Deep Generative Models

Lecture 4

Roman Isachenko

AI Masters

2024, Summer

Recap of previous lecture

Flow log-likelihood

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})) + \log |\det(\mathbf{J}_{\mathbf{f}})|$$

The main challenge is a determinant of the Jacobian.

Linear flows

$$\mathbf{z} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{W}\mathbf{x}, \quad \mathbf{W} \in \mathbb{R}^{m \times m}, \quad \boldsymbol{\theta} = \mathbf{W}, \quad \mathbf{J}_{\mathbf{f}} = \mathbf{W}^T$$

- LU-decomposition
$$\mathbf{W} = \mathbf{P}\mathbf{L}\mathbf{U}.$$
- QR-decomposition
$$\mathbf{W} = \mathbf{Q}\mathbf{R}.$$

Decomposition should be done only once in the beggining. Next, we fit decomposed matrices ($\mathbf{P}/\mathbf{L}/\mathbf{U}$ or $\mathbf{Q}/\mathbf{R}$).

Kingma D. P., Dhariwal P. Glow: Generative Flow with Invertible 1x1 Convolutions, 2018

Hoogeboom E., et al. Emerging convolutions for generative normalizing flows, 2019

Recap of previous lecture

Consider an autoregressive model

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{j=1}^{m} p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}), \quad p(x_j|\mathbf{x}_{1:j-1}, \boldsymbol{\theta}) = \mathcal{N}\left(\mu_j(\mathbf{x}_{1:j-1}), \sigma_j^2(\mathbf{x}_{1:j-1})\right).$$

Gaussian autoregressive NF

$$\mathbf{x} = \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{z}) \quad \Rightarrow \quad x_j = \sigma_j(\mathbf{x}_{1:j-1}) \cdot z_j + \mu_j(\mathbf{x}_{1:j-1}).$$

$$\mathbf{z} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}) \quad \Rightarrow \quad z_j = (x_j - \mu_j(\mathbf{x}_{1:j-1})) \cdot \frac{1}{\sigma_j(\mathbf{x}_{1:j-1})}.$$

- We have an **invertible** and **differentiable** transformation from $p(\mathbf{z})$ to $p(\mathbf{x}|\boldsymbol{\theta})$.
- Jacobian of such transformation is triangular!

Generation function $\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{z})$ is **sequential**.
Inference function $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})$ is **not sequential**.

Papamakarios G., Pavlakou T., Murray I. Masked Autoregressive Flow for Density Estimation, 2017

Recap of previous lecture

Let split $\mathbf{x}$ and $\mathbf{z}$ in two parts:

$$\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2] = [\mathbf{x}_{1:d}, \mathbf{x}_{d+1:m}]; \quad \mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2] = [\mathbf{z}_{1:d}, \mathbf{z}_{d+1:m}].$$

Coupling layer

$$\begin{cases} \mathbf{x}_1 = \mathbf{z}_1; \\ \mathbf{x}_2 = \mathbf{z}_2 \odot \boldsymbol{\sigma}_{\boldsymbol{\theta}}(\mathbf{z}_1) + \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_1). \end{cases} \quad \begin{cases} \mathbf{z}_1 = \mathbf{x}_1; \\ \mathbf{z}_2 = (\mathbf{x}_2 - \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_1)) \odot \frac{1}{\boldsymbol{\sigma}_{\boldsymbol{\theta}}(\mathbf{x}_1)}. \end{cases}$$

Estimating the density takes 1 pass, sampling takes 1 pass!

Jacobian

$$\det \left(\frac{\partial \mathbf{z}}{\partial \mathbf{x}} \right) = \det \begin{pmatrix} \mathbf{I}_d & 0_{d \times m-d} \\ \frac{\partial \mathbf{z}_2}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{z}_2}{\partial \mathbf{x}_2} \end{pmatrix} = \prod_{j=1}^{m-d} \frac{1}{\sigma_j(\mathbf{x}_1)}.$$

Coupling layer is a special case of autoregressive NF.

Dinh L., Sohl-Dickstein J., Bengio S. Density estimation using Real NVP, 2016

Recap of previous lecture

Continuous-in-time dynamic (neural ODE)

$$\frac{d\mathbf{z}(t)}{dt} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t); \quad \text{with initial condition } \mathbf{z}(t_0) = \mathbf{z}_0.$$

$$\mathbf{z}(t_1) = \int_{t_0}^{t_1} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t) dt + \mathbf{z}_0 \approx \text{ODESolve}(\mathbf{z}(t_0), \mathbf{f}_{\boldsymbol{\theta}}, t_0, t_1).$$

Euler update step

$$\frac{\mathbf{z}(t + \Delta t) - \mathbf{z}(t)}{\Delta t} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t) \;\Rightarrow\; \mathbf{z}(t+\Delta t) = \mathbf{z}(t) + \Delta t \cdot \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)$$

Theorem (Picard)

If $\mathbf{f}$ is uniformly Lipschitz continuous in $\mathbf{z}$ and continuous in t, then the ODE has a **unique** solution.

$$\mathbf{x} = \mathbf{z}(t_1) = \mathbf{z}(t_0) + \int_{t_0}^{t_1} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t) dt$$

$$\mathbf{z} = \mathbf{z}(t_0) = \mathbf{z}(t_1) + \int_{t_1}^{t_0} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t) dt$$

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Recap of previous lecture

Theorem (Kolmogorov-Fokker-Planck: special case)

If $\mathbf{f}$ is uniformly Lipschitz continuous in $\mathbf{z}$ and continuous in t, then

$$\frac{d \log p(\mathbf{z}(t), t)}{dt} = -\text{tr} \left(\frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right).$$

$$\log p(\mathbf{z}(t_1), t_1) = \log p(\mathbf{z}(t_0), t_0) - \int_{t_0}^{t_1} \text{tr} \left(\frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \right) dt.$$

- **Discrete-in-time NF**: evaluation of determinant of the Jacobian costs $O(m^3)$ (we need invertible $\mathbf{f}$).
- **Continuous-in-time NF**: getting the trace of the Jacobian costs $O(m^2)$ (we need smooth $\mathbf{f}$).

Hutchinson's trace estimator

$$\log p(\mathbf{z}(t_1)) = \log p(\mathbf{z}(t_0)) - \mathbb{E}_{p(\boldsymbol{\epsilon})} \int_{t_0}^{t_1} \left[\boldsymbol{\epsilon}^T \frac{\partial \mathbf{f}}{\partial \mathbf{z}} \boldsymbol{\epsilon} \right] dt.$$

Grathwohl W. et al. FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models, 2018

Outline

1. Continuous-in-time NF: adjoint method
2. Latent variable models (LVM)
3. Variational lower bound (ELBO)

Outline

1. Continuous-in-time NF: adjoint method
2. Latent variable models (LVM)
3. Variational lower bound (ELBO)

Neural ODE

Continuous-in-time NF

$$\frac{d\mathbf{z}(t)}{dt} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t) \qquad \frac{d \log p(\mathbf{z}(t), t)}{dt} = -\mathrm{tr}\left(\frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)}\right)$$

$$\mathbf{x} = \mathbf{z} + \int_{t_0}^{t_1} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t) dt \quad \log p(\mathbf{x}|\boldsymbol{\theta}) = \log p(\mathbf{z}) - \int_{t_0}^{t_1} \mathrm{tr}\left(\frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)}\right) dt$$

How to get optimal parameters of $\boldsymbol{\theta}$?
For fitting parameters we need gradients. We need the analogue of the backpropagation.

Forward pass (Loss function)

$$\mathbf{z} = \mathbf{x} + \int_{t_1}^{t_0} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t) dt, \quad L(\mathbf{z}) = -\log p(\mathbf{x}|\boldsymbol{\theta})$$

$$L(\mathbf{z}) = -\log p(\mathbf{z}) + \int_{t_0}^{t_1} \mathrm{tr}\left(\frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)}\right) dt$$

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Neural ODE

Adjoint functions

$$\mathbf{a}_{\mathbf{z}}(t) = \frac{\partial L}{\partial \mathbf{z}(t)}; \quad \mathbf{a}_{\boldsymbol{\theta}}(t) = \frac{\partial L}{\partial \boldsymbol{\theta}(t)}.$$

These functions show how the gradient of the loss depends on the hidden state $\mathbf{z}(t)$ and parameters $\boldsymbol{\theta}$.

Theorem (Pontryagin)

$$\frac{d\mathbf{a}_{\mathbf{z}}(t)}{dt} = -\mathbf{a}_{\mathbf{z}}(t)^T \cdot \frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}}; \quad \frac{d\mathbf{a}_{\boldsymbol{\theta}}(t)}{dt} = -\mathbf{a}_{\mathbf{z}}(t)^T \cdot \frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \boldsymbol{\theta}}.$$

Solution for adjoint function

$$\frac{\partial L}{\partial \boldsymbol{\theta}(t_1)} = \mathbf{a}_{\boldsymbol{\theta}}(t_1) = -\int_{t_0}^{t_1} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \boldsymbol{\theta}(t)} dt + 0$$
$$\frac{\partial L}{\partial \mathbf{z}(t_1)} = \mathbf{a}_{\mathbf{z}}(t_1) = -\int_{t_0}^{t_1} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(t_0)}$$

Note: These equations are solved in reverse time direction.

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Adjoint method

Forward pass

$$\mathbf{z} = \mathbf{z}(t_0) = \int_{t_0}^{t_1} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)dt + \mathbf{x} \quad \Rightarrow \quad \text{ODE Solver}$$

Backward pass

$$\left.\begin{aligned}
\frac{\partial L}{\partial \boldsymbol{\theta}(t_1)} &= \mathbf{a}_{\boldsymbol{\theta}}(t_1) = -\int_{t_0}^{t_1} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \boldsymbol{\theta}(t)} dt + 0 \\
\frac{\partial L}{\partial \mathbf{z}(t_1)} &= \mathbf{a}_{\mathbf{z}}(t_1) = -\int_{t_0}^{t_1} \mathbf{a}_{\mathbf{z}}(t)^T \frac{\partial \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} dt + \frac{\partial L}{\partial \mathbf{z}(t_0)} \\
\mathbf{z}(t_1) &= -\int_{t_1}^{t_0} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{z}(t), t)dt + \mathbf{z}_0.
\end{aligned}\right\} \Rightarrow \text{ODE Solver}$$

Note: These scary formulas are the standard backprop in the discrete case.

Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018

Outline

1. Continuous-in-time NF: adjoint method
2. Latent variable models (LVM)
3. Variational lower bound (ELBO)

Bayesian framework

Bayes theorem

$$p(\mathbf{t}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{t})p(\mathbf{t})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathbf{t})p(\mathbf{t})}{\int p(\mathbf{x}|\mathbf{t})p(\mathbf{t})d\mathbf{t}}$$

- $\mathbf{x}$ – observed variables, $\mathbf{t}$ – unobserved variables (latent variables/parameters);
- $p(\mathbf{x}|\mathbf{t})$ – likelihood;
- $p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{t})p(\mathbf{t})d\mathbf{t}$ – evidence;
- $p(\mathbf{t})$ – prior distribution, $p(\mathbf{t}|\mathbf{x})$ – posterior distribution.

Meaning

We have unobserved variables $\mathbf{t}$ and some prior knowledge about them $p(\mathbf{t})$. Then, the data $\mathbf{x}$ has been observed. Posterior distribution $p(\mathbf{t}|\mathbf{x})$ summarizes the knowledge after the observations.

Bayesian framework

Let consider the case, where the unobserved variables $\mathbf{t}$ is our model parameters $\boldsymbol{\theta}$.

- $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n$ – observed samples;
- $p(\boldsymbol{\theta})$ – prior parameters distribution (we treat model parameters $\boldsymbol{\theta}$ as random variables).

Posterior distribution

$$p(\boldsymbol{\theta}|\mathbf{X}) = \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{X})} = \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

If evidence $p(\mathbf{X})$ is intractable (due to multidimensional integration), we can't get posterior distribution and perform the exact inference.

Maximum a posteriori (MAP) estimation

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{X}) = \arg\max_{\boldsymbol{\theta}} \bigl(\log p(\mathbf{X}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})\bigr)$$

Latent variable models (LVM)

MLE problem

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{X}|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \prod_{i=1}^{n} p(\mathbf{x}_i|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log p(\mathbf{x}_i|\boldsymbol{\theta}).$$

The distribution $p(\mathbf{x}|\boldsymbol{\theta})$ could be very complex and intractable (as well as real distribution $\pi(\mathbf{x})$).

Extended probabilistic model

Introduce latent variable $\mathbf{z}$ for each sample $\mathbf{x}$

$$p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z}); \quad \log p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) + \log p(\mathbf{z}).$$

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} = \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})p(\mathbf{z}) d\mathbf{z}.$$

Motivation

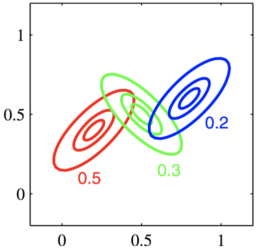
The distributions $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$ and $p(\mathbf{z})$ could be quite simple.

Latent variable models (LVM)

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) p(\mathbf{z}) d\mathbf{z} \to \max_{\boldsymbol{\theta}}$$

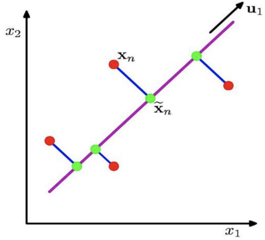
Examples

Mixture of gaussians



- $p(\mathbf{x}|z, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$
- $p(z) = \text{Categorical}(\boldsymbol{\pi})$

PCA model



- $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{W}\mathbf{z} + \boldsymbol{\mu}, \sigma^2 \mathbf{I})$
- $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$

Bishop C. Pattern Recognition and Machine Learning, 2006

Maximum likelihood estimation for LVM

MLE for extended problem

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \prod_{i=1}^{n} p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}) =$$

$$= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}).$$

However, $\mathbf{Z}$ is unknown.

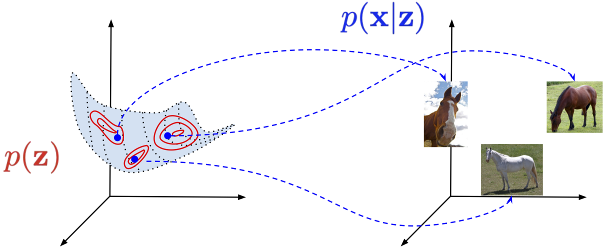
MLE for original problem

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \log p(\mathbf{X}|\boldsymbol{\theta}) = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log p(\mathbf{x}_i|\boldsymbol{\theta}) =$$

$$= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log \int p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}) d\mathbf{z}_i =$$

$$= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \log \int p(\mathbf{x}_i|\mathbf{z}_i, \boldsymbol{\theta}) p(\mathbf{z}_i) d\mathbf{z}_i.$$

Naive approach



Monte-Carlo estimation

$$p(\mathbf{x}|\boldsymbol{\theta}) = \int p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) p(\mathbf{z}) d\mathbf{z} = \mathbb{E}_{p(\mathbf{z})} p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) \approx \frac{1}{K} \sum_{k=1}^{K} p(\mathbf{x}|\mathbf{z}_k, \boldsymbol{\theta}),$$

where $\mathbf{z}_k \sim p(\mathbf{z})$.

Challenge: to cover the space properly, the number of samples grows exponentially with respect to dimensionality of $\mathbf{z}$.

image credit: https://jmtomczak.github.io/blog/4/4_VAE.html

Outline

1. Continuous-in-time NF: adjoint method
2. Latent variable models (LVM)
3. Variational lower bound (ELBO)

Variational lower bound (ELBO)

Derivation 1 (inequality)

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \log \int p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} = \log \int \frac{q(\mathbf{z})}{q(\mathbf{z})} p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) d\mathbf{z} =$$

$$= \log \mathbb{E}_q \left[\frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} \right] \geq \mathbb{E}_q \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} = \mathcal{L}(q, \boldsymbol{\theta})$$

Derivation 2 (equality)

$$\mathcal{L}(q, \boldsymbol{\theta}) = \int q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} = \int q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}) p(\mathbf{x}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} =$$

$$= \int q(\mathbf{z}) \log p(\mathbf{x}|\boldsymbol{\theta}) d\mathbf{z} + \int q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} =$$

$$= \log p(\mathbf{x}|\boldsymbol{\theta}) - KL(q(\mathbf{z}) || p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}))$$

Variational decomposition

$$\log p(\mathbf{x}|\boldsymbol{\theta}) = \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{z}) || p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \geq \mathcal{L}(q, \boldsymbol{\theta}).$$

Variational lower bound (ELBO)

$$
\begin{aligned}
\mathcal{L}(q, \boldsymbol{\theta}) &= \int q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta})}{q(\mathbf{z})} d\mathbf{z} = \\
&= \int q(\mathbf{z}) \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) d\mathbf{z} + \int q(\mathbf{z}) \log \frac{p(\mathbf{z})}{q(\mathbf{z})} d\mathbf{z} \\
&= \mathbb{E}_q \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) - KL(q(\mathbf{z})||p(\mathbf{z}))
\end{aligned}
$$

Log-likelihood decomposition

$$
\begin{aligned}
\log p(\mathbf{x}|\boldsymbol{\theta}) &= \mathcal{L}(q, \boldsymbol{\theta}) + KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})) \\
&= \mathbb{E}_q \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) - KL(q(\mathbf{z})||p(\mathbf{z})) + KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})).
\end{aligned}
$$

- Instead of maximizing incomplete likelihood, maximize ELBO

$$
\max_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta}) \quad \rightarrow \quad \max_{q, \boldsymbol{\theta}} \mathcal{L}(q, \boldsymbol{\theta})
$$

- Maximization of ELBO by **variational** distribution q is equivalent to minimization of KL

$$
\arg\max_{q} \mathcal{L}(q, \boldsymbol{\theta}) \equiv \arg\min_{q} KL(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})).
$$

Summary

- Adjoint method generalizes backpropagation procedure and allows to train Neural ODE solving ODE for adjoint function back in time.
- Bayesian framework is a generalization of most common machine learning tasks.
- LVM introduces latent representation of observed samples to make model more interpretative.
- LVM maximizes variational evidence lower bound (ELBO) to find MLE for the parameters.