Dynamical Variational Autoencoders : discrete-time and continuous-time models. Links to Stochastic calculus

ENS Paris-Saclay, MVA

Benjamin Deporte : benjamin.deporte@ens-paris-saclay.fr

August 2025

Abstract

This report describes a particular class of Variational Auto Encoders (VAEs) : the Dynamical Variational Auto Encoders (DVAEs). DVAEs are a specific class of models, adapted to the study of data sequences. In DVAEs, the latent variables are structured themselves as a correlated set (usually a sequence also), aiming at encoding the temporal dimension of the data. We will review the general formulation of DVAEs and the detailed implementation of three models : extended Kalman filter, Variational Recurrent Neural Network (VRNN), and Gaussian Process Variational Auto Encoder (GP-VAE).

We then provide an overview of the information theory framework for data sequences -specifically some results on entropy rates- as an attempt to empirically quantify the degree of ”randomness” of data sequences. In doing so, we can try and evaluate the expressiveness of DVAEs, and their relative performance with respect to other well-known models (such as Long Short Term Memorys (LSTMs)).

However, it is the theory of stochastic calculus that will provide us with great insights regarding the expressiveness of DVAEs. First, we will see that the solution of a Stochastic Differential Equation (SDE) is a Markov Process, that can quite naturally be expressed as a DVAE. Furthermore, if the SDE is actually linear, we will see that its solution is actually a Gaussian Process, and fits naturally into the GP-VAE model. All the associated results of the kernels theory (notably results on the spectral theory) then apply.

Armed with those results, we carry out some experiments, to demonstrate... SURPRISE.

The code is available at : `https://github.com/BenjaminDeporte/MVA_Stage`

Acknowledgements

Contents

List of Figures

Part I

Introduction

1 Subject

Variational Autoencoders are a well-known class of generative models, where the latent variables are usually assumed to be idependent and identically distributed. This assumption is inappropriate when the data is time-dependent, such as in time-series, images sequences or videos. It is then natural to structure some sort of temporal dependency in the latent prior : this is the main idea of Dynamical VAEs.

A first question is whether the use of Dynamical VAEs on time-dependent data allow better performance than the ”legacy” models. A natural test framework is time series, where the usual models -such as ARIMA- have been performing successfully for quite a while.

A second question takes the thinking a little bit deeper. If we consider a time-dependent data sequence as the realization of a stochastic process -which is quite natural for time series, but can be envisioned as well for videos or other sources-, then we infer that some data sequences are, by design, easier than others to predict and learn generative models on. An idea is then to quantity the ”randomness” of a data sequence -and more importantly, the ”randomness” of the underlying stochastic process-, so we can measure when the use of Dynamical VAEs is likely to add value. A natural tool is Information Theory, where some results exist regarding stochastic processes.

Last but not least, assuming that we have learnt ”reasonably well” a generative model, we can wonder how good the model performs at detecting anomalies in the data sequence. The Pandora’s box of anomaly detection is vast, and covers connex notions such as detecting outliers in a stationary distribution, detecting shifts of a distribution towards a new one, etc. If using the likelihood of the data according to the learnt distribution is quite straightforward an idea, some recent results show that this may be misleading.

2 Report Outline

Part II covers the mathematical background that is required for most of the report. Section 1 covers the main results regarding stochastic processes and introduces the Brownian motion. (Stochastic Differential Equations are introduced later in Part III). Section 2 recaps the main first definitions and results regarding information theory, its application to stochastic processes, some theoretical results for stationary processes and experimental metrics.

Part II covers discrete-time dynamical Variational AutoEncoders. Section 1 is a recap of graphical models and D-Separation, that is used throughout the report. Section 2 introduces the generic formulation of DVAEs : generative and inference models, variational lower bound. Section 3 presents the first, and most simple model : the Deep Kalman Filter. Section 4 presents the most expressive model, ie the Variationnal RNN. Section 5 is a summary of the take-aways regarding the PyTorch-implementations.

Part III takes us to continuous-time Dynamical VAEs. Section 1 is a recap of the theory regarding Gaussian Processes, that will be at the core of this part. Section 2 is intended to be a self-sustained presentation of the stochastic calculus, where we go quickly over the construction of the Ito's integral to derive the necessary Ito's lemma and subsequent calculation rules for the rest of the report. Section 3 presents the GP-VAE model. Section 4 is a summary of the take-aways regarding the PyTorch implementation.

Part IV summarizes the experiments : Blembet UK tides, Brownian motion, O.U + Student observation model, cryptos...

Part V is an opening to what could be covered next ! Section 1 presents the link between SDEs and Markovian kernel Gaussian Processes. Section 2 presents some results regarding information theory and stochastic processes.

3
Related work

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Part II

Dynamical Variational Autoencoders

This part presents the general framework of DVAEs, after a reminder of the key notion of **D-separation**, which is key in Graphical Probabilistic Models (GPMs). Three models are described in details:

- **Deep Kalman Filter** : this model arises as the first evolution of the well-known Kalman Filter, with richer, Multi Layer Perceptrons (MLPs) networks for the encoder and decoder.
- **Variational Recurrent Neural Network** : at the other end of the spectrum, VRNNs provide the most expressive discrete-time formulation of the encoder and decoder. We describe here a different implementation from the one described in [14].
- **Gaussian Process Variational Auto Encoder** : in this model, the prior over the latent variables is no longer discrete, but is a Gaussian Process (GP). This allows for sampling data at irregular intervals. Another benefit is the use of richer kernel families to encode prior knowledge.

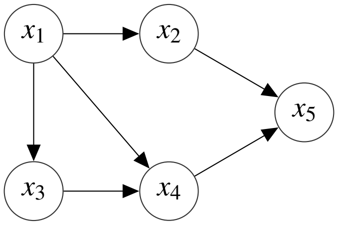
4 D-separation

Graphical models are an efficient way to describe families of factorized joint distributions of a data set $(x_i)_{i=1,...,n}$ into a Directed Acyclic Graph (DAG).

Given such a dataset, we can build a DAG where each node is indexed by an integer that is higher than the indexes of its *parent* nodes, such that the joint distribution over the dataset factorizes as:

$$p(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} p(x_i|pa_i) \tag{4.1}$$

where pa_i is the set of parent nodes of x_i.
For example, the following DAG



describes:

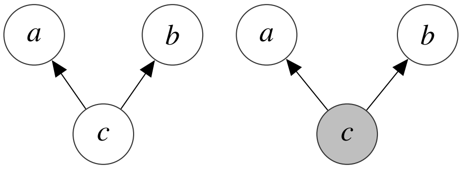
$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_1, x_3)p(x_5|x_2, x_4) \tag{4.2}$$

Describing a factorized joint probability distribution by a DAG allows to determine graphically whether two sets of nodes (ie random variables) are independent, conditioned on a third set of nodes. This allows subsequently to simplify the expressions of the observation models (ie $p(x|z)$, and/or the posterior models ($q(z|x)$).

D-Separation is the set of rules that determine whether there is conditional independence between two sets given a third one. D-Separation is well described in key books such as [3], [17] or [15]. We will enunciate here the key concepts, and refer the interested reader to those books.

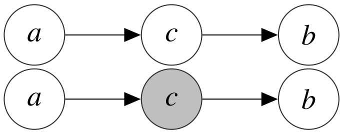
D-Separation is a way to find out graphically conditional (in)dependence relationships between random variables, that would be more difficult to calculate by marginalizing the joint distribution over the conditioning variables. A nice way to demonstrate this, is to review the three examples of 3-node DAG. NB : The observed (ie conditioning) variables are noted with gray background.

Example 1 : c is said *tail-to-tail* with a and b, and *blocks the path between a and b*, making them conditionally independent : $a \not\perp\!\!\!\perp b \,|\, \varnothing$, $a \perp\!\!\!\perp b \,|\, c$



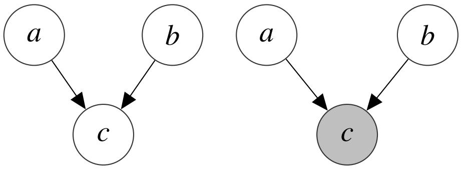
$$\begin{aligned} p(a,b,c) &= p(c)p(a|c)p(b|c) \\ p(a,b) &= \sum_c p(c)p(a|c)p(b|c) \neq p(a)p(b) \\ &\implies a \not\perp\!\!\!\perp b \,|\, \emptyset \\ p(a,b|c) &= p(a|c)p(b|c) \\ &\implies a \perp\!\!\!\perp b \,|\, c \end{aligned}$$

Example 2 : a, c, b form a Markov chain. c is said *head-to-tail* with a and b and, here also, *blocks the path between a and b*, making them conditionally independent : $a \not\perp\!\!\!\perp b \,|\, \emptyset$, $a \perp\!\!\!\perp b|c$



$$\begin{aligned} p(a,b,c) &= p(a)p(c|a)p(b|c) \\ p(a,b) &\neq p(a)p(b) \implies a \not\perp\!\!\!\perp b|\emptyset \\ p(a,b|c) &= \frac{p(a)p(c|a)}{p(c)}p(b|c) = p(a|c)p(b|c) \implies a \perp\!\!\!\perp b \,|\, c \end{aligned}$$

Example 3 : c is said *head-to-head* with a and b. In this head-to-head configuration, contrary to the two examples above, *the path between a and b is blocked when c is unobserved* : $a \perp\!\!\!\perp b \,|\, \emptyset$, $a \not\perp\!\!\!\perp b \,|\, c$



$$\begin{aligned} p(a,b,c) &= p(a)p(b)p(c|a,b) \\ p(a,b) &= \sum_c p(a)p(b)p(c|a,b) = p(a)p(b) \\ &\implies a \perp\!\!\!\perp b \,|\, \emptyset \\ p(a,b|c) &\neq p(a|c)p(b|c) \\ &\implies a \not\perp\!\!\!\perp b \,|\, c \end{aligned}$$

We can extend the notion to full sets of nodes.

D-Separation

Let $\mathcal{G}$ be a DAG.
Let A, B, C three disjoint sets of nodes in $\mathcal{G}$: $A \cap B = A \cap C = B \cap C = \emptyset$.
C is the set of ”conditioning nodes”, or ”observed variables”.
We aim to determine whether $A \perp\!\!\!\perp B | C$.

Algorithm

1. **Evaluate each path between A and B**
   Evaluate each possible path between any point $a \in A$, and any point $b \in B$. Such a path between a and b is said **blocked** if it contains one node n such that one of two following conditions is true:
   - arrows in the path are *head-to-tail* or *tail-to-tail* at node n, and $n \in C$ (n is an observed/conditioning node).
   - arrows in the path are *head-to-head* at node n, and $n \notin C$ and none of n descendants is in C
2. **Assess all paths**
   - If all paths $(a, b), a \in A, b \in B$ are blocked, then A is said **D-separated** from B by C, and the joint distribution defined by $\mathcal{G}$ verifies $A \perp\!\!\!\perp B | C$.
   - If there is at least one path $(a, b), a \in A, b \in B$ that is not blocked then $A \not\perp\!\!\!\perp B | C$.

5 Dynamical Variational Auto Encoders

VAE models are well known and documented (see for example the seminal paper [11]. (A self-contained brief summary of VAE can be found in appendix A).

When dealing with sequential data, the i.i.d assumption on latent variables z_i is a limitation. By D-separation, all x_i's are independent of each other conditionally by z_i : $p(x_i|x_1, x_2, ..., x_{i-1}, x_{i+1}, .., x_n, z_i) = p(x_i|z_i)$. Therefore, a vanilla VAE can not account for correlations between x_i across time.

DVAEs encode a temporal dependency in the latent variables prior distribution. In this chapter, we review the general discrete-time setting, where the latent variables are countable and indexed by time. An exhaustive review of discrete-time DVAEs can be found in [14].

We start by some notations.

Notations

- the data is a sequence of T points noted $x_{1:T} = \{(x_t)_{t=1,...,T}\} \in \mathbb{R}^F$.
- the sequence of the associated T latent variables is $z_{1:T} = \{(z_t)_{t=1,...,T}\} \in \mathbb{R}^L$
- optionally, there may be a sequence of -usually deterministic- T inputs $u_{1:T} = \{(u_t)_{t=1,...,T}\} \in \mathbb{R}^U$

The generative model is given by the general expression of the joint distribution (here with a sequence of inputs) $p(x_{1:T}, z_{1:T}|u_{1:T})$:

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

where the only assumption that is made is a a causal dependency of the x_t, z_t on the inputs $u_{1:t}$, thus allowing to change the conditioning $|u_{1:T}$ into $|u_{1:t}$

In the rest of the report, we will consider systems with no input, and drop the conditioning on $u_{1:t}$ to simplify notations. However, the reasoning remains the same with inputs.

The true posterior $p(z_{1:T}|x_{1:T})$ is usually untractable, but can be developed:

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

It can be noted that the true posterior exhibits a dependence of z_t on *past* $z_{1:t-1}$, but a dependence on the *whole* data sequence $x_{1:T}$ (think Kalman smoother).

As in vanilla VAEs, the inference model is the approximation of the true posterior by an parametric encoder $q_\phi(z_{1:T}|x_{1:T})$, where ϕ is the set of parameters:

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

Depending on the chosen graphical models and the corresponding D-separation results, the observation model $p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}, u_{1:t})$ (with θ_x the set of parameters of the observation model) and approximate posterior $q_\phi(z_t|z_{1:t-1}, x_{1:T})$ will simplify.

It is also considered a good practice to copy the expression of $q_\phi(z_t|z_{1:t-1}, x_{1:T})$ from the expression of the true posterior resulting from the D-separation analysis (see next chapters for examples).

Equipped with the generative model and the inference model, we compute the log likelihood of the data $x_{1:T}$ and derive an Variational Lower Bound (VLB) for training (using the same manipulation as for vanilla VAE : multiplying both sides of the equation by q_ϕ and integrating over $dz_{1:T}$)

$$\begin{aligned}
\log p(x_{1:T}) &= \log \frac{p(x_{1:T}, z_{1:T})}{p(z_{1:T}|x_{1:T})} && (5.1)\\
&= \mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_\phi(z_{1:T}|x_{1:T})} \frac{q_\phi(z_{1:T}|x_{1:T})}{p(z_{1:T}|x_{1:T})} && (5.2)\\
&= \mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_\phi(z_{1:T}|x_{1:T})} + \mathbb{KL}\left(q_\phi(z_{1:T}|x_{1:T}) \| p(z_{1:T}|x_{1:T})\right) && (5.3)\\
&\geq \mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})} \log \frac{p(x_{1:T}, z_{1:T})}{q_\phi(z_{1:T}|x_{1:T})} = \mathcal{L}(\theta, \phi, X) && (5.4)
\end{aligned}$$

The dependence of $\mathcal{L}(\theta, \phi, X)$ on θ is made more obvious when developing $\mathcal{L}(\theta, \phi, X)$.

Remember we have (making the set of parameters explicit) :

$$\begin{aligned}
p_\theta(x_{1:T}, z_{1:T}) &= \prod_{t=1}^{T} p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1}) && (5.5)\\
q_\phi(z_{1:T}|x_{1:T}) &= \prod_{t=1}^{T} q_\phi(z_t|z_{1:t-1}, x_{1:T}) && (5.6)
\end{aligned}$$

Therefore

$$\mathcal{L}(\theta, \phi, X) = \mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})} \log\left(\frac{\prod_{t=1}^{T} p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})}{\prod_{t=1}^{T} q_\phi(z_t|z_{1:t-1}, x_{1:T})}\right) \tag{5.7}$$

$$= \mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})}\left(\sum_{t=1}^{T} \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) - \sum_{t=1}^{T} \log \frac{q_\phi(z_t|z_{1:t-1}, x_{1:T})}{p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})}\right) \tag{5.8}$$

At this point, the expectations require some work. First, we note that, as q_ϕ develops as 5.6, for any function Ψ, the first expectation can be written (note the change in indexes of z)

$$\mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})}\Psi(z_{1:t}) = \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})}\Psi(z_{1:t})$$

Second, we develop further and write:

$$\begin{aligned}\mathbb{E}_{q_\phi(z_{1:T}|x_{1:T})}\Psi(z_{1:t}) &= \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})}\Psi(z_{1:t}) \\ &= \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})}\mathbb{E}_{q_\phi(z_t|z_{1:t-1}, x_{1:T})}\Psi(z_{1:t})\end{aligned}$$

Therefore the VLB becomes:

$$\mathcal{L}(\theta, \phi, X) = \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})} \sum_{t=1}^{T} \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) - \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})}\left[\mathbb{E}_{q_\phi(z_t|z_{1:t-1}, x_{1:T})} \log \frac{q_\phi(z_t|z_{1:t-1}, x_{1:T})}{p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})}\right] \tag{5.9}$$

$$= \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})} \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) - \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})}\mathbb{KL}\left(q_\phi(z_t|z_{1:t-1}, x_{1:T})||p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})\right) \tag{5.10}$$

As for the vanilla VAE, the VLB contains two terms.

- The first term is the reconstruction error. it is the sum over the time steps, of the average log likelihood the data at time t, given the approximate distribution of the past and present latent variables, and the past data.

- The second term is a regularization term, summing over the time steps the average divergence between the approximate posterior distribution of the latent variable at time t, and its real distribution.

As in vanilla VAE, the sampling over q_ϕ requires the use of the ”re parametrization trick” (see [11]), for $\mathcal{L}(\theta, \phi, X)$ to be differentiable w.r.t. θ, ϕ.

Here is the summary regarding DVAE:

General Dynamical VAEs

- **generative model**

$$p(x_{1:T}, z_{1:T}) = \prod_{t=1}^{T} p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1}) \tag{5.11}$$

- **inference model**

$$q_\phi(z_{1:T}|x_{1:T}) = \prod_{t=1}^{T} q_\phi(z_t|z_{1:t-1}, x_{1:T}) \tag{5.12}$$

- **VLB for training**

$$\begin{aligned}\mathcal{L}(\theta, \phi, X) = &\sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})} \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) \\ &- \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})} \mathbb{KL}\Big(q_\phi(z_t|z_{1:t-1}, x_{1:T}) || p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})\Big)\end{aligned} \tag{5.13}$$

6 Deep Kalman Filter

The Kalman Filter is a well known model, widely used to denoise time series observations and make predictions. The latent variables form a Markov Chain, and all the probability distributions (ie encoder, decoder and transition model) are linear Gaussians. This allows to derive close form expressions for the solutions (Kalman filter and Kalman smoother).

In a **Deep Kalman Filter**, the temporal structure of the latent variables is still a Markov Chain. The probaility models are still Gaussians, but with parameters mean and covariance learnt by neural networks.

More specifically, the DAG describing a Deep Kalman Filter is:

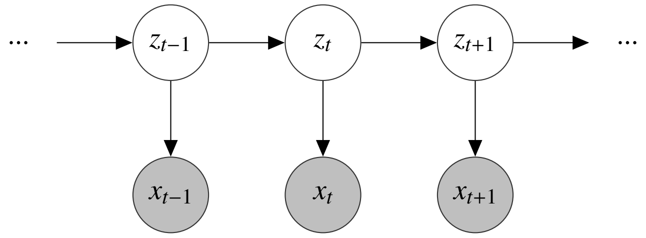


Figure 6.1: Probabilistic model of a Deep Kalman Filter

It is then particularly useful to use D-separation on the DAG to simplify the general DVAE expressions 5.11 and 5.12. Conditioning on z_t and z_{t-1} drives:

$$p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) = p_{\theta_x}(x_t|z_t) \tag{6.1}$$

$$p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t}) = p_{\theta_z}(z_t|z_{t-1}) \tag{6.2}$$

$$q_\phi(z_t|z_{1:t-1}, x_{1:T}) = q_\phi(z_t|z_{t-1}, x_{t:T}) \tag{6.3}$$

We then choose Gaussian distributions for $p_{\theta_x}, p_{\theta_z}$ and q_ϕ, with mean and diagonal covariance, learnt by neural networks.

$$p_{\theta_x}(x_t|z_t) = \mathcal{N}(x_t|\mu_{\theta_x}(z_t), \operatorname{diag}\sigma^2_{\theta_x}(z_t)) \tag{6.4}$$

$$p_{\theta_z}(z_t|z_{t-1}) = \mathcal{N}(z_t|\mu_{\theta_z}(z_{t-1}), \operatorname{diag}\sigma^2_{\theta_z}(z_{t-1})) \tag{6.5}$$

$$q_\phi(z_t|z_{t-1}, x_{t:T}) = \mathcal{N}(z_t|\mu_\phi(z_{t-1}, x_{t:T}), \operatorname{diag}\sigma^2_{\theta_z}(z_{t-1}, x_{t:T})) \tag{6.6}$$

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

$$
\begin{gathered}
q_\phi(z_t|z_{t-1}, x_t) \\
q_\phi(z_t|z_{1:t}, x_{1:t}) \\
q_\phi(z_t|z_{1:T}, x_{1:T})
\end{gathered}
$$

We have chosen 6.3 for the implementation, as it has the same formulation as the true posterior and respects the corresponding dependencies.

Taking note that:

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

And using D-Separation, the Evidence Lower Bound (ELBO) 5.13 simplifies into:

$$
\begin{aligned}
\mathcal{L}(\theta, \phi, X) &= \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})} \log p_{\theta_x}(x_t|z_t) - \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})} \mathbb{KL}\left(q_\phi(z_t|z_{t-1}, x_{t:T}) || p_{\theta_z}(z_t|z_{t-1})\right) && (6.7) \\
&= \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_t|x_{1:T})} \log p_{\theta_x}(x_t|z_t) - \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{t-1}|x_{1:T})} \mathbb{KL}\left(q_\phi(z_t|z_{t-1}, x_{t:T}) || p_{\theta_z}(z_t|z_{t-1})\right) && (6.8)
\end{aligned}
$$

As a summary:

Deep Kalman Filter

- **generative model**

$$
\begin{aligned}
p_{\theta_x}(x_t|z_t) &= \mathcal{N}(x_t|\mu_{\theta_x}(z_t), \operatorname{diag} \sigma^2_{\theta_x}(z_t)) && (6.9) \\
p_{\theta_z}(z_t|z_{t-1}) &= \mathcal{N}(z_t|\mu_{\theta_z}(z_{t-1}), \operatorname{diag} \sigma^2_{\theta_z}(z_{t-1})) && (6.10)
\end{aligned}
$$

- **inference model**

$$q_\phi(z_t|z_{t-1}, x_{t:T}) = \mathcal{N}(z_t|\mu_\phi(z_{t-1}, x_{t:T}), \operatorname{diag} \sigma^2_{\theta_z}(z_{t-1}, x_{t:T})) \qquad (6.11)$$

- **VLB for training**

$$\mathcal{L}(\theta, \phi, X) = \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_t|x_{1:T})} \log p_{\theta_x}(x_t|z_t) - \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{t-1}|x_{1:T})} \mathbb{KL}\left(q_\phi(z_t|z_{t-1}, x_{t:T}) || p_{\theta_z}(z_t|z_{t-1})\right) \qquad (6.12)$$

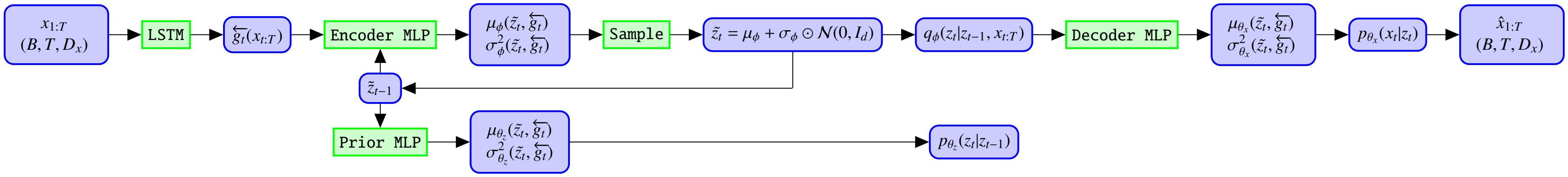
The $\mathbb{KL}\left(q_\phi || p_{\theta_z}\right)$'s have a close form, as the two distributions are Gaussians (see C)

From a code stand-point, following [14], we have used forward LSTM to encode sequences such as $x_{1:t}$, and backward LSTM to encode sequences such as $x_{t:T}$, as inputs into the MLP parametrizing the distributions.

For example:

$$
\begin{aligned}
\overleftarrow{g_t} &= \text{Backward LSTM}(\overleftarrow{g_{t+1}}, x_t) \text{ (encodes } x_{t:T}) \\
q_\phi(z_t|z_{t-1}, x_{t:T}) &= \mathcal{N}(z_t|\mu_\phi(z_{t-1}, \overleftarrow{g_t}), \operatorname{diag} \sigma^2_\phi(z_{t-1}, \overleftarrow{g_t}))
\end{aligned}
$$

The PyTorch implementation is described below:


$x_{1:T}$
(B, T, D_x)
LSTM
$\overleftarrow{g}_t(x_{t:T})$
Encoder MLP
$\mu_\phi(\tilde{z}_t, \overleftarrow{g}_t)$
$\sigma^2_\phi(\tilde{z}_t, \overleftarrow{g}_t)$
Sample
$\tilde{z}_t = \mu_\phi + \sigma_\phi \odot \mathcal{N}(0, I_d)$
$q_\phi(z_t|z_{t-1}, x_{t:T})$
Decoder MLP
$\mu_{\theta_x}(\tilde{z}_t, \overleftarrow{g}_t)$
$\sigma^2_{\theta_x}(\tilde{z}_t, \overleftarrow{g}_t)$
$p_{\theta_x}(x_t|z_t)$
$\hat{x}_{1:T}$
(B, T, D_x)
$\tilde{z}_{t-1}$
Prior MLP
$\mu_{\theta_z}(\tilde{z}_t, \overleftarrow{g}_t)$
$\sigma^2_{\theta_z}(\tilde{z}_t, \overleftarrow{g}_t)$
$p_{\theta_z}(z_t|z_{t-1})$

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

Figure 6.2: Deep Kalman Filter Model Architecture

7

Variational Recurrent Neural Network

The VRNN is the most expressive DVAE, in that sense that the general expressions 5.11, 5.12 and VLB 5.13 can not be simplified.

The GPM of the VRNN assumes full connections between latent variables, and between observed variables, to account for the full unsimplified expressions. Specifically:

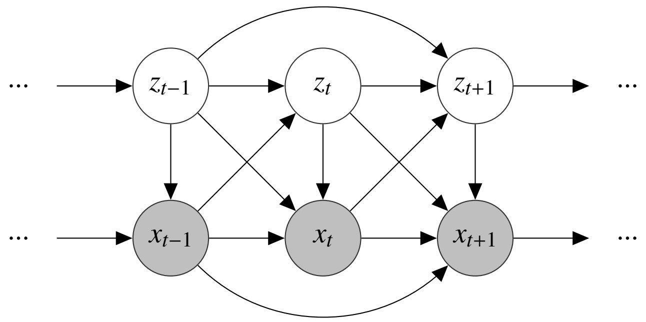


Figure 7.1: Probabilistic model of a Variational RNN

We remember that:

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

And posit Gaussian distributions with diagonal covariance and mean given by two networks:

$$p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) = \mathcal{N}(x_t|\mu_{\theta_x}(x_{1:t-1}, z_{1:t}), \operatorname{diag} \sigma^2_{\theta_x}(x_{1:t-1}, z_{1:t})) \tag{7.1}$$

$$p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1}) = \mathcal{N}(z_t|\mu_{\theta_z}(z_{1:t-1}, x_{1:t-1}), \operatorname{diag} \sigma^2_{\theta_z}(z_{1:t-1}, x_{1:t-1})) \tag{7.2}$$

The true posterior being

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

we choose the encoder with the same conditional dependencies and a Gaussian expression:

$$q_\phi(z_t|z_{1:t-1}, x_{1:T}) = \mathcal{N}(z_t|\mu_\phi(z_{1:t-1}, x_{1:T}), \operatorname{diag}\sigma^2_\phi(z_{1:t-1}, x_{1:T}))$$

The VLB is:

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

As a summary

Variational RNN

- **generative model**

$$p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) = \mathcal{N}(x_t|\mu_{\theta_x}(x_{1:t-1}, z_{1:t}), \operatorname{diag}\sigma^2_{\theta_x}(x_{1:t-1}, z_{1:t})) \tag{7.3}$$

$$p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1}) = \mathcal{N}(z_t|\mu_{\theta_z}(z_{1:t-1}, x_{1:t-1}), \operatorname{diag}\sigma^2_{\theta_z}(z_{1:t-1}, x_{1:t-1})) \tag{7.4}$$

- **inference model**

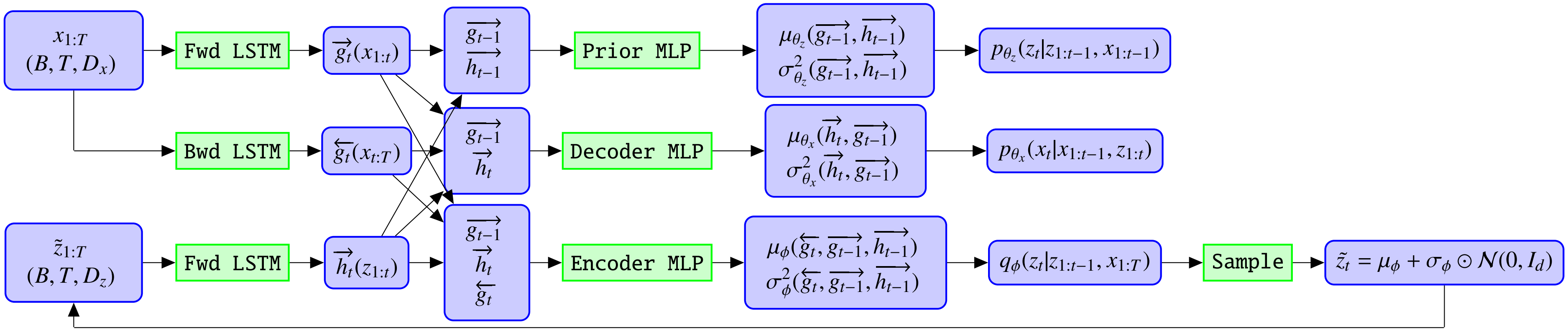
$$q_\phi(z_t|z_{1:t-1}, x_{1:T}) = \mathcal{N}(z_t|\mu_\phi(z_{1:t-1}, x_{1:T}), \operatorname{diag}\sigma^2_\phi(z_{1:t-1}, x_{1:T})) \tag{7.5}$$

- **VLB for training**

$$\begin{aligned}\mathcal{L}(\theta, \phi, X) = &\sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t}|x_{1:T})} \log p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t}) \\ &- \sum_{t=1}^{T} \mathbb{E}_{q_\phi(z_{1:t-1}|x_{1:T})} \mathbb{KL}\left(q_\phi(z_t|z_{1:t-1}, x_{1:T}) || p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})\right)\end{aligned} \tag{7.6}$$

We have chosen a different implementation from [14] and used three different LSTM networks to encode $z_{1:t}$, $x_{1:t-1}$ and $x_{t:T}$ respectively.

The PyTorch implementation is described below:


$x_{1:T}$
(B, T, D_x)
Fwd LSTM
Bwd LSTM
$\overrightarrow{g_t}(x_{1:t})$
$\overleftarrow{g_t}(x_{t:T})$
$\overrightarrow{g_{t-1}}$
$\overrightarrow{h_{t-1}}$
Prior MLP
$\mu_{\theta_z}(\overrightarrow{g_{t-1}}, \overrightarrow{h_{t-1}})$
$\sigma^2_{\theta_z}(\overrightarrow{g_{t-1}}, \overrightarrow{h_{t-1}})$
$p_{\theta_z}(z_t|z_{1:t-1}, x_{1:t-1})$
$\overrightarrow{g_{t-1}}$
$\overrightarrow{h_t}$
Decoder MLP
$\mu_{\theta_x}(\overrightarrow{h_t}, \overrightarrow{g_{t-1}})$
$\sigma^2_{\theta_x}(\overrightarrow{h_t}, \overrightarrow{g_{t-1}})$
$p_{\theta_x}(x_t|x_{1:t-1}, z_{1:t})$
$\tilde{z}_{1:T}$
(B, T, D_z)
Fwd LSTM
$\overrightarrow{h_t}(z_{1:t})$
$\overrightarrow{g_{t-1}}$
$\overrightarrow{h_t}$
$\overleftarrow{g_t}$
Encoder MLP
$\mu_\phi(\overleftarrow{g_t}, \overrightarrow{g_{t-1}}, \overrightarrow{h_{t-1}})$
$\sigma^2_\phi(\overleftarrow{g_t}, \overrightarrow{g_{t-1}}, \overrightarrow{h_{t-1}})$
$q_\phi(z_t|z_{1:t-1}, x_{1:T})$
Sample
$\tilde{z}_t = \mu_\phi + \sigma_\phi \odot \mathcal{N}(0, I_d)$

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

Figure 7.2: Variational RNN Model Architecture

8 Gaussian Process Variational Auto Encoder

DVAEs are a natural and straightforward extension of VAEs to the time domain. However, the discretization of time comes with limitations. First, one has to choose a relevant time interval to sample the data, which can prove arbitrary if the observed process is not well known. Second, that time interval is fixed for training and inference, can not be changed depending on the time dynamics of the observed process, and can not account for different time scales.

It is therefore interesting to allow a continuous-time formulation of the prior of the latent variables, to provide additional flexibility and expressiveness. The natural and straightforward framework for such a continuous time prior is the GP, that constitutes the core of GP-VAEs. (A summary of GP can be found in B).

If the use of GPs for time-series modeling is not recent (see for example [6] and [7]), structuring a latent prior as a GP is somewhat newer. In [9], Casale and al. build a GP-VAE generative model to predict images with different objects and views. A specific kernel is designed for the task, taking advantage of the kernel construction rules and multiplying a view-based kernel by an object-based kernel. The kernel parameters are learnt with the inference model, and the covariance matrix of the kernel is built with a low-rank approximation (VV^T) to reduce computation costs (naïvely in $O(T^3)$). In [13], Fortuin and al. focus on time-series missing values imputations. A Cauchy kernel is used, which is an instance of a Rational Quadratic Kernel, that can be viewed as an infinite sum of Gaussian kernels over the space of lengthscales. The encoder q_ϕ is a multivariate normal distribution, whose precision matrix is built mutliplying a bi-band matrix and its transpose, again to reduce computation cost. [14] cites GP-VAE but remains focused on discrete-time models. The paper [16] establishes the Markovian nature of a GP as the solution of a linear SDE, which allows to significantly reduce the computation cost of the model.

The main insight is that the solution of a linear SDE is a Gaussian Process, as the transition probabilities given by the Fokker-Plank equations are Gaussian. Thus, GP-VAEs have the potential to express naturally many phenomena described by SDEs. We will review later those results, issued form stochastic calculus reference books such as [5], [12] and [18].

We now move to the GP-VAE model itself.

We can consider **data taken at irregular time intervals**. We change our notation accordingly, and note $(t_1, ..., t_{i-1}, t_i, t_{i+1}, ..., t_T)$ the T times (or timestamps) considered.

The GPM of the GP-VAE is:

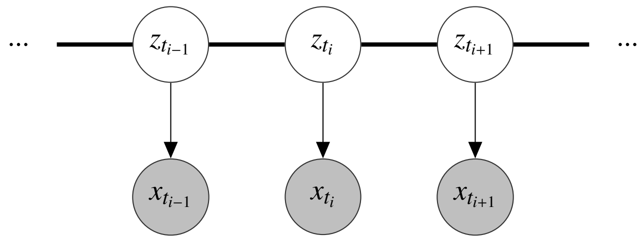


Figure 8.1: Probabilistic model of a GP-VAE

Where the thick black lines between latent variables define a fully connected graph : all latent variables are -a priori- correlated between each other in a Gaussian Process. (NB : this GPM is not, per say, a DAG in this regard. However, D-separation still applies for observed variables x_{t_i}).

The joint distribution writes somehow differently from the one for DVAEs, as:

$$\begin{aligned} p(x_{t_1:t_T}, z_{t_1:t_T}) &= p(z_{t_1:t_T}) p(x_{t_1:t_T} | z_{t_1:t_T}) && (8.1) \\ &= p(z_{t_1:t_T}) \prod_{i=1}^{T} p(x_{t_i} | x_{t_1:t_{i-1}}, z_{t_1:t_T}) && (8.2) \\ &= p(z_{t_1:t_T}) \prod_{i=1}^{T} p(x_{t_i} | z_{t_i}) && (8.3) \end{aligned}$$

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 $l \in \{1, ..., L\}$ of the latent variables. Formally:

$$p_{\theta_z}(z^l_{t_1:t_T}) = \mathcal{GP}(m_{\theta_z,l}(t_1 : t_T), k_{\theta_z,l}(t_1 : t_T, t_1 : t_T)) \qquad l = 1, .., L \tag{8.4}$$

where the $m_{\theta_z,l}$ are the L mean functions of the GP priors (usually chosen constant null), and the $k_{\theta_z,l}$ are the kernel functions of the GP priors.

We note at this point that:

- by design, each of the component of the z_{t_i} is a scalar GP, with correlation over time stamps. However, the different components of a z_{t_i} are not correlated between them. The correlation across dimensions is encoded into the observation model $p_{\theta_x}(x_{t_i} | z_{t_i})$.

- the kernels $k_{\theta_z,l}$ can be chosen differently to account for different prior knowledge of the data sequence. In [13] for example, Fortuin and al. uses a set of Gaussian Kernels with different lenghtscales.

Accordingly, the approximate posterior -encoder- q_ϕ is a set of L Gaussian distributions of dimension T, each one accounting for a component of z_{t_i}. Formally :

$$\begin{aligned} q_\phi(z^l_{t_1:t_T} | x^l_{t_1:t_T}) &= \mathcal{N}(m^l_\phi(x_{t_1:t_T}), \Sigma^l_\phi(x_{t_1:t_T})) \qquad l = 1, .., L && (8.5) \\ &= \mathcal{N}(m^l_\phi(x_{t_1:t_T}), \Lambda^l_\phi(x_{t_1:t_T})^{-1}) && (8.6) \\ &= \mathcal{N}(m^l_\phi(x_{t_1:t_T}), L^l_\phi(x_{t_1:t_T}) L^l_\phi(x_{t_1:t_T})^T) && (8.7) \end{aligned}$$

where we have made explicit the different ways of defining the multivariate normal distribution, with its covariance matrix Σ^l_ϕ, its precision matrix Λ^l_ϕ, or with a Cholesky decomposition $L^l_\phi L^l_\phi$.

The observation model, by D-separation, is:

$$p(x_{t_1:t_T}|z_{t_1:t_T}) = \prod_{i=1}^{T} p_{\theta_x}(x_{t_i}|z_{t_i}) \tag{8.8}$$

The log-likelihood of the data writes:

$$\log p(x_{t_1:t_T}) = \log \frac{p(x_{t_1:t_T}, z_{t_1:t_T})}{p(z_{t_1:t_T}|x_{t_1:t_T})} \tag{8.9}$$

As usual, we multiply by $q_\phi(z_{t_1:t_T}|x_{t_1:t_T})$ and integrate over $dz_{t_1:t_T}$ to form the VLB:

$$\begin{aligned}
\log p(x_{t_1:t_T}) &= \int q_\phi(z_{t_1:t_T}|x_{t_1:t_T}) \log \frac{p(x_{t_1:t_T}, z_{t_1:t_T})}{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} \frac{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})}{p(z_{t_1:t_T}|x_{t_1:t_T})} dz_{t_1:t_T} && (8.10)\\
&= \mathbb{E}_{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} \log \frac{p(x_{t_1:t_T}, z_{t_1:t_T})}{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} + \mathbb{KL}\left(q_\phi(z_{t_1:t_T}|x_{t_1:t_T})||p(z_{t_1:t_T}|x_{t_1:t_T})\right) && (8.11)\\
&\geq \mathbb{E}_{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} \log \frac{p(x_{t_1:t_T}, z_{t_1:t_T})}{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} = \mathcal{L}(\theta, \phi, X) && (8.12)
\end{aligned}$$

Factoring in 8.1 and 8.8, we get:

$$\begin{aligned}
\mathcal{L}(\theta, \phi, X) &= \mathbb{E}_{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} \log \left[\left(\prod_{i=1}^{T} p_{\theta_x}(x_{t_i}|z_{t_i})\right) \frac{p_{\theta_z}(z_{t_1:t_T})}{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})}\right] && (8.13)\\
&= \sum_{i=1}^{T} \mathbb{E}_{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} \log p_{\theta_x}(x_{t_i}|z_{t_i}) - \mathbb{KL}\left(q_\phi(z_{t_1:t_T}|x_{t_1:t_T})||p_{\theta_z}(z_{t_1:t_T})\right) && (8.14)
\end{aligned}$$

We have $\mathbb{E}_{q_\phi(z_{t_1:t_T}|x_{t_1:t_T})} f(z_{t_i}) = \mathbb{E}_{q_\phi(z_{t_i}|x_{t_1:t_T})} f(z_{t_i})$ for any f, so we get finally:

$$\mathcal{L}(\theta, \phi, X) = \sum_{i=1}^{T} \mathbb{E}_{q_\phi(z_{t_i}|x_{t_1:t_T})} \log p_{\theta_x}(x_{t_i}|z_{t_i}) - \mathbb{KL}\left(q_\phi(z_{t_1:t_T}|x_{t_1:t_T})||p_{\theta_z}(z_{t_1:t_T})\right) \tag{8.15}$$

We note that:

- the $\mathbb{KL}$-divergence is actually the sum of the L $\mathbb{KL}$-divergences $\mathbb{KL}\left(q_\phi^l||p_{\theta_z}^l\right)$, which have a close form solution as both distributions are Gaussian. (see the well-known result C)

- the reconstruction 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 $t_1, ...t_T$. If the kernel parameters are fixed -such as in [13]- 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 [16]), then the computation must occur at each training iteration.

As a summary:

Gaussian Process VAEs

- **generative model**

$$p(x_{t_1:t_T}, z_{t_1:t_T}) = p(z_{t_1:t_T}) \prod_{i=1}^{T} p(x_{t_i}|z_{t_i}) \tag{8.16}$$

$$p_{\theta_z}(z^l_{t_1:t_T}) = \mathcal{GP}(m_{\theta_z,l}(t_1 : t_T), k_{\theta_z,l}(t_1 : t_T)) \qquad l = 1, .., L \tag{8.17}$$

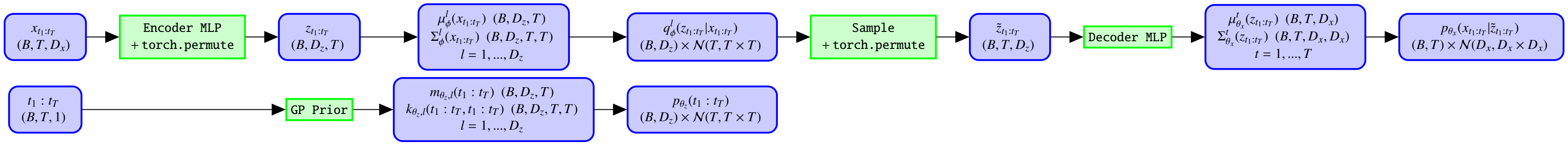
- **inference model**

$$\begin{aligned} q_\phi(z^l_{t_1:t_T}|x^l_{t_1:t_T}) &= \mathcal{N}(m^l_\phi(x_{t_1:t_T}), \Sigma^l_\phi(x_{t_1:t_T})) \qquad l = 1, .., L && (8.18)\\ &= \mathcal{N}(m^l_\phi(x_{t_1:t_T}), \Lambda^l_\phi(x_{t_1:t_T})^{-1}) && (8.19)\\ &= \mathcal{N}(m^l_\phi(x_{t_1:t_T}), L^l_\phi(x_{t_1:t_T}) L^l_\phi(x_{t_1:t_T})^T) && (8.20) \end{aligned}$$

- **VLB for training**

$$\mathcal{L}(\theta, \phi, X) = \sum_{i=1}^{T} \mathbb{E}_{q_\phi(z_{t_i}|x_{t_1:t_T})} \log p_{\theta_x}(x_{t_i}|z_{t_i}) - \mathbb{KL}\left(q_\phi(z_{t_1:t_T}|x_{t_1:t_T}) \| p_{\theta_z}(z_{t_1:t_T})\right) \tag{8.21}$$

The PyTorch implementation is described here:


$x_{t_1:t_T}$ (B, T, D_x)
Encoder MLP + torch.permute
$z_{t_1:t_T}$ (B, D_z, T)
$\mu^l_\phi(x_{t_1:t_T})$ (B, D_z, T)
$\Sigma^l_\phi(x_{t_1:t_T})$ (B, D_z, T, T)
$l = 1, ..., D_z$
$q^l_\phi(z_{t_1:t_T}|x_{t_1:t_T})$ $(B, D_z) \times \mathcal{N}(T, T \times T)$
Sample + torch.permute
$\tilde{z}_{t_1:t_T}$ (B, T, D_z)
Decoder MLP
$\mu^t_{\theta_x}(z_{t_1:t_T})$ (B, T, D_x)
$\Sigma^t_{\theta_x}(z_{t_1:t_T})$ (B, T, D_x, D_x)
$t = 1, ..., T$
$p_{\theta_x}(x_{t_1:t_T}|\tilde{z}_{t_1:t_T})$ $(B, T) \times \mathcal{N}(D_x, D_x \times D_x)$
$t_1 : t_T$ $(B, T, 1)$
GP Prior
$m_{\theta_z,l}(t_1 : t_T)$ (B, D_z, T)
$k_{\theta_z,l}(t_1 : t_T, t_1 : t_T)$ (B, D_z, T, T)
$l = 1, ..., D_z$
$p_{\theta_z}(t_1 : t_T)$ $(B, D_z) \times \mathcal{N}(T, T \times T)$

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

Figure 8.2: Gaussian Process VAE Model Architecture

Part III

Stochastic Processes and stochastic calculus

9 Entropy and Randomness

In its most general form, a data sequence can be considered as a realization of a stochastic process. A stochastic process is usually described as either a sequence of random variables (ie a set of $(X_t)_{t\geq 0} : \omega \rightarrow X_t(\omega)$), or a single random variable over the space of functions (ie $X : \omega \rightarrow X(\omega) = X_\omega = \{t \rightarrow X_\omega(t)\}$.

As we wish to measure -somehow- the degree of ”randomness” of a data sequence, we will find more convenient to use the former view (ie consider a data sequence as a countable sequence of random variables), as it allows to use the framework of information theory.

First, we will recall the basic definitions of information theory : entropy, relative entropy (ie KL divergence), conditional entropy and mutual information. Then we will write some results regarding the application of Information Theory (IT) to data sequence. Last, we will describe two of the most used empirical measurements : Approximate Entropy (ApEn) and Sample Entropy (SampEn).

The basic of IT are introduced in [8], [17], and [15]. One of the reference books on the subject is [4], and goes much further than the scope of this report. Of course, the interested reader will also refer to the seminal paper by Shannon : [1].

Entropy : given a random variable X, either discrete or continuous, taking values in a measurable space $\mathcal{X}$, and its probability distribution p, the amount of information given by a given realization x is given by $\log \frac{1}{p(x)} = -\log p(x)$ (the lower the probability, the higher the amount of information).

The average amount of information (over all possible values of x) required to describe the random variable X is the entropy of X :

$$\mathcal{H}(X) = -\sum_{x\in\mathcal{X}} p(x) \log p(x) \tag{9.1}$$

(or $-\int_{\mathcal{X}} p(x) \log p(x) dx$).

Relative entropy, KL divergence : when approximating the true data distribution p by a distribution q, we require in average a quantity of information $-\sum_{x\in\mathcal{X}} p(x) \log q(x)$ to describe X. The difference between the optimal amount of information (ie the entropy $\mathcal{H}(X)$) and this quantity is the well-known relative entropy, of KL-divergence between

p and q :

$$\mathbb{KL}(p||q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} \tag{9.2}$$

$$= \int_{\mathcal{X}} p(x) \log \frac{p(x)}{q(x)} dx \tag{9.3}$$

The properties of KL-divergence (positiveness, non symmetry) are well described in the sources above.

Conditional Entropy : we now consider two random variables X and Y, and wish to measure the degree of relationship between them. We define the conditional entropy of, for example, Y given X, as the amount of information we get observing the values of Y given X, averaged over the joint probability. Formally :

$$\mathcal{H}(Y|X) = - \sum_{x,y \in \mathcal{X},\mathcal{Y}} p(x,y) \log p(y|x)$$

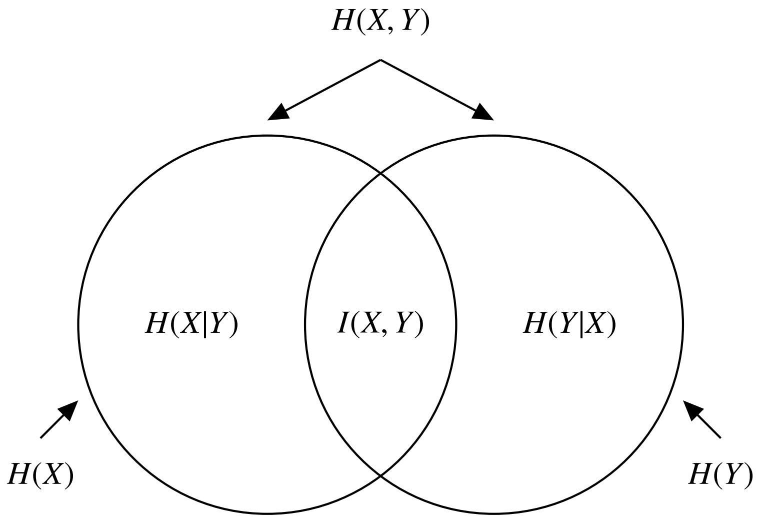
$$= - \int_{\mathcal{X},\mathcal{Y}} p(x,y) \log p(y|x)\, dxdy$$

By basic calculation, we get $\mathcal{H}(X,Y) = \mathcal{H}(Y|X) + \mathcal{H}(X) = \mathcal{H}(X|Y) + \mathcal{H}(Y)$

Mutual Information - last, still considering two random variables X, Y, the mutual information is the additional amount of information we need to describe X, Y when we assume independence (ie use $p(x)p(y)$) rather than use the true joint probability $p(x,y)$. This amount is $-\log p(x)p(y) - (-\log p(x,y)) = \log \frac{p(x,y)}{p(x)p(y)}$, that we average over the true distribution $p(x,y)$:

$$\mathcal{I}(X,Y) = \sum_{x,y \in \mathcal{X},\mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{9.4}$$

The relationships between entropy, relative entropy, conditional entropy and mutual information, are well described using a Venn diagram:



For example : $\mathcal{I}(X,Y) = \mathcal{H}(X) + \mathcal{H}(Y) - \mathcal{H}(X,Y)$, etc.

If we now consider a sequence of n random variables, a way to describe the randomness of the sequence is to measure how the entropy of the joint distribution grow with n. (see [4] p63). Typically, if the random variables are independent, we can expect the entropy to grow at each step n by the full amount of $\mathcal{H}(X_n)$. On the other hand, if correlations exist, then we can expect the overall entropy to grow by a lesser amount.

Formally, the **entropy rate** of a stochastic process $(X_n)_{n\in\mathbb{N}^*}$ is defined by:

$$\mathcal{H}(X) = \lim_{n\to\infty} \frac{1}{n} \mathcal{H}(X_1, X_2, ..., X_n) \tag{9.5}$$

when the limit exists. This is the entropy per symbol.

[4] also defines a **conditional entropy rate**

$$\mathcal{H}'(X) = \lim_{n\to\infty} \frac{1}{n} \mathcal{H}(X_n|X_{n-1}, ...X_2, X_1) \tag{9.6}$$

when the limit exists. This is the entropy of the latest random variable, given the past realizations.

An important result is that, when (X_n) is stationary, both limits exist and are equal.

The question naturally arises that, given a data sequence, how do we compute an approximation of 9.5 or 9.6. We now introduce the ApEn and SampEn (see [10] and [2]).

Approximate Entropy - Approximate Entropy is a measure of the log probability that patterns, that were identified in the data through the examination of sub sequences of a given length, still remain when considering longer subsequences.

Formally, let $x = (x_1, x_2, ..., x_N)$ be a data sequence of length N, m an integer $(0 < m \leq N)$, and $r > 0$ a measure of acceptable noise. We define as *blocks* of length m the subsequences $b_i^m = \{x_i, x_{i+1}, ..., x_{i+m-1}\}$, starting at i with $(1 \leq i \leq N - m + 1)$. For two blocks b_i^m and b_j^m, we define a component-wise distance $d_{ij}^m = \max_{k=0,1,..,m-1} |b_{i+k}^m - b_{j+k}^m|$. We consider "close enough" (ie similar) those blocks whose distance is lower than the acceptable noise (tolerance) r, and calculate the frequency of those similar blocks (b_i^m, b_j^m) w.r.t. all blocks of length m for a given b_i^m :

$$C_i^m(r) = \frac{\text{number of } j \leq N - m + 1 \text{ s.t. } d_{ij} \leq r}{\text{number of blocks of length } m : N - m + 1} \tag{9.7}$$

We then average the $\log C_i^m$ over all possible subsequences b_i :

$$\Phi^m(r) = \frac{1}{N - m + 1} \sum_{i=1}^{N-m+1} \log C_i^m(r)$$

We can see $\Phi^m(r)$ as the average of the log probability of two subsequences of length m to be similar (up to the tolerance r). Finally, we compute:

$$\text{ApEn}(m, r, N) = \Phi^m(r) - \Phi^{m+1}(r) \tag{9.8}$$

When $m \ll N$, then $-\text{ApEn}(m, r, N) \sim \frac{1}{N+1} \sum_{i=1}^{N+1} \log \frac{C_i^{m+1}}{C_i^m}$, which is the average over i of the log of the conditional probability of two sequences b_i^{m+1}, b_j^{m+1} of length $m + 1$ be similar given that they are already similar for lengths $1, 2, ..., m$.

Last, $\text{ApEn}(m, r, N)$ is a statistical estimator of:

$$\text{ApEn}(m, r) = \lim_{N\to\infty} \text{ApEn}(m, r, N) \tag{9.9}$$

Intuitively, the greater the regularity in a data sequence, the greater the likelihood that patterns existing for subsequences of length m still remain for subsequences of greater length, ie the smaller ApEn, and conversely.

Key properties of ApEn include:

- ApEn is independent of any model of the data sequence.
- due to its construction, ApEn is non-negative, is finite for stochastic processes and deterministic processes with noise.
- following [2], it is imperative to eliminate any trend in the data sequence before computing ApEn and drawing conclusions.
- typical recommended values for m are 2 and 3. Typical recommended values for r are in the range of 0.1 to 0.25 the standard deviation of the data sequence, in order to allow a sufficient number of subsequences close within a distance r, and reasonable estimates of the conditional probabilities.
- if the noise is significant (ie signal-to-noise ratio lower than three), then precautions must be taken with the interpretation of ApEn.
- Regular measurements of the data sequence over time are required.
- Normalization of data sequences is required before computations of ApEn to compare data sequences between each other.

Sample Entropy - we can see in 9.7 that a given subsequence b_i^m is counted in both the numerator and the denominator. If this ensures numerical stability (ie no attempt to calculate $\log 0$ for example), this also introduces a bias in the calculation of the probability estimates. SampEn explicitly discounts the subsequence b_i^m from the calculations to remove the bias.

Formally:

$$A_i^m(r) = \frac{1}{N-m-1}\{\text{number of blocks } b_j^{m+1} \text{ of length } m+1 \text{ s.t. } i \neq j \text{ and } d_{ij}^{m+1} \leq r\} \tag{9.10}$$

$$B_i^m(r) = \frac{1}{N-m}\{\text{number of blocks } b_j^{m} \text{ of length } m \text{ s.t. } i \neq j \text{ and } d_{ij}^{m} \leq r\} \tag{9.11}$$

$$B^m(r) = \frac{1}{N-m}\sum_{i=1}^{N-m} B_i^m(r) \tag{9.12}$$

$$A^m(r) = \frac{1}{N-m-1}\sum_{i=1}^{N-m-1} A_i^m(r) \tag{9.13}$$

And

$$\text{SampEn}(m,r,N) = -\log\frac{A^m(r)}{B^m(r)} \tag{9.14}$$

$$\text{SampEn}(m,r) = -\lim_{n\to\infty}\log\frac{A^m(r)}{B^m(r)} \tag{9.15}$$

10

Stochastic Calculus

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

11

Stochastic processes basics

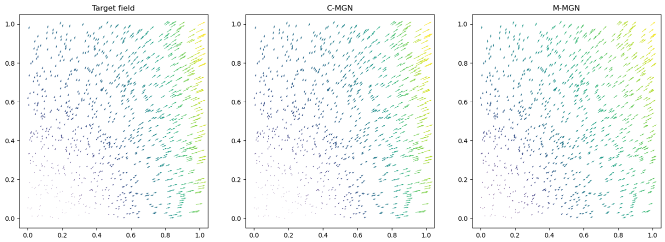


Figure 11.1: Caption

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

12

Perspective 1 : Linear SDEs and Markovian kernel GPs

Where we discuss stuff.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

13

Perspective 2 : more IT on stochastic processes

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Part IV

Experiments

14

Experiments

XPs

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Part V

Conclusion and Discussion

15

Conclusions

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Part VI

Appendices

Appendices

A Vanilla Variational Auto Encoder

We consider a sequence of i.i.d points $(x_i)_{i=1,...,N} \in \mathbb{R}^D$, and the associated latent variables $(z_i)_{i=1,...,N} \in \mathbb{R}^L$.

In the vanilla VAE setting, the observation model (decoder) is $p_{\theta_x}(x|z)$, the approximate posterior (encoder) is $q_\phi(z|x)$, the latent prior is $p_{\theta_z}(z)$.

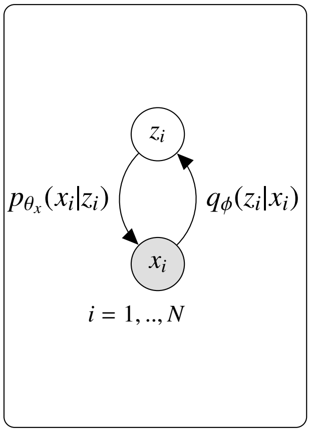


Figure A.1: Vanilla VAE

The log likelihood of the data is:

$$\log p_\theta(x) = \log \frac{p(x,z)}{p(z|x)}$$

Multiplying both sides by $q_\phi(z|x)$ and integrating over dz leads to:

$$\begin{aligned}
\log p_\theta(x) &= \int q_\phi(z|x) \log \frac{p_\theta(x,z)}{p(z|x)} dz \\
&= \int q_\phi(z|x) \log \frac{p_\theta(x,z)}{q_\phi(z|x)} \frac{q_\phi(z|x)}{p(z|x)} dz \\
&= \mathbb{E}_{q_\phi(z|x)} \log \frac{p_\theta(x,z)}{q_\phi(z|x)} + \mathbb{KL}(q_\phi(z|x) \| p(z|x)) \\
&\geq \mathbb{E}_{q_\phi(z|x)} \log \frac{p_\theta(x,z)}{q_\phi(z|x)} = \mathcal{L}(\theta, \phi, X)
\end{aligned}$$

In this setting, the D-separation is obvious and the joint distribution factorizes over n:

$$
\begin{aligned}
p_\theta(x, z) &= \prod_{i=1}^{n} p_{\theta_x}(x_i|z_i)p_{\theta_z}(z_i) \\
q_\phi(z|x) &= \prod_{i=1}^{n} q_\phi(z_i|x_i)
\end{aligned}
$$

The VLB (or ELBO) $\mathcal{L}(\theta, \phi, X)$ simplifies into:

$$
\begin{aligned}
\mathcal{L}(\theta, \phi, X) &= \mathbb{E}_{q_\phi(z|x)} \log \frac{\prod_{i=1}^{n} p_{\theta_x}(x_i|z_i)p_{\theta_z}(z_i)}{\prod_{i=1}^{n} q_\phi(z_i|x_i)} \\
&= \sum_{i=1}^{n} \mathbb{E}_{q_\phi(z_i|x_i)} p_{\theta_x}(x_i|z_i) - \sum_{i=1}^{n} \mathbb{KL}(q_\phi(z_i|x_i) \| p_{\theta_z}(z_i))
\end{aligned}
$$

The first term is the reconstruction loss, and is estimated via Monte Carlo sampling over $z_i \sim q_\phi(z_i|x_i)$. The second term is a KL-divergence, which can be computed analytically when q_ϕ and p_{θ_z} are chosen to be Gaussians.

B

Gaussian Process

We summarize here most of the results of the Gaussian Process, and refers the reader to [6] for further details.

We first recall the Gaussian marginal and conditional result:

Let x and y be jointly Gaussian vectors, ie:

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} A & C \\ C^T & B \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \tilde{A} & \tilde{C} \\ \tilde{C}^T & \tilde{B} \end{bmatrix}^{-1}\right) \tag{B.1}$$

where A, B, C is the block decomposition of the covariance matrix, and $\tilde{A}, \tilde{B}, \tilde{C}$ the block decomposition of the precision matrix.

Then the marginal distribution of x and the conditional distribution of x given y are :

$$x \sim \mathcal{N}(\mu_x, A) \tag{B.2}$$

$$x|y \sim \mathcal{N}(\mu_x + CB^{-1}(y - \mu_y), A - CB^{-1}C^T) \tag{B.3}$$

$$= \mathcal{N}(\mu_x - \tilde{A}^{-1}\tilde{C}(y - \mu_y), \tilde{A}^{-1}) \tag{B.4}$$

We now consider a Gaussian Process with mean function $m(.)$ and kernel $k(.,.)$

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{B.5}$$

At the training points $X = \{x_1, ..., x_n\}$, the observations are $Y = \{y_1, ..., y_n\}$ with some noise $y = f(x) + \epsilon$ with $\epsilon \overset{i.i.d}{\sim} \mathcal{N}(0, \sigma_n^2)$.

The covariance between observations writes:

$$\text{cov}(y_p, y_q) = k(x_p, x_q) + \delta_{pd}\sigma_n^2 \tag{B.6}$$

$$\text{cov}(y) = K(X, X) + \sigma_n^2 I \tag{B.7}$$

At some test points X_*, we aim to predict $f_* = f(X_*)$. Then:

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right) \tag{B.8}$$

From which we get:

$$f_*|X_*, X, Y \sim \mathcal{N}(\overline{f_*}, \mathrm{cov}(\mathrm{f}_*)) \tag{B.9}$$

$$\overline{f_*} = K(X_*, X)\left(K(X, X) + \sigma_n^2 I\right)^{-1} Y \tag{B.10}$$

$$\mathrm{cov}(\mathrm{f}_*) = K(X_*, X_*) - K(X_*, X)\left(K(X, X) + \sigma_n^2 I\right)^{-1} K(X, X_*) \tag{B.11}$$

C

KL divergence between two exponential-family distributions

We recall the family of distributions parameterized by $\eta \in \mathbb{R}^K$, over a fixed support $\mathcal{X}^D \in \mathbb{R}^D$: the **exponential family** of distributions $p(x|\eta)$ is given by:

$$p(x|\eta) = \frac{1}{Z(\eta)} h(x) \exp\left(\eta^T \mathcal{T}(x)\right) \tag{C.1}$$

$$= h(x) \exp\left(\eta^T \mathcal{T}(x) - A(\eta)\right) \tag{C.2}$$

with:

- $h(x)$ is the base measure, ie a scaling constant (often 1)
- $\mathcal{T}(x)$ are the sufficient statistics
- η are the natural parameters, or canonical parameters
- $Z(\eta)$ is the partition function, $A(\eta)$ is the log partition function.

The Bernoulli, categorical (ie multinomial for one observation), Gaussian distributions are part of the exponential family.

The $\mathbb{KL}$**-divergence between two exponential family distributions of the same family** is:

$$\mathbb{KL}(p(x|\eta_1)||p(x|\eta_2)) = \mathbb{E}_{\eta_1}\left[(\eta_1 - \eta_2)\mathcal{T}(x) - A(\eta_1) + A(\eta_2)\right] \tag{C.3}$$

$$= (\eta_1 - \eta_2)^T \mathbb{E}_{\eta_1} \mathcal{T}(x) - A(\eta_1) + A(\eta_2) \tag{C.4}$$

The most important example is the $\mathbb{KL}$-divergence between two multivariate Gaussian distributions of dimension D:

KL between two multivariate Gaussians of dimension D

$$\mathbb{KL}(\mathcal{N}(x|\mu_1, \Sigma_1)||\mathcal{N}(x|\mu_2, \Sigma_2) = \frac{1}{2}\left[\mathrm{tr}(\Sigma_2^{-1}\Sigma_1) + (\mu_2 - \mu_1)^T \Sigma_2^{-1}(\mu_2 - \mu_1) - D + \log \frac{|\Sigma_2|}{|\Sigma_1|}\right] \tag{C.5}$$

Bibliography

[1] C. E. Shannon, “A mathematical theory of communication,” *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, Jul. 1948, ISSN: 0005-8580. DOI: 10.1002/j.1538-7305.1948.tb01338.x Accessed: Jul. 28, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/6773024

[2] S. M. Pincus, “Approximate entropy as a measure of system complexity.,” en, *Proceedings of the National Academy of Sciences*, vol. 88, no. 6, pp. 2297–2301, Mar. 1991, ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.88.6.2297 Accessed: Jan. 20, 2025. [Online]. Available: https://pnas.org/doi/full/10.1073/pnas.88.6.2297

[3] C. Bishop, *Pattern recognition and machine learning*, Accessed on Month Day, Year, 2006. [Online]. Available: https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf

[4] T. M. Cover and J. A. Thomas, *Elements of Information Theory 2nd Edition*, English. Hoboken, N.J: Wiley-Interscience, 2006, ISBN: 9780471241959.

[5] *Mouvement brownien et calcul d’Itô-Léonard Gallardo-Editions Hermann*, Sep. 2008. Accessed: Apr. 16, 2025. [Online]. Available: https://www.editions-hermann.fr/livre/mouvement-brownien-et-calcul-d-ito-leonard-gallardo

[6] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning* (Adaptive computation and machine learning), eng, 3. print. Cambridge, Mass.: MIT Press, 2008, ISBN: 9780262182539.

[7] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain, “Gaussian processes for time-series modelling,” en, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 371, no. 1984, p. 20 110 550, Feb. 2013, ISSN: 1364-503X, 1471-2962. DOI: 10.1098/rsta.2011.0550 Accessed: Jul. 23, 2025. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rsta.2011.0550

[8] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information science and statistics), eng, Softcover reprint of the original 1st edition 2006 (corrected at 8th printing 2009). New York, NY: Springer New York, 2016, ISBN: 9781493938438.

[9] F. P. Casale, A. V. Dalca, L. Saglietti, J. Listgarten, and N. Fusi, *Gaussian Process Prior Variational Autoencoders*, en, Oct. 2018. Accessed: Jun. 7, 2025. [Online]. Available: https://arxiv.org/abs/1810.11738v2

[10] A. Delgado-Bonal and A. Marshak, “Approximate Entropy and Sample Entropy: A Comprehensive Tutorial,” en, *Entropy*, vol. 21, no. 6, p. 541, Jun. 2019, ISSN: 1099-4300. DOI: 10.3390/e21060541 Accessed: Jan. 20, 2025. [Online]. Available: https://www.mdpi.com/1099-4300/21/6/541

[11] D. P. Kingma and M. Welling, “An Introduction to Variational Autoencoders,” 2019. DOI: 10.48550/ARXIV.1906.02691 Accessed: Jul. 21, 2025. [Online]. Available: https://arxiv.org/abs/1906.02691

[12] S. Särkkä and A. Solin, *Applied Stochastic Differential Equations* (Institute of Mathematical Statistics Textbooks). Cambridge: Cambridge University Press, 2019, ISBN: 9781316510087. DOI: 10.1017/9781108186735 Accessed: Jul. 23, 2025. [Online]. Available: https://www.cambridge.org/core/books/applied-stochastic-differential-equations/6BB1B8B0819F8C12616E4A0C78C29EAA

[13] V. Fortuin, D. Baranchuk, G. Rätsch, and S. Mandt, *GP-VAE: Deep Probabilistic Time Series Imputation*, arXiv:1907.04155, Feb. 2020. DOI: 10.48550/arXiv.1907.04155 Accessed: Jun. 7, 2025. [Online]. Available: http://arxiv.org/abs/1907.04155

[14] L. Girin, S. Leglaive, X. Bie, J. Diard, T. Hueber, and X. Alameda-Pineda, *Dynamical Variational Autoencoders: A Comprehensive Review*, arXiv:2008.12595, Jul. 2022. DOI: 10.48550/arXiv.2008.12595 Accessed: Jan. 19, 2025. [Online]. Available: http://arxiv.org/abs/2008.12595

[15] K. Murphy, *Probabilistic macine learning advanced topics*, 2023. [Online]. Available: https://mitpress.mit.edu/9780262048439/probabilistic-machine-learning/

[16] H. Zhu, C. B. Rodas, and Y. Li, *Markovian Gaussian Process Variational Autoencoders*, arXiv:2207.05543, Aug. 2023. DOI: 10.48550/arXiv.2207.05543 Accessed: Jul. 23, 2025. [Online]. Available: http://arxiv.org/abs/2207.05543

[17] F. Koller, *Probabilistic Graphical Models*, en-US. Accessed: Jul. 21, 2025. [Online]. Available: https://mitpress.mit.edu/9780262013192/probabilistic-graphical-models/

[18] *Page Web de Jean-François Le Gall*. Accessed: Apr. 16, 2025. [Online]. Available: https://www.imo.universite-paris-saclay.fr/~jean-francois.le-gall/