Neural networks and Variational Inference

8th BCD ISHEP Cargèse School

27-31 March 2023

Julien Donini – Université Clermont Auvergne / LPC

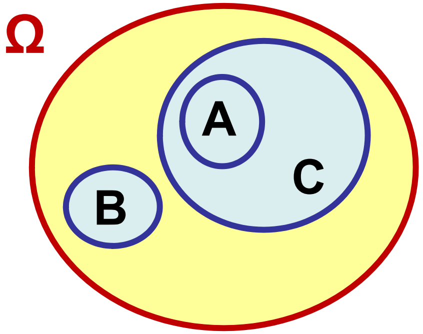
Outline

Part1: Modern ML for HEP

Part 2: Neural Networks and variational inference

- **Bayesian statistics**
- **Variational inference**
- **Autoencoders and variational inference (VAE)**
- **Examples**

Reminder: probabilities



Frequentist: related to **frequency** of occurrence

$$P(A) = \frac{\text{number of time event A occurs}}{\text{number of time experience is repeated}}$$

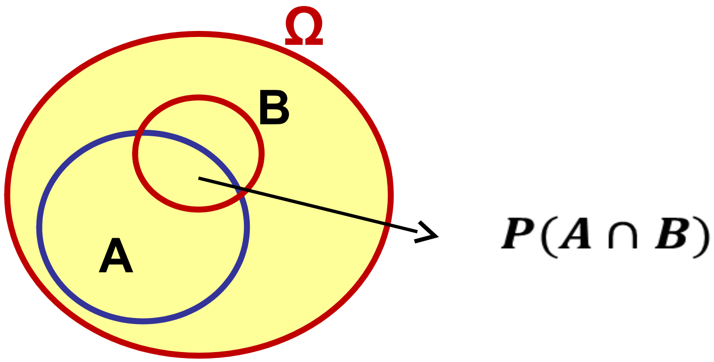
Bayesian: degree of belief that A is true introduces concepts of **prior** and **posterior** probability

$$P(A|\text{data}) \propto P(\text{data}|A) \times P(A)$$

Knowledge on A increases using data

Probability of **A** given that **B is true**:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$



But, similarly: $$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

Hence: $$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$

$$\Leftrightarrow \mathbf{P(A|B) = \frac{P(B|A)P(A)}{P(B)}}$$

Bayes theorem

Thomas Bayes (?)
c. 1701 –1761

An Essay towards solving a Problem in the Doctrine of Chances. By the late Rev. Mr. Bayes, *communicated by Mr.* Price (1763)

"If there be two subsequent events, the probability of the second b/N and the probability of both together P/N, and it being first discovered that the second event has also happened, from hence I guess that the first event has also happened, the probability I am right is P/b."

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

If the sample space Ω can be divided in disjoint subsets A_i

$$P(A|B) = \frac{P(B|A)P(A)}{\sum_i P(B|A_i)P(A_i)}$$

Mandatory coin-flip example

Example: 10 coins, **1** of which is **unfair** (two-sided tail): You flip a random coin and obtain **tail**. What is the probability that this is the unfair coin ?

A: event where the coin is **unfair**, **B**: event where the result is **tail**

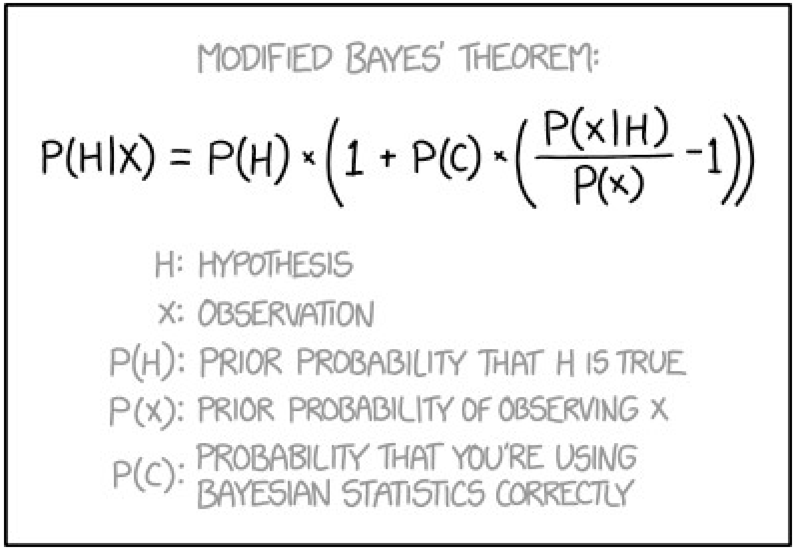
You want **P(A|B)**: $$\boldsymbol{P(A|B) = \frac{P(B|A)P(A)}{P(B)}}$$

where: $$P(B) = P(B \cap A) + P(B \cap \bar{A}) = P(B|A)P(A) + P(B|\bar{A})P(\bar{A})$$

$$P(B|A) = 1, P(A) = \frac{1}{10}$$

$$\boldsymbol{\Rightarrow P(A|B) = \frac{1 \times \frac{1}{10}}{1 \times \frac{1}{10} + \frac{1}{2} \times \frac{9}{10}} = \frac{2}{11}}$$

In **Bayesian** language: P(A) is the **prior** probability and P(A|B) the **posterior**


MODIFIED BAYES' THEOREM:
P(H|X) = P(H) × (1 + P(C) × (P(x|H) / P(x) − 1))
H: HYPOTHESIS
X: OBSERVATION
P(H): PRIOR PROBABILITY THAT H IS TRUE
P(x): PRIOR PROBABILITY OF OBSERVING X
P(C): PROBABILITY THAT YOU'RE USING BAYESIAN STATISTICS CORRECTLY

xkcd.com

Frequentist vs Bayesian approaches

Frequentist

- Probabilities are related to **frequencies** of real or hypothetical events
- True **parameters** of the model: fixed and **unknown**
- **Estimate** parameters (estimator) and uncertainties using **likelihood**

Bayesian

- Improve **prior knowledge** using data and Bayes theorem
- Estimate **probability** of true **parameters**: P(parameter | data)
- Fundamentally contrary to the frequentist philosophy !

***Bayes theorem** is not Bayesian per se, it is its **interpretation** that makes it **Bayesian** !*

Frequentist vs Bayesian approaches

Frequentist

- Probabilities are related to **frequencies** of real or hypothetical events
- True **parameters** of the model: fixed and **unknown**
- **Estimate** parameters (estimator) and uncertainties using **likelihood**

Bayesian

- Improve **prior knowledge** using data and Bayes theorem
- Estimate **probability** of true **parameters**: P(parameter | data)
- Fundamentally contrary to the frequentist philosophy !

*In **simple** problems, the two approaches can yield **similar results**. As data and models grow in **complexity**, however, the two approaches can **diverge greatly**.*

Bayes Theorem and statistical inference

Likelihood of observing these data given a theory

Posterior knowledge on theory

Prior knowledge on theory

$$P(\text{theory}|\text{data}) = \frac{P(\text{data}|\text{theory})P(\text{theory})}{P(\text{data})}$$

Marginal likelihood (a **normalisation** factor)

Example : coin flip (again)

After **n** trials and **k** observation of heads what is the probability **p** of heads ?

$$\begin{cases} p = 0.5 : \text{a fair coin} \\ p \neq 0.5 : \text{a tricky coin !} \end{cases}$$

Let's treat this problem using both **frequentist** and **bayesian** approches

Coin flip: frequentist approach

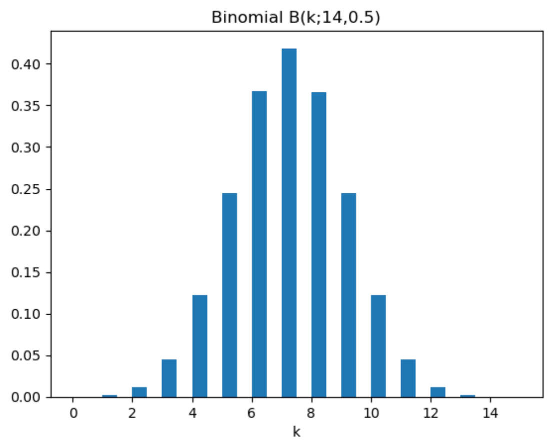
After **n** trials and **k** observation of heads what is the probability **p** of heads ?

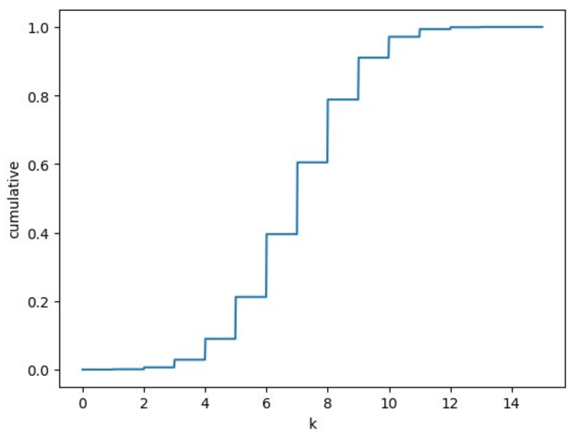
Example: **n=14** trial, **k = 10** 'head' results

For this we can use the **Binomial probability law**

- Estimator of p: $\hat{p} = k/n$
- Standard deviation: $\sigma_p = \sqrt{\dfrac{\hat{p}(1-\hat{p})}{n}}$ → $\hat{p} = 0.71 \pm 0.12$

What is the **compatibility** of the result with the **p=0.5 hypothesis** ?





p-value = 9%

Significance: 1.3σ

In fact: rather compatible

Coin flip: bayesian approach

After **n** trials and **k** observation of heads what is the probability **p** of heads ?

Bayesian inference deduce **probabilistic** statements about the distribution of p.

==> **p is not a value, it's a distribution**

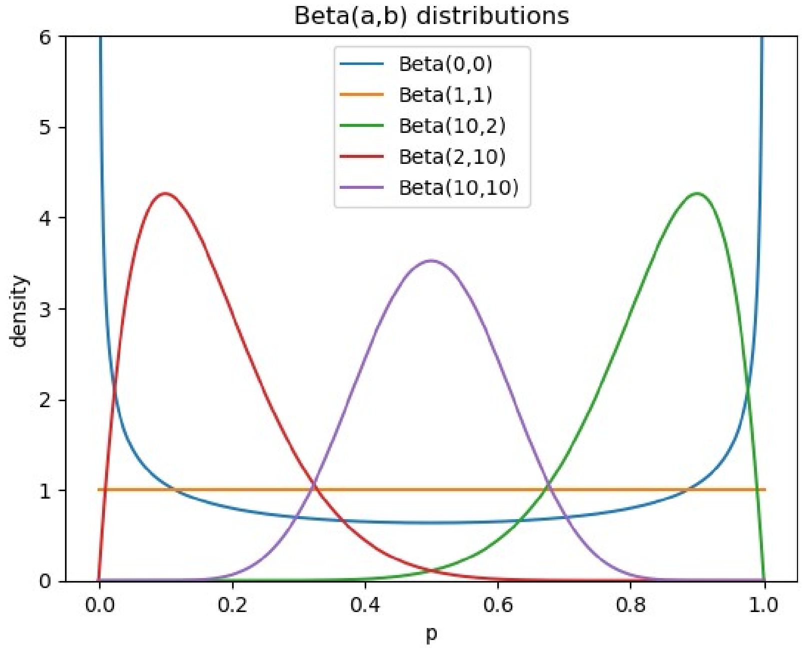
The probability **p**, given the observed **data,** is obtained by **Bayes** theorem:

$$\overbrace{P(p|data)}^{\text{posterior}} = \frac{\overbrace{P(data|p)}^{\text{likelihood}}\,\overbrace{P(p)}^{\text{prior}}}{\underbrace{P(data)}_{\text{marginal likelihood}}}$$

Coin flip: bayesian approach

After **n** trials and **k** observation of heads what is the probability **p** of heads ?

A very convenient **prior** for this scenario is the **Beta distribution** Beta(**a**,**b**)



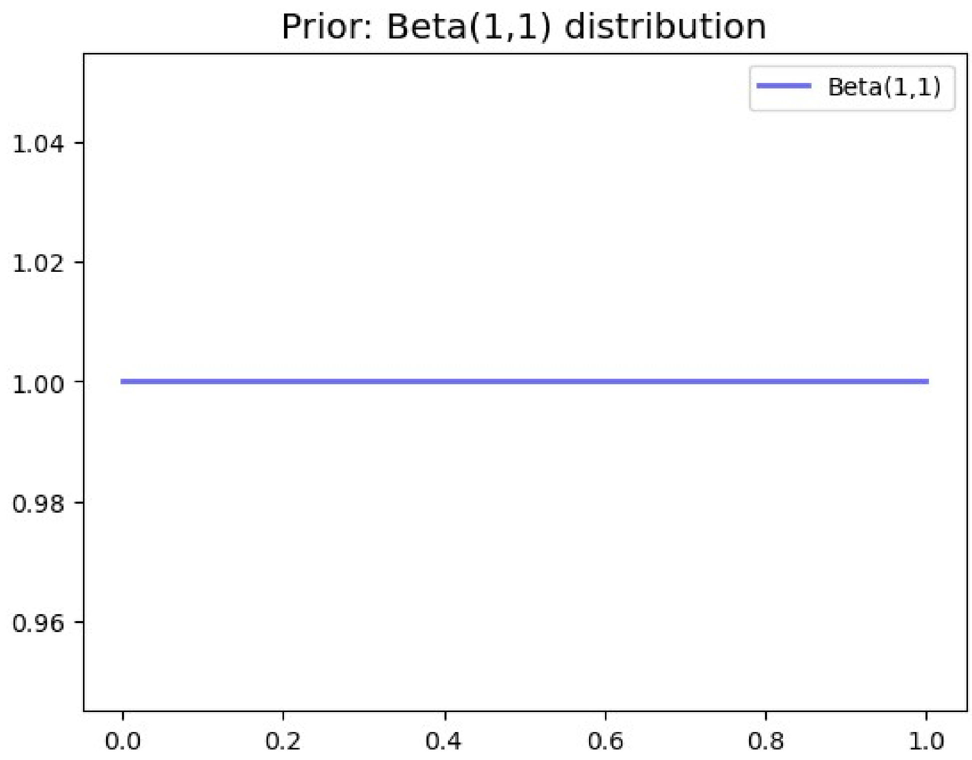
In this case the **posterior distribution** can be calculated analytically :

$$P(p|data) = Beta(p|k+a, n-k+b)$$

Coin flip: bayesian approach

Let’s assume that we know nothing about p = **uniform prior**

This corresponds to the Beta(a,b) distribution with **a=1** and **b=1**



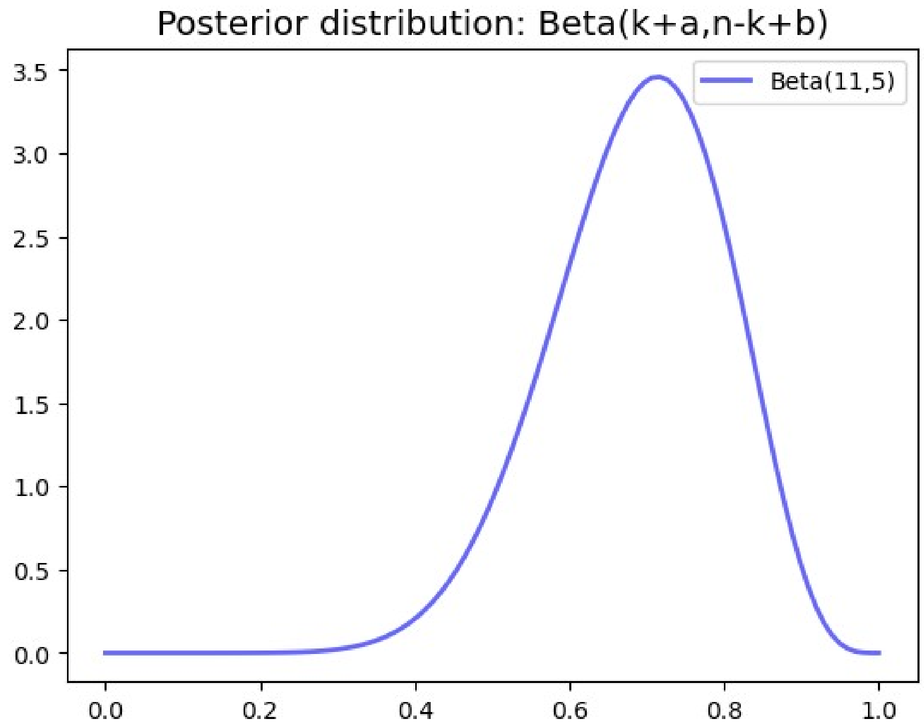
Coin flip: bayesian approach

Example: **n=14** trial, **k = 10** ‘head’ results

$$P(p|data) = Beta(p|k+a, n-k+b)$$

The **posterior** distribution is then

with **a=1** and **b=1**

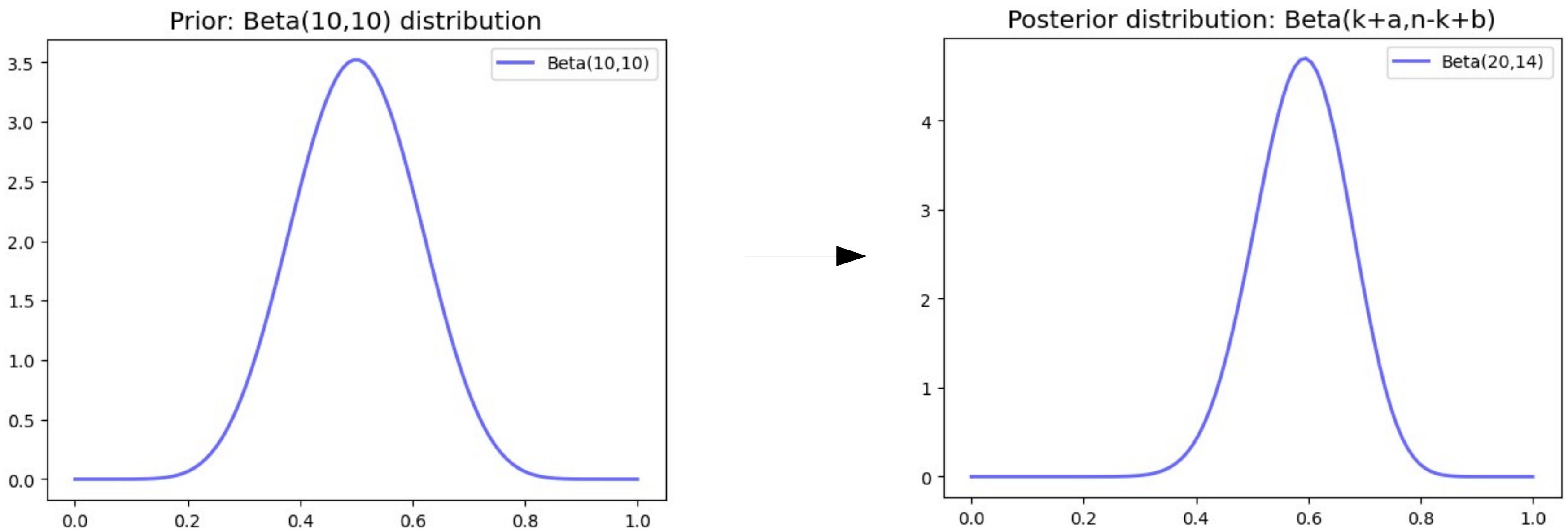


Interval containing 95% of the distribution: (0.45, 0.88)

Coin flip: bayesian approach

Example: **n=14** trial, **k = 10** ‘head’ results

Different prior : centered on 0.5, Beta(a,b) distribution with **a=10** and **b=10**



Interval containing 95% of the distribution: (0.42, 0.75)

Coin flip: Bayes factor

Which statistical model is **better** ? An answer is given by the **Bayes factor**

The Bayes factor is the **ratio** of the **marginal likelihoods** of the two models

Marginal likelihood: $P(data) = \int P(data|p)P(p)dp$

Bayes factor:

$$K = \frac{P(data|\text{Model}_1)}{P(data|\text{Model}_2)} = \frac{\int P(data|p, \text{Model}_1)P(p|\text{Model}_1)dp}{\int P(data|p, \text{Model}_2)P(p|\text{Model}_2)dp}$$

A value of K > 1 means that Model_1 is more strongly supported by the data under consideration than Model_2

Coin flip: Bayes factor

For the binomial model the **Bayes factor** is given by:

$$K = \frac{P(data|\text{Model}_1)}{P(data|\text{Model}_2)} = \frac{B(k+a_1, n-k+b_1)}{B(a_1, b_1)} \times \frac{B(a_2, b_2)}{B(k+a_2, n-k+b_2)}$$

Comparing the ‘peaked’ prior model with the uniform prior model gives K = 1.2

The first model is more supported than the alternative hypothesis by the data.

Exemple: 2D counting experiment

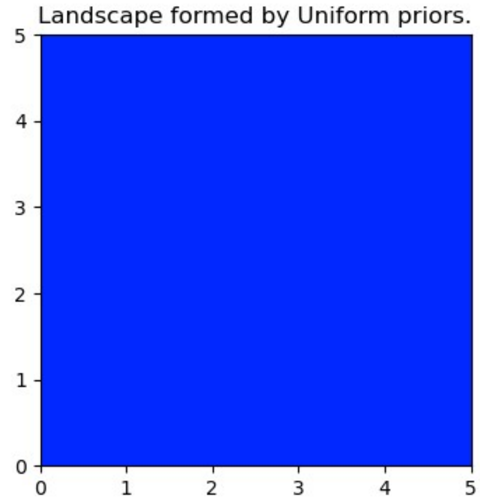
We have two **sensors** that measure events from different processes (decay rate, etc)

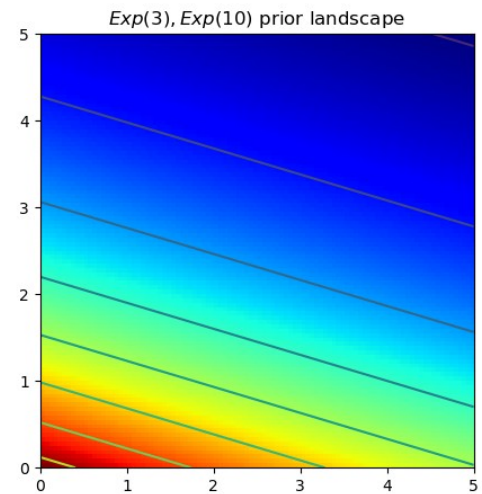
The number of **counts** for each process follows a **Poisson distribution**

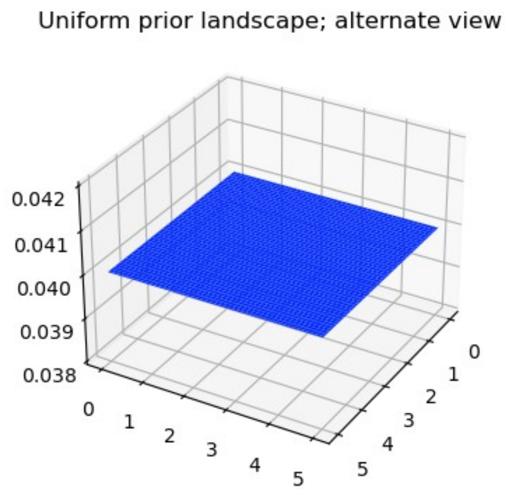
$$P(n_i|\lambda_i) = \frac{e^{-\lambda_i}\lambda_i^{n_i}}{n_i!}, \;\; i = \mathrm{process}\{1, 2\}$$

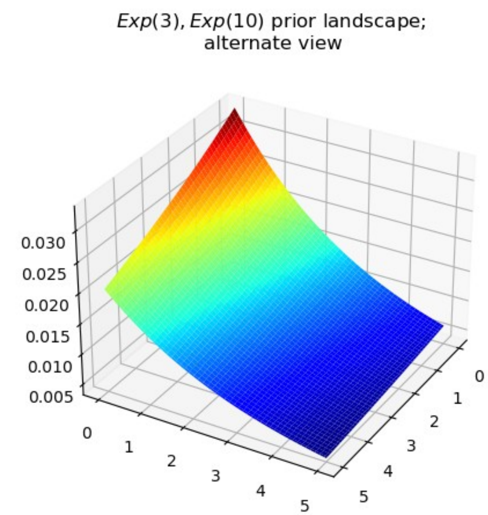
Let's see how the **posterior** distribution $P(\lambda_1, \lambda_2|\mathrm{data})$ evolves with **data**

We assume two different **priors** for $\{\lambda_1, \lambda_2\}$



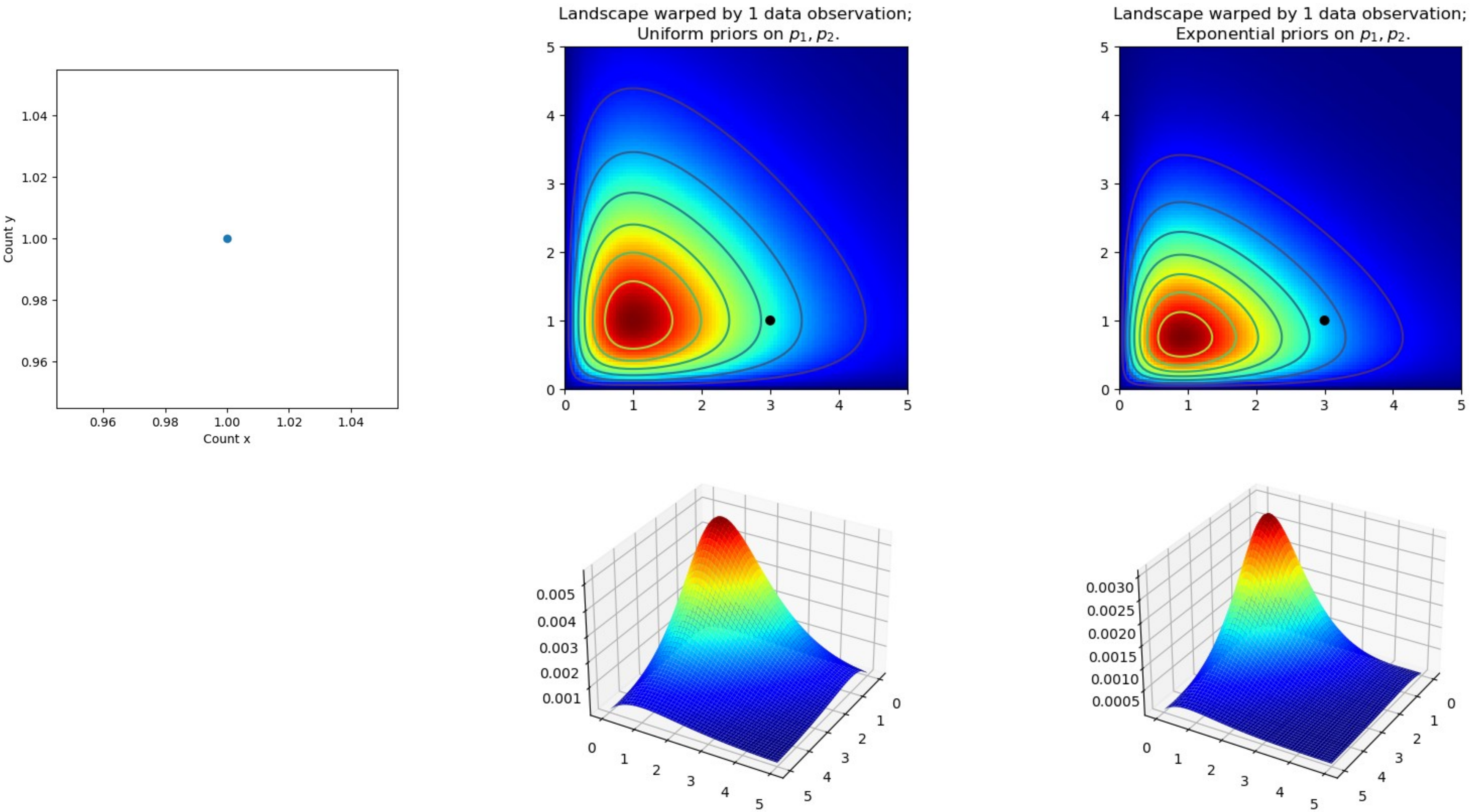






Exemple: 2D counting experiment

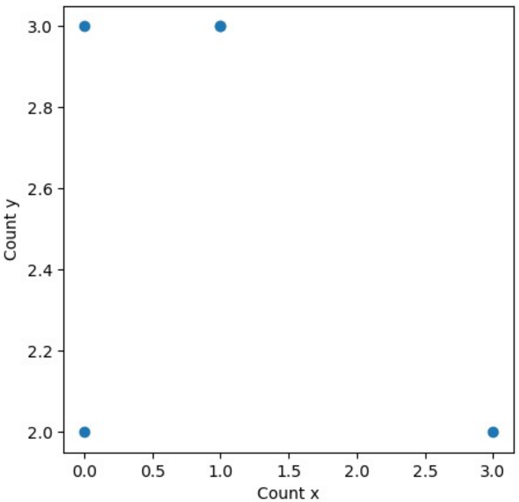
For each data point we update the **posterior**: here for N = 1 data

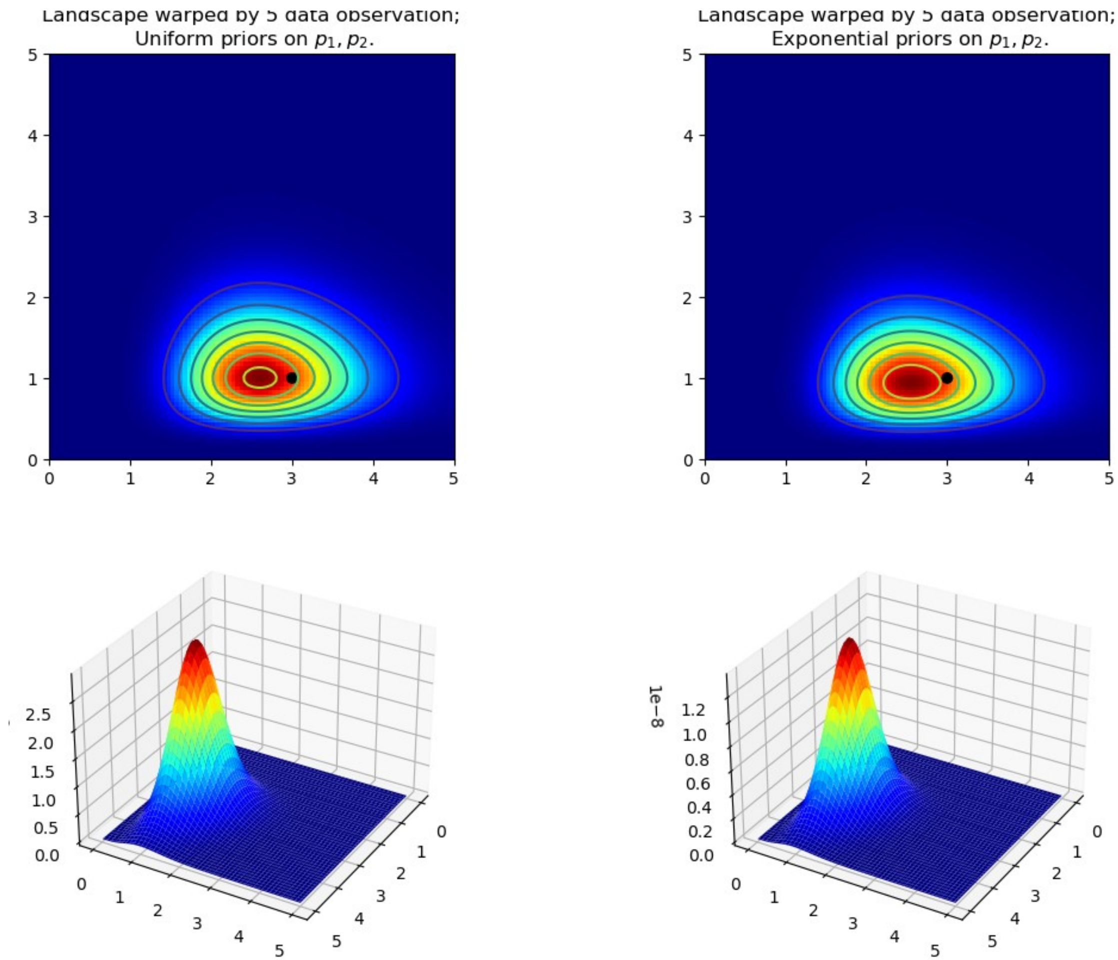


The ‘true’ value used to generate the data corresponds to $\lambda_1 = 3, \lambda_2 = 1$

Exemple: 2D counting experiment

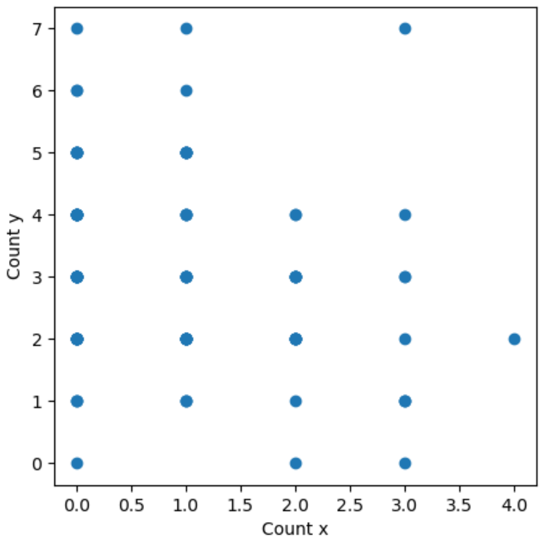
For each data point we update the **posterior**: here for N = 5 data

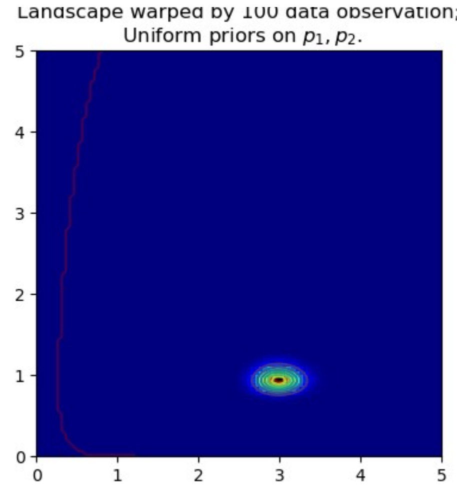


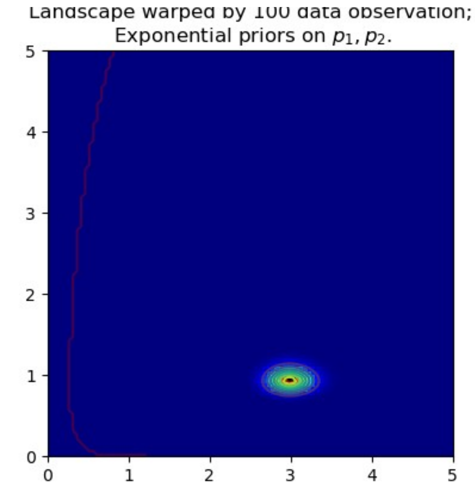


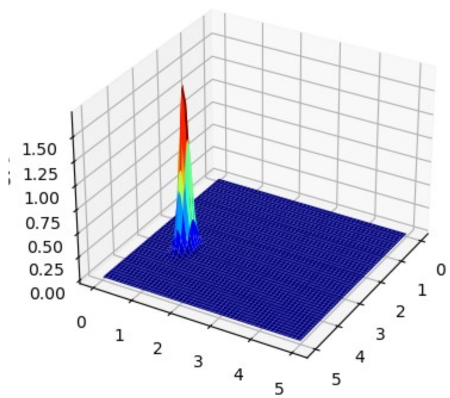
Exemple: 2D counting experiment

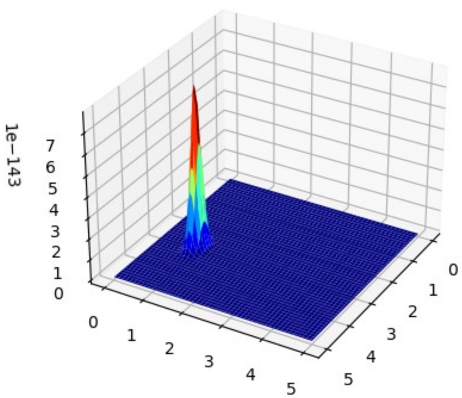
For each data point we update the **posterior**: here for N = 100 data



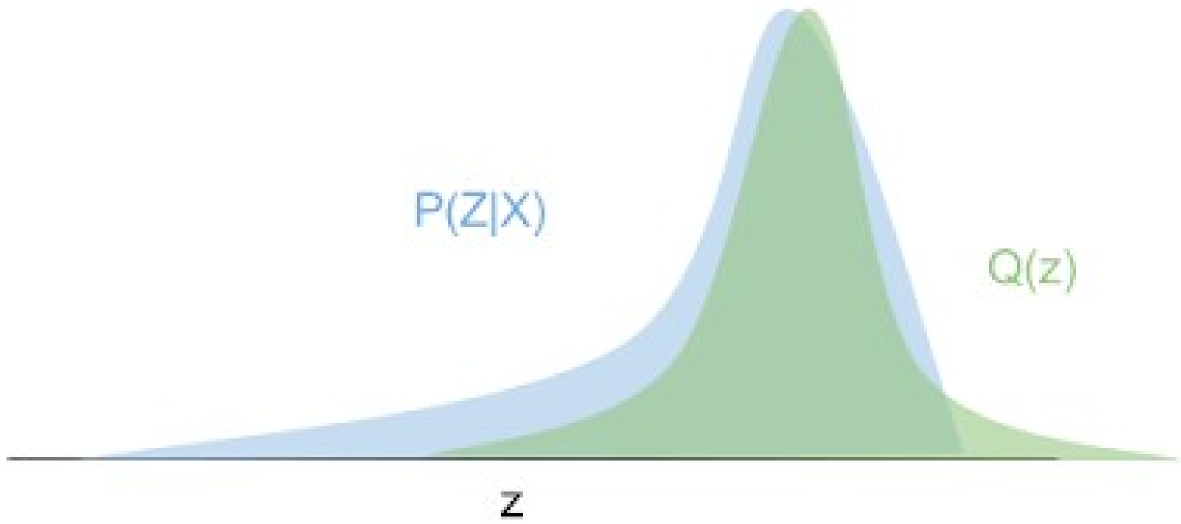








Variational Inference (VI)



Variational Inference (VI)

How can we sample **posterior** densities p(θ|data) efficiently ?

Methods such as Markov Chain Monte Carlo (MCMC) do that but they scale poorly with **data** size and can become inefficient in very **high** dimensions.

Variational inference is an alternative approach: fitting an **approximation**

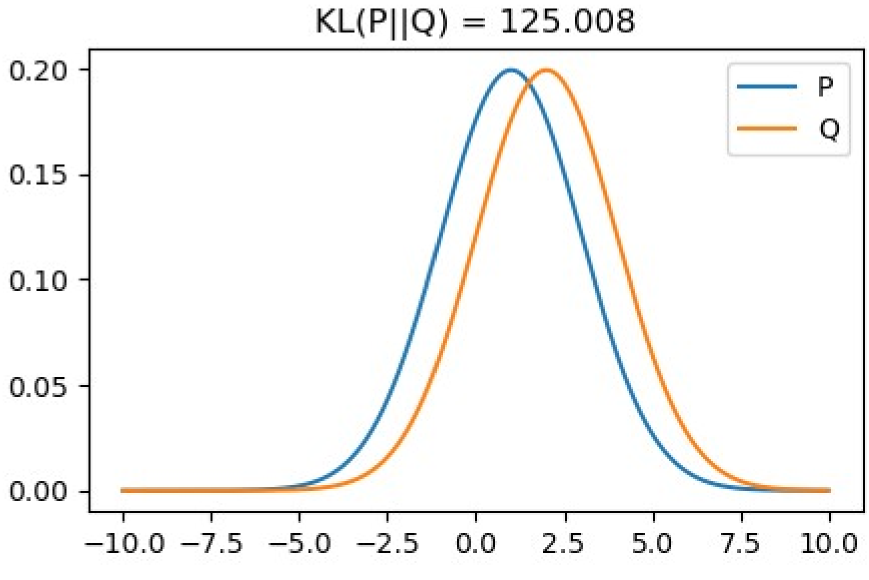
$$q(\theta) \simeq p(\theta|\mathrm{data})$$

with a simple functional form, such as a normal distribution, and casting the inference task as an optimisation problem.

Variational Inference (VI)

Variational inference is based on fitting an approximation q(θ) to the posterior by minimising the **Kullback–Leibler (KL) divergence**

$$D_{\mathrm{KL}}(q(\theta)||p(\theta|\mathrm{data})) = \mathrm{E}_{\theta\sim q(\theta)}\left[\log\left(\frac{q(\theta)}{p(\theta|\mathrm{data})}\right)\right] = \int \log\left(\frac{q(\theta)}{p(\theta|\mathrm{data})}\right) q(\theta) d\theta.$$



ELBO

Approximating q(θ) to the true posterior → **minimizing** the KL divergence.

However this is **difficult** as the **evidence**, or marginal likelihood, p(data) appears in the expression of KL making the calculation in general **intractable**.

In practice the so-called **Evidence Lower-BOund** (ELBO) is used instead:

$$\mathrm{ELBO} = \mathrm{E}_{q(\theta)}\left[\log p(\mathrm{data}, \theta) - \log q(\theta)\right].$$

Indeed **maximizing** the ELBO is equivalent to **minimizing** the KL divergence

Maximizing the ELBO

Maximizing the ELBO is equivalent to minimizing the KL divergence, as:

$$\begin{aligned} \log p(\text{data}) &\geq \text{ELBO} \\ &\geq -D_{\text{KL}}(q(\theta)||p(\theta|\text{data})) + \text{E}_{q(\theta)}\left[\log p(\text{data}|\theta)\right]. \end{aligned}$$

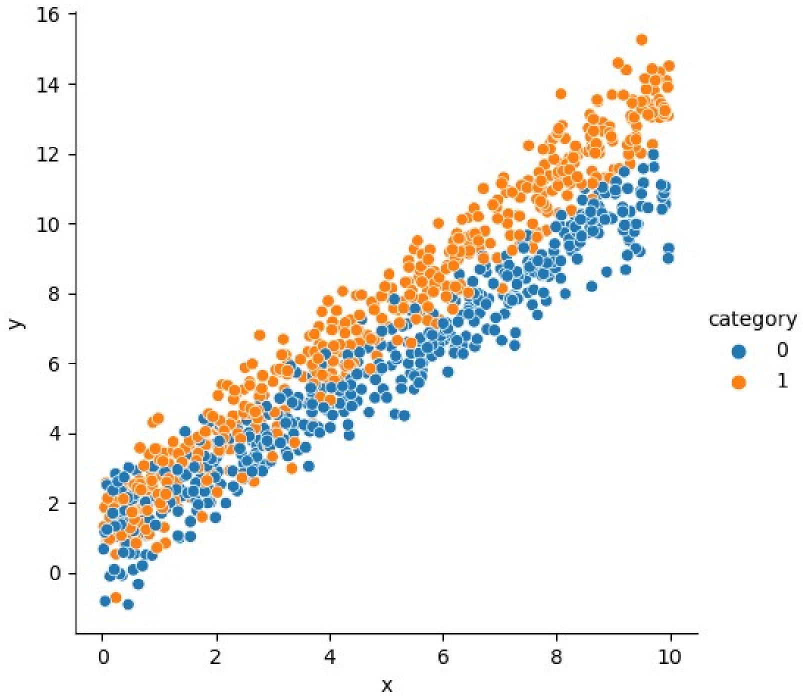
Optimising the ELBO serves a dual purpose:

- q(θ) yields the best **approximation** of the posterior p(θ|data)
- The value provides an approximation (bound) on the marginal likelihood, which can be used for model **comparison**.

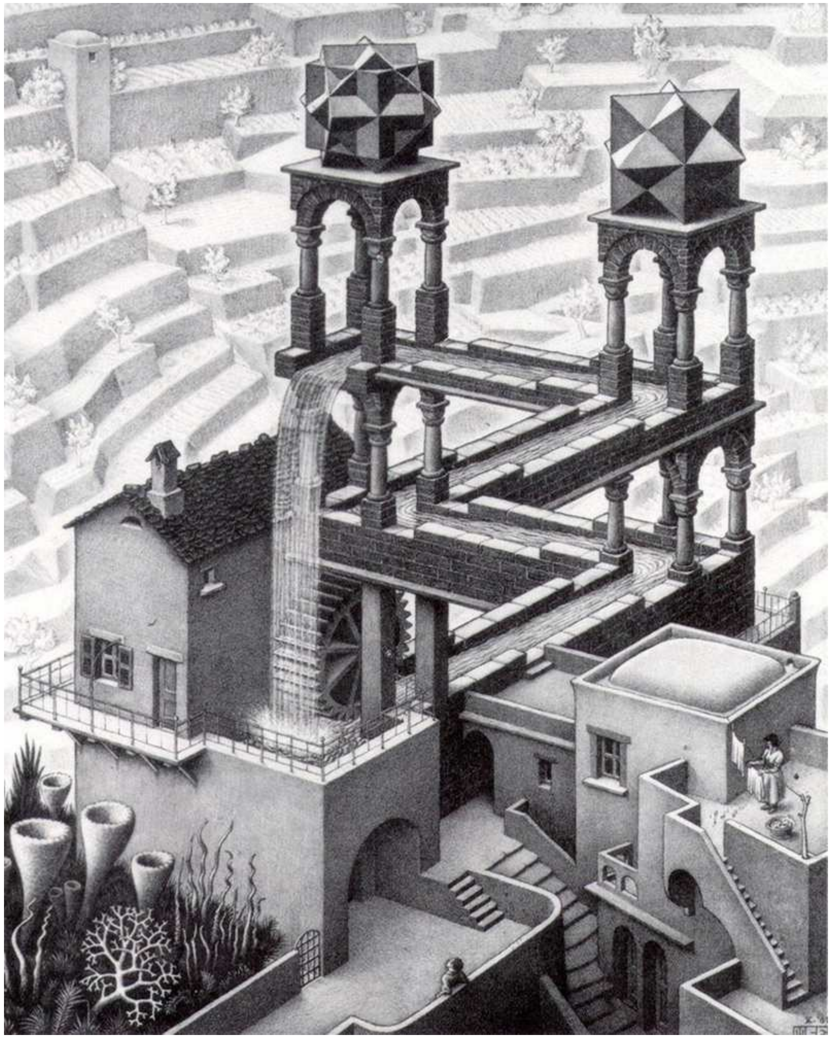
To approximate the **posterior** $p(\theta|\text{data})$, parameters ϕ describing the **density** $q(\theta)$ are optimized using **automatic differentiation:**

→ ADVI: **A**utomatic Differentiation **V**ariational **I**nference
https://arxiv.org/abs/1603.00788

See example notebook:



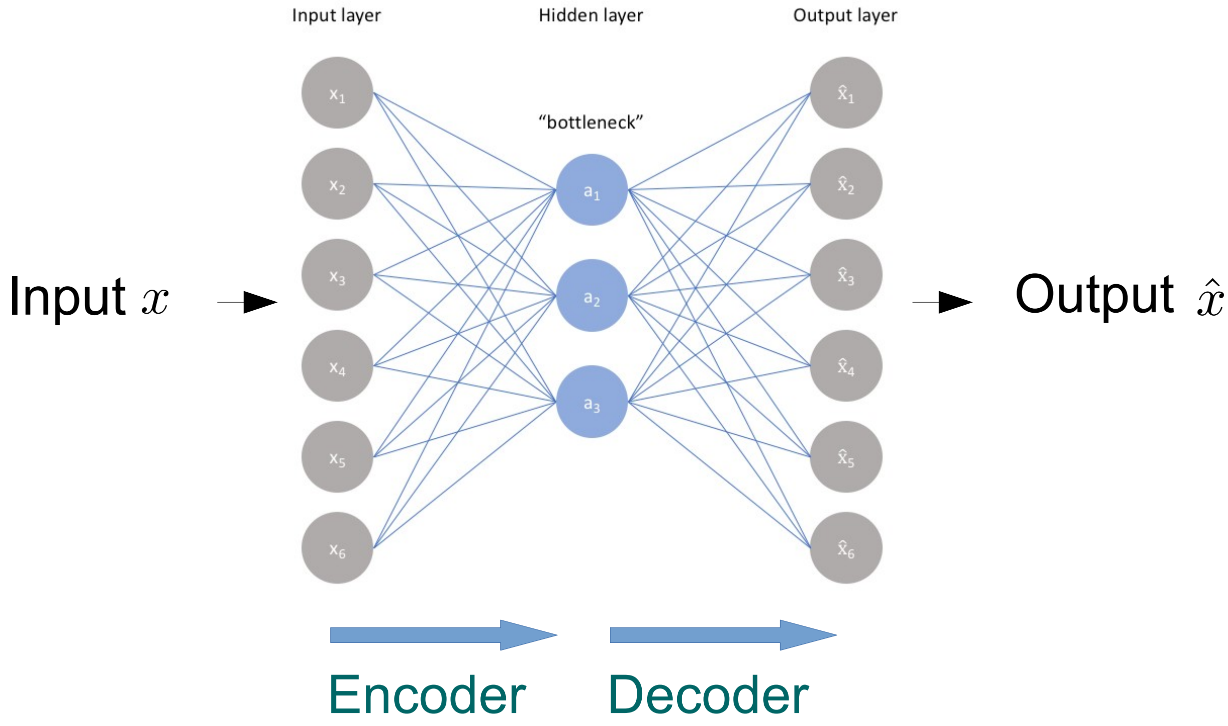
Autoencoders and variational inference



Autoencoders (AE)

NN trained to **reproduce** the input data using a **constrained** network.

Use cases: **anomaly detection, data-compression, data generation**

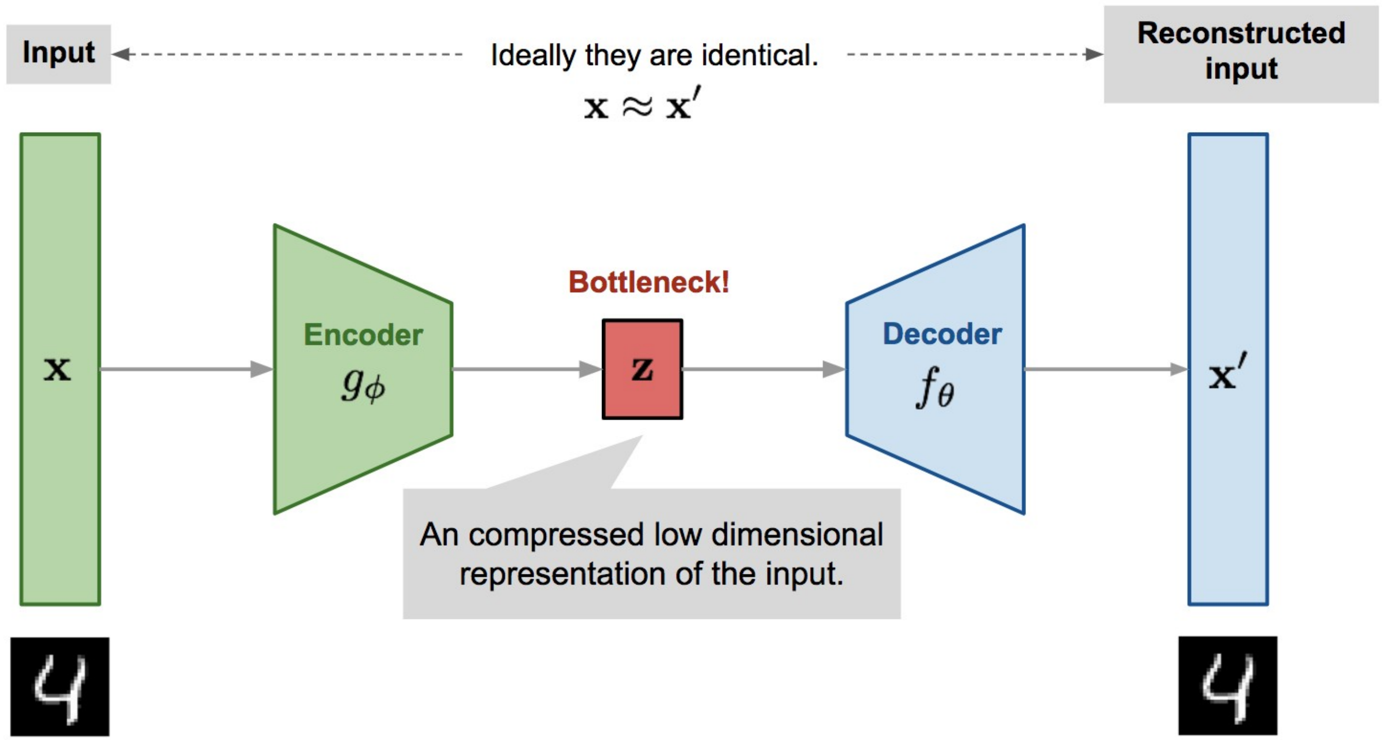


The network is constrained to learn important **data features**.

Mandatory MNIST example

MNIST database : 60,000 training images and 10,000 testing images

[lilianweng.github.io]



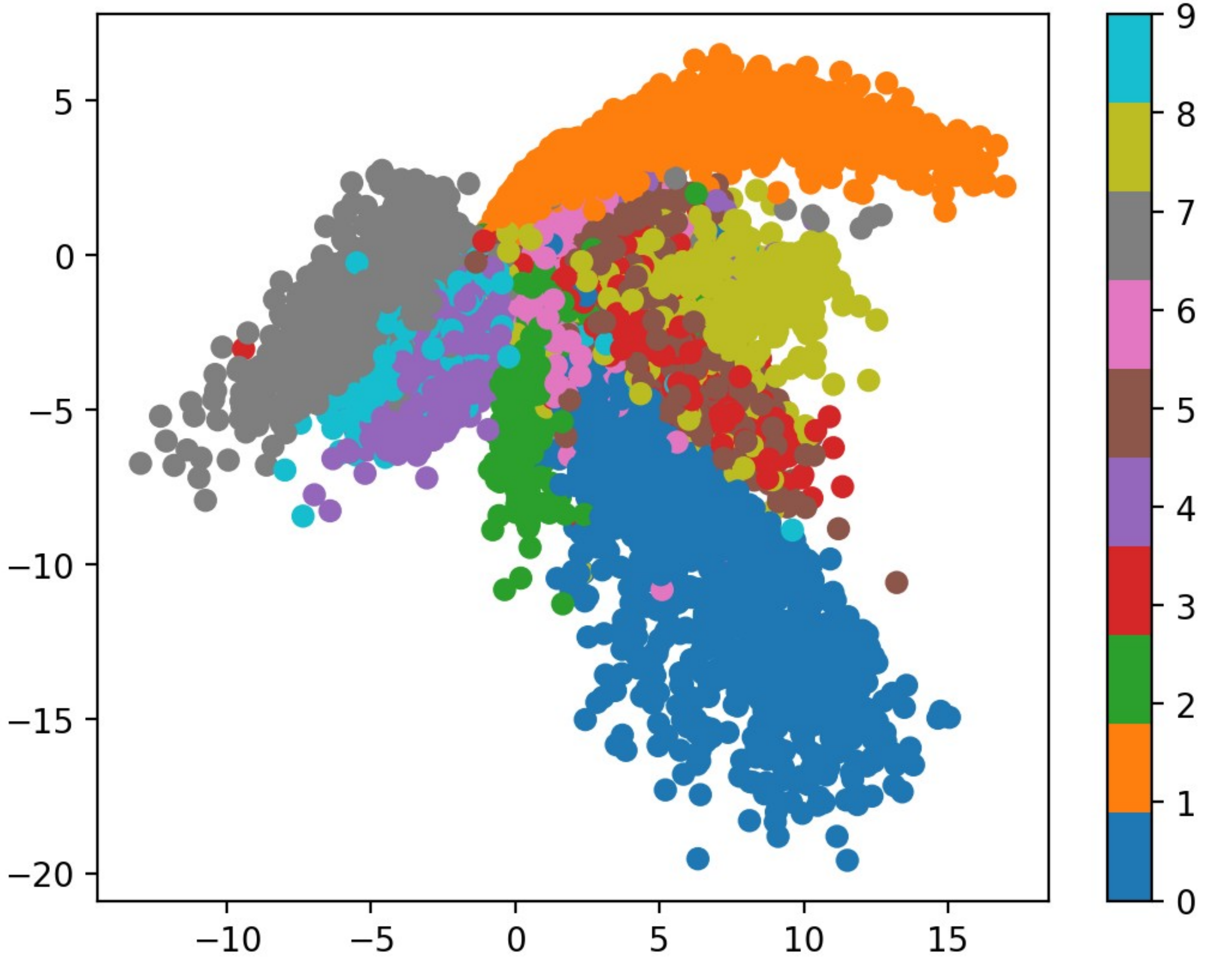
Encoding x to latence space z: $z = g_\phi(x)$

Decoding z to reconstructed space x': $x' = f_\theta(z)$

Training: minimize MSE loss: $\ell(\theta, \phi) = \frac{1}{N} \sum_{\text{batch}} [x_i - f_\theta(g_\phi(x_i))]^2$

Mapping to latence space

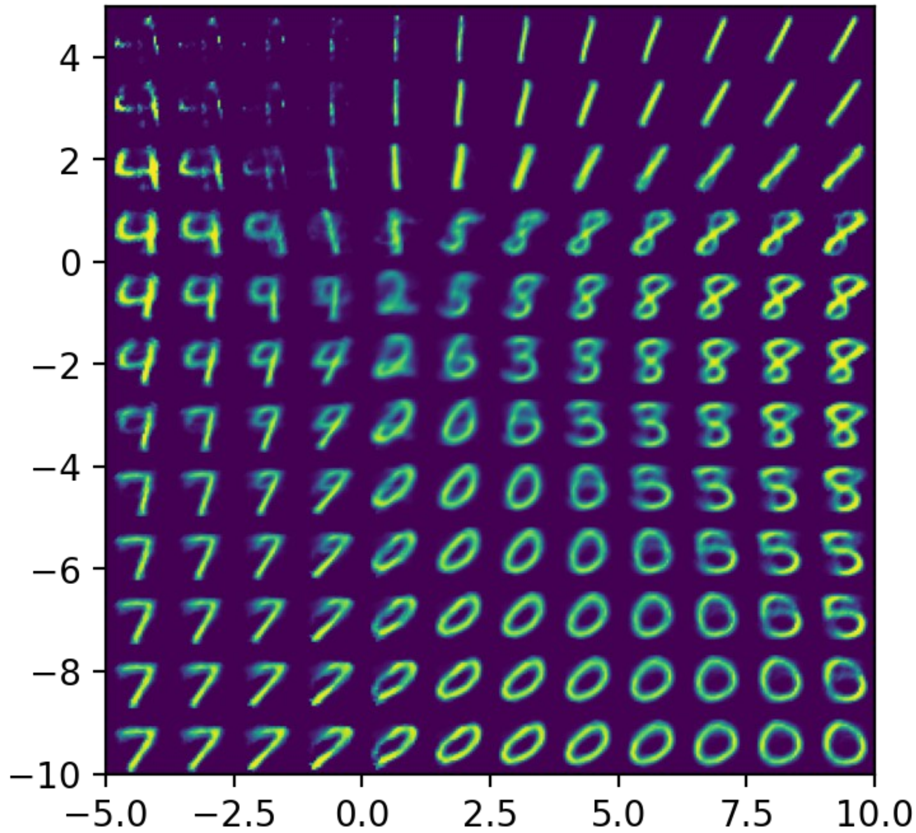
Example for 2-dimension latence space:



[A. Van de Kleut]

Generate new images ?

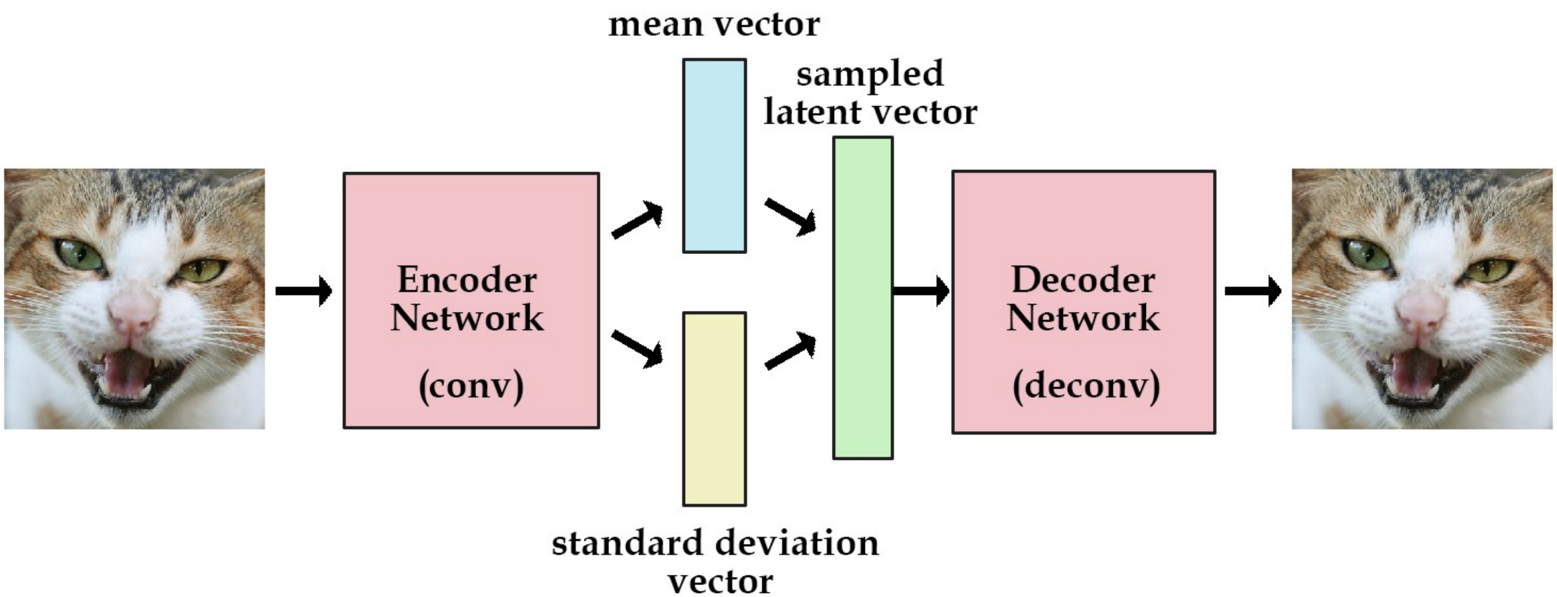
We can **sample** uniformly from the latent space and see how the decoder reconstructs inputs from arbitrary latent vectors.

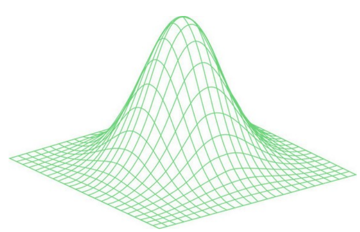


Problems: **gaps** in the latence space, scaling to higher dim will be even worse

Variational Autoencoders (VAE)

VAE [Kingma et al., 1312.6114] are probabilistic (deep) **generative** models.



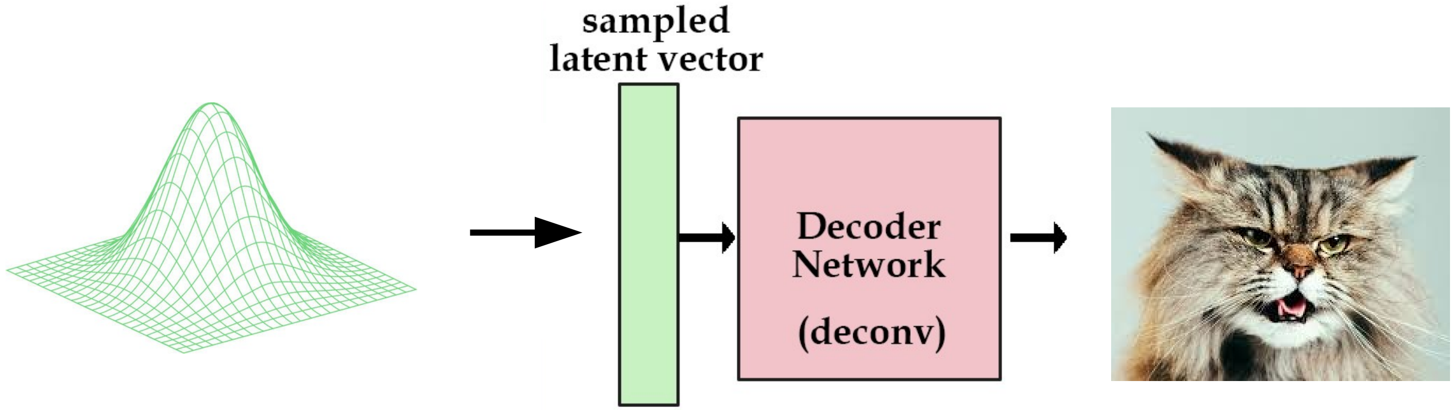


Input data is mapped to a **multidimensional** normal distribution

For more information on VAE see these nice blogs: here, here and here.

Variational Autoencoders (VAE)

Once trained only decoder is kept and **new** images are randomly generated !



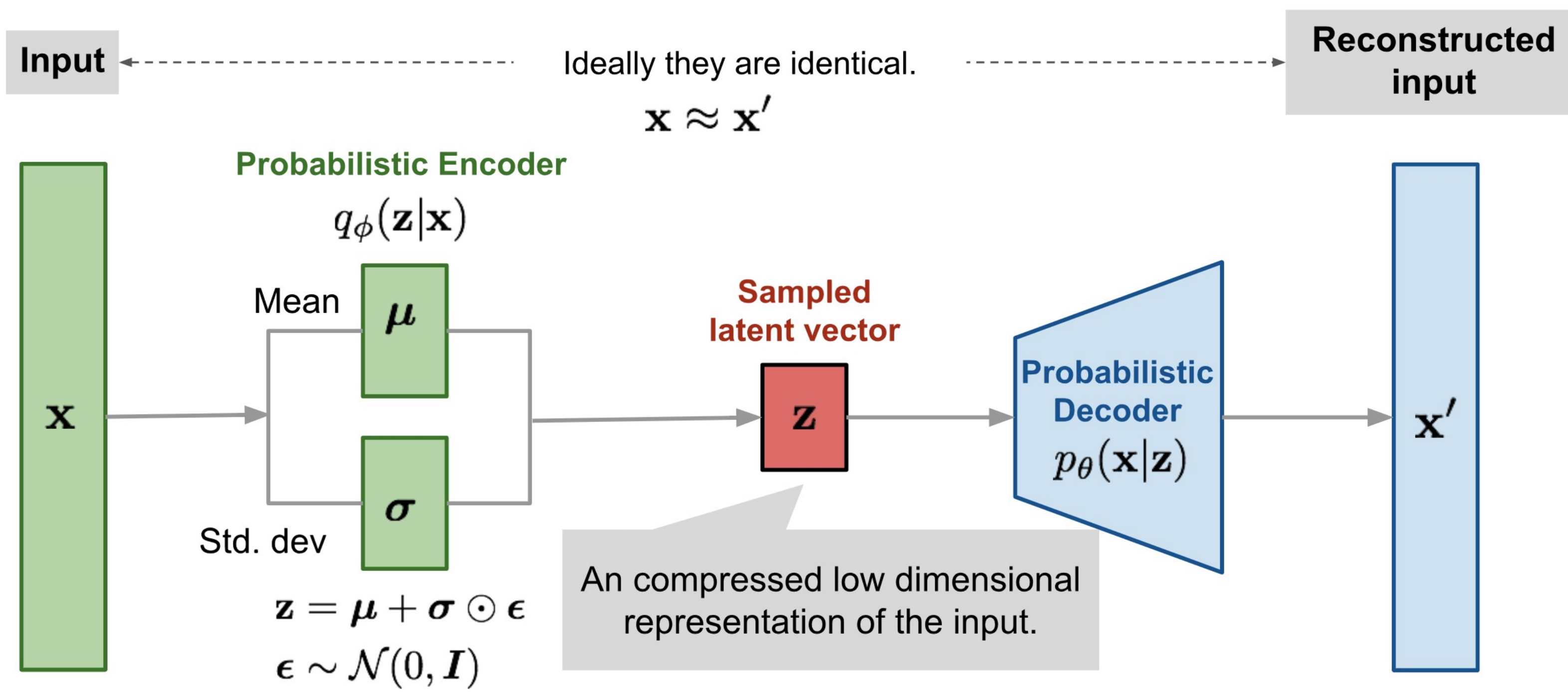
Multidimensional normal
distribution is randomly sampled

Variational Autoencoders (VAE)

Inputs are mapped to a **probability distribution** over latent vectors

Encoding x to latence space z: $g_\phi(z|x) \rightarrow \text{approximated posterior}$

Decoding z to reconstructed space x': $f_\theta(x|z) \rightarrow \text{likelihood}$



The return of the KL divergence

The estimated posterior $g_\phi(z|x)$ should be very close to the real one $q(z|x)$

We use Kullback-Leibler divergence to quantify the distance between these:

$$D_{\mathrm{KL}}(g_\phi(z|x)||q(z|x)) = \mathrm{E}_{z\sim g_\phi(z|x)}\left[\log\left(\frac{g_\phi(z|x)}{q(z|x)}\right)\right]$$

q is specified as a standard normal distribution: $\mathcal{N}(0,1)$

$\mathbf{D_{KL}}$ will penalize g_φ if it differs from q

VAE loss

For **Variational Inference** we have seen (page 30) that the evidence is such:

$$\begin{aligned} \log p(\text{data}) &\geq \text{ELBO} \\ &\geq -D_{\text{KL}}(g_\phi(z|x)||q(z)) + \text{E}_{g_\phi(z|x)}\left[\log p(x|z)\right]. \end{aligned}$$

The right-handed term will constitue the loss of our NN during the training.

For $g_\phi(z|x) = \mathcal{N}(\mu, \sigma)$ and $q(z) = \mathcal{N}(0, 1)$ one can show that the loss is:

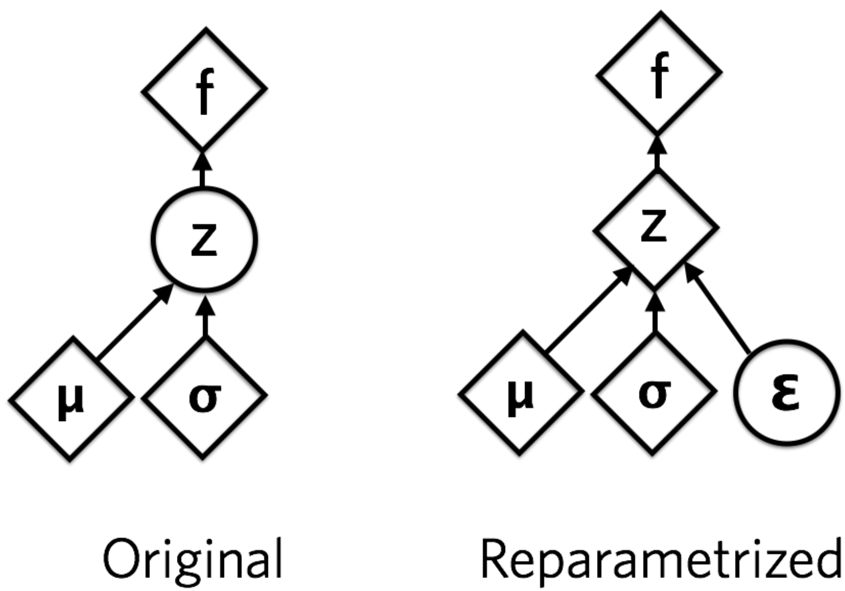
$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \simeq \underbrace{\frac{1}{2}\sum_{j=1}^{J}\left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right)}_{\text{KL regularization}} + \underbrace{\frac{1}{L}\sum_{l=1}^{L}\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)})}_{\text{Likelihood of reconstructed output}}$$

The reparametrization trick

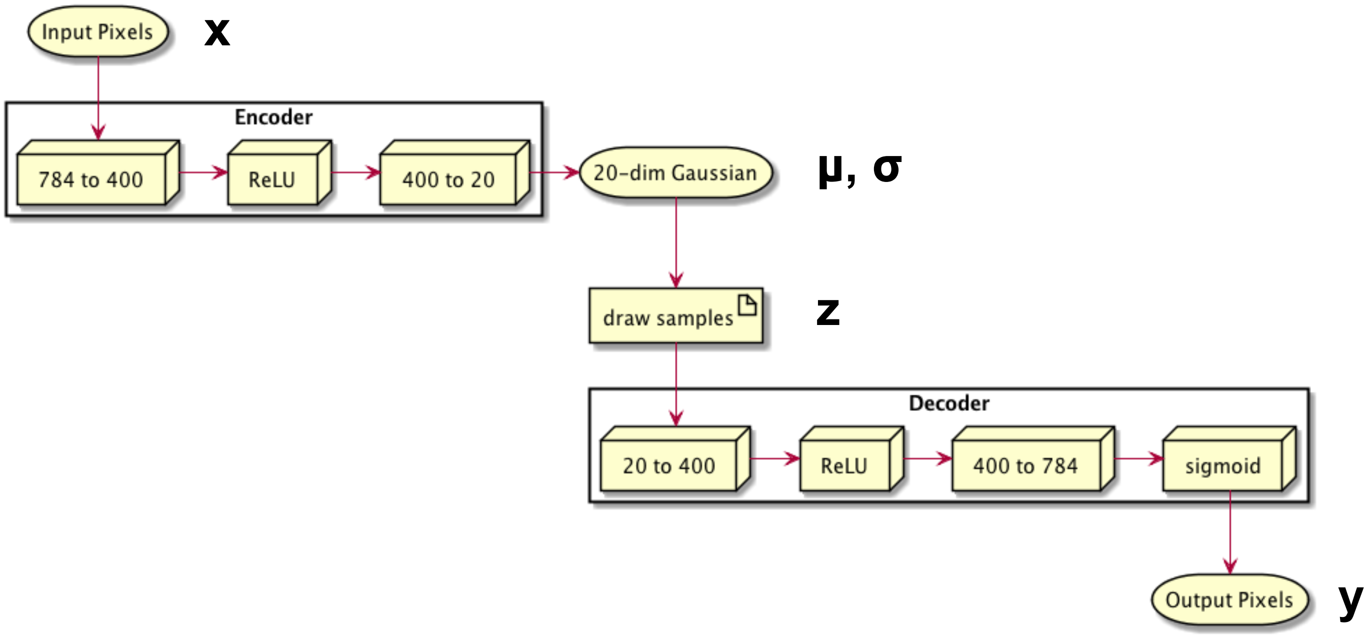
How take **derivatives** with respect to the parameters of a **stochastic** variable ?

$$z \sim \mathcal{N}(\mu, \sigma)$$

$$z = \mu + \sigma \times \epsilon, \text{ where } \epsilon \sim \mathcal{N}(0,1)$$



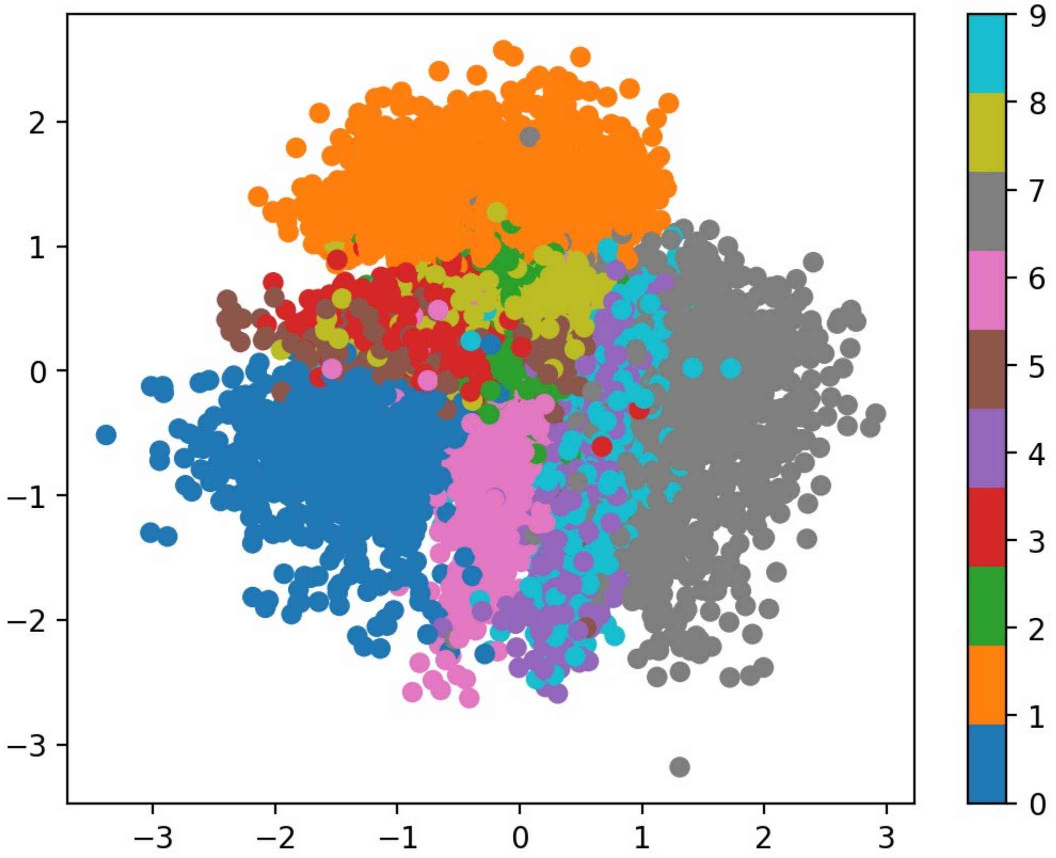
We can thus take gradients of functions involving z, f(z) with respect to the parameters of its distribution μ and σ.



$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) \simeq \frac{1}{2} \sum_{j=1}^{J} \left(1 + \log((\sigma_j^{(i)})^2) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2\right) + \frac{1}{L} \sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)} | \mathbf{z}^{(i,l)})$$

where $\mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)} \odot \boldsymbol{\epsilon}^{(l)}$ and $\boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I})$

VAE latent space



Compared to the AE, the range of values for latent vectors is much smaller, and more centralized. The distribution overall of q(z|x) appears to be much closer to a Gaussian distribution.

Reconstructed digits from the **latent** space: