPs) (Kim et al., 2019) incorporate attention mechanism to a deterministic variable, which is additional context information for each target example to improve the expressive power of the model and prevent the underfitting issues of vanilla NPs. Similarly, Wang & Van Hoof (2020) introduce local latent variables to incorporate example-specific stochasticity, which extends the graphical model of NPs to the hierarchical one. Our MTP formulation also involves a hierarchical latent variable model but has a different structure orthogonal to the prior works. MTPs use a global latent variable to jointly model multiple task functions while using per-task latent variables to capture task-specific stochasticity. Although extending the model to contain example-level local latent variables is possible, we adopt the deterministic local representation as in ANPs for simplicity.

5 EXPERIMENTS

We evaluate MTP on three datasets, including both synthetic and real-world tasks. In all experiments, we construct incomplete context data by selecting a complete subset $C \subset D$ of size $m = |\mathcal{I}(C)|$ from the target, then randomly drop the output points independently according to the *missing rate* $\gamma \in [0, 1]$ ($\gamma = 0$ means complete data). We repeat the procedure with five different random seeds and report the mean values of each evaluation metric.

Baselines In each experiment, we compare MTP with two NP variants, STP and JTP. We adopt ANP (Kim et al., 2019) as a backbone architecture for STP and JTP, which is a strong NP baseline. Since JTP cannot handle incomplete data, we build a stronger baseline by the combination of STP and JTP (STP+JTP), where missing labels are imputed by STP and then used to jointly infer the tasks by JTP. In 1D regression tasks, we additionally compare two Multi-Output Gaussian Processes baselines, CSM (Ulrich et al., 2015) and MOSM (Parra & Tobar, 2017). At training time, we set $\gamma = 0.5$ for all models but keeping $\gamma = 0$ for JTP. At test time, we evaluate the models in various missing rates $\gamma \in \{0, 0.25, 0.5, 0.75\}$. We provide architectural and training details in Appendix B.

5.1 1D CURVE REGRESSION ON SYNTHETIC DATA

Dataset and Metric We begin with 1D synthetic regression tasks where the target functions are correlated by shared parameters (*e.g.,* scale, bias, phase) but have different shapes. Inspired by Guo et al. (2020b), we first randomly sample global parameters $a, b, c, w \in \mathbb{R}$ shared across the tasks, then generate four correlated tasks using different activation functions as follows.

$$y_i^t = a \cdot \text{act}_t(wx_i + b) + c, \quad \text{act}_t \in \{\text{Sine}, \text{Tanh}, \text{Sigmoid}, \text{Gaussian}\},\ x_i \sim \mathcal{U}(-5, 5). \tag{15}$$

To simulate task-specific stochasticity, we perturb the parameters (a, b, c, w) with small *i.i.d.* Gaussian noises per task. In this setting, the model has to learn per-task functional characteristics imposed by different activation functions and per-task noises, as well as how to share the underlying parameters unseen during training among the tasks. For evaluation, we generate training and testing sets via non-overlapping splits of the parameters, then measure mean squared error (MSE) normalized by the scale parameter a to aggregate results on functions with different scales. See Appendix C for details.

Results Table 1 shows the quantitative results with $\gamma = 0.5$. More comprehensive results with different missing rates are provided in Appendix D. As it shows, MTP outperforms all baselines in all tasks and context sizes. This can be attributed to the ability of MTP to (1) exploit all available context examples to infer inter-task general knowledge (*i.e.* a, b, c, w) and (2) translate it back to functional representations for each task. In contrast, STP fails to predict multiple tasks accurately due to the independent task assumption. Although JTP is designed to discover and utilize inter-task correlations, its performances do not show dramatic improvement over STP since its observations are largely based on noisy imputations from STP. We also observe that GP baselines (MOSM, CSM) perform even worse than STP, despite their inherent ability to joint inference on incomplete data. We conjecture that it is because GPs lack a meta-training mechanism that allows NPs (and MTPs) to learn the underlying stochastic processes through various realizations in the meta-training set. As an illustrating example, we also plot predicted distributions from the models in a highly incomplete scenario ($m = 10$ and $\gamma = 0.5$) in Figure 3 (a). We observe that STP generally suffers from inaccurate predictions due to limited context, while MTP successfully exploits incomplete observations from different tasks to improve the predictive performance.

We also perform an ablation study on the latent variable model to justify the effectiveness of our hierarchical formulation. We consider two variants of MTP that consist of the global latent variable

Table 1: Average normalized MSE on synthetic tasks, with varying context size (m) and $\gamma = 0.5$.

task	Sine			Tanh			Sigmoid			Gaussian		
m	5	10	20	5	10	20	5	10	20	5	10	20
MOSM	1.8653	0.7057	0.4452	1.9980	0.6220	0.3625	1.6795	0.3758	0.1769	1.5285	0.4438	0.2045
CSM	0.5516	0.6724	0.6975	0.3580	0.4445	0.5134	0.2584	0.3003	0.3821	0.2992	0.3345	0.4240
STP	0.5212	0.2609	0.0993	0.1307	0.0468	0.0159	0.0203	0.0067	0.0025	0.0799	0.0409	0.0222
STP+JTP	0.3848	0.2340	0.1114	0.1015	0.0418	0.0168	0.0163	0.0065	0.0032	0.0613	0.0318	0.0161
MTP	**0.2636**	**0.1137**	**0.0485**	**0.0435**	**0.0115**	**0.0040**	**0.0066**	**0.0014**	**0.0006**	**0.0360**	**0.0132**	**0.0069**

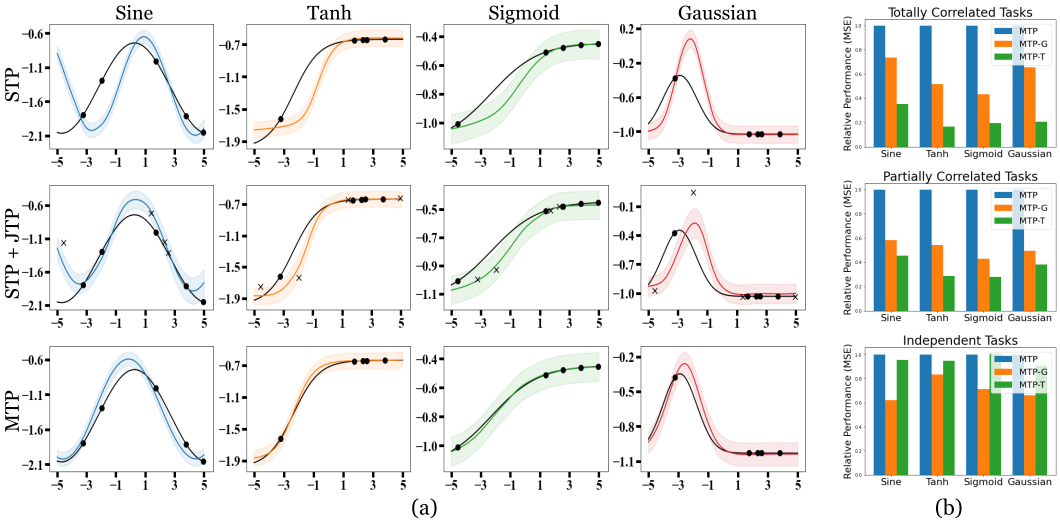


Figure 3: (a) Predicted mean (colored lines) and variance (shaded region) in synthetic tasks. Black line: true function. Black dots: context points. Black crosses: imputed points from STP. (b) Relative performance of MTP variants on synthetic tasks with different levels of inter-task correlation. Top: on totally correlated tasks. Middle: on partially correlated tasks. Bottom: on independent tasks.

only (MTP-G) and the task-specific latent variables only (MTP-T). Then we evaluate the models in three synthetic datasets generated with different levels of inter-task correlation. Specifically, we construct partially correlated tasks as described before, totally correlated tasks by removing the task-specific noises, and independent tasks by sampling the data parameters a, b, c, w independently to each task. Figure 3 (b) shows the result in $m = 10$ and $\gamma = 0.5$. When the tasks are correlated (first and second rows of the figure), we can see introducing global latent improves the overall performance, which is further improved by the hierarchical formulation. When the tasks are independent (third row of the figure), sharing all knowledge through a single global latent degrades the performance. The result demonstrates that incorporating both global and task-specific information is the most effective and robust against various levels of inter-task correlation.

5.2 1D Time-Series Regression on Weather Data

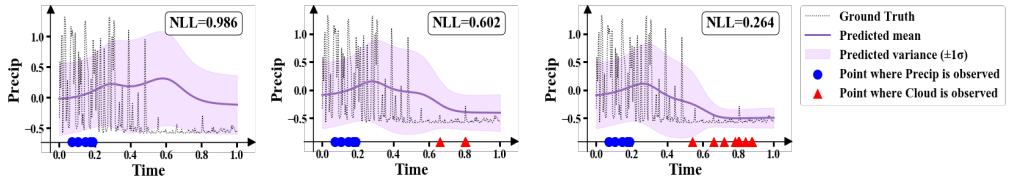
Dataset and Metric To demonstrate our method in a practical, real-world domain, we perform an experiment on weather data. Weather attributes are physically correlated with each other, and the observations are often incomplete due to different sensor configurations or coverage per station. Also, the observed attributes are highly stochastic, making MTP's stochastic process formulation fits it well. We use a dataset gathered by Dark Sky API [1], consisting of 12 daily weather attributes collected at 266 cities for 258 days. We choose six attributes, namely low and high temperatures (TempMin, TempMax), humidity (Humidity), precipitation probability (Precip), cloud cover (Cloud), and dew point (Dew), which forms six correlated tasks. We normalize each attribute to be standard Gaussian and the time to be in $[0, 1]$. We divide the data into 200 training, 30 valid, and 33 test sets of time series, where each set corresponds to a unique city. We evaluate the prediction performance by MSE. Since the data is noisy, we also report negative log-likelihood as a metric of uncertainty estimation.

Results Table 2 summarizes quantitative results. More comprehensive results with different missing rates and context sizes are provided in Appendix E. MTP outperforms all baselines in both accuracy and uncertainty estimation, which demonstrates that it generalizes well to real-world stochastic data.

[1] https://github.com/imantsm/COVID-19

Table 2: Average MSE and NLL on weather tasks, with $m = 10$ and $\gamma = 0.5$.

task	TempMin		TempMax		Humidity		Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL	MSE	NLL	MSE	NLL	MSE	NLL
CSM	0.0299	-0.4795	0.0407	-0.1961	0.1387	1.4324	0.4016	3.3915	0.4224	3.2398	0.0360	-0.2418
MOSM	0.0188	-0.6842	0.0274	-0.5108	0.1082	0.8513	0.3693	3.5583	0.3657	3.2127	0.0185	-0.6240
STP	0.0052	-1.1281	0.0071	-1.0362	0.0622	0.0980	0.2652	0.9767	0.2617	0.7814	0.0089	-0.9634
STP+JTP	0.0045	-1.1435	0.0070	-1.0161	0.0610	-0.0351	0.2400	0.6877	0.2400	0.7049	0.0086	-0.9625
MTP	**0.0040**	**-1.1884**	**0.0059**	**-1.0912**	**0.0553**	**-0.1267**	**0.2313**	**0.6560**	**0.2265**	**0.6890**	**0.0077**	**-1.0239**



(a) 5 complete contexts (b) +3 contexts (Cloud only) (c) +10 contexts (Cloud only)

Figure 4: Visualization of MTP's internal knowledge transfer. By observing additional data from Cloud task (at red triangles) given upon a few context points (at blue dots), the predicted mean and variance of Precip task improve at the additionally observed region.

More interestingly, Figure 4 illustrates how MTP transfer its knowledge from one task (Cloud) to another (Precip) given the incomplete observations. When the observation is sparse (Figure 4(a)), the model produces an inaccurate prediction with high uncertainty for unobserved input domains. However, when the additional observations are available for the other attribute (Figure 4(b),(c)), MTP successfully transfers the knowledge to improve the prediction. It shows that MTP can effectively learn to exploit the incomplete observation by transferring knowledge across tasks.

5.3 2D Image Completion on Face Data

Dataset and Metric We further demonstrate our approach to more challenging 2D structured function regression tasks. Following Garnelo et al. (2018a), we interpret an RGB image as a function that maps a 2D pixel location $x_i \in [0, 1]^2$ to its RGB values $y_i \in [0, 1]^3$, and extend its concept to pixel-aligned 2D spatial data for the multi-task setting. Specifically, we consider four pixel-aligned visual modalities with a resolution of 32×32 on celebrity faces as a set of tasks, namely RGB image (RGB) (Liu et al., 2015), semantic segmentation map (Segment) (Lee et al., 2020), Sobel edge (Edge) (Kanopoulos et al., 1988), and Projected Normalized Coordinate Code (PNCC) (Zhu et al., 2016). We then construct training and testing sets with non-overlapping splits of face images. To evaluate the Segment task, we report mean Intersection-over-Union (mIoU). For the other tasks, we report MSE. We also measure prediction coherency across tasks to evaluate the task correlation captured by models. To measure the coherency between the predictions, we generate pseudo-labels by translating the RGB prediction into the other three modalities using image-to-image translation methods (Kanopoulos et al., 1988; Guo et al., 2020a; Chen et al., 2018), then measure errors (MSE or 1 - mIoU) between the pseudo-labels and predictions. Additional details are provided in Appendix F.

Results Table 3 summarizes the quantitative comparison results. More comprehensive results with different missing rates are provided in Appendix G. Overall, we observe similar results with the 1D regression experiments where MTP generates more accurate predictions over STP and STP+JTP by effectively exploiting the incomplete data. We also observe that the MTP produces more coherent predictions over the baselines, which shows that it indeed learns to exploit the correlation across tasks effectively. To further validate the results, we present qualitative comparison results in Figure 5. We observe that STP and STP+JTP produce inaccurate (red boxes) or incoherent (green box) outputs when the number of contexts is extremely small. On the other hand, MTP (1) consistently regresses coherent functions regardless of the number of observable contexts, and (2) its predictions are more accurate than the baselines given the same number of contexts (green box).

Finally, we investigate the discovery and exploitation of task correlations achieved by MTP. We first partition tasks into *source* and *target* tasks. Then, we measure relative performance improvement on the target tasks before and after the model observes data from source tasks. We summarize the results in Table 4, where we average performance gains coming from all possible combinations of source tasks for each target task. By observing which task is the most beneficial to each of the other tasks, we observe that there are two groups of highly correlated tasks (RGB-Edge) and (Segment-PNCC). These

Table 3: Quantitative results on 2D function regression ($\gamma = 0.5$). Upper rows show prediction performance and lower rows show prediction coherency, reported by MSE and (1-mIoU) for continuous and categorical data, respectively (lower-the-better).

Tasks	RGB			Edge			Segment			PNCC		
m	10	100	512	10	100	512	10	100	512	10	100	512
STP	0.0440	0.0154	0.0054	0.0359	0.0256	0.0116	0.6637	0.4669	0.2958	0.0102	0.0015	0.00061
STP+JTP	0.0421	0.0129	0.0046	0.0338	0.0190	0.0090	0.6316	0.4341	0.3171	0.0105	0.0021	0.00088
MTP	**0.0400**	**0.0114**	**0.0032**	**0.0323**	**0.0166**	**0.0060**	**0.6073**	**0.4013**	**0.2882**	**0.0082**	**0.0012**	**0.00058**
STP	-	-	-	0.0314	0.0261	0.0174	0.6632	0.5863	0.5361	0.0362	0.0267	0.0231
STP+JTP	-	-	-	0.0187	0.0124	0.0143	0.5298	0.5110	0.5140	0.0106	0.0139	0.0194
MTP	-	-	-	**0.0184**	**0.0089**	**0.0053**	**0.5161**	**0.4923**	**0.4963**	**0.0104**	**0.0115**	**0.0134**

Figure 5: Qualitative results on 2D function regression. Performances of all models improve as the number of observable contexts (m) increases. However, under the limited number of observable contexts (*e.g.* $m = 10$), STP and STP+JTP produce inaccurate outputs (*e.g.* mis-predicting hairs and poses as in the green box) or incoherent outputs (*e.g.* different head poses as in the red box).

results demonstrate that MTP successfully captured dependence among tasks considering that (1) RGB and Edge are composed of two correlated low-level signals (*e.g.* color intensity and its gradients) (2) while both Segment and PNCC contain high-level semantic information on facial landmarks. Note that discovering inter-task correlations is one of the actively studied topics in the machine learning literature, where the efforts often come at the cost of extra computations and resources due to hard-coded (Zamir et al., 2018; Standley et al., 2020) or hand-crafted (Pal & Balasubramanian, 2019) algorithms.

Table 4: Relative performance gain (%).

Source \ Target	RGB	Edge	Segment	PNCC
RGB	-	**52.89**	8.74	18.45
Edge	**7.02**	-	8.19	15.01
Segment	5.43	33.03	-	**28.64**
PNCC	5.69	31.73	**15.89**	-

6 Conclusion

We propose Multi-Task Processes (MTPs), a new family of stochastic processes designed to infer multiple functions jointly from incomplete data, along with a hierarchical latent variable model. Through extensive experiments, we demonstrate that the proposed MTPs can leverage incomplete data to solve multiple heterogenous tasks by learning to discover and exploit task-agnostic and task-specific knowledge. Scaling up our method to large-scale datasets will be a promising research direction. To this end, our method can be improved in several aspects by (1) generalizing to unseen task space $\mathcal{T}$ and (2) allowing empty context data for some tasks such that we can generalize MTPs in more diverse real-world scenarios such as zero-shot inference, semi-supervised learning, *etc.*

Ethics Statement Recently, detecting and removing data bias have become essential problems towards producing fair machine learning models. We believe that our work can contribute to detect unintentional data bias present in multi-attribute data. MTP can be seen as a universal correlation learner who learns arbitrary correlation across tasks purely data-driven way. Therefore, given potentially biased multi-attribute data (e.g., multiple personal attributes), MTP may detect any biased relationship by learning the correlation between them. For example, we may perform the task-to-task transfer analysis on a trained MTP as discussed in Section 5.2 and Section 5.3, then see which task (or attribute) has a high correlation with another task (or attribute).

Reproducibiltiy Statement In this work, we present two major theoretical results (Proposition 1 and Eq. 7), a neural network model (Section 3.3), and experiments on three datasets (Section 5). We give a complete proof of Proposition 1 in Appendix A.2 and the ELBO derivation for Eq. 7 in Appendix A.3. We provide architectural details and training hyper-parameters of the models used in the experiments in Appendix B. Finally, details on the experimental settings and datasets are provided in Appendix C and Appendix F.

REFERENCES

Mauricio A Álvarez, Lorenzo Rosasco, and Neil D Lawrence. Kernels for vector-valued functions: A review. *Found. Trends Mach. Learn.*, 2012.

Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *CVPR*, 2020.

John Canny. A computational approach to edge detection. *PAMI*, 1986.

Rich Caruana. Multitask learning. *Machine learning*, 1997.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

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 *ICML*, 2018a.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. In *ICML Workshop*, 2018b.

Jonathan Gordon, Wessel P. Bruinsma, Andrew Y. K. Foong, James Requeima, Yann Dubois, and Richard E. Turner. Convolutional conditional neural processes. In *ICLR*, 2020.

Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z Li. Towards fast, accurate and stable 3d dense face alignment. In *ECCV*, 2020a.

Pengsheng Guo, Chen-Yu Lee, and Daniel Ulbricht. Learning to branch for multi-task learning. In *ICML*, 2020b.

Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

Kiyosi Itô et al. *An Introduction to Probability Theory*. Cambridge University Press, 1984.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 2016.

Nick Kanopoulos, Nagesh Vasanthavada, and Robert L Baker. Design of an image edge detection filter using the sobel operator. *JSSC*, 1988.

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

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012.

Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *CVPR*, 2020.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*, 2019.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.

Alexander Norcliffe, Cristian Bodnar, Ben Day, Jacob Moss, and Pietro Liò. Neural ode processes. In *ICLR*, 2021.

Arghya Pal and Vineeth N Balasubramanian. Zero-shot task transfer. In *CVPR*, 2019.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, 2018.

Gabriel Parra and Felipe Tobar. Spectral mixture kernels for multi-output gaussian processes. In *NeurIPS*, 2017.

Shenghao Qin, Jiacheng Zhu, Jimmy Qin, Wenshuo Wang, and Ding Zhao. Recurrent attentive neural process for sequential data. In *NeurIPS Workshop*, 2019.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *NeurIPS*, 2019.

Gautam Singh, Jaesik Yoon, Youngsung Son, and Sungjin Ahn. Sequential neural processes. In *NeurIPS*, 2019.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *ICML*, 2020.

Kyle Ulrich, David E Carlson, Kafui Dzirasa, and Lawrence Carin. Gp kernels for cross-spectrum analysis. In *NeurIPS*, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Qi Wang and Herke Van Hoof. Doubly stochastic variational inference for neural processes with hierarchical latent variables. In *ICML*, 2020.

Tongli Wang, Andreas Hamann, Dave Spittlehouse, and Carlos Carroll. Locally downscaled and spatially customizable climate data for historical and future periods for north america. *PloS one*, 2016.

Pavel Yakubovskiy. Segmentation models pytorch. `https://github.com/qubvel/segmentation_models.pytorch`, 2020.

Jaesik Yoon, Gautam Singh, and Sungjin Ahn. Robustifying sequential neural processes. In *ICML*, 2020.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, 2018.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

Xiangyu Zhu, Zhen Lei, Xiaoming Liu, Hailin Shi, and Stan Z. Li. Face alignment across large poses: A 3d solution. In *CVPR*, 2016.

APPENDIX

A THEORETICAL JUSTIFICATIONS

In this section, we give a proof of Proposition 1 with a brief introduction of the Kolmogorov Extension Theorem (Itô et al., 1984), and derive training objectives of STP, JTP, and MTP.

A.1 STOCHASTIC PROCESSES AND THE KOLMOGOROV EXTENSION THEOREM

A stochastic process $F : \mathcal{X} \times \Omega \to \mathcal{Y}$ is a collection of random variables $\{\mathbb{Y}_x : \Omega \to \mathcal{Y}\}_{x \in \mathcal{X}}$ which is indexed by an index set $\mathcal{X}$. Also, all the random variables are defined on a single probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and a value space $\mathcal{Y}$. This can be interpreted as a distribution over a function space $\mathcal{Y}^{\mathcal{X}}$, such that sampling a function f corresponds to $f(\cdot) = F(\cdot, \omega),\ \omega \in \Omega$. Another interpretation of F is a random function, since $F(x, \cdot) = \mathbb{Y}_x$ is a random variable.

Suppose we have observed input and output sequences $X = (x_1, x_2, \cdots, x_n)$ and $Y = (y_{x_1}, y_{x_2}, \cdots, y_{x_n})$ of a function $f : \mathcal{X} \to \mathcal{Y}$. With a slight abuse of a notation, let $p(Y|X) = \rho_X(Y)$ be the marginal distribution of Y on a product space $\prod_{i=1}^{n} \mathcal{Y}$, where each i-th space of the product is the output space of $\mathbb{Y}_{x_i}$, $1 \le i \le n$. Then by the Kolmogorov Extension Theorem, the data (X, Y) induces a stochastic process F such that $\exists\, \omega \in \Omega$ s.t. $y_{x_i} = F(x_i, \omega)$ for all $i = 1, 2, \cdots, n$, if the distribution $p(Y|X)$ satisfies two conditions: consistency and exchangability.

1. (Consistency) For any m such that $1 \le m < n$,
$$\int p(Y|X) dY_{m+1:n} = p(Y_{1:m}|X_{1:m}), \tag{16}$$
where $X_{i_1:i_2} = (x_{i_1}, x_{i_1+1}, \cdots, x_{i_2})$ and $Y_{i_1:i_2} = (y_{i_1}, y_{i_1+1}, \cdots, y_{i_2})$ for all $i_1 \le i_2$.
2. (Exchangability) For any permutation π on $\mathcal{X}_{1:n}$ (a permutation π on set S is a bijection $\pi : S \to S$),
$$p(\pi \circ Y | \pi \circ X) = p(Y|X), \tag{17}$$
where $\mathcal{X}_{1:n} = \{x_1, \cdots, x_n\}$, $\pi \circ X = (\pi(x_1), \pi(x_2), \cdots, \pi(x_n))$, and $\pi \circ Y = (y_{\pi(x_1)}, y_{\pi(x_2)}, \cdots, y_{\pi(x_n)})$.

A.2 MTP IS A STOCHASTIC PROCESS

In the case of MTP, we observe input and output sequences $X = ((x_1, t_1), (x_2, t_2), \cdots, (x_n, t_n))$ and $Y = (y_{(x_1,t_1)}, y_{(x_2,t_2)}, \cdots, y_{(x_n,t_n)})$ of a function $h : \mathcal{X} \times \mathcal{T} \to \bigcup_{t \in \mathcal{T}} \mathcal{Y}^t$. Note that in the main text, we abbreviate $x_i^t = (x_i, t)$ and $y_i^t = y_{(x_i,t)}$ for visibility. Now we want to show the existence of a stochastic process $H : \mathcal{X} \times \mathcal{T} \times \Omega \to \bigcup_{t \in \mathcal{T}} \mathcal{Y}^t$, where the data $D = (X, Y)$ is generated. This can be done by showing the following conditions.

1. (Consistency) For any m such that $1 \le m < n$,
$$\int p(Y|X, C) dY_{m+1:n} = p(Y_{1:m}|X_{1:m}, C), \tag{18}$$
where $X_{i_1:i_2} = ((x_{i_1}, t_{i_1}), \cdots, (x_{i_2}, t_{i_2}))$ and $Y_{i_1:i_2} = (y_{(x_{i_1},t_{i_1})}, \cdots, y_{(x_{i_2},t_{i_2})})$.
2. (Exchangability) For any permutation π on $\mathcal{X}_{1:n} \times \mathcal{T}$,
$$p(\pi \circ Y | \pi \circ X, C) = p(Y|X, C), \tag{19}$$
where $\mathcal{X}_{1:n} = \{x_1, \cdots, x_n\}$, $\pi \circ X = (\pi((x_1, t_1)), \cdots, \pi((x_n, t_n)))$, and $\pi \circ Y = (y_{\pi((x_1,t_1))}, \cdots, y_{\pi((x_n,t_n))})$.

Here $p(Y|X, C) = \rho_X(Y|C)$ is the conditional distribution of Y given any context C. Note that C is conditioned since we are modeling functional *posterior* of h, rather than *prior*.

Now we provide the proof of Proposition 1, which states that the following generative model defines a stochastic process.

$$z \sim p(z|C),\ v^t \sim p(v^t|z, t, C),\ y_{(x_i,t)} \sim p(y_{(x_i,t)}|x_i, t, v^t)\ \forall t \in \mathcal{T},\ \forall x_i \in \mathcal{X}_{1:n}, \tag{20}$$

To show the conditions of Kolmogorov Extension Theorem, we need two assumptions on the data generating process (Eq. 20). First, we assume the distribution defined by the data generating process is finite so that the order of integral can be swapped. Also, we assume that the conditional distribution $p(y_{(x_i,t)}|x_i,t,v^t)$ can implicitly *select* the per-task latent variable v^t among $v^{1:T}$ using the given task index t, *i.e.,* there exists a distribution $\tilde{p}$ such that $\tilde{p}(y_{(x_i,t)}|x_i,t,v^{1:T}) = p(y_{(x_i,t)}|x_i,t,v^t)$. This means no more than that the latent variables $v^{1:T}$ are indeed *task-specific*, such that each v^t corresponds to task f^t. Note that our neural-network model of MTP (Figure 2) indeed satisfies the second assumption, since the decoder selects the corresponding per-task latent variable v^t given the task index $t \in \mathcal{T}$.

Proof. We first show the consistency condition. From the data generating process (Eq. 20),

$$\int p(Y|X,C)dY_{m+1:n} \tag{21}$$

$$= \int\int\int \left(\prod_{i=1}^{n} p(y_{(x_i,t_i)}|x_i,t_i,v^{t_i})\right)\left(\prod_{t=1}^{T} p(v^t|z,t,C)\right)p(z|C)dv^{1:T}dzdY_{m+1:n} \tag{22}$$

$$= \int\int \left(\prod_{i=1}^{m} p(y_{(x_i,t_i)}|x_i,t_i,v^{t_i})\right) \left(\int \prod_{i=m+1}^{n} p(y_{(x_i,t_i)}|x_i,t_i,v^{t_i})dY_{m+1:n}\right)\left(\prod_{t=1}^{T} p(v^t|z,t,C)\right)p(z|C)dv^{1:T}dz \tag{23}$$

$$= \int\int \left(\prod_{i=1}^{m} p(y_{(x_i,t_i)}|x_i,t_i,v^{t_i})\right)\left(\prod_{t=1}^{T} p(v^t|z,t,C)\right)p(z|C)dv^{1:T}dz \tag{24}$$

$$= p(Y_{1:m}|X_{1:m},C). \tag{25}$$

Next, we show the exchangability condition. Let π_1, π_2 be the values of first and second coordinate of π, such that $\pi((x_i,t_i)) = (\pi_1((x_i,t_i)), \pi_2((x_i,t_i)))$. Then

$$p(\pi \circ Y|\pi \circ X, C) \tag{26}$$

$$= \int\int \left(\prod_{i=1}^{n} \tilde{p}(y_{\pi((x_i,t_i))}|\pi((x_i,t_i)),v^{1:T})\right)\left(\prod_{t=1}^{T} p(v^t|z,t,C)\right)p(z|C)dv^{1:T}dz \tag{27}$$

$$= \int\int \left(\prod_{i=1}^{n} p(y_{\pi((x_i,t_i))}|\pi((x_i,t_i)),v^{\pi_2((x_i,t_i))})\right)\left(\prod_{t=1}^{T} p(v^t|z,t,C)\right)p(z|C)dv^{1:T}dz \tag{28}$$

$$= \int\int \left(\prod_{i=1}^{n} p(y_{(x_i,t_i)}|x_i,t_i,v^{t_i})\right)\left(\prod_{t=1}^{T} p(v^t|z,t,C)\right)p(z|C)dv^{1:T}dz \tag{29}$$

$$= p(Y|X,C). \tag{30}$$

Here we used the assumption about $p(y_{(x_i,t)}|x_i,t,v^t)$ such that $\tilde{p}(y_{\pi((x_i,t_i))}|\pi((x_i,t_i)),v^{1:T}) = p(y_{\pi((x_i,t_i))}|\pi((x_i,t_i)),v^{\pi_2((x_i,t_i))})$. Since $1 \leq m < n$ and π are arbitrarily chosen, the data generating process (Eq. 20) satisfies the conditions of the Kolmogorov Extension Theorem. Thus there exists a stochastic process $H : \mathcal{X} \times \mathcal{T} \times \Omega \to \bigcup_{t\in\mathcal{T}} \mathcal{Y}^t$, whose realizations are functions of the form $h : \mathcal{X} \times \mathcal{T} \to \bigcup_{t\in\mathcal{T}} \mathcal{Y}^t$. □

Note that the latent variable model of MTP (Eq. 5) is a special case of the data generating process (Eq. 20), where $p(v^t|z,t,C) = p(v^t|z,C^t)$. Thus MTP is a stochastic process over the functions of form $h : \mathcal{X} \times \mathcal{T} \to \bigcup_{t\in\mathcal{T}} \mathcal{Y}^t$.

A.3 ELBO DERIVATION FOR MTP

We first derive the exact evidence lower bound (ELBO) for $\log p_\theta(Y_D^{1:T}|X_D^{1:T}, C)$, and then approximate the intractable prior terms with variational posteriors. Recall that $C = (X_C^{1:T}, Y_C^{1:T}) = \bigcup_{t\in\mathcal{T}} C^t$ where $C^t = (X_C^t, Y_C^t) = \{(x_i^t, y_i^t)\}_{i\in\mathcal{I}(C^t)}$ and $D = (X_D^{1:T}, Y_D^{1:T}) = \bigcup_{t\in\mathcal{T}} D^t$ where $D^t = (X_D^t, Y_D^t) = \{(x_i^t, y_i^t)\}_{i\in\mathcal{I}(D^t)}$.

$$\log p_\theta(Y_D^{1:T}|X_D^{1:T}, C) \tag{31}$$

$$= \mathbb{E}_{q_\phi(z|D)}\Big[\log \frac{p_\theta(Y_D^{1:T}|X_D^{1:T}, C, z)p_\theta(z|X_D^{1:T}, C)}{p_\theta(z|D)}\Big] \tag{32}$$

$$= \mathbb{E}_{q_\phi(z|D)}\Big[\log \frac{p_\theta(Y_D^{1:T}|X_D^{1:T}, C, z)p_\theta(z|C)}{p_\theta(z|D)}\Big] \tag{33}$$

$$= \mathbb{E}_{q_\phi(z|D)}\Big[\log p_\theta(Y_D^{1:T}|X_D^{1:T}, C, z)\Big] + D_{\mathrm{KL}}\Big(q_\phi(z|D) \,||\, p_\theta(z|D)\Big) - D_{\mathrm{KL}}\Big(q_\phi(z|D) \,||\, p_\theta(z|C)\Big) \tag{34}$$

$$\geq \mathbb{E}_{q_\phi(z|D)}\Big[\log p_\theta(Y_D^{1:T}|X_D^{1:T}, C, z)\Big] - D_{\mathrm{KL}}\Big(q_\phi(z|D) \,||\, p_\theta(z|C)\Big), \tag{35}$$

where $p_\theta(Y_D^{1:T}|X_D^{1:T}, C, z)$ in Eq. 35 can be further expanded by

$$\log p_\theta(Y_D^{1:T}|X_D^{1:T}, C, z) \tag{36}$$

$$= \mathbb{E}_{\prod_{t=1}^T q_\phi(v^t|z, D^t)}\Big[\log \frac{p_\theta(Y_D^{1:T}|X_D^{1:T}, C, z, v^{1:T})p_\theta(v^{1:T}|X_D^{1:T}, C, z)}{p_\theta(v^{1:T}|z, D)}\Big] \tag{37}$$

$$= \mathbb{E}_{\prod_{t=1}^T q_\phi(v^t|z, D^t)}\Big[\sum_{t=1}^T \log \frac{p_\theta(Y_D^t|X_D^t, v^t)p_\theta(v^t|z, C^t)}{p_\theta(v^t|z, D^t)}\Big] \tag{38}$$

$$= \sum_{t=1}^T \mathbb{E}_{q_\phi}\Big[\log \frac{p_\theta(Y_D^t|X_D^t, v^t)p_\theta(v^t|z, C^t)}{p_\theta(v^t|z, D^t)}\Big] \tag{39}$$

$$= \sum_{t=1}^T \mathbb{E}_{q_\phi}\Big[\log p_\theta(Y_D^t|X_D^t, v^t)\Big] + D_{\mathrm{KL}}\Big(q_\phi(v^t|z, D^t) \,||\, p_\theta(v^t|z, D^t)\Big) - D_{\mathrm{KL}}\Big(q_\phi(v^t|z, D^t) \,||\, p_\theta(v^t|z, C^t)\Big) \tag{40}$$

$$\geq \sum_{t=1}^T \mathbb{E}_{q_\phi}\Big[\log p_\theta(Y_D^t|X_D^t, v^t)\Big] - D_{\mathrm{KL}}\Big(q_\phi(v^t|z, D^t) \,||\, p_\theta(v^t|z, C^t)\Big). \tag{41}$$

On Eq. 33 and Eq. 38, we use the conditional independence relation follows from the latent variable model (Eq. 5) By combining Eq. 35 and Eq. 41, and also by approximating $p_\theta(z|C)$ with $q_\phi(z|C)$ and $p_\theta(v^t|z, C^t)$ with $q_\phi(v^t|z, C^t)$, we get the following (approximate) lower bound.

$$\log p_\theta(Y_D^{1:T}|X_D^{1:T}, C) \tag{42}$$

$$\geq \mathbb{E}_{q_\phi}\Bigg[\sum_{t=1}^T \mathbb{E}_{q_\phi}\Big[\log p_\theta(Y_D^t|X_D^t, v^t)\Big] - D_{\mathrm{KL}}\Big(q_\phi(v^t|z, D^t) \,||\, p_\theta(v^t|z, C^t)\Big)\Bigg] - D_{\mathrm{KL}}\Big(q_\phi(z|D) \,||\, p_\theta(z|C)\Big) \tag{43}$$

$$\approx \mathbb{E}_{q_\phi}\Bigg[\sum_{t=1}^T \mathbb{E}_{q_\phi}\Big[\log p_\theta(Y_D^t|X_D^t, v^t)\Big] - D_{\mathrm{KL}}\Big(q_\phi(v^t|z, D^t) \,||\, q_\phi(v^t|z, C^t)\Big)\Bigg] - D_{\mathrm{KL}}\Big(q_\phi(z|D) \,||\, q_\phi(z|C)\Big). \tag{44}$$

A.4 ELBO FOR STP AND JTP

STP is no more than a collection of independent Neural Processes (NPs), where each NP corresponds to each task. Using T encoders and decoders $\{(p_{\theta_t}, q_{\phi_t})\}_{t=1}^{T}$, the objective for STP can be derived by summing up the NP objectives (Eq. 2). We omit the ELBO derivation for NP.

$$\log p_\theta(Y_D^{1:T}|X_D, C) \tag{45}$$

$$\geq \sum_{t=1}^{T} \mathbb{E}_{q_{\phi_t}(v^t|D^t)}\Big[\log p_{\theta_t}(Y_D^t|X_D, v^t)\Big] - D_{\mathrm{KL}}\Big(q_{\phi_t}(v^t|D^t) \,||\, p_{\theta_t}(v^t|C^t)\Big) \tag{46}$$

$$\approx \sum_{t=1}^{T} \mathbb{E}_{q_{\phi_t}(v^t|D^t)}\Big[\log p_{\theta_t}(Y_D^t|X_D, v^t)\Big] - D_{\mathrm{KL}}\Big(q_{\phi_t}(v^t|D^t) \,||\, q_{\phi_t}(v^t|C^t)\Big). \tag{47}$$

On the other hand, JTP is a single NP that models all tasks jointly, by concatenating the output variables into a single vector. Using an encoder q_ϕ and a decoder p_θ, the objective for JTP is the same as the NP objective.

$$\log p_\theta(Y_D^{1:T}|X_D, C) \tag{48}$$

$$\geq \mathbb{E}_{q_\phi(z|D)}\Big[\log p_\theta(Y_D^{1:T}|X_D, z)\Big] - D_{\mathrm{KL}}\Big(q_\phi(z|D) \,||\, p_\theta(z|C)\Big) \tag{49}$$

$$\approx \mathbb{E}_{q_\phi(z|D)}\Big[\log p_\theta(Y_D^{1:T}|X_D, z)\Big] - D_{\mathrm{KL}}\Big(q_\phi(z|D) \,||\, q_\phi(z|C)\Big). \tag{50}$$

Note that STP and JTP model functions with input space $\mathcal{X}$, so there is no superscript t in the input variables.

B ARCHITECTURAL AND TRAINING DETAILS

In this section, we provide architectural and training details about models used in the experiments (Section 5).

B.1 ATTENTIVE NEURAL PROCESS

As a strong NP baseline, we adopt Attentive Neural Processes (ANPs) (Kim et al., 2019) architecture for STP and JTP. The encoder of ANP consists of a latent path and a deterministic path, each computes a latent variable z and a deterministic representation r_i specific to each target example x_i. Then the decoder produces a distribution for the target output y_i, which is assumed to be Normal.

The general architecture of ANP follows Kim et al. (2019), while two major modifications have made as follows. First, we replace the average pooling operation in stochastic path by a Pooling by Multihead Attention (PMA) layer which is introduced in Lee et al. (2019). Next, since we have a categorical task (Segment) in 2D experiment, we modify the decoder for each Segment task to output logits for the Categorical distribution it models.

B.2 ANP MODEL FOR JTP

This section presents a detailed description of the JTP architecture used in the experiments. The JTP consists of a single ANP, which consists of a latent encoder, a deterministic encoder, and a decoder. Then JTP produces target output distribution by conditioning on the context set $C = \{(x_i, y_i^{1:T})\}_{i \in \mathcal{I}(C)}$ where $y_i^{1:T} = (y_i^1, \cdots, y_i^T)$.

Latent Encoder The latent encoder samples a global latent z. For each context example $(x_i, y_i^{1:T}) \in C$, we first project it to a hidden representation $s_i^{1:T} = \psi_s(x_i, y_i^{1:T})$ using a single MLP ψ_s. Then we aggregate them to a global representation s via self-attention followed by a pooling operation.

$$s = \textit{pool}(\textit{Attn}(\{s_i^{1:T}\}_{i \in \mathcal{I}(C)}, \{s_i^{1:T}\}_{i \in \mathcal{I}(C)}, \{s_i^{1:T}\}_{i \in \mathcal{I}(C)}). \tag{51}$$

Then the global latent is sampled via two MLPs.

$$z \sim \mathcal{N}(\psi_{(z,1)}(s), \psi_{(z,2)}(s)). \tag{52}$$

Deterministic Encoder The deterministic encoder produces local representation r_i for each $i \in \mathcal{D}$. We first project each context example $(x_i, y_i^{1:T}) \in C^t$ to a hidden representation $d_i^{1:T} = \psi_d(x_i, y_i^{1:T})$ that serves as value embedding in cross-attention. Then by using the context and target input x_i as key and query embeddings, we apply a cross-attention along the example axis (per-task).

$$\{r_i^{1:T}\}_{i \in \mathcal{I}(D)} = \textit{Attn}(\{x_i\}_{i \in \mathcal{I}(D)}, \{x_i\}_{i \in \mathcal{I}(C)}, \{d_i^{1:T}\}_{i \in \mathcal{I}(D)}). \tag{53}$$

Decoders Finally, the decoder produces predictive distribution for each joint target output $y_i^{1:T}$. We first project the input to $w_i = \psi_w(x_i)$, then concatenate it with the global latent variable z and deterministic representation $r_i^{1:T}$. To compute the output distribution, we first apply two MLPs on the triple $(w_i, z, r_i^{1:T})$.

$$\mu_i = \psi_{(y,1)}(w_i, z, r_i^{1:T}) \tag{54}$$

$$\sigma_i^2 = \psi_{(y,2)}(w_i, z, r_i^{1:T})). \tag{55}$$

Then for each dimension, we construct the predictive distributions as Normal or Categorical, depending on the corresponding task type.

$$y_i^t \sim \begin{cases} \mathcal{N}((\mu_i)_{\mathcal{Y}^t}, (\sigma_i^2)_{\mathcal{Y}^t}), \text{if } y_i^t \text{ is continuous,} \\ \text{Categorical}((\mu_i)_{\mathcal{Y}^t}), \text{if } y_i^t \text{ is discrete,} \end{cases} \tag{56}$$

where $(\mu_i)_{\mathcal{Y}^t}$ (or $(\sigma_i^2)_{\mathcal{Y}^t}$)) denotes the projection of μ_i (or σ_i^2) into the task-specific output space $\mathcal{Y}^t$, by indexing the corresponding dimension from μ_i. For example, if all tasks are one-dimensional, then this corresponds to selecting t-th coordinate of μ_i.

B.3 ANP MODEL FOR STP

This section presents a detailed description of the STP architecture used in the experiments. The STP consists of T independent ANPs, which consists of T latent encoders, T deterministic encoders, and T decoders. Then STP produces target output distribution by conditioning on the context set $C = \bigcup_{t \in \mathcal{T}} C^t$ where $C^t = \{(x_i, y_i^t)\}_{t \in \mathcal{T}}$. In the following, we deonte a stacked multi-head attention block (Parmar et al., 2018) by $\textit{Attn}(Q, K, V)$ and a MLP by $\psi(x)$, as in Section 3.3.

Latent Encoders The latent encoders sample per-task latents $v^{1:T} = (v^1, \cdots, v^T)$. For each context example $(x_i, y_i^t) \in C^t$, we first project it to a hidden representation $s_i^t = \psi_s^t(x_i, y_i^t)$ using a MLP ψ_s^t specific to task f^t. Then we aggregate them to a task-specific representation s^t via self-attention followed by a pooling operation.

$$s^t = \textit{pool}(\textit{Attn}(\{s_i^t\}_{i \in \mathcal{I}(C^t)}, \{s_i^t\}_{i \in \mathcal{I}(C^t)}, \{s_i^t\}_{i \in \mathcal{I}(C^t)})), \quad \forall t \in \mathcal{T}. \tag{57}$$

Note that each attention is applied along the example axis (per-task) and independent to each task. Then the per-task latent variables are sampled independently, via MLPs.

$$v^t \sim \mathcal{N}(\psi_{(v^t,1)}(s^t), \psi_{(v^t,2)}(s^t)), \quad \forall t \in \mathcal{T}. \tag{58}$$

Deterministic Encoders The deterministic encoders produce local representation r_i^t for each $i \in \mathcal{D}$ and $t \in \mathcal{T}$. We first project each context example $(x_i, y_i^t) \in C^t$ to a hidden representation $d_i^t = \psi_d^t(x_i, y_i^t)$ that serves as value embedding in cross-attention. Then by using the context and target input x_i as key and query embeddings, we apply T independent cross-attention along the example axis (per-task).

$$\{r_i^t\}_{i \in \mathcal{I}(D^t)} = \textit{Attn}(\{x_i^t\}_{i \in \mathcal{I}(D^t)}, \{x_i^t\}_{i \in \mathcal{I}(C^t)}, \{d_i^t\}_{i \in \mathcal{I}(D^t)}), \quad \forall t \in \mathcal{T}. \tag{59}$$

Decoders Finally, the decoders produce predictive distributions for each target output y_i^t. We first project the input to $w_i^t = \psi_w^t(x_i)$, then concatenate it with the corresponding latent variable v^t and deterministic representation r_i^t. The output distribution is computed similar to MTP described in Section 3.3.

$$y_i^t \sim \begin{cases} \mathcal{N}(\psi_{(y^t,1)}(w_i^t, v^t, r_i^t), \psi_{(y^t,2)}(w_i^t, v^t, r_i^t)), & \text{if } y_i^t \text{ is continuous,} \\ \text{Categorical}(\psi_{(y^t,1)}(w_i^t, v^t, r_i^t)), & \text{if } y_i^t \text{ is discrete,} \end{cases} \quad \forall i \in \mathcal{I}(D^t), \forall t \in \mathcal{T}. \tag{60}$$

We use the hidden dimension $d = 128$ for all models in synthetic and CelebA experiments and $d = 64$ and in weather experiments.

B.4 ARCHITECTURAL HYPER-PARAMETERS

Table 5 summarizes the number of layers for each module in three models (STP, JTP, MTP) used in the experiments. We use the same number of layers in all experiments.

Module	STP	JTP	MTP
ψ_s (or ψ_s^t)	3	3	3
ψ_d (or ψ_d^t)	3	3	3
ψ_w (or ψ_w^t)	1	1	1
ψ_y (or ψ_y^t)	5	5	5
per-task *Attn* (in STP, MTP)	3	-	3
global *Attn* (in JTP)	-	3	-
across-task *Attn* (in MTP)	-	-	2

Table 5: Number of layers of each module in STP, JTP, and MTP used in the experiments.

Table 6 summarizes the hidden dimension dim_{hidden} of all models used in each experiment. In our implementation, all layers (including the positional embedding) except the input and output layers have the dimension dim_{hidden}.

Dataset	STP	JTP	MTP
Synthetic	128	128	128
Weather	64	64	64
Face	128	128	128

Table 6: Hidden dimensions of the models used in the experiments.

B.5 Training Details

For all three models, we schedule learning rate lr by

$$\text{lr} = \text{base_lr} \times 1000^{0.5} \times \min(\text{n_iters} \times 1,000^{-1.5}, \text{n_iters}^{-0.5}), \tag{61}$$

where n_iters is the number of total iterations and base_lr is the base learning rate. We also introduce *beta* coefficient on the ELBO objective following Higgins et al. (2017), which is multiplied by each KL term. The beta coefficient is scheduled to be linearly increased from 0 to 1 during the first 10000 iters, then fixed to 1. We summarize the training hyper-parameters of models used in the experiments in Table 7.

Dataset	n_iters	base_lr	batch size
Synthetic	300000	0.00025	24
Weather	50000	0.00025	16
Face	300000	0.0005	16

Table 7: Training hyper-parameters used in the experiments.

B.6 Parameter Sharing in MTP

The overall description of our neural network model for MTP is provided in Section 3.3. We use different parameter sharing techniques in the datasets, depending on whether the tasks are homogeneous or not. In synthetic and weather tasks, all output values are one-dimensional. Thus we tie the parameters of the per-task paths in encoder and decoder, which makes more efficient parametrization compared to per-task encoders and decoders. In visual tasks, however, the tasks have different output dimensionalities. Thus in this case, we separate the parameters of all per-task paths.

C EXPERIMENTAL DETAILS OF 1D SYNTHETIC FUNCTION REGRESSION

In this section, we describe details of the data generating process and experimental settings of 1D function regression on synthetic tasks.

C.1 SYNTHETIC DATASET

As discussed in the paper, we simulate synthetic tasks which are correlated by a set of parameters $a, b, c, w \in \mathbb{R}$ as follow:

$$f^t(x_i) = a \cdot \text{act}_t(wx_i + b) + c, \quad \text{act}_t \in \{\text{Sine}, \text{Tanh}, \text{Sigmoid}, \text{Gaussian}\}, \tag{62}$$

where Sine, Tanh, Sigmoid are sine, hyperbolic tangent, logistic sigmoid function, respectively, and Gaussian(x) is defined as $\exp(-x^2)$. Rather than sharing the exactly same parameters a, b, c, w across tasks, we add a task-specific noise to each parameter, to control the amount of correlation across tasks as follow:

$$\alpha^t = \alpha + \epsilon, \ \epsilon \sim \mathcal{N}(0, 0.1), \ \forall t \in \mathcal{T}, \ \forall \alpha \in \{a, b, c, w\}. \tag{63}$$

Thus in fact the input-output pairs of each task is generated as follow:

$$f^t(x_i) = a^t \cdot \text{act}_t(w^t x_i + b^t) + c^t, \quad \text{act}_t \in \{\text{Sine}, \text{Tanh}, \text{Sigmoid}, \text{Gaussian}\}, \tag{64}$$

We split the 1,000 functions into 800 training, 100 validation, and 100 test sets of four correlated tasks. Then we construct a training dataset, a validataion dataset, and a test dataset using the corresponding set of generated tasks. For each training and validation data, we sample 200 input points uniformly within the interval $[-5, 5]$, and applied the corresponding tasks to generate multi-task output values. For each test data, we choose 1000 input points in the uniform grid of the interval $[-5, 5]$, and generate the multi-task output values similarly. Finally, simulating the incomplete data is achieved by randomly dropping each output value y_i^t with probability $\gamma \in [0, 1]$.

C.2 EVALUATION PROTOCOL

For evaluation, we average the normalized MSE $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i^t - \hat{y}_i^t)^2/a^2$ on test dataset. [2] For prediction $\hat{Y}^{1:4}$, we approximate the predictive posterior mean with Monte Carlo sampling. For example in MTP,

$$p(y_i^t|x_i, C) = \int\int p(y_i^t|x_i, v^t)p(v^t|z, C^t)p(z|C)dv^t dz \tag{65}$$

$$\approx \frac{1}{NM}\sum_{k=1}^{N}\sum_{l=1}^{M} p(y_i^t|x_i, v_{k,l}^t), \text{ where } v_{k,l}^t \overset{\text{i.i.d.}}{\sim} p(v^t|z_k, C^t), \ z_k \overset{\text{i.i.d.}}{\sim} p(z|C). \tag{66}$$

We use $N = M = 5$, resulting total 25 samples. For STP (or JTP), we sample each latent v^t (or z) 5 times, since there is no hierarchy. Since all the output distributions are Gaussian, the posterior predictive mean can be computed by averaging the means of each sample distribution $p(y_i^t|x_i, v_{k,l}^t)$. To plot the predictions in Figure 3 (a), we use the posterior means for both z and v^t (which corresponds to the Maximum A Posteriori estimation) and plot the mean and variance of resulting $p(y_i^t|x_i, v^t)$.

[2] We normalize the MSE for fair consideration of difference in amplitude a across different functions.

D ADDITIONAL RESULTS ON SYNTHETIC 1D FUNCTION REGRESSION

In this section, we provide additional results on the synthetic experiment, with various missing rates γ and also with standard deviation from 5 different random seeds. When the data is incomplete and missing some task labels (i.e., $\gamma = 0.25, 0.5, 0.75$), we can see that MTP clearly outperforms the GP and NP baselines in all cases. When the complete data ($\gamma = 0$) is given, MTP still outperforms the GP baselines and STP while achieves at least competitive performance to JTP.

Table 8: Average normalized MSE on synthetic tasks, with varying context size (m) and $\gamma = 0$.

task	Sine			Tanh		
m	5	10	20	5	10	20
MOSM	1.5131 ± 0.0414	0.6810 ± 0.0832	0.7692 ± 0.1041	1.6206 ± 0.1389	0.6897 ± 0.0807	0.7721 ± 0.1283
CSM	0.8775 ± 0.1218	1.0008 ± 0.1636	0.7400 ± 0.2514	0.7010 ± 0.0950	0.8156 ± 0.1547	0.6681 ± 0.1975
STP	0.2562 ± 0.0114	0.0910 ± 0.0104	0.0200 ± 0.0020	0.0502 ± 0.0071	0.0104 ± 0.0023	0.0024 ± 0.0004
JTP	**0.1213** ± 0.0095	**0.0560** ± 0.0039	0.0291 ± 0.0011	**0.0210** ± 0.0038	0.0079 ± 0.0004	0.0057 ± 0.0004
MTP	0.1793 ± 0.0191	0.0705 ± 0.0063	**0.0186** ± 0.0027	0.0287 ± 0.0027	**0.0060** ± 0.0017	**0.0015** ± 0.0002

task	Sigmoid			Gaussian		
m	5	10	20	5	10	20
MOSM	1.5004 ± 0.1248	0.5752 ± 0.0706	0.9066 ± 0.1321	1.3364 ± 0.1227	0.5557 ± 0.0486	0.6479 ± 0.1213
CSM	0.5255 ± 0.0792	0.6542 ± 0.1953	0.5337 ± 0.2881	0.5653 ± 0.0829	0.6234 ± 0.1528	0.5759 ± 0.2136
STP	0.0064 ± 0.0029	0.0017 ± 0.0002	0.0008 ± 0.0000	0.0393 ± 0.0048	0.0179 ± 0.0030	0.0071 ± 0.0006
JTP	0.0041 ± 0.0008	0.0020 ± 0.0002	0.0016 ± 0.0001	**0.0170** ± 0.0015	0.0093 ± 0.0007	0.0072 ± 0.0004
MTP	**0.0030** ± 0.0010	**0.0006** ± 0.0001	**0.0003** ± 0.0000	0.0197 ± 0.0008	**0.0092** ± 0.0015	**0.0043** ± 0.0009

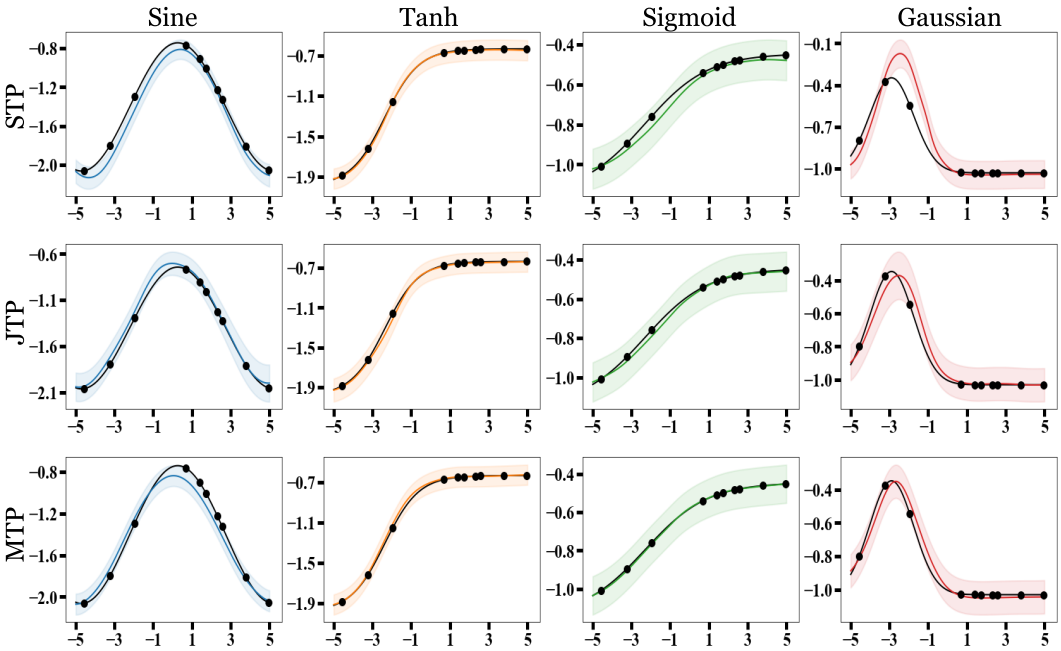


Figure 6: Predicted mean and variance in synthetic task with $\gamma = 0.$, $m = 10$.

Table 9: Average normalized MSE on synthetic tasks, with varying context size (m) and $\gamma = 0.25$.

task	Sine			Tanh		
m	5	10	20	5	10	20
MOSM	1.6304 ± 0.0467	0.5118 ± 0.0379	0.6681 ± 0.0714	1.7275 ± 0.0770	0.4180 ± 0.0747	0.5895 ± 0.1061
CSM	0.7882 ± 0.1668	0.8919 ± 0.1347	0.8504 ± 0.2052	0.5321 ± 0.1508	0.6421 ± 0.1172	0.6216 ± 0.0911
STP	0.3768 ± 0.0152	0.1547 ± 0.0145	0.0492 ± 0.0060	0.0711 ± 0.0077	0.0191 ± 0.0033	0.0070 ± 0.0013
STP+JTP	0.2906 ± 0.0241	0.1481 ± 0.0049	0.0738 ± 0.0066	0.0531 ± 0.0073	0.0169 ± 0.0008	0.0098 ± 0.0004
MTP	**0.1871** ± 0.0211	**0.0705** ± 0.0027	**0.0297** ± 0.0026	**0.0300** ± 0.0029	**0.0055** ± 0.0011	**0.0023** ± 0.0002

task	Sigmoid			Gaussian		
m	5	10	20	5	10	20
MOSM	1.5174 ± 0.0930	0.3083 ± 0.0627	0.5442 ± 0.1087	1.3788 ± 0.0507	0.3778 ± 0.0857	0.5537 ± 0.0726
CSM	0.4230 ± 0.0903	0.5258 ± 0.1759	0.4556 ± 0.1018	0.4600 ± 0.1391	0.5341 ± 0.0995	0.5566 ± 0.2494
STP	0.0121 ± 0.0038	0.0036 ± 0.0004	0.0013 ± 0.0001	0.0532 ± 0.0072	0.0260 ± 0.0037	0.0150 ± 0.0021
STP+JTP	0.0097 ± 0.0017	0.0038 ± 0.0002	0.0022 ± 0.0001	0.0403 ± 0.0055	0.0181 ± 0.0005	0.0112 ± 0.0006
MTP	**0.0040** ± 0.0017	**0.0008** ± 0.0001	**0.0004** ± 0.0000	**0.0234** ± 0.0025	**0.0087** ± 0.0012	**0.0048** ± 0.0003

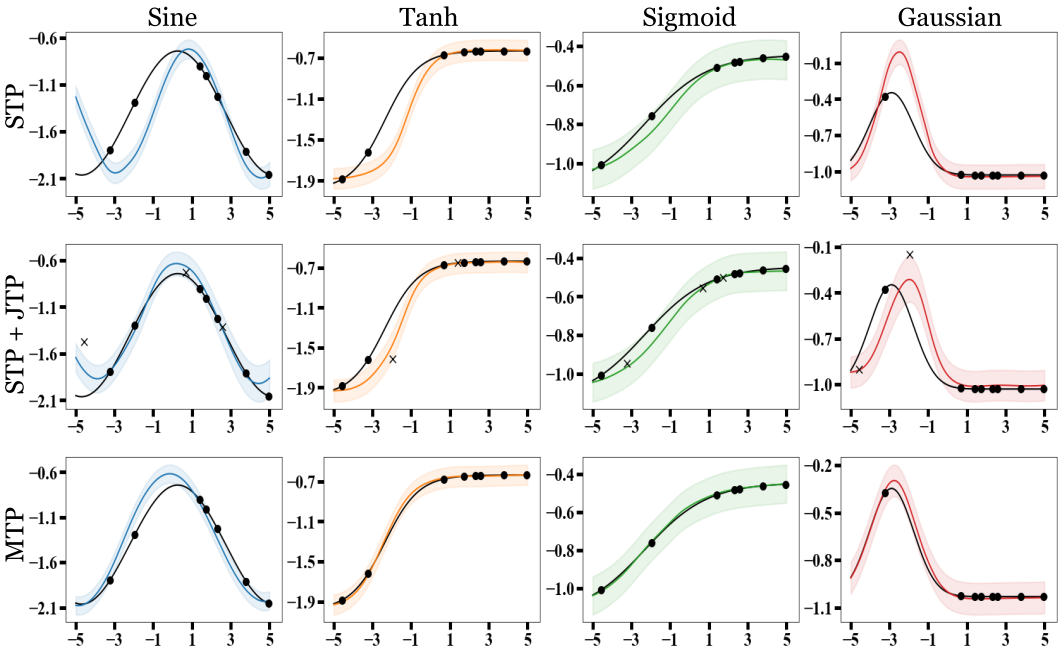


Figure 7: Predicted mean and variance in synthetic task with $\gamma = 0.25$, $m = 10$.

Table 10: Average normalized MSE on synthetic tasks, with varying context size (m) and $\gamma = 0.5$.

task	Sine			Tanh		
m	5	10	20	5	10	20
MOSM	1.8653 ± 0.1052	0.7057 ± 0.0795	0.4452 ± 0.0844	1.9980 ± 0.1267	0.6220 ± 0.0751	0.3625 ± 0.1290
CSM	0.5516 ± 0.1280	0.6724 ± 0.0985	0.6975 ± 0.1878	0.3580 ± 0.1220	0.4445 ± 0.0172	0.5134 ± 0.1358
STP	0.5212 ± 0.0157	0.2609 ± 0.0382	0.0993 ± 0.0182	0.1307 ± 0.0134	0.0468 ± 0.0074	0.0159 ± 0.0028
STP+JTP	0.3848 ± 0.0203	0.2340 ± 0.0169	0.1114 ± 0.0084	0.1015 ± 0.0160	0.0418 ± 0.0066	0.0168 ± 0.0026
MTP	**0.2636** ± 0.0105	**0.1137** ± 0.0078	**0.0485** ± 0.0034	**0.0435** ± 0.0047	**0.0115** ± 0.0021	**0.0040** ± 0.0002

task	Sigmoid			Gaussian		
m	5	10	20	5	10	20
MOSM	1.6795 ± 0.1223	0.3758 ± 0.1016	0.1769 ± 0.0397	1.5285 ± 0.0759	0.4438 ± 0.0541	0.2045 ± 0.0599
CSM	0.2584 ± 0.0975	0.3003 ± 0.0477	0.3821 ± 0.1419	0.2992 ± 0.0753	0.3345 ± 0.0197	0.4240 ± 0.1218
STP	0.0203 ± 0.0034	0.0067 ± 0.0013	0.0025 ± 0.0005	0.0799 ± 0.0098	0.0409 ± 0.0041	0.0222 ± 0.0042
STP+JTP	0.0163 ± 0.0024	0.0065 ± 0.0015	0.0032 ± 0.0004	0.0613 ± 0.0045	0.0318 ± 0.0021	0.0161 ± 0.0019
MTP	**0.0066** ± 0.0019	**0.0014** ± 0.0001	**0.0006** ± 0.0001	**0.0360** ± 0.0018	**0.0132** ± 0.0008	**0.0069** ± 0.0012

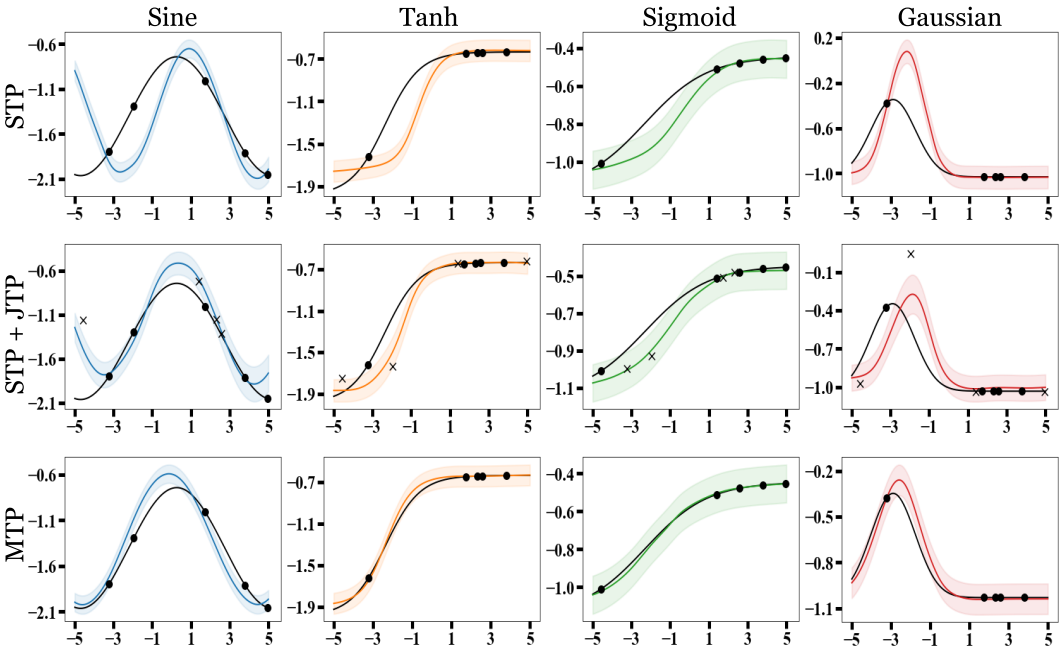


Figure 8: Predicted mean and variance in synthetic task with $\gamma = 0.5$, $m = 10$.

Table 11: Average normalized MSE on synthetic tasks, with varying context size (m) and $\gamma = 0.75$.

task	Sine			Tanh		
m	5	10	20	5	10	20
MOSM	2.0538 ± 0.0598	1.2807 ± 0.0901	0.6749 ± 0.1324	2.3614 ± 0.0903	1.2357 ± 0.1575	0.4068 ± 0.0606
CSM	0.6939 ± 0.1528	0.7677 ± 0.1330	0.8041 ± 0.2094	0.4799 ± 0.1389	0.5079 ± 0.0866	0.5599 ± 0.1177
STP	0.7329 ± 0.0581	0.5053 ± 0.0289	0.2770 ± 0.0286	0.1975 ± 0.0256	0.1128 ± 0.0111	0.0443 ± 0.0116
STP+JTP	0.5807 ± 0.0573	0.4115 ± 0.0348	0.2521 ± 0.0218	0.1654 ± 0.0195	0.0989 ± 0.0127	0.0426 ± 0.0115
MTP	**0.3784** ± 0.0395	**0.2432** ± 0.0230	**0.1295** ± 0.0172	**0.0838** ± 0.0085	**0.0340** ± 0.0020	**0.0118** ± 0.0027

task	Sigmoid			Gaussian		
m	5	10	20	5	10	20
MOSM	2.0146 ± 0.0898	1.0354 ± 0.1764	0.1531 ± 0.0690	1.8051 ± 0.0645	1.0470 ± 0.1464	0.4137 ± 0.1132
CSM	0.3421 ± 0.0761	0.3531 ± 0.0358	0.4466 ± 0.1325	0.3624 ± 0.0823	0.4000 ± 0.0411	0.4790 ± 0.1122
STP	0.0303 ± 0.0038	0.0191 ± 0.0030	0.0067 ± 0.0008	0.1232 ± 0.0037	0.0800 ± 0.0094	0.0469 ± 0.0037
STP+JTP	0.0260 ± 0.0044	0.0165 ± 0.0021	0.0070 ± 0.0005	0.0970 ± 0.0101	0.0608 ± 0.0035	0.0334 ± 0.0023
MTP	**0.0136** ± 0.0024	**0.0059** ± 0.0004	**0.0019** ± 0.0004	**0.0643** ± 0.0048	**0.0323** ± 0.0016	**0.0155** ± 0.0018

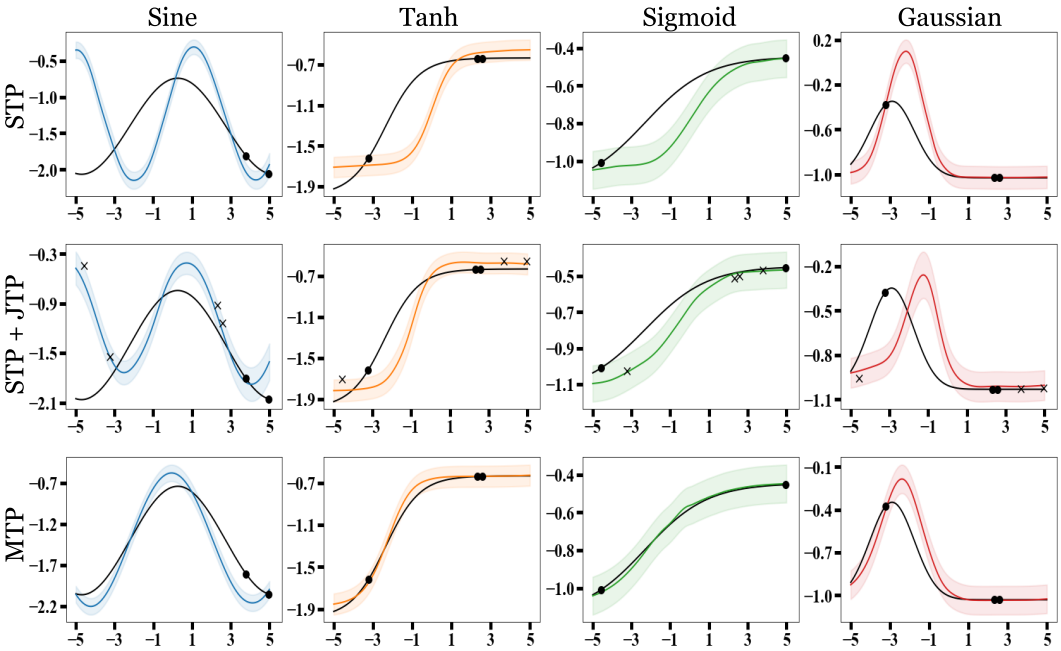


Figure 9: Predicted mean and variance in synthetic task with $\gamma = 0.75$, $m = 10$.

E ADDITIONAL RESULTS ON WEATHER TIME-SERIES REGRESSION

In this section, we provide additional results on the time-series regression experiment on weather data, with various missing rates γ and also with standard deviation from 5 different random seeds. In this section, we provide additional results on the synthetic experiment, with various missing rates γ and also with standard deviation from 5 different random seeds. When the data is highly incomplete (i.e., $\gamma = 0.5, 0.75$), we can see that MTP clearly outperforms the GP and NP baselines in all cases. When the complete or less incomplete data ($\gamma = 0, 0.25$) is given, MTP still outperforms the GP baselines while achieves at least competitive performance to STP and JTP.

Table 12: Average MSE and NLL on weather tasks, with $m = 10$ and $\gamma = 0$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0305 ± 0.0131	-0.4666 ± 0.2579	0.0339 ± 0.0106	-0.3287 ± 0.2586	0.1220 ± 0.0150	1.3475 ± 0.3959
CSM	0.0230 ± 0.0050	-0.6762 ± 0.0777	0.0291 ± 0.0071	-0.4947 ± 0.1303	0.1105 ± 0.0145	0.8071 ± 0.1993
STP	**0.0035** ± 0.0002	-1.2081 ± 0.0111	0.0053 ± 0.0002	-1.1229 ± 0.0075	0.0495 ± 0.0015	-0.1586 ± 0.0617
JTP	0.0036 ± 0.0001	-1.1839 ± 0.0095	0.0055 ± 0.0002	-1.0877 ± 0.0213	**0.0476** ± 0.0014	-0.2180 ± 0.0202
MTP	**0.0035** ± 0.0002	**-1.2139** ± 0.0121	**0.0051** ± 0.0001	**-1.1314** ± 0.0049	0.0495 ± 0.0007	**-0.2250** ± 0.0435

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3967 ± 0.0208	4.2690 ± 0.3649	0.4265 ± 0.0360	4.3886 ± 0.5976	0.0341 ± 0.0101	-0.2682 ± 0.2585
CSM	0.3555 ± 0.0148	2.3356 ± 0.2325	0.3688 ± 0.0230	2.1598 ± 0.3086	0.0280 ± 0.0046	-0.4979 ± 0.0923
STP	0.2436 ± 0.0044	1.5858 ± 0.1445	0.2360 ± 0.0013	0.7520 ± 0.0279	0.0075 ± 0.0004	-1.0310 ± 0.0164
JTP	0.2233 ± 0.0032	**0.6078** ± 0.0399	**0.2175** ± 0.0024	**0.6383** ± 0.0113	**0.0071** ± 0.0002	-1.0266 ± 0.0125
MTP	**0.2241** ± 0.0016	0.6920 ± 0.0614	**0.2175** ± 0.0050	0.6869 ± 0.0894	0.0072 ± 0.0003	**-1.0564** ± 0.0062

Table 13: Average MSE and NLL on weather tasks, with $m = 10$ and $\gamma = 0.25$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0190 ± 0.0044	-0.7163 ± 0.0700	0.0287 ± 0.0042	-0.4994 ± 0.1296	0.1044 ± 0.0090	0.8660 ± 0.1461
CSM	0.0238 ± 0.0026	-0.6402 ± 0.0910	0.0262 ± 0.0074	-0.5493 ± 0.1259	0.1133 ± 0.0256	0.8675 ± 0.3159
STP	0.0038 ± 0.0002	-1.1942 ± 0.0087	0.0057 ± 0.0003	-1.1026 ± 0.0117	0.0513 ± 0.0032	-0.1307 ± 0.0674
STP+JTP	0.0040 ± 0.0001	-1.1630 ± 0.0041	0.0062 ± 0.0004	-1.0521 ± 0.0178	0.0517 ± 0.0009	-0.1431 ± 0.0308
MTP	**0.0036** ± 0.0002	**-1.2056** ± 0.0076	**0.0055** ± 0.0001	**-1.1155** ± 0.0083	**0.0506** ± 0.0024	**-0.2100** ± 0.0392

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3576 ± 0.0150	3.6337 ± 0.7758	0.3862 ± 0.0334	3.6937 ± 0.7993	0.0225 ± 0.0054	-0.5552 ± 0.0994
CSM	0.3774 ± 0.0427	2.9056 ± 0.0950	0.4027 ± 0.0283	2.6626 ± 0.1525	0.0248 ± 0.0035	-0.5302 ± 0.0859
STP	0.2504 ± 0.0060	1.2501 ± 0.1382	0.2428 ± 0.0112	0.7505 ± 0.0252	0.0081 ± 0.0006	-1.0027 ± 0.0241
STP+JTP	0.2313 ± 0.0028	**0.6640** ± 0.0357	0.2244 ± 0.0052	**0.6778** ± 0.0184	0.0076 ± 0.0002	-0.9988 ± 0.0168
MTP	**0.2256** ± 0.0034	0.6830 ± 0.0804	**0.2206** ± 0.0071	0.6870 ± 0.0752	**0.0075** ± 0.0005	**-1.0486** ± 0.0093

Table 14: Average MSE and NLL on weather tasks, with $m = 10$ and $\gamma = 0.5$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0188 ± 0.0063	-0.6842 ± 0.1643	0.0274 ± 0.0036	-0.5108 ± 0.1097	0.1082 ± 0.0169	0.8513 ± 0.3009
CSM	0.0299 ± 0.0054	-0.4795 ± 0.1467	0.0407 ± 0.0108	-0.1961 ± 0.2804	0.1387 ± 0.0163	1.4324 ± 0.1959
STP	0.0052 ± 0.0007	-1.1281 ± 0.0361	0.0071 ± 0.0004	-1.0362 ± 0.0181	0.0622 ± 0.0061	0.0980 ± 0.1243
STP+JTP	0.0045 ± 0.0003	-1.1435 ± 0.0102	0.0070 ± 0.0003	-1.0161 ± 0.0160	0.0610 ± 0.0072	-0.0351 ± 0.0737
MTP	**0.0040** ± 0.0002	**-1.1884** ± 0.0071	**0.0059** ± 0.0004	**-1.0912** ± 0.0158	**0.0553** ± 0.0044	**-0.1267** ± 0.0745

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3693 ± 0.0311	3.5583 ± 1.3957	0.3657 ± 0.0093	3.2127 ± 1.2921	0.0185 ± 0.0033	-0.6240 ± 0.1243
CSM	0.4016 ± 0.0196	3.3915 ± 0.3079	0.4224 ± 0.0375	3.2398 ± 0.5735	0.0360 ± 0.0103	-0.2418 ± 0.2946
STP	0.2652 ± 0.0058	0.9767 ± 0.1743	0.2617 ± 0.0031	0.7814 ± 0.0401	0.0089 ± 0.0008	-0.9634 ± 0.0400
STP+JTP	0.2400 ± 0.0029	0.6877 ± 0.0382	0.2400 ± 0.0073	0.7049 ± 0.0155	0.0086 ± 0.0007	-0.9625 ± 0.0225
MTP	**0.2313** ± 0.0023	**0.6560** ± 0.0686	**0.2265** ± 0.0054	**0.6890** ± 0.0519	**0.0077** ± 0.0004	**-1.0239** ± 0.0192

Table 15: Average MSE and NLL on weather tasks, with $m = 10$ and $\gamma = 0.75$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0195 ± 0.0049	-0.7027 ± 0.0750	0.0366 ± 0.0052	-0.3276 ± 0.0683	0.1085 ± 0.0202	0.7282 ± 0.2360
CSM	0.0392 ± 0.0101	-0.2885 ± 0.1752	0.0600 ± 0.0083	0.3251 ± 0.1818	0.1459 ± 0.0190	1.6520 ± 0.2919
STP	0.0081 ± 0.0012	-0.9859 ± 0.0593	0.0108 ± 0.0027	-0.8553 ± 0.1239	0.0759 ± 0.0109	0.2644 ± 0.0924
STP+JTP	0.0071 ± 0.0013	-1.0397 ± 0.0459	0.0106 ± 0.0021	-0.8705 ± 0.1327	0.0781 ± 0.0076	0.1515 ± 0.0769
MTP	**0.0053** ± 0.0004	**-1.1280** ± 0.0184	**0.0080** ± 0.0008	**-0.9938** ± 0.0424	**0.0645** ± 0.0025	**-0.0390** ± 0.0316

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3837 ± 0.0389	2.8610 ± 0.3129	0.3984 ± 0.0119	2.4205 ± 0.2328	0.0248 ± 0.0027	-0.5116 ± 0.0439
CSM	0.4207 ± 0.0576	4.0535 ± 0.5513	0.4490 ± 0.0314	3.5539 ± 0.5012	0.0499 ± 0.0085	0.1558 ± 0.1975
STP	0.2937 ± 0.0154	1.0656 ± 0.1474	0.2928 ± 0.0176	0.8199 ± 0.0396	0.0127 ± 0.0013	-0.7837 ± 0.0698
STP+JTP	0.2625 ± 0.0107	0.7416 ± 0.0321	0.2642 ± 0.0085	0.7507 ± 0.0167	0.0110 ± 0.0017	-0.8753 ± 0.0689
MTP	**0.2455** ± 0.0073	**0.6429** ± 0.0473	**0.2407** ± 0.0042	**0.6977** ± 0.0214	**0.0090** ± 0.0003	**-0.9640** ± 0.0206

Table 16: Average MSE and NLL on weather tasks, with $m = 20$ and $\gamma = 0$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0370 ± 0.0078	-0.5514 ± 0.0719	0.0338 ± 0.0041	-0.4498 ± 0.0827	0.1024 ± 0.0235	0.5679 ± 0.1783
CSM	0.0150 ± 0.0035	-0.8343 ± 0.0482	0.0176 ± 0.0038	-0.7398 ± 0.0647	0.0772 ± 0.0063	0.1726 ± 0.1020
STP	**0.0031** ± 0.0001	-1.2276 ± 0.0048	**0.0046** ± 0.0001	-1.1547 ± 0.0055	**0.0435** ± 0.0014	-0.2927 ± 0.0348
STP+JTP	0.0033 ± 0.0001	-1.2037 ± 0.0087	0.0049 ± 0.0002	-1.1224 ± 0.0116	0.0442 ± 0.0008	-0.2752 ± 0.0069
MTP	0.0032 ± 0.0001	**-1.2280** ± 0.0013	**0.0046** ± 0.0001	**-1.1564** ± 0.0050	0.0453 ± 0.0011	**-0.3008** ± 0.0200

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3726 ± 0.0246	2.5010 ± 0.2586	0.3964 ± 0.0177	2.4315 ± 0.2607	0.0326 ± 0.0050	-0.4808 ± 0.0597
CSM	0.2999 ± 0.0102	1.2725 ± 0.1057	0.3027 ± 0.0168	1.1891 ± 0.1053	0.0198 ± 0.0054	-0.6957 ± 0.0726
STP	0.2242 ± 0.0040	0.7904 ± 0.0828	0.2143 ± 0.0032	0.6173 ± 0.0115	**0.0064** ± 0.0002	-1.0782 ± 0.0108
STP+JTP	0.2152 ± 0.0025	**0.5396** ± 0.0151	**0.2075** ± 0.0029	0.6090 ± 0.0052	0.0067 ± 0.0001	-1.0489 ± 0.0085
MTP	**0.2144** ± 0.0018	b0.5601 ± 0.0457	0.2092 ± 0.0025	**0.6039** ± 0.0069	0.0067 ± 0.0002	**-1.0810** ± 0.0053

Table 17: Average MSE and NLL on weather tasks, with $m = 20$ and $\gamma = 0.25$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0238 ± 0.0091	-0.7257 ± 0.0913	0.0293 ± 0.0039	-0.5360 ± 0.0861	0.0952 ± 0.0139	0.4975 ± 0.1471
CSM	0.0167 ± 0.0032	-0.8198 ± 0.0477	0.0181 ± 0.0044	-0.7441 ± 0.0776	0.0780 ± 0.0115	0.2415 ± 0.1726
STP	0.0034 ± 0.0002	-1.2173 ± 0.0082	0.0048 ± 0.0001	-1.1467 ± 0.0070	0.0454 ± 0.0014	-0.2683 ± 0.0391
STP+JTP	0.0036 ± 0.0001	-1.1924 ± 0.0065	0.0054 ± 0.0003	-1.1042 ± 0.0133	0.0481 ± 0.0008	-0.2208 ± 0.0212
MTP	**0.0033** ± 0.0001	**-1.2219** ± 0.0031	**0.0047** ± 0.0002	**-1.1490** ± 0.0074	**0.0464** ± 0.0006	**-0.2920** ± 0.0264

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3762 ± 0.0219	2.8770 ± 0.1718	0.3861 ± 0.0213	2.8236 ± 0.1137	0.0294 ± 0.0077	-0.5362 ± 0.1413
CSM	0.3294 ± 0.0144	1.5523 ± 0.1850	0.3230 ± 0.0161	1.5174 ± 0.1625	0.0183 ± 0.0051	-0.7241 ± 0.0844
STP	0.2271 ± 0.0045	0.6554 ± 0.0801	0.2223 ± 0.0027	0.6285 ± 0.0154	**0.0067** ± 0.0001	-1.0686 ± 0.0090
STP+JTP	0.2231 ± 0.0023	0.5946 ± 0.0219	0.2173 ± 0.0036	0.6379 ± 0.0067	0.0073 ± 0.0002	-1.0198 ± 0.0122
MTP	**0.2158** ± 0.0012	**0.5626** ± 0.0420	**0.2104** ± 0.0039	**0.6060** ± 0.0180	0.0069 ± 0.0003	**-1.0749** ± 0.0054

Table 18: Average MSE and NLL on weather tasks, with $m = 20$ and $\gamma = 0.5$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0192 ± 0.0100	-0.7610 ± 0.1631	0.0223 ± 0.0085	-0.6620 ± 0.1610	0.0872 ± 0.0141	0.4558 ± 0.1597
CSM	0.0220 ± 0.0047	-0.7287 ± 0.0560	0.0251 ± 0.0059	-0.6132 ± 0.1054	0.0890 ± 0.0059	0.4180 ± 0.0227
STP	0.0036 ± 0.0002	-1.2025 ± 0.0098	0.0051 ± 0.0002	-1.1301 ± 0.0096	0.0490 ± 0.0012	-0.2077 ± 0.0386
STP+JTP	0.0039 ± 0.0003	-1.1838 ± 0.0136	0.0058 ± 0.0003	-1.0930 ± 0.0094	0.0514 ± 0.0011	-0.1921 ± 0.0176
MTP	**0.0034** ± 0.0001	**-1.2154** ± 0.0045	**0.0048** ± 0.0000	**-1.1417** ± 0.0032	**0.0474** ± 0.0014	**-0.2723** ± 0.0222

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3719 ± 0.0314	3.2131 ± 0.8301	0.3815 ± 0.0246	3.2188 ± 0.5647	0.0188 ± 0.0059	-0.6882 ± 0.1302
CSM	0.3694 ± 0.0330	2.0367 ± 0.1129	0.3574 ± 0.0239	2.1260 ± 0.1623	0.0238 ± 0.0031	-0.6237 ± 0.0602
STP	0.2440 ± 0.0047	0.6535 ± 0.0940	0.2373 ± 0.0033	0.6735 ± 0.0313	0.0073 ± 0.0005	-1.0352 ± 0.0164
STP+JTP	0.2299 ± 0.0023	0.6062 ± 0.0165	0.2255 ± 0.0036	0.6693 ± 0.0233	0.0075 ± 0.0003	-1.0110 ± 0.0156
MTP	**0.2197** ± 0.0020	**0.5660** ± 0.0275	**0.2164** ± 0.0038	**0.6211** ± 0.0140	**0.0068** ± 0.0002	**-1.0713** ± 0.0058

Table 19: Average MSE and NLL on weather tasks, with $m = 20$ and $\gamma = 0.75$.

task	TempMin		TempMax		Humidity	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.0159 ± 0.0065	-0.7846 ± 0.1314	0.0231 ± 0.0077	-0.5808 ± 0.1787	0.1021 ± 0.0111	0.8335 ± 0.4399
CSM	0.0275 ± 0.0066	-0.6147 ± 0.1081	0.0382 ± 0.0114	-0.3232 ± 0.1882	0.1236 ± 0.0179	1.0896 ± 0.1494
STP	0.0047 ± 0.0008	-1.1502 ± 0.0474	0.0074 ± 0.0011	-1.0168 ± 0.0571	0.0584 ± 0.0072	-0.0346 ± 0.0640
STP+JTP	0.0046 ± 0.0005	-1.1485 ± 0.0221	0.0077 ± 0.0008	-1.0107 ± 0.0361	0.0597 ± 0.0059	-0.0868 ± 0.0605
MTP	**0.0040** ± 0.0003	**-1.1860** ± 0.0161	**0.0059** ± 0.0002	**-1.0897** ± 0.0118	**0.0530** ± 0.0036	**-0.1990** ± 0.0407

task	Precip		Cloud		Dew	
metric	MSE	NLL	MSE	NLL	MSE	NLL
MOSM	0.3579 ± 0.0203	3.3159 ± 1.2100	0.3852 ± 0.0142	3.2035 ± 1.1725	0.0165 ± 0.0035	-0.6827 ± 0.1129
CSM	0.3971 ± 0.0345	3.3520 ± 0.3413	0.4109 ± 0.0294	3.0007 ± 0.1722	0.0303 ± 0.0030	-0.4133 ± 0.1111
STP	0.2667 ± 0.0085	0.7932 ± 0.0515	0.2669 ± 0.0126	0.7268 ± 0.0284	0.0096 ± 0.0010	-0.9289 ± 0.0488
STP+JTP	0.2441 ± 0.0061	0.6585 ± 0.0192	0.2475 ± 0.0087	0.7097 ± 0.0223	0.0087 ± 0.0007	-0.9563 ± 0.0152
MTP	**0.2333** ± 0.0079	**0.5895** ± 0.0540	**0.2288** ± 0.0054	**0.6421** ± 0.0177	**0.0077** ± 0.0004	**-1.0326** ± 0.0156

F EXPERIMENTAL DETAILS OF IMAGE 2D FUNCTION REGRESSION

In this section, we describe details of the data generating process and experimental settings of the image function regression experiment.

F.1 DATASET

We use 30,000 RGB images from CelebA HQ dataset (Liu et al., 2015) for RGB task and the corresponding semantic segmentation masks among 19 semantic classes from CelebA Mask-HQ dataset (Lee et al., 2020) for Segment task. For Edge task, we apply the Sobel filter (Kanopoulos et al., 1988) on the RGB images to generate continuous-valued edges. This corresponds to the Canny edge (Canny, 1986) without non-maximum suppression, which is also used in Zamir et. al. (2018) (Zamir et al., 2018). For PNCC task, we apply a pretrained 3D face reconstruction model on the RGB images to generate PNCC label maps (Guo et al., 2020a). At each pixel, the PNCC label consists of the (x, y, z) coordinate of the facial keypoint located at the pixel. In summary, labels for RGB are 3-dimensional, Edge are 1-dimensional, Segment are 19-dimensional, and PNCC are 3-dimensional vectors. We split the 30,000 images into 27,000 train, 1,500 valid, and 1,500 test images.

F.2 EVALUATION PROTOCOL

To evaluate the accuracy of the multi-task prediction, we average MSE $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i^t - \hat{y}_i^t)^2$ on the test images for continuous tasks (RGB, Edge, PNCC), and average mean IoU on the test images for discrete task (Segment). The predictive posterior mean is computed by Monte Carlo sampling, the same as in the 1D experiment. For categorical outputs, we discretize the prediction with the argmax operator $\hat{y_i}^t = \text{argmax}_k \, p(y_i^t = k | x_i, v^t)$.

To evaluate the consistency of predictions across tasks (coherency), we translate each RGB prediction to other task labels. For Edge and PNCC, we use the ground-truth label generation algorithm and the pretrained model used to generate the ground-truth labels for the translation, respectively. For Segment, we fine-tuned DeeplabV3+ (Chen et al., 2018) with ImageNet (Krizhevsky et al., 2012) pretrained ResNet-50 (He et al., 2016) backbone. We refer to the github repository of Yakubovskiy (2019) (Yakubovskiy, 2020) for the DeeplabV3+ model. After the translation, we measure MSE and 1 - mIoU for continuous and discrete tasks respectively, to evaluate the *disagreement* (as oppose to the coherency) between the predictions.

To examine the learned correlation across tasks by MTP (Table 3 in the main paper), we compare the performance before and after MTP observes a set of source data. The source data consist of all examples labeled with the source tasks. For example, if the target task is RGB and the source tasks are Edge and Segment, we give Edge and Segment labels for all pixels, while no RGB or PNCC labels are given. Since MTP requires at least one labeled example for each task, we give a single completely labeled example which is chosen randomly to MTP as a base context, before MTP observes the source data. There are total $\binom{4}{1} + \binom{4}{2} + \binom{4}{3} = 14$ different combinations of source tasks exist. By excluding the case where target task is in the set of source tasks, total 7 different combinations of source tasks remain for each target task. To measure the performance gain from task f^1 to f^2, we average the performance gain of f^2 from all sets of source tasks that containing f^1. For example, the performance gain from Edge to RGB is computed by averaging performance gains $\delta_{\text{Edge}\to\text{RGB}}$, $\delta_{\text{Edge,Segment}\to\text{RGB}}$, $\delta_{\text{Edge,PNCC}\to\text{RGB}}$, and $\delta_{\text{Edge,Segment,PNCC}\to\text{RGB}}$, where we denote $\delta_{A\to B}$ by the performance gain from source tasks A to target task B.

G ADDITIONAL RESULTS ON IMAGE 2D FUNCTION REGRESSION

In this section, we provide additional results on the image regression experiment, with various missing rates γ and also with standard deviation from 5 different random seeds.

Table 24: Relative performance gain with 5 different random seeds (%).

Source \ Target	RGB	Edge	Segment	PNCC
RGB	-	**0.5302** $\pm$ **0.0471**	0.0873 $\pm$ 0.0650	0.1857 $\pm$ 0.1294
Edge	**0.0635** $\pm$ **0.0195**	-	0.0818 $\pm$ 0.0663	0.1570 $\pm$ 0.1311
Segment	0.0513 $\pm$ 0.0204	0.3330 $\pm$ 0.2394	-	**0.2924** $\pm$ **0.0262**
PNCC	0.0558 $\pm$ 0.0253	0.3188 $\pm$ 0.2364	**0.1588** $\pm$ **0.0221**	-

Table 20: Average reconstruction errors (performance) and disagreement of predictions (coherency) on 2D function regression, with varying context size (m) and $\gamma = 0$.

task	RGB (performance)		
m	10	100	512
STP	0.0344 ± 0.0002	0.0101 ± 0.0000	0.0034 ± 0.0000
JTP	0.0304 ± 0.0002	0.0073 ± 0.0000	0.0021 ± 0.0000
MTP	**0.0295 ± 0.0001**	**0.0066 ± 0.0000**	**0.0015 ± 0.0000**

task	Edge (performance)			Edge (coherency)		
m	10	100	512	10	100	512
STP	0.0342 ± 0.0001	0.0198 ± 0.0000	0.0066 ± 0.0000	0.0307 ± 0.0002	0.0238 ± 0.0001	0.0128 ± 0.0001
JTP	0.0297 ± 0.0001	0.0124 ± 0.0000	0.0041 ± 0.0000	0.0175 ± 0.0002	0.0120 ± 0.0001	0.0138 ± 0.0000
MTP	**0.0283 ± 0.0001**	**0.0108 ± 0.0000**	**0.0025 ± 0.0000**	**0.0160 ± 0.0001**	**0.0067 ± 0.0000**	**0.0040 ± 0.0000**

task	Segment (performance)			Segment (coherency)		
m	10	100	512	10	100	512
STP	0.6155 ± 0.0024	0.3961 ± 0.0014	0.2072 ± 0.0012	0.6503 ± 0.0026	0.5609 ± 0.0012	0.5197 ± 0.0009
JTP	0.5746 ± 0.0031	0.3800 ± 0.0011	0.2339 ± 0.0024	0.5256 ± 0.0009	0.4959 ± 0.0016	0.5055 ± 0.0007
MTP	**0.5399 ± 0.0019**	**0.3413 ± 0.0017**	**0.1889 ± 0.0011**	**0.5033 ± 0.0014**	**0.4866 ± 0.0016**	**0.4928 ± 0.0006**

task	PNCC (performance)			PNCC (coherency)		
m	10	100	512	10	100	512
STP	0.0068 ± 0.0001	0.0009 ± 0.0000	0.0005 ± 0.0000	0.0317 ± 0.0005	0.0260 ± 0.0003	0.0209 ± 0.0002
JTP	0.0073 ± 0.0001	0.0017 ± 0.0000	0.0007 ± 0.0000	0.0116 ± 0.0002	0.0146 ± 0.0001	0.0174 ± 0.0001
MTP	**0.0052 ± 0.0000**	**0.0007 ± 0.0000**	**0.0004 ± 0.0000**	**0.0098 ± 0.0001**	**0.0120 ± 0.0001**	**0.0136 ± 0.0001**

Figure 10: Qualitative results on 2D function regression. ($\gamma = 0$)

Table 21: Average reconstruction errors (performance) and disagreement of predictions (coherency) on 2D function regression, with varying context size (m) and $\gamma = 0.25$.

task	RGB (performance)		
m	10	100	512
STP	0.0378 ± 0.0002	0.0121 ± 0.0001	0.0041 ± 0.0000
STP+JTP	0.0342 ± 0.0003	0.0094 ± 0.0001	0.0031 ± 0.0000
MTP	**0.0329 ± 0.0002**	**0.0084 ± 0.0000**	**0.0021 ± 0.0000**

task	Edge (performance)			Edge (coherency)		
m	10	100	512	10	100	512
STP	0.0349 ± 0.0001	0.0223 ± 0.0001	0.0085 ± 0.0000	0.0309 ± 0.0002	0.0254 ± 0.0002	0.0146 ± 0.0001
STP+JTP	0.0312 ± 0.0001	0.0152 ± 0.0000	0.0061 ± 0.0000	0.0184 ± 0.0002	0.0119 ± 0.0001	0.0139 ± 0.0000
MTP	**0.0298 ± 0.0001**	**0.0133 ± 0.0000**	**0.0040 ± 0.0000**	**0.0176 ± 0.0001**	**0.0079 ± 0.0001**	**0.0046 ± 0.0000**

task	Segment (performance)			Segment (coherency)		
m	10	100	512	10	100	512
STP	0.6315 ± 0.0040	0.4243 ± 0.0021	0.2463 ± 0.0017	0.6587 ± 0.0012	0.5766 ± 0.0014	0.5246 ± 0.0011
STP+JTP	0.5940 ± 0.0024	0.4013 ± 0.0013	0.2741 ± 0.0018	0.5303 ± 0.0011	0.5037 ± 0.0008	0.5076 ± 0.0013
MTP	**0.5615 ± 0.0022**	**0.3677 ± 0.0009**	**0.2374 ± 0.0014**	**0.5103 ± 0.0028**	**0.4910 ± 0.0011**	**0.4935 ± 0.0010**

task	PNCC (performance)			PNCC (coherency)		
m	10	100	512	10	100	512
STP	0.0079 ± 0.0001	0.0011 ± 0.0000	0.0005 ± 0.0000	0.0334 ± 0.0003	0.0267 ± 0.0001	0.0218 ± 0.0002
STP+JTP	0.0082 ± 0.0001	0.0018 ± 0.0000	0.0008 ± 0.0000	0.0109 ± 0.0001	0.0141 ± 0.0001	0.0181 ± 0.0001
MTP	**0.0061 ± 0.0000**	**0.0009 ± 0.0000**	**0.0005 ± 0.0000**	**0.0098 ± 0.0001**	**0.0117 ± 0.0001**	**0.0135 ± 0.0002**

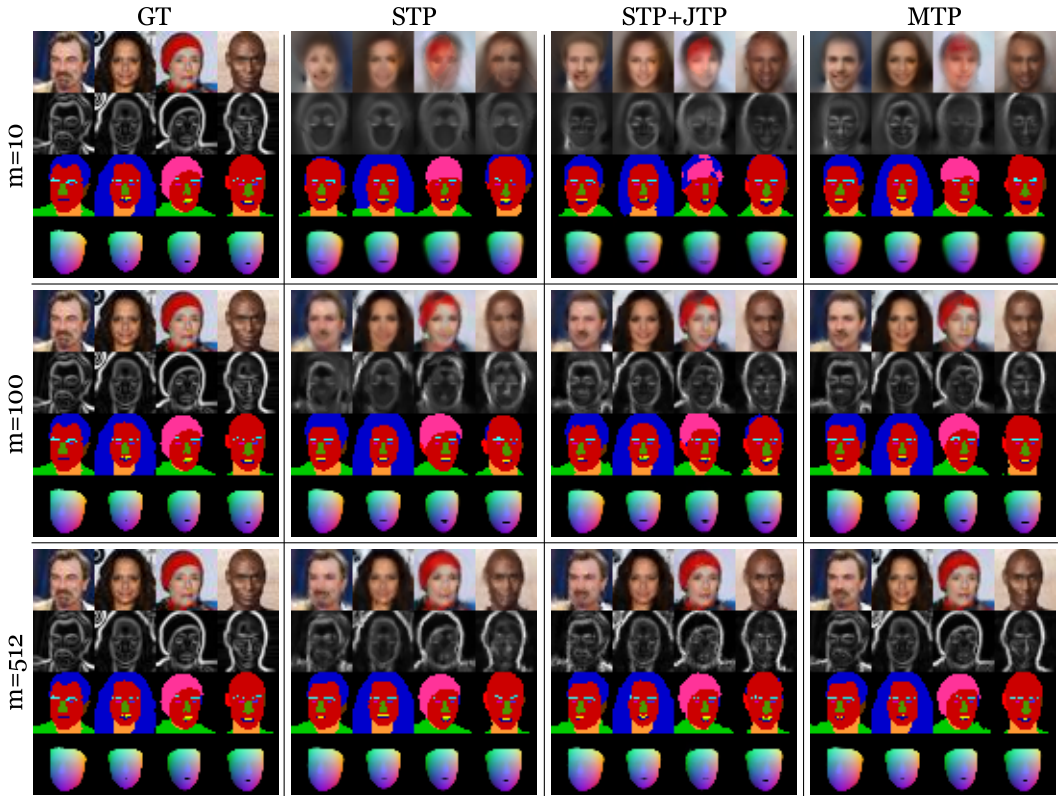


Figure 11: Qualitative results on 2D function regression. ($\gamma = 0.25$)

Table 22: Average reconstruction errors (performance) and disagreement of predictions (coherency) on 2D function regression, with varying context size (m) and $\gamma = 0.5$.

task	RGB (performance)		
m	10	100	512
STP	0.0446 ± 0.0005	0.0154 ± 0.0001	0.0054 ± 0.0000
STP+JTP	0.0422 ± 0.0004	0.0129 ± 0.0001	0.0046 ± 0.0000
MTP	**0.0407 ± 0.0005**	**0.0113 ± 0.0000**	**0.0032 ± 0.0000**

task	Edge (performance)			Edge (coherency)		
m	10	100	512	10	100	512
STP	0.0359 ± 0.0001	0.0256 ± 0.0001	0.0116 ± 0.0000	0.0317 ± 0.0002	0.0271 ± 0.0002	0.0173 ± 0.0001
STP+JTP	0.0337 ± 0.0001	0.0191 ± 0.0000	0.0090 ± 0.0000	0.0200 ± 0.0002	0.0122 ± 0.0001	0.0142 ± 0.0001
MTP	**0.0324 ± 0.0001**	**0.0167 ± 0.0000**	**0.0060 ± 0.0000**	**0.0196 ± 0.0001**	**0.0097 ± 0.0002**	**0.0053 ± 0.0000**

task	Segment (performance)			Segment (coherency)		
m	10	100	512	10	100	512
STP	0.6641 ± 0.0021	0.4669 ± 0.0009	0.2969 ± 0.0019	0.6710 ± 0.0021	0.5966 ± 0.0027	0.5353 ± 0.0012
STP+JTP	0.6322 ± 0.0024	0.4347 ± 0.0013	0.3176 ± 0.0018	0.5317 ± 0.0015	0.5169 ± 0.0019	0.5126 ± 0.0006
MTP	**0.6095 ± 0.0025**	**0.4010 ± 0.0016**	**0.2891 ± 0.0019**	**0.5224 ± 0.0021**	**0.4956 ± 0.0011**	**0.4948 ± 0.0004**

task	PNCC (performance)			PNCC (coherency)		
m	10	100	512	10	100	512
STP	0.0102 ± 0.0002	0.0015 ± 0.0000	0.0006 ± 0.0000	0.0382 ± 0.0005	0.0271 ± 0.0003	0.0232 ± 0.0002
STP+JTP	0.0104 ± 0.0002	0.0022 ± 0.0000	0.0009 ± 0.0000	0.0104 ± 0.0002	0.0134 ± 0.0001	0.0192 ± 0.0002
MTP	**0.0082 ± 0.0001**	**0.0012 ± 0.0000**	**0.0006 ± 0.0000**	**0.0098 ± 0.0001**	**0.0112 ± 0.0001**	**0.0132 ± 0.0001**

Figure 12: Qualitative results on 2D function regression. ($\gamma = 0.5$)

Table 23: Average reconstruction errors (performance) and disagreement of predictions (coherency) on 2D function regression, with varying context size (m) and $\gamma = 0.75$.

task	RGB (performance)		
m	10	100	512
STP	0.0512 ± 0.0005	0.0224 ± 0.0001	0.0086 ± 0.0000
STP+JTP	0.0496 ± 0.0005	0.0203 ± 0.0001	0.0079 ± 0.0000
MTP	$\mathbf{0.0483 \pm 0.0002}$	$\mathbf{0.0178 \pm 0.0001}$	$\mathbf{0.0058 \pm 0.0000}$

task	Edge (performance)			Edge (coherency)		
m	10	100	512	10	100	512
STP	0.0366 ± 0.0001	0.0302 ± 0.0001	0.0176 ± 0.0000	0.0323 ± 0.0003	0.0294 ± 0.0001	0.0222 ± 0.0001
STP+JTP	0.0356 ± 0.0001	0.0251 ± 0.0001	0.0145 ± 0.0000	0.0214 ± 0.0002	0.0138 ± 0.0000	0.0157 ± 0.0001
MTP	$\mathbf{0.0344 \pm 0.0001}$	$\mathbf{0.0223 \pm 0.0001}$	$\mathbf{0.0103 \pm 0.0000}$	$\mathbf{0.0202 \pm 0.0001}$	$\mathbf{0.0130 \pm 0.0001}$	$\mathbf{0.0069 \pm 0.0000}$

task	Segment (performance)			Segment (coherency)		
m	10	100	512	10	100	512
STP	0.6907 ± 0.0018	0.5351 ± 0.0021	0.3712 ± 0.0014	0.6804 ± 0.0015	0.6297 ± 0.0023	0.5601 ± 0.0014
STP+JTP	0.6611 ± 0.0013	0.4916 ± 0.0024	0.3778 ± 0.0031	0.5361 ± 0.0016	0.5385 ± 0.0004	0.5298 ± 0.0009
MTP	$\mathbf{0.6485 \pm 0.0029}$	$\mathbf{0.4573 \pm 0.0025}$	$\mathbf{0.3472 \pm 0.0017}$	$\mathbf{0.5307 \pm 0.0012}$	$\mathbf{0.5031 \pm 0.0014}$	$\mathbf{0.4962 \pm 0.0010}$

task	PNCC (performance)			PNCC (coherency)		
m	10	100	512	10	100	512
STP	0.0123 ± 0.0001	0.0030 ± 0.0001	0.0008 ± 0.0000	0.0453 ± 0.0005	0.0283 ± 0.0003	0.0255 ± 0.0001
STP+JTP	0.0126 ± 0.0001	0.0036 ± 0.0001	0.0011 ± 0.0000	0.0114 ± 0.0002	0.0121 ± 0.0001	0.0195 ± 0.0003
MTP	$\mathbf{0.0103 \pm 0.0001}$	$\mathbf{0.0023 \pm 0.0000}$	$\mathbf{0.0007 \pm 0.0000}$	$\mathbf{0.0091 \pm 0.0001}$	$\mathbf{0.0102 \pm 0.0001}$	$\mathbf{0.0126 \pm 0.0001}$

Figure 13: Qualitative results on 2D function regression. ($\gamma = 0.75$)