Latent Variable Models

M2 IASD - Y. Chevaleyre

Outline

1. Introduction
2. Gaussian mixtures and the EM algorithm
3. Variational Analysis of the EM algorithm
4. Variational Auto-Encoders
5. Exercise: probabilistic PCA

Note:
latent var appear everywhere, but in this lecture, we will focus on generative problems ($\neq$ supervised learning)

A Generative problem is learning a distribution from data $(x_1 \dots x_N) \in \mathbb{R}^d$ in order to generate new points from this distribution

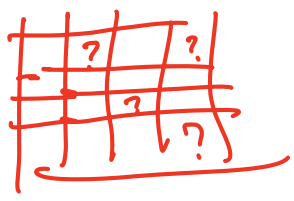
Outline

1. **Introduction**
2. Gaussian mixtures and the EM algorithm
3. Variational Analysis of the EM algorithm
4. Variational Auto-Encoders
5. Exercise: probabilistic PCA

Introduction: what are latent variable models ?

- A latent variable z_i is a <u>missing information</u> about an example x_i
- example:
  - In the **clustering** setting
  - in **Dimensionality reduction** setting
  - in the case of **missing data**
- Usually, best learning algorithm is **ExpectationMaximisation (EM)**

inputs are $x_1 .. x_N$
the class z_i of each point is missing



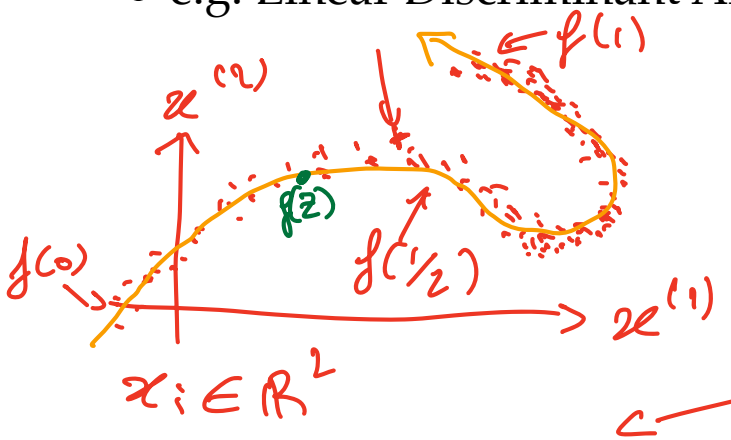
foreach x_i, the $z_i \in \mathbb{R}$ is its coordinate on the orange curve

Probabilistic Machine Learning

- The most appropriate framework for handling latent variables is **Probabilistic ML**.
- The Probabilistic Machine Learning is when you assume that your data (x_i or y_i or both) was generated according to some *generative model* with some unknown parameters

forward generative model

- Learning a model is done by **MLE** or **MAP**
- e.g: Linear Discriminant Analysis, **Linear Regression**, …



Assume I have $f : [0,1] \to \mathbb{R}^2$ describing my orange curve.

Generative model of the points:

for each $i \in 1..N$
- $z_i \sim \text{Unif}([0,1])$
- $\varepsilon_i \sim \mathcal{N}(0, I)$ with $\varepsilon_i \in \mathbb{R}^2$
- $x_i = f(z_i) + \varepsilon_i$

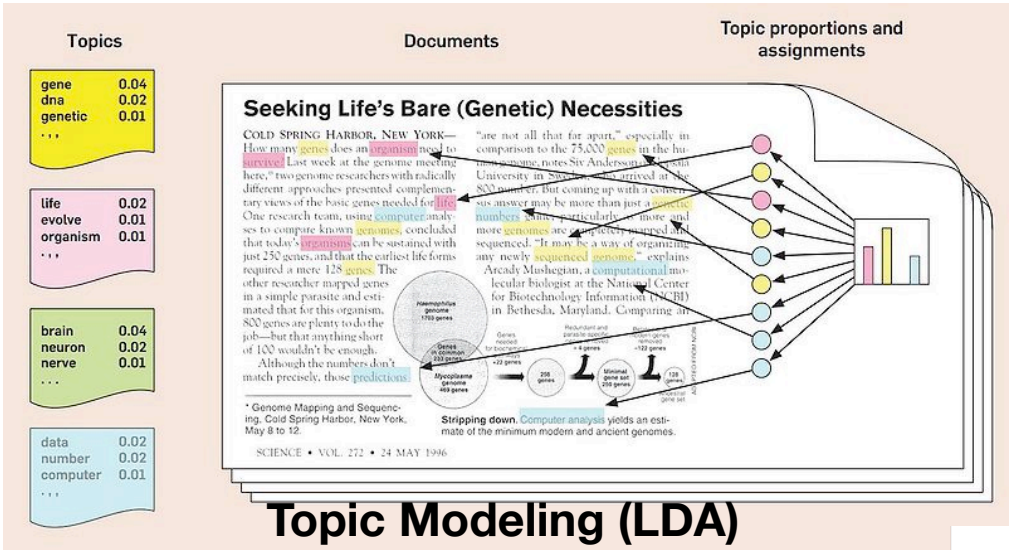
$$p(x|z) = \mathcal{N}(x; f(z), I)$$

$$p(x) = \int p(x|z)\,p(z)\,dz = \int_0^1 \mathcal{N}(x; f(z), I)\,dz$$

$$p(z|x) = \frac{p(x|z)}{p(z)} \cdot p(x)$$

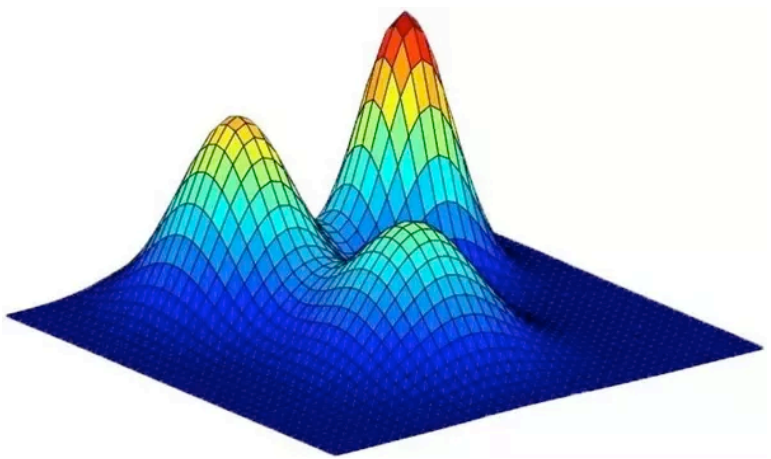
Probabilistic Machine Learning

- This *generative model* may include unobserved variables. We call these *Latent Variables*.

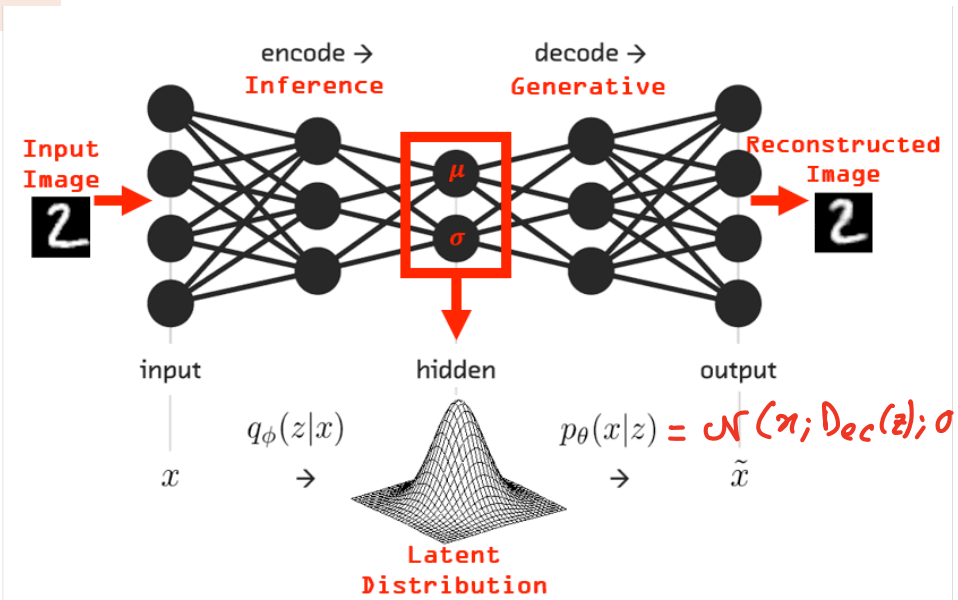


Topic Modeling (LDA)

Latent Diffusion Processes (e.g. DALLE-2)



Gaussian Mixture Models (GMM)



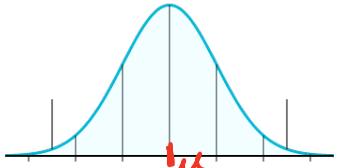
Variational AutoEncoders (VAE)

Outline

1. Introduction
2. **Gaussian mixtures and the EM algorithm**
3. Variational Analysis of the EM algorithm
4. Variational Auto-Encoders
5. Exercise: probabilistic PCA

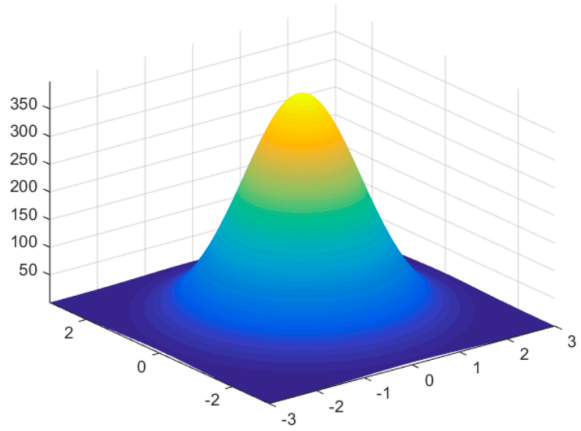
Reminder: The multivariate Normal Distribution

- Gaussienne univariée



$$\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

- Gaussienne multivariée:



A normal (or Gaussian) distribution in 2 variables

$$\mu \in \mathbb{R}^d, \quad \Sigma \in \mathbb{R}^{d\times d}, \quad x \in \mathbb{R}^d$$

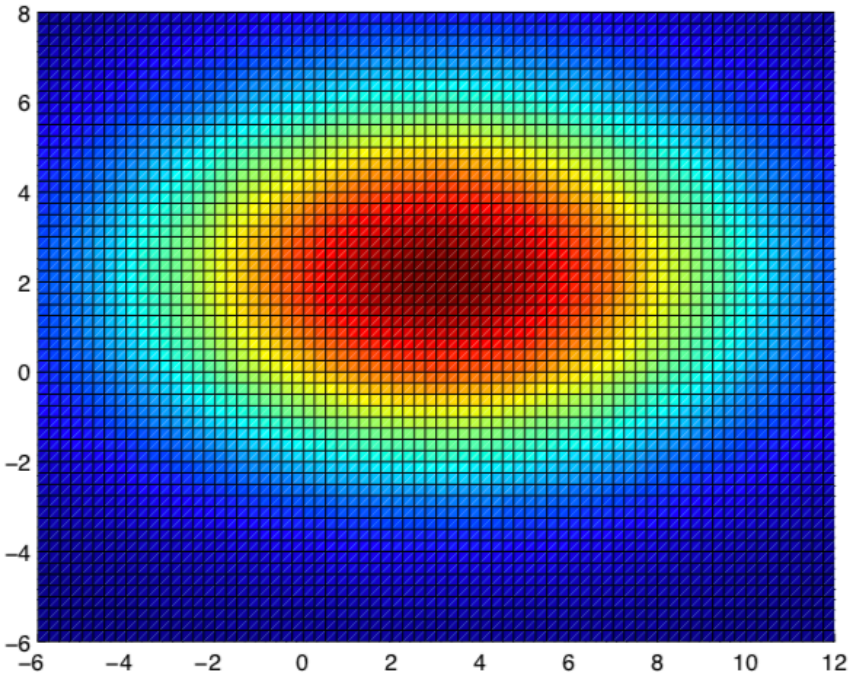
$$\mathcal{N}(x;\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{-\frac{1}{2}}} \exp\left\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}$$

if $\Sigma = I_d$
then $\mathcal{N}(x;\mu,I) = \frac{1}{(2\pi)^{d/2}} \exp\left\{-\frac{1}{2}\|x-\mu\|^2\right\}$

- Notation:
  - $\mathcal{N}(\mu,\sigma^2)$ and $\mathcal{N}(\mu,\Sigma)$ represent the normal distribution.
  - We write $X \sim \mathcal{N}(\mu,\sigma^2)$ or $X \sim \mathcal{N}(\mu,\Sigma)$
  - $\mathcal{N}(x \mid \mu,\Sigma)$ represents the p.d.f. of $\mathcal{N}(\mu,\Sigma)$ at point x

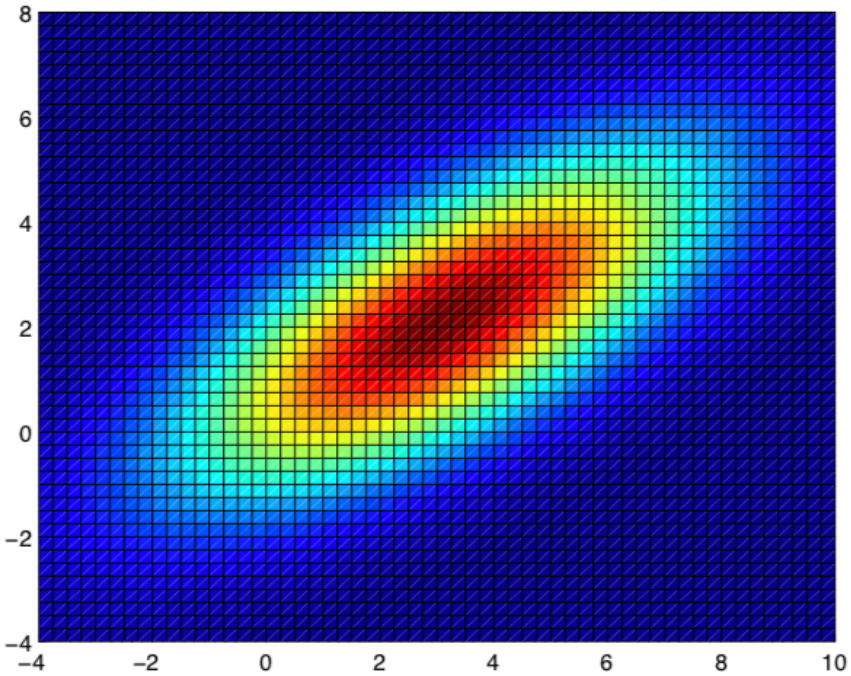
Reminder: The multivariate Normal Distribution

- Gaussienne multivariée:



$$\mu = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 25 & 0 \\ 0 & 9 \end{bmatrix}$$



$$\mu = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

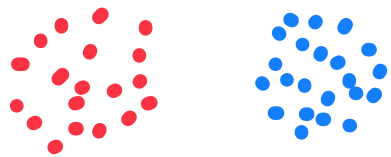
$$\Sigma = \begin{bmatrix} 10 & 5 \\ 5 & 5 \end{bmatrix}$$

Gaussian Mixture Models

- One of the simplest latent variable model is *Gaussian Mixture Models (GMM)*
- same as LDA without class information. Recall LDA with 2 classes first:

(linear discriminant analysis)

LDA ⚠ supervised learning



generative model (with 2 classes)

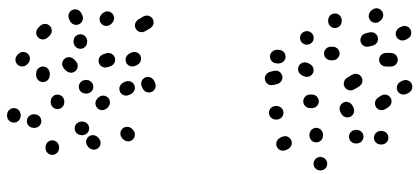
- two gaussians $\mathcal{N}(\mu_1, \Sigma_1)$, $\mathcal{N}(\mu_2, \Sigma_2)$
- a parameter $\pi \in [0, 1]$

for each $i = 1..N$

- $y_i \sim \text{Ber}(\pi) + 1$
- if $y_i = 1$ then $x_i \sim \mathcal{N}(\mu_1, \Sigma_1)$
  if $y_i = 2$, $x_i \sim \mathcal{N}(\mu_2, \Sigma_2)$

Note: this defines $P(X, Y)$, $P(X)$, $P(Y)$..

GMM Unsupervised



generative model (with 2 classes)

- two gaussians $\mathcal{N}(\mu_1, \Sigma_1)$, $\mathcal{N}(\mu_2, \Sigma_2)$
- a parameter $\pi \in [0, 1]$

for each $i = 1..N$

- $z_i \sim \text{Ber}(\pi) + 1$
- if $z_i = 1$ then $x_i \sim \mathcal{N}(\mu_1, \Sigma_1)$
  if $z_i = 2$, $x_i \sim \mathcal{N}(\mu_2, \Sigma_2)$

Note: this defines $P(X, Z)$, $P(X)$, $P(Z)$..

At the end, only $x_1 .. x_N$ are observed. $z_1 .. z_N$ are unknown.

Gaussian Mixture Models: The log-likelihood

exercise:

$K=2$ gaussiens, $d=1$ $(x_1 \dots x_N) \in \mathbb{R}$

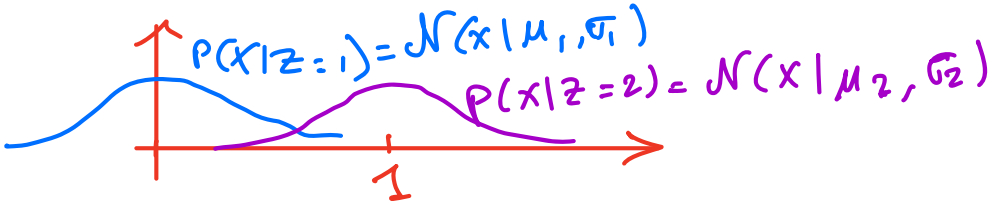
$\mu_1 = 0$ $\mu_2 = 1$

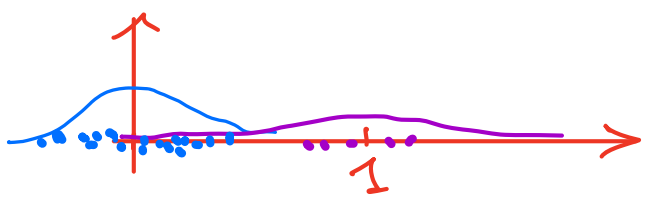
$\sigma_1 = 1$ $\sigma_2 = 1$

$\pi = P(z_i=1) = 80\%$ $P(z_i=2) = 1-\pi$

1) Draw informally $P(X|z=1)$, $P(X|z=2)$ on a graph
2) Draw $P(X, z=1)$ and $P(X, z=2)$ on another graph

} these are all functions of X





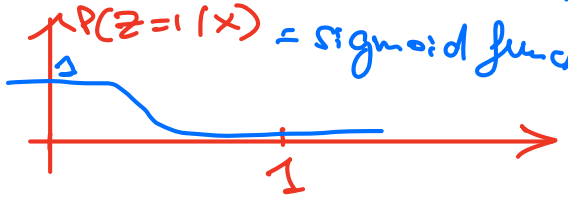
$P(X, z=1) = P(X|z=1) \times P(z=1) = \mathcal{N}(x|\mu_1,\sigma_1) \times \pi$

$P(X, z=2) = \dots\dots\dots = \mathcal{N}(x|\mu_2,\sigma_2) \times (1-\pi)$

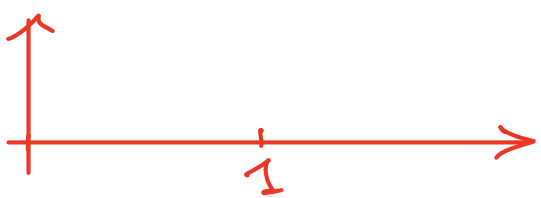
3) Draw $P(z=1|X)$

$$P(z=1|X) = \frac{P(X, z=1)}{P(X)} = \frac{P(X|z=1)\,P(z=1)}{\pi\, P(X|z=1) + (1-\pi)\, P(X|z=2)}$$

$P(z=1|X)$ = sigmoid function



4) Draw points "by hand" from this process.



5) Exercise:
compute and simplify $P(z=1|X)$

exercise:

$K=2$ gaussiens, $d=1$ $(x_1 \dots x_N) \in \mathbb{R}$

$\mu_1 = 0$ $\mu_2 = 1$

$\sigma_1 = 1$ $\sigma_2 = 1$

$\pi = P(z_i = 1) = 80\%$ $P(z_i = 2) = 1 - \pi$

$$P(x \mid z = c) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-\mu_c)^2}$$

$$P(z=1 \mid x) = \frac{P(x \mid z=1)\,\pi}{\pi \cdot P(x \mid z=1) + (1-\pi)\,P(x \mid z=2)}$$

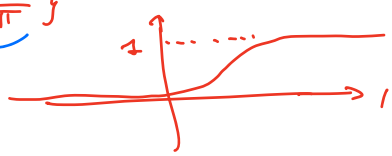
$$= \frac{\pi \exp\left(-\frac{1}{2}(x-\mu_1)^2\right)}{\pi \exp\left(-\frac{1}{2}(x-\mu_1)^2\right) + (1-\pi)\exp\left(-\frac{1}{2}(x-\mu_2)^2\right)}$$

multiply everything by $\exp\left(\frac{1}{2}(x-\mu_1)^2\right) \times \frac{1}{\pi}$

$$= \frac{1}{1 + \frac{1-\pi}{\pi}\exp\left\{-\frac{1}{2}(x-\mu_2)^2 + \frac{1}{2}(x-\mu_1)^2\right\}}$$

$$= \frac{1}{1+\exp\left\{x \times \underbrace{(\mu_2-\mu_1)}_{a} + \underbrace{\frac{\mu_2^2-\mu_1^2}{2} + \ln\frac{1-\pi}{\pi}}_{b}\right\}}$$

$$= \frac{1}{1+\exp(ax+b)}$$



Compute the log likelihood of a GMM model

- K gaussians $\mathcal{N}(\mu_1, \Sigma_1) \dots \mathcal{N}(\mu_K, \Sigma_K)$, with priors $p(z_i = k) = \pi_k$ for $k \in \{1..K\}$
- We have a dataset $x_1 \dots x_N \in \mathbb{R}^N$
- the log likelihood is

$$\ell\ell(\theta) = \log P_\theta(x_1 \dots x_N) = \sum_{i=1}^{N} \log P_\theta(x_i)$$

$$= \sum_{i=1}^{N} \log \sum_{k=1}^{K} P_\theta(x_i \mid z_i = k) P_\theta(z_i = k)$$

(total law of proba)

$$= \sum_{i=1}^{N} \log \sum_{k=1}^{K} \mathcal{N}(x_i \mid \mu_k, \Sigma_k) \cdot \pi(k)$$

- The max likelihood estimate is $\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, \ell\ell(\theta)$

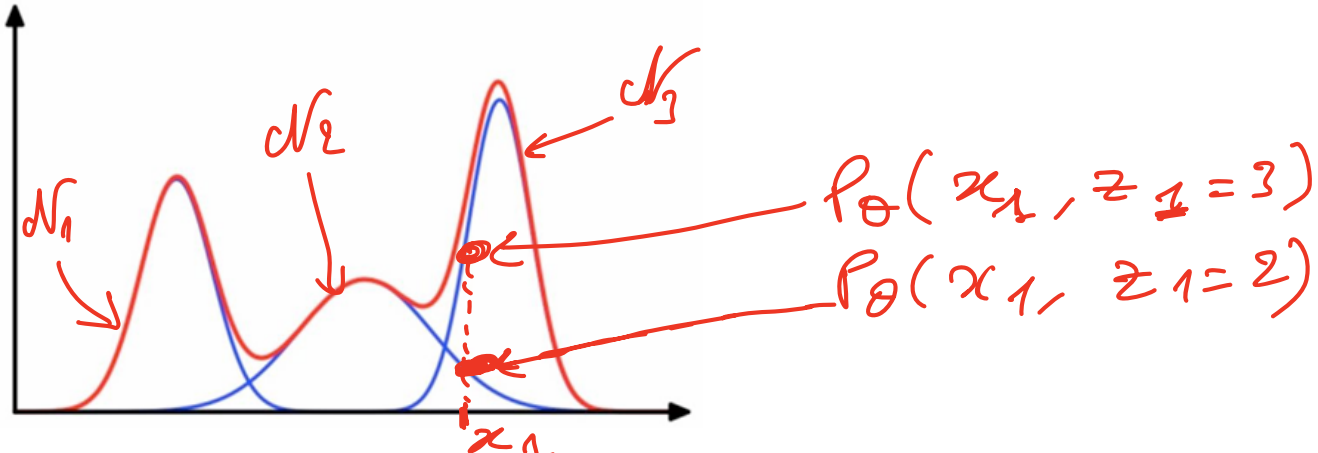
- $\theta = (\mu_1, \Sigma_1, \dots \mu_K, \Sigma_K, \pi_1 \dots \pi_K)$
- $P_\theta(x)$ is the distribution induced by parameters θ

Note: for $K = 1$, computing $\hat{\theta}$ is easy because we know how to compute it.
for $K > 1$: non convex, often "hard" to compute
We could learn θ by gradient descent, but much better strategy

GMM with K>1. The Complete Log Likelihood

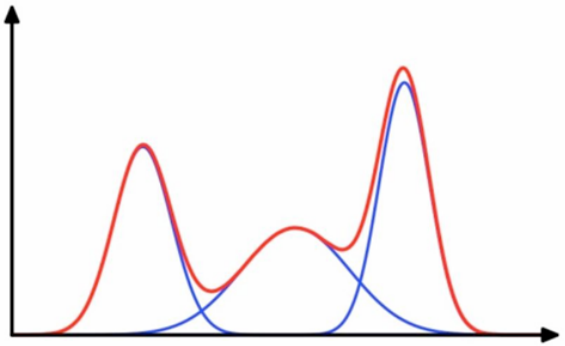
- For $K > 1$, maximising the likelihood is hard
- Idea: optimize a *surrogate* instead: the **Complete Log Likelihood (CLL)**
- The CLL assumes we know $z_1 \ldots z_N$ so we still can't compute it either

$$CLL(\theta, z_1 \ldots z_N) = \log p_\theta(x_1 \ldots x_N, z_1 \ldots z_N)$$



GMM with K>1. The **Expected** Complete Log Likelihood

- Idea: optimize a *surrogate* instead: the **Complete Log Likelihood (CLL)**
- The CLL assumes we know $z_1 \dots z_N$ so we still can't compute it either
- But if for each i we can have an *estimate* the distribution $p(z_i = k \mid x_i; \hat{\theta})$ then we can compute the **Expected CLL**. Let $q_i(k)$ be this estimation.



$$q_i(k) \approx p_\theta(z_i = k \mid x_i)$$

$$\mathcal{L}(q_i, \theta, i) = \mathbb{E}_{z_i \sim q_i} \log p_\theta(x_i, z_i) = \mathbb{E}_{z_i \sim q_i} \log\left(p_\theta(x_i \mid z_i) \cdot \pi_i\right)$$

$$= \sum_{k=1}^{K} q_i(k) \log\left[p_\theta(x_i \mid z_i = k) \cdot \pi_k\right]$$

$$ECLL = \mathcal{L}(q, \theta) = \sum_{i=1}^{N} \mathcal{L}(q_i, \theta, i) = \mathbb{E}_{z_1 \sim q_1, \dots, z_N \sim q_N}\left[CLL(\theta, z_1 \dots z_N)\right]$$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} q_i(k) \log\left[p_\theta(x_i \mid z_i = k) \cdot \pi_k\right]$$

⚠ $\mathcal{L}(q, \theta)$ is convex in $\mu_1 \dots \mu_K$!

The EM Algorithm

Recall: $q_i(k) = \frac{\mathcal{N}(x_i \mid \mu_k, \Sigma_k)\pi_k}{\sum_{k'=1}^{K} \mathcal{N}(x_i \mid \mu_{k'}, \Sigma_{k'})\pi_{k'}}$

= Sigmoid for $K=2$, softmax in general

- EM Algorithm:

  1. Initialize $\hat{\theta}$ arbitrarily

  2. *E-Step:* for each $i \in \{1 \dots N\}, k \in \{1 \dots K\}$
     compute $q_i(k) = p(z_i = k \mid x_i; \hat{\theta})$ $= p_{\hat{\theta}}(z_i = k \mid x_i) \neq p(z_i = k \mid x_i)$

  3. *M-Step:* compute $\hat{\theta} \leftarrow \arg\max_{\theta} \mathcal{L}(q, \theta)$ ↑ (expected complete LL).

- Details: (in 1D):

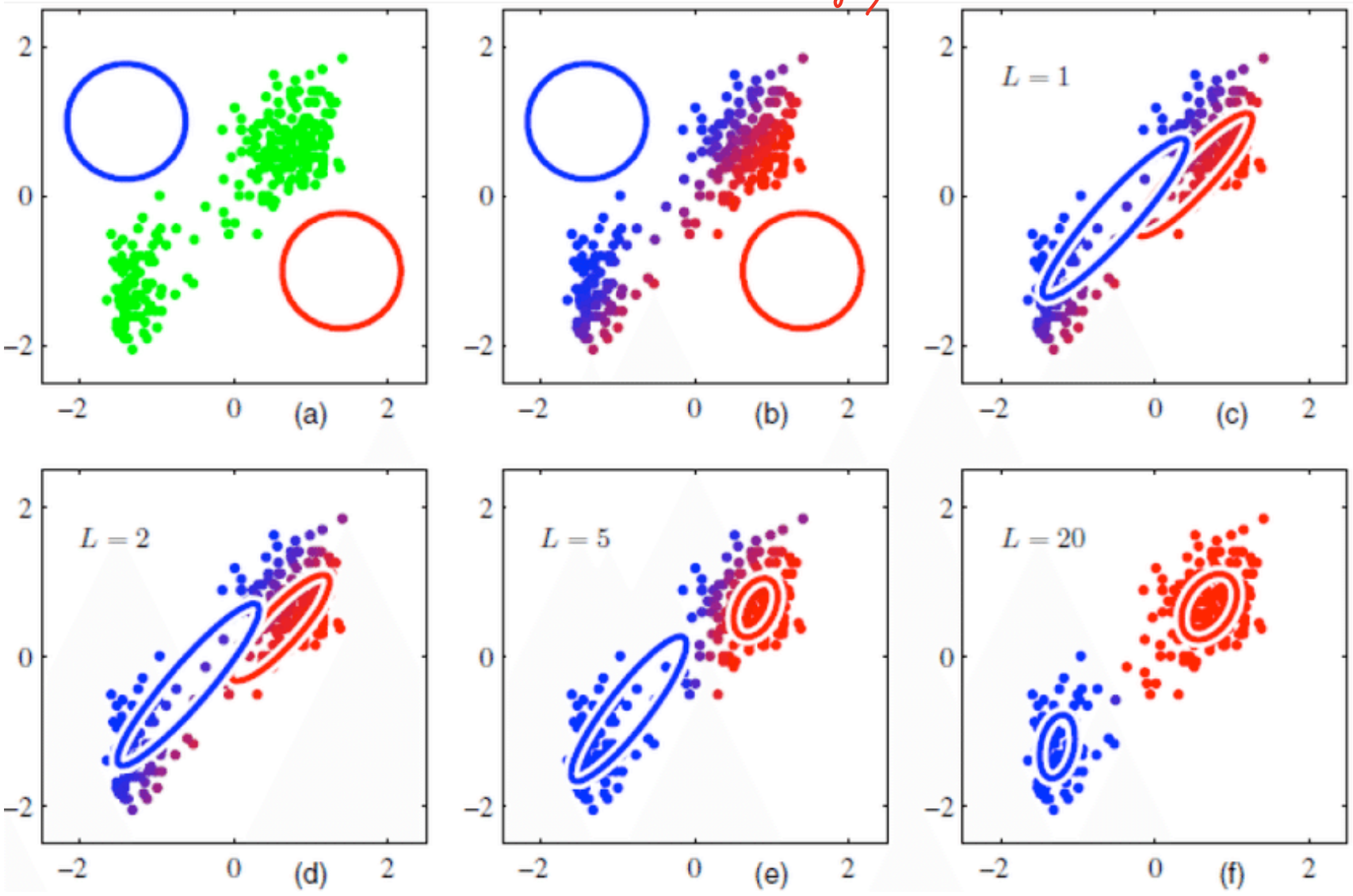
the opt is convex.

$$\arg\max_{\theta} \mathcal{L}(q,\theta) = \arg\max_{\theta} \sum_{i=1}^{N} \sum_{k=1}^{K} q_i(k) \left(\text{cste} - \ln \sigma_k - \frac{(x_i - \mu_k)^2}{2\sigma_k^2} \right)$$

For fixed $\sigma_k = \sigma$, I get $\hat{\mu}_k = \left(\frac{1}{\sum_{i'=1}^{N} q_{i'}(k)} \right) \times \sum_{i=1}^{N} x_i \times q_i(k)$

The EM Algorithm

E-step: estimating q



Outline

1. Introduction
2. Gaussian mixtures and the EM algorithm
3. **Variational Analysis of the EM algorithm**
4. Variational Auto-Encoders
5. Exercise: probabilistic PCA

Variational Analysis of EM: The ELBO (1)

- We really want to optimise the likelihood $p_\theta(x_1 \ldots x_N)$, but instead we optimise the a surrogate: the expected CLL $\mathscr{L}(q, \theta)$. What is the link ?
- Recall that $\mathscr{L}(q, \theta) = \sum_{i=1}^{N} \mathscr{L}(q, \theta, i)$ where $\mathscr{L}(q, \theta, i) = \mathbb{E}_{z_i \sim q_i}\left[p_\theta(X = x_i, z_i = k)\right]$
- Let us pick a single point x and compute its likelihood

$q_x(k) \approx p_\theta(z=k \mid x=x)$

$$\log p_\theta(x) = \ln \sum_{k=1}^{K} p_\theta(x, z=k) = \ln \sum_{k=1}^{K} q_x(k) \times \frac{p_\theta(x, z=k)}{q_x(k)}$$

Jensen's inequality : if f is concave then $\mathbb{E}(f(x)) \leq f(\mathbb{E}(x))$

$$\log p_\theta(x) = \ln \mathbb{E}_{k \sim q_x}\left[\frac{p_\theta(x, z=k)}{q_x(k)}\right] \geq \mathbb{E}_{k \sim q_x} \ln\left[\frac{p_\theta(x, z=k)}{q_x(k)}\right]$$

$$\log p_\theta(x) \geq \sum_{k=1}^{K} q_x(k) \ln p_\theta(x, z=k) - \sum_{k=1}^{K} q_x(k) \ln q_x(k)$$

$\mathcal{L}(q_x, \theta, x)$ = ECLL for a single example

Entropy

ELBO(x, θ, q_x) Evidence Lower Bound

Variational Analysis of EM: The ELBO (2)

- We really want to optimise the likelihood $p(x_1 \ldots x_N \mid \theta)$, but instead we optimise the a surrogate: the expected CLL $\mathscr{L}(q, \theta)$. What is the link ?

- Recall that $\mathscr{L}(q,\theta) = \sum_{i=1}^{N} \mathscr{L}(q,\theta,i)$ where $\mathscr{L}(q,\theta,i) = \mathbb{E}_{z_i \sim q_i}\left[p(X = x_i, z_i = k \mid \theta)\right]$

How close the log likelihood and the ELBO are?

$$\ln p_\theta(x) - \text{ELBO}(x,\theta,q_x) = \ln p_\theta(x) - \mathbb{E}_{k\sim q_x}\left[\ln\left(\frac{p_\theta(x, z=k)}{q_x(k)}\right)\right]$$

$$= \mathbb{E}_{k\sim q_x}\left[\ln \frac{p_\theta(x)\, q_x(k)}{p_\theta(x, z=k)}\right] = \mathbb{E}\left[\ln \frac{q_x(k)}{p_\theta(z=k \mid x)}\right]$$

$$= KL\left(q_x(z) \,\|\, p_\theta(z \mid x)\right)$$

Properties of the Kullback Leibler divergence

- $KL(q \| q') = \mathbb{E}_{k\sim q}\left[\ln\left(\frac{q(x)}{q'(x)}\right)\right]$
- $KL(q \| q') \geq 0 \quad \forall q, q'$
- $KL(q \| q') = 0$ iff $q = q'$
- $KL(q \| q') \neq KL(q' \| q)$

$\Rightarrow$

- $\ln p_\theta(x) = \text{ELBO}(x,\theta,q_x)$ iff $q_x(z) = p_\theta(z \mid x)$
  $\Downarrow$
  $\operatorname{argmax}_{q_x(\cdot)} \text{ELBO}(x,\theta,q_x) = p_\theta(z \mid x)$
- $\operatorname{argmax}_\theta \text{ELBO}(x,\theta, p_\theta(z \mid x)) = \operatorname{argmax}_\theta \ln p_\theta(x)$
- $\operatorname{argmax}_{\theta, q_x} \text{ELBO}(x,\theta,q) = \operatorname{argmax}_\theta \ln p_\theta(x)$

Variational Analysis of EM: reinterpreting the algorithm

- EM Algorithm:

  1. Initialize $\hat{\theta}$ arbitrarily

  2. *E-Step:* for each $i \in \{1 \ldots N\}, k \in \{1 \ldots K\}$

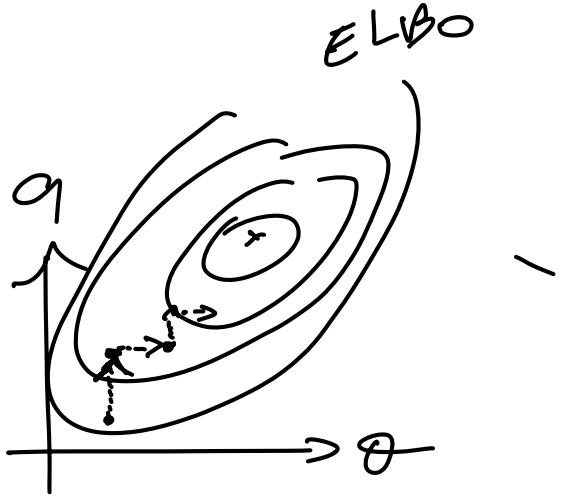
     compute $q_i(k) = p(z_i = k \mid x_i; \hat{\theta})$

     $$q_1 \ldots q_N = \arg\max_{q_1 \ldots q_N} \sum_i ELBO(x_i, \hat{\theta}, q_i)$$

     $= p_\theta(z_1 \mid x_1) \ldots p_\theta(z_N \mid x_N)$

  3. *M-Step:* compute $\hat{\theta} \leftarrow \arg\max_\theta \mathcal{L}(q, \theta)$

     $$\hat{\theta} = \arg\max_\theta \sum_i ELBO(x_i, \theta, q_i)$$



- Because $ELBO(x_i, \theta, q_i) = \ln p_\theta(x_i) - KL(q_i(z), p_\theta(z \mid x_i))$ the E-step minimizes the KL and the M-step maximizes the log likelihood while possibly increasing the KL
- Each step (E and M) increases the ELBO. Unless we get stuck in a local minimum, this will reach the optimal ELBO which coincides with the optimal log likelihood.

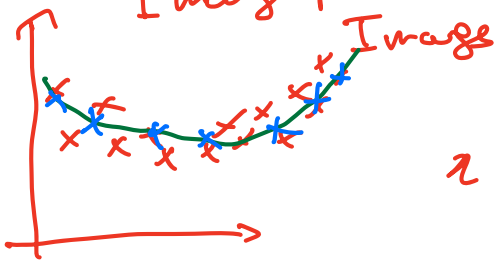
Outline

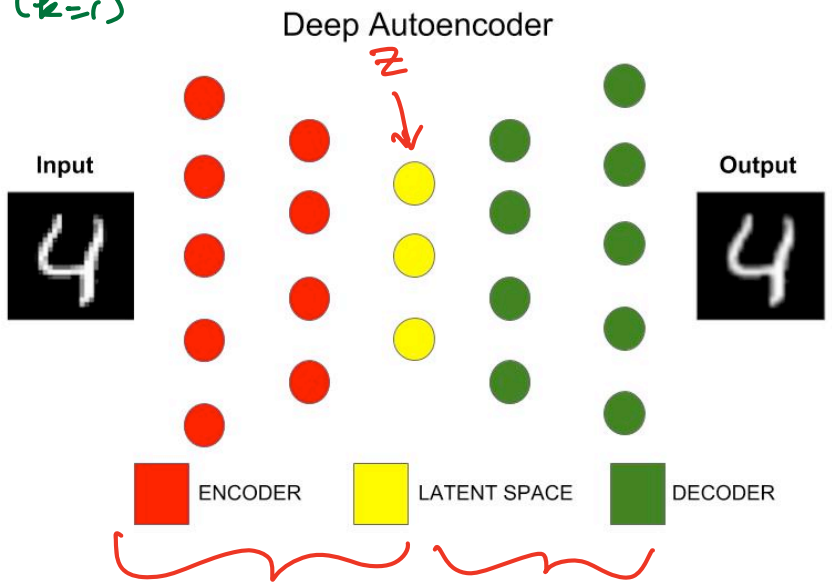
1. Introduction
2. Gaussian mixtures and the EM algorithm
3. Variational Analysis of the EM algorithm
4. **Variational Auto-Encoders**
5. Exercise: probabilistic PCA

A word about auto-encoders

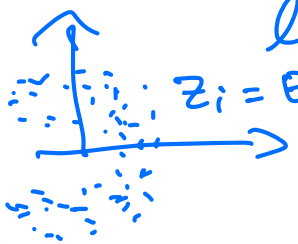
- Standard auto-encoders learn to reduce the dimensionality of examples in a dataset

Reconstructed image
Latent space is $\mathbb{R}^k$ ($k=1$)
Image space $\mathbb{R}^d$





$Enc_\varphi(x)$ Dec_ϕ



latent space $\mathbb{R}^k$

$\tilde{x}$ (reconstructed image)

φ: weights of the encoder

ϕ: weights of the decoder

$x_i, \tilde{x}_i \in \mathbb{R}^d$

$z_i \in \mathbb{R}^k$, $k \ll d$

To train this:

$$\underset{\varphi,\phi}{\operatorname{argmin}} \sum_{i=1}^{N} \|x_i - \tilde{x}_i\|^2$$

where $\tilde{x}_i = Dec_\phi(z_i)$,
$z_i = Enc_\varphi(x_i)$

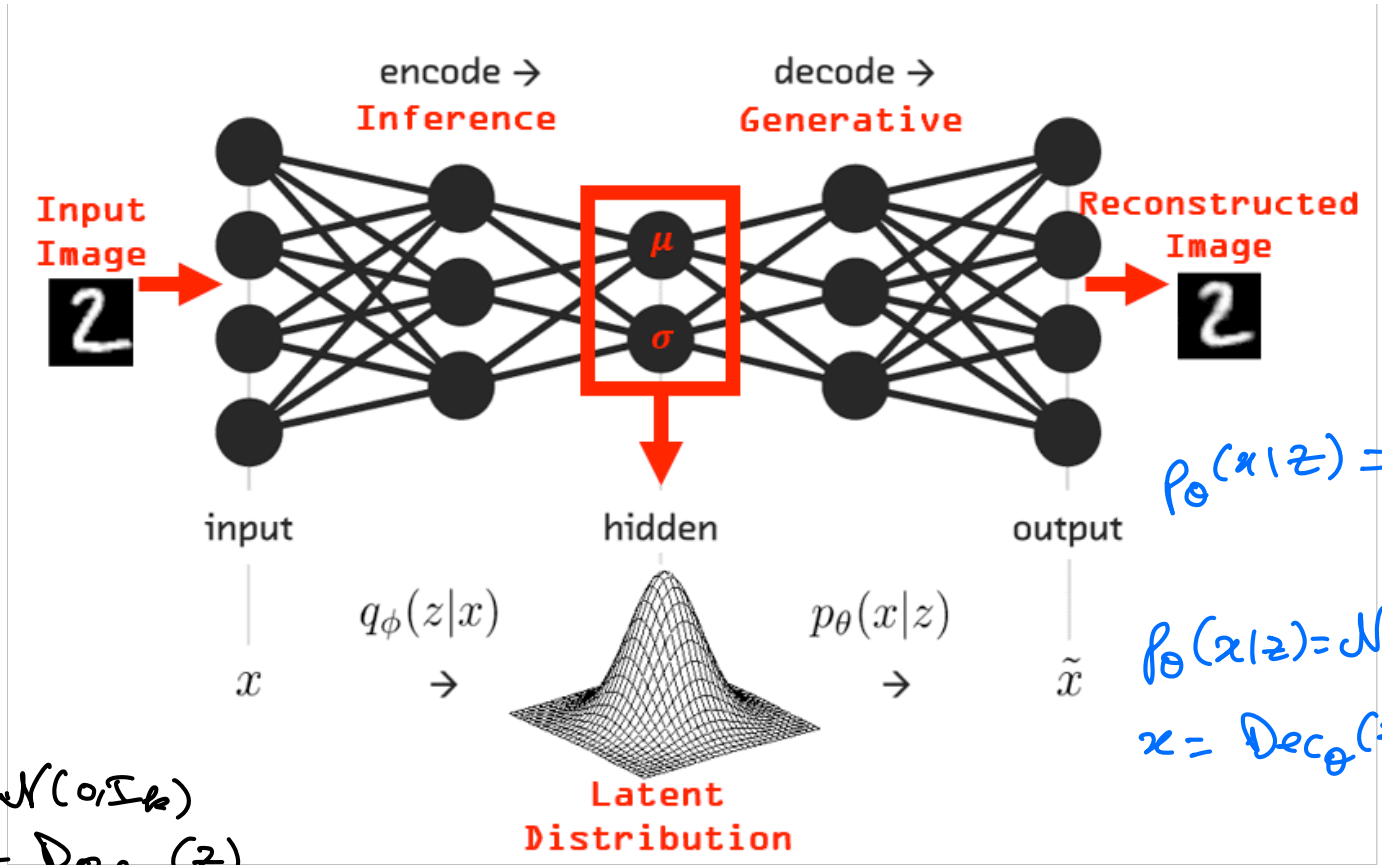
variant: denoising auto encoder

- add noise ε_i to the input

$$\underset{\varphi,\phi}{\operatorname{argmin}} \sum_i \|x_i - Dec_\phi(Enc_\varphi(x_i + \varepsilon_i))\|^2$$

A word about *variational* auto-encoders

- **Variational** auto-encoders (VAE) learn to reduce the dimensionality of examples in a dataset *and* produces for each example x_i a distribution of latent vectors z_i such that overall, latent vectors $z_1 \ldots z_N$ are normally distributed.



$p_\theta(x|z) = \mathcal{N}(x|\mu_z, Id)$

$\mu_z = Dec_\theta(z)$

$p_\theta(x|z) = \mathcal{N}(x|Dec_\theta(z), I)$

$x = Dec_\theta(z) + \varepsilon$

$\varepsilon \sim \mathcal{N}(0, I)$

$z \sim \mathcal{N}(0, I_k)$

$x = Dec_\theta(z)$

Training the decoder alone

Assume we have a dataset $\{(x_i, z_i)\}_{i=1..N}$

$x_i \in \mathbb{R}^d$, $z_i \in \mathbb{R}^k$ $k < d$.

$$\hat{\theta} = \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x_i \mid z_i) = \arg\max_\theta \sum_{i=1}^{N} \log \mathcal{N}(x_i \mid \text{Dec}_\theta(z_i), I)$$

$$= \arg\max_\theta \sum -\frac{1}{2} \|x_i - \text{Dec}_\theta(z_i)\|^2$$

$$= \arg\min_\theta \sum_{i=1}^{N} \|x_i - \text{Dec}_\theta(z_i)\|^2$$

Of course, in practice we only have access to $\{x_i\}_{i=1..N}$. Thus, we will train the encoder to learn the distribution of z for each x_i.

Understanding $p(x)$ and deriving the ELBO

$z_i \sim \mathcal{N}(0, I_k)$ $\quad$ I_k is $\mathbb{R}^{k\times k}$ identity matrix.

$p_\theta(x|z)$.

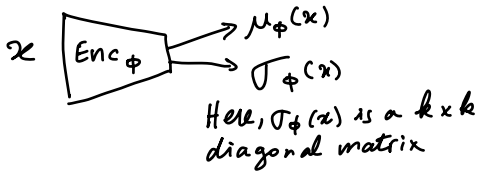
$p_\theta(x) = \int_{\mathbb{R}^k} p_\theta(x|z)\, p(z)\, dz$ $\xrightarrow{\text{the goal is to train}}$ $\hat{\theta} = \text{argmax} \sum_{i=1}^{n} \log p_\theta(x_i)$

Unfortunately, this integral is hard to estimate because for almost all $z \sim p(z)$, we have $p_\theta(x|z)$ close to zero

Idea: apply the ELBO which requires

$$q_x(z) = q(z|x)$$

to represent $q(z|x)$, we will use an encoder



Here, $\sigma_\phi(x)$ is a $k \times k$ diagonal matrix

$$q_\phi(z|x) = \mathcal{N}(z \mid \underbrace{\mu_\phi(x)}_{\mathbb{R}^k}, \underbrace{\sigma^2_\phi(x)}_{\mathbb{R}^{k\times k}})$$

ELBO objective

$$\log p_\theta(x) \geq \text{ELBO}(x, \theta, q_\phi)$$

$$= \mathbb{E}_{z\sim q_\phi(\cdot|x)}\left[\ln \frac{p_\theta(x,z)}{q_\phi(z|x)}\right]$$

$$= \mathbb{E}_{z\sim q_\phi(\cdot|x)}\left[\ln p_\theta(x|z) - \ln \frac{q_\phi(z|x)}{p(z)}\right]$$

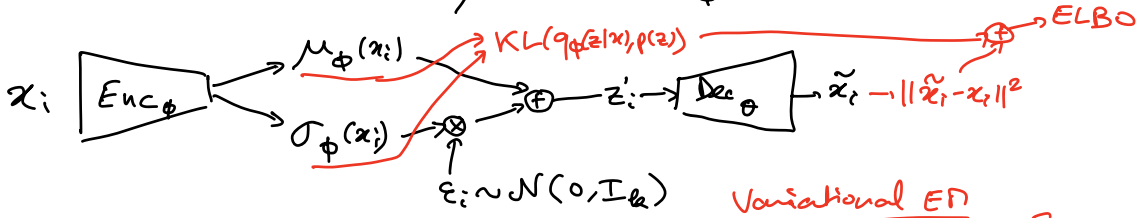
$$= \mathbb{E}_z\left[\ln p_\theta(x|z)\right] - KL(q_\phi(z|x), p(z))$$

$$= \mathbb{E}_{z\sim q_\phi(\cdot|x)}\left[-\tfrac{1}{2}\|x - \text{dec}_\theta(z)\|^2\right] - \underbrace{KL(q_\phi(z|x), p(z))}_{\frac{1}{2}\|\mu_\phi(x)\|^2 + \text{tr}(\Sigma_\phi(x)) - k - \log|\Sigma_\phi(x)|} + \text{cste}$$

Reparametrization trick

- Pb: ELBO will not backpropagate on ϕ (encoder) because of $z \sim q_\phi$
  $\Rightarrow$ can't compute ∇_ϕ ELBO $\Rightarrow$ no training possible.

- trick: $z \sim q_\phi(\cdot|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi^2(x))$
  is distributed identically to
  $z' = \mu_\phi(x) + \sigma_\phi(x) \cdot \epsilon$ where $\epsilon \sim \mathcal{N}(0,1)$



Variational EM

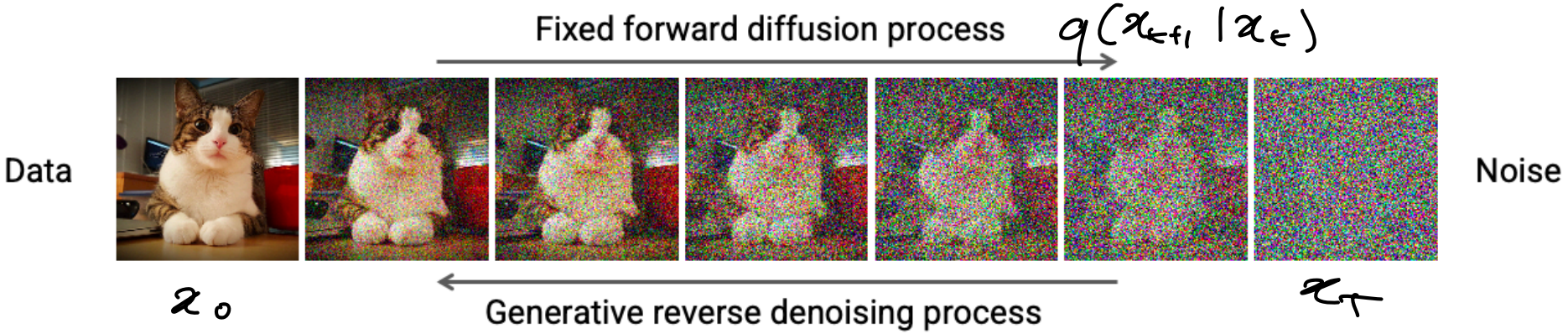
1) Init $\hat{\phi}$ and $\hat{\theta}$
2) $\hat{\phi} \leftarrow \underset{\phi}{\text{argmax}}\ \text{ELBO}(\phi, \hat{\theta})$
3) $\hat{\theta} \leftarrow \underset{\theta}{\text{argmax}}\ \text{ELBO}(\hat{\phi}, \theta)$

Outline

1. Introduction
2. Gaussian mixtures and the EM algorithm
3. Variational Analysis of the EM algorithm
4. Variational Auto-Encoders
5. **Exercise: probabilistic PCA**

Probabilistic PCA

Denoising Diffusion Models



$q(x_0)$: distribution of image in the dataset

$$q(x_t | x_{t-1}) = \mathcal{N}(x_t \mid \sqrt{1-\beta_t} \cdot x_{t-1}, \beta_t I_d)$$

$$x_t = x_{t-1} \times \sqrt{1-\beta_t} + \beta_t \cdot \varepsilon_t$$

$\varepsilon_t \sim \mathcal{N}(0, I)$

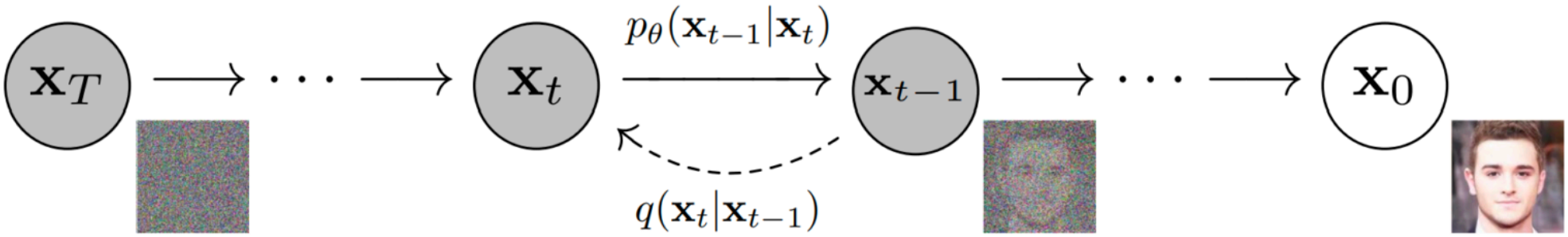
$\beta_t \in]0, 1[$

$\beta_1 < \beta_2 < \beta_3 \dots$

$$q(x_1 \dots x_T | x_0) = \prod_{t=1}^{T} q(x_t | x_{t-1})$$

(chain rule of probability)

Denoising Diffusion Models



Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

observation:

- for simplicity, $\alpha_t = 1 - \beta_t \qquad \bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$

$$x_t = \sqrt{\alpha_t} \cdot x_{t-1} + \sqrt{1-\alpha_t} \cdot \varepsilon_{t-1}$$

$$x_{t-1} = \sqrt{\alpha_{t-1}} \cdot x_{t-2} + \sqrt{1-\alpha_{t-2}} \cdot \varepsilon_{t-2}$$

$$\vdots$$

$$\boxed{x_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1-\bar{\alpha}_t} \cdot \varepsilon}$$

$$q(x_t | x_0) = \mathcal{N}(x_t | \sqrt{\bar{\alpha}_t} x_0 ; (1-\bar{\alpha}_t) I_d)$$

- $p_\theta(x_0 \dots x_T) = p(x_T) \cdot \prod_{t=1}^{T} p_\theta(x_{t-1} | x_t)$

Assume $p_\theta(x_{t-1} | x_t) = \mathcal{N}(x_{t-1} | \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$

- Ideally, to learn $p_\theta(x_{t-1} | x_t)$, we would like to compute $q(x_{t-1} | x_t)$ → hard.
- Instead, compute $q(x_{t-1} | x_t, x_0)$

$$q(x_{t-1} | x_t, x_0)$$

$$= q(x_t | x_{t-1}, x_0) \frac{q(x_{t-1} | x_0)}{q(x_t | x_0)}$$

conditional Bayes rule

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

known / normally distributed

$$= \mathcal{N}(x_{t-1} | \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I)$$

$$\tilde{\mu}_t(x_t, x_0) = \frac{\sqrt{\alpha_t} \cdot (1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \cdot x_t + \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1-\bar{\alpha}_t} \cdot x_0$$

$$x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} (x_t - \sqrt{1-\bar{\alpha}_t} \varepsilon_t)$$

$$\Rightarrow \tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \varepsilon_t \right)$$

At the end, we want $q(x_{t-1}, x_t)$ to be close to $p_\theta(x_{t-1}, x_t)$

We want to max the ELBO, or minimize -ELBO

$$-\text{ELBO} = \mathbb{E}_q \left[-\log \frac{p_\theta(x_0 \dots x_T)}{q(\underbrace{x_1 \dots x_T}_{\text{latent var}} | x_0)} \right]$$

$$= \mathbb{E}_q \left[-\log p_\theta(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} \right]$$

$$= \mathbb{E}_q \left[-\log p_\theta(x_T) - \sum_{t \geq 2} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1}, x_0)} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right]$$

↑ does not change the value of $q(\cdot|\cdot)$

↓ conditional Bayes rule

$$= \mathbb{E}_q \left[-\log p_\theta(x_T) - \sum_{t \geq 2} \log \left\{ \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} \cdot \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} \right\} - \log \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \right]$$

$$= \mathbb{E}_q \left[-\log \frac{p_\theta(x_T)}{q(x_T|x_0)} - \sum_{t \geq 2} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} - \log p_\theta(x_0|x_1) \right]$$

$$= \mathbb{E}_q \Big[\underbrace{KL(q(x_T|x_0), p_\theta(x_T))}_{L_T = \text{cste}} + \sum_{t \geq 2} \underbrace{KL(q(x_{t-1}|x_t, x_0), p_\theta(x_{t-1}|x_t))}_{L_{t-1}} \underbrace{- \log p_\theta(x_0|x_1)}_{L_0} \Big]$$

Let us focus on L_{t-1}

$$L_{t-1} = KL(\underbrace{q(x_{t-1}|x_t, x_0)}_{\mathcal{N}(x_{t-1}|\tilde{\mu}_t, \tilde{\beta}_t I)}, \underbrace{p_\theta(x_{t-1}|x_t)}_{\mathcal{N}(x_{t-1}|\mu_\theta(x_t), \tilde{\beta}_t I)})$$

$$= \|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t)\|^2 \times \text{cste}$$

recall that

$$\hat{\mu}_t = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\cdot\varepsilon_t\right)$$

let us define $\mu_\theta(x_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\cdot\varepsilon_\theta(x_t,t)\right)$

$$L_{t-1} = \|\varepsilon_t - \varepsilon_\theta(x_t,t)\|^2 \times \text{cste}$$

$$= \|\varepsilon_t - \varepsilon_\theta\left(\sqrt{\bar{\alpha}_t}\cdot x_0 + \sqrt{1-\bar{\alpha}_t}\cdot\varepsilon_t, t\right)\|^2 \times \text{cste}$$