

# Dynamical Variational Autoencoders

## Links to stochastic differential equations

Benjamin Deporte :

[benjamin.deporte@ens-paris-saclay.fr](mailto:benjamin.deporte@ens-paris-saclay.fr)

[benjamin.deporte@polytechnique.org](mailto:benjamin.deporte@polytechnique.org)

September 9th, 2025 - Mémoire MVA

# Summary

1 Introduction

2 Dynamical Variational AutoEncoders - Theory

3 Dynamical Variational AutoEncoders - Experiments

4 Link to stochastic calculus

5 Latent Ordinary and Stochastic Differential Equations models : a trailer

6 Take-aways

# Introduction

- **Variational Auto Encoders (VAEs)** have limitations when dealing with data sequences, due to i.i.d assumption.
- **Dynamical Variational Auto Encoders ([11])** are adapted to data sequences
  - Temporal dependency built into latent prior
  - Discrete or continuous time prior
  - Observation model as in regular Variational Auto Encoders (VAEs): Gaussian, Bernoulli, Student-t...
- **Link to Stochastic Calculus**
  - Solutions to linear Stochastic Differential Equations (SDEs) are Gaussian Processs (GPs) : run prior GP regression at linear cost.
  - Solutions to general SDEs are Markov processes : link to more general Latent Stochastic Differential Equation models (Latent SDEs)
- **What we will cover today**
  - General theory of Dynamical Variational Auto Encoders (DVAEs)
  - Review of some discrete and continuous time DVAEs with XPs : Deep Kalman Filter (DKF), Variational Recurrent Neural Network (VRNN), Gaussian Process Variational Auto Encoder (GP-VAE)
  - Stochastic calculus survival kit
  - Relationships between DVAEs and stochastic calculus
  - Early perspective on Latent Ordinary Differential Equation models (Latent ODEs) and Latent Stochastic Differential Equation models (Latent SDEs)

# Introduction

- **VAEs** have limitations when dealing with data sequences, due to i.i.d assumption.
- **Dynamical Variational Auto Encoders ([11])** are adapted to data sequences
  - Temporal dependency built into latent prior
  - Discrete or continuous time prior
  - Observation model as in regular Variational Auto Encoders (VAEs): Gaussian, Bernoulli, Student-t...
- **Link to Stochastic Calculus**
  - Solutions to linear Stochastic Differential Equations (SDEs) are Gaussian Processs (GPs) : run prior GP regression at linear cost.
  - Solutions to general SDEs are Markov processes : link to more general Latent SDEs
- **What we will cover today**
  - General theory of Dynamical Variational Auto Encoders (DVAEs)
  - Review of some discrete and continuous time DVAEs with XPs : Deep Kalman Filter (DKF), Variational Recurrent Neural Network (VRNN), Gaussian Process Variational Auto Encoder (GP-VAE)
  - Stochastic calculus survival kit
  - Relationships between DVAEs and stochastic calculus
  - Early perspective on Latent Ordinary Differential Equation models (Latent ODEs) and Latent Stochastic Differential Equation models (Latent SDEs)

# Introduction

- **VAEs** have limitations when dealing with data sequences, due to i.i.d assumption.
- **Dynamical Variational Auto Encoders ([11])** are adapted to data sequences
  - Temporal dependency built into latent prior
  - Discrete or continuous time prior
  - Observation model as in regular Variational Auto Encoders (VAEs): Gaussian, Bernoulli, Student-t...
- **Link to Stochastic Calculus**
  - Solutions to linear Stochastic Differential Equations (SDEs) are Gaussian Processs (GPs) : run prior GP regression at linear cost.
  - Solutions to general SDEs are Markov processes : link to more general Latent SDEs
- **What we will cover today**
  - General theory of Dynamical Variational Auto Encoders (DVAEs)
  - Review of some discrete and continuous time DVAEs with XPs : Deep Kalman Filter (DKF), Variational Recurrent Neural Network (VRNN), Gaussian Process Variational Auto Encoder (GP-VAE)
  - Stochastic calculus survival kit
  - Relationships between DVAEs and stochastic calculus
  - Early perspective on Latent Ordinary Differential Equation models (Latent ODEs) and Latent Stochastic Differential Equation models (Latent SDEs)

# Introduction

- **VAEs** have limitations when dealing with data sequences, due to i.i.d assumption.
- **Dynamical Variational Auto Encoders ([11])** are adapted to data sequences
  - Temporal dependency built into latent prior
  - Discrete or continuous time prior
  - Observation model as in regular Variational Auto Encoders (VAEs): Gaussian, Bernoulli, Student-t...
- **Link to Stochastic Calculus**
  - Solutions to linear Stochastic Differential Equations (SDEs) are Gaussian Processs (GPs) : run prior GP regression at linear cost.
  - Solutions to general SDEs are Markov processes : link to more general Latent SDEs
- **What we will cover today**
  - General theory of Dynamical Variational Auto Encoders (DVAEs)
  - Review of some discrete and continuous time DVAEs with XPs : Deep Kalman Filter (DKF), Variational Recurrent Neural Network (VRNN), Gaussian Process Variational Auto Encoder (GP-VAE)
  - Stochastic calculus survival kit
  - Relationships between DVAEs and stochastic calculus
  - Early perspective on Latent Ordinary Differential Equation models (Latent ODEs) and Latent Stochastic Differential Equation models (Latent SDEs)

# Dynamical Variational Auto Encoders : generalities

- Time : regular  $t_1, t_2, \dots, t_T$  or irregular  $t_{i_1}, t_{i_2}, \dots, t_{i_T}$  sampling times
- Observations : a sequence of  $T$  points  $x_{1:T} = \{(x_t)_{t=1,\dots,T}\}$  (or  $x_{t_1:t_T} = \{(x_{t_i})_{i=1,\dots,T}\} \in \mathbb{R}^F$ ).
- Latent variables : an associated sequence of  $T$  latent variables  $z_{1:T} = \{(z_t)_{t=1,\dots,T}\} \in \mathbb{R}^L$  (resp.  $z_{t_1}, \dots, z_{t_T}$ )
- Optionally, a sequence of -usually deterministic-  $T$  inputs  $u_{1:T} = \{(u_t)_{t=1,\dots,T}\} \in \mathbb{R}^U$  (resp.  $u_{t_i}, i = 1, \dots, T$ )
- Encoding temporal dependency in prior over  $z_i$ 's
- Arbitrary likelihood (observation model)
- Approximate posterior usually same form as true posterior
- Use D-Separation in Graphical Probabilistic Model (GPM) to simplify expressions
- Untractable true posterior  $\implies$  training by maximizing a Variational Lower Bound (VLB)

# Dynamical Variational Auto Encoders : generalities

- Time : regular  $t_1, t_2, \dots, t_T$  or irregular  $t_{i_1}, t_{i_2}, \dots, t_{i_T}$  sampling times
- Observations : a sequence of  $T$  points  $x_{1:T} = \{(x_t)_{t=1,\dots,T}\}$  (or  $x_{t_1:t_T} = \{(x_{t_i})_{i=1,\dots,T}\} \in \mathbb{R}^F$ ).
- Latent variables : an associated sequence of  $T$  latent variables  $z_{1:T} = \{(z_t)_{t=1,\dots,T}\} \in \mathbb{R}^L$  (resp.  $z_{t_1}, \dots, z_{t_T}$ )
- Optionally, a sequence of -usually deterministic-  $T$  inputs  $u_{1:T} = \{(u_t)_{t=1,\dots,T}\} \in \mathbb{R}^U$  (resp.  $u_{t_i}, i = 1, \dots, T$ )
- **Encoding temporal dependency in prior over  $z_i$ 's**
- Arbitrary likelihood (observation model)
- Approximate posterior usually same form as true posterior
- Use **D-Separation** in Graphical Probabilistic Model (GPM) to simplify expressions
- Untractable true posterior  $\implies$  training by maximizing a Variational Lower Bound (VLB)

# General formulation of DVAE

## Generative model

$$p(x_{1:T}, z_{1:T} | u_{1:T}) = \prod_{t=1}^T p(x_t, z_t | x_{1:t-1}, z_{1:t-1}, u_{1:T}) \quad (1)$$

$$= \prod_{t=1}^T p(x_t | x_{1:t-1}, z_{1:t}, u_{1:T}) p(z_t | x_{1:t-1}, z_{1:t-1}, u_{1:T}) \quad (2)$$

$$= \prod_{t=1}^T p(x_t | x_{1:t-1}, z_{1:t}, u_{1:t}) p(z_t | x_{1:t-1}, z_{1:t-1}, u_{1:t}) \quad (3)$$

**Only assumption : causal dependency  $x_t, z_t$  on inputs  $u_{1:t} \implies |u_{1:T} = |u_{1:t}$ .**

(NB : assume no input in the rest of the presentation)

# Posteriors

- True posterior  $p(z_{1:T}|x_{1:T})$  usually untractable::

$$p(z_{1:T}|x_{1:T}) = \prod_{t=1}^T p(z_t|z_{1:t-1}, x_{1:T})$$

- Inference model : approximate posterior by an parametric encoder  $q_\phi(z_{1:T}|x_{1:T})$  ( $\phi$  the set of parameters):

$$q_\phi(z_{1:T}|x_{1:T}) = \prod_{t=1}^T q_\phi(z_t|z_{1:t-1}, x_{1:T})$$

- D-separation on GPM to simplify  $p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t})$  and  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$
- Good practice : use the true posterior expression  $p(z_{1:T}|x_{1:T})$  for  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$

# Posteriors

- True posterior  $p(z_{1:T}|x_{1:T})$  usually untractable::

$$p(z_{1:T}|x_{1:T}) = \prod_{t=1}^T p(z_t|z_{1:t-1}, x_{1:T})$$

- Inference model : approximate posterior by an parametric encoder  $q_\phi(z_{1:T}|x_{1:T})$  ( $\phi$  the set of parameters):

$$q_\phi(z_{1:T}|x_{1:T}) = \prod_{t=1}^T q_\phi(z_t|z_{1:t-1}, x_{1:T})$$

- D-separation on GPM to simplify  $p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t})$  and  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$
- Good practice : use the true posterior expression  $p(z_{1:T}|x_{1:T})$  for  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$

## Posteriors

- True posterior  $p(z_{1:T}|x_{1:T})$  usually untractable::

$$p(z_{1:T}|x_{1:T}) = \prod_{t=1}^T p(z_t|z_{1:t-1}, x_{1:T})$$

- Inference model : approximate posterior by an parametric encoder  $q_\phi(z_{1:T}|x_{1:T})$  ( $\phi$  the set of parameters):

$$q_\phi(z_{1:T}|x_{1:T}) = \prod_{t=1}^T q_\phi(z_t|z_{1:t-1}, x_{1:T})$$

- D-separation on GPM to simplify  $p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t})$  and  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$
- Good practice : use the true posterior expression  $p(z_{1:T}|x_{1:T})$  for  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$

# Posteriors

- True posterior  $p(z_{1:T}|x_{1:T})$  usually untractable::

$$p(z_{1:T}|x_{1:T}) = \prod_{t=1}^T p(z_t|z_{1:t-1}, x_{1:T})$$

- Inference model : approximate posterior by an parametric encoder  $q_\phi(z_{1:T}|x_{1:T})$  ( $\phi$  the set of parameters):

$$q_\phi(z_{1:T}|x_{1:T}) = \prod_{t=1}^T q_\phi(z_t|z_{1:t-1}, x_{1:T})$$

- D-separation on GPM to simplify  $p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t})$  and  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$
- Good practice : use the true posterior expression  $p(z_{1:T}|x_{1:T})$  for  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$

# Likelihood

- Observation model and encoder:

$$p_{\theta}(x_{1:T}, z_{1:T}) = \prod_{t=1}^T p_{\theta_x}(x_t | x_{1:t-1}, z_{1:t}) p_{\theta_z}(z_t | z_{1:t-1}, x_{1:t-1}) \quad (4)$$

$$q_{\phi}(z_{1:T} | x_{1:T}) = \prod_{t=1}^T q_{\phi}(z_t | z_{1:t-1}, x_{1:T}) \quad (5)$$

- Log likelihood

$$\log p(x_{1:T}) = \log \frac{p(x_{1:T}, z_{1:T})}{p(z_{1:T} | x_{1:T})} \quad (6)$$

$$= \mathbb{E}_{q_{\phi}(z_{1:T} | x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_{\phi}(z_{1:T} | x_{1:T})} \frac{q_{\phi}(z_{1:T} | x_{1:T})}{p(z_{1:T} | x_{1:T})} \quad (7)$$

$$= \mathbb{E}_{q_{\phi}(z_{1:T} | x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_{\phi}(z_{1:T} | x_{1:T})} + \text{KL}(q_{\phi}(z_{1:T} | x_{1:T}) || p(z_{1:T} | x_{1:T})) \quad (8)$$

$$\geq \mathbb{E}_{q_{\phi}(z_{1:T} | x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_{\phi}(z_{1:T} | x_{1:T})} = \mathcal{L}(\theta, \phi, X) \quad (9)$$

# Likelihood

- Observation model and encoder:

$$p_{\theta}(x_{1:T}, z_{1:T}) = \prod_{t=1}^T p_{\theta_x}(x_t | x_{1:t-1}, z_{1:t}) p_{\theta_z}(z_t | z_{1:t-1}, x_{1:t-1}) \quad (4)$$

$$q_{\phi}(z_{1:T} | x_{1:T}) = \prod_{t=1}^T q_{\phi}(z_t | z_{1:t-1}, x_{1:T}) \quad (5)$$

- Log likelihood

$$\log p(x_{1:T}) = \log \frac{p(x_{1:T}, z_{1:T})}{p(z_{1:T} | x_{1:T})} \quad (6)$$

$$= \mathbb{E}_{q_{\phi}(z_{1:T} | x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_{\phi}(z_{1:T} | x_{1:T})} \frac{q_{\phi}(z_{1:T} | x_{1:T})}{p(z_{1:T} | x_{1:T})} \quad (7)$$

$$= \mathbb{E}_{q_{\phi}(z_{1:T} | x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_{\phi}(z_{1:T} | x_{1:T})} + \mathbb{KL}(q_{\phi}(z_{1:T} | x_{1:T}) || p(z_{1:T} | x_{1:T})) \quad (8)$$

$$\geq \mathbb{E}_{q_{\phi}(z_{1:T} | x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_{\phi}(z_{1:T} | x_{1:T})} = \mathcal{L}(\theta, \phi, X) \quad (9)$$

## Variational Lower Bound

- Lower bound:

$$\mathcal{L}(\theta, \phi, X) = \mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})} \log \left( \frac{\prod_{t=1}^T p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})}{\prod_{t=1}^T q_\phi(z_t|z_{1:t-1}, x_{1:T})} \right) \quad (10)$$

$$= \mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})} \left( \sum_{t=1}^T \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) - \sum_{t=1}^T \log \frac{q_\phi(z_t|z_{1:t-1}, x_{1:T})}{p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})} \right) \quad (11)$$

$$= \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})} \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) - \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})} \mathbb{KL}(q_\phi(z_t|z_{1:t-1}, x_{1:T}) || p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})) \quad (12)$$

- First term : often called **reconstruction error** (formally a log likelihood)
- Second term : **regularization term**, average divergence between the approximate posterior distribution at time  $t$ , and its real distribution.
- Sampling over  $q_\phi$  requires the "re-parametrization trick" (see [12]), for  $\mathcal{L}(\theta, \phi, X)$  to be differentiable w.r.t.  $\theta, \phi$ .

# Summary DVAE

General Dynamical VAEs : generative and inference models; variational lower bound

$$p(x_{1:T}, z_{1:T}) = \prod_{t=1}^T p_{\theta_x}(x_t | x_{1:t-1}, z_{1:t}) p_{\theta_z}(z_t | z_{1:t-1}, x_{1:t-1}) \quad (13)$$

$$q_\phi(z_{1:T} | x_{1:T}) = \prod_{t=1}^T q_\phi(z_t | z_{1:t-1}, x_{1:T}) \quad (14)$$

$$\begin{aligned} \mathcal{L}(\theta, \phi, X) &= \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{1:t} | x_{1:T})} \log p_{\theta_x}(x_t | x_{1:t-1}, z_{1:t}) \\ &\quad - \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{1:t-1} | x_{1:T})} \mathbb{KL}(q_\phi(z_t | z_{1:t-1}, x_{1:T}) || p_{\theta_z}(z_t | z_{1:t-1}, x_{1:t-1})) \end{aligned} \quad (15)$$

## First model : Deep Kalman Filter

Deep Kalman Filter Directed Acyclic Graph (DAG): State Space Model (SSM) with Gaussian likelihood parameterized by neural nets.

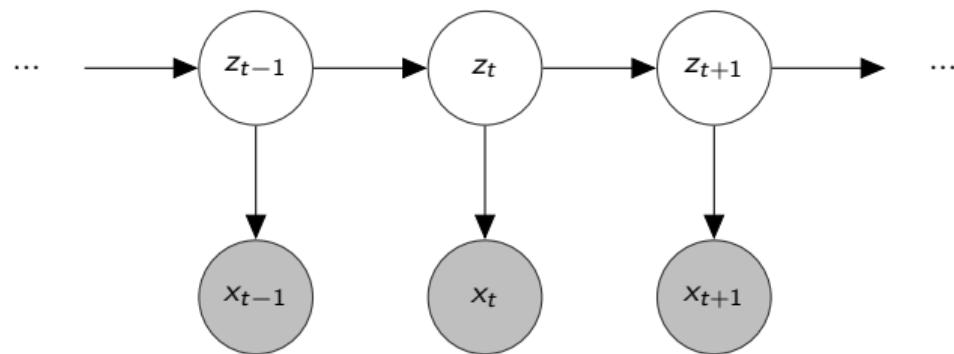


Figure: Probabilistic model of a Deep Kalman Filter

# Deep Kalman Filter - generative model

- Use **D-separation** ([1], [14], [18]) to simplify the general DVAE expressions 13 and 14.
- Conditioning on  $z_t$  and  $z_{t-1}$  drives:

$$p_{\theta_x}(x_t | x_{1:t-1}, z_{1:t}) = p_{\theta_x}(x_t | z_t) \quad (16)$$

$$p_{\theta_z}(z_t | z_{1:t-1}, x_{1:t}) = p_{\theta_z}(z_t | z_{t-1}) \quad (17)$$

$$q_{\phi}(z_t | z_{1:t-1}, x_{1:T}) = q_{\phi}(z_t | z_{t-1}, x_{t:T}) \quad (18)$$

## Deep Kalman Filter - generative model - 2

- Gaussian distributions for  $p_{\theta_x}$ ,  $p_{\theta_z}$  and  $q_\phi$  (mean and diagonal covariance learnt by neural networks).

$$p_{\theta_x}(x_t|z_t) = \mathcal{N}(x_t|\mu_{\theta_x}(z_t), \text{diag } \sigma_{\theta_x}^2(z_t)) \quad (19)$$

$$p_{\theta_z}(z_t|z_{t-1}) = \mathcal{N}(z_t|\mu_{\theta_z}(z_{t-1}), \text{diag } \sigma_{\theta_z}^2(z_{t-1})) \quad (20)$$

$$q_\phi(z_t|z_{t-1}, x_{t:T}) = \mathcal{N}(z_t|\mu_\phi(z_{t-1}, x_{t:T}), \text{diag } \sigma_{\theta_z}^2(z_{t-1}, x_{t:T})) \quad (21)$$

- Some other formulations of the approximate posterior (encoder) are possible. For example:

$$q_\phi(z_t|z_{t-1}, x_t)$$

$$q_\phi(z_t|z_{1:t}, x_{1:t})$$

$$q_\phi(z_t|z_{1:T}, x_{1:T})$$

- We have chosen 18 for the implementation (same form as the true posterior).

# Deep Kalman Filter - ELBO

Using D-Separation, the Evidence Lower Bound (ELBO) 15 simplifies into:

$$\mathcal{L}(\theta, \phi, X) = \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})} \log p_{\theta_x}(x_t|z_t) - \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})} \mathbb{KL}(q_\phi(z_t|z_{t-1}, x_{t:T}) || p_{\theta_z}(z_t|z_{t-1})) \quad (22)$$

$$= \sum_{t=1}^T \mathbb{E}_{q_\phi(z_t|x_{1:T})} \log p_{\theta_x}(x_t|z_t) - \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{t-1}|x_{1:T})} \mathbb{KL}(q_\phi(z_t|z_{t-1}, x_{t:T}) || p_{\theta_z}(z_t|z_{t-1})) \quad (23)$$

# Deep Kalman Filter - summary

## Deep Kalman Filter

- **generative model**

$$p_{\theta_x}(x_t|z_t) = \mathcal{N}(x_t|\mu_{\theta_x}(z_t), \text{diag } \sigma_{\theta_x}^2(z_t)) \quad (24)$$

$$p_{\theta_z}(z_t|z_{t-1}) = \mathcal{N}(z_t|\mu_{\theta_z}(z_{t-1}), \text{diag } \sigma_{\theta_z}^2(z_{t-1})) \quad (25)$$

- **inference model**

$$q_{\phi}(z_t|z_{t-1}, x_{t:T}) = \mathcal{N}(z_t|\mu_{\phi}(z_{t-1}, x_{t:T}), \text{diag } \sigma_{\theta_z}^2(z_{t-1}, x_{t:T})) \quad (26)$$

- **VLB for training**

$$\mathcal{L}(\theta, \phi, X) = \sum_{t=1}^T \mathbb{E}_{q_{\phi}(z_t|x_{1:T})} \log p_{\theta_x}(x_t|z_t) - \sum_{t=1}^T \mathbb{E}_{q_{\phi}(z_{t-1}|x_{1:T})} \mathbb{KL}(q_{\phi}(z_t|z_{t-1}, x_{t:T}) || p_{\theta_z}(z_t|z_{t-1})) \quad (27)$$

# DKF - Torch

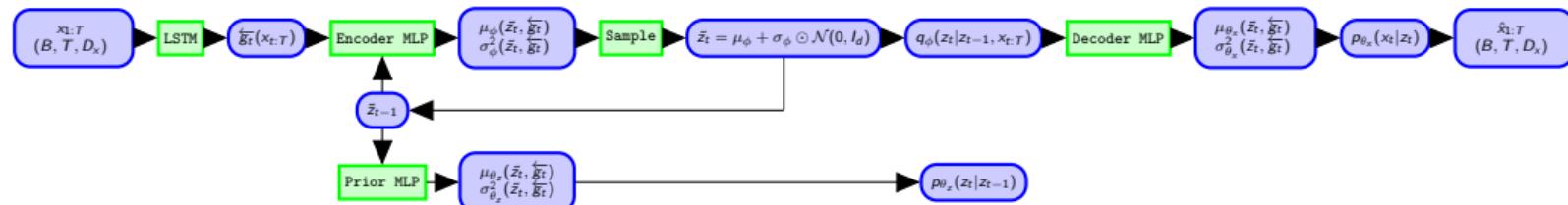
- The  $\text{KL}(q_\phi || p_{\theta_z})$ 's have a close form, as the two distributions are Gaussians.
- Encode  $x_{1:t}$  with forward Long Short Term Memory (LSTM)
- Encode  $x_{t:T}$  with backward LSTM
- Use LSTMs states as inputs to the Multi Layer Perceptron (MLP) parametrizing the distributions.
- For example:

$$\overleftarrow{g_t} = \text{Backward LSTM}(\overleftarrow{g_{t+1}}, x_t) \text{ (encodes } x_{t:T})$$

$$q_\phi(z_t | z_{t-1}, x_{t:T}) = \mathcal{N}(z_t | \mu_\phi(z_{t-1}, \overleftarrow{g_t}), \text{diag } \sigma_\phi^2(z_{t-1}, \overleftarrow{g_t}))$$

## DKF - Torch - Schematic blocks

The PyTorch implementation is described below:



$$\mathcal{L}(\theta, \phi, X) = \sum_{t=1}^T \mathbb{E}_{q_\phi(z_t | x_{1:T})} \log p_{\theta_x}(x_t | z_t) - \sum_{t=1}^T \mathbb{E}_{q_\phi(z_{t-1} | x_{1:T})} \mathbb{KL}(q_\phi(z_t | z_{t-1}, x_{1:T}) || p_{\theta_z}(z_t | z_{t-1}))$$

## Second model : Variational RNN

- The VRNN is the most expressive DVAE : the general expressions 13, 14 and VLB 15 can not be simplified.
- The GPM of the VRNN assumes full forward connections between latent variables, and between observed variables.

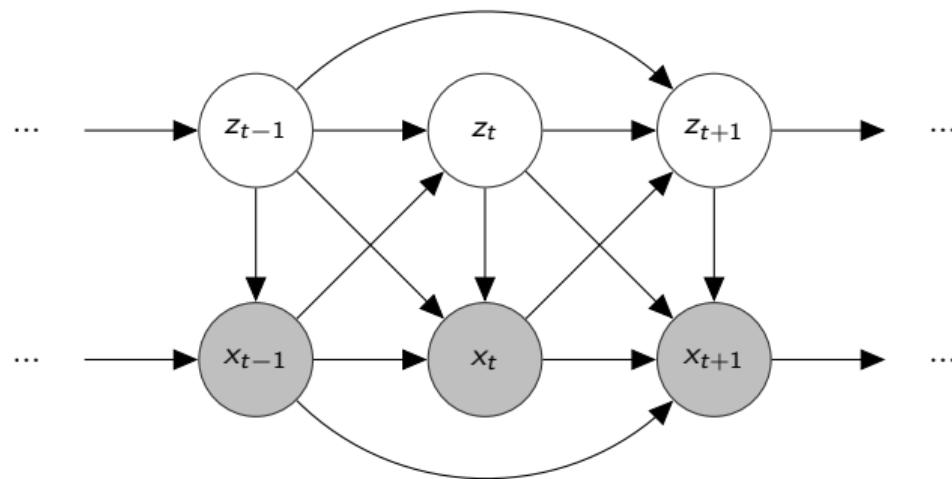


Figure: Probabilistic model of a Variational RNN

# VRNN - Summary

## Variational RNN

- generative model

$$p_{\theta_x}(x_t | x_{1:t-1}, z_{1:t}) = \mathcal{N}(x_t | \mu_{\theta_x}(x_{1:t-1}, z_{1:t}), \text{diag } \sigma_{\theta_x}^2(x_{1:t-1}, z_{1:t})) \quad (28)$$

$$p_{\theta_z}(z_t | z_{1:t-1}, x_{1:t-1}) = \mathcal{N}(z_t | \mu_{\theta_z}(z_{1:t-1}, x_{1:t-1}), \text{diag } \sigma_{\theta_z}^2(z_{1:t-1}, x_{1:t-1})) \quad (29)$$

- inference model

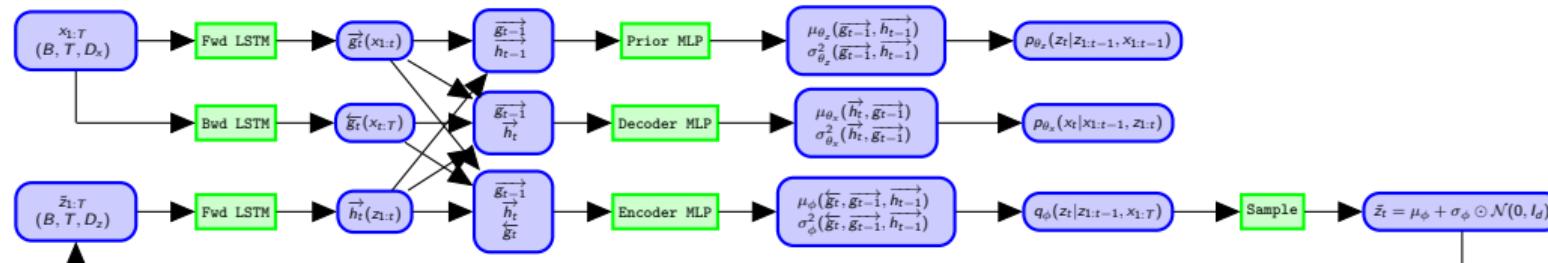
$$q_{\phi}(z_t | z_{1:t-1}, x_{1:T}) = \mathcal{N}(z_t | \mu_{\phi}(z_{1:t-1}, x_{1:T}), \text{diag } \sigma_{\phi}^2(z_{1:t-1}, x_{1:T})) \quad (30)$$

- VLB for training

$$\begin{aligned} \mathcal{L}(\theta, \phi, X) &= \sum_{t=1}^T \mathbb{E}_{q_{\phi}(z_{1:t} | x_{1:T})} \log p_{\theta_x}(x_t | x_{1:t-1}, z_{1:t}) \\ &\quad - \sum_{t=1}^T \mathbb{E}_{q_{\phi}(z_{1:t-1} | x_{1:T})} \mathbb{KL}(q_{\phi}(z_t | z_{1:t-1}, x_{1:T}) || p_{\theta_z}(z_t | z_{1:t-1}, x_{1:t-1})) \end{aligned} \quad (31)$$

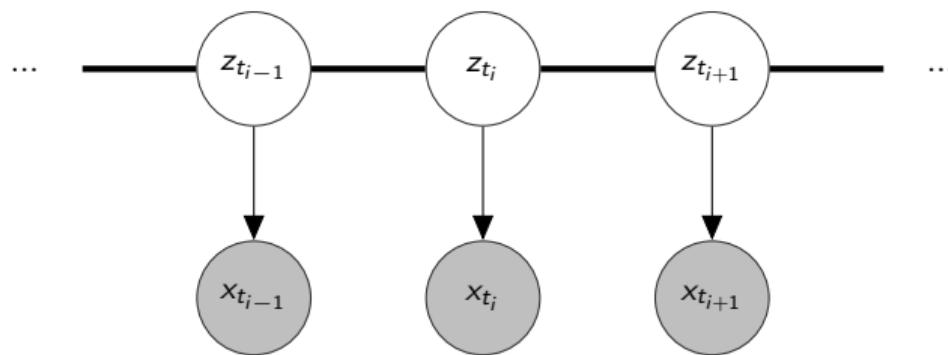
# VRNN - Torch

We have chosen a different implementation from [11] and used three different LSTM networks to encode  $z_{1:t}$ ,  $x_{1:t-1}$  and  $x_{t:T}$  respectively.



$$\mathcal{L}(\theta, \phi, X) = \sum_{t=1}^T \mathbb{E}_{q_{\phi}(z_{1:t}|x_{1:T})} \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) - \sum_{t=1}^T \mathbb{E}_{q_{\phi}(z_{1:t-1}|x_{1:T})} \text{KL}(q_{\phi}(z_t|z_{1:t-1}, x_{1:T}) || p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1}))$$

## Third model : Gaussian Process - Variational Auto Encoder



**Figure:** Probabilistic model of a GP-VAE

- **thick black lines** indicate a Gaussian Process prior over latent variables.
- The joint distribution writes:

$$p(x_{t_1:t_T}, z_{t_1:t_T}) = p(z_{t_1:t_T}) \prod_{i=1}^T p(x_{t_i} | z_{t_i}) \quad (32)$$

## GP-VAE generative model

- **Gaussian Process prior** : the prior over the latent variables  $z_{t_i} \in \mathbb{R}^L$  is a set of scalar Gaussian Process over each of the dimension  $l \in \{1, \dots, L\}$  of the latent variables. Formally:

$$p_{\theta_z}(z_{t_1:t_T}^l) = \mathcal{GP}(m_{\theta_z,l}(t_1:t_T), k_{\theta_z,l}(t_1:t_T, t_1:t_T)) \quad l = 1, \dots, L \quad (33)$$

- $m_{\theta_z,l}$  are the  $L$  mean functions of the GP priors (usually chosen constant null)
- $k_{\theta_z,l}$  are the kernel functions of the GP priors. Can be chosen differently to account for prior knowledge of the data sequence.
- Each  $\mathcal{GP}^l$  encode a temporal dependency over  $z^{(l)}$
- Correlation accross dimensions of data is encoded within likelihood  $p_{\theta_x}$

- **Approximate posterior -encoder** :  $q_\phi$  is a set of  $L$  Gaussian distributions of dimension  $T$ , each one accounting for a component of  $z_{t_i}$ . Formally :

$$q_\phi(z_{t_1:t_T}^l | x_{t_1:t_T}^l) = \mathcal{N}(m_\phi^l(x_{t_1:t_T}), \Sigma_\phi^l(x_{t_1:t_T})) \quad l = 1, \dots, L \quad (34)$$

$$= \mathcal{N}(m_\phi^l(x_{t_1:t_T}), \Lambda_\phi^l(x_{t_1:t_T})^{-1}) \quad (35)$$

$$= \mathcal{N}(m_\phi^l(x_{t_1:t_T}), L_\phi^l(x_{t_1:t_T}) L_\phi^l(x_{t_1:t_T})^T) \quad (36)$$

Code covariance with covariance matrix  $\Sigma_\phi^l$ , precision matrix  $\Lambda_\phi^l$ , or with a Cholesky decomposition  $L_\phi^l L_\phi^l$ .

## GP-VAE generative model

From [16]

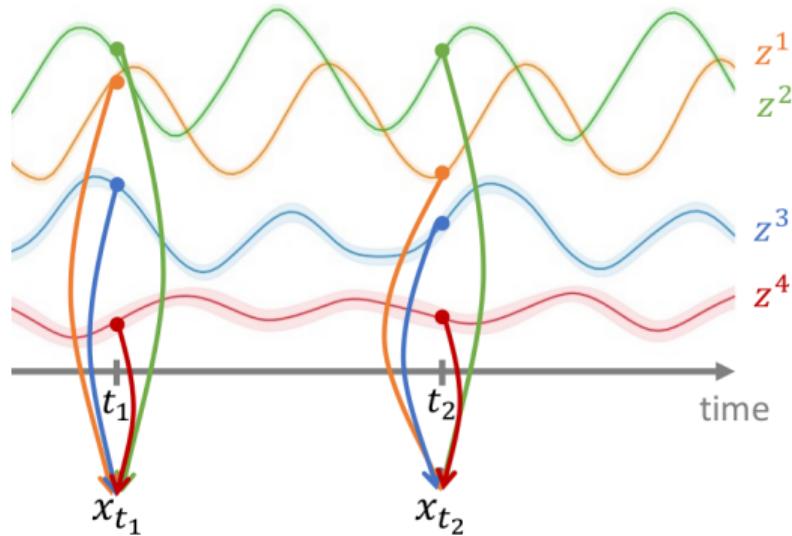


Figure: GP-VAE Schematics

# GP-VAE likelihood and ELBO

$$\mathcal{L}(\theta, \phi, X) = \sum_{i=1}^T \mathbb{E}_{q_\phi(z_{t_i} | x_{t_1:t_T})} \log p_{\theta_x}(x_{t_i} | z_{t_i}) - \text{KL}(q_\phi(z_{t_1:t_T} | x_{t_1:t_T}) || p_{\theta_z}(z_{t_1:t_T})) \quad (37)$$

We note that:

- the KL-divergence is the sum of the  $L$  KL-divergences  $\text{KL}(q_\phi^l || p_{\theta_z}^l)$  (close form solution)
- the negative log likelihood loss term requires sampling from  $q_\phi(z_{t_i} | x_{t_1:t_T})$  using the reparameterization trick as usual.
- the GP priors  $p_{\theta_z}(z_{t_1:t_T})$  depend only on the time stamps  $t_1, \dots, t_T$ . ( $O(N^3)$  cost)
  - If the kernel parameters are fixed -such as in [9]- then the priors can be computed before the training loop.
  - If the kernel parameters are learnt with the weights of the neural nets (such as in [30]), then the computation must occur at each training iteration.

# GP-VAE Summary

## Gaussian Process VAEs

$$p(x_{t_1:t_T}, z_{t_1:t_T}) = p(z_{t_1:t_T}) \prod_{i=1}^T p(x_{t_i} | z_{t_i}) \quad (38)$$

$$p_{\theta_z}(z_{t_1:t_T}^l) = \mathcal{GP}(m_{\theta_z,l}(t_1 : t_T), k_{\theta_z,l}(t_1 : t_T)) \quad l = 1, \dots, L \quad (39)$$

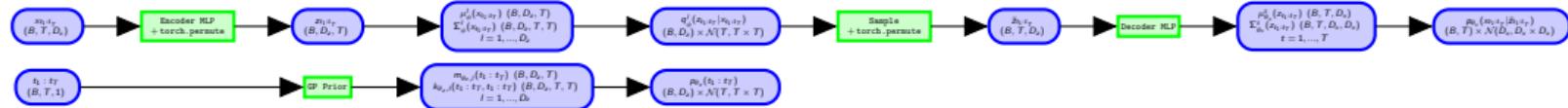
$$q_\phi(z_{t_1:t_T}^l | x_{t_1:t_T}^l) = \mathcal{N}(m_\phi^l(x_{t_1:t_T}), \Sigma_\phi^l(x_{t_1:t_T})) \quad l = 1, \dots, L \quad (40)$$

$$= \mathcal{N}(m_\phi^l(x_{t_1:t_T}), \Lambda_\phi^l(x_{t_1:t_T})^{-1}) \quad (41)$$

$$= \mathcal{N}(m_\phi^l(x_{t_1:t_T}), L_\phi^l(x_{t_1:t_T}) L_\phi^l(x_{t_1:t_T})^T) \quad (42)$$

$$\mathcal{L}(\theta, \phi, X) = \sum_{i=1}^T \mathbb{E}_{q_\phi(z_{t_i} | x_{t_1:t_T})} \log p_{\theta_x}(x_{t_i} | z_{t_i}) - \mathbb{KL}(q_\phi(z_{t_1:t_T} | x_{t_1:t_T}) || p_{\theta_z}(z_{t_1:t_T})) \quad (43)$$

# GP-VAE - Torch



$$\mathcal{L}(\theta, \phi, X) = \sum_{i=1}^T \mathbb{E}_{q_\phi(z_{t_i} | x_{t_1:t_T})} \log p_{\theta_X}(x_{t_i} | z_{t_i}) - \text{KL}(q_\phi(z_{t_1:t_T} | x_{t_1:t_T}) || p_{\theta_Z}(z_{t_1:t_T}))$$

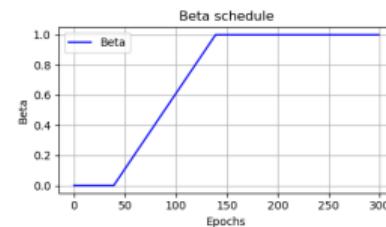
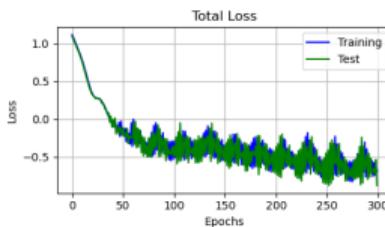
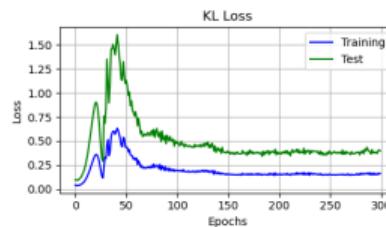
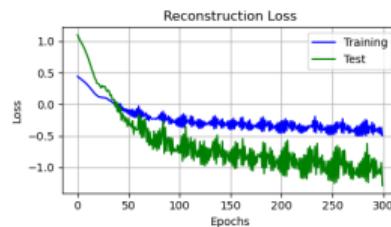
# Code

GitHub repo

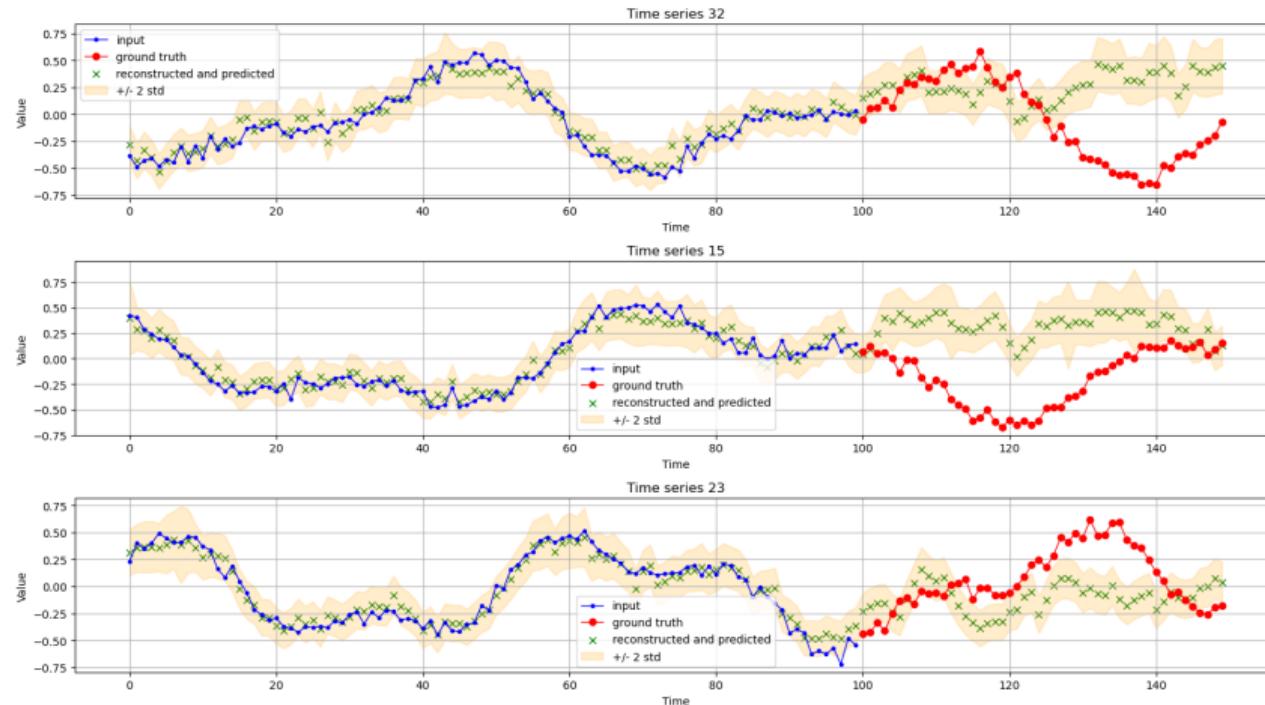
All code at Benjamin GitHub repo

# DKF - Toy training - Loss

- Posterior collapse !
- Need  $\beta$  schedule (weight between reconstruction loss and KL)

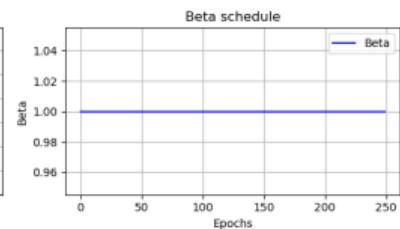
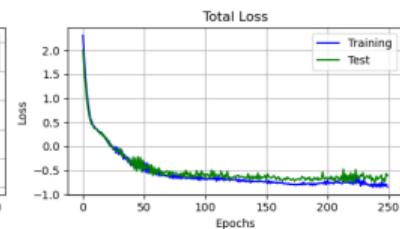
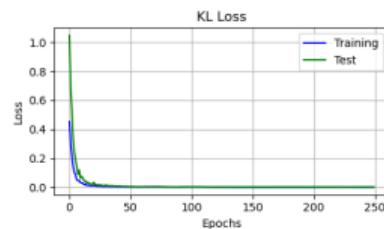
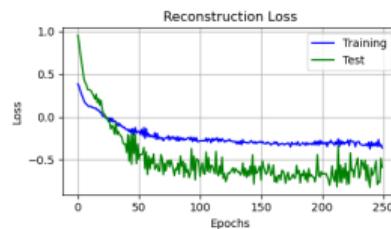


# DKF - Toy training - Generations

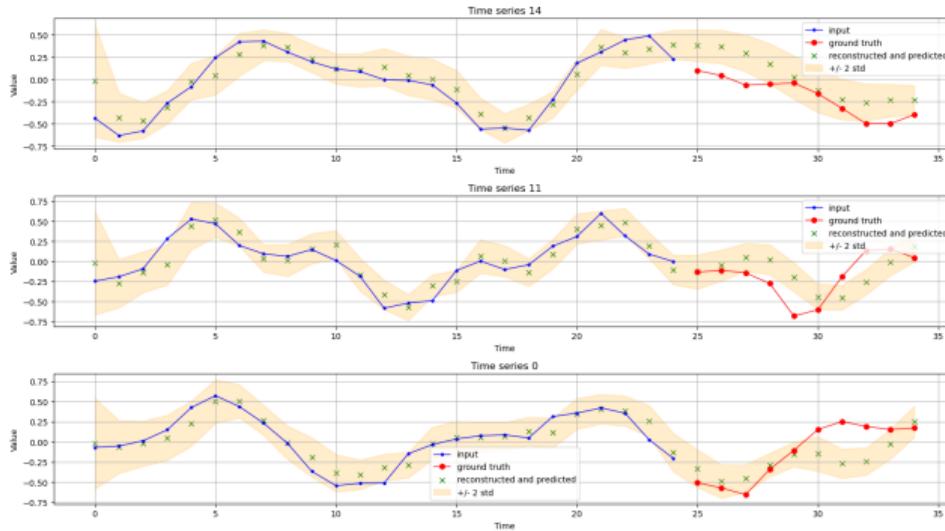


## VRNN - Toy training - Loss

- No posterior collapse
- Shorter toy time series to reduce training time



# VRNN - Toy training - Generation



## Sprites dataset

- Synthetic cartoon characters dataset from [16]
- RGB images  $64 \times 64 \times 3$ .
- Each character has
  - 4 attributes (hair, shoes, top cloth, bottom cloth) with 6 possible values each
  - 3 possible poses (left, front, right)
- Motion across time : 3 possible actions (walk, spell, slash) across 8 consecutive frames.

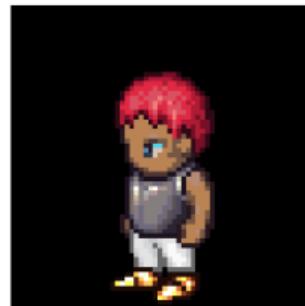
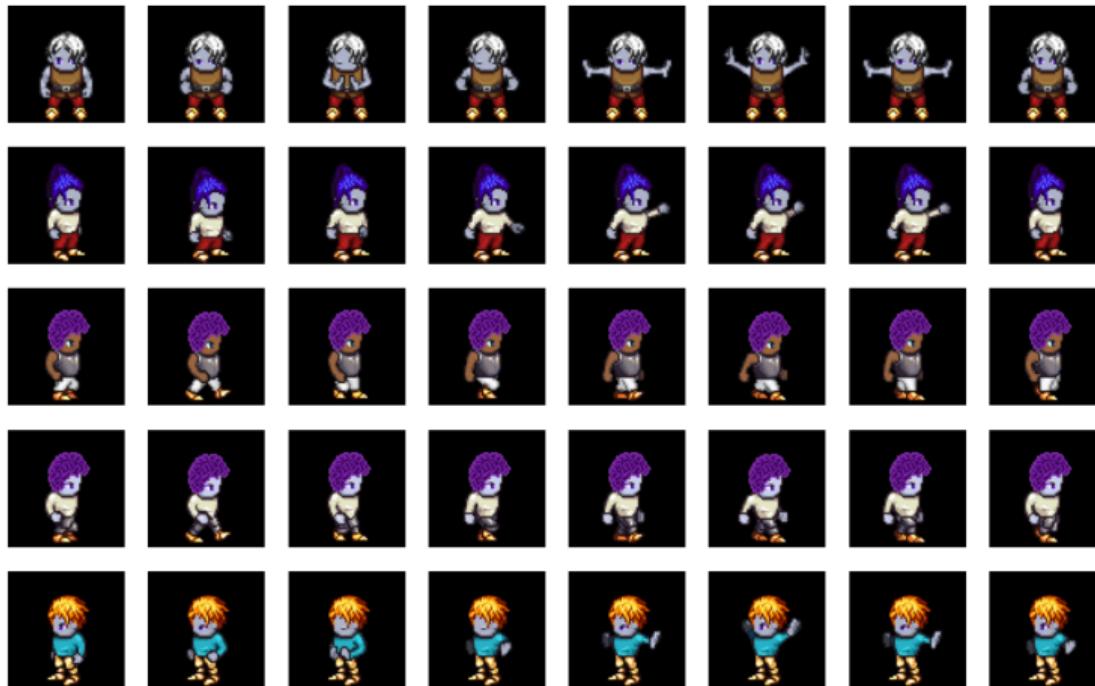
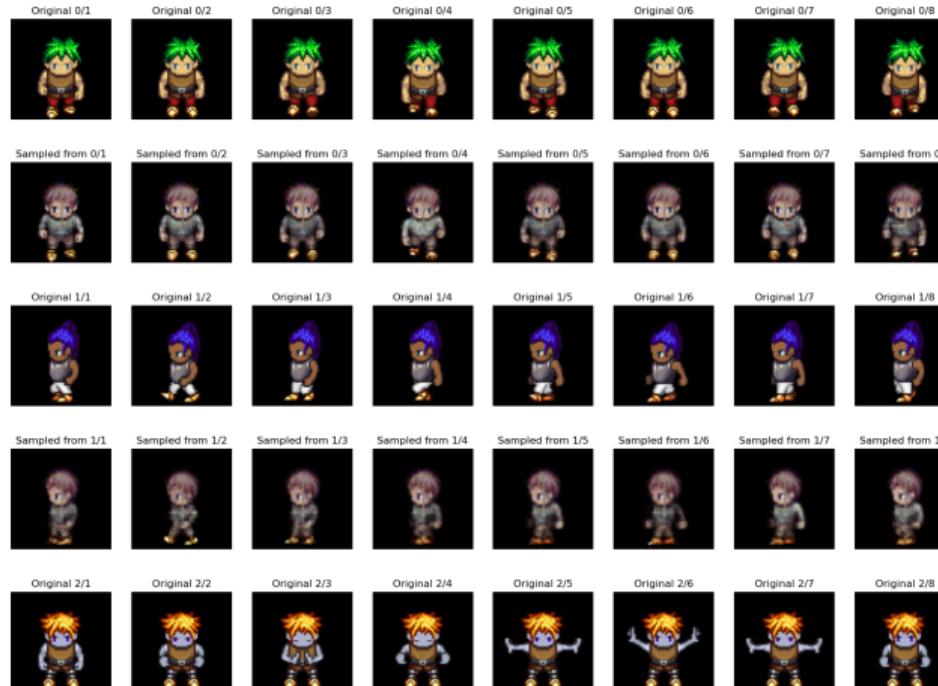


Figure: One sprite

## Sprites series



# VRNN Reconstruction on Sprites



# VRNN Generation on Sprites

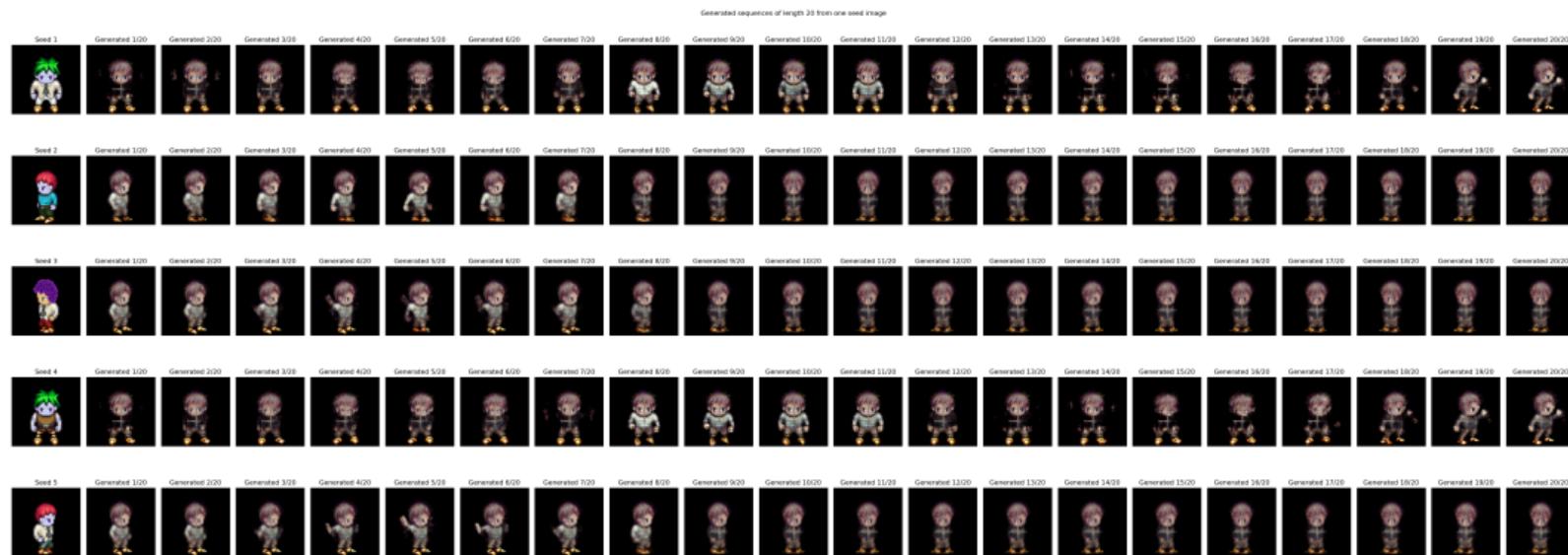
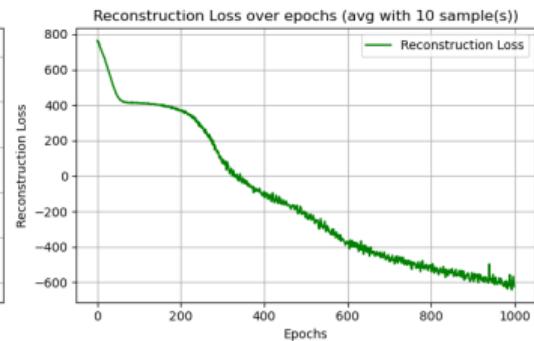
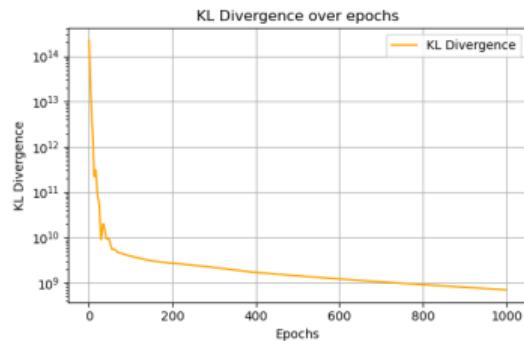
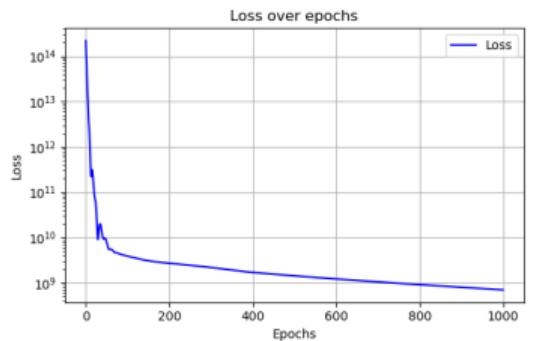


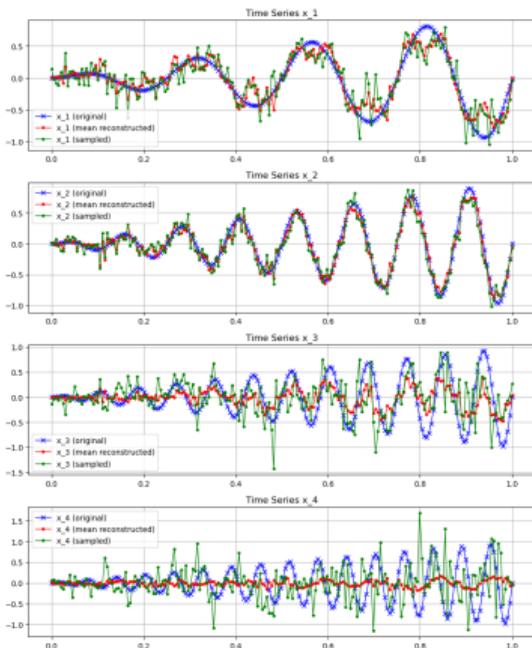
Figure: VRNN Sprites generation

## GPVAE - Toy training - Loss

$D_x = 4$ ,  $D_y = 8$ , RBF kernels with decreasing lengthscales.



## GPVAE - Toy training - Generations



# GPVAE - Sprites Reconstruction - 1

Top: original, below: reconstructions

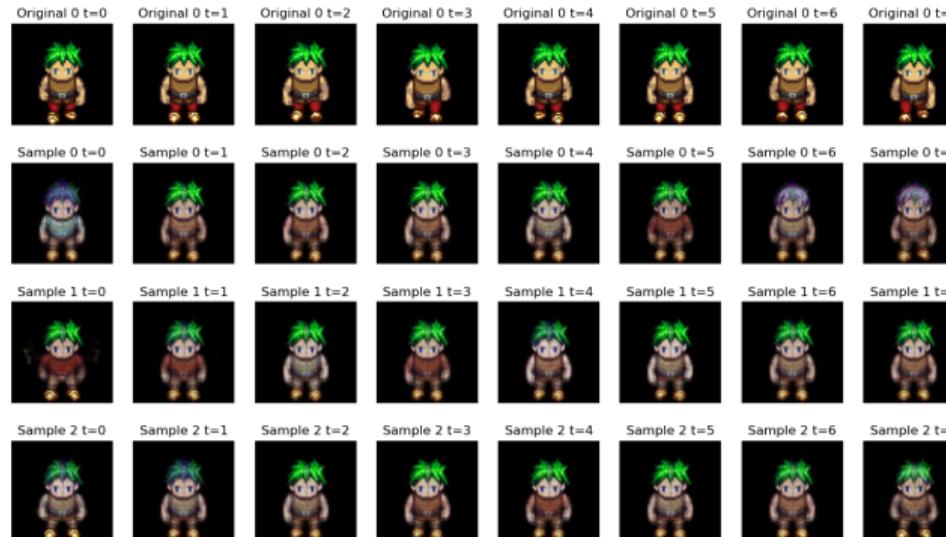


Figure: GPVAE Sprites reconstruction 1

## GPVAE - Sprites Reconstruction - 2

Top: original, below: reconstructions

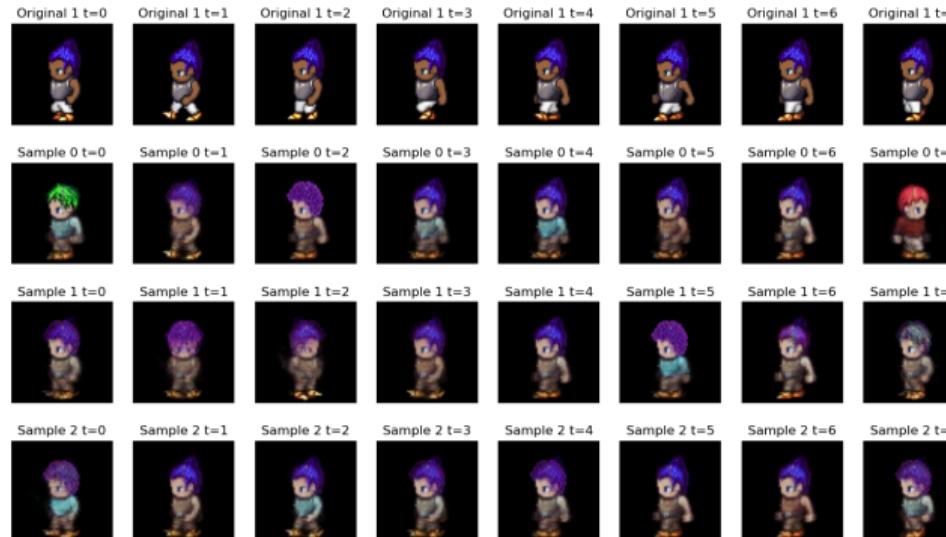


Figure: GPVAE Sprites reconstruction 2

# GPVAE - Sprites Reconstruction - 1

Top: original, below: reconstructions



Figure: GPVAE Sprites reconstruction 3

# GPVAE - Generation

Use Gaussian and Matern kernels : not as good as [16] (Cauchy kernels) !!

Generations from prior



# Stochastic calculus survival kit - Stochastic process

## Definition

A **stochastic process** is defined as:

$$X = (\Omega, \mathcal{F}, (\mathcal{F}_t)_{t \in T}, (X_t)_{t \in T}, \mathbb{P}) \quad (44)$$

where:

- $\Omega$  is a set (universe of possibles).
- $\mathcal{F}$  is a  $\sigma$ -algebra of parts of  $\Omega$
- $\mathbb{P}$  is a probability measure on  $(\Omega, \mathcal{F})$
- $T \subset \mathbb{R}_+$  represents time
- $(\mathcal{F}_t)_{t \in T}$  is a **filtration**, ie an increasing family of sub- $\sigma$ -algebras of  $\mathcal{F}$  indexed by  $t : \forall 0 \leq s \leq t \in T, \mathcal{F}_s \subset \mathcal{F}_t \subset \mathcal{F}$ .
- $(X_t)_{t \in T}$  is a family of RV defined on  $(\Omega, \mathcal{F})$  with values in a measurable space  $(E, \mathcal{E})$  or more simply  $(E, \mathcal{B}(E))$  (set  $E$  endowed with its Borelian  $\sigma$ -algebra).
- $(X_t)_{t \in T}$  is assumed **adapted to the filtration**  $(\mathcal{F}_t)_{t \in T}$ , meaning  $\forall t \in T, X_t$  is  $\mathcal{F}_t$ -measurable

# Stochastic calculus survival kit - Brownian motion

## Definition

A stochastic process  $B = (\Omega, \mathcal{F}, (\mathcal{F}_t)_{t \geq 0}, (B_t)_{t \geq 0}, \mathbb{P})$  with values in  $\mathbb{R}^d$  is called **Brownian motion** iff:

- $B_0 = 0$   $\mathbb{P}$ -a.s.
- $\forall 0 \leq s \leq t$ , the random variable  $B_t - B_s$  is independent from  $\mathcal{F}_s$ .
- $\forall 0 \leq s \leq t$ ,  $B_t - B_s \sim \mathcal{N}(0, Q(t-s))$
- $B$  is continuous <sup>a</sup>

where the matrix  $Q \in \mathbb{S}_d^{++}$  is called the **diffusion matrix**.

---

<sup>a</sup>or more exactly there exists a continuous version of  $B$ , see [17]

A core result is that the quadratic variation of the Brownian motion over an interval  $[s, t]$  (equipped with a subdivision  $\pi = \{s = t_0 < t_1 < \dots < t_k < \dots < t_n = t\}$ ), and defined as the limit when  $|\pi| \rightarrow 0$  of  $V_\pi^{(2)} = \sum_{k=0}^{n-1} |f(t_{k+1}) - f(t_k)|^2$ , is:

$$\lim_{|\pi| \rightarrow 0} V_\pi^{(2)} = Q(t-s) \text{ in } L^2 \quad (45)$$

# Stochastic calculus survival kit - Stochastic Integrals

Ito then proceeds to define **stochastic integrals**, starting with elementary processes:

## Definition

A stochastic process  $X = (X_s)_{s \in [a,b]}$  is called **elementary** if there exists a subdivision  $a = t_0 < t_1 < \dots < t_n = b$  of  $[a, b]$ , such that:

$$\forall t \in [a, b], \forall \omega \in \Omega, X_t(\omega) = \sum_{i=0}^{n-1} X_i(\omega) \mathbf{1}_{[t_i, t_{i+1}[}(t)$$

with  $\forall i \in \{0, 1, \dots, n-1\}$ ,  $X_i$  is  $\mathcal{F}_{t_i}$ -measurable.

This means that, in each interval  $[t_i, t_{i+1}[$ ,  $X_t(\omega)$  is independent of  $t$  and  $X_t(\omega) = X_i(\omega)$ .

We define  $\mathcal{E}$  (resp.  $\mathcal{E}_n$ ,  $n > 0$ ) the set of all elementary processes on  $[a, b]$  (resp. the subset of the  $X \in \mathcal{E}$  such that all  $X_i$  have a finite moment  $\mathbb{E}X_i < \infty$  (resp  $\mathbb{E}(|X_i|^n) < \infty$ ).

# Stochastic calculus survival kit - Stochastic Integrals 2

## Definition

Let  $X \in \mathcal{E}$ , ie

$$X_t(\omega) = \sum_{i=0}^{n-1} X_i(\omega) \mathbf{1}_{[t_i, t_{i+1}[}(t)$$

**The stochastic integral of  $X$  is the real random variable :**

$$\int_a^b X_t dB_t := \sum_{i=0}^{n-1} X_i(B_{t_{i+1}} - B_{t_i})$$

The notion is then extended to other stochastic processes (in spaces of square integrable processes, see the annex).

# Stochastic calculus survival kit - Ito's process

## Definition

A process  $X = (X_t)_{t \in [0, T]}$  is called a **Ito's process** if it can be written as:

$$X_t = X_0 + \int_0^t a_s ds + \int_0^t b_s dB_s \quad \forall t \in [0, T] \tag{46}$$

where  $a$  and  $b$  are two stochastic processes such that the integrals exist (ie  $a \in \Lambda^1$  and  $b \in \Lambda^2$ ). Equivalently, we write  $X_t$  as the solution to the **Stochastic Differential Equation**:

$$dX_t = a_t dt + b_t dB_t$$

# Stochastic calculus survival kit - Ito's formula

## Theorem

*An Itô's process remains an Itô's process when it is transformed by a deterministic function that is "smooth enough".*

*Let  $X$  be a Itô's process on  $[0, T]$  :  $dX_t = a_t dt + b_t dB_t$ .*

*Let  $f : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}, (x, t) \mapsto f(x, t)$  be  $\mathcal{C}^{2,1}$  :  $\mathcal{C}^2$  in  $x$ , and  $\mathcal{C}^1$  in  $t$ .*

*Then  $(f(X_t, t))_{t \in [0, T]}$  is also an Itô's process and:*

$$d(f(X_t, t)) = \frac{\partial f}{\partial t}(X_t, t)dt + \frac{\partial f}{\partial x}(X_t, t)dX_t + \frac{1}{2} \frac{\partial^2 f}{\partial x^2}(X_t, t)b_t^2 dt \quad (47)$$

*The last term is Itô's complementary term.*

*In dimension  $d > 1$ :*

$$d(f(X_t, t)) = \frac{\partial f}{\partial t}(X_t, t)dt + (\nabla f)^T(X_t, t)dX_t + \frac{1}{2} \text{Tr}\left((\nabla \nabla^T f)dX_t dX_t^T\right) \quad (48)$$

# Definition of a Stochastic Differential Equation

## Definition

Let:

- $B$  be a Brownian motion  $B_t \in \mathbb{R}^S$ , of diffusion matrix  $Q$
- $F$  be a deterministic function **drift**  $F : \mathbb{R}^D \times \mathbb{R} \rightarrow \mathbb{R}^{D \times D}$
- $L$  be a deterministic function **diffusion** (aka dispersion)  $L : \mathbb{R}^D \times \mathbb{R} \rightarrow \mathbb{R}^{D \times S}$

The SDE is:

$$dX_t = F(X_t, t)dt + L(X_t, t)dB_t \quad (49)$$

$$X_{t_0} = X_0 \quad (50)$$

where  $X_0$  can be a scalar constant or a random variable. A stochastic process  $X$  is said to be solution of 49 if it verifies:

$$\forall t, \quad X_t = X_0 + \int_0^t F(X_u, u)du + \int_0^t L(X_u, u)dB_u \quad (51)$$

## A solution to an SDE is a Markov Process

- As for Ordinary Differential Equation (ODE), a solution to 49 might not exist. Also, results similar to Cauchy-Lipschitz exist for existence and unicity, based on assumptions on  $F$  and  $L$ .
- Intuitively, we can see that an "infinitesimal increment" of  $X_t$  to  $X_{t+\Delta t}$  verifies :  
 $\Delta X_t \approx F(X_t, t)\Delta t + L(X_t, t)dB_t$ . But  $dB_t$  is a Brownian increment independent of  $X_t$ , This suggests that  $X_{t+\Delta t}$  depends on the past only by  $X_t$ .
- In other words,  $X_t|\mathcal{F}_s = X_t|X_s$  for any  $0 < s < t$ . ie : **the solution of a SDE is a Markov process.** (The formal proof is given in [17].)

The solution to a SDE is a Markov process

## Transition Kernels

- Formally, a Markov process is characterized by its **transition kernels**.
- That is, for any  $s < t$ , and any  $A \in \mathcal{B}_{\mathbb{R}^D}$ , a Markov process verifies  $\mathbb{P}(X_t \in A | \mathcal{F}_s) = \mathbb{P}(X_t \in A | X_s)$ .
- the transition kernels of  $X$  are the applications  $P_{s,t} : \mathbb{R}^D \times \mathcal{B}_{\mathbb{R}^D} \rightarrow [0, 1]$ , such that for any  $f : \mathbb{R}^D \rightarrow \mathbb{R}$  measurable and bounded, we have:

$$P_{s,t}f(x) = \int_{\mathbb{R}^D} P_{s,t}(x, dy)f(y) \quad (52)$$

So  $P_{s,t}$  actually is the probability measure of starting from  $x$  at time  $s$ , and reach  $y \in dy$  at time  $t$ .

When the transition kernels have densities  $p(x, t|y, s)$  (ie starting from  $y$  at time  $s$ , and reaching  $x$  at time  $t$ ), then a fundamental result is the **Fokker Plank Kolmogorov** equation (also known as forward Kolomogorov) :

$$\frac{\partial p}{\partial y} = \mathcal{A}^* p \quad (53)$$

$$\mathcal{A}^*(\bullet) = - \sum_{i=1}^D \frac{\partial}{\partial x_i} (F_i(x, t)(\bullet)) + \frac{1}{2} \sum_{i,j=1}^D \frac{\partial^2}{\partial x_i \partial x_j} (L(x, t) Q L(x, t)^T |_{i,j} (\bullet)) \quad (54)$$

# Linear SDE

A particularly useful flavor of SDE is the linear SDE, that allows some close-form (or at least nicer) solutions:

## Definition

With the same notations as 49:

The linear SDE is:

$$dX_t = F(t)X_t dt + L(t)dB_t \quad (55)$$

$$X_{t_0} = X_0 \sim \mathcal{N}(m_0, P_0) \quad (56)$$

## Transition kernels for linear SDEs

In this case, the transition kernels family and the solution write:

$$\Psi : \mathbb{R}^2 \rightarrow \mathbb{R}^D \quad (57)$$

$$\frac{\partial \Psi(\tau, t)}{\partial \tau} = F(\tau)\Psi(\tau, t) \quad (58)$$

$$\frac{\partial \Psi(\tau, t)}{\partial t} = -\Psi(\tau, t)F(t) \quad (59)$$

$$\Psi(\tau, t) = \Psi(\tau, s)\Psi(s, t) \text{ (Chapman-Kolmogorov)} \quad (60)$$

$$\Psi(\tau, t) = \Psi(t, \tau)^{-1} \quad (61)$$

$$\Psi(t, t) = I_d \quad (62)$$

$$X_t = \Psi(t, t_0)X_0 + \int_{t_0}^t \Psi(t, \tau)L(\tau)dB_\tau \quad (63)$$

$$X_{t_0} = X_0 \sim \mathcal{N}(m_0, P_0) \quad (64)$$

The solution to a Linear SDE is a Gaussian process. (The converse is NOT true!)

## Gaussian Processes as solutions of linear SDE

- **Brownian motion** : solution to  $dZ_t = dB_t$ , GP with kernel  $k(t, t') = \min(t, t')$  (see ??)
- **Ornstein Uhlenbeck** : the O.U. process

$$dZ_t = -\frac{1}{I} Z_t dt + dB_t \quad (65)$$

where  $dB_t$  has diffusion coefficient  $\frac{2\sigma^2}{I}$ , is a GP with kernel:

$$k_{\text{exp}} = \sigma^2 \exp\left(-\frac{|t - t'|}{I}\right) \quad (66)$$

- **Matern** : the SDE representation with

$$F = \begin{pmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{pmatrix}, L = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, H = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (67)$$

is a GP with the Matern kernel with  $\nu = \frac{3}{2}$ :

$$k_{\text{Matern}} = \sigma^2 \left(1 + \frac{\sqrt{3}|t - t'|}{I}\right) \exp\left(-\frac{\sqrt{3}|t - t'|}{I}\right) \quad (68)$$

and  $\lambda = \frac{\sqrt{3}}{I}$ , diffusion is  $q = 4\lambda^3\sigma^2$ .

## Some Gaussian Processes are NOT solutions to linear SDE

Conversely, the following kernels can not be used to derive an associated linear SDE:

- **squared exponential** : the widely used

$$k_{\text{se}}(t, t') = \sigma^2 \exp\left(-\frac{|t - t'|^2}{2l^2}\right) \quad (69)$$

- **rational quadratic**:

$$k_{\text{rq}}(t, t') = \sigma^2 \left(1 + \frac{|t - t'|^2}{2\alpha l^2}\right)^{-\alpha} \quad (70)$$

with  $\alpha > 0$ .

In that latter case, one can use spectral decomposition (ie Mercer's theorem, see MVA kernel class [6]) to approximate the kernel function and determine an associated linear SDE.

## General Filtering/Smoothing equations with SDEs

- In practice, we posit a stochastic process prior defined by a general SDE, and we have discrete-time measurements.
- Formally, the Continuous-Discrete State Space Model (CD-SSM) is defined by:

### Continuous-Discrete State Space model

$$dZ_t = F(Z_t, t)dt + L(Z_t, t)dB_t \quad (71)$$

$$x_k \sim p(x_k | z_{t_k}) \quad (72)$$

where:

- $Z_t \in \mathbb{R}^D$  is the *state*, ie a stochastic process defining the latent variable.
- $B_t \in \mathbb{R}^S$  is a Brownian motion with diffusion matrix  $Q$ .
- $F \in \mathbb{R}^D$  and  $L \in \mathbb{R}^{D \times S}$  are the usual drift and dispersion functions.
- $x_k$  are the observations taken at **discrete times**  $(t_k)_{k=1, \dots, n}$

NB : the observations are assumed to conditionnally independent of the state.

# Filtering

- **Filtering** is the problem of determining the posterior probability of the latent  $Z_t$  given the discrete measurements, ie finding  $p(Z_t|x_{1:k})$  with  $t_k \leq t$ . This corresponds to determining the generative transition probability  $p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})$  in our DVAE setting.
- In general, close-form solutions can be derived when the latent variables SDE is linear. In continuous time, we get the **Kalman-Bucy** filter equations, which discretize in the well-known **Kalman filter**.

# Filtering

## Kalman-Bucy filter

$$dZ_t = F(t)Z_t dt + L(t)dB_t \quad (73)$$

$$dX_t = H(t)X_t dt + d\eta_t \quad (74)$$

NB : the observations are assumed to conditionnally independent of the state. Then the Bayesian filter (Kalman-Bucy) is:

$$p(z_t | x_{<t}) = \mathcal{N}(Z_t | m_t, P_t) \quad (75)$$

$$K = PH(t)^T R^{-1} \quad (76)$$

$$dm = F(t)mdt + K(dX_t - H(t)mdt) \quad (77)$$

$$\frac{dP}{dt} = F(t)P + PF(t)^T + L(t)QL(t)^T - KRK^T \quad (78)$$

# Smoothing

- **Smoothing** is the problem of determining the posterior probability of the latent  $Z_t$  given all known observations, ie finding  $p(Z_t|x_{1:T})$  for all  $t \in [0, T]$ .
- This corresponds to determining the inference model  $q_\phi(z_t|z_{1:t-1}, x_{1:T})$  in the DVAE setting.
- Discretizing the transition density in CD-SSM, we have

$$Z_{t_{k+1}} \sim p(Z_{t_{k+1}}|Z_{t_k}) \quad (79)$$

$$X_k \sim p(X_k|Z_{t_k}) \quad (80)$$

## Bayesian smoother

$$Z_{t_{k+1}} \sim p(Z_{t_{k+1}} | Z_{t_k}) \quad (81)$$

$$X_k \sim p(X_k | Z_{t_k}) \quad (82)$$

The *Bayesian smoother* is, for any  $k < T$ :

$$p(Z_{t_{k+1}} | X_{1:k}) = \int p(Z_{t_{k+1}} | Z_{t_k}) p(Z_{t_k} | X_{1:k}) dZ_{t_k} \quad (83)$$

$$p(Z_{t_k} | X_{1:T}) = p(Z_{t_k} | X_{1:k}) \int \left( \frac{p(Z_{t_{k+1}} | Z_{t_k}) p(Z_{t_{k+1}} | X_{1:T})}{p(Z_{t_{k+1}} | X_{1:k})} dZ_{t_{k+1}} \right) \quad (84)$$

The backward recursion is started from the final step, where the filtering and smoothing densities are the same :  $p(Z_{t_T} | X_{1:T})$ .

## GP-VAE with accomodating kernels : filtering/smoothing in $O(n)$

- We wrap up here linking the filtering/smoothing theory of linear SDE with the GP-VAE model of [9].
- Using the formalization above, a GP-VAE with Gaussian observation is basically:

$$Z_t \sim \mathcal{GP}(m(\bullet), k(\bullet, \bullet)) \quad (85)$$

$$X_{t_k} \sim \mathcal{N}(X_{t_k} | Z_{t_k}, \sigma^2) \quad (86)$$

- Computing the posterior distribution  $p(Z_t | X_{t_1:t_T})$  is performing a Gaussian Process regression (see [23]), which naively scales in  $O(n^3)$ .
- However, if the Gaussian process can be written as a linear SDE:

$$dZ_t = F(t)Z_t dt + L(t)dB_t \quad (87)$$

$$X_{t_k} \sim \mathcal{N}(X_{t_k} | Z_{t_k}, \sigma^2) \quad (88)$$

then the Kalman filter and smoother apply, that scale in  $O(n)$ .

GP Prior regression in  $O(N)$  cost

If the GP prior can be written as the solution to a linear SDE, then the GP prior regression problem can be performed with linear cost with the Kalman-Bucy equations.

# Latent ODE

- Latent ODEs is a class of models introduced in [4] : "Neural Ordinary Differential Equations" by Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, David Duvenaud. ArXiV : Neural ODE Best Paper Award NeurIPS 2018.
- Starting point : assume ResNet-like evolution of latent

$$z_{t+1} = z_t + f(z_t, \theta_t) \quad (89)$$

- becomes in continuous time:

$$\frac{dz_t}{dt} = f(z_t, t, \theta_f) \quad (90)$$

where  $\theta_f$  is a set of parameters, that can typically be the parameters of a neural network learning  $f$ .

# Latent ODE - model

- Generative model

$$z_{t_0} \sim p_{\theta_z}(z_{t_0}) \quad (91)$$

$$z_{t_1}, z_{t_2}, \dots, z_{t_N} = \text{ODE Solver}(z_{t_0}, f, \theta_f, t_0, \dots, t_N) \quad (92)$$

$$x_{t_i} \sim p_{\theta_x}(x_t | z_t) \quad (93)$$

- Inference

$$[\mu_\phi, \Sigma_\phi] = \text{LSTM}(x_{t_0:t_N}) \quad (94)$$

$$q_\phi(z_{t_0} | x_{t_0:t_N}) = \mathcal{N}(z_{t_0} | \mu_\phi, \Sigma_\phi) \quad (95)$$

## Latent ODE - model

We reproduce here the drawing from the paper:

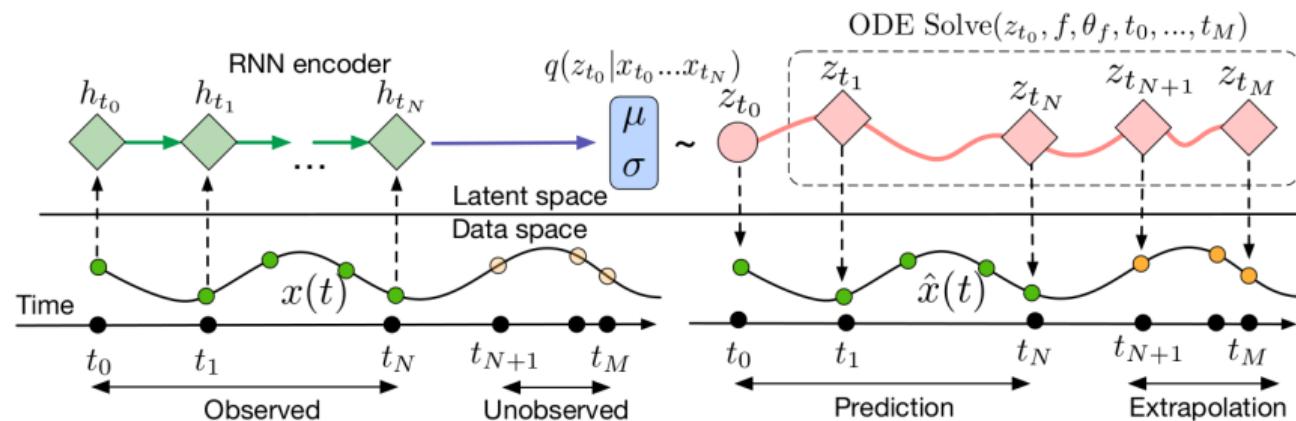


Figure: Neural ODE model

## Latent ODE - ELBO

- the stochastic variables are  $z_{t_0}$  and the  $x_{t_i}$ 's.
- the joint distribution writes:

$$p(x_{t_1:t_N}, z_{t_0}) = p(z_{t_0}) \prod_{t=1}^N p_{\theta_x}(x_{t_i}|z_{t_i}) \quad (96)$$

- likelihood:

$$\log p(x_{t_1:t_N}) \geq \mathbb{E}_{q_\phi(z_{t_0}|x_{t_1:t_N})} \log \frac{p(x_{t_1:t_N}, z_{t_0})}{q_\phi(z_{t_0}|x_{t_1:t_N})} \quad (97)$$

$$= \sum_{i=1}^N \mathbb{E}_{q_\phi(z_{t_0}|x_{t_1:t_N})} \log p_{\theta_x}(x_{t_i}|z_{t_i}) - \text{KL}(q_\phi(z_{t_0}|x_{t_1:t_N})||p(z_{t_0})) \quad (98)$$

- Need to compute the gradients of  $\log p_{\theta_x}(x_{t_i}|z_{t_i})$  w.r.t.  $\theta_f$ . Methods are forward sensitivity, backpropagation through ODE solver, or **adjoint sensitivity method**. (see [22], [27]).

## Latent SDE Model

Latent ODEs models have limitations :

- the latent dynamic is deterministic by design
- the initial variable  $z_{t_0}$  encompasses the entire randomness of the prior, and can become unnaturally large to account for randomness along the entire timeline

The idea in [15] is to add some noise to the deterministic computation of the latent variable:

$$\frac{dz_t}{dt} = f_{\theta_f}(z_t, t) + \epsilon_t \quad (99)$$

$$\epsilon_t \sim \mathcal{N}(0, QId) \quad (100)$$

Which leads to an SDE prior:

$$dZ_t = f_{\theta}(Z_t, t)dt + \sigma_{\theta}(Z_t, t)dB_t \quad (101)$$

$$Z_{t_0} \sim Z_0 \quad (102)$$

(where we used  $\sigma_{\theta}$  instead of our usual  $L(Z_t, t)$  to stick to the notations of the paper).

101 defines a prior distribution over functions. In order to draw a sample function, we would:

- draw a sample  $z_{t_0} \sim Z_0$
- draw a random Brownian motion  $\tilde{B}_t$  path from  $B_t$
- compute  $z_t - z_{t_0} = \int_{t_0}^t f_{\theta}(Z_t, t)dt + \int_{t_0}^t \sigma_{\theta}(Z_t, t)d\tilde{B}_t$

## Latent SDE - generative model

The approximate posterior can also be described as a SDE:

$$dZ_t = f_\phi(Z_t, t)dt + \sigma_{\phi=\theta}(Z_t, t)dB_t \quad (103)$$

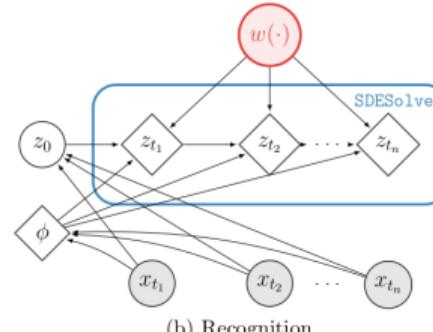
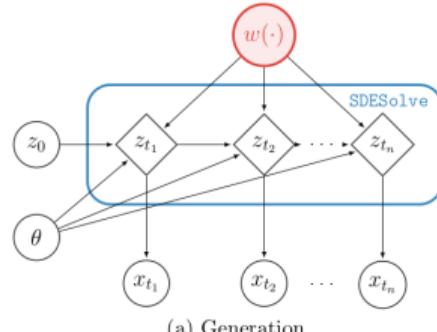
$$Z_{t_0} = z_0 \text{ from prior} \quad (104)$$

- the prior and the approximate posterior share the same diffusion for the KL to have the same support
- the prior and the approximate posterior have the same starting value  $z_0$

---

Xuechen Li\*, Ting-Kam Leonard Wong, Ricky T. Q. Chen, David Duvenaud

---



## Latent SDE - inference

- neural networks to learn the drift  $f_\phi$  and the diffusion  $\sigma_\theta$
- requires to compute the gradient of functionals (loss) of type:

$$L(\theta, \phi) = L \left( \int_{t_0}^{t_1} f_\phi(Z_t, t) dt + \int_{t_0}^t \sigma_\theta(Z_t, t) dB_t \right) \quad (105)$$

where  $\int_{t_0}^t \sigma_\theta(Z_t, t) dB_t$  is actually a random variable!

- It appears that the adjoint sensitivity method can be adapted to SDEs (see [15]).

The stochastic VLB writes:

$$\mathcal{L}(\theta, \phi, x_{1:T}) = \mathbb{E} \left( \frac{1}{2} \int_0^T \left| \frac{f_\theta(z_t, t) - f_\phi(z_t, t)}{\sigma_\theta(z_t, t)} \right|^2 dt - \sum_{i=1}^N \log p_{\theta_x}(x_{t_i} | z_{t_i}) \right) \quad (106)$$

## Take-aways

- DVAEs are a natural and powerful extension of VAEs in which the prior expresses the temporal dependency of the data sequence.
- Discrete-time DVAEs use discretized latent priors to encode temporal dynamics. They work best with regularly spaced data.
- Continuous-time DVAEs posit a stochastic process as prior for the latent variables. This allows additional flexibility and irregularly-sampled data.
- In GP-VAE, the latent prior is a Gaussian Process that can sometimes be expressed as a solution to a linear SDE. In that case, Kalman-Bucy filtering and smoothing algorithms can speed up computations.
- Stochastic calculus provides a framework for more general continuous-time priors.
- When expressed as solutions to general SDE, those more general stochastic process priors are used in Latent SDEs.

## Take-aways

- DVAEs are a natural and powerful extension of VAEs in which the prior expresses the temporal dependency of the data sequence.
- Discrete-time DVAEs use discretized latent priors to encode temporal dynamics. They work best with regularly spaced data.
- Continuous-time DVAEs posit a stochastic process as prior for the latent variables. This allows additional flexibility and irregularly-sampled data.
- In GP-VAE, the latent prior is a Gaussian Process that can sometimes be expressed as a solution to a linear SDE. In that case, Kalman-Bucy filtering and smoothing algorithms can speed up computations.
- Stochastic calculus provides a framework for more general continuous-time priors.
- When expressed as solutions to general SDE, those more general stochastic process priors are used in Latent SDEs.

## Take-aways

- DVAEs are a natural and powerful extension of VAEs in which the prior expresses the temporal dependency of the data sequence.
- Discrete-time DVAEs use discretized latent priors to encode temporal dynamics. They work best with regularly spaced data.
- Continuous-time DVAEs posit a stochastic process as prior for the latent variables. This allows additional flexibility and irregularly-sampled data.
- In GP-VAE, the latent prior is a Gaussian Process that can sometimes be expressed as a solution to a linear SDE. In that case, Kalman-Bucy filtering and smoothing algorithms can speed up computations.
- Stochastic calculus provides a framework for more general continuous-time priors.
- When expressed as solutions to general SDE, those more general stochastic process priors are used in Latent SDEs.

## Take-aways

- DVAEs are a natural and powerful extension of VAEs in which the prior expresses the temporal dependency of the data sequence.
- Discrete-time DVAEs use discretized latent priors to encode temporal dynamics. They work best with regularly spaced data.
- Continuous-time DVAEs posit a stochastic process as prior for the latent variables. This allows additional flexibility and irregularly-sampled data.
- In GP-VAE, the latent prior is a Gaussian Process that can sometimes be expressed as a solution to a linear SDE. In that case, Kalman-Bucy filtering and smoothing algorithms can speed up computations.
- Stochastic calculus provides a framework for more general continuous-time priors.
- When expressed as solutions to general SDE, those more general stochastic process priors are used in Latent SDEs.

## Take-aways

- DVAEs are a natural and powerful extension of VAEs in which the prior expresses the temporal dependency of the data sequence.
- Discrete-time DVAEs use discretized latent priors to encode temporal dynamics. They work best with regularly spaced data.
- Continuous-time DVAEs posit a stochastic process as prior for the latent variables. This allows additional flexibility and irregularly-sampled data.
- In GP-VAE, the latent prior is a Gaussian Process that can sometimes be expressed as a solution to a linear SDE. In that case, Kalman-Bucy filtering and smoothing algorithms can speed up computations.
- Stochastic calculus provides a framework for more general continuous-time priors.
- When expressed as solutions to general SDE, those more general stochastic process priors are used in Latent SDEs.

## Take-aways

- DVAEs are a natural and powerful extension of VAEs in which the prior expresses the temporal dependency of the data sequence.
- Discrete-time DVAEs use discretized latent priors to encode temporal dynamics. They work best with regularly spaced data.
- Continuous-time DVAEs posit a stochastic process as prior for the latent variables. This allows additional flexibility and irregularly-sampled data.
- In GP-VAE, the latent prior is a Gaussian Process that can sometimes be expressed as a solution to a linear SDE. In that case, Kalman-Bucy filtering and smoothing algorithms can speed up computations.
- Stochastic calculus provides a framework for more general continuous-time priors.
- When expressed as solutions to general SDE, those more general stochastic process priors are used in Latent SDEs.

# Thank you for your attention!

## References |

- [1] C Bishop. *Pattern Recognition and Machine Learning*. Accessed on Month Day, Year. 2006. URL: <https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf>.
- [2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. eng. Softcover reprint of the original 1st edition 2006 (corrected at 8th printing 2009). Information science and statistics. New York, NY: Springer New York, 2016. ISBN: 9781493938438.
- [3] Francesco Paolo Casale et al. *Gaussian Process Prior Variational Autoencoders*. en. Oct. 2018. URL: <https://arxiv.org/abs/1810.11738v2> (visited on 06/07/2025).
- [4] Ricky T. Q. Chen et al. *Neural Ordinary Differential Equations*. arXiv:1806.07366. Dec. 2019. DOI: 10.48550/arXiv.1806.07366. URL: <http://arxiv.org/abs/1806.07366> (visited on 09/05/2025).
- [5] Junyoung Chung et al. *A Recurrent Latent Variable Model for Sequential Data*. arXiv:1506.02216. Apr. 2016. DOI: 10.48550/arXiv.1506.02216. URL: <http://arxiv.org/abs/1506.02216> (visited on 06/07/2025).
- [6] *Course materials: (Slides) - Machine learning with kernel methods / Spring 2025*. URL: <https://mva-kernel-methods.github.io/course-page/> (visited on 08/11/2025).

## References II

- [7] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory 2nd Edition*. English. Hoboken, N.J: Wiley-Interscience, 2006. ISBN: 9780471241959.
- [8] Alfonso Delgado-Bonal and Alexander Marshak. "Approximate Entropy and Sample Entropy: A Comprehensive Tutorial". en. In: *Entropy* 21.6 (June 2019), p. 541. ISSN: 1099-4300. DOI: 10.3390/e21060541. URL: <https://www.mdpi.com/1099-4300/21/6/541> (visited on 01/20/2025).
- [9] Vincent Fortuin et al. *GP-VAE: Deep Probabilistic Time Series Imputation*. arXiv:1907.04155. Feb. 2020. DOI: 10.48550/arXiv.1907.04155. URL: <http://arxiv.org/abs/1907.04155> (visited on 06/07/2025).
- [10] *Gaussian Processes for Machine Learning: Book webpage*. URL: <https://gaussianprocess.org/gpml/> (visited on 06/07/2025).
- [11] Laurent Girin et al. *Dynamical Variational Autoencoders: A Comprehensive Review*. arXiv:2008.12595. July 2022. DOI: 10.48550/arXiv.2008.12595. URL: <http://arxiv.org/abs/2008.12595> (visited on 01/19/2025).
- [12] Diederik P. Kingma and Max Welling. "An Introduction to Variational Autoencoders". In: (2019). DOI: 10.48550/ARXIV.1906.02691. URL: <https://arxiv.org/abs/1906.02691> (visited on 07/21/2025).

## References III

- [13] Diederik P. Kingma and Max Welling. *Auto-Encoding Variational Bayes*. arXiv:1312.6114. Dec. 2022. DOI: 10.48550/arXiv.1312.6114. URL: <http://arxiv.org/abs/1312.6114> (visited on 08/08/2025).
- [14] Friedman Koller. *Probabilistic Graphical Models*. en-US. URL: <https://mitpress.mit.edu/9780262013192/probabilistic-graphical-models/> (visited on 07/21/2025).
- [15] Xuechen Li et al. *Scalable Gradients for Stochastic Differential Equations*. arXiv:2001.01328. Oct. 2020. DOI: 10.48550/arXiv.2001.01328. URL: <http://arxiv.org/abs/2001.01328> (visited on 09/05/2025).
- [16] Yingzhen Li and Stephan Mandt. *Disentangled Sequential Autoencoder*. arXiv:1803.02991. June 2018. DOI: 10.48550/arXiv.1803.02991. URL: <http://arxiv.org/abs/1803.02991> (visited on 07/23/2025).
- [17] *Mouvement brownien et calcul d'Itô-Léonard Gallardo*-Editions Hermann. Sept. 2008. URL: <https://www.editions-hermann.fr/livre/mouvement-brownien-et-calcul-d-ito-leonard-gallardo> (visited on 04/16/2025).
- [18] K Murphy. *Probabilistic Machine Learning Advanced Topics*. 2023. URL: <https://mitpress.mit.edu/9780262048439/probabilistic-machine-learning/>.

## References IV

- [19] *Page Web de Jean-François Le Gall.* URL:  
<https://www.imo.universite-paris-saclay.fr/~jean-francois.le-gall/> (visited on 04/16/2025).
- [20] Stefano Peluchetti and Stefano Favaro. *Infinitely deep neural networks as diffusion processes.* arXiv:1905.11065. Mar. 2020. DOI: 10.48550/arXiv.1905.11065. URL:  
<http://arxiv.org/abs/1905.11065> (visited on 09/05/2025).
- [21] S M Pincus. "Approximate entropy as a measure of system complexity." . en. In: *Proceedings of the National Academy of Sciences* 88.6 (Mar. 1991), pp. 2297–2301. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.88.6.2297. URL: <https://pnas.org/doi/full/10.1073/pnas.88.6.2297> (visited on 01/20/2025).
- [22] L S. Pontriagin et al. *The mathematical theory of optimal processes.* eng. Ed. by Lucien W. Neustadt. Classics of Soviet mathematics. OCLC: 1035389999. Boca Raton: CRC Press, 2018. ISBN: 9780203749319.
- [23] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning.* eng. 3. print. Adaptive computation and machine learning. Cambridge, Mass.: MIT Press, 2008. ISBN: 9780262182539.

## References V

- [24] S. Roberts et al. "Gaussian processes for time-series modelling". en. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.1984 (Feb. 2013), p. 20110550. ISSN: 1364-503X, 1471-2962. DOI: 10.1098/rsta.2011.0550. URL: <https://royalsocietypublishing.org/doi/10.1098/rsta.2011.0550> (visited on 07/23/2025).
- [25] Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. *Latent ODEs for Irregularly-Sampled Time Series*. arXiv:1907.03907. July 2019. DOI: 10.48550/arXiv.1907.03907. URL: <http://arxiv.org/abs/1907.03907> (visited on 09/05/2025).
- [26] Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Institute of Mathematical Statistics Textbooks. Cambridge: Cambridge University Press, 2019. ISBN: 9781316510087. DOI: 10.1017/9781108186735. URL: <https://www.cambridge.org/core/books/applied-stochastic-differential-equations/6BB1B8B0819F8C12616E4A0C78C29EAA> (visited on 07/23/2025).
- [27] B. Sengupta, K.J. Friston, and W.D. Penny. "Efficient gradient computation for dynamical models". en. In: *NeuroImage* 98 (Sept. 2014), pp. 521–527. ISSN: 10538119. DOI: 10.1016/j.neuroimage.2014.04.040. URL: <https://linkinghub.elsevier.com/retrieve/pii/S1053811914003097> (visited on 09/07/2025).

## References VI

- [28] C. E. Shannon. "A mathematical theory of communication". In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 379–423. ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1948.tb01338.x. URL: <https://ieeexplore.ieee.org/document/6773024> (visited on 07/28/2025).
- [29] Michalis Titsias and Neil D. Lawrence. "Bayesian Gaussian Process Latent Variable Model". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, May 2010, pp. 844–851. URL: <https://proceedings.mlr.press/v9/titsias10a.html>.
- [30] Harrison Zhu, Carles Balsells Rodas, and Yingzhen Li. *Markovian Gaussian Process Variational Autoencoders*. arXiv:2207.05543. Aug. 2023. DOI: 10.48550/arXiv.2207.05543. URL: <http://arxiv.org/abs/2207.05543> (visited on 07/23/2025).

# Glossary I

**CD-SSM** Continuous-Discrete State Space Model. 62, 65

**DAG** Directed Acyclic Graph. 18

**DKF** Deep Kalman Filter. 3–6

**DVAE** Dynamical Variational Auto Encoder. 3–6, 19, 25, 63, 65, 75–80

**ELBO** Evidence Lower Bound. 21

**GP** Gaussian Process. 3–6, 29, 31, 60, 67

**GP-VAE** Gaussian Process Variational Auto Encoder. 3–6, 67, 75–80

**GPM** Graphical Probabilistic Model. 7, 8, 10–13, 25

**Latent ODE** Latent Ordinary Differential Equation model. 3–6, 68, 72

**Latent SDE** Latent Stochastic Differential Equation model. 3–6, 75–80

**LSTM** Long Short Term Memory. 23, 27

## Glossary II

**MLP** Multi Layer Perceptron. 23

**ODE** Ordinary Differential Equation. 56, 71

**SDE** Stochastic Differential Equation. 3–6, 55, 56, 58–63, 67, 72, 73, 75–80

**SSM** State Space Model. 18

**VAE** Variational Auto Encoder. 3–6, 75–80

**VLB** Variational Lower Bound. 7, 8, 22, 25, 26, 74

**VRNN** Variational Recurrent Neural Network. 3–6, 25