Structured latent variable models

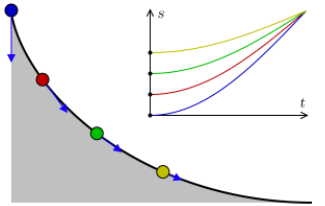
Harri Lähdesmäki

Department of Computer Science
Aalto University

May 19, 2025

Motivation

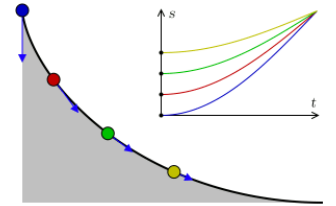
Models for some dynamical systems can be derived from first principles



https://en.wikipedia.org/wiki/Tautochrone_curve

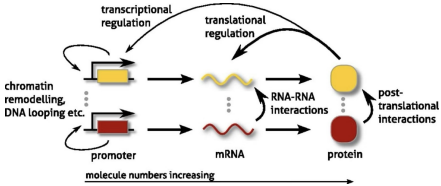
Motivation

Models for some dynamical systems can be derived from first principles



https://en.wikipedia.org/wiki/Tautochrone_curve

Biological networks



(Thomas et al., 2014)

Video applications

Previous frame Current frame

Contents

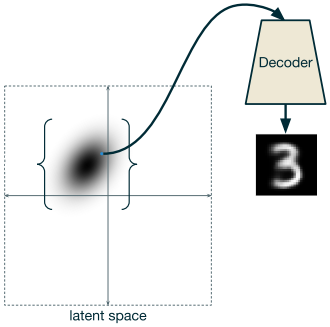
- Deep latent variable models (recap)
- Differential equation models (recap)
- Latent neural ODEs
- Reading: references at the end of slides

Deep latent variable models

Generative models of the form

$$\mathbf{z} \sim p_\theta(\mathbf{z})$$
$$\mathbf{y} \mid \mathbf{z} \sim p_\psi(\mathbf{y} \mid \mathbf{z}) = \mathrm{ExpFam}(\mathbf{y} \mid d_\psi(\mathbf{z}))$$

where $\mathbf{z} \in \mathbb{R}^L$ and $\mathbf{y} \in \mathcal{Y} = \mathbb{R}^D$ and $L \ll D$



https://ijdykeman.github.io/ml/2016/12/21/cvae.html

Deep latent variable models

Generative models of the form

$$\mathbf{z} \sim p_\theta(\mathbf{z})$$
$$\mathbf{y} \mid \mathbf{z} \sim p_\psi(\mathbf{y} \mid \mathbf{z}) = \mathrm{ExpFam}(\mathbf{y} \mid d_\psi(\mathbf{z}))$$

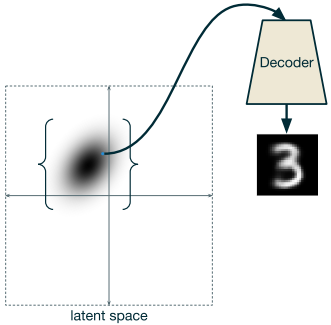
where $\mathbf{z} \in \mathbb{R}^L$ and $\mathbf{y} \in \mathcal{Y} = \mathbb{R}^D$ and $L \ll D$

Upon observing $\mathbf{y}$, we are interested in

$$\text{posterior:} \quad p(\mathbf{z} \mid \mathbf{y}) = \frac{p_\psi(\mathbf{y} \mid \mathbf{z}) p_\theta(\mathbf{z})}{p(\mathbf{y})}$$

$$\text{evidence:} \quad p(\mathbf{y}) = \int p(\mathbf{y} \mid \mathbf{z}) p_\theta(\mathbf{z}) d\mathbf{z}$$

but both quantities are generally intractable



https://ijdykeman.github.io/ml/2016/12/21/cvae.html

Deep latent variable models: amortized VI

Generative models of the form

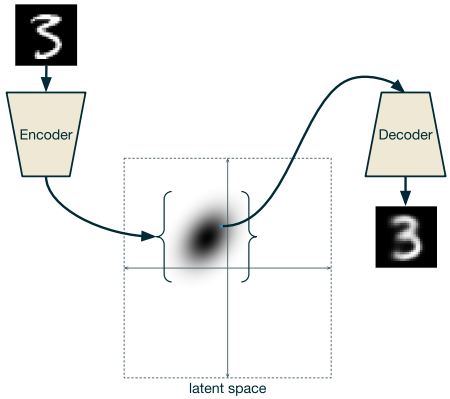
$$\mathbf{z} \sim p_\theta(\mathbf{z})$$
$$\mathbf{y} \mid \mathbf{z} \sim p_\psi(\mathbf{y} \mid \mathbf{z}) = \mathrm{ExpFam}(\mathbf{y} \mid d_\psi(\mathbf{z}))$$

where $\mathbf{z} \in \mathbb{R}^L$ and $\mathbf{y} \in \mathcal{Y} = \mathbb{R}^D$ and $L \ll D$

Auto-encoding variational Bayes uses amortized variational inference

$$p(\mathbf{z} \mid \mathbf{y}) \approx q_\phi(\mathbf{z} \mid \mathbf{y}) = q(\mathbf{z} \mid e_\phi(\mathbf{y}))$$

When ϕ and ψ are neural nets, the model is called the variational autoencoder (VAE)



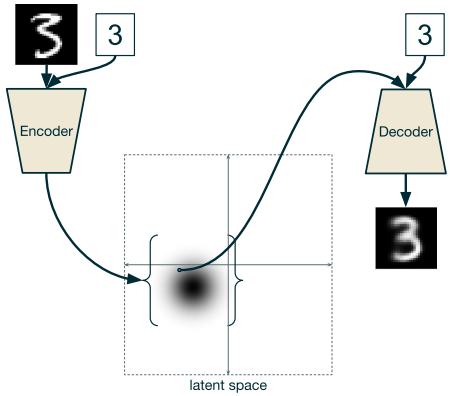
https://ijdykeman.github.io/ml/2016/12/21/cvae.html

Conditional deep latent variable models

Conditional generative models of the form

$$\mathbf{z} \sim p_\theta(\mathbf{z} \mid \mathbf{x})$$
$$\mathbf{y} \mid \mathbf{z} \sim p_\psi(\mathbf{y} \mid \mathbf{z}, \mathbf{x}) = \mathrm{ExpFam}(\mathbf{y} \mid d_\psi(\mathbf{z}, \mathbf{x}))$$

where $\mathbf{x} \in \mathcal{X}$ and use amortized VI with $q_\phi(\mathbf{z} \mid \mathbf{y}, \mathbf{x})$



https://ijdykeman.github.io/ml/2016/12/21/cvae.html

Conditional deep latent variable models

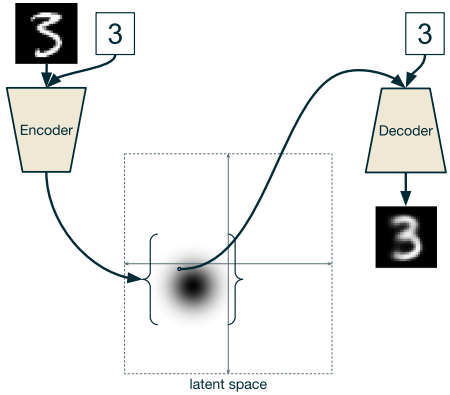
Conditional generative models of the form

$$\mathbf{z} \sim p_\theta(\mathbf{z} \mid \mathbf{x})$$
$$\mathbf{y} \mid \mathbf{z} \sim p_\psi(\mathbf{y} \mid \mathbf{z}, \mathbf{x}) = \mathrm{ExpFam}(\mathbf{y} \mid d_\psi(\mathbf{z}, \mathbf{x}))$$

where $\mathbf{x} \in \mathcal{X}$ and use amortized VI with $q_\phi(\mathbf{z} \mid \mathbf{y}, \mathbf{x})$

Conditional VAE (CVAE) variants:

$$p(\mathbf{y}, \mathbf{z} \mid \mathbf{x}) = p_\psi(\mathbf{y} \mid \mathbf{z}, \mathbf{x}) p_\theta(\mathbf{z} \mid \mathbf{x})$$
$$p(\mathbf{y}, \mathbf{z} \mid \mathbf{x}) = p_\psi(\mathbf{y} \mid \mathbf{z}, \mathbf{x}) p_\theta(\mathbf{z})$$
$$p(\mathbf{y}, \mathbf{z} \mid \mathbf{x}) = p_\psi(\mathbf{y} \mid \mathbf{z}) p_\theta(\mathbf{z} \mid \mathbf{x})$$



https://ijdykeman.github.io/ml/2016/12/21/cvae.html

Limitations of VAEs and CVAEs

1. Cannot model dynamical systems
2. Data samples are independent

Differential equation models

In many applications, it is easier to formulate a mathematical formula for the rate of change of some quantitities (than the quantity itself)

Differential equation models

In many applications, it is easier to formulate a mathematical formula for the rate of change of some quantitities (than the quantity itself)

Consider a state variable $\mathbf{x}(t) \in \mathbb{R}^L$ at time t

Rate of change is often defined w.r.t. time, $\frac{d}{dt}\mathbf{x}(t)$

Differential equation models

In many applications, it is easier to formulate a mathematical formula for the rate of change of some quantitities (than the quantity itself)

Consider a state variable $\mathbf{x}(t) \in \mathbb{R}^L$ at time t

Rate of change is often defined w.r.t. time, $\frac{d}{dt}\mathbf{x}(t)$

A function

$$f_{\boldsymbol{\theta}} : \mathbb{R}^L \times \mathbb{R} \to \mathbb{R}^L$$

is used to model the rate of change

Differential equation models

In many applications, it is easier to formulate a mathematical formula for the rate of change of some quantitities (than the quantity itself)

Consider a state variable $\mathbf{x}(t) \in \mathbb{R}^L$ at time t

Rate of change is often defined w.r.t. time, $\frac{d}{dt}\mathbf{x}(t)$

A function

$$f_{\boldsymbol{\theta}} : \mathbb{R}^L \times \mathbb{R} \to \mathbb{R}^L$$

is used to model the rate of change

This leads to the ordinary differential equation system

$$\frac{d\mathbf{x}(t)}{dt} = f_{\boldsymbol{\theta}}(\mathbf{x}(t), t)$$

Differential equation models

In many applications, it is easier to formulate a mathematical formula for the rate of change of some quantitities (than the quantity itself)

Consider a state variable $\mathbf{x}(t) \in \mathbb{R}^L$ at time t

Rate of change is often defined w.r.t. time, $\frac{d}{dt}\mathbf{x}(t)$

A function

$$f_{\boldsymbol{\theta}} : \mathbb{R}^L \times \mathbb{R} \to \mathbb{R}^L$$

is used to model the rate of change

This leads to the ordinary differential equation system

$$\frac{d\mathbf{x}(t)}{dt} = f_{\boldsymbol{\theta}}(\mathbf{x}(t), t)$$

Example: 2-D Lotka-Volterra system

$$\begin{cases} \frac{dx_1(t)}{dt} = k_1 x_1(t) - k_2 x_1(t) x_2(t) \\ \frac{dx_2}{dt} = k_2 x_1(t) x_2(t) - k_3 x_2(t) \end{cases}$$

Differential equation models

In many applications, it is easier to formulate a mathematical formula for the rate of change of some quantitities (than the quantity itself)

Consider a state variable $\mathbf{x}(t) \in \mathbb{R}^L$ at time t

Rate of change is often defined w.r.t. time, $\frac{d}{dt}\mathbf{x}(t)$

A function

$$f_{\boldsymbol{\theta}} : \mathbb{R}^L \times \mathbb{R} \to \mathbb{R}^L$$

is used to model the rate of change

This leads to the ordinary differential equation system

$$\frac{d\mathbf{x}(t)}{dt} = f_{\boldsymbol{\theta}}(\mathbf{x}(t), t)$$

Example: 2-D Lotka-Volterra system

$$\begin{cases} \frac{dx_1(t)}{dt} = k_1 x_1(t) - k_2 x_1(t) x_2(t) \\ \frac{dx_2}{dt} = k_2 x_1(t) x_2(t) - k_3 x_2(t) \end{cases}$$

If the function $f_{\boldsymbol{\theta}}$ is a neural network, then this model is called neural ODE (NODE) (Chen et al., 2018)

ODE defines state variables, $\mathbf{x}(t)$ $\forall t$, implicitly

Initial value problem for ODEs

We need to be able to solve the trajectories $\mathbf{x}(t)$ over time given some initial value $\mathbf{x}(t_0)$

The initial value problem consists of

- An ODE model $\frac{d}{dt}\mathbf{x}(t) = f_\theta(\mathbf{x}(t))$, and
- An initial value $\mathbf{x}(t_0) = \mathbf{x}_0$

Initial value problem for ODEs

We need to be able to solve the trajectories $\mathbf{x}(t)$ over time given some initial value $\mathbf{x}(t_0)$

The initial value problem consists of

- An ODE model $\frac{d}{dt}\mathbf{x}(t) = f_\theta(\mathbf{x}(t))$, and
- An initial value $\mathbf{x}(t_0) = \mathbf{x}_0$

A solution to an initial value problem is a function $\mathbf{x}(t)$ defined for all $t \in \mathbb{R}$ that

- Is a solution of the ODE model
- Satisfies the initial value $\mathbf{x}(t_0) = \mathbf{x}_0$

Initial value problem for ODEs

We need to be able to solve the trajectories $\mathbf{x}(t)$ over time given some initial value $\mathbf{x}(t_0)$

The initial value problem consists of

- An ODE model $\frac{d}{dt}\mathbf{x}(t) = f_\theta(\mathbf{x}(t))$, and
- An initial value $\mathbf{x}(t_0) = \mathbf{x}_0$

A solution to an initial value problem is a function $\mathbf{x}(t)$ defined for all $t \in \mathbb{R}$ that

- Is a solution of the ODE model
- Satisfies the initial value $\mathbf{x}(t_0) = \mathbf{x}_0$

The solution of the initial value problem is unique* and can be obtained by integrating both sides of the system

$$\mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} f_\theta(\mathbf{x}(t))dt$$

Often the solution to the initial value problem must be obtained using numerical techniques

Numerical simulation methods: Euler method for ODEs

- Euler method is the simplest numerical simulation method for ODEs
- State vector $\mathbf{x}(t) \in \mathbb{R}^L$ and an arbitrary function $f_{\boldsymbol{\theta}} : \mathbb{R}^L \to \mathbb{R}^L$ of $\mathbf{x}(t)$ with parameters $\boldsymbol{\theta}$

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= f_{\boldsymbol{\theta}}(\mathbf{x}(t)) \\ \lim_{\delta t \to 0} \frac{\mathbf{x}(t+\delta t) - \mathbf{x}(t)}{\delta t} &= f_{\boldsymbol{\theta}}(\mathbf{x}(t)) \end{aligned}$$

Numerical simulation methods: Euler method for ODEs

- Euler method is the simplest numerical simulation method for ODEs
- State vector $\mathbf{x}(t) \in \mathbb{R}^L$ and an arbitrary function $f_{\boldsymbol{\theta}} : \mathbb{R}^L \to \mathbb{R}^L$ of $\mathbf{x}(t)$ with parameters $\boldsymbol{\theta}$

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= f_{\boldsymbol{\theta}}(\mathbf{x}(t)) \\ \lim_{\delta t \to 0} \frac{\mathbf{x}(t+\delta t) - \mathbf{x}(t)}{\delta t} &= f_{\boldsymbol{\theta}}(\mathbf{x}(t)) \end{aligned}$$

- For small values of δt this can be approximated with the finite diffidence Δt as

$$\frac{\mathbf{x}(t+\Delta t) - \mathbf{x}(t)}{\Delta t} \approx f_{\boldsymbol{\theta}}(\boldsymbol{\theta}(t))$$

and by solving for $\mathbf{x}(t+\Delta t)$ one gets

$$\mathbf{x}(t+\Delta t) \approx \mathbf{x}(t) + \Delta t \cdot f_{\boldsymbol{\theta}}(\boldsymbol{\theta}(t))$$

Euler method for ODEs: algorithm

- Given an initial value $\mathbf{x}(t_0)$, the first-order can be applied recursively to compute $\mathbf{x}(t_0)$, $\mathbf{x}(t_0 + \Delta t)$, $\mathbf{x}(t_0 + 2\Delta t)$, $\mathbf{x}(t_0 + 3\Delta t)$, ...

$$\begin{aligned}
\mathbf{x}(t_0 + \Delta t) &= \mathbf{x}(t_0) + \Delta t \cdot f_{\boldsymbol{\theta}}(\mathbf{x}(t_0)) \\
\mathbf{x}(t_0 + 2\Delta t) &= \mathbf{x}(t_0 + \Delta t) + \Delta t \cdot f_{\boldsymbol{\theta}}(\mathbf{x}(t_0 + \Delta t)) \\
\mathbf{x}(t_0 + 3\Delta t) &= \mathbf{x}(t_0 + 2\Delta t) + \Delta t \cdot f_{\boldsymbol{\theta}}(\mathbf{x}(t_0 + 2\Delta t)) \\
&\vdots
\end{aligned}$$

  which can be used to approximate the exact solution

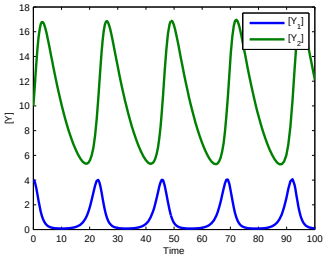
$$\mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} f_{\boldsymbol{\theta}}(\mathbf{x}(t))dt$$

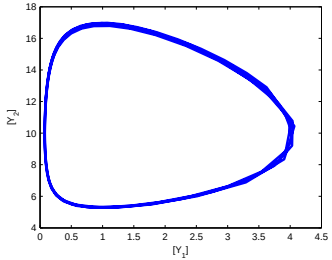
  at discrete time points $t_0, t_0 + \Delta t, t_0 + 2\Delta t, \ldots$
- Interpolation between consecutive time points $\mathbf{x}(t_0 + i \cdot \Delta t)$ and $\mathbf{x}(t_0 + (i+1) \cdot \Delta t)$, if needed
- A number of more advanced ODE simulation methods exist

Euler method for ODEs: example

Lotka–Volterra (system of ODEs)

$$\begin{cases} \frac{dx_1(t)}{dt} = k_1 x_1(t) - k_2 x_1(t) x_2(t) \\ \frac{dx_2(t)}{dt} = k_2 x_1(t) x_2(t) - k_3 x_2(t) \end{cases}$$

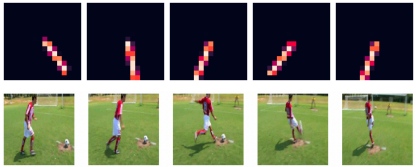




Problem setting: latent dynamics

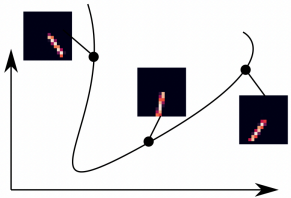
Data: high-dimensional observations $\mathbf{y} \in \mathbb{R}^D$ of a dynamical system at arbitrary time points across several trajectories

$$\mathbf{y}_{1:N} = (\mathbf{y}_1, \ldots, \mathbf{y}_N)$$
$$\mathbf{t}_{1:N} = (t_1, \ldots, t_N)$$



Goal: model the data using continuous-time dynamics in a low dimensional space $\mathcal{X} = \mathbb{R}^d$, $d \ll D$

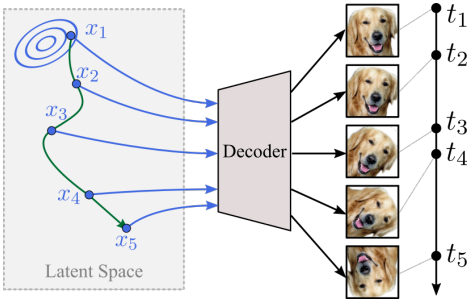
$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(t, \mathbf{x}(t), \boldsymbol{\theta}_{\text{dyn}})$$



Generative latent dynamics model

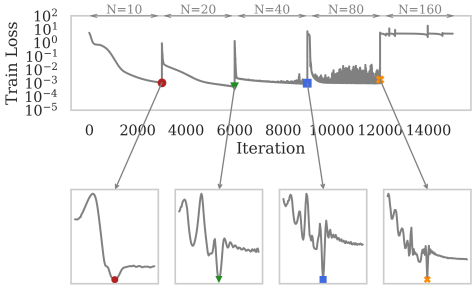
Latent neural ODEs (Chen et al, 2018; Rubinova et al, 2019; Yildiz et al, 2019)

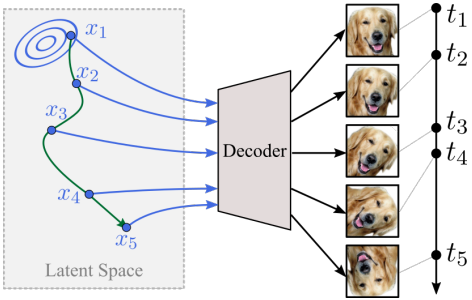
$$
\begin{aligned}
\mathbf{x}_1 &\sim p(\mathbf{x}_1) \\
\boldsymbol{\theta}_{\text{dyn}} &\sim p(\boldsymbol{\theta}_{\text{dyn}}) \\
\mathbf{x}_i &= \text{ODEsolve}(\mathbf{x}_1, t_1, t_i, \theta_{\text{dyn}}), \quad i > 1 \\
\mathbf{y}_i \mid \mathbf{x}_i &\sim p(\mathbf{y}_i \mid g(\mathbf{x}_i, \boldsymbol{\theta}_{\text{dec}}))
\end{aligned}
$$



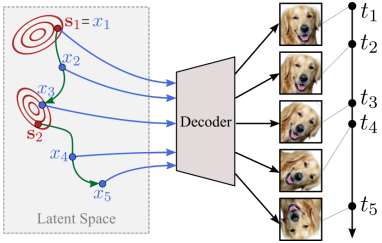
Challenges with latent NODE models

Training on long trajectories is slow and unstable





Multiple shooting and continuity promoting prior

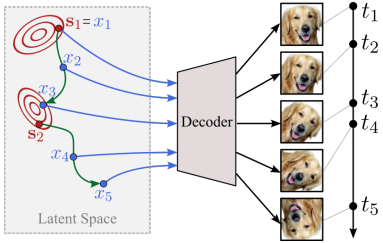


Multiple shooting and continuity promoting prior

- Shooting variables $\mathbf{s}_{1:B} = (\mathbf{s}_1, \ldots, \mathbf{s}_B)$, $B \leq N - 1$
- Continuity promoting prior

$$p(\mathbf{s}_{1:B} \mid \boldsymbol{\theta}_{\text{dyn}}) = p(\mathbf{s}_1) \prod_{b=2}^{B} p(\mathbf{s}_b \mid \mathbf{s}_{b-1}, \boldsymbol{\theta}_{\text{dyn}})$$

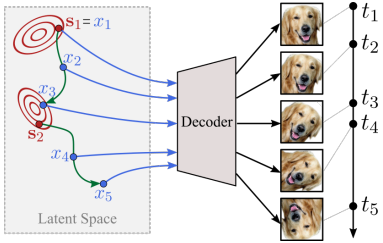
$$= p(\mathbf{s}_1) \prod_{b=2}^{B} \mathcal{N}\left(\mathbf{s}_b \mid \text{ODEsolve}(\mathbf{s}_{b-1}, t_{[b-1]}, t_{[b]}, \boldsymbol{\theta}_{\text{dyn}}), \sigma_c^2 I\right)$$



Latent NODEs with sparse Bayesian MS (Iakovlev et al., 2023)

- Generative model

$$
\begin{aligned}
\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}} &\sim p(\boldsymbol{\theta}_{\text{dyn}}) p(\boldsymbol{\theta}_{\text{dec}}) \\
\mathbf{s}_{1:B} \mid \boldsymbol{\theta}_{\text{dyn}} &\sim p(\mathbf{s}_{1:B} \mid \boldsymbol{\theta}_{\text{dyn}}) \\
\mathbf{x}_i &= \text{ODEsolve}(\mathbf{s}_b, t_{[b]}, t_i, \boldsymbol{\theta}_{\text{dyn}}), \quad i > 1 \\
\mathbf{y}_i \mid \mathbf{x}_i &\sim p(\mathbf{y}_i \mid g(\mathbf{x}_i, \boldsymbol{\theta}_{\text{dec}}))
\end{aligned}
$$



Amortized variational inference

Approximate the posterior $p(\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_{1:B} \mid \mathbf{y}_{1:N})$ with

$$q(\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_{1:B}) = q(\boldsymbol{\theta}_{\text{dyn}}; \psi_{\text{dyn}}) q(\boldsymbol{\theta}_{\text{dec}}; \psi_{\text{dec}}) \prod_{b=1}^{B} q(\mathbf{s}_b; \psi_b),$$

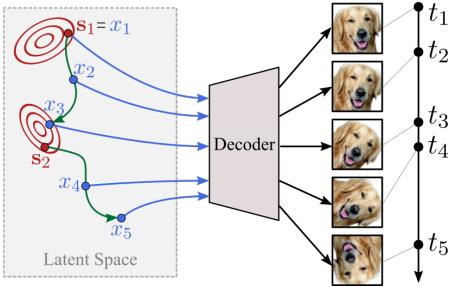
where inference for the local variables $\mathbf{s}_{1_B}$ are amortized $\psi_{1:B} = h(\mathbf{y}_{1:N}; \boldsymbol{\theta}_{\text{enc}})$

Amortized variational inference

Approximate the posterior $p(\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_{1:B} \mid \mathbf{y}_{1:N})$ with

$$q(\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_{1:B}) = q(\boldsymbol{\theta}_{\text{dyn}}; \psi_{\text{dyn}}) q(\boldsymbol{\theta}_{\text{dec}}; \psi_{\text{dec}}) \prod_{b=1}^{B} q(\mathbf{s}_b; \psi_b),$$

where inference for the local variables $\mathbf{s}_{1_B}$ are amortized $\psi_{1:B} = h(\mathbf{y}_{1:N}; \boldsymbol{\theta}_{\text{enc}})$

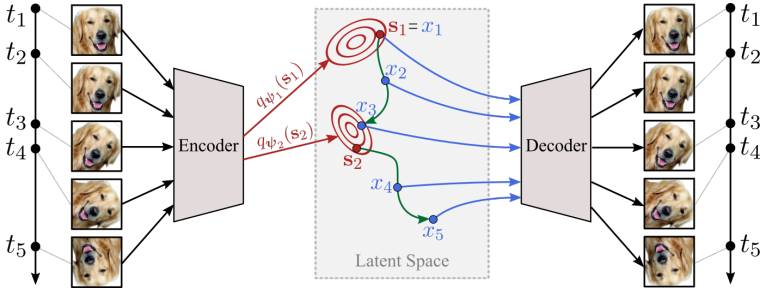


Amortized variational inference

Approximate the posterior $p(\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_{1:B} \mid \mathbf{y}_{1:N})$ with

$$q(\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_{1:B}) = q(\boldsymbol{\theta}_{\text{dyn}}; \boldsymbol{\psi}_{\text{dyn}}) q(\boldsymbol{\theta}_{\text{dec}}; \boldsymbol{\psi}_{\text{dec}}) \prod_{b=1}^{B} q(\mathbf{s}_b; \boldsymbol{\psi}_b),$$

where inference for the local variables $\mathbf{s}_{1_B}$ are amortized $\boldsymbol{\psi}_{1:B} = h(\mathbf{y}_{1:N}; \boldsymbol{\theta}_{\text{enc}})$



Attention-based encoder

Flexible and efficient encoder for

- Irregular time grids
- New time grids
- Long trajectories
- Parallelization
- Noisy and partially observed data

Attention-based encoder

Flexible and efficient encoder for

- Irregular time grids
- New time grids
- Long trajectories
- Parallelization
- Noisy and partially observed data

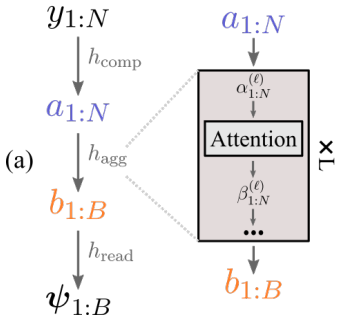
$$\begin{aligned}\psi_{1:B} &= h(\mathbf{y}_{1:N}; \boldsymbol{\theta}_{\text{enc}}) \\ &= h_{\text{read}}(h_{\text{agg}}(h_{\text{comp}}(\mathbf{y}_{1:N})))\end{aligned}$$

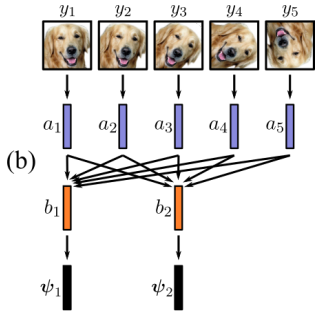
Attention-based encoder

Flexible and efficient encoder for

- Irregular time grids
- New time grids
- Long trajectories
- Parallelization
- Noisy and partially observed data

$$\begin{aligned}\psi_{1:B} &= h(\mathbf{y}_{1:N}; \boldsymbol{\theta}_{\text{enc}}) \\ &= h_{\text{read}}(h_{\text{agg}}(h_{\text{comp}}(\mathbf{y}_{1:N})))\end{aligned}$$





Attention-based encoder

Standard attention

$$\boldsymbol{C}_{ij}^{\mathrm{DP}} = \frac{\langle W_Q \boldsymbol{\alpha}_i, W_K \boldsymbol{\alpha}_j \rangle}{\sqrt{D_{\mathrm{low}}}}$$

$$\boldsymbol{C}_{ij} = \frac{\exp\left(\boldsymbol{C}_{ij}^{\mathrm{DP}} \blacksquare\right)}{\sum_{k=1}^{N} \exp\left(\boldsymbol{C}_{ik}^{\mathrm{DP}} \blacksquare\right)}$$

$$\boldsymbol{\beta}_i = \sum_{j=1}^{N} \boldsymbol{C}_{ij} (W_V \boldsymbol{\alpha}_j \blacksquare)$$

Attention-based encoder

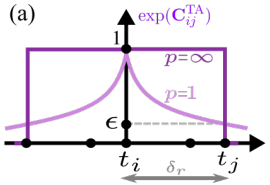
$$C_{ij}^{\mathrm{DP}} = \frac{\langle W_Q \boldsymbol{\alpha}_i, W_K \boldsymbol{\alpha}_j \rangle}{\sqrt{D_{\mathrm{low}}}}$$

$$C_{ij} = \frac{\exp\left(C_{ij}^{\mathrm{DP}} + C_{ij}^{\mathrm{TA}}\right)}{\sum_{k=1}^{N} \exp\left(C_{ik}^{\mathrm{DP}} + C_{ik}^{\mathrm{TA}}\right)}$$

$$\boldsymbol{\beta}_i = \sum_{j=1}^{N} C_{ij} (W_V \boldsymbol{\alpha}_j [redacted])$$

Time-aware attention

$$C_{ij}^{\mathrm{TA}} = \ln(\epsilon) \left(\frac{|t_j - t_i|}{\delta_r} \right)^p$$



Attention-based encoder

$$\boldsymbol{C}_{ij}^{\mathrm{DP}} = \frac{\langle W_Q \boldsymbol{\alpha}_i, W_K \boldsymbol{\alpha}_j \rangle}{\sqrt{D_{\mathrm{low}}}}$$

$$\boldsymbol{C}_{ij} = \frac{\exp\left(\boldsymbol{C}_{ij}^{\mathrm{DP}} + \boldsymbol{C}_{ij}^{\mathrm{TA}}\right)}{\sum_{k=1}^{N} \exp\left(\boldsymbol{C}_{ik}^{\mathrm{DP}} + \boldsymbol{C}_{ik}^{\mathrm{TA}}\right)}$$

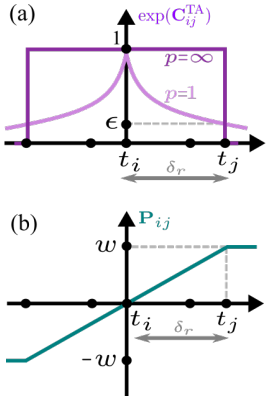
$$\boldsymbol{\beta}_i = \sum_{j=1}^{N} \boldsymbol{C}_{ij} (W_V \boldsymbol{\alpha}_j + \boldsymbol{P}_{ij})$$

Time-aware attention

$$\boldsymbol{C}_{ij}^{\mathrm{TA}} = \ln(\epsilon) \left(\frac{|t_j - t_i|}{\delta_r} \right)^p$$

$$\boldsymbol{P}_{ij} = \boldsymbol{w} \odot \mathrm{hardtanh}\left(\frac{t_j - t_i}{\delta_r} \right)$$

Continuous relative positional encodings



Training objective

Maximize the ELBO w.r.t. $\boldsymbol{\psi}_{\text{dyn}}, \boldsymbol{\psi}_{\text{dec}}, \boldsymbol{\theta}_{\text{enc}}$

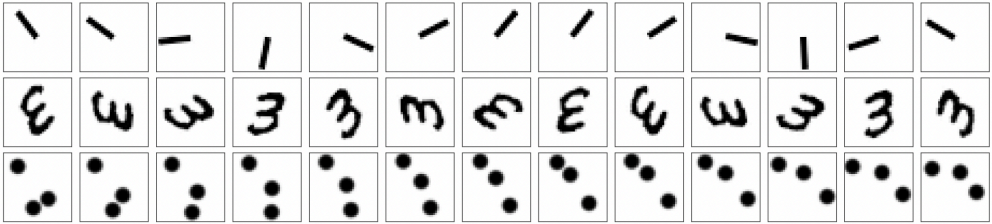
$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_1)}\big[\log p(\mathbf{y}_1 \mid \mathbf{s}_1, \boldsymbol{\theta}_{\text{dec}})\big]}_{(i)\text{ data likelihood}} + \sum_{b=1}^{B} \sum_{i \in \mathcal{I}_b} \underbrace{\mathbb{E}_{q(\boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}}, \mathbf{s}_b)}\big[\log p(\mathbf{y}_i \mid \mathbf{s}_b, \boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{dec}})\big]}_{(ii)\text{ data likelihood}}$$

$$- \underbrace{\mathrm{KL}\big[q(\mathbf{s}_1) \| p(\mathbf{s}_1)\big]}_{(iii)\text{ initial state prior}} - \sum_{b=2}^{B} \underbrace{\mathbb{E}_{q(\boldsymbol{\theta}_{\text{dyn}}, \mathbf{s}_{b-1})}\Big[\mathrm{KL}\big[q(\mathbf{s}_b) \| p(\mathbf{s}_b \mid \mathbf{s}_{b-1}, \boldsymbol{\theta}_{\text{dyn}})\big]\Big]}_{(iv)\text{ continuity prior}}$$

$$- \underbrace{\mathrm{KL}\big[q(\boldsymbol{\theta}_{\text{dyn}}) \| p(\boldsymbol{\theta}_{\text{dyn}})\big]}_{(v)\text{ dynamics prior}} - \underbrace{\mathrm{KL}\big[q(\boldsymbol{\theta}_{\text{dec}}) \| p(\boldsymbol{\theta}_{\text{dec}})\big]}_{(vi)\text{ decoder prior}}$$

Multiple shooting + factorized approximation $\prod q(\mathbf{s}_b; \boldsymbol{\psi}_b)$ allow efficient parallelization

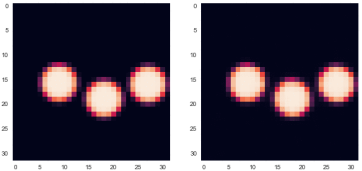
Benchmark datasets, $D = 1024$



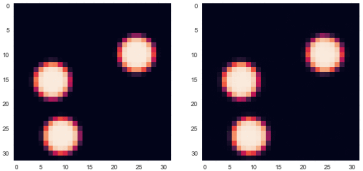
- PENDULUM: images of pendulum moving under the influence of gravity, $n = 400$
- RMNIST: images of rotating digits with varying angular velocity, $n = 4000$
- BOUNCING BALLS: images of three balls bouncing in a box, $n = 10000$

L-NODE visualization

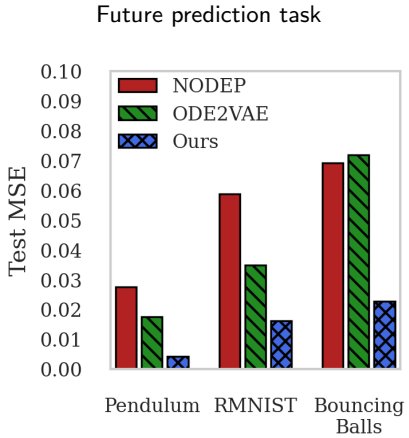
Data Prediction



Data Prediction



L-NODE vs. previous methods

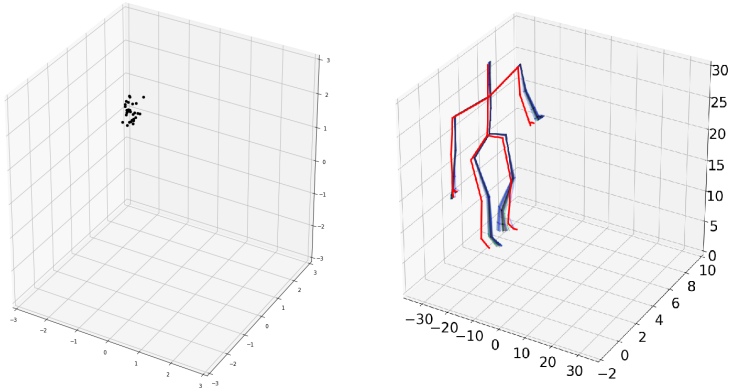


Ours = L-NODE methods from (Iakovlev et al., 2023)

CMU walking data

Probabilistic modeling and inference of latent dynamics models provide

- Uncertainty quantification
- Posterior predictive sampling



Modeling transcriptional dynamics using scRNA-seq data

Time-series scRNA-seq

- Large number of high-dim. observations $\mathbf{y}_{tc}$ of individual cells c per time point t
- Cells have no correspondence between time points

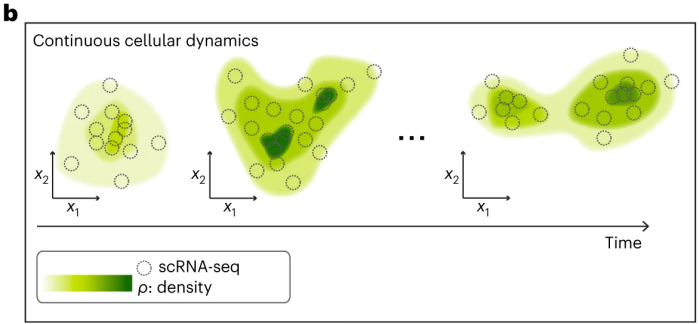


Figure from (Sha et a., 2024)

Model transcriptional dynamics with scNODE and optimal transport

- Wasserstein distance between observed and predicted transcriptional profiles (Zhang et al., 2024)

$$\mathbf{Y}_t = (\mathbf{y}_{t1}, \ldots, \mathbf{y}_{tN})$$
$$\hat{\mathbf{Y}}_t = (\hat{\mathbf{y}}_{t1}, \ldots, \hat{\mathbf{y}}_{tN})$$
$$D_{ij} = \|\mathbf{y}_{ti} - \mathbf{y}_{tj}\|_2$$
$$\mathrm{W}(\mathbf{Y}_t, \hat{\mathbf{Y}}_t) = \left(\min_{\Gamma \in \Pi(\mathbf{Y}_t, \hat{\mathbf{Y}}_t)} \sum_{i,j} D_{ij}^2 \Gamma_{ij} \right)^{1/2}$$

Model transcriptional dynamics with scNODE and optimal transport

- Wasserstein distance between observed and predicted transcriptional profiles (Zhang et al., 2024)

$$\mathbf{Y}_t = (\mathbf{y}_{t1}, \ldots, \mathbf{y}_{tN})$$
$$\hat{\mathbf{Y}}_t = (\hat{\mathbf{y}}_{t1}, \ldots, \hat{\mathbf{y}}_{tN})$$
$$D_{ij} = \|\mathbf{y}_{ti} - \mathbf{y}_{tj}\|_2$$
$$\mathrm{W}(\mathbf{Y}_t, \hat{\mathbf{Y}}_t) = \left(\min_{\Gamma \in \Pi(\mathbf{Y}_t, \hat{\mathbf{Y}}_t)} \sum_{i,j} D_{ij}^2 \Gamma_{ij} \right)^{1/2}$$

- Train latent NODE with Wasserstein-2 loss (Zhang et al., 2024)

$$\mathcal{L} = \sum_t \mathrm{W}(\mathbf{Y}_t, \hat{\mathbf{Y}}_t) + \beta \mathrm{W}(\mathbf{X}_t, \hat{\mathbf{X}}_t)$$

Model transcriptional dynamics with scNODE and optimal transport

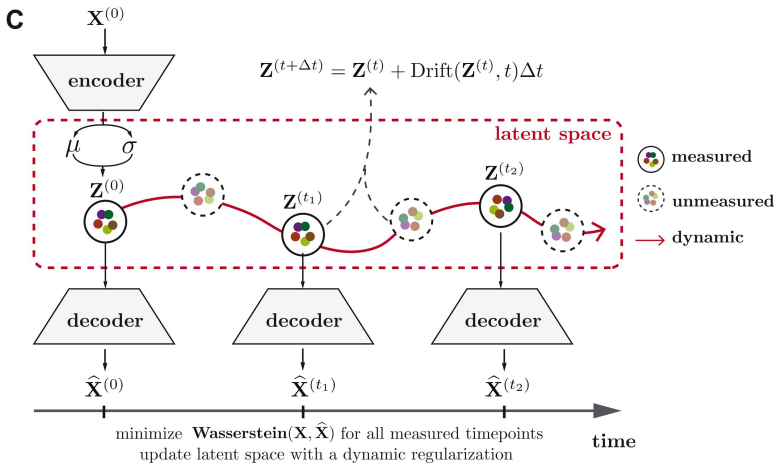


Figure from (Zhang et al., 2024)

Model transcriptional dynamics with scNODE and optimal transport

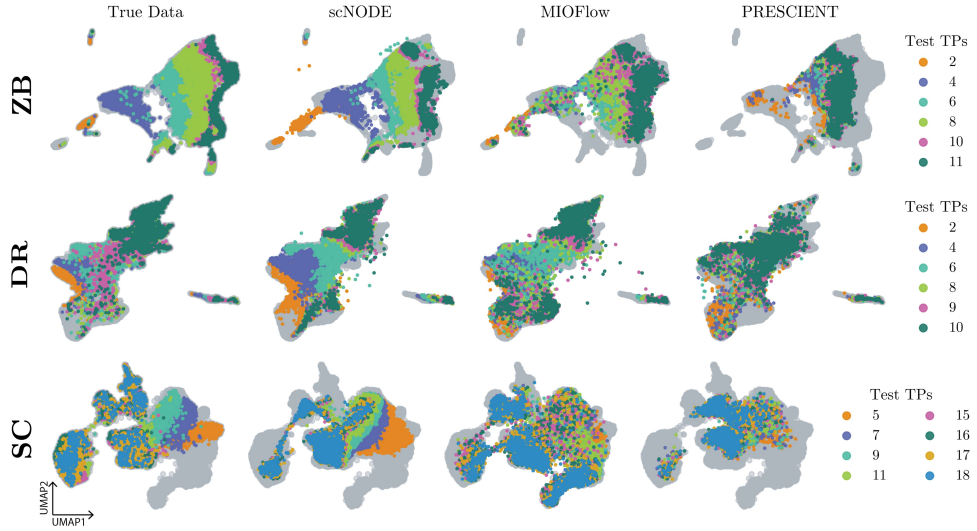


Figure from (Zhang et al., 2024)

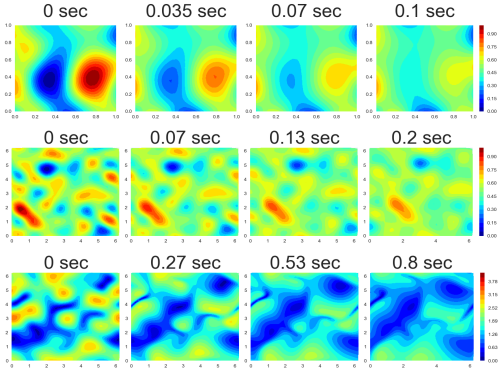
Spatiotemporal processes

Many physics systems are inherently spatiotemporal and continuous over both

- time t
- space $\mathbf{x} \in \Omega$

(Iakovlev et al., 2024)

A latent variable method to learn space-time continuous neural PDEs from partially observed states



Figures from (Iakovlev et al., 2021)

Limitations of VAEs and CVAEs

1. Cannot model dynamical systems
2. Data samples are independent

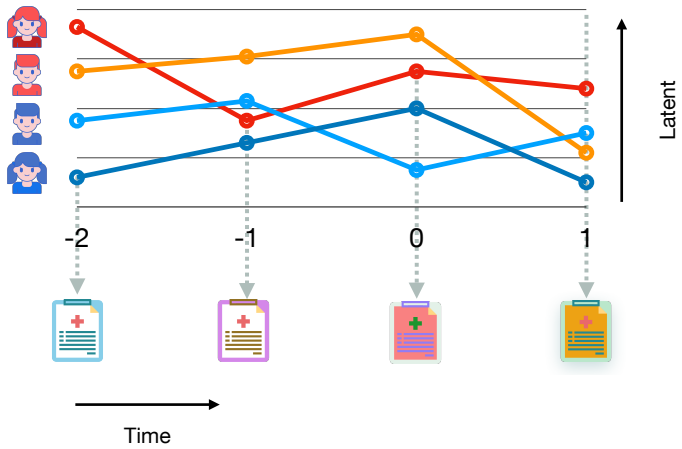
Why correlation are important: longitudinal data

Time-series data that consist of

- Multiple subjects
- Each subject measured repeatedly over time

Observations have correlations

- Within a subject
- Across multiple subjects



Gaussian process prior VAE (Casale et al., 2018)

The L-dimensional latent $\mathbf{z}$ is assigned a multi-output Gaussian process prior, conditioned on $\mathbf{x}$

$$\mathbf{z} = \mathbf{f}(\mathbf{x}) \quad \text{where} \quad \mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}(\mathbf{x}, \mathbf{x}'|\theta))$$

where $\mathbf{K}(\mathbf{x}, \mathbf{x}'|\theta)$ is a matrix-valued $(L \times L)$ cross-covariance function

Gaussian process prior VAE (Casale et al., 2018)

The L-dimensional latent $\mathbf{z}$ is assigned a multi-output Gaussian process prior, conditioned on $\mathbf{x}$

$$\mathbf{z} = \mathbf{f}(\mathbf{x}) \quad \text{where} \quad \mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}(\mathbf{x}, \mathbf{x}'|\theta))$$

where $\mathbf{K}(\mathbf{x}, \mathbf{x}'|\theta)$ is a matrix-valued $(L \times L)$ cross-covariance function

For any finite collection of inputs $X = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, the latents have a joint correlated Gaussian distribution

$$p_\theta \left(\begin{bmatrix} \mathbf{f}(\mathbf{x}_1) \\ \mathbf{f}(\mathbf{x}_2) \\ \vdots \\ \mathbf{f}(\mathbf{x}_N) \end{bmatrix} \right) = \mathcal{N} \left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K}(\mathbf{x}_1, \mathbf{x}_1|\theta) & \mathbf{K}(\mathbf{x}_1, \mathbf{x}_2|\theta) & \cdots & \mathbf{K}(\mathbf{x}_1, \mathbf{x}_N|\theta) \\ \mathbf{K}(\mathbf{x}_2, \mathbf{x}_1|\theta) & \mathbf{K}(\mathbf{x}_2, \mathbf{x}_2|\theta) & \cdots & \mathbf{K}(\mathbf{x}_2, \mathbf{x}_N|\theta) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{K}(\mathbf{x}_N, \mathbf{x}_1|\theta) & \mathbf{K}(\mathbf{x}_N, \mathbf{x}_2|\theta) & \cdots & \mathbf{K}(\mathbf{x}_N, \mathbf{x}_N|\theta) \end{bmatrix} \right)$$

Multi-output additive GP prior VAE (Ramchandran et al., 2021)

The L-dimensional latent $\mathbf{z}$ has an additive GP prior, conditioned on $\mathbf{x}$

$$\mathbf{z} = \mathbf{f}^{(1)}(\mathbf{x}) + \mathbf{f}^{(2)}(\mathbf{x}) + \ldots + \mathbf{f}^{(R)}(\mathbf{x}) + \mathrm{diag}(\sigma_{z1}^2, \ldots, \sigma_{zL}^2),$$

where each

$$\mathbf{f}^{(r)}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}^{(r)}(\mathbf{x}, \mathbf{x}'|\theta))$$

Multi-output additive GP prior VAE (Ramchandran et al., 2021)

The L-dimensional latent $\mathbf{z}$ has an additive GP prior, conditioned on $\mathbf{x}$

$$\mathbf{z} = \mathbf{f}^{(1)}(\mathbf{x}) + \mathbf{f}^{(2)}(\mathbf{x}) + \ldots + \mathbf{f}^{(R)}(\mathbf{x}) + \mathrm{diag}(\sigma_{z1}^2, \ldots, \sigma_{zL}^2),$$

where each

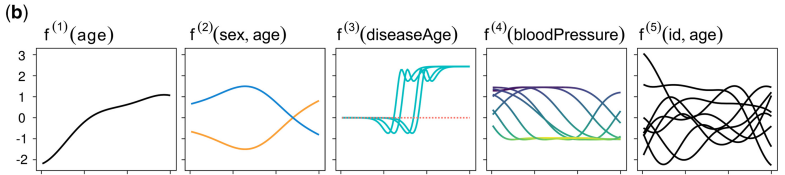
$$\mathbf{f}^{(r)}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}^{(r)}(\mathbf{x}, \mathbf{x}'|\theta))$$

The conditional generative model becomes

$$\mathbf{z} \mid \mathbf{x} \sim \mathcal{GP}\left(\mathbf{0}, \sum_{r=1}^{R} \mathbf{K}^{(r)}(\mathbf{x}, \mathbf{x}'|\theta) + \mathrm{diag}(\sigma_{z1}^2, \ldots, \sigma_{zL}^2)\right)$$
$$\mathbf{y} \mid \mathbf{z} \sim \mathrm{ExpFam}(\mathbf{y} \mid d_\psi(\mathbf{z}))$$

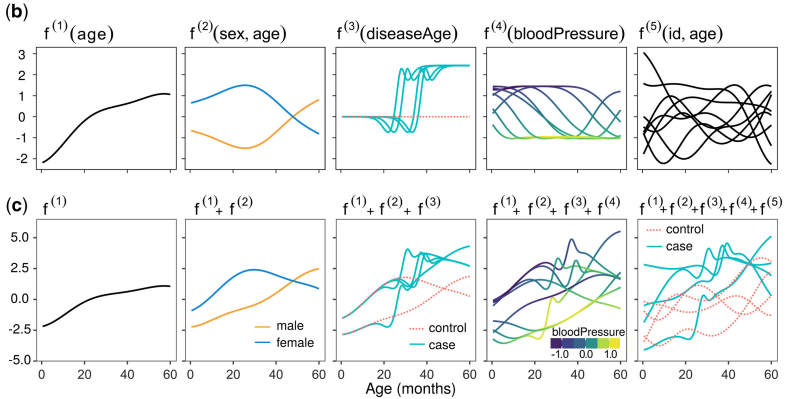
Example: covariance functions for longitudinal data

- Shared effects: the exponentiated quadratic (EQ) kernel
- Category effects: product of the zero-sum and EQ kernels
- Nonstationary shared effects: EQ kernel with monotonic nonlinear input warping
- Continuous time-varying effects: EQ kernel
- Individual random components



Example: covariance functions for longitudinal data

- Shared effects: the exponentiated quadratic (EQ) kernel
- Category effects: product of the zero-sum and EQ kernels
- Nonstationary shared effects: EQ kernel with monotonic nonlinear input warping
- Continuous time-varying effects: EQ kernel
- Individual random components



The generative model: L-VAE

Dataset (X, Y) of size N

$$X = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \quad \textit{covariates}$$
$$Y = [\mathbf{y}_1, \ldots, \mathbf{y}_N] \quad \textit{data}$$
$$Z = [\mathbf{z}_1, \ldots, \mathbf{z}_N] \quad \textit{latents}$$

The generative model: L-VAE

Dataset (X, Y) of size N

$$
\begin{aligned}
X &= [\mathbf{x}_1, \ldots, \mathbf{x}_N] && \textit{covariates} \\
Y &= [\mathbf{y}_1, \ldots, \mathbf{y}_N] && \textit{data} \\
Z &= [\mathbf{z}_1, \ldots, \mathbf{z}_N] && \textit{latents}
\end{aligned}
$$

The marginal likelihood of Y given X

$$
\begin{aligned}
p(Y \mid X) &= \int_Z p_\psi(Y \mid Z) p_\theta(Z \mid X) dZ \\
&= \int_Z \left(\prod_{n=1}^{N} \underbrace{p(\mathbf{y}_n \mid d_\psi(\mathbf{z}_n))}_{\text{likelihoods}} \right) \underbrace{p_\theta(Z \mid X)}_{\text{GP prior}} dZ,
\end{aligned}
$$

Auto-encoding variational Bayes for L-VAE

Approximate the posterior of Z (and estimate θ, ψ, ϕ) using amortized VI

$$P(Z|X,Y) \approx q_\phi(Z|Y) = \prod_{n=1}^{N} \mathcal{N}\left(\mathbf{z}_n \mid \boldsymbol{\mu}_\phi(\mathbf{y}_n), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{y}_n))\right)$$

where $\boldsymbol{\mu}_\phi$ and $\boldsymbol{\sigma}^2_\phi$ are encoder neural networks

Auto-encoding variational Bayes for L-VAE

Approximate the posterior of Z (and estimate θ, ψ, ϕ) using amortized VI

$$P(Z|X,Y) \approx q_\phi(Z|Y) = \prod_{n=1}^{N} \mathcal{N}\left(\mathbf{z}_n \mid \boldsymbol{\mu}_\phi(\mathbf{y}_n), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{y}_n))\right)$$

where $\boldsymbol{\mu}_\phi$ and $\boldsymbol{\sigma}^2_\phi$ are encoder neural networks

ELBO objective:

$$\log p(Y \mid X) \geq \mathcal{L}(\phi, \psi, \theta; Y, X) \triangleq \underbrace{\mathbb{E}_{q_\phi(Z|Y)}\left[\log p_\psi(Y \mid Z)\right]}_{\text{reconstruction: easy}} - \underbrace{D_{\mathbb{KL}}(q_\phi(Z \mid Y) \,||\, \overbrace{p_\theta(Z|X)}^{\text{GP prior}})}_{\text{regularization: closed form, but slow}}$$

Auto-encoding variational Bayes for L-VAE

Approximate the posterior of Z (and estimate θ, ψ, ϕ) using amortized VI

$$P(Z|X,Y) \approx q_\phi(Z|Y) = \prod_{n=1}^{N} \mathcal{N}\left(\mathbf{z}_n \mid \boldsymbol{\mu}_\phi(\mathbf{y}_n), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{y}_n))\right)$$

where $\boldsymbol{\mu}_\phi$ and $\boldsymbol{\sigma}^2_\phi$ are encoder neural networks

ELBO objective:

$$\log p(Y \mid X) \geq \mathcal{L}(\phi,\psi,\theta;Y,X) \triangleq \underbrace{\mathbb{E}_{q_\phi(Z|Y)}\left[\log p_\psi(Y \mid Z)\right]}_{\text{reconstruction: easy}} - \underbrace{D_{\mathbb{KL}}(q_\phi(Z \mid Y) \,||\, \overbrace{p_\theta(Z|X)}^{\text{GP prior}})}_{\text{regularization: closed form, but slow}}$$

Theoretical results from (Ramchandran et al., 2021)

- A novel and provably tighter evidence lower bound for longitudinal GPs $\rightarrow$ upper bound for the KL
- A novel mini-batch compatible KL upper bound

The Health MNIST experiment

- Simulated longitudinal data using MNIST
- Train: $P = 1000$ "patients" ($N = 20000$)
- $Q = 6$ covariates: *id*, *age*, *diseasePresence*, *diseaseAge*, *sex*, and *location*
- Test: 100 "patients":
  - Given time points $[-10, \ldots, -6]$
  - Predict time points $[-5, \ldots, 9]$

The Health MNIST experiment

- Simulated longitudinal data using MNIST
- Train: $P = 1000$ "patients" ($N = 20000$)
- $Q = 6$ covariates: *id*, *age*, *diseasePresence*, *diseaseAge*, *sex*, and *location*
- Test: 100 "patients":
  - Given time points $[-10, \ldots, -6]$
  - Predict time points $[-5, \ldots, 9]$

Model	Latent dimension	MSE
GPPVAE	64	0.057 ± 0.003
GP-VAE	64	0.059 ± 0.002
VRNN	64	0.049 ± 0.004
BRITS	N/A	0.047 ± 0.004
GRUI-GAN	64	0.053 ± 0.007
L-VAE	8	0.038 ± 0.003
L-VAE	16	0.033 ± 0.0018
L-VAE	**32**	$\mathbf{0.025 \pm 0.0015}$

References

- Casale et al., Gaussian process prior variational autoencoders, Advances in Neural Information Processing Systems (NeurIPS), 2018.
- Chen et al., Neural ordinary differential equations, Advances in Neural Information Processing Systems (NeurIPS), 2018.
- Iakovlev et al., Learning continuous-time PDEs from sparse data with graph neural networks. International Conference on Learning Representations (ICLR), 2021.
- Iakovlev et al., Latent neural ODEs with sparse Bayesian multiple shooting, The Eleventh International Conference on Learning Representations (ICLR), 2023.
- Iakovlev et al., Learning space-time continuous neural PDEs from partially observed states, Proceedings of Neural Information Processing Systems (NeurIPS), 2023.
- Ramchandran S, Longitudinal variational autoencoder, International Conference on Artificial Intelligence and Statistics (AISTATS), 2021.
- Rubinova et al., Latent ordinary differential equations for irregularly-sampled time series, Advances in Neural Information Processing Systems (NeurIPS), 2019.
- Yildiz et al., ODE2VAE: Deep generative second order ODEs with Bayesian neural networks, Advances in Neural Information Processing Systems (NeurIPS), 2019.