# DEEP BAYESIAN ACTIVE LEARNING FOR ACCELERATING STOCHASTIC SIMULATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Stochastic simulations such as large-scale, spatiotemporal, age-structured epidemic models are computationally expensive at fine-grained resolution. While deep surrogate models can speed up the simulations, doing so for stochastic simulations and with active learning approaches is an underexplored area. We propose Interactive Neural Process (INP), a deep Bayesian active learning framework for learning deep surrogate models to accelerate stochastic simulations. INP consists of two components, a spatiotemporal surrogate model built upon Neural Process (NP) family and an acquisition function for active learning. For surrogate modeling, we develop Spatiotemporal Neural Process (STNP) to mimic the simulator dynamics. For active learning, we propose a novel acquisition function, Latent Information Gain (LIG), calculated in the latent space of NP based models. We perform a theoretical analysis and demonstrate that LIG reduces sample complexity compared with random sampling in high dimensions. We also conduct empirical studies on two complex spatiotemporal simulators for reaction diffusion and infectious disease. The results demonstrate that STNP outperforms the baselines in the offline learning setting and LIG achieves the state-of-the-art for Bayesian active learning.

## 1 INTRODUCTION

Computational modeling is now more than ever at the forefront of infectious disease research due to the COVID-19 pandemic. Stochastic simulations play a critical role in understanding and forecasting infectious disease dynamics, creating what-if scenarios, and informing public health policy making (Cramer et al., 2021). More broadly, stochastic simulations (Ripley, 2009; Asmussen & Glynn, 2007) produce forecasts about complex interactions among people, environment, space, and time given a set of parameters. They provide the numerical tools to simulate stochastic processes in finance (Lamberton & Lapeyre, 2007), chemistry (Gillespie, 2007) and many other scientific disciplines.

Unfortunately, stochastic simulations at fine-grained spatial and temporal resolution can be extremely computationally expensive. In example, epidemic models for realistic diffusion dynamics simulation via in-silico experiments require a large parameter space (e.g. characteristics of a virus, policy interventions, people's behavior). Similarly, reaction-diffusion systems that play an important role in chemical reaction and bio-molecular processes also involve a large number of simulation conditions. Therefore, hundreds of thousands of simulations are required to explore and calibrate the simulation model with observed experimental data. This process significantly hinders the adaptive capability of existing stochastic simulators, especially in "war time" emergencies, due to the lead time needed to execute new simulations and produce actionable insights that could help guide decision makers.

Learning deep surrogate models to speed up complex simulation has been explored in climate modeling and fluid dynamics for *deterministic* dynamics (Sanchez-Gonzalez et al., 2020; Wang et al., 2020; Holl et al., 2019; Rasp et al., 2018; Cachay et al., 2021), but not for *stochastic* simulations. These surrogate models can only approximate specific system dynamics and fail to generalize under different parametrization. Especially for pandemic scenario planning, we desire models that can predict futuristic scenarios under different conditions. Furthermore, the majority of the surrogate models are trained *passively* using a simulation data set. This requires a large number of simulations beforehands to cover different parameter regimes of the simulator and ensure generalization.

We propose Interactive Neural Process (INP), a deep Bayesian active learning framework to speed up stochastic simulations. Given parameters such as disease reproduction number, incubation and

infectious periods, mechanistic simulators generate future outbreak states with time-consuming numerical integration. INP accelerates the simulation by guiding a surrogate model to learn the input-output map between parameters and future states, hence bypassing numerical integration.

The deep surrogate model of INP is built upon Neural Process (NP) Garnelo et al. (2018), which lies between Gaussian process (GP) and neural network (NN). NPs can approximate stochastic processes and therefore are well-suitable for surrogate modeling of stochastic simulators. They learn distributions over functions and can generate prediction uncertainty for Bayesian active learning. Compared with GPs, NPs are more flexible and scalable for high-dimensional data with spatiotemporal dependencies. We design a novel Spatiotemporal Neural Process (STNP) by introducing a time-evolving latent process for temporal dynamics and integrating spatial convolution for spatial modeling.

Instead of learning passively, we design *active learning* algorithms to interact with the simulator and update our model in "real-time". We derive a new acquisition function, Latent Information Gain (LIG), based on our unique model design. Our algorithm selects the parameters with the highest LIG, queries the simulator to generate new simulation data, and continuously updates our model. We provide theoretical guarantees for the sample efficiency of this procedure over random sampling. We also demonstrate the efficacy of our method on large-scale spatiotemporal epidemic and reaction diffusion models. In summary, our contributions include:

- Interactive Neural Process: a deep Bayesian active learning framework for accelerating large-scale stochastic simulation.
- A novel Spatiotemporal Neural Process model (STNP) for high-dimensional time series data that integrates temporal latent process and spatial convolution.
- New acquisition function, Latent Information Gain (LIG), based on the inferred temporal latent process to quantify uncertainty with theoretical guarantees.
- Real-world application to speed up complex stochastic spatiotemporal simulations including age-structured epidemic dynamics and reaction-diffusion system.

## 2 Related Work

**Bayesian Active Learning and Experimental Design.** Bayesian active learning, or experimental design is well-studied in statistics and machine learning (Chaloner & Verdinelli, 1995; Cohn et al., 1996). Gaussian Processes (GPs) are popular for posterior estimation e.g. Houlsby et al. (2011) and (Zimmer et al., 2018), but often struggle in high-dimension. Deep neural networks provide scalable solutions for active learning. Deep active learning has been applied to discrete problems such as image classification (Gal et al., 2017) and sequence labeling (Siddhant & Lipton, 2018) whereas our task is continuous time series. Our problem can also be viewed as sequential experimental design where we design simulation parameters to obtain the desired outcome (imitating the simulator). Kleinegesse & Gutmann (2020) and Foster et al. (2021) propose deep design networks for Bayesian experiment design but they require a explicit likelihood model, conditional independence in experiments, and are limited to low (1-2) dimensional design. In contrast, our design space is of much higher-dimension and we do not have access to an explicit likelihood model for the simulator.

**Neural Processes.** Neural Processes (NP) (Garnelo et al., 2018) model distributions over functions and imbue neural networks with the ability of GPs to estimate uncertainty. NP has many extensions such as attentive NP (Kim et al., 2019) and functional NP (Louizos et al., 2019). However, NP implicitly assumes permutation invariance in the latent variables and can be limiting in modeling temporal dynamics. Singh et al. (2019) proposes sequential NP by incorporating a temporal transition model into NP. Still, sequential NP assumes the latent variables are independent conditioned on the hidden states. We propose *spatiotemporal* NP with temporal latent process and spatial convolution, which is well-suited for modeling the spatiotemporal dynamics of infectious disease. We apply our model to real-world large-scale Bayesian active learning. Note that even though Garnelo et al. (2018) has demonstrated NP for Bayesian optimization, it is only for toy 1-D functions.

**Stochastic Simulation and Dynamics Modeling.** Stochastic simulations are fundamental to many scientific fields (Ripley, 2009), especially epidemic modeling. Data-driven models of infectious diseases are increasingly used to forecast the evolution of an ongoing outbreak (Arik et al., 2020; Cramer et al., 2021; Lourenco et al., 2020). However, very few models can mimic the internal

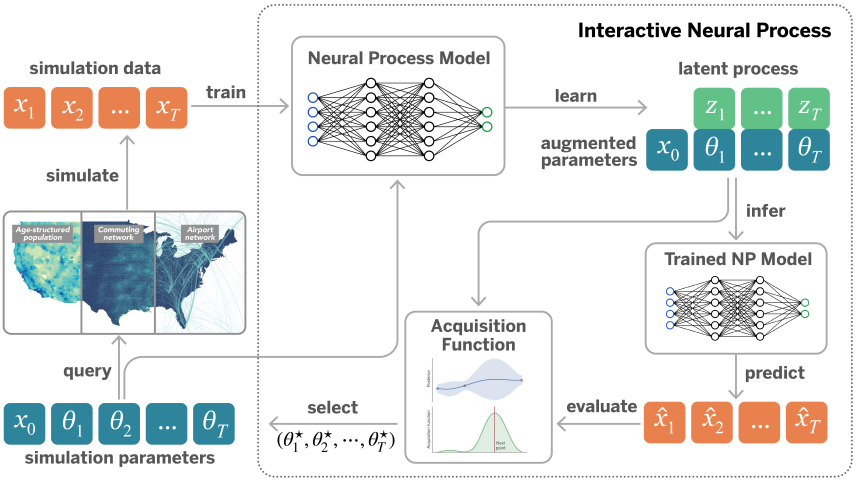


Figure 1: Illustration of the interactive Neural Process (INP). Given simulation parameters and data, INP trains a surrogate model (e.g. STNP) to infer the latent process. The inferred latent process allows prediction and uncertainty quantification. The uncertainty is used to calculate the acquisition function (e.g. LIG) to select the next set of parameters to query, and simulate more data.

mechanism of a stochastic simulator and answer "what-if questions". Recently, Qian et al. (2020) proposed to use Gaussian process (GPs) as a prior for a SEIR compartmental model for learning lockdown policy effects, but GPs are computationally expensive and the simple SEIR model cannot capture the real-world large-scale, spatiotemporal dynamics considered in this work. We demonstrate the use of deep sequence model as a prior distribution in Bayesian active learning. Our framework is also compatible with other deep sequence models for time series, e.g. Deep State Space (Rangapuram et al., 2018), Neural ODE (Chen et al., 2018).

## 3 Methodology

Consider a stochastic process $\{X_1, \cdots, X_T\}$, governed by time-varying parameters $\theta_t \in \mathbb{R}^K$, and the initial state $x_0 \in \mathbb{R}^D$. In epidemic modeling, $\theta_t$ can represent the effective reproduction number of the virus at a given time, the effective contact rates between individuals belonging to different age groups, the people's degree of short- or long-range mobility, or the effects of time varying policy interventions (e.g. non-pharmaceutical interventions). The state $x_t \in \mathbb{R}^D$ includes both the daily prevalence and daily incidence for each compartment of the epidemic model (e.g. number of people that are infectious and number of new infected individuals at time $t$).

Stochastic simulation uses a mechanistic model $F(\theta; \xi)$ to simulate the process where the random variable $\xi$ represents the randomness in the simulator. Let $\theta := (x_0, \theta_1, \cdots, \theta_T)$ represent the initial state and all the parameters over time. For each $\theta$, we obtain a different set of simulation data $\{(x_1, \cdots, x_T)_m\}_{m=1}^M$. However, realistic large-scale stochastic simulations require the exploration of a large parameter space and are extremely computationally intensive. In the following section, we describe the Interactive Neural Process (INP) framework to proactively query the stochastic simulator, generate simulation data, in order to learn a fast surrogate model for rapid simulation.

### 3.1 Interactive Neural Process

INP is used to train a deep surrogate model to mimic the stochastic simulator. As shown in Figure 1, given parameters $\theta$, we query the simulator, i.e., the mechanistic model to obtain a set of simulations $\{(x_1, \cdots, x_T)_m\}_{m=1}^M$. We train a NP based model to learn the probabilistic map from parameters to future states. Our NP model can be spatiotemporal to capture complex dynamics such as the disease dynamics of the epidemic simulator. During inference, the model needs to generate predictions $(\hat{x}_1, \cdots, \hat{x}_T)$ at the target parameters $\theta$ corresponding to different scenarios.

Instead of simulating at a wide range of parameter regimes, we take a Bayesian active learning approach to proactively query the simulator and update the model incrementally. Using NP, we can infer the latent temporal process $(z_1, \cdots, z_T)$ that encodes the uncertainty of the current surrogate model. Then we propose a new acquisition function, Latent Information Gain (LIG), to select the $\theta^\star$

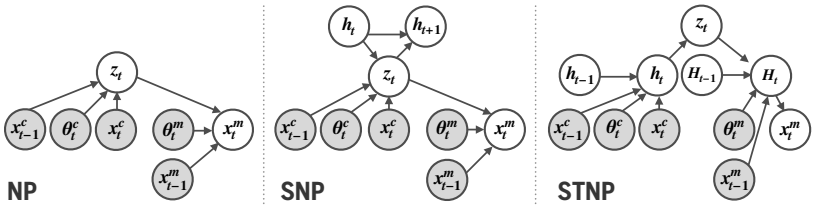


Figure 2: Graphical model comparison: Neural Process, Sequential Neural Process and our Spatiotemporal Neural Process.

with the highest reward. We use $\theta^\star$ to query the simulator, and in turn generate new simulation to further improve the model. Next, we describe each of the components in detail.

### 3.2 Spatiotemporal Neural Process

Neural Process (NP) (Garnelo et al., 2018) is a type of deep generative model that represents distributions over functions. It introduces a global latent variable $z$ to capture the stochasticity and learns the conditional distribution $p(x_{1:T}|\theta)$ by optimizing the evidence lower bound (ELBO):

$$\log p(x_{1:T}|\theta) \geq \mathbb{E}_{q(z|x_{1:T},\theta)}\big[\log p(x_{1:T}|z,\theta)\big] - \mathrm{KL}\big(q(z|x_{1:T},\theta)\|p(z)\big) \tag{1}$$

Here $p(z)$ is the prior distribution for the latent variable. We use $x_{1:T}$ as a shorthand for $(x_1, \cdots, x_T)$. The prior distribution $p(z)$ is conditioned on a set of context points $\theta^c, x^c_{1:T}$ as $p(z|x^c_{1:T}, \theta^c)$.

However, the global latent variable $z$ in NP can be limiting for non-stationary, spatiotemporal dynamics in the epidemics. We propose Spatiotemporal Neural Process (STNP) with two extensions. First, we introduce a temporal latent process $(z_1, \cdots, z_T)$ to represent the unknown dynamics. The latent process provides an expressive description of the internal mechanism of the stochastic simulator. Each latent variable $z_t$ is sampled conditioning on the past history. Second, we explicitly model the spatial dependency in $x_t \in \mathbb{R}^D$. Rather than treating the dimensions in $x_t$ as independent features, we capture their correlations with regular grids or graphs. For instance, the travel graph between locations can be represented as an adjacency matrix $A \in \mathbb{R}^{D\times D}$.

Given parameters $\{\theta\}$, simulation data $\{x_{1:T}\}$, and the spatial graph $A$ as inputs, STNP models the conditional distribution $p(x_{1:T}|\theta, A)$ by optimizing the following ELBO objective:

$$\log p(x_{1:T}|\theta, A) \geq \mathbb{E}_{q(z_{1:T}|x_{1:T},\theta,A)} \log p(x_{1:T}|z_{1:T},\theta,A) - \mathrm{KL}\big(q(z_{1:T}|x_{1:T},\theta,A)\|p(z_{1:T})\big) \tag{2}$$

where the distributions $q(z_{1:T}|x_{1:T},\theta,A)$ and $p(x_{1:T}|z_{1:T},\theta,A)$ are parameterized with neural networks. The prior distribution $p(z_{1:T})$ is conditioned on a set of contextual sequences $p(z_{1:T}|x^c_{1:T},\theta^c,A)$. Figure 2 visualizes the graphical models of our STNP, the original NP (Garnelo et al., 2018) model and Sequential NP (Singh et al., 2019). The main difference between STNP and baselines is the encoding procedure to infer the temporal latent process. Compared with STNP which directly embeds the history for $z$ inference at the current timestamp, NP ignores the history and SNP only embeds the partial history information from the previous $z$.

We implement STNP following an encoder-decoder architecture. The encoder parametrizes the mean and standard deviation of the variational posterior $q(z_{1:T}|x_{1:T},\theta,A)$ and the decoder approximates the predictive distribution $p(x_{1:T}|z_{1:T},\theta,A)$. To incorporate the spatial graph information, we use a Diffusion Convolutional Gated Recurrent Unit (DCGRU) layer (Li et al., 2017) which integrates graph convolution in a GRU cell. We use multi-layer GRUs to obtain hidden states from the inputs. Using re-parametrization (Kingma & Welling, 2013), we sample $z_t$ from the encoder and then decode $x_t$ conditioned on $z_t$ in an auto-regressive fashion. Noted if the spatial dependency is regular grid-based, then the DCGRU layer is replaced to Convolutional LSTM layer Lin et al. (2020); Wang et al. (2017); Shi et al. (2015); Yao et al. (2019; 2018), and there is no adjacency matrix $A$ in Equation 2.

### 3.3 Bayesian Active Learning

Algorithm 1 details a Bayesian active learning algorithm, based on Bayesian optimization (Shahriari et al., 2015; Frazier, 2018). We train an NP model to interact with the simulator and improve learning. Let the superscript $^{(i)}$ denote the $i$-th interaction. We start with an initial data set $\mathcal{S}_1 = \{\theta^{(1)}, x^{(1)}_{1:T}\}$

**Algorithm 1:** Interactive Neural Process

**Input:** Initial simulation dataset $\mathcal{S}_1$

Train the model $\texttt{NP}^{(1)}(\mathcal{S}_1)$ ;

**for** $i = 1, 2, \cdots$ **do**

  Learn $(z_1, z_2, \cdots, z_T) \sim q^{(i)}(z_{1:T}|x_{1:T}, \theta, \mathcal{S}_i)$;

  Predict $(\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_T) \sim p^{(i)}(x_{1:T}|z_{1:T}, \theta, \mathcal{S}_i)$ ;

  Select a batch $\{\theta^{(i+1)}\} \leftarrow \arg\max_\theta \mathbb{E}_{p(x_{1:T}|z_{1:T},\theta)}\left[r(\hat{x}_{1:T}|z_{1:T}, \theta)\right]$;

  Simulate $\{x_{1:t}^{(i+1)}\} \leftarrow$ Query the simulator $F(\theta^{(i+1)}; \xi)$ ;

  Augment training set $\mathcal{S}_{i+1} \leftarrow \mathcal{S}_i \cup \{\theta^{(i+1)}, x_{1:T}^{(i+1)}\}$ ;

  Update the model $\texttt{NP}^{(i+1)}(\mathcal{S}_{i+1})$ ;

**end**

and use it to train our NP model and learn the latent process. During inference, given the augmented parameters $\theta$, we use the trained NP model to predict the future states $(\hat{x}_1, \cdots, \hat{x}_T)$. We evaluate the current models' predictions with an acquisition function $r(\hat{x}_{1:T}, z_{1:T}, \theta)$ and select the set of parameters $\{\theta^{(i+1)}\}$ with the highest reward. We query the simulator with $\{\theta^{(i+1)}\}$ to augment the training data set $\mathcal{S}_{i+1}$ and update the NP model for the next iteration.

The choice of the reward (acquisition) function $r$ depends on the goal of the active learning task. For example, to find the model that best fits the data, the reward function can be the log-likelihood $r = \log p(\hat{x}_{1:T}|\theta, A)$. To collect data and reduce model uncertainty in Bayesian experimental design, the reward function can be the mutual information. In what follows, we discuss different strategies to design the reward/acquisition function. We also propose a novel acquisition function based on information gain in the latent space tailored to our STNP model.

### 3.4 Reward/Acquisition functions

For regression tasks, standard acquisition functions for active learning include Maximum Mean Standard Deviation (Mean STD), Maximum Entropy, Bayesian Active Learning by Disagreement (BALD) or expected information gain (EIG), and random sampling (Gal et al., 2017). We explore various acquisition functions and their approximations in the context of NP. We also introduce a new acquisition function based on our unique NP design called Latent Information Gain (LIG). The details of Mean STD and Maximum Entropy are shown in the Appendix B.4.

**BALD/Expected Information Gain (EIG).** BALD (Houlsby et al., 2011) quantifies the mutual information between the prediction and model posterior $H(\hat{x}_{1:T}|\theta) - H(\hat{x}_{1:T}|z_{1:T}, \theta)$, which is equivalent to the expected information gain (EIG). Computing the EIG for surrogate modeling is challenging since $p(\hat{x}_{1:T}|z_{1:T}, \theta)$ cannot be found in closed form in general. The integrand is intractable and conventional MC methods are not applicable (Foster et al., 2019). One way to get around this is to employ a nested MC estimator with quadratic computational cost for sampling (Myung et al., 2013; Vincent & Rainforth, 2017), which is computationally infeasible. To reduce the computational cost, we assume $p(\hat{x}_{1:T}|z_{1:T}, \theta)$ follows multivariate Gaussian distribution. Each feature of $\hat{x}_{1:T}$ can be parameterized with mean and standard deviation predicted from the surrogate model, assuming output features are independent with each other. This distribution assumption can be limiting in the high-dimensional spatiotemporal domain, which makes EIG less informative.

**Latent Information Gain (LIG).** To overcome the limitations mentioned above, we propose a novel acquisition function by computing the expected information gain in the latent space rather than the observational space. To design this acquisition function, we prove the equivalence between the expected information gain in the observational space and the expected KL divergence in the latent processes w.r.t. a candidate parameter $\theta$, as illustrated by the following proposition.

**Proposition 1.** *The expected information gain (EIG) for Neural Process is equivalent to the KL divergence between the prior and posterior in the latent process, that is*

$$\text{EIG}(\hat{x}_{1:T}, \theta) := \mathbb{E}[H(\hat{x}_{1:T}) - H(\hat{x}_{1:T}|z_{1:T}, \theta)] = \mathbb{E}_{p(\hat{x}_{1:T}|\theta)}\left[\text{KL}\big(p(z_{1:T}|\hat{x}_{1:T}, \theta)\|p(z_{1:T})\big)\right] \quad (3)$$

See proof in the Appendix A.1. Inspired by this fact, we propose a novel acquisition function computing the expected KL divergence in the latent processes and name it LIG. Specifically, the trained NP model produces a variational posterior given the current dataset $\mathcal{S}$ as $p(z_{1:T}|\mathcal{S})$. For every parameter $\theta$ remained in the search space, we can predict $\hat{x}_{1:T}$ with the decoder. We use $\hat{x}_{1:T}$ and $\theta$ as input to the encoder to re-evaluate the posterior $p(z_{1:T}|\hat{x}_{1:T}, \theta, \mathcal{S})$. LIG computes the distributional difference with respect to the latent process $z_{1:T}$ as $\mathbb{E}_{p(\hat{x}_{1:T}|\theta)}\left[\mathrm{KL}\left(p(z_{1:T}|\hat{x}_{1:T}, \theta, \mathcal{S})\|p(z_{1:T}|\mathcal{S})\right)\right]$, where $\mathrm{KL}(\cdot\|\cdot)$ denotes the KL-divergence between two distributions.

In this way, conventional MC method becomes applicable, which helps reduce the quadratic computational cost to linear. At the same time, although $z_{1:T}$ are assumed to be multivariate Gaussian and are parameterized with mean and standard deviation, they are only in the latent space not the observational space. Moreover, LIG is also more computationally efficient and accurate for batch active learning. Due to the context aggregation mechanism of NP, we can directly calculate LIG with respect to a batch of $\theta$ in the candidate set. This is not available for baseline acquisition functions. They all require calculating the scores one by one for all $\theta$ in the candidate set and select a batch of $\theta$ based on their scores. Such approach is both slow and inaccurate as acquiring points that are informative individually are not necessarily informative jointly (Kirsch et al., 2019).

### 3.5 Theoretical Analysis

We shed light onto the intuition behind choosing adaptive sample selection over random sampling via analyzing a simplifying situation. Assume that at a certain stage we have learned a feature map $\Psi$ which maps the input $\theta$ of the neural network to the last layer. Then the output $X$ can be modeled as $X = \langle \Psi(\theta), z^* \rangle + \epsilon$, where $z^*$ is the true hidden variable, $\epsilon$ is the random noise.

Our goal is to generate an estimate $\hat{z}$, and use it to make predictions $\langle \Psi(\theta), \hat{z} \rangle$. A good estimate shall achieve small error in terms of $\|\hat{z}_t - z^*\|_2$ with high probability. In the following theorem, we prove that greedily maximizing the variance of the prediction to choose $\theta$ will lead to an error of order $\mathcal{O}(d)$ less than that of random exploration in the space of $\theta$, which is significant in high dimension.

**Theorem 1.** *For random feature map $\Psi(\cdot)$, greedily optimizing the KL divergence, $\mathrm{KL}\left(p(z|\hat{x}, \theta)\|p(z)\right)$ , or equivalently the variance of the posterior predictive distribution $\mathbb{E}\left[\left(\langle \Psi(\theta), \hat{z} \rangle - \mathbb{E}\langle \Psi(\theta), \hat{z} \rangle\right)^2\right]$ in search of $\theta$ will lead to an error $\|\hat{z}_t - z^*\|_2$ of order $\mathcal{O}\left(\sigma d/\sqrt{t}\right)$ with high probability. On the other hand, random sampling of $\theta$ will lead to an error of order $\mathcal{O}\left(\sigma d^2/\sqrt{t}\right)$ with high probability.*

See proofs in the Appendix A.2.

## 4 Experiments

We evaluate our proposed STNP for its surrogate modeling performance in the offline learning setting and LIG acquisition function for active learning performance. We aim to verify that (a) LIG outperforms other acquisition functions in the NP and GP model setting for deep Bayesian active learning on non-spatiotemporal surrogate modeling, (b) STNP outperforms other existing NP baselines for spatiotemporal surrogate modeling in the offline learning setting, and (c) LIG outperforms other acquisition functions in the STNP model setting for deep Bayesian active learning on spatiotemporal surrogate modeling.

### 4.1 Experimental Setup

We experiment with the following three stochastic simulators.

**SEIR Compartmental Model.** To highlight the difference between NP and GP, we begin with a simple stochastic, discrete, chain-binomial SEIR compartmental model as our stochastic simulator. In this model, susceptible individuals ($S$) become exposed ($E$) through interactions with infectious individuals ($I$) and are eventually removed ($R$), details are deferred to the Appendix B.1.

We set the total population $N = S + E + I + R$ as $100,000$, the initial number of exposed individuals as $E_0 = 2,000$, and the initial number of infectious individuals as $I_0 = 2,000$. We assume latent individuals move to the infectious stage at a rate $\varepsilon \in [0.25, 0.65]$ (step 0.05), the infectious period

$\mu^{-1}$ is set to be equal to 1 day, and we let the basic reproduction number $R_0$ (which in this case coincides with the transmissibility rate $\beta$) vary between $1.1$ and $4.0$ (step $0.1$). Here, each $(\beta, \varepsilon)$ pair corresponds to a specific scenario, which determines the parameters $\theta$. We simulate the first $100$ days of the epidemic with a total of $300$ scenarios and generate $30$ samples for each scenario.

We predict the number of individuals in the infectious compartment. The input is $(\beta, \varepsilon)$ pair and the output is the $100$ days' infection prediction. As the simulator is not spatiotemporal, we use the vanilla NP model with the global latent variable $z$. For each epoch, we randomly select $10\%$ of the samples as context. Implementation details are deferred to Appendix B.5.

**Reaction Diffusion Model.** The reaction-diffusion (RD) system (Turing, 1990) is a spatiotemporal model that simulates how two chemicals might react to each other as they diffuse through a medium together. The simulation is based on initial pattern, feed rate ($\theta_0$), removal rate ($\theta_1$) and reaction between two substances. We use an RD simulator to generate sequences from 0 to 500 timestamps, sampled every 100 timestamps, resulting into $5$ timestamps for each simulated sequence. Every timestamp is a 3D tensor $(2 \times 32 \times 32)$ with dimension $0$ corresponds to the two substances in the reaction and dimension $1, 2$ are the image representation of the reaction diffusion processes. Each sequence is simulated with a unique feed rate $\theta_0 \in [0.029, 0.045]$ and kill rate $\theta_1 \in [0.055, 0.062]$ combination. There are $200$ uniformly sampled scenarios, corresponding to $(\theta_0, \theta_1)$ combinations.

We implement STNP to mimic the reaction diffusion simulator with feed rate ($\theta_0$) and kill rate ($\theta_1$) as input. The initial state of the reaction is fixed. We use multiple convolutional layers with a linear layer to encode the spatial data into latent space. We use an LSTM layer to encode the latent spatial data with $\theta_0$, $\theta_1$ to map the input-output pairs to hidden features $z_{1:5}$. With $(\theta_0, \theta_1)$, and $z_{1:5}$ sampled from the posterior distribution, we use an LSTM layer and deconvolutional layers to simulate reaction diffusion sequence. For each epoch, we randomly select $20\%$ samples as context sequence.

**Local Epidemic and Mobility model.** The Local Epidemic and Mobility model (LEAM-US) is a stochastic, spatial, age-structured epidemic model based on a metapopulation approach which divides the US in more than 3,100 subpopulations, each one corresponding to a each US county or statistically equivalent entity. Population size and county-specific age distributions reflect Census' annual resident population estimates for year 2019. We consider individuals divided into $10$ age groups. Contact mixing patterns are age-dependent and state specific and modeled considering contact matrices that describe the interaction of individuals in different social settings (Mistry et al., 2021). LEAM-US integrates a human mobility layer, represented as a network, using both short-range (i.e., commuting) and long-range (i.e., flights) mobility data, see more details in Appendix B.2.

We separate data in California monthly to predict the 28 days' sequence from the 2nd to the 29th day of each month from March to December. Each $\theta$ includes the county-level parameters of LEAM-US and state level incidence and prevalence compartments. The total number of dimension in $\theta$ is $16,912$, see details in Appendix B.2. Overall, there are $315$ scenarios in the search space, corresponding to $315$ different $\theta$ with total $16,254$ samples. We split $78\%$ of the data as the candidate set, and $11\%$ for validation and test. For active learning, we use the candidate set as the search space.

We instantiate an STNP model to mimic an epidemic simulator that has $\theta$ at both county and state level and $x_t$ at the state level. We use county-level parameter $\theta$ together with a county-to-county mobility graph $A$ in California as input. We use the DCGRU layer (Li et al., 2017) to encode the mobility graph in a GRU. We use a linear layer to map the county-level output to hidden features at the state level. For both the state-level encoder and decoder, we use multi-layer GRUs. For each epoch, we randomly select $20\%$ samples as context sequence.

### 4.2 Offline Learning Performance

We compared our proposed STNP with vanilla NP (Garnelo et al., 2018) and SNP (Singh et al., 2019). The key innovation of STNP is the introduced temporal latent process. To ensure fair comparison, we modified NP for the RD model by adding convolutional layers for data encoding and deconvolutional layers for sequence generation. For the LEAM-US model, we modified NP by adding the convolutional layers with diffusion convolution (Li et al., 2018) to embed the graphs. Similarly, we modified SNP by replacing the convolutional layers with diffusion convolution. Table 1 shows the testing MAE of different NP models trained in an offline fashion. Our STNP significantly improves the performance and can accurately learn the simulator dynamics for both experiments.

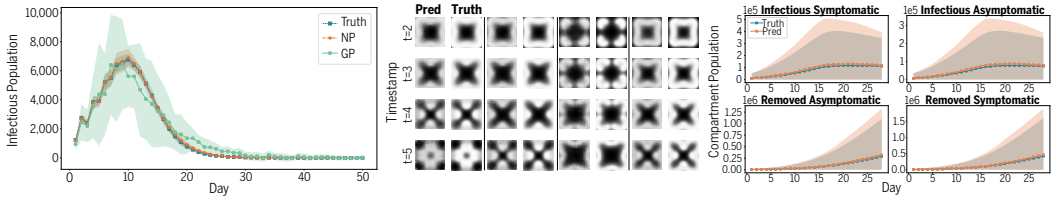


Figure 3: Prediction visualizations, Left: Accuracy and uncertainty quantification comparison between Neural Process (NP) and Gaussian process (GP) in SEIR simulator. Middle: STNP predictions for spatiotemporal patterns of substances in Reaction-Diffusion simulator. Right: STNP predictions for the number of individuals in Infectious and Removed compartments in LEAM-US simulator.

Figure 3 left compares the NP and GP performance on one scenario in the held-out test set. It shows the ground truth and the predicted number of infectious population for the first 50 days. We also include the confidence intervals (CI) with 5 standard deviations for ground truth and NP predictions and 1 standard deviation for GP predictions. We observe that NP fits the simulation dynamics better than GP for mean prediction. Moreover, NP has closer CIs to the truth, reflecting the simulator's intrinsic uncertainty. GP shows larger CIs which represent the model's own uncertainty. Note that NP is much more flexible than GP and can scale easily to high-dimensional data. Figure 3 middle indicates STNP can accurately predict various patterns corresponding to different $(\theta_0, \theta_1)$. This confirms that our STNP is able to capture the high-dimensional spatiotemporal dependencies in RD simulations. Figure 3 right visualize the STNP predictions in four key compartments of a typical scenario with $R_0 = 3.1$ from March 2nd to March 29th. The confidence interval is plotted with 2 standard deviations. We can see that both the mean and confidence interval of STNP predictions match the truth well. These two results demonstrate the promise that the generative STNP model can serve as a deep surrogate model for RD and LEAM-US simulator.

Table 1: Surrogate model performance comparison using MAE in Reaction-Diffusion simulator and LEAM simulator (population divided by 1000).

| Model | RD | LEAM |
|---|---|---|
| NP | $3.37 \pm 0.18$ | $24.2 \pm 5.9$ |
| SNP | $3.11 \pm 0.07$ | $21.8 \pm 0.8$ |
| STNP | $\mathbf{2.84} \pm 0.17$ | $\mathbf{6.3} \pm 0.8$ |

### 4.3 Active Learning

**Implementation Details.** We compare 6 different acquisition functions with NP for SEIR model and STNP for RD and LEAM-US model. For SEIR, the initial training dataset has 2 scenarios and we continue adding 1 scenario per iteration to the training set until the test loss converges to the offline modeling performance. We also include GP with 3 different acquisition functions. For the RD model, all acquisition functions start with the same 5 scenarios randomly picked from the training dataset. Then we continue adding 5 scenarios per iteration to the training set until the test loss converges. Similarly, the LEAM-US model begins with 27 training data and we continue adding 8 scenarios per iteration to the training set until the validation loss converges. We measure the average performance over three random runs and report the MAE for the test set.

**Active Learning Performance.** Figure 4 shows the testing MAE versus the percentage of samples included for training. The percentage of data is linearly proportional to the overall running time. This

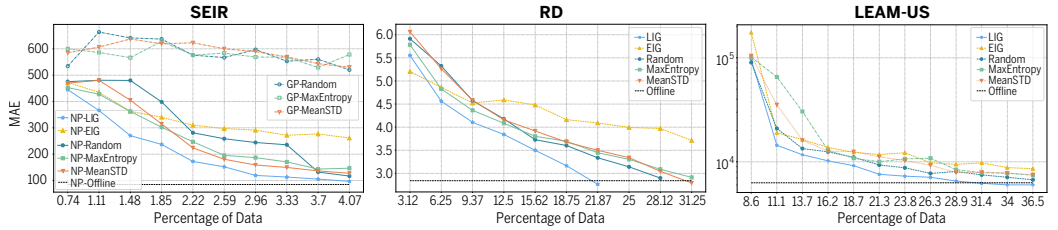


Figure 4: MAE loss versus the percentage of samples for Bayesian active learning. The Black dash line shows the offline learning performance with the entire data set available for training. Left: GP and NP for SEIR. Middle: STNP for RD. Right: STNP for LEAM-US.

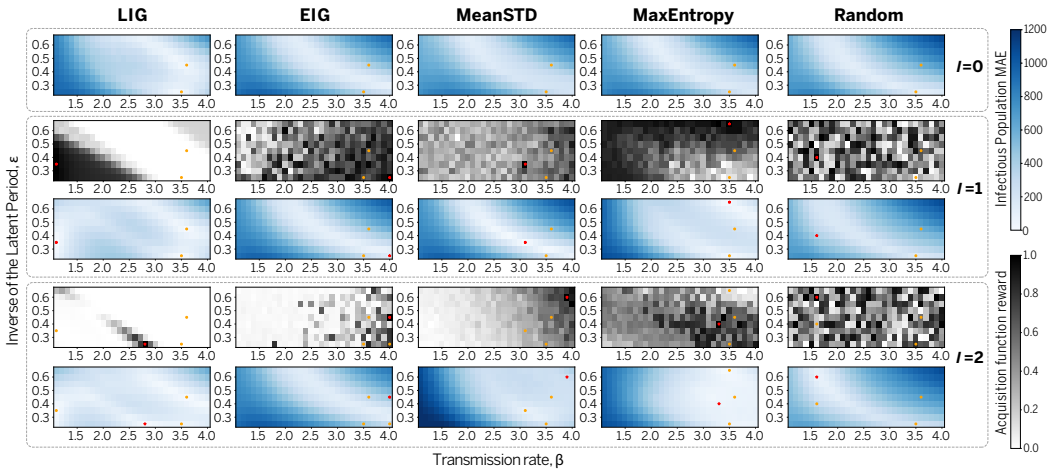


Figure 5: Acquisition function behavior visualization in SEIR model. For each iteration, top row is the current MAE mesh in infectious population for all $(\beta, \varepsilon)$ candidates. Bottom row is the acquisition function score. Yellow dots are existing parameters. Red stars are the newly selected parameters.

figure shows our proposed LIG always has the best MAE performance until the convergence for all three experiments. Specifically, as shown in figure 4 left, we compare different acquisition functions on both NP and GP for SEIR model. It shows none of the GP methods converge after selecting $4.07\%$ of the data for training while NP methods converge much faster. Our proposed acquisition function LIG is the most sample efficient in acquisition functions used for NP. It takes only $4.07\%$ of the data to converge and reach the NP offline performance, which uses the entire training set for training. Moreover, there is an enormous gap between LIG and EIG with respect to the active learning performance. This validates our theory that the uncertainty of the deep surrogate model is better measured on the latent space instead of the predictions. Similarly in figure 4 middle and right, we compared LIG with other acquisition functions on STNP for RD and LEAM-US model. It shows LIG converges to the offline performance using only $21.87\%$ of data for RD experiment and $31.4\%$ of data for LEAM-US experiment. Therefore, it is consistent among all three experiments that our proposed LIG always has the best MAE performance until convergence. Notice that for figure 4 right, it shows the log scale MAE versus the percentage of samples included for training.

**Exploration Exploitation Trade-off.** To understand the large performance gap for LIG vs. baselines, we visualize the values of test MAE and the acquisition function score for each Bayesian active learning iteration for SEIR model, shown in Figure 5. For EIG, Mean STD, and Maximum Entropy, they all tend to exploit the region with large transmission rate for the first 2 iterations. Including these scenarios makes the training set unbalanced. The MAE in the region with small transmission rate become worse after 2 iterations. Meanwhile, Random is doing pure exploration. The improvement of MAE performance is not apparent after 2 iterations. Our proposed LIG is able to reach a balance by exploiting the uncertainty in the latent process and encouraging exploration. Hence, with a small number of iterations ($I = 2$), it has already selected "informative scenarios" in the search space.

## 5 Conclusion

We propose a unified framework Interactive Neural Processes (INP) for deep Bayesian active learning, that can seamlessly interact with existing stochastic simulators and accelerate simulation. Specifically, we design STNP to approximate the underlying simulation dynamics. It infers the latent process which describes the intrinsic uncertainty of the simulator. We exploit this uncertainty and propose LIG as a powerful acquisition function in deep Bayesian active learning. We perform a theoretical analysis and demonstrate that our approach reduces sample complexity compared with random sampling in high dimension. We also did extensive empirical evaluations on several complex real-world spatiotemporal simulators to demonstrate the superior performance of our proposed STNP and LIG. For the future work, we plan to leverage Bayesian optimization techniques to directly optimize for the target parameters with auto-differentiation.

## REPRODUCIBILITY STATEMENT

The implementation code is included in the supplementary material. The readme file includes the corresponding instructions. The full proof of the Theorem 1 can be found in Appendix A.2.

## REFERENCES


Sercan Arik, Chun-Liang Li, Jinsung Yoon, Rajarishi Sinha, Arkady Epshteyn, Long Le, Vikas Menon, Shashank Singh, Leyou Zhang, Martin Nikoltchev, et al. Interpretable sequence learning for covid-19 forecasting. *Advances in Neural Information Processing Systems*, 33, 2020.

Søren Asmussen and Peter W Glynn. *Stochastic simulation: algorithms and analysis*, volume 57. Springer Science & Business Media, 2007.

Duygu Balcan, Vittoria Colizza, Bruno Gonçalves, Hao Hu, José J Ramasco, and Alessandro Vespignani. Multiscale mobility networks and the spatial spreading of infectious diseases. *Proceedings of the National Academy of Sciences*, 106(51):21484–21489, 2009.

Duygu Balcan, Bruno Gonçalves, Hao Hu, José J Ramasco, Vittoria Colizza, and Alessandro Vespignani. Modeling the spatial spread of infectious diseases: The global epidemic and mobility computational model. *Journal of computational science*, 1(3):132–145, 2010.

Salva Rühling Cachay, Venkatesh Ramesh, Jason N. S. Cole, Howard Barker, and David Rolnick. ClimART: A benchmark dataset for emulating atmospheric radiative transfer in weather and climate models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL `https://arxiv.org/abs/2111.14671`.

Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, pp. 273–304, 1995.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 6572–6583, 2018.

Zizhong Chen and Jack J. Dongarra. Condition numbers of gaussian random matrices. *SIAM Journal on Matrix Analysis and Applications*, 27(3):603–620, 2005.

Matteo Chinazzi, Jessica T Davis, Marco Ajelli, Corrado Gioannini, Maria Litvinova, Stefano Merler, Ana Pastore y Piontti, Kunpeng Mu, Luca Rossi, Kaiyuan Sun, et al. The effect of travel restrictions on the spread of the 2019 novel coronavirus (covid-19) outbreak. *Science*, 2020.

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

Estee Y Cramer, Velma K Lopez, Jarad Niemi, Glover E George, Jeffrey C Cegan, Ian D Dettwiller, William P England, Matthew W Farthing, Robert H Hunter, Brandon Lafferty, et al. Evaluation of individual and ensemble probabilistic forecasts of covid-19 mortality in the us. *medRxiv*, 2021.

Jessica T Davis, Matteo Chinazzi, Nicola Perra, Kunpeng Mu, Ana Pastore y Piontti, Marco Ajelli, Natalie E Dean, Corrado Gioannini, Maria Litvinova, Stefano Merler, Luca Rossi, Kaiyuan Sun, Xinyue Xiong, M. Elizabeth Halloran, Ira M Longini, Cécile Viboud, and Alessandro Vespignani. Estimating the establishment of local transmission and the cryptic phase of the covid-19 pandemic in the usa. *medRxiv*, 2020.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pp. 1675–1685, 2019.

A Foster, M Jankowiak, E Bingham, P Horsfall, YW Tee, T Rainforth, and N Goodman. Variational bayesian optimal experimental design. Conference on Neural Information Processing Systems, 2019.

Adam Foster, Desi R Ivanova, Ilyas Malik, and Tom Rainforth. Deep adaptive design: Amortizing sequential bayesian experimental design. *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pp. 1183–1192. PMLR, 2017.

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

Daniel T Gillespie. Stochastic simulation of chemical kinetics. *Annu. Rev. Phys. Chem.*, 58:35–55, 2007.

Friedrich Götze and Alexander Tikhomirov. Rate of convergence in probability to the Marchenko-Pastur law. *Bernoulli*, 10(3):503 – 548, 2004.

Philipp Holl, Nils Thuerey, and Vladlen Koltun. Learning to control pdes with differentiable physics. In *International Conference on Learning Representations*, 2019.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

IATA, International Air Transport Association, 2021. URL `https://www.iata.org/`. `https://www.iata.org/`.

Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *International Conference on Learning Representation*, 2019.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019.

Steven Kleinegesse and Michael U Gutmann. Bayesian experimental design for implicit models by mutual information neural estimation. In *International Conference on Machine Learning*, pp. 5316–5326. PMLR, 2020.

Damien Lamberton and Bernard Lapeyre. *Introduction to stochastic calculus applied to finance*. CRC press, 2007.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations (ICLR)*, 2018.

Haoxing Lin, Rufan Bai, Weijia Jia, Xinyu Yang, and Yongjian You. Preserving dynamic attention for long-term spatial-temporal prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 36–46, 2020.

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

Jose Lourenco, Robert Paton, Mahan Ghafari, Moritz Kraemer, Craig Thompson, Peter Simmonds, Paul Klenerman, and Sunetra Gupta. Fundamental principles of epidemic spread highlight the immediate need for large-scale serological surveys to assess the stage of the sars-cov-2 epidemic. *MedRxiv*, 2020.

Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and double descent curve. arXiv: 1908.05355, 2019.

Dina Mistry, Maria Litvinova, Ana Pastore y Piontti, Matteo Chinazzi, Laura Fumanelli, Marcelo FC Gomes, Syed A Haque, Quan-Hui Liu, Kunpeng Mu, Xinyue Xiong, et al. Inferring high-resolution human mixing patterns for disease modeling. *Nature communications*, 12(1):1–12, 2021.

Jay I Myung, Daniel R Cavagnaro, and Mark A Pitt. A tutorial on adaptive design optimization. *Journal of mathematical psychology*, 57(3-4):53–67, 2013.

OAG, Aviation Worlwide Limited, 2021. URL `http://www.oag.com/`. `http://www.oag.com/`.

Zhaozhi Qian, Ahmed M Alaa, and Mihaela van der Schaar. When and how to lift the lockdown? global covid-19 scenario analysis and policy assessment using compartmental gaussian processes. *Advances in Neural Information Processing Systems*, 33, 2020.

Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31:7785–7794, 2018.

Stephan Rasp, Michael S Pritchard, and Pierre Gentine. Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, 115(39):9684–9689, 2018.

Brian D Ripley. *Stochastic simulation*, volume 316. John Wiley & Sons, 2009.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pp. 8459–8468. PMLR, 2020.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2015.

Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

Aditya Siddhant and Zachary C Lipton. Deep bayesian active learning for natural language processing: Results of a large-scale empirical study. *arXiv preprint arXiv:1808.05697*, 2018.

Gautam Singh, Jaesik Yoon, Youngsung Son, and Sungjin Ahn. Sequential neural processes. *Advances in Neural Information Processing Systems*, 32:10254–10264, 2019.

Michele Tizzoni, Paolo Bajardi, Chiara Poletto, José J Ramasco, Duygu Balcan, Bruno Gonçalves, Nicola Perra, Vittoria Colizza, and Alessandro Vespignani. Real-time numerical forecast of global epidemic spreading: case study of 2009 a/h1n1pdm. *BMC medicine*, 10(1):165, 2012.

Alan Mathison Turing. The chemical basis of morphogenesis. *Bulletin of mathematical biology*, 52 (1):153–197, 1990.

Benjamin T Vincent and Tom Rainforth. The darc toolbox: automated, flexible, and efficient delayed and risky choice experiments using bayesian adaptive design. *PsyArXiv. October*, 20, 2017.

Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2020*, 2020.

Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. *Advances in neural information processing systems*, 30, 2017.

Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. Deep multi-view spatial-temporal network for taxi demand prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 5668–5675, 2019.

Qian Zhang, Kaiyuan Sun, Matteo Chinazzi, Ana Pastore y Piontti, Natalie E Dean, Diana Patricia Rojas, Stefano Merler, Dina Mistry, Piero Poletti, Luca Rossi, et al. Spread of Zika virus in the Americas. *Proceedings of the National Academy of Sciences*, 114(22):E4334–E4343, 2017.

Christoph Zimmer, Mona Meister, and Duy Nguyen-Tuong. Safe active learning for time-series modeling with gaussian processes. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 2735–2744, 2018.

# A THEORETICAL ANALYSIS

## A.1 LATENT INFORMATION GAIN

**Proposition 1.** *The expected information gain (EIG) for Neural Process is equivalent to the KL divergence between the prior and posterior in the latent process, that is*

$$\mathrm{EIG}(\hat{x}, \theta) := \mathbb{E}[H(\hat{x}) - H(\hat{x}|z, \theta)] = \mathbb{E}_{p(\hat{x}|\theta)}\left[\mathrm{KL}\big(p(z|\hat{x}, \theta) \| p(z)\big)\right] \tag{4}$$

*Proof of Proposition 1.* The information gained in the latent process $z$, by selecting the parameter $\theta$ and generate $\hat{x}$ is the reduction in entropy from the prior to the posterior $\mathrm{IG}(\theta) = H(\hat{x}) - H(\hat{x}|z, \theta)$. Take the expectation of $\mathrm{IG}(\hat{x}, \theta)$ under the marginal distribution, we obtain from the conditional independence of $z$ and $\theta$ that

$$\begin{aligned}
\mathbb{E}_{p(\hat{x}|\theta)}\left[\mathrm{KL}\big(p(z|\hat{x}, \theta) \| p(z)\big)\right] &= \mathbb{E}_{p(\hat{x}, z|\theta)}\left[\log \frac{p(z|\hat{x}, \theta)}{p(z)}\right] \\
&= \mathbb{E}_{p(\hat{x}, z|\theta)}\left[\log \frac{p(z|\hat{x}, \theta)}{p(z|\theta)}\right] \\
&= \mathbb{E}_{p(\hat{x}, z|\theta)}\big[\log p(z, \hat{x}, \theta) - \log p(\hat{x}, \theta) - \log p(z, \theta) + \log p(\theta)\big] \\
&= \mathbb{E}_{p(\hat{x}, z|\theta)}\big[\log p(\hat{x}|z, \theta) - \log p(\hat{x}|\theta)\big] \\
&= \mathbb{E}_{p(z)}\Big[\mathbb{E}_{p(\hat{x}|z, \theta)}[\log p(\hat{x}|z, \theta)] - \mathbb{E}_{p(\hat{x}|\theta)}[\log p(\hat{x}|\theta)]\Big] \\
&= \mathbb{E}_{p(z)}[H(\hat{x}|\theta) - H(\hat{x}|z, \theta)] \\
&= \mathrm{EIG}(\hat{x}, \theta).
\end{aligned}$$

□

## A.2 SAMPLE EFFICIENCY OF ACTIVE LEARNING

From the main text we know that in each round, the output random variable

$$X = \langle \Psi(\theta), z^* \rangle + \epsilon. \tag{5}$$

We further assume that the random noise $\epsilon$ is mean zero and $\sigma$-subGaussian.

Using this information, we treat $z$ as an unknown parameter and define a likelihood function so that $p(X|z; \theta)$ has good coverage over the observations:

$$p(X_k|z; \theta_k) \propto \exp\left(-\frac{1}{2\sigma^2}\left(X_k - \langle \Psi(\theta_k), z \rangle\right)^2\right).$$

Let the prior distribution over $z$ be $p(z|\theta_k) = p(z) \propto \exp\left(-\frac{m}{2\sigma^2}\|z\|^2\right)$. Here we use $k$ instead of $(i)$ in the Algorithm 1 to represent the number of iterations. We can form a posterior over $z$ in the $k$-th round:

$$p(z|X_1, \theta_1, \ldots, X_k, \theta_k) \propto \exp\left(-\frac{m}{2\sigma^2}\|z\|^2 - \frac{1}{2\sigma^2}\sum_{s=1}^{k}\left(X_s - \langle \Psi(\theta_s), z \rangle\right)^2\right).$$

Focusing on the random variable $z \sim p(\cdot|X_1, \theta_1, \ldots, X_k, \theta_k)$, the estimate of the hidden variable, we can express it at $k$-th round as:

$$z_k = \hat{z}_k + \sigma V_k^{-1} \eta_k, \tag{6}$$

where $\hat{z}_k = V_k^{-1} \sum_{s=1}^{k} X_s \Psi(\theta_s)$, $V_k = m\mathrm{I} + \sum_{s=1}^{k} \Psi(\theta_s)\Psi(\theta_s)^{\mathrm{T}}$, and $\eta_k$ is a standard normal random variable.

We can either choose action $\theta$ randomly or greedily. A random choice of $\theta$ corresponds to taking

$$\theta_k \sim \mathcal{N}(0, \mathrm{I}), \tag{7}$$

A greedy procedure is to choose action $\theta_k$ in the $k$-th round to optimize $\mathrm{KL}(p(z|\hat{x}, \theta) \| p(z)) = \mathbb{E}_{p(z|\hat{x},\theta)} \left( \log \frac{p(z|\hat{x},\theta)}{p(z)} \right)$, where we denote the estimated output variable $\hat{x}$ given $\theta$ and $z$ as $\hat{x} = \langle \Psi(\theta), z \rangle$. This optimization procedure is equivalent to maximizing the variance of the prediction:

$$\theta_k = \arg\max_{\theta \in \mathbb{R}^d} \mathbb{E}_{z \sim p(\cdot|X_1, \theta_1, \ldots, X_{k-1}, \theta_{k-1})} \left[ \left( \langle \Psi(\theta), z \rangle - \mathbb{E}_{z \sim p(\cdot|X_1, \theta_1, \ldots, X_{k-1}, \theta_{k-1})} \langle \Psi(\theta), z \rangle \right)^2 \right]. \tag{8}$$

For both approaches, we assume that the features $\Psi(\theta)$ are normalized.

We compare the statistical risk of this approach with the random sampling approach.

Assume that the features are normalized, so that for all $\theta \in \mathbb{R}^d$, $\Psi(\theta) \in \mathbb{S}^{d-1}$. Define a matrix $\mathrm{A}_k \in \mathbb{R}^{d \times k}$ containing all the column vectors $\{\Psi(\theta_1), \ldots, \Psi(\theta_k)\}$. We can then express the estimation error in the following lemma.

**Lemma 1.** *The estimation error $\|\hat{z}_k - z^*\|_2$ can be bounded as follow.*

$$\begin{aligned} \|\hat{z}_k - z^*\|_2 \leq\ & m \left( m + \sigma_{\min}\left( \mathrm{A}_k \mathrm{A}_k^{\mathrm{T}} \right) \right)^{-1} \cdot \|z^*\|_2 \\ & + \min \left\{ 1/\left(2\sqrt{m}\right), 1/\left( \sqrt{\sigma_{\min}\left( \mathrm{A}_k \mathrm{A}_k^{\mathrm{T}} \right)} + \frac{m}{\sqrt{\sigma_{\min}\left( \mathrm{A}_k \mathrm{A}_k^{\mathrm{T}} \right)}} \right) \right\} \cdot \sigma \sqrt{d}. \end{aligned}$$

We now analyze random sampling of $\theta$ versus greedy search for $\theta$.

If the feature map $\Psi(\cdot) = \mathrm{id}$, then from random matrix theory, we know that for $\theta$ randomly sampled from a normal distribution and normalized to $\|\theta\| = 1$, $\sigma_{\min}\left( \frac{1}{k} \mathrm{A}_k \mathrm{A}_k^{\mathrm{T}} \right)$ will converge to $\left( \sqrt{1/k} - \sqrt{1/d} \right)^2$ for large $k$, which is of order $\Omega(1/d)$. This will lead to an appealing risk bound for $\|\hat{z}_k - z^*\|_2$ on the order of $\mathcal{O}\left( d/\sqrt{k} \right)$.

However, in high dimension, this feature map is often far from identity. In the proof of Theorem 1 below, we demonstrate that even when $\Psi(\cdot)$ is simply a linear random feature map, with i.i.d. normal entries, random exploration in $\theta$ can lead to deteriorated error bound. This setting is motivated by the analyses in wide neural networks, where the features learned from gradient descent are close to those generated from random initialization Du et al. (2019); Mei & Montanari (2019).

**Theorem 1** (Formal statement)**.** *Assume that the noise $\epsilon$ in equation 5 is $\sigma$-subGaussian.*

*For a normalized linear random feature map $\Psi(\cdot)$, greedily optimizing the KL divergence, $\mathrm{KL}(p(z|\hat{x}, \theta) \| p(z))$ (or equivalently the variance of the posterior predictive distribution defined in equation equation 8) in search of $\theta$ will lead to an error $\|\hat{z}_k - z^*\|_2 = \mathcal{O}\left( \sigma d/\sqrt{k} \right)$ with high probability.*

*On the other hand, random sampling of $\theta$ following equation 7 will lead to $\|\hat{z}_k - z^*\|_2 = \mathcal{O}\left( \sigma d^2/\sqrt{k} \right)$ with high probability.*

*Proof of Theorem 1.* For a linear random feature map, we can express $\Psi(\theta) = \Psi\theta$, where entries in $\Psi \in \mathbb{R}^{d \times d}$ are i.i.d. normal. The entries of $\Psi\theta$ are then normalized.

- For random exploration of $\theta$, the matrix containing the feature vectors becomes $\mathrm{A}_k = \Psi\Theta_k$, where matrix $\Theta_k \in \mathbb{R}^{d \times k}$ collects all the $k$ column vectors of $\{\theta_1, \ldots, \theta_k\}$. Then $\mathrm{A}_k \mathrm{A}_k^{\mathrm{T}} = \Psi\Theta_k\Theta_k^{\mathrm{T}}\Psi^{\mathrm{T}}$. From random matrix theory, we know that the condition number of $\Psi$ is equal to $d$ with high probability Chen & Dongarra (2005). Hence for normalized $\Psi$ and $\theta$, $\sigma_{\min}\left( \Psi\Theta_k\Theta_k^{\mathrm{T}}\Psi^{\mathrm{T}} \right) \geq \sigma_{\min}^2(\Psi)\, \sigma_{\min}\left( \Theta_k\Theta_k^{\mathrm{T}} \right) = \frac{1}{d^2} \sigma_{\min}\left( \Theta_k\Theta_k^{\mathrm{T}} \right)$. The inequality holds because the smallest singular value is the inverse of the norm of the inverse matrix.

We then use the fact from random matrix theory that for normalized random $\theta$, the asymptotic distribution of the eigenvalues of $\frac{1}{k}\Theta_k\Theta_k^{\mathrm{T}}$ follow the (scaled) Marchenko–Pastur distribution, which is supported on $\lambda \in \left[\left(\sqrt{1/k}-\sqrt{1/d}\right)^2, \left(\sqrt{1/k}+\sqrt{1/d}\right)^2\right]$, where the $1/d$ scaling comes from the fact that $\theta$ is normalized Götze & Tikhomirov (2004). Hence for large $k$, $\sigma_{\min}\left(\Theta_k\Theta_k^{\mathrm{T}}\right) \geq \left(1-\sqrt{k/d}\right)^2$ with high probability. This combined with the previous paragraph yields that for the random feature model,

$$\sigma_{\min}\left(\mathrm{A}_k\mathrm{A}_k^{\mathrm{T}}\right) = \Omega\left(\frac{1}{d^2}\left(1-\sqrt{k/d}\right)^2\right)$$

with high probability. Plugging this result into Lemma 1, we obtain that the error $\|\hat{z}_k - z^*\|_2$ for random exploration in the space of $\theta$ is of order $\mathcal{O}\left(d^2/\sqrt{k}\right)$.

- We then analyze the error associated with greedy maximization of the posterior predictive variance. We first note that the variance of the posterior predictive distribution in equation equation 8 can be expressed as follows using equation equation 6:

$$\mathbb{E}\left[\left(\langle\Psi(\theta), z\rangle - \mathbb{E}\langle\Psi(\theta), z\rangle\right)^2\right] = \sigma^2\mathbb{E}\left[\left(\left\langle\Psi(\theta), V_{k-1}^{-1}\eta_k\right\rangle\right)^2\right] = \sigma^2\Psi(\theta)^{\mathrm{T}}V_{k-1}^{-2}\Psi(\theta), \tag{9}$$

where the expectations are with respect to $z \sim p(\cdot|X_1, \theta_1, \ldots, X_{k-1}, \theta_{k-1})$.

We perform a singular value decomposition $\mathrm{A}_k = U_k\Lambda_k W_k$. Then $\sum_{s=1}^k \Psi(\theta_s)\Psi(\theta_s)^{\mathrm{T}} = \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}} = U_k\Lambda_k\Lambda_k^{\mathrm{T}}U_k^{\mathrm{T}}$, and that $V_{k-1}^{-2} = \left(m\mathrm{I} + A_{k-1}A_{k-1}^{\mathrm{T}}\right)^{-2} = U_{k-1}\left(m\mathrm{I} + \Lambda_{k-1}\Lambda_{k-1}^{\mathrm{T}}\right)^{-2}U_{k-1}^{\mathrm{T}}$. Via this formulation, we see that maximizing $\Psi(\theta)^{\mathrm{T}}V_{k-1}^{-2}\Psi(\theta)$ in equation equation 9 to choose $\theta_k$ is equivalent to choosing $\Psi(\theta_k) = (U_{k-1})_{(\cdot,l)}^{\mathrm{T}}$, where $l = \arg\min_{i\in\{1,\ldots,d\}}\left(\Lambda_{k-1}\Lambda_{k-1}^{\mathrm{T}}\right)_{(i,i)}$. In words, when we use greedy method and maximize the variance of the prediction, it corresponds to taking $\Psi(\theta_k)$ in the direction of the smallest eigenvector of $V_{k-1}$.

Since every $\Psi(\theta)$ is normalized and we initialize uniformly: $V_0 = m\mathrm{I}$, the process is equivalent to scanning the orthogonal spaces of normalized vectors in $\mathbb{R}^d$ for $\lfloor k/d \rfloor$ times. For large $k$, entries in $\Lambda_k\Lambda_k^{\mathrm{T}}$ are approximately uniform and are all larger than or equal to $\lfloor k/d \rfloor$. Then $\sigma_{\min}\left(\mathrm{A}_k\mathrm{A}_k^{\mathrm{T}}\right) = \Omega(k/d)$. Plugging into the bound of Lemma 1, we obtain that

$$\|\hat{z}_k - z^*\|_2 = \mathcal{O}\left(\frac{\sigma d}{\sqrt{k}}\right).$$

□

*Proof of Lemma 1.* We first express the estimate $\hat{z}_k$ as follows.

$$\hat{z}_k = V_k^{-1}\sum_{s=1}^k X_s\Psi(\theta_s) = V_k^{-1}\sum_{s=1}^k \Psi(\theta_s)\Psi(\theta_s)^{\mathrm{T}}z^* + V_k^{-1}\sum_{s=1}^k \epsilon_s\Psi(\theta_s).$$

Then

$$\begin{aligned}\|\hat{z}_k - z^*\|_2 &= \left\|\left(V_k^{-1}\sum_{s=1}^k \Psi(\theta_s)\Psi(\theta_s)^{\mathrm{T}} - \mathrm{I}\right)z^* + V_k^{-1}\sum_{s=1}^k \epsilon_s\Psi(\theta_s)\right\|_2 \\ &\leq \underbrace{\left\|\left(V_k^{-1}\sum_{s=1}^k \Psi(\theta_s)\Psi(\theta_s)^{\mathrm{T}} - \mathrm{I}\right)z^*\right\|_2}_{T_1} + \underbrace{\left\|V_k^{-1}\sum_{s=1}^k \epsilon_s\Psi(\theta_s)\right\|_2}_{T_2}.\end{aligned}$$

Define a matrix $\mathrm{A}_k \in \mathbb{R}^{d\times k}$ containing all the column vectors $\{\Psi(\theta_1), \ldots, \Psi(\theta_k)\}$ and perform a singular value decomposition $\mathrm{A}_k = U_k\Lambda_k W_k$. Then $\sum_{s=1}^k \Psi(\theta_s)\Psi(\theta_s)^{\mathrm{T}} = \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}} = U_k\Lambda_k\Lambda_k^{\mathrm{T}}U_k^{\mathrm{T}}$,

and $V_k = m\mathrm{I} + \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}}$, We further define vector $e_k \in \mathbb{R}^s$ where $(e_k)_s = \epsilon_s$. We use this definition to simplify the two terms further.

For term $T_1$,

$$\begin{aligned}\left\| \left( V_k^{-1} \sum_{s=1}^{k} \Psi(\theta_s)\Psi(\theta_s)^{\mathrm{T}} - \mathrm{I} \right) z^* \right\|_2 &= m \left\| V_k^{-1} z^* \right\|_2 \\ &\leq m \left\| V_k^{-1} \right\|_2 \cdot \left\| z^* \right\|_2 \\ &= m \left( m + \sigma_{\min}\left( \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}} \right) \right)^{-1} \cdot \left\| z^* \right\|_2 .\end{aligned}$$

For term $T_2$, we define a diagonal matrix $\bar{\Lambda}_k \in \mathbb{R}^{k\times k}$ which satisfies $\left(\bar{\Lambda}_k\right)_{i,i} = 1$ if $i \leq d$ and $\left(\bar{\Lambda}_k\right)_{i,i} = 0$ if $i > d$, when $k > d$. The following bound on $T_2$ can be achieved.

$$\begin{aligned}\left\| V_k^{-1} \sum_{s=1}^{k} \epsilon_s \Psi(\theta_s) \right\|_2 &= \left\| V_k^{-1} \mathrm{A}_k e_k \right\|_2 \\ &= \left\| U_k \left( \Lambda_k\Lambda_k^{\mathrm{T}} + m\mathrm{I} \right)^{-1} U_k^{\mathrm{T}} U_k \Lambda_k \bar{\Lambda}_k W_k e_k \right\|_2 \\ &\leq \left\| U_k \left( \Lambda_k\Lambda_k^{\mathrm{T}} + m\mathrm{I} \right)^{-1} \Lambda_k \right\|_2 \cdot \left\| \bar{\Lambda}_k W_k e_k \right\|_2 \\ &= \left\| \left( \Lambda_k\Lambda_k^{\mathrm{T}} + m\mathrm{I} \right)^{-1} \Lambda_k \right\|_2 \cdot \left\| \bar{\Lambda}_k W_k e_k \right\|_2 \\ &\leq \min\left\{ 1/\left(2\sqrt{m}\right), 1/\left( \sqrt{\sigma_{\min}\left( \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}} \right)} + \frac{m}{\sqrt{\sigma_{\min}\left( \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}} \right)}} \right) \right\} \cdot \left\| \bar{\Lambda}_k W_k e_k \right\|_2 .\end{aligned}$$

Assuming that noise $\epsilon_s$ is $\sigma$-subGaussian, then so is $W_k e_k$ since $W_k$ is a unitary matrix. Multiplied by the diagonal matrix $\bar{\Lambda}_k$ which has zero, $\left\| \bar{\Lambda}_k W_k e_k \right\|_2 \leq \sigma\sqrt{d}$. Therefore,

$$\left\| V_k^{-1} \sum_{s=1}^{k} \epsilon_s \Psi(\theta_s) \right\|_2 \leq \min\left\{ 1/\left(2\sqrt{m}\right), 1/\left( \sqrt{\sigma_{\min}\left( \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}} \right)} + \frac{m}{\sqrt{\sigma_{\min}\left( \mathrm{A}_k\mathrm{A}_k^{\mathrm{T}} \right)}} \right) \right\} \cdot \sigma\sqrt{d}.$$

□

# B Experiment Details

## B.1 SEIR Model

Our SEIR simulator is a simple stochastic, discrete, chain-binomial compartmental model. In this model, susceptible individuals ($S$) become exposed ($E$) through interactions with infectious individuals ($I$). Exposed individuals which are infected but not yet infectious transition to infectious compartment at a rate $\varepsilon$ that is inversely proportional to the latent period of the disease. Lastly, infectious individuals transition to the removed compartment at a rate $\mu$ which is inversely proportional to the infectious period. Removed individuals ($R$) are assumed to be no longer infectious and they are to be considered either recovered or dead. All transitions are simulated by randomly drawn from a binomial distribution.

## B.2 LEAM-US Model

LEAM-US integrates a human mobility layer, represented as a network, using both short-range (i.e., commuting) and long-range (i.e., flights) mobility data. Commuting flows between counties are obtained from the 2011-2015 5-Year ACS Commuting Flows survey and properly adjusted to account for differences in population totals since the creation of the dataset. Instead, long-range air traveling flows are quantified using origin-destination daily passenger flows between airport pairs as reported by the Official Aviation Guide (OAG) and IATA databases (updated in 2021) (OAG,

Aviation Worlwide Limited, 2021; IATA, International Air Transport Association, 2021). In addition, flight probabilities are age and country specific.

The model is initialized using a multi-scale modeling approach that utilizes GLEAM, the Global and Epidemic Mobility model (Balcan et al., 2009; 2010; Tizzoni et al., 2012; Zhang et al., 2017; Chinazzi et al., 2020; Davis et al., 2020), to simulate a set of 500 different initial conditions for LEAM-US starting on February 16th, 2020. The disease dynamics are modeled using a classic SEIR-like model and initial conditions are determined using the Global and Epidemic Mobility model (Balcan et al., 2009; 2010; Tizzoni et al., 2012; Zhang et al., 2017) calibrated to realistically represent the evolution of the COVID-19 pandemic (Chinazzi et al., 2020; Davis et al., 2020). Lastly, travel restrictions, mobility reductions, and government interventions are explicitly modeled to mimic the real timeline of interventions of the events that occurred during the COVID-19 pandemic.

### B.3 Spatiotemporal NP Model

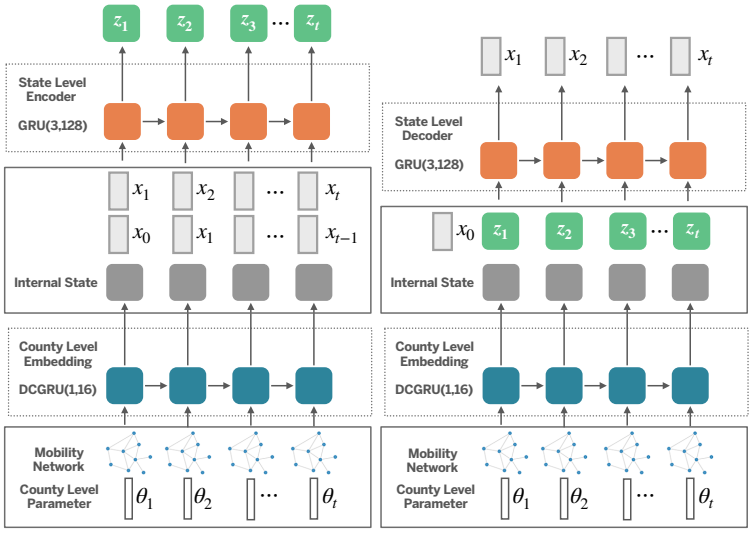


Figure 6: Visualization of the STNP model architecture. For both the encoder and the decoder, we use a diffusion convolutional GRU (DCGRU) Li et al. (2018) to capture spatiotemporal dependency.

As shown in Figure 6, our model has $\theta$ at both county and state level and $x_t$ at the state level. We use county-level parameter $\theta$ together with a county-to-county mobility graph $A$ as input. We use the DCGRU layer (Li et al., 2017) to encode the graph in a GRU. We use a linear layer to map the county-level output to hidden features at the state level. For both the state-level encoder and decoder, we use multi-layer GRUs.

The input $\theta_{1:t}$ is the county-level parameters for LEAM-US with a dimension of 10. The county level embedding uses 1 layer DCGRU with a width of 16. The internal state is at state level with dimension of 16. The state level encoder and decoder use 3 layer GRUs with width of 128. The dimension of the latent process $z_{1:t}$ is 32. The dimension of output $x_{1:t}$ is 24, including the incidence and prevalence for 12 compartments. We trained STNP model for 500 steps with learning rate fixed at $10^{-3}$ using Adam optimizer. We perform early stopping with 50 patience for both offline learning and Bayesian active learning.

### B.4 Acquisition Function

**Maximum Mean STD.** Mean STD (Gal & Ghahramani, 2016) is a heuristic used to estimate the model uncertainty. For each augmented parameter $\theta$, we sample multiple $z_{1:T}$ and generate a set of predictions $\{\hat{x}_{1:T}\}$. For a length $T$ sequence with dimension $D$, we compute the standard deviation $\sigma_{t,d}$ for time step $t$ and feature $d$. Mean STD computes $\bar{\sigma} = \frac{1}{TD}\sum_{t=1}^{T}\sum_{d=1}^{D}\sum \sigma_{t,d}$ for each parameter $\theta$. We select the $\theta$ with the maximum $\bar{\sigma}$. Empirically, we found that Mean STD often becomes over-conservative and tends to explore less.

Table 2: Performance comparison of different acquisition functions in NP model for SEIR simulator

| Percentage of samples | LIG | EIG | Random | MeanSTD | MaxEntropy |
|---|---|---|---|---|---|
| 1.11% | **365.87** $\pm$ 142.87 | 435.08 $\pm$ 32.38 | 480.68 $\pm$ 5.24 | 480.22 $\pm$ 12.63 | 427.73 $\pm$ 61.36 |
| 1.85% | **236.9** $\pm$ 50.6 | 340.27 $\pm$ 30.84 | 398.33 $\pm$ 131.05 | 314.75 $\pm$ 111.42 | 302.24 $\pm$ 119.84 |
| 2.96% | **119.26** $\pm$ 14.22 | 291.15 $\pm$ 10.60 | 244.27 $\pm$ 148.89 | 158.94 $\pm$ 36.6 | 186.88 $\pm$ 57.48 |
| 4.07% | **96.73** $\pm$ 17.07 | 261.60 $\pm$ 7.78 | 116.8 $\pm$ 9.1 | 127.36 $\pm$ 27.97 | 146.72 $\pm$ 26.06 |

Table 3: Performance comparison of different acquisition functions in GP model for SEIR simulator

| Percentage of samples | Random | MeanSTD | MaxEntropy |
|---|---|---|---|
| 1.11% | 663.76 $\pm$ 46.36 | 606.81 $\pm$ 6.89 | **586.25** $\pm$ 58.44 |
| 1.85% | 637.12 $\pm$ 13.45 | **619.15** $\pm$ 36.42 | 628.54 $\pm$ 71.34 |
| 2.96% | 597.3 $\pm$ 19.59 | 589.72 $\pm$ 24.9 | **568.84** $\pm$ 19.05 |
| 4.07% | **519.98** $\pm$ 17.86 | 530.07 $\pm$ 32.95 | 578.34 $\pm$ 68.7 |

Table 4: Performance comparison of different acquisition functions in STNP model for RD simulator

| Percentage of samples | LIG | EIG | Random | MeanSTD | MaxEntropy |
|---|---|---|---|---|---|
| 6.25% | **4.562** $\pm$ 0.114 | 4.861 $\pm$ 0.433 | 5.325 $\pm$ 0.361 | 5.264 $\pm$ 0.298 | 4.826 $\pm$ 0.336 |
| 12.50% | **3.841** $\pm$ 0.253 | 4.590 $\pm$ 0.529 | 4.179 $\pm$ 0.045 | 4.157 $\pm$ 0.252 | 4.084 $\pm$ 0.042 |
| 18.75% | **3.165** $\pm$ 0.142 | 4.162 $\pm$ 0.696 | 3.602 $\pm$ 0.182 | 3.675 $\pm$ 0.229 | 3.694 $\pm$ 0.140 |
| 25.00% | **2.415** $\pm$ 0.083 | 3.993 $\pm$ 0.847 | 3.140 $\pm$ 0.165 | 3.339 $\pm$ 0.111 | 3.302 $\pm$ 0.284 |
| 31.25% | **2.302** $\pm$ 0.007 | 3.714 $\pm$ 0.861 | 2.561 $\pm$ 0.243 | 2.791 $\pm$ 0.072 | 2.912 $\pm$ 0.473 |

Table 5: Performance comparison of different acquisition functions in STNP model for LEAM-US simulator, population divided by 1000.

| Percentage of samples | LIG | EIG | Random | MeanSTD | MaxEntropy |
|---|---|---|---|---|---|
| 11.1% | **14.447** $\pm$ 1.087 | 19.067 $\pm$ 3.981 | 20.961 $\pm$ 5.548 | 35.356 $\pm$ 28.706 | 65.498 $\pm$ 13.324 |
| 13.7% | **11.704** $\pm$ 0.216 | 16.372 $\pm$ 3.663 | 13.418 $\pm$ 0.815 | 16.092 $\pm$ 3.11 | 30.496 $\pm$ 24.333 |
| 21.3% | **7.593** $\pm$ 0.822 | 11.754 $\pm$ 1.713 | 9.332 $\pm$ 0.601 | 11.191 $\pm$ 0.184 | 10.028 $\pm$ 2.065 |
| 28.9% | **6.539** $\pm$ 0.618 | 9.455 $\pm$ 0.595 | 8.077 $\pm$ 0.657 | 7.908 $\pm$ 0.536 | 8.417 $\pm$ 0.616 |
| 36.5% | **6.008** $\pm$ 1.079 | 8.596 $\pm$ 0.741 | 6.719 $\pm$ 0.383 | 7.533 $\pm$ 0.861 | 7.431 $\pm$ 0.776 |

**Maximum Entropy.** Maximum entropy computes the maximum predictive entropy as $H(\hat{x}) = -\mathbb{E}[\log p(\hat{x}_{1:T})]$. In general, entropy is intractable for continuous output. Our NP model implicitly assumes the predictions follow a multivariate Gaussian, which allows us to compute the differential entropy (Jaynes, 1957). We follow the same procedure as Mean STD to estimate the empirical covariance $\Sigma \in \mathbb{R}^{TD \times TD}$ and compute the differential entropy for each parameter as $H = \frac{1}{2} \ln |\Sigma| + \frac{TD}{2}(1 + \ln 2\pi)$. We select the parameter $\theta$ with the maximum entropy.

### B.5 Implementation Details

For both GP and INP model mimicking SEIR simulation, we ran experiments using CPU. No GPU accelerator is needed for this simple model. It takes 5 hours to converge. For INP model mimicking LEAM-US simulation, we ran experiments with GEFORCE RTX 2080. It takes one day for the training to converge. For all experiments, we run with three different random seeds.

## C Additional Results

### C.1 INP Model and GP Model

Table 2 shows the average results together with the standard deviation of INP model for SEIR simulator after running experiments three times. Table 3 shows the average results together with the

standard deviation of GP model for SEIR simulator. Table 5 shows the average results together with the standard deviation of INP model for LEAM-US simulator.

### C.2 Batch Active Learning with LIG.

Figure 7 compares 4 different setups: 8 batches (size 1), 4 batches (size 2), 2 batches (size 4), and 1 batch (size 8).

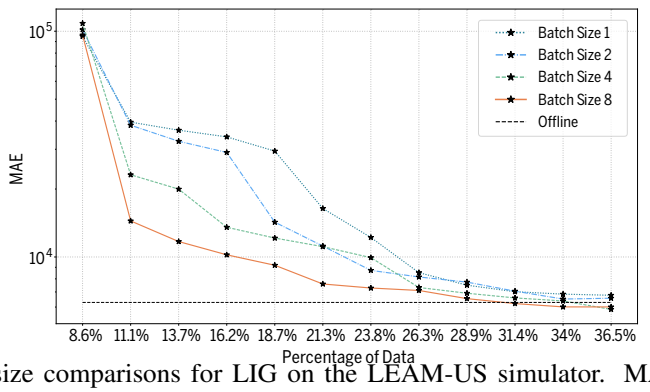


Figure 7: Batch size comparisons for LIG on the LEAM-US simulator. MAE loss versus the percentage of samples for INP during Bayesian active learning.