10707
Deep Learning: Spring 2021

Andrej Risteski

Machine Learning Department

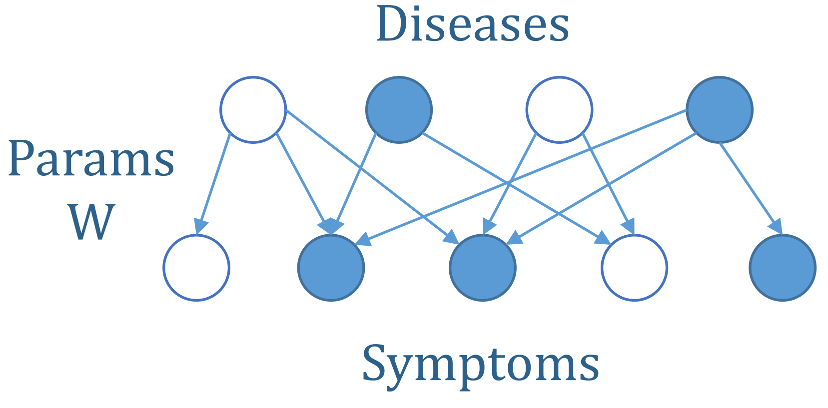
Lecture 11+12:
Variational methods, directed latent-variable models

Bayesian Networks

Directed Graphs are useful for expressing causal relationships between random variables.

Your symptoms: fever + red spots.

Probability that you have measles?

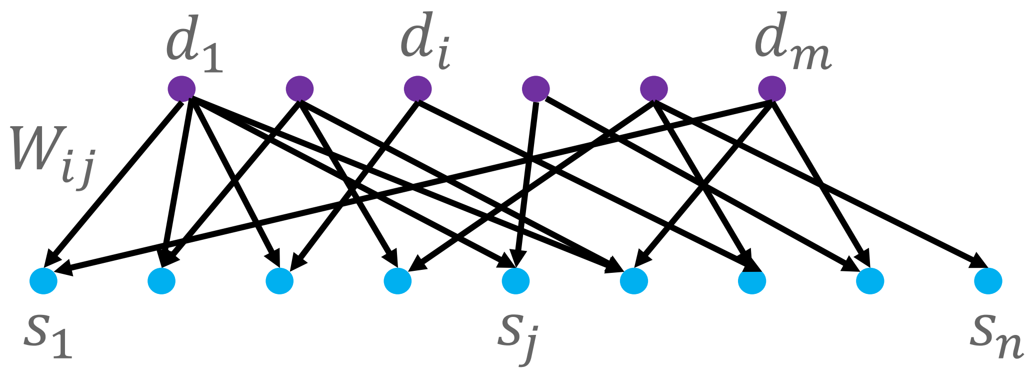


Bayesian network succinctly describes Pr[symptom| diseases]

Noisy-OR networks

$d_i, s_j \in \{0,1\}$
$W_{ij} \geq 0$

Diseases (causes)



Symptoms (effects)

- Each d_i is on **independently** with prob. ρ
- When d_i is on, it **activates** s_j with probability $1 - \exp(-W_{ij})$.
- s_j is **on** if one of d_i's **activates** s_j

Bayesian Networks

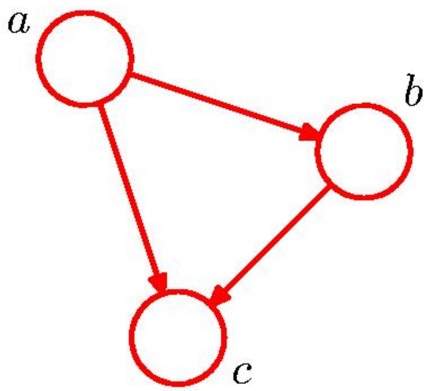
Directed Graphs are useful for expressing causal relationships between random variables.

"Deriving" Bayesian Networks as restrictions of arbitrary distributions:

An **arbitrary** joint distribution $p(a, b, c)$ over three random variables a,b, and c can be written as

$$p(a, b, c) = p(c|a, b)p(a, b) = p(c|a, b)p(b|a)p(a)$$

Associate a graph with the decomposition:



- Node for each of the random variables.
- Add **directed** links to the graph from the nodes corresponding to the vars on which the distribution is conditioned.

Bayesian Networks

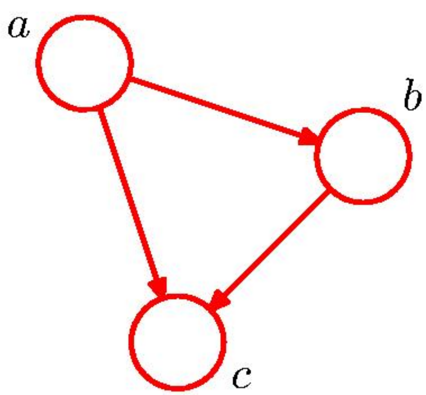
Directed Graphs are useful for expressing causal relationships between random variables.

"Deriving" Bayesian Networks as restrictions of arbitrary distributions:

An **arbitrary** joint distribution $p(a, b, c)$ over three random variables a,b, and c can be written as

$$p(a, b, c) = p(c|a, b)p(a, b) = p(c|a, b)p(b|a)p(a)$$

Associate a graph with the decomposition:



Different ordering => different graphical representation.

Joint distribution over K variables factorizes:

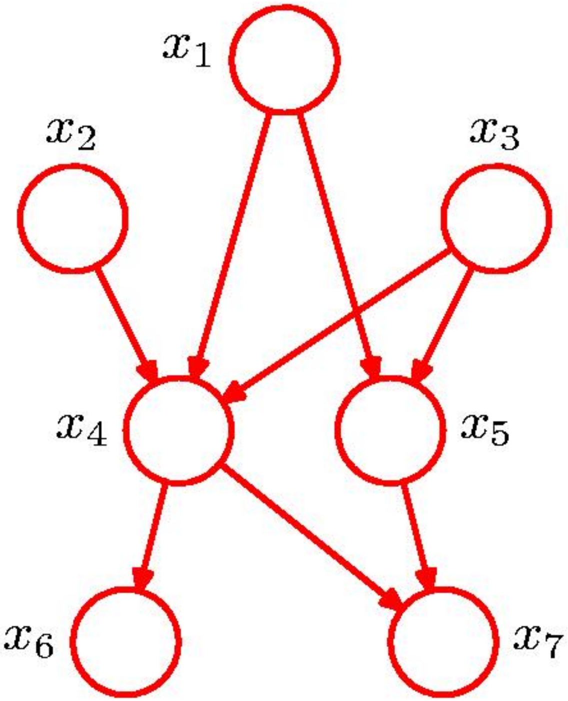
$$p(x_1, \dots, x_K) = p(x_K|x_1, \dots, x_{K-1}) \dots p(x_2|x_1)p(x_1)$$

Corresponding undirected graph is fully connected:

(as each lower-numbered node points to each higher-numbered node)

Bayesian Networks

A graph that is **not** fully connected conveys information about the conditional independence structure of the distribution it encodes.



E.g. consider the graph on the left.

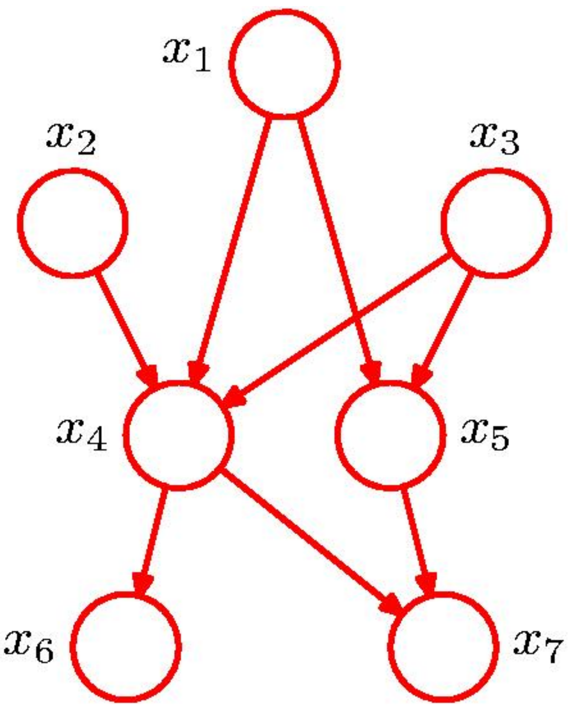
It encodes distributions over $x_1,\ldots,x_7$ that can be written as the product:

$$
\begin{aligned}
p(x_1,\ldots,x_7) \quad = \quad & p(x_1)p(x_2)p(x_3)p(x_4|x_1,x_2,x_3) \\
& p(x_5|x_1,x_3)p(x_6|x_4)p(x_7|x_4,x_5)
\end{aligned}
$$

Note the change from the previous slide: e.g. x_5 is **not** conditioned on all of x_1, x_2, x_3, x_4 but only on x_1, x_3.

The general case: factorization

The joint distribution defined by the graph is given by the product of a conditional distribution for each node conditioned on its parents:



$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\mathrm{pa}_k)$$

where pa_k denotes a set of parents for the node x_k.

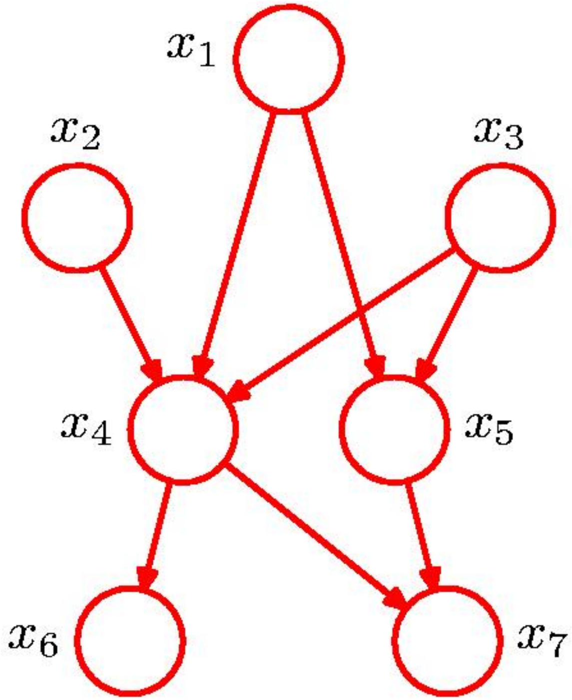
Each of the conditional distributions will typically have some parametric form. (e.g. product of Bernoullis in the noisy-OR case)

Important restriction: There must be **no directed cycles!** (i.e. graph is a DAG)

Crucial property: easy sampling

Consider a joint distribution over K random variables $p(x_1, x_2, ..., x_K)$ that factorizes as:

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k | \mathrm{pa}_k)$$



Suppose each of the conditional distributions are easy to sample from. How do we sample from the joint?

Start at the top and sample in order.

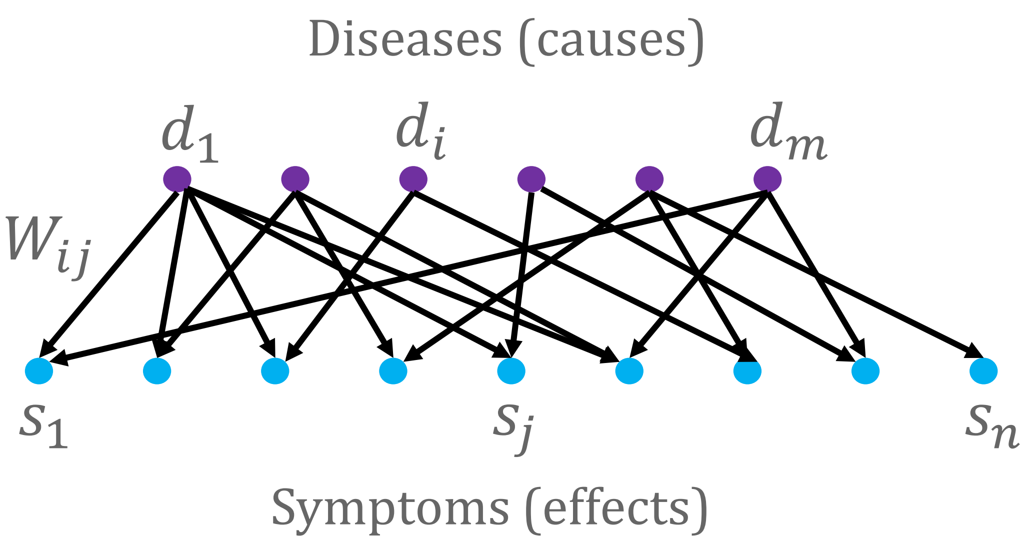
$$\hat{x}_1 \sim p(x_1)$$
$$\hat{x}_2 \sim p(x_2)$$
$$\hat{x}_3 \sim p(x_3)$$
$$\hat{x}_4 \sim p(x_4 | \hat{x}_1, \hat{x}_2, \hat{x}_3)$$
$$\hat{x}_5 \sim p(x_5 | \hat{x}_1, \hat{x}_3)$$

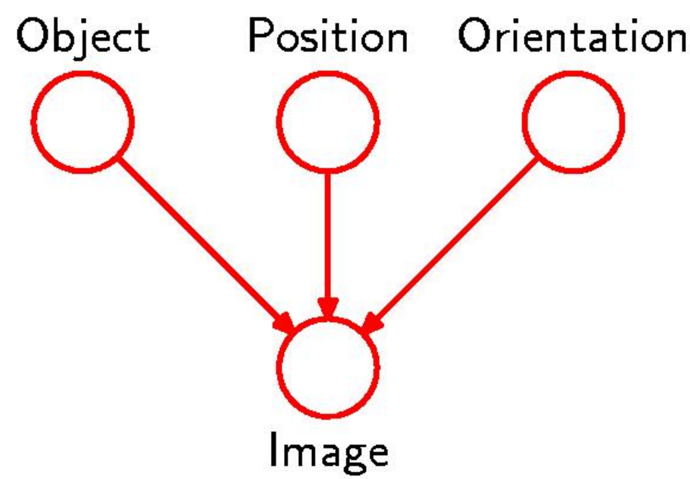
The parent variables are set to their sampled values

To obtain a sample from the marginal distribution, e.g. $p(x_2, x_5)$, sample from the full joint distribution, retain $\hat{x}_2, \hat{x}_5$, discard the remaining values.

The latent-variable paradigm

More often than not, we need to model part of the data that is **not observable**. We already saw examples of this:





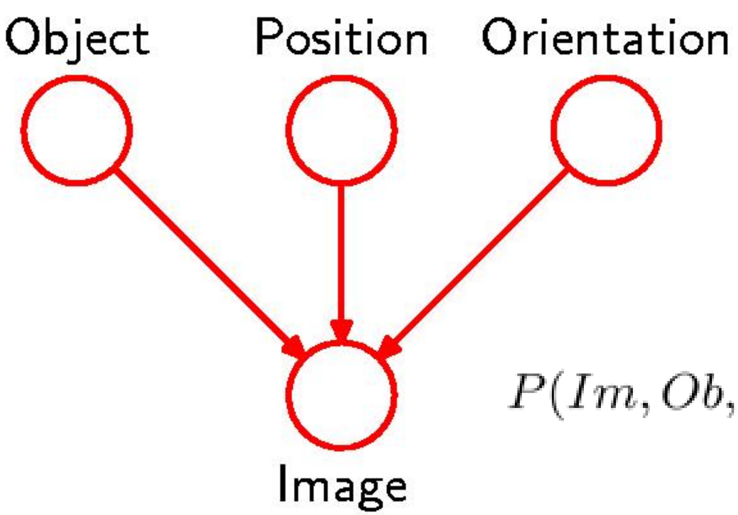
This is also a natural way to extract **features/representation**: the latent variables contain "meaningful" information.

The latent-variable paradigm: deep learning

Higher-up nodes will typically represent **latent** (hidden) random variables.

The role of latent variables is to allow modeling a **complicated** distribution over observed variables **constructed** from **simpler** conditional distributions.

Latent-variable model of image



Object identity, position, and orientation have independent *prior probabilities*.

Image has probability distr that depends on object identity, position, and orientation (*conditional distribution/likelihood*).

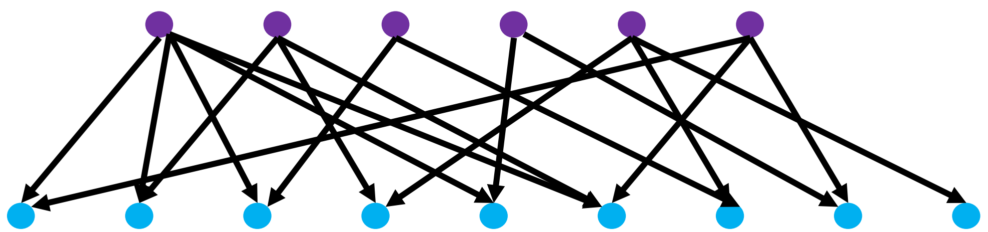
$$P(Im, Ob, Po, Or) = \underbrace{P(Im|Ob, Po, Or)}_{\text{Likelihood}}\underbrace{P(Ob)P(Po)P(Or)}_{\text{Prior}}$$

Likelihood and prior are modeled by parametric distribution whose parameters are fitted throughout training.

Examples: single-layer latent-variable Bayesian networks

Simple, but powerful paradigm:
single-layer Bayesian networks, where top nodes are latent.

Latent variables Z



Observable variables X

$$p_\theta(X, Z) = p_\theta(Z)\, p_\theta(X|Z)$$

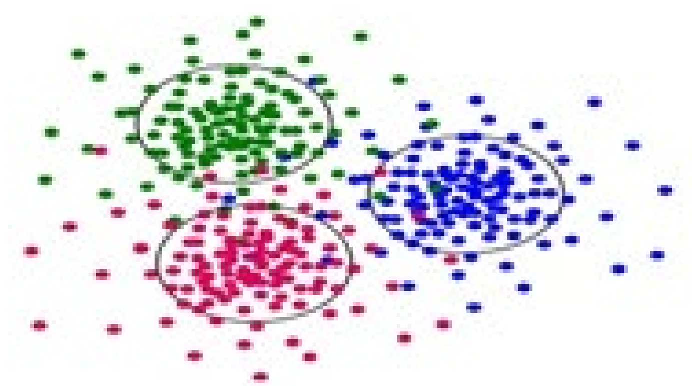
Example 1: Mixture distributions

Mixture models: observables = points; latent = clustering

To draw a sample (X,Z):

Sample Z from a categorial distr. on K components with parameters $\{\pi_i\}$

Sample X from the corresponding component in the mixture.



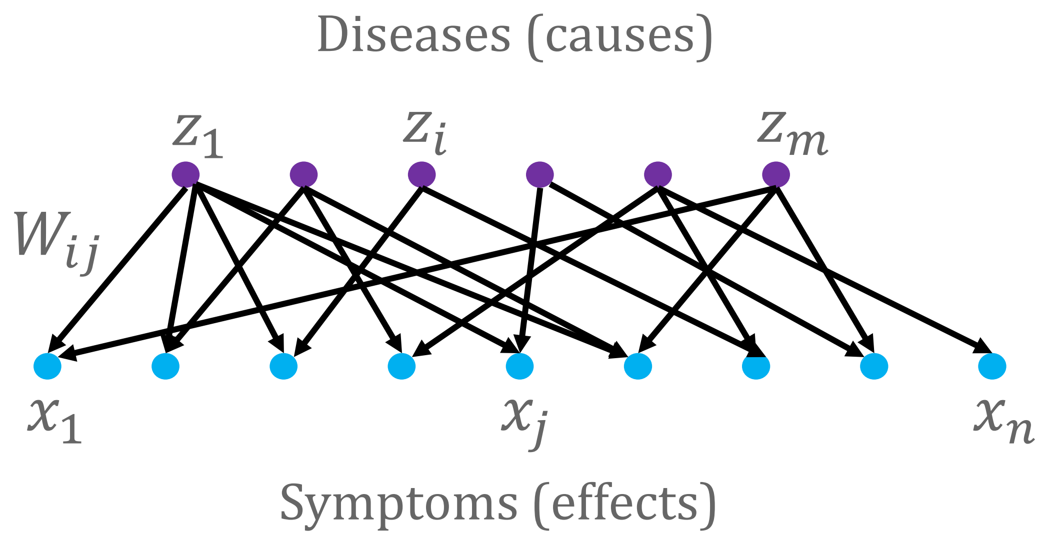
$$\forall k : \pi_k \geqslant 0 \qquad \sum_{k=1}^{K} \pi_k = 1$$

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \underbrace{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}_{\text{Component}}$$

Mixing coefficient

Example 2: Noisy-OR networks

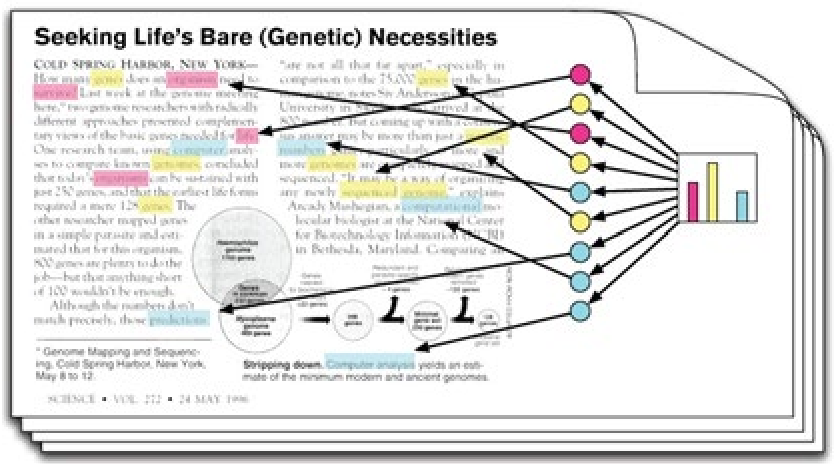
$x_i, z_j \in \{0,1\}$
$W_{ij} \geq 0$



- Sample each z_i is on **independently** with prob. ρ
- When z_i is on, it **activates** x_j with probability $1 - \exp(-W_{ij})$.
- x_j is **on** if one of z_i's **activates** x_j

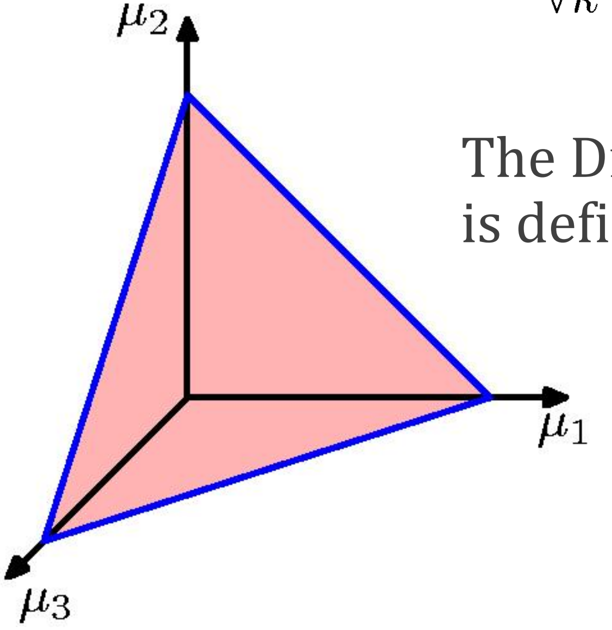
Example 3: Topic models (LDA)

Latent Dirichlet Allocation: famous model for modeling topic structure of documents of text. (Blei, Ng, Jordan '03)



Side-remark: Dirichlet Distribution

Consider a distribution over simplex, namely over points $\{\mu_i\}_{i=1}^{K}$



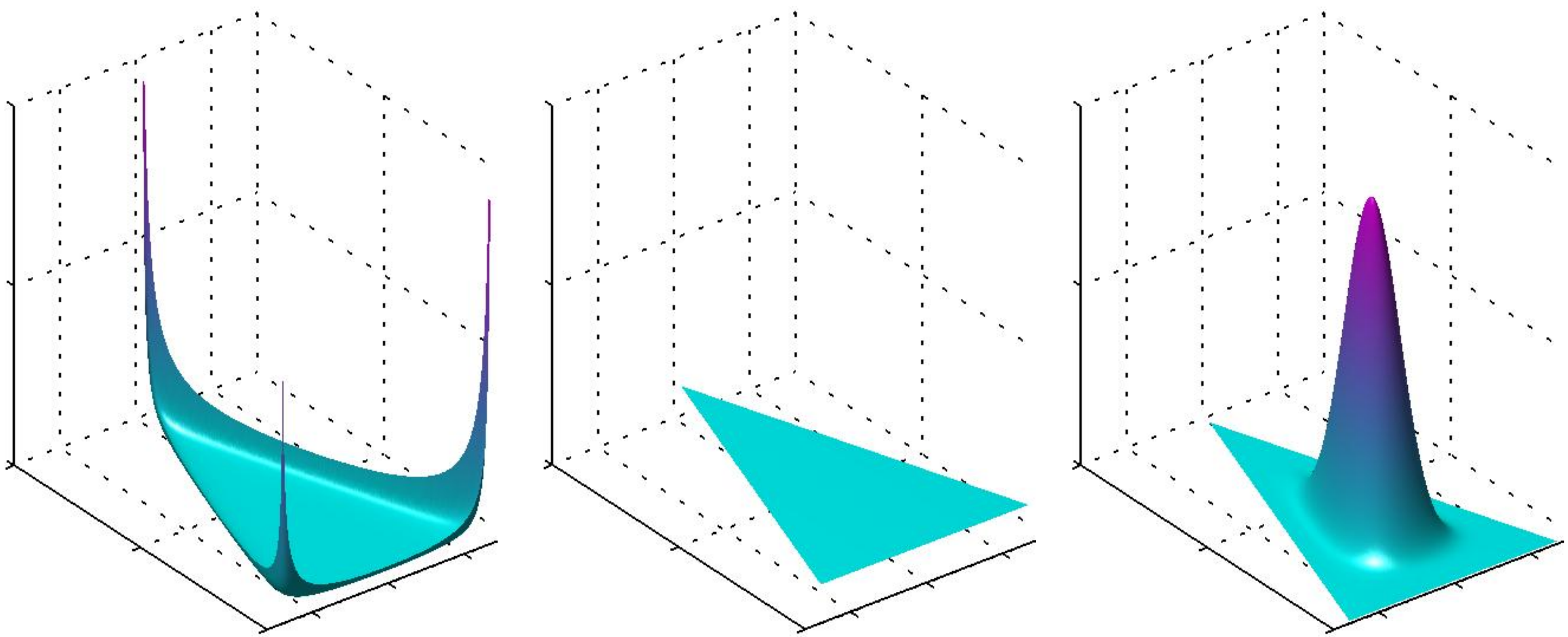
$$\forall k : \mu_k \geqslant 0 \quad \text{and} \quad \sum_{k=1}^{K} \mu_k = 1$$

The Dirichlet distribution (with params $\{\alpha_i \geq 0\}_{i=1}^{K}$) is defined as:

$$\mathrm{Dir}(\mu|\alpha) \propto \Pi_{k=1}^{K} \mu_k^{\alpha_k - 1}$$

Side-remark: Dirichlet Distribution

Plots of the Dirichlet distribution over three variables.



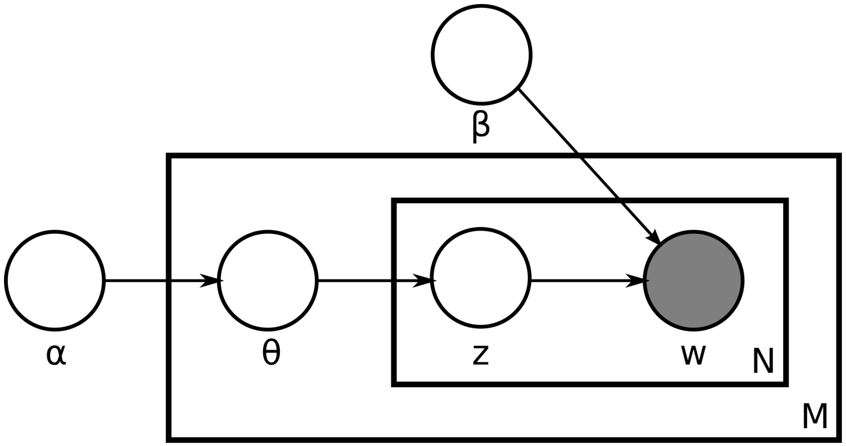
$\alpha_k = 10^{-1}$ $\alpha_k = 10^{0}$ $\alpha_k = 10^{1}$

Example 3: Topic models (LDA)

Defines a distribution over documents, involving K topics.

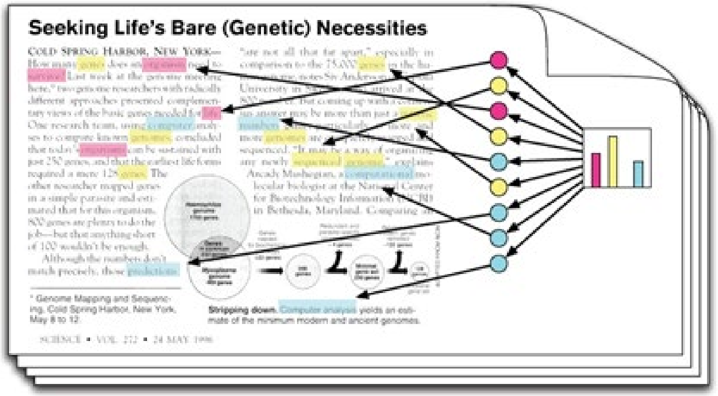
The **parameters** are: $\{\alpha_i\}_{i=1}^{K}$ (Dirichlet parameters) and matrix $\beta \in \mathbb{R}_+^{N\times K}$, where N is the size of the vocabulary.

The columns of β satisfy $\sum_{j=1}^{N} \beta_{ij} = 1$ (the distribution of words in a topic i)



To produce document:

- First, sample $\theta \sim \mathrm{Dir}(\cdot\,|\alpha)$: this will be the topic proportion vector for the document.
- Each word in the document is generated in order, independently.
- To generate word i:
  - Sample topic z_i with categorical distribution with parameters θ
  - Sample word w_i with categorical distribution with parameters β_{z_i}



Example 3: Topic models (LDA)

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

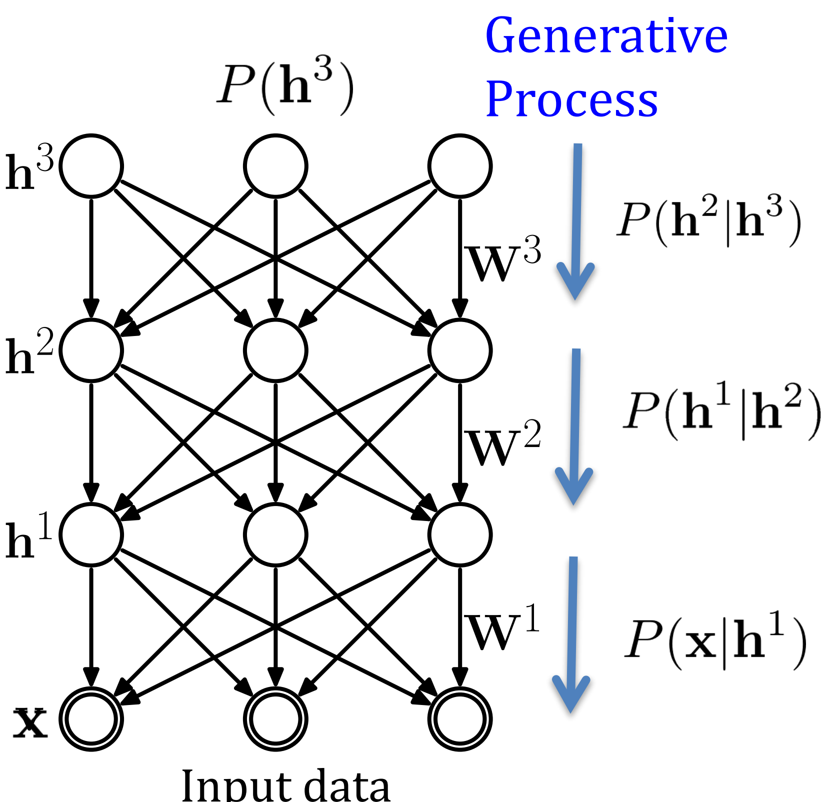
The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Example 4: Variational Autoencoder

Directed Bayesian network with Gaussian layers

$$p(\mathbf{x}|\boldsymbol{\theta}) = \sum_{\mathbf{h}^1,\ldots,\mathbf{h}^L} p(\mathbf{h}^L|\boldsymbol{\theta})p(\mathbf{h}^{L-1}|\mathbf{h}^L,\boldsymbol{\theta})\cdots p(\mathbf{x}|\mathbf{h}^1,\boldsymbol{\theta})$$

Each term may denote a complicated nonlinear relationship



Layers are parametrized as:

$$p(\boldsymbol{h}^{L-1}|\boldsymbol{h}^L,\boldsymbol{\theta}) = \mathcal{N}(\mu_{\boldsymbol{\theta}}(\boldsymbol{h}^L),\Sigma_\theta(\boldsymbol{h}^L))$$

Gaussians, means/covariances functions (e.g. neural net) of previous layer and model parameters θ.

Easy to sample!

Part II: variational methods

How do you learn a latent-variable Bayesian network?

The most obvious strategy: maximum likelihood estimation

Given data $x_1, x_2, \ldots, x_n$, solve the optimization problem

$$\max_{\theta \in \Theta} \sum_{i=1}^{n} \log p_\theta(x_i)$$

How would you evaluate $p_\theta(x_i)$?

Same partition function problem as in unnormalized models…

$$p_\theta(x) = \int_h p_\theta(h) p_\theta(x|h), \text{ e.g. } \sum_{\text{Diseases}} P(\text{Diseases}, \text{Symptoms})$$

Again, can be #P-hard to approximate.

Another way to view the problem

Sampling from Bayesian networks is easy.

But: sampling/approximating the posterior distribution P(Z|X) is **hard**:

$$P(\text{Diseases}, \text{Symptoms}) = P(\text{Diseases}) \;\; P(\text{Symptoms}|\text{Diseases})$$

Latent

Data

Simple, explicit

By Bayes rule, $P(\text{Diseases}|\text{Symptoms}) \propto P(\text{Diseases}, \text{Symptoms})$

Up to normalizing const, simple…

Complicated partition function:

$$\sum_{\text{Diseases}} P(\text{Diseases}, \text{Symptoms})$$

Variational methods for approximating posteriors

Gibbs variational principle: Let $p(z,x)$ be a joint distribution over latent variables and observables. Then,

$$p(z|x) = \underset{q(z|x):\text{distribution over } Z}{\operatorname{argmax}} \; \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x)$$

$$-H\big(q\,(z|x)\big) \quad - \quad \mathbb{E}_{z\sim q}[\log \; p(z,x)]$$

Furthermore, for every $q(z|x)$, we have

$$\log p(x) \;\; = -\big(-H\big(q\,(z|x)\big) - \mathbb{E}_{z\sim q}[\log \;\; p(z,x)]\big) + KL\big(q\,(z|x)\big|\big|\, p(z|x)\big)$$

Variational methods for partition functions

Gibbs variational principle: Let $p(z,x)$ be a joint distribution over latent variables and observables. Then,

$$p(z|x) = \underset{q(z|x):\text{distribution over } Z}{\operatorname{argmax}} \ \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x)$$

In fact, for every $q(z|x)$, we have

$$\log p(x) = KL(q\,(z|x) || \, p(z|x)) - \left(-H\big(q\,(z|x)\big) - \mathbb{E}_{z\sim q}[\log \ p(z,x)]\right)$$

Why:

$$0 \le KL(q\,(z|x) || \, p(z|x)) = \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} p(z|x)$$

$$= -H\big(q\,(z|x)\big) - \mathbb{E}_{q\,(z|x)} \log \frac{p(z,x)}{p(x)}$$

$$= -H\big(q\,(z|x)\big) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x) + \log p(x)$$

Equality is attained if and only if $KL(q\,(z|x) || \, p(z|x))$=0 i. e. $q\,(z|x) = p(z|x)$

Variational methods for approximating posteriors

Gibbs variational principle: Let $p(z, x)$ be a joint distribution over latent variables and observables. Then,

$$p(z|x) = \underset{q(z|x):\text{distribution over } Z}{\operatorname{argmax}} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z, x)$$

$$\log p(x) \;= -\big(-H\big(q\,(z|x)\big) - \mathbb{E}_{z\sim q}[\log\;p(z,x)]\big) + KL(q\,(z|x)||\,p(z|x))$$

Why is this useful?

(1) ***Approximating posteriors***: Instead of finding the argmax over **all** distributions over Z, we can maximize over some **simpler** parametric family $\mathcal{Q}$, i.e. we can solve

$$\max_{q(z|x)\in\mathcal{Q}} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z, x)$$

The argmax of the above distribution solves $\min_{q(Z|X)\in\mathcal{Q}} KL(q\,(z|x)||\,p(z|x))$.

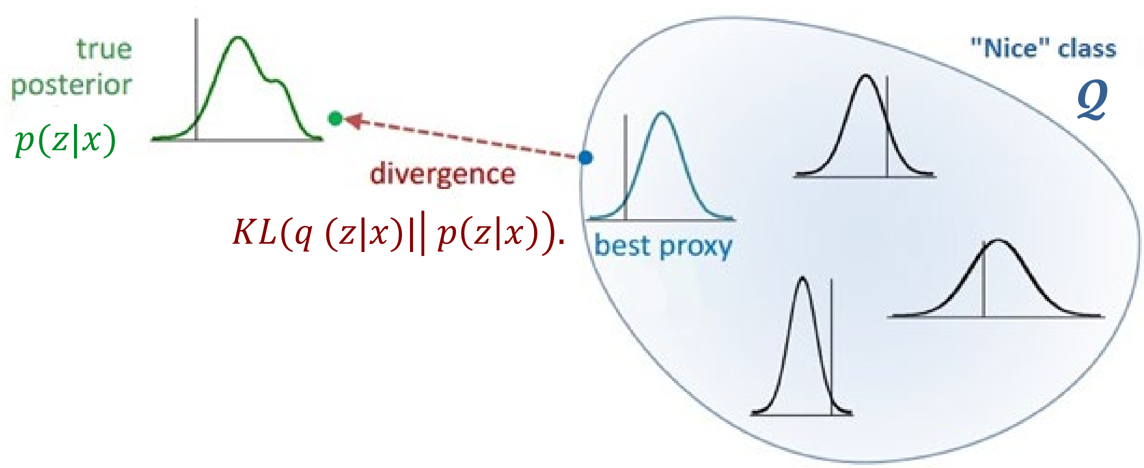
In other words, we are finding the **projection** of $p(z|x)$ onto $\mathcal{Q}$.

Variational methods for approximating posteriors

Gibbs variational principle: Let $p(z,x)$ be a joint distribution over latent variables and observables. Then,

$$p(z|x) = \underset{q(z|x):\text{distribution over } Z}{\operatorname{argmax}} \ \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x)$$

$$\log p(x) \quad = -\big(-H\big(q\,(z|x)\big) - \mathbb{E}_{z\sim q}[\log\ p(z,x)]\big) + KL\big(q\,(z|x)\big|\big|\,p(z|x)\big)$$



Variational methods for approximating posteriors

Gibbs variational principle: Let $p(z,x)$ be a joint distribution over latent variables and observables. Then,

$$p(z|x) = \underset{q(z|x):\text{distribution over } Z}{\operatorname{argmax}} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x)$$

$$\log p(x) \quad = -\left(-H\big(q\,(z|x)\big) - \mathbb{E}_{z\sim q}[p(z,x)]\right) + KL(q\,(z|x)||\,p(z|x))$$

Why is this useful?

(1) ***Approximating posteriors***: Instead of finding the argmax over **all** distributions over Z, we can maximize over some **simpler** parametric family Q, i.e. we can solve

$$\max_{q(z|x)\in Q} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x)$$

There are several common families Q that are used for which the above optimization is solveable – we will see **mean-field** families and **neural-net** parametrized families.

Variational methods for approximating posteriors

Gibbs variational principle: Let $p(z, x)$ be a joint distribution over latent variables and observables. Then,

$$p(z|x) = \underset{q(z|x):\text{distribution over } Z}{\operatorname{argmax}} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z, x)$$

$$\log p(x) \;\; = -\big(-H\big(q\,(z|x)\big) - \mathbb{E}_{z\sim q}[p(z, x)]\big) + KL(q\,(z|x) \| \, p(z|x))$$

Why is this useful?

(2) **Approximate likelihood-based learning**: provides a lower bound on $\log p(x)$ -- sometimes called the **ELBO (evidence lower bound)**, since

$$\log p(x) \geq \max_{q(z|x)\in \mathcal{Q}} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z, x)$$

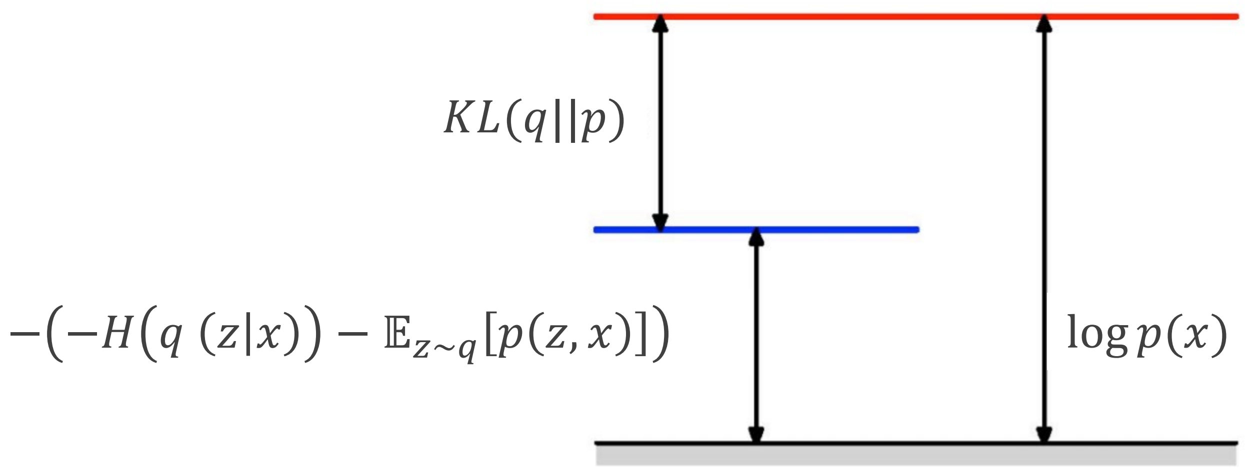
This will be useful for learning latent-variable directed models (stay tuned !).

Variational methods for approximating posteriors

Gibbs variational principle: Let $p(z, x)$ be a joint distribution over latent variables and observables. Then,

$$p(z|x) = \underset{q(z|x):\text{distribution over } Z}{\text{argmax}} \; \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z, x)$$

$$\log p(x) \;\; = -\big(-H\big(q\,(z|x)\big) - \mathbb{E}_{z\sim q}[p(z,x)]\big) + KL\big(q\,(z|x)\big|\big|\,p(z|x)\big)$$



Example: solving the mean-field relaxation via coordinate ascent

Inspiration from physics: consider the case where $\mathcal{Q}$ contains product distributions, that is, for every $q(\cdot\,|x) \in \mathcal{Q}$:

$$q(z|x) = \Pi_{i=1}^{d} q_i(z_i|x).$$

Consider updating a **single** coordinate of the mean-field distribution, that is keep $q_{-i}\,(z_i|x)$ fixed and optimize for $q_i\,(z_i|x)$. We have:

$$\min_{q(Z|X)\in\mathcal{Q}} KL(q\,(z|x)||\,p(z|x)) = \min_{q(Z|X)\in\mathcal{Q}} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x)$$

$$= \min_{q(Z|X)\in\mathcal{Q}} \sum_i \mathbb{E}_{q_i\,(z_i|x)} \log q_i\,(z_i|x) - \mathbb{E}_{q_i\,(z_i|x)}\left[\mathbb{E}_{q_{-i}(z_{-i}|x)} \log p\,(z_i, z_{-i}, x)\right]$$

$$= \min_{q(Z|X)\in\mathcal{Q}} \mathbb{E}_{q_i\,(z_i|x)} \log q_i\,(z_i|x) - \mathbb{E}_{q_i\,(z_i|x)}[\log \tilde{p}\,(z_i, x)] + C$$

Renormalize to make it a distribution

Example: solving the mean-field relaxation via coordinate ascent

Inspiration from physics: consider the case where Q contains product distributions, that is, for every $q(\cdot\,|x) \in Q$:

$$q(z|x) = \Pi_{i=1}^{d} q_i(z_i|x).$$

Consider updating a **single** coordinate of the mean-field distribution, that is keep $q_{-i}\,(z_i|x)$ fixed, and optimize for $q_i\,(z_i|x)$. We have:

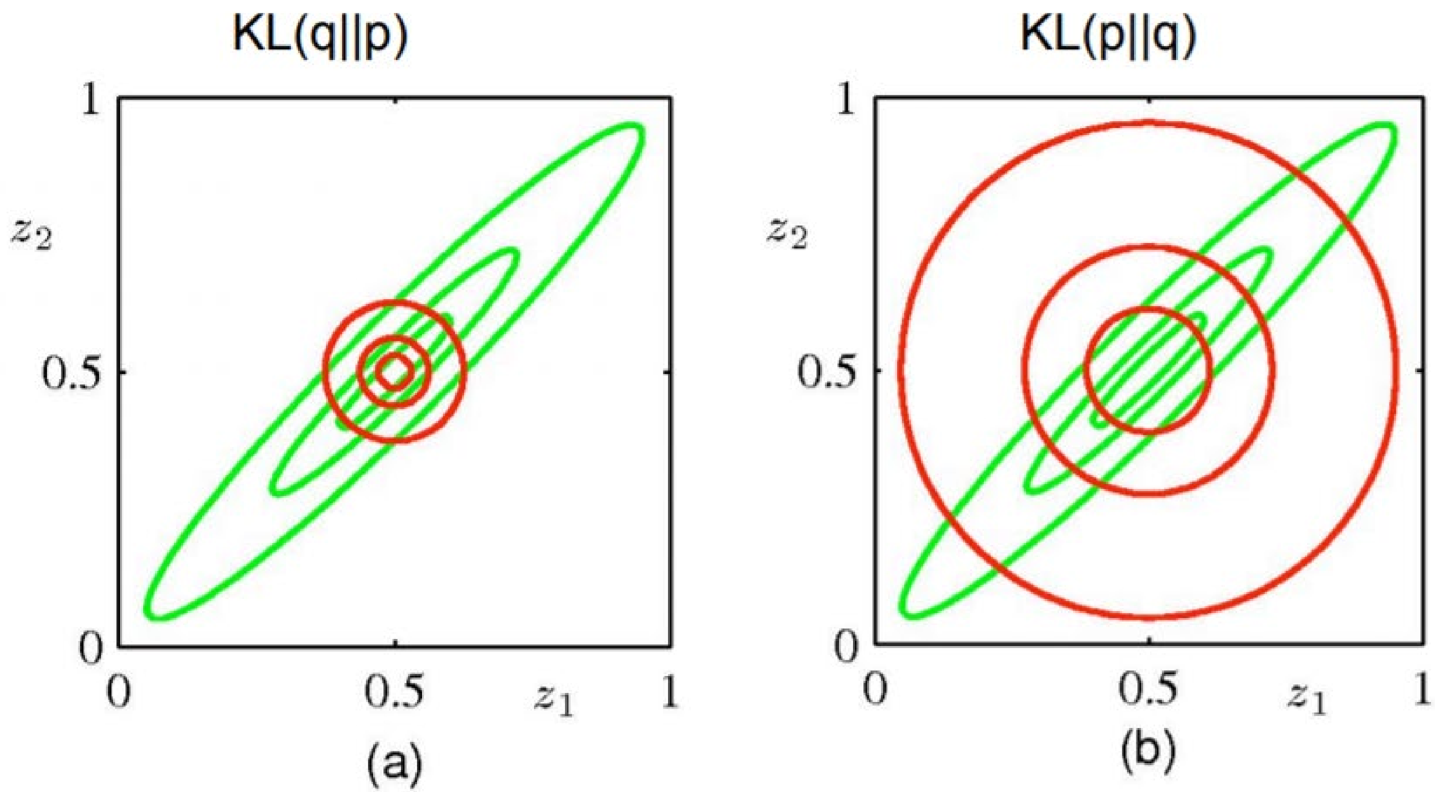
$$\min_{q(Z|X)\in Q} KL(q\,(z|x)||\,p(z|x)) = \min_{q(Z|X)\in Q} \mathbb{E}_{q\,(z|x)} \log q\,(z|x) - \mathbb{E}_{q\,(z|x)} \log p\,(z,x)$$

$$= \min_{q(Z|X)\in Q} KL(q_i\,(z_i|x)||\,\tilde{p}(z_i,x)) + C$$

Optimum is $q_i\,(z_i|x) = \tilde{p}(z_i,x)$
$$= \frac{\mathbb{E}_{q_{-i}(z_{-i}|x)} \log p\,(z_i,z_{-i},x)}{\int_{z_i} \mathbb{E}_{q_{-i}(z_{-i}|x)} \log p\,(z_i,z_{-i},x)}$$

Coordinate ascent: iterate above updates!

A tale of two KL divergences



Approximation is too compact.

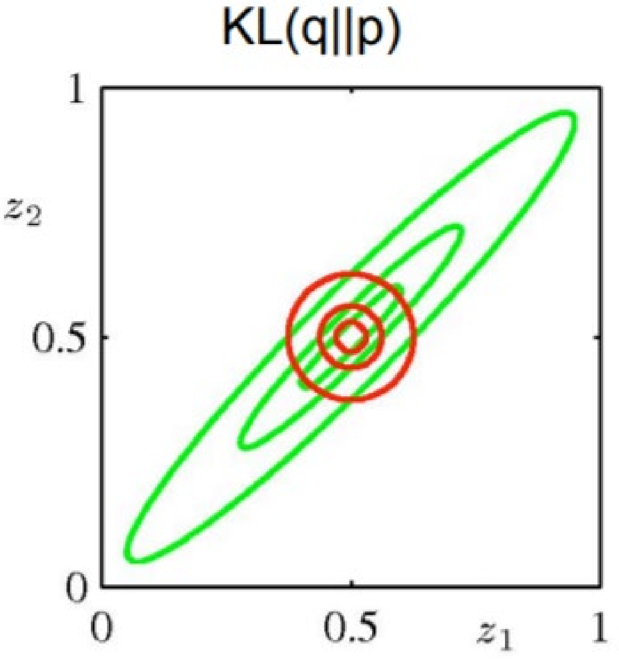
Approximation is too spread.

The “variational” KL divergence

$$\mathrm{KL}(q||p) = -\int q(\mathbf{Z}) \ln \frac{p(\mathbf{Z})}{q(\mathbf{Z})} \mathrm{d}\mathbf{Z}.$$

There is a large positive contribution to the KL divergence from regions of Z space in which:
- p(Z) is near zero
- unless q(Z) is also close to zero.

Minimizing KL(q||p) leads to distributions q(Z) that avoid regions in which p(Z) is small.



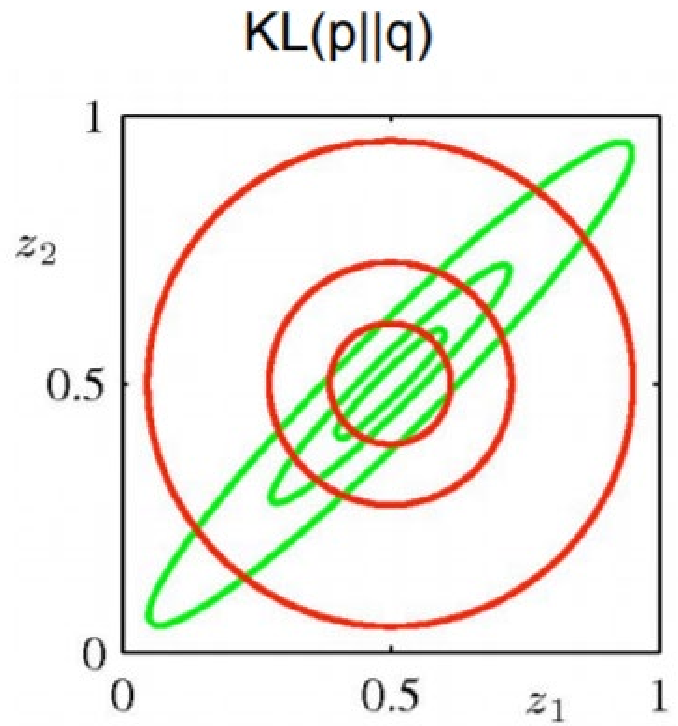
(a)

The "maximum likelihood" KL divergence

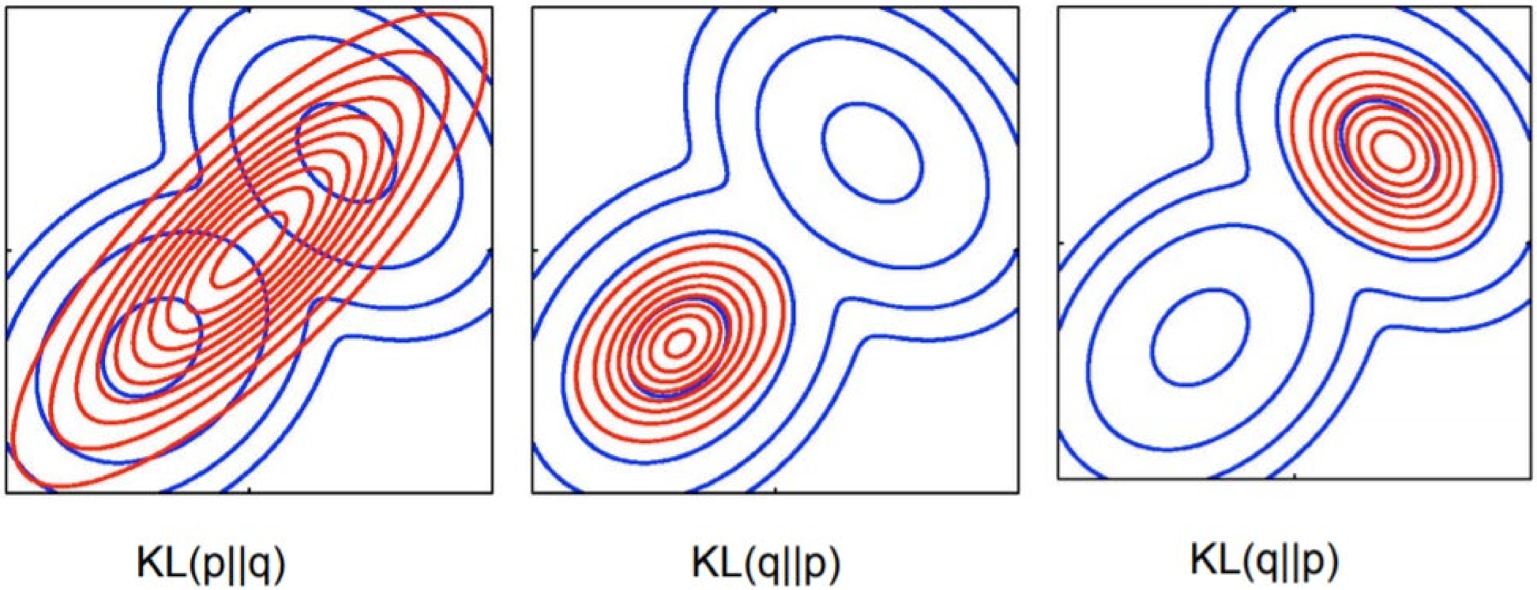
$$\mathrm{KL}(p||q) = -\int p(\mathbf{Z}) \ln \frac{q(\mathbf{Z})}{p(\mathbf{Z})} \mathrm{d}\mathbf{Z}.$$

There is a large positive contribution to the KL divergence from regions of Z space in which:
- q(Z) is near zero,
- unless p(Z) is also close to zero.

Minimizing KL(p||q) leads to distributions q(Z) that are nonzero in regions where p(Z) is nonzero.



What happens when distribution class for Q is not rich enough?



Blue contours show bimodal distribution, red contours single Gaussian distribution that best approximates it.

KL(q||p) will tend to find a single mode, whereas KL(p||q) will average across all of the modes.

Learning single-layer latent-variable Bayesian networks using variational methods

Rewriting the maximum likelihood objective

Recall maximum likelihood objective:

Given data $x_1, x_2, \ldots, x_n$, solve the optimization problem

$$\max_{\theta \in \Theta} \sum_{i=1}^{n} \log p(x_i)$$

Latent variables: we will use the Gibbs variational principle again!

$$\log_\theta \ p(x) = \max_{q(z|x)\text{: distribution over } z} H(q(z|x)) + \mathbb{E}_{q(z|x)}[\log p_\theta(x, z)]$$

Hence, MLE objective can be written as double maximization:

$$\max_{\theta \in \Theta} \max_{\{q_i(z|x_i)\}} \sum_{i=1}^{n} H(q_i(z|x_i)) + \mathbb{E}_{q_i(z|x_i)}[\log p_\theta(x_i, z)]$$

Expectation-maximization/ variational inference

The canonical algorithm for learning a single-layer latent-variable Bayesian network is an iterative algorithm as follows.

Consider the max-likelihood objective, rewritten as in the previous slide:

$$\max_{\theta \in \Theta} \max_{\{q_i(z|x_i) \in \mathcal{Q}\}} \sum_{i=1}^{n} H(q_i(z|x_i)) + \mathbb{E}_{q_i(z|x_i)}[\log p_\theta(x_i, z)]$$

Algorithm maintains iterates $\theta^t, \{q_i^t(z|x_i)\}$, and updates them iteratively

(1) **Expectation (E)-step**:

Keep θ^t fixed, set $\{q_i^{t+1}(z|x_i) \in \mathcal{Q}\}$, s.t. they maximize the objective above.

(2) **Maximization (M)-step**:

Keep $\{q_i^t(z|x_i)\}$ fixed, set θ^{t+1} s.t. it maximizes the objective above.

Clearly, every step cannot make the objective worse!

Does *not* mean it converges to global optimum – could, e.g. get stuck in a local minimum.

Expectation-maximization/ variational inference

The canonical algorithm for learning a single-layer latent-variable Bayesian network is an iterative algorithm as follows.

Consider the max-likelihood objective, rewritten as in the previous slide: $$\max_{\theta\in\Theta} \max_{q_i(z|x_i)} \sum_{i=1}^{n} H(q_i(z|x_i)) + \mathbb{E}_{q_i(z|x_i)}[\log p_\theta(x_i,z)]$$

Algorithm maintains iterates $\theta^t, q_i^t(z|x_i)$, and updates them iteratively

(1) **Expectation step**:

Keep θ^t and set $q_i^{t+1}(z|x_i)$, s.t. they maximize the objective above.

If the class is infinitely rich, the optimum is $q_i^{t+1}(z|x_i) = p_{\theta^t}(z|x_i)$

This is called **expectation-maximization (EM)**.
If class is not infinitely rich, it's called **variational inference**.

Example

<u>Mixture of spherical Gaussians</u>

Consider a mixture of K Gaussians with unknown means $p = \sum_{i=1}^{K} \frac{1}{K} \mathcal{N}(\mu_i, I_d)$

Let's try to calculate the E and M steps.

E-step: the optimal $q_i^{t+1}(z|x_i)$ is $p_{\theta^t}(z|x_i)$. Can we calculate this?

By Bayes rule, $p_{\theta^t}(z = k|x_i) \propto p(x_i|z = k) \propto e^{-\left|\left|x_i - \mu_k^t\right|\right|^2}$

Writing out the normalizing constant, we have

$$p_{\theta^t}(z = k|x_i) = \frac{e^{-\left|\left|x_i - \mu_k^t\right|\right|^2}}{\sum_{k'} e^{-\left|\left|x_i - \mu_{k'}^t\right|\right|^2}}$$

"Soft" version of assigning point to nearest cluster

Example

<u>Mixture of spherical Gaussians</u>

Consider a mixture of K Gaussians with unknown means $p = \sum_{i=1}^{K} \frac{1}{K} \mathcal{N}(\mu_i, I_d)$

Let's try to calculate the E and M steps.

M-step: given a quess $q_i^t(z|x_i)$, we can rewrite the maximization for θ as:

$$\max_{\theta \in \Theta} \sum_{i=1}^{n} H\big(q_i^t(z|x_i)\big) + \mathbb{E}_{q_i^t(z|x_i)}[\log p_\theta(x_i, z)]$$

$$= \mathbb{E}_{q_i^t(z|x_i)}\,[\cancel{\log p(z)} + \log p_\theta(x|z)]$$

Doesn't depend on θ

$$\mathbb{E}_{q_i^t(z|x_i)}[\,\log p_\theta(x|z)]$$

Example

Mixture of spherical Gaussians

Consider a mixture of K Gaussians with unknown means $p = \sum_{i=1}^{K} \frac{1}{K} \mathcal{N}(\mu_i, I_d)$

Let's try to calculate the E and M steps.

M-step: given a quess $q_i^t(z|x_i)$, we can rewrite the maximization for θ as:

$$\max_{\theta} \ \mathbb{E}_{q_i^t(z|x_i)}[\ \log p_\theta(x|z)] = \max_{\theta} \ -\sum_{i=1}^{n} \sum_{k=1}^{K} q_i^t(z=k|x_i)||x_i - \mu_k\,||^2$$

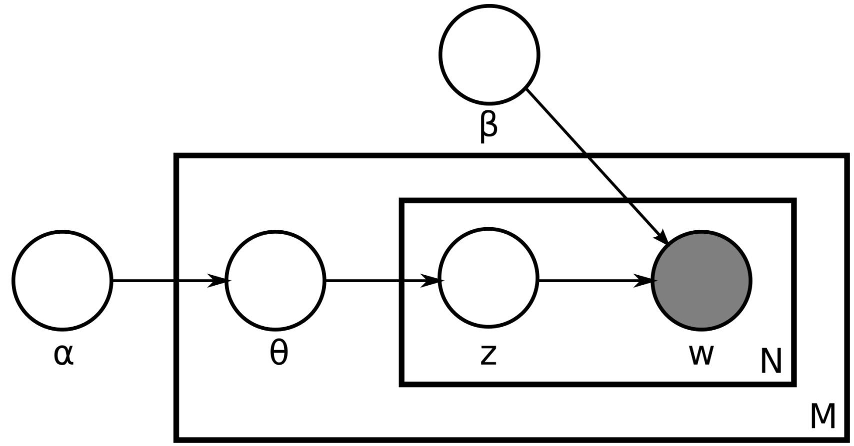
Setting the derivative wrt to μ_k to 0, we have:

$$\mu_k^{t+1} = \sum_{i=1}^{n} \frac{e^{-\left|\left|x_i - \mu_k^t\right|\right|^2}}{\sum_{k'} e^{-\left|\left|x_i - \mu_{k'}^t\right|\right|^2}} x_i$$

Average points, weighing nearby points more

Examples

2: Latent Dirichlet Allocation

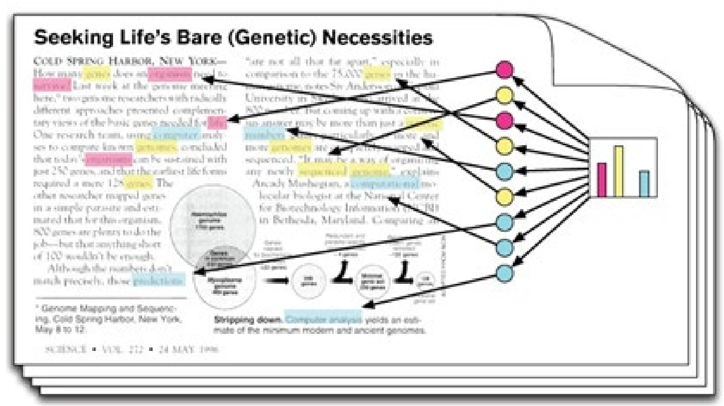


The **parameters** are: $\{\alpha_i\}_{i=1}^{K}$ (Dirichlet parameters) and matrix $\beta \in \mathbb{R}_+^{N\times K}$, where N is the size of the vocabulary.

The columns of β satisfy $\sum_{j=1}^{N} \beta_{ij} = 1$ (the distribution of words in a topic i)

To produce document:

- First, sample $\theta \sim \mathrm{Dir}(\cdot\,|\alpha)$: this will be the topic proportion vector for the document.
- Each word in the document is generated in order, independently.
- To generate word i:
  - Sample topic z_i with categorical distribution with parameters θ
  - Sample word w_i with categorical distribution with parameters β_{z_i}



Examples

The E-step cannot be done in closed form:

$$p(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K} \mid w_{1:D,1:N}, \alpha, \eta) = \frac{p(\vec{\theta}_{1:D}, \vec{z}_{1:D}, \vec{\beta}_{1:K} \mid \vec{w}_{1:D}, \alpha, \eta)}{\int_{\vec{\beta}_{1:K}} \int_{\vec{\theta}_{1:D}} \sum_{\vec{z}} p(\vec{\theta}_{1:D}, \vec{z}_{1:D}, \vec{\beta}_{1:K} \mid \vec{w}_{1:D}, \alpha, \eta)}$$

(In fact, can be shown to be #P-hard to perform in the worst case.)

The variational family to approximate the posterior is commonly chosen to be a mean-field family:

$$q(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K}) = \prod_{k=1}^{K} q(\vec{\beta}_k \mid \vec{\lambda}_k) \prod_{d=1}^{D} \left(q(\vec{\theta}_d \mid \vec{\gamma}_d) \prod_{n=1}^{N} q(z_{d,n} \mid \vec{\phi}_{d,n}) \right)$$

- **Probability of topic** z **given document** d: $q(\theta_d \mid \gamma_d)$
  Each document has its own Dirichlet prior γ_d
- **Probability of word** w **given topic** z: $q(\beta_z \mid \lambda_z)$
  Each topic has its own Dirichlet prior λ_z
- **Probability of topic assignment to word** $w_{d,n}$: $q(z_{d,n} \mid \varphi_{d,n})$
  Each word position *word[d][n]* has its own prior $\varphi_{d,n}$

Examples

$$q(\vec{\theta}_{1:D}, z_{1:D,1:N}, \vec{\beta}_{1:K}) = \prod_{k=1}^{K} q(\vec{\beta}_k \mid \vec{\lambda}_k) \prod_{d=1}^{D} \left(q(\vec{\theta}_{d d} \mid \vec{\gamma}_d) \prod_{n=1}^{N} q(z_{d,n} \mid \vec{\phi}_{d,n}) \right)$$

- **Probability of topic** z **given document** d: $q(\theta_d \mid \gamma_d)$
  Each document has its own Dirichlet prior γ_d
- **Probability of word** w **given topic** z: $q(\beta_z \mid \lambda_z)$
  Each topic has its own Dirichlet prior λ_z
- **Probability of topic assignment to word** $w_{d,n}$: $q(z_{d,n} \mid \varphi_{d,n})$
  Each word position *word[d][n]* has its own prior $\varphi_{d,n}$

One iteration of mean field variational inference for LDA

(1) For each topic k and term v:

(8) $$\lambda_{k,v}^{(t+1)} = \eta + \sum_{d=1}^{D} \sum_{n=1}^{N} 1(w_{d,n} = v)\phi_{n,k}^{(t)}.$$

(2) For each document d:

(a) Update γ_d:

(9) $$\gamma_{d,k}^{(t+1)} = \alpha_k + \sum_{n=1}^{N} \phi_{d,n,k}^{(t)}.$$

(b) For each word n, update $\vec{\phi}_{d,n}$:

(10) $$\phi_{d,n,k}^{(t+1)} \propto \exp\left\{ \Psi(\gamma_{d,k}^{(t+1)}) + \Psi(\lambda_{k,w_n}^{(t+1)}) - \Psi(\textstyle\sum_{v=1}^{V} \lambda_{k,v}^{(t+1)}) \right\},$$

where Ψ is the digamma function, the first derivative of the $\log \Gamma$ function.

Parameter updates:

$$\beta_{ij} \propto \sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi_{dni} w_{dn}^{j}.$$

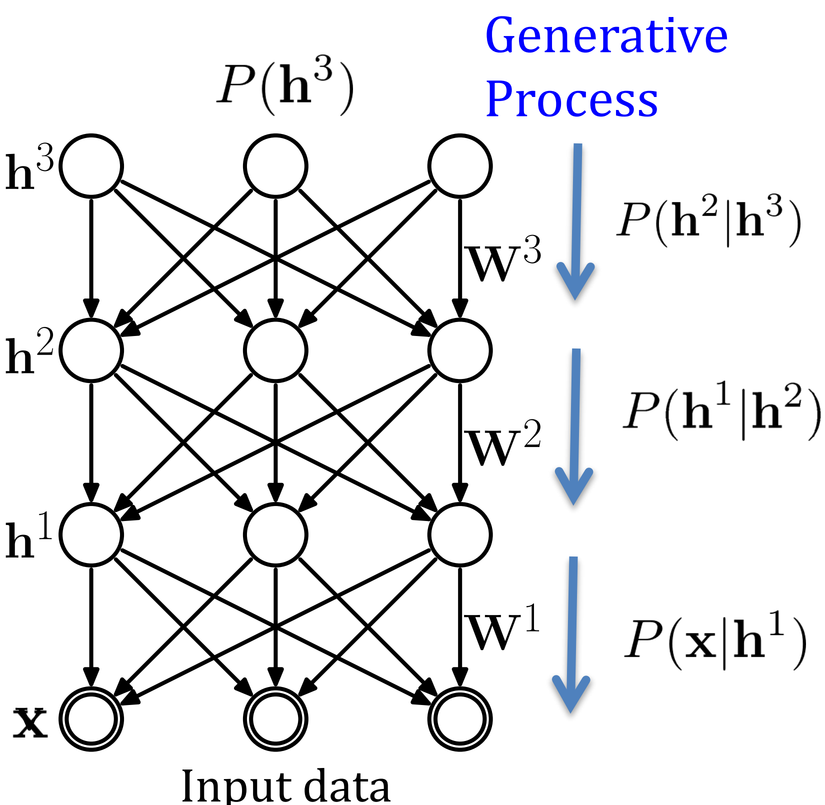
Learning variational autoencoders using variational methods

Variational autoencoders

Directed Bayesian network with Gaussian layers

$$p(\mathbf{x}|\boldsymbol{\theta}) = \sum_{\mathbf{h}^1,\ldots,\mathbf{h}^L} p(\mathbf{h}^L|\boldsymbol{\theta})p(\mathbf{h}^{L-1}|\mathbf{h}^L,\boldsymbol{\theta})\cdots p(\mathbf{x}|\mathbf{h}^1,\boldsymbol{\theta})$$

Each term may denote a complicated nonlinear relationship



Typically, directed layers are parametrized as:

$$p(\boldsymbol{h}^{L-1}|\boldsymbol{h}^L,\boldsymbol{\theta}) = \mathcal{N}(\mu_{\boldsymbol{\theta}}(\boldsymbol{h}^L),\Sigma_\theta(\boldsymbol{h}^L))$$

Gaussians, means/covariances functions (e.g. one-layer neural net) of previous layer and model parameters θ.

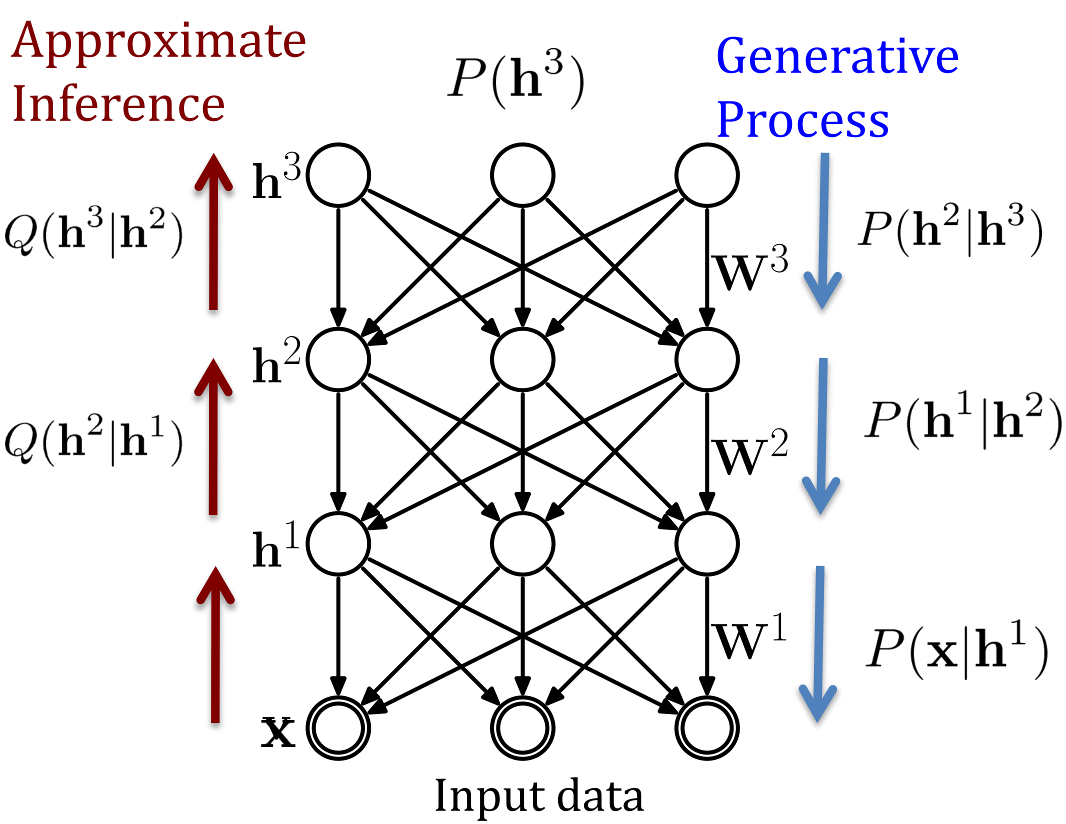
Easy to sample!

Approximate posterior family

The posterior family is defined in terms of an analogous factorization:

$$q(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{\theta}) = q(\boldsymbol{h}^1|\boldsymbol{x},\boldsymbol{\theta})q(\boldsymbol{h}^2|\boldsymbol{h}^1,\boldsymbol{\theta}) \ldots q(\boldsymbol{h}^L|\boldsymbol{h}^{L-1},\boldsymbol{\theta})$$

Each term may denote a complicated nonlinear relationship



Typically, directed layers are parametrized as:

$$q\left(\boldsymbol{h}^l\middle|\boldsymbol{h}^{l-1},\boldsymbol{\theta}\right) = \mathcal{N}\left(\mu_{\boldsymbol{\theta}}\left(\boldsymbol{h}^{l-1}\right),\Sigma_{\boldsymbol{\theta}}\left(\boldsymbol{h}^{l-1}\right)\right)$$

Means/covariances fns (e.g. one-layer neural net) of previous layer and parameters $\boldsymbol{\theta}$.

How to train?

Max-likelihood can be written as:

$$\max_{\theta \in \Theta} \max_{\{q_\theta(h^L|x)\}} \sum_x \mathbb{E}_{q_\theta(h^L|x)} \log \frac{p_\theta(x, h^L)}{q_\theta(h^L|x)}$$

We want to be able to take gradients in θ

The problem: the expectation is with respect to $q_\theta(h^L|x)$, which depends on the variables we are taking a derivative with respect to.

Observation: a derivative of the type $\nabla_\theta \mathbb{E}_p f(\theta)$ is easy to approximate if p does not depend on θ:

$$\nabla_\theta \mathbb{E}_p f(\theta) = \mathbb{E}_p \nabla_\theta f(\theta)$$

θ_i: iid samples from p

Exchange only works if p doesn't dep on θ

$$\approx \frac{1}{N} \sum_i \nabla_{\theta_i} f(\theta_i)$$

How to train?

Max-likelihood can be written as:

$$\max_{\theta\in\Theta} \max_{\{q_\theta(h^L|x)\}} \sum_x \mathbb{E}_{q_\theta(h^L|x)} \log \frac{p_\theta(x, h^L)}{q_\theta(h^L|x)}$$

As usual: we need to be able to take gradients in θ

Remark*:* there is a common Monte Carlo estimator of $\nabla_\theta \mathbb{E}_{p(\theta)} f(\theta)$ as well but it typically has high variance:

$$\nabla_\theta \mathbb{E}_{p(\theta)} f(\theta) = \int \nabla_\theta f(\theta)\, p(\theta) d\theta + \int f(\theta)\, \nabla_\theta p(\theta) d\theta$$

$$= \int \nabla_\theta f(\theta)\, p(\theta) d\theta + \int f(\theta) \frac{p(\theta)}{p(\theta)} \nabla_\theta p(\theta) d\theta$$

$$= \int \nabla_\theta f(\theta)\, p(\theta) d\theta + \int f(\theta) \nabla_\theta \log p(\theta)\, p(\theta) d\theta$$

$$= \mathbb{E}_{p(\theta)}[\nabla_\theta f(\theta) + \ f(\theta) \nabla_\theta \log p(\theta)]$$

This term typically is high var.,

How to train?

Max-likelihood can be written as:

$$\max_{\theta \in \Theta} \max_{\{q_\theta(h^L|x)\}} \sum_x \mathbb{E}_{q_\theta(h^L|x)} \log \frac{p_\theta(x, h^L)}{q_\theta(h^L|x)}$$

As usual: we need to be able to take gradients in θ

The solution: write the expectation $\mathbb{E}_{q_\theta(h^L|x)} \log \frac{p_\theta(x,h^L)}{q_\theta(h^L|x)}$ as an expectation over a distribution not dependent on θ.

Kingma-Welling '13: reparametrization trick!

Main idea: a sample from $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be generated as follows

Sample $\mathbf{x} \sim \mathcal{N}(0, \mathrm{I})$.

Output $\boldsymbol{y} = \boldsymbol{\mu} + \boldsymbol{\Sigma}^{1/2} \boldsymbol{x}$.

How to train?

Max-likelihood can be written as:

$$\max_{\theta\in\Theta} \max_{\{q_\theta(h^L|x)\}} \sum_x \mathbb{E}_{q_\theta(h^L|x)} \log \frac{p_\theta(x,h^L)}{q_\theta(h^L|x)}$$

As usual: we need to be able to take gradients in θ

Recall that $q(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{\theta}) = q(\boldsymbol{h}^1|\boldsymbol{x},\boldsymbol{\theta})q(\boldsymbol{h}^2|\boldsymbol{h}^1,\boldsymbol{\theta}) \ldots q(\boldsymbol{h}^L|\boldsymbol{h}^{L-1},\boldsymbol{\theta})$

where $q(\boldsymbol{h}^l|\boldsymbol{h}^{l-1},\boldsymbol{\theta}) = \mathcal{N}(\mu_{\boldsymbol{\theta}}(\boldsymbol{h}^{l-1}),\Sigma_{\boldsymbol{\theta}}(\boldsymbol{h}^{l-1}))$

To produce a sample from $q(\boldsymbol{h}|\boldsymbol{x},\boldsymbol{\theta})$, sample iid standard Gaussians $\boldsymbol{\epsilon}_1,\boldsymbol{\epsilon}_2,\ldots,\boldsymbol{\epsilon}_L$. Set

$$\mathbf{h}^\ell\left(\boldsymbol{\epsilon}^\ell,\mathbf{h}^{\ell-1},\boldsymbol{\theta}\right) = \boldsymbol{\Sigma}(\mathbf{h}^{\ell-1},\boldsymbol{\theta})^{1/2}\boldsymbol{\epsilon}^\ell + \boldsymbol{\mu}(\mathbf{h}^{\ell-1},\boldsymbol{\theta})$$

Using the reparametrization trick

Max-likelihood can be written as:

$$\max_{\theta\in\Theta} \max_{\{q_\theta(h)\}} \sum_x \mathbb{E}_{q_\theta(h|x)} \log \frac{p_\theta(x,h)}{q_\theta(h|x)}$$

We can hence write the gradient wrt to θ :

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{h}\sim q(\mathbf{h}|\mathbf{x},\boldsymbol{\theta})} \left[\log \frac{p(\mathbf{x},\mathbf{h}|\boldsymbol{\theta})}{q(\mathbf{h}|\mathbf{x},\boldsymbol{\theta})} \right]$$

$$= \nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\epsilon}^1,\ldots,\boldsymbol{\epsilon}^L \sim \mathcal{N}(\mathbf{0},\boldsymbol{I})} \left[\log \frac{p(\mathbf{x},\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\boldsymbol{\theta})}{q(\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\mathbf{x},\boldsymbol{\theta})} \right]$$

$$= \mathbb{E}_{\boldsymbol{\epsilon}^1,\ldots,\boldsymbol{\epsilon}^L \sim \mathcal{N}(\mathbf{0},\boldsymbol{I})} \left[\nabla_{\boldsymbol{\theta}} \log \frac{p(\mathbf{x},\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\boldsymbol{\theta})}{q(\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\mathbf{x},\boldsymbol{\theta})} \right]$$

Using the reparametrization trick

Max-likelihood can be written as:

$$\max_{\theta\in\Theta} \max_{\{q_\theta(h)\}} \sum_x \mathbb{E}_{q_\theta(h|x)} \log \frac{p_\theta(x,h)}{q_\theta(h|x)}$$

We can hence write the gradient wrt to θ :

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{h}\sim q(\mathbf{h}|\mathbf{x},\boldsymbol{\theta})} \left[\log \frac{p(\mathbf{x},\mathbf{h}|\boldsymbol{\theta})}{q(\mathbf{h}|\mathbf{x},\boldsymbol{\theta})}\right] = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\epsilon}^1,\dots,\boldsymbol{\epsilon}^L\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})} \left[\log \frac{p(\mathbf{x},\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\boldsymbol{\theta})}{q(\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\mathbf{x},\boldsymbol{\theta})}\right]$$

$$= \mathbb{E}_{\boldsymbol{\epsilon}^1,\dots,\boldsymbol{\epsilon}^L\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})} \left[\nabla_{\boldsymbol{\theta}} \log \frac{p(\mathbf{x},\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\boldsymbol{\theta})}{q(\mathbf{h}(\boldsymbol{\epsilon},\mathbf{x},\boldsymbol{\theta})|\mathbf{x},\boldsymbol{\theta})}\right]$$

We can approximate the expectation by an empirical average as before.

For **fixed** $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, \dots, \boldsymbol{\epsilon}_L$: log p and log q are easy to take gradients of via backpropagating.

It's common to have **diagonal covariance mxs** for training efficiency.