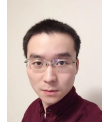
Doubly Stochastic Variational Inference for Neural Processes with Hierarchical Latent Variables

Q. Wang & Herke van Hoof



Amsterdam Machine Learning Lab

ICML 2020

Universiteit van Amsterdam

Highlights in this Work

Highlights in this Work

- A systematical revisit to SPs with an Implicit Latent Variable Model
  - conceptualization of latent SP models
  - comprehension about SPs with LVMs

Highlights in this Work

- A systematical revisit to SPs with an Implicit Latent Variable Model
  - conceptualization of latent SP models
  - comprehension about SPs with LVMs
- A novel exchangeable SP within a Hierarchical Bayesian Framework
  - formalization of a hierarchical SP
  - plausible approximate inference method

Highlights in this Work

- A systematical revisit to SPs with an Implicit Latent Variable Model
  - conceptualization of latent SP models
  - comprehension about SPs with LVMs
- A novel exchangeable SP within a Hierarchical Bayesian Framework
  - formalization of a hierarchical SP
  - plausible approximate inference method
- Competitive performance on extensive Uncertainty-aware Applications
  - high dimensional regressions on simulators/real-world dataset
  - classification and o.o.d. detection on image dataset

Outline of this Talk

1. Motivation for SPs
2. Study of SPs with LVMs
3. NP with Hierarchical Latent Variables
4. Experiments and Applications

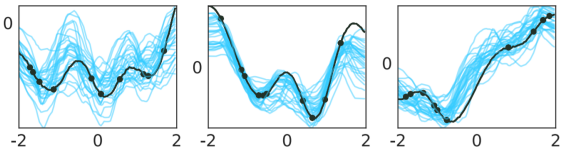
Motivation for SPs

Why Do We Need *Stochastic Processes*?

The stochastic process (SP) is a math tool to describe the distribution over functions. (Fig. refers to [1])

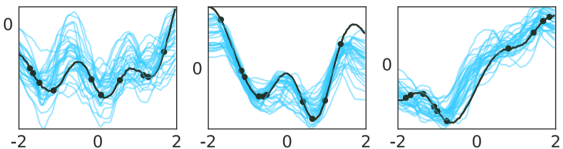
Why Do We Need *Stochastic Processes*?

The stochastic process (SP) is a math tool to describe the distribution over functions. (Fig. refers to [1])



Why Do We Need *Stochastic Processes*?

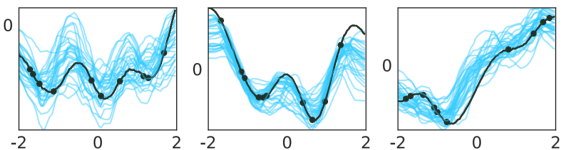
The stochastic process (SP) is a math tool to describe the distribution over functions. (Fig. refers to [1])



- Flexible to handle correlations among samples : significant for non-*i.i.d.* dataset ;

Why Do We Need *Stochastic Processes*?

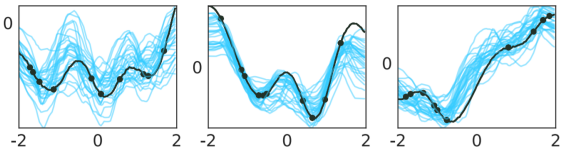
The stochastic process (SP) is a math tool to describe the distribution over functions. (Fig. refers to [1])



- Flexible to handle correlations among samples : significant for non-*i.i.d.* dataset ;
- Quantify uncertainty in risk-sensitive applications : *e.g.* forecast $p(s_{t+1} j s_t; a_t)$ in autonomous driving [2] ;

Why Do We Need *Stochastic Processes*?

The stochastic process (SP) is a math tool to describe the distribution over functions. (Fig. refers to [1])



- Flexible to handle correlations among samples : significant for non-*i.i.d.* dataset ;
- Quantify uncertainty in risk-sensitive applications : *e.g.* forecast $p(s_{t+1}|s_t; a_t)$ in autonomous driving [2] ;
- Model distributions instead of point estimates : working as a *generative model* for more realizations [3].

Two Consistencies in Exchangeable SPs

Some required properties for *exchangeable stochastic process* [4] :

Two Consistencies in Exchangeable SPs

Some required properties for *exchangeable stochastic process* [4] :

- **Marginalization Consistency.** For any finite collection of random variables $fy_1; y_2; \dots; y_{N+M}g$, the probability after *marginalization* over subset is unchanged.

$$\int x_{1:N+M}(y_{1:N+M})dy_{N+1:N+M} = x_{1:N}(y_{1:N}) \tag{1.1}$$

- **Exchangeability Consistency.** Any random *permutation* over set of variables does not influence joint probability.

$$x_{1:N}(y_{1:N}) = x_{(1:N)}(y_{(1:N)}) \tag{1.2}$$

Two Consistencies in Exchangeable SPs

Some required properties for *exchangeable stochastic process* [4] :

- **Marginalization Consistency.** For any finite collection of random variables $fy_1; y_2; \dots; y_{N+M}g$, the probability after *marginalization* over subset is unchanged.

$$\int x_{1:N+M}(y_{1:N+M})dy_{N+1:N+M} = x_{1:N}(y_{1:N}) \tag{1.1}$$

- **Exchangeability Consistency.** Any random *permutation* over set of variables does not influence joint probability.

$$x_{1:N}(y_{1:N}) = x_{(1:N)}(y_{(1:N)}) \tag{1.2}$$

With these two conditions, an exchangeable SP can be induced. (Refer to *Kolmogorov Extension Theorem*)

SPs in Progress and Primary Concerns

Crucial properties for SPs :

- Scalability in large-scale dataset:
- Flexibility in distributions:
- Extension to high dimensions:

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

SPs in Progress and Primary Concerns

Crucial properties for SPs :

- Scalability in large-scale dataset:
  ! Optimization/Computational bottleneck
- Flexibility in distributions:
- Extension to high dimensions:

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

*SP*s in Progress and Primary Concerns

Crucial properties for *SP*s :

- Scalability in large-scale dataset:
  ! Optimization/Computational bottleneck
- Flexibility in distributions:
  ! Non-Gaussian or Multi-modal property
- Extension to high dimensions:

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

*SP*s in Progress and Primary Concerns

Crucial properties for *SP*s :

- Scalability in large-scale dataset:
  ! Optimization/Computational bottleneck
- Flexibility in distributions:
  ! Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  ! Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
- Neural Processes (NPs)

SPs in Progress and Primary Concerns

Crucial properties for SPs :

- Scalability in large-scale dataset:
  ! Optimization/Computational bottleneck
- Flexibility in distributions:
  ! Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  ! Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  ! less scalable with computational complexity $O(N^3)$
- Neural Processes (NPs)

SPs in Progress and Primary Concerns

Crucial properties for SPs :

- Scalability in large-scale dataset:
  ! Optimization/Computational bottleneck
- Flexibility in distributions:
  ! Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  ! Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  ! less scalable with computational complexity $O(N^3)$
  ! less flexible with Gaussian distributions
- Neural Processes (NPs)

SPs in Progress and Primary Concerns

Crucial properties for SPs :

- Scalability in large-scale dataset:
  ! Optimization/Computational bottleneck
- Flexibility in distributions:
  ! Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  ! Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  ! less scalable with computational complexity $O(N^3)$
  ! less flexible with Gaussian distributions
- Neural Processes (NPs)
  ! more scalable with computational complexity $O(N)$

SPs in Progress and Primary Concerns

Crucial properties for SPs :

- Scalability in large-scale dataset:
  ! Optimization/Computational bottleneck
- Flexibility in distributions:
  ! Non-Gaussian or Multi-modal property
- Extension to high dimensions:
  ! Correlations among or across Input/Output

Analysis on GPs/NPs :

- Gaussian Processes (GPs)
  ! less scalable with computational complexity $O(N^3)$
  ! less flexible with Gaussian distributions
- Neural Processes (NPs)
  ! more scalable with computational complexity $O(N)$
  ! more flexible with no explicit distributions

Study of SPs with LVMs

Deep Latent Variable Model as SPs

Here we present an implicit Latent Variable Model for SPs :

- Generation paradigm with (potentially correlated) latent variables :

- Predictive distribution in SPs : Let the *context* and *target input* be $C = f(x_i; y_i)ji = 1; 2; : : : ; Ng$ and x_T, the computation is

(2.3)

mostly intractable.

Deep Latent Variable Model as SPs

Here we present an implicit Latent Variable Model for SPs :

- Generation paradigm with (potentially correlated) latent variables :

$$\underset{\text{index depend. l.v.}}{z_i} = \underset{\text{deter. term}}{(x_i)} + \underset{\text{stoch. term}}{(x_i)} \tag{2.1}$$

- Predictive distribution in SPs : Let the *context* and *target input* be $C = f(x_i; y_i)ji = 1; 2; : : : ; Ng$ and x_T, the computation is

(2.3)

mostly intractable.

Deep Latent Variable Model as SPs

Here we present an implicit Latent Variable Model for SPs :

- Generation paradigm with (potentially correlated) latent variables :

$$z_i = (x_i) + (x_i) \tag{2.1}$$

index depend. l.v. | deter. term | stoch. term

$$y_i = '(x_i; z_i) + \ ; \tag{2.2}$$

obs: | trans. | obs. noise

- Predictive distribution in SPs : Let the *context* and *target input* be $C = f(x_i; y_i) ji = 1; 2; : : : ; Ng$ and x_T, the computation is

(2.3)

mostly intractable.

Deep Latent Variable Model as SPs

Here we present an implicit Latent Variable Model for SPs :

- Generation paradigm with (potentially correlated) latent variables :

$$\underset{\text{index depend. l.v.}}{z_i} = \underset{\text{deter. term}}{(x_i)} + \underset{\text{stoch. term}}{(x_i)} \tag{2.1}$$

$$\underset{\textit{obs:}}{y_i} = \underset{\text{trans.}}{'(x_i; z_i)} + \underset{\text{obs. noise}}{i} \tag{2.2}$$

- Predictive distribution in SPs : Let the *context* and *target input* be $C = f(x_i; y_i) ji = 1; 2; : : : ; Ng$ and x_T, the computation is

$$p\ (z_T j x_C; y_C; x_T) = \frac{p(z_C; z_T)}{\int p(z_C; z_T) dz_C}; \tag{2.3}$$

  mostly intractable.

Deep Latent Variable Model as SPs

Here we present an implicit Latent Variable Model for SPs :

- Generation paradigm with (potentially correlated) latent variables :

$$z_i = (x_i) + (x_i) \tag{2.1}$$

index depend. l.v. — deter. term — stoch. term

$$y_i = '(x_i; z_i) + i \tag{2.2}$$

obs: — trans. — obs. noise

- Predictive distribution in SPs : Let the *context* and *target input* be $C = f(x_i; y_i) ji = 1; 2; : : : ; Ng$ and x_T, the computation is

$$p\ (z_T j x_C; y_C; x_T) = \int \frac{p(z_C; z_T)}{p(z_C; z_T) dz_C}; \quad y_T \quad p(y_T j x_T; z_T; \ ) \tag{2.3}$$

mostly intractable.

Gaussian Processes & Neural Processes

NP family approximates SPs in the form of LVMs :

- GP as an exchangeable SP with latent variables :

- NP as an exchangeable SP with a global latent variable :

NP family approximates SPs in the form of LVMs :

- GP as an exchangeable SP with latent variables :

$$_x(y) = \int N(y; z; \quad ^1 I) \underbrace{N(z; m(x); K(:;:))}_{\text{l.v.}} dz \tag{2.4}$$

- NP as an exchangeable SP with a global latent variable :

NP family approximates SPs in the form of LVMs :

- GP as an exchangeable SP with latent variables :

$$x(y) = \int N(y; z; {}^{1}I)\, \underbrace{N(z; m(x); K(:;:))}_{\text{l.v.}}\, dz \tag{2.4}$$

- NP as an exchangeable SP with a global latent variable :

$$x_{1:N+M}(y_{1:N+M}) = \int \prod_{i=1}^{N+M} \underbrace{p(y_i | x_i; z_G)}_{\text{trans.}}\, \underbrace{p(z_G)}_{\text{global l.v.}}\, dz_G \tag{2.5}$$

NP family approximates SPs in the form of LVMs :

- GP as an exchangeable SP with latent variables :

$$x(y) = \int N(y; z; {}^{1}I)\, \underbrace{N(z; m(x); K(:;:))}_{\text{l.v.}}\, dz \tag{2.4}$$

- NP as an exchangeable SP with a global latent variable :

$$x_{1:N+M}(y_{1:N+M}) = \int \prod_{i=1}^{N+M} \underbrace{p(y_i | x_i; z_G)}_{\text{trans.}}\, \underbrace{p(z_G)}_{\text{global l.v.}}\, dz_G \tag{2.5}$$

Remark

Some other models, such as Hierarchical GPs [5] and Deep GPs [6], [7] can also be expressed with LVMs.

Inference for Neural Processes

A general ELBO with **a context prior** in NP models [1] :

Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :

Inference for Neural Processes

A general ELBO with **a context prior** in NP models [1] :

$$\ln p(y_T | x_C, y_C, x_T) \geq \mathbb{E}_q \ln \underbrace{p(y_T | x_T, z_G)}_{\text{data likelihood}} - D_{KL}\left[\underbrace{q(z_G | x_C, y_C, x_T, y_T)}_{\text{global posterior}} \,\|\, \underbrace{p(z_G | x_C, y_C)}_{\text{global prior}}\right] \tag{2.6}$$

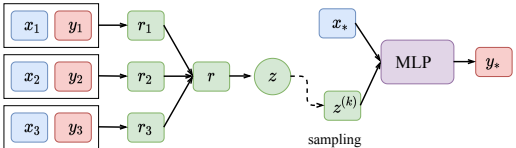
Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :

Inference for Neural Processes

A general ELBO with **a context prior** in NP models [1] :

$$\ln p(y_T | x_C, y_C, x_T) \geq \mathbb{E}_q \Big[\ln \underbrace{p(y_T | x_T, z_G)}_{\text{data likelihood}} \Big] - D_{KL} \Big[\underbrace{q(z_G | x_C, y_C, x_T, y_T)}_{\text{global posterior}} \,\|\, \underbrace{p(z_G | x_C, y_C)}_{\text{global prior}} \Big] \tag{2.6}$$

Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :



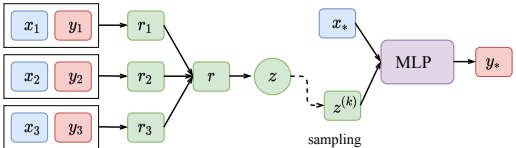
Inference for Neural Processes

A general ELBO with **a context prior** in NP models [1] :

$$\ln p(y_T|x_C, y_C, x_T) \geq \mathbb{E}_q \Big[\ln \underbrace{p(y_T|x_T, z_G)}_{\text{data likelihood}} \Big] - D_{KL}\Big[\underbrace{q(z_G|x_C, y_C, x_T, y_T)}_{\text{global posterior}} \,\|\, \underbrace{p(z_G|x_C, y_C)}_{\text{global prior}} \Big] \tag{2.6}$$

Statistics of the context invariant to the order in set instances, such as pooling of element-wise embeddings :

$$r_i = h(x_i, y_i); \quad r = \bigoplus_{i=1}^{M} r_i; \quad p(z_C|x_C, y_C) = N(z_C|[f(r), f(r)]) \tag{2.7}$$



NPs with Hierarchical Latent Variables

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures *!* more expressiveness.

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures *!* more expressiveness.
- Involving local l.v. *!* reveal local dependencies across input/output in high-dim cases.

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures *!* more expressiveness.
- Involving local l.v. *!* reveal local dependencies across input/output in high-dim cases.

As a result, a hierarchical LVM is induced as Doubly Stochastic Variational Neural Process (DSVNP):

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures ! more expressiveness.
- Involving local l.v. ! reveal local dependencies across input/output in high-dim cases.

As a result, a hierarchical LVM is induced as Doubly Stochastic Variational Neural Process (DSVNP):

$$x_{1:N+M}(y_{1:N+M}) = \int\int \prod_{i=1}^{N+M} p(y_i | z_G; z_i; x_i) \, p(z_i | x_i; z_G) p(z_G) dz_{1:N+M} dz_G \tag{3.1}$$

Extending NPs from A Hierarchical Bayes Perspective

Our work starts with motivations:

- Hierarchical Bayesian structures ! more expressiveness.
- Involving local l.v. ! reveal local dependencies across input/output in high-dim cases.

As a result, a hierarchical LVM is induced as Doubly Stochastic Variational Neural Process (DSVNP):

$$x_{1:N+M}(y_{1:N+M}) = \int\int \prod_{i=1}^{N+M} p(y_i | z_G; z_i; x_i) \, p(z_i | x_i; z_G) p(z_G) dz_{1:N+M} dz_G \tag{3.1}$$

Remark

DSVNP satisfies **Marginalization** and **Exchangeability Consistencies**, so it is a new exchangeable SP.

Approximate Inference for DSVNP

Exact inference for this hierarchical LVM is mostly intractable, hence approximate inference is used here.

- Evidence Lower Bound for DSVNP :

- Generative (Black Lines) and Recognition Models (Blue/Pink Lines) in Graphs : Specify generative process with black line

Approximate Inference for DSVNP

Exact inference for this hierarchical LVM is mostly intractable, hence approximate inference is used here.

- Evidence Lower Bound for DSVNP :

$$\begin{aligned} \ln\ p(y\ jx_C; y_C; x\ ) \qquad & \mathrm{E}_{q\ _{1;1}} \mathrm{E}_{q\ _{2;1}} \ln[p(y\ jz_G; z\ ; x\ )] \\ & \mathrm{E}_{q\ _{1;1}}[D_{KL}[q\ _{2;1}(z\ jz_G; x\ ; y\ )\ k\ p\ _{2;2}(z\ jz_G; x\ )] \\ & D_{KL}\ q\ _{1;1}(z_G jx_C; y_C; x_T; y_T)\ k\ p\ _{1;2}(z_G jx_C; y_C) \end{aligned} \tag{3.2}$$

- Generative (Black Lines) and Recognition Models (Blue/Pink Lines) in Graphs : Specify generative process with black line

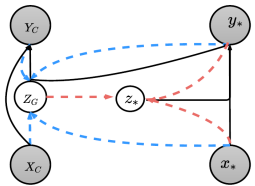
Approximate Inference for DSVNP

Exact inference for this hierarchical LVM is mostly intractable, hence approximate inference is used here.

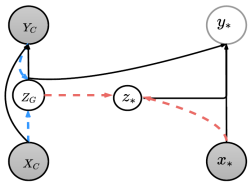
- Evidence Lower Bound for DSVNP :

$$\ln\ p(y\ jx_C; y_C; x\ ) \qquad \mathrm{E}_{q\ _{1;1}} \mathrm{E}_{q\ _{2;1}} \ln[p(y\ jz_G; z\ ; x\ )]$$
$$\mathrm{E}_{q\ _{1;1}} [D_{KL}[q\ _{2;1}(z\ jz_G; x\ ; y\ )\ k\ p\ _{2;2}(z\ jz_G; x\ )] \tag{3.2}$$
$$D_{KL}\ q\ _{1;1}(z_G jx_C; y_C; x_T; y_T)\ k\ p\ _{1;2}(z_G jx_C; y_C)$$

- Generative (Black Lines) and Recognition Models (Blue/Pink Lines) in Graphs : Specify generative process with black line



Training Process



Testing Process

Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

- Testing/Forecasting with priors and Monte Carlo estimates :

Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

Algorithm 1 Variational Inference for DSVNP in Training.

Input: Dataset $\mathcal{D} = \{x_C, y_C; x_T, y_T\}$, Maximum context points N_{max}, etc.
Output: Model parameters ϕ_1, ϕ_2 and θ.
for $i = 1$ **to** m **do**
  Draw some context number $N_C \sim U[1, N_{max}]$;
  Draw mini-batch pair instances $\{(x_C, y_C, x_T, y_T)_{bs}\}_{bs=1}^{B} \sim \mathcal{D}$;
  Feedforward instances to recognition model q_{ϕ_1};
  Feedforward latent variables to generative model p_θ;
  Update parameters by Optimizing Eq. (12):
    $\phi_1 \leftarrow \phi_1 + \alpha \nabla_{\phi_1} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{1,1}, \phi_{1,2}]$
    $\phi_2 \leftarrow \phi_2 + \alpha \nabla_{\phi_2} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{2,1}, \phi_{2,2}]$
    $\theta \leftarrow \theta + \alpha \nabla_{\theta} \mathcal{L}_{MC}$
end for

- Testing/Forecasting with priors and Monte Carlo estimates :

Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

Algorithm 1 Variational Inference for DSVNP in Training.

Input: Dataset $\mathcal{D} = \{x_C, y_C; x_T, y_T\}$, Maximum context points N_{max}, etc.
Output: Model parameters ϕ_1, ϕ_2 and θ.
for $i = 1$ **to** m **do**
  Draw some context number $N_C \sim U[1, N_{max}]$;
  Draw mini-batch pair instances $\{(x_C, y_C, x_T, y_T)_{bs}\}_{bs=1}^{B} \sim \mathcal{D}$;
  Feedforward instances to recognition model q_{ϕ_1};
  Feedforward latent variables to generative model p_θ;
  Update parameters by Optimizing Eq. (12):
    $\phi_1 \leftarrow \phi_1 + \alpha \nabla_{\phi_1} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{1,1}, \phi_{1,2}]$
    $\phi_2 \leftarrow \phi_2 + \alpha \nabla_{\phi_2} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{2,1}, \phi_{2,2}]$
    $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_{MC}$
end for

- Testing/Forecasting with priors and Monte Carlo estimates :

$$p(y\ jx_C; y_C; x\ ) \quad \frac{1}{KS} \sum_{k=1}^{K} \sum_{s=1}^{S} p\ (y\ jx\ ; z^{(s)}; z_G^{(k)}) \tag{3.3}$$

Training and Testing in Practice

Similar to that in NPs, DSVNP is trained in a SGVB way [8].

- Scalable training with random context points :

Algorithm 1 Variational Inference for DSVNP in Training.

Input: Dataset $\mathcal{D} = \{x_C, y_C; x_T, y_T\}$, Maximum context points N_{max}, etc.
Output: Model parameters ϕ_1, ϕ_2 and θ.
for $i = 1$ **to** m **do**
  Draw some context number $N_C \sim U[1, N_{max}]$;
  Draw mini-batch pair instances $\{(x_C, y_C, x_T, y_T)_{bs}\}_{bs=1}^{B} \sim \mathcal{D}$;
  Feedforward instances to recognition model q_{ϕ_1};
  Feedforward latent variables to generative model p_θ;
  Update parameters by Optimizing Eq. (12):
    $\phi_1 \leftarrow \phi_1 + \alpha \nabla_{\phi_1} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{1,1}, \phi_{1,2}]$
    $\phi_2 \leftarrow \phi_2 + \alpha \nabla_{\phi_2} \mathcal{L}_{MC}$ ▷ $\phi_1 = [\phi_{2,1}, \phi_{2,2}]$
    $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}_{MC}$
end for

- Testing/Forecasting with priors and Monte Carlo estimates :

$$p(y\ jx_C; y_C; x\ ) \quad \frac{1}{KS} \sum_{k=1}^{K} \sum_{s=1}^{S} p\ (y\ jx\ ; z^{(s)}; z_G^{(k)}) \tag{3.3}$$

using latent variables sampled in prior networks as $z_G^{(k)} \quad p\ _{1;2}(z_G jx_C; y_C)$ and $z^{(s)} \quad p\ _{2;2}(z\ jz_G^{(k)}; x\ )$.

Experiments and Applications

Toy Experiments

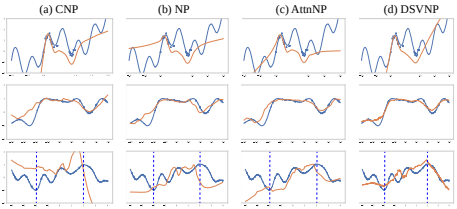
Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :

- Episdemic uncertainty in a single curve :
- Interpolation in curves of a SP:
- Extrapolation in curves of a SP:

Toy Experiments

Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :

- Episdemic uncertainty in a single curve : NP/AttnNP *!* over-confident in some regions



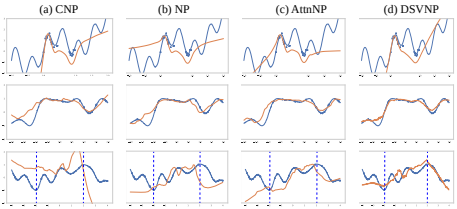
- Interpolation in curves of a SP:

- Extrapolation in curves of a SP:

Toy Experiments

Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :

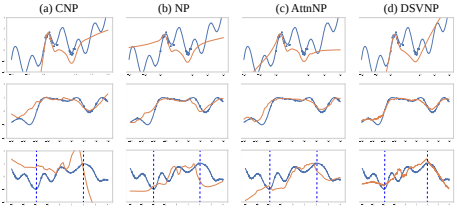
- Episdemic uncertainty in a single curve : NP/AttnNP *!* over-confident in some regions
- Interpolation in curves of a SP: AttnNP DSVNP NP CNP (Fitting/UQ Performance)
- Extrapolation in curves of a SP:



Toy Experiments

Discoveries in 1-D Simulation Experiments in terms of fitting errors and uncertainty quantification (UQ) :

- Episdemic uncertainty in a single curve : NP/AttnNP *!* over-confident in some regions
- Interpolation in curves of a SP: AttnNP DSVNP NP CNP (Fitting/UQ Performance)
- Extrapolation in curves of a SP: Tough for all in fitting; NP/AttnNP *!* over-confident; DSVNP *!* better UQ



Multi-output Regression: Simulation/Real-world Dataset

Investigations on (1) system identification on cart-pole transitions [9]; (2) regression on real-world dataset :

- System identification :

- High-dim regression :

Multi-output Regression: Simulation/Real-world Dataset

Investigations on (1) system identification on cart-pole transitions [9]; (2) regression on real-world dataset :

- System identification :

- High-dim regression :

Multi-output Regression: Simulation/Real-world Dataset

Investigations on (1) system identification on cart-pole transitions [9]; (2) regression on real-world dataset :

- System identification :
  MSE & NLL not in accordance; DSVNP & CNP $\not\Rightarrow$ better UQ; DSVNP & AttnNP $\not\Rightarrow$ lower fitting error.
- High-dim regression :

Multi-output Regression: Simulation/Real-world Dataset

Investigations on (1) system identi cation on cart-pole transitions [9]; (2) regression on real-world dataset :

System identi cation :
MSE & NLL not in accordance; DSVNP & CNP ! better UQ; DSVNP & AttnNP ! lower tting error.

High-dim regression :
Hierarchical latent variables advance performance signi cantly.

Classification with Uncertainty Quantification

Observations in image classification and out of distribution detection (based on cumulative distribution of entropies) :

Classification with Uncertainty Quantification

Observations in image classification and out of distribution detection (based on cumulative distribution of entropies) :

- MNIST: no significant difference in classification performance/o.o.d detection (all above 99%) ; DSVNP *!* better o.o.d. detection on FMNIST/KMNIST ; MC-D more robust to Gaussian/Uniform noise.

Classification with Uncertainty Quantification

Observations in image classification and out of distribution detection (based on cumulative distribution of entropies) :

- MNIST: no significant difference in classification performance/o.o.d detection (all above 99%) ; DSVNP *!* better o.o.d. detection on FMNIST/KMNIST ; MC-D more robust to Gaussian/Uniform noise.
- CIFAR10: DSVNP(86.3%) MC/CNP AttnNP/NP NN (Classification Performance) ; DSVNP *!* best entropy distributions in domain dataset and most robust to Rademacher noise.

Future Works

Some Challenging and Promising Directions

- More effective inference methods for our proposed hierarchical SPs

Some Challenging and Promising Directions

- More effective inference methods for our proposed hierarchical SPs
- More expressive context latent variable using higher order statistics

Some Challenging and Promising Directions

- More effective inference methods for our proposed hierarchical SPs
- More expressive context latent variable using higher order statistics
- More explorations to Uncertainty-aware Decision-making Problems

Thanks for Your Listening

M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh, "Neural processes," *arXiv preprint arXiv:1807.01622*, 2018.

M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

F. P. Casale, A. Dalca, L. Saglietti, J. Listgarten, and N. Fusi, "Gaussian process prior variational autoencoders," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 369–10 380.

D. Heath and W. Sudderth, "De finetti's theorem on exchangeable variables," *The American Statistician*, vol. 30, no. 4, pp. 188–189, 1976.

S. Park and S. Choi, "Hierarchical gaussian process regression," in *Proceedings of 2nd Asian Conference on Machine Learning*, 2010, pp. 95–110.

A. Damianou and N. Lawrence, "Deep gaussian processes," in *Arti cial Intelligence and Statistics*, 2013, pp. 207–215.

Z. Dai, A. Damianou, J. González, and N. Lawrence, "Variational auto-encoded deep gaussian processes," *arXiv preprint arXiv:1511.06455*, 2015.

D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

Y. Gal, R. McAllister, and C. E. Rasmussen, "Improving pilco with bayesian neural network dynamics models," in *Data-E cient Machine Learning workshop, ICML*, vol. 4, 2016, p. 34.