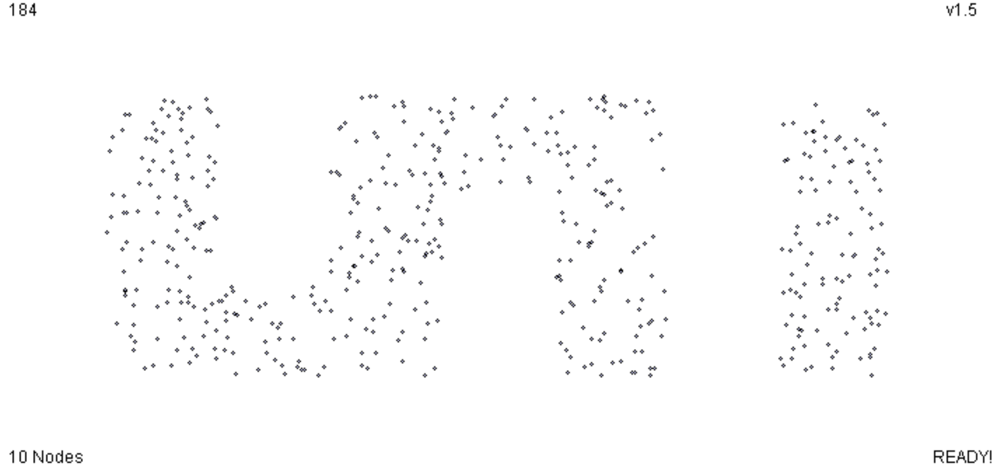
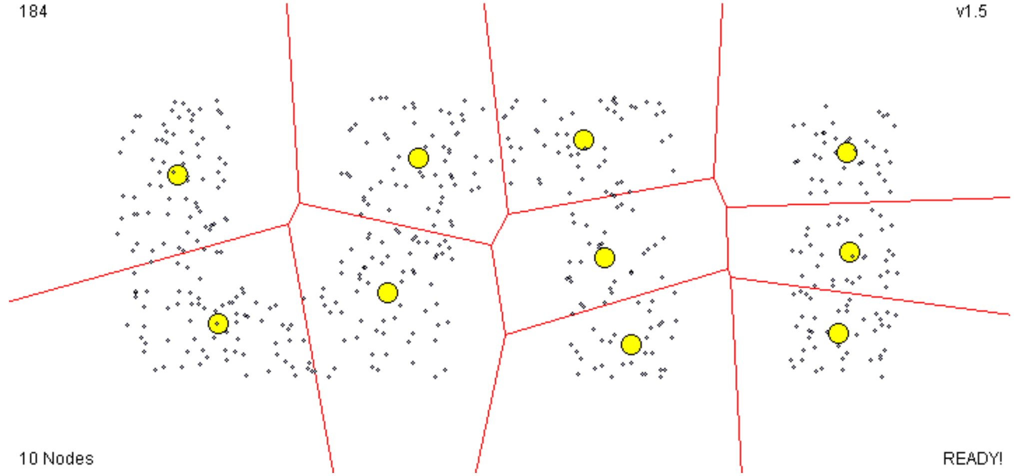
Unsupervised Learning

Marco Piastra

Vector quantization



Original data



Quantization (compression via prototypes)

The basic idea is to replace each real-valued vector $\mathbf{x} \in \mathbb{R}^d$ *with a discrete symbol* $\mathbf{w}_j \in \mathbb{R}^d$ *which belongs to a codebook of* k *prototypes* $\theta := \{\mathbf{w}_1, \ldots, \mathbf{w}_k\}$

Each data vector is encoded by using the index of the most similar prototype, where similarity is measured in terms, for instance, of Euclidean distance:

$$w(\mathbf{x}) := \mathrm{argmin}_{\mathbf{w}_j} \|\mathbf{x} - \mathbf{w}_j\|$$

For instance, part of mpeg4 and QuickTime (Apple) video compression algorithms work in this way and so does the Ogg Vorbis audio compression algorithm

k-means (*Generalized Lloyd's Algorithm – Vector quantization*)

Given a set $D := \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ *o f observations (i.e. vectors in* $\mathbb{R}^d$*)*
and a set $\theta := \{\mathbf{w}_1, \ldots, \mathbf{w}_k\}$ *of* k *<u>prototypes</u> (i.e. vectors in* $\mathbb{R}^d$*)*

Clustering problem: find an assignment function $w : \mathbb{R}^d \to W$
such that the objective (loss) function:

$$J(D, \theta) := \frac{1}{2} \sum_{i=1}^{N} \|\mathbf{x}_i - w(\mathbf{x}_i)\|^2$$

is minimized.

***k-means* algorithm:**

1) Position the *k* prototypes at random
2) Assign each observation to its closest prototype

$$w(\mathbf{x}_i) := \text{argmin}_{\mathbf{w}_j} \|\mathbf{x}_i - \mathbf{w}_j\|$$

3) Position each prototype at the *centroid* of the observations assigned to it

$$\mathbf{w}_j = \frac{1}{|D(\mathbf{w}_j)|} \sum_{D(\mathbf{w}_j)} \mathbf{x}_i \qquad \text{where} \quad D(\mathbf{w}_j) := \{\mathbf{x}_i \in D \mid w(\mathbf{x}_i) = \mathbf{w}_j\}$$

4) Unless no prototype was moved in step 3), go back to step 2)

This algorithm converges to a <u>local</u> minimum of $J(D, \theta)$

k-means (Generalized Lloyd's Algorithm – Vector quantization)

Why does the algorithm work: *alternate optimization (also 'coordinate descent')*

Step 2): Assign observations while keeping the k prototype fixed

$$w(\mathbf{x}_i) := \mathrm{argmin}_{\mathbf{w}_j} \|\mathbf{x}_i - \mathbf{w}_j\|$$

which minimizes each of the terms in $J(D,\theta) := \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{x}_i - w(\mathbf{x}_i)\|^2$

Step 3): Reposition the k prototypes while keeping the assignments fixed

$$J(D,\theta) := \frac{1}{2}\sum_{i=1}^{N} \|\mathbf{x}_i - w(\mathbf{x}_i)\|^2 = \frac{1}{2}\sum_{j}\sum_{D(\mathbf{w}_j)} (\mathbf{x}_i - \mathbf{w}_j)^2$$

$$\frac{\partial}{\partial \mathbf{w}_j} J(D,\theta) = \frac{\partial}{\partial \mathbf{w}_j}\frac{1}{2}\sum_{D(\mathbf{w}_j)} (\mathbf{x}_i - \mathbf{w}_j)^2 = \frac{\partial}{\partial \mathbf{w}_j}\frac{1}{2}\sum_{D(\mathbf{w}_j)} (\mathbf{x}_i - \mathbf{w}_j)^T(\mathbf{x}_i - \mathbf{w}_j)$$

$$= \frac{\partial}{\partial \mathbf{w}_j}\frac{1}{2}\sum_{D(\mathbf{w}_j)} (\mathbf{x}_i^2 + \mathbf{w}_j^2 - 2\,\mathbf{x}_i^T\,\mathbf{w}_j) = \sum_{D(\mathbf{w}_j)} (\mathbf{w}_j - \mathbf{x}_i)$$

then, by imposing $\frac{\partial}{\partial \mathbf{w}_j} J(D,\theta) = 0$ we obtain

$$\mathbf{w}_j = \frac{1}{|D(\mathbf{w}_j)|}\sum_{D(\mathbf{w}_j)} \mathbf{x}_i$$

k-means (*Generalized Lloyd's Algorithm – Vector quantization*)

Discussion of the *k-means algorithm*

a) At each step of the algorithm $J(D, \theta)$ cannot increase: only decrease or stay equal

b) The algorithm is a variant of a *gradient descent*, in which at each step the *gradient descent* is performed on one subset of variables only

c) It must reach a *fixed point*, where both gradients vanish

d) But the only guarantee is that the algorithm reaches a local minimum (*unless it gets stuck in a saddle point*)

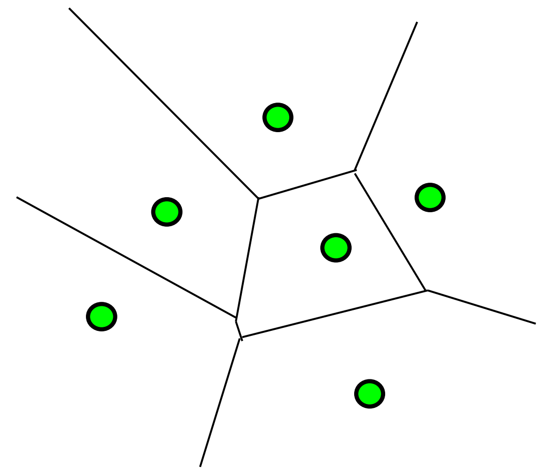
k-means (*Generalized Lloyd's Algorithm – Vector quantization*)

Given a set $D := \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ *of observations (i.e. vectors in* $\mathbb{R}^d$ *)*
and a set $\theta := \{\mathbf{w}_1, \ldots, \mathbf{w}_k\}$ *of* k *prototypes (i.e. vectors in* $\mathbb{R}^d$*)*

Voronoi cell:

$$V(\mathbf{w}_j) := \{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x} - \mathbf{w}_j\| \leq \|\mathbf{x} - \mathbf{w}_l\| , \forall\, l \neq j\}$$

Voronoi tesselation: the complex of all Voronoi cells of θ



Algorithm (*rewritten*)**:**

1) Position the *k* prototypes at random
2) Assign each observation to its Voronoi cell

$$w(\mathbf{x}_i) := \mathbf{w}_j \mid \mathbf{x}_i \in V(\mathbf{w}_j)$$

3) Position each prototype at the *centroid* of the observations in its Voronoi cell

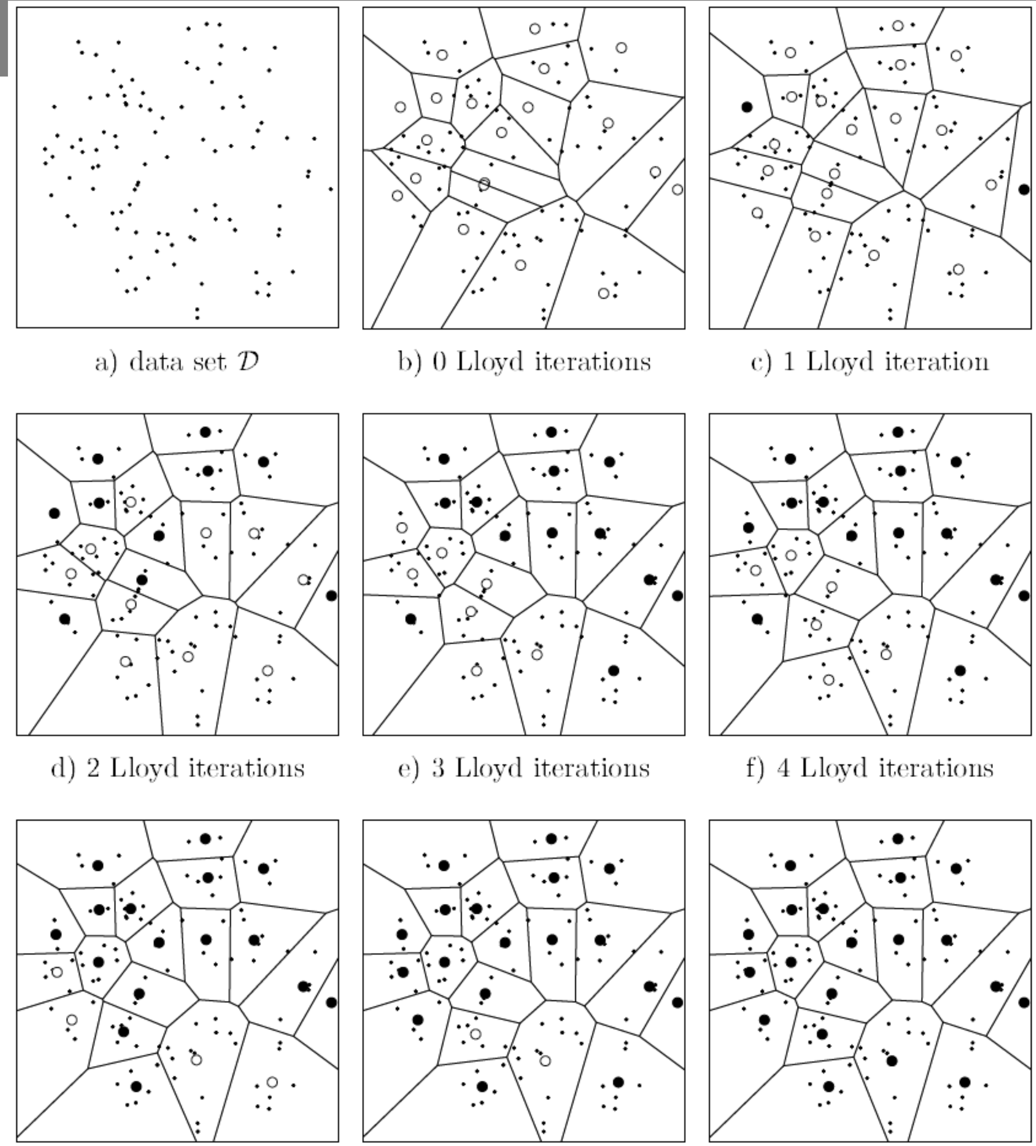
$$\mathbf{w}_j = \frac{1}{|\{\mathbf{x}_i \in V(\mathbf{w}_j)\}|} \sum_{\{\mathbf{x}_i \in V(\mathbf{w}_j)\}} \mathbf{x}_i$$

4) Unless no prototype was moved in step 3), go back to step 2)

k-means

An example run of the algorithm

The landmarks (empty circles) become black when they cease to move



a) data set $\mathcal{D}$ b) 0 Lloyd iterations c) 1 Lloyd iteration d) 2 Lloyd iterations e) 3 Lloyd iterations f) 4 Lloyd iterations g) 5 Lloyd iterations h) 6 Lloyd iterations i) 7 Lloyd iterations

Expected value of a random variable

(also *expectation*)

Basic definition

$$\mathbb{E}_X[X] := \sum_{x \in \mathcal{X}} x \; P(X = x)$$

More concise notation

$$\mathbb{E}[X] := \sum_{x \in \mathcal{X}} x \; P(x)$$

Continuous case

$$\mathbb{E}[X] := \int_{x \in \mathcal{X}} x \; p(x) dx$$

A linear operator

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$$

$$\mathbb{E}[cX] = c\mathbb{E}[X]$$

Conditional expectation

$$\mathbb{E}_X[X|Y = y] = \mathbb{E}[X|Y = y] := \sum_{x \in \mathcal{X}} x \; P(X = x|Y = y)$$

Iterated expectation (*see Wikipedia*)

$$\mathbb{E}_X[X] = \mathbb{E}_Y[\mathbb{E}_X[X|Y]]$$

Joint Expected Value

The ***expected value*** of a function f of a set of random variables $\{X_i\}$ is

$$\mathbb{E}[f(\{X_i\})] := \sum_{\{X_i\}} f(\{X_i\})\ P(\{X_i\})$$

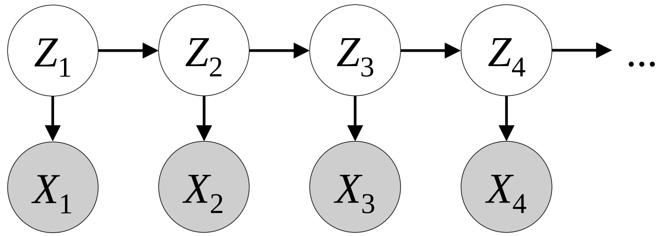
the sum is over all possible combinations of values of the random variables

(Unless specified otherwise, the $\mathbb{E}$ operator acts over *all* the random variables enclosed)

The extension to the continuous case is obvious

Incomplete observations

Example: 'Hidden Markov' model



Terminology:
hidden = latent = always unobserved
missing = unobserved (in a data set)

Typically, Z_i nodes are *hidden*, i.e. *non-observables*

$$P(\{X_i\},\{Z_j\}) = P(Z_1)\,P(X_1 \mid Z_1)\prod_{i=2}^{n} P(Z_i \mid Z_{i-1})\,P(X_i \mid Z_i)$$

Joint distribution

- Problem

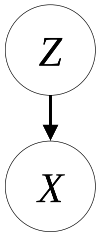
MLE of parameters θ starting from *partial* observations of the $\{X_i\}$ variables *only*

In other terms, this is the *MLE* of the *likelihood function*

$$L(\theta \mid D) = P(D \mid \theta) = \sum_{\{Z_j\}} P(D,\{Z_j\} \mid \theta)$$

Note that the model (= the probability function) and the (partial) observations are known, the parameters and the values of some variables are hidden

Expectation Maximization: a preliminary example



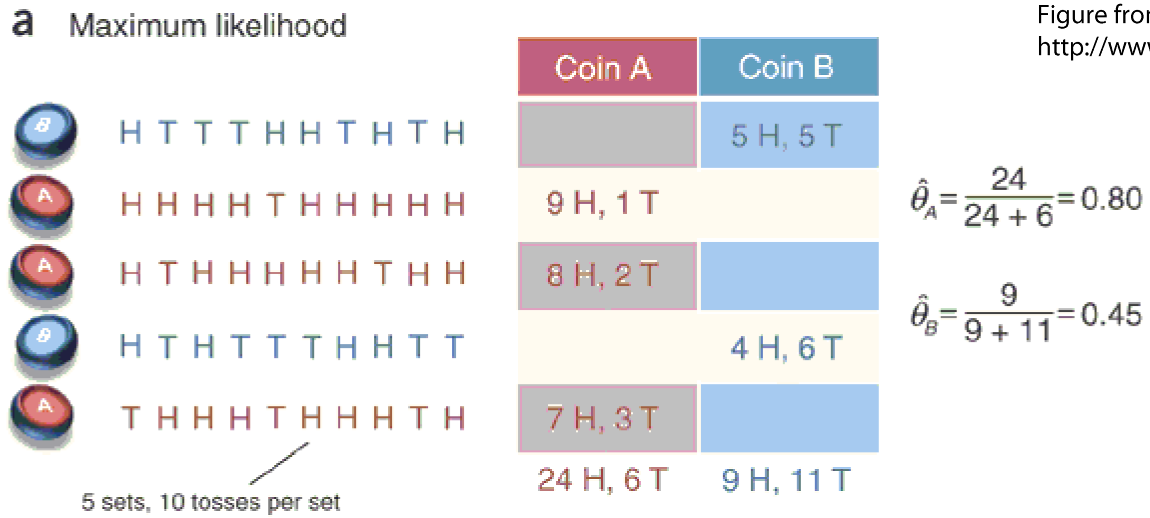


Figure from
http://www.nature.com/nbt/journal/v26/n8/full/nbt1406.html

- An experiment with two coins

  At each step, one coin is selected at random (*with equal probability*) and then tossed ten times

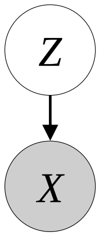
  Random variables: X number of heads, Z selected coin (i.e A or B)

  Parameters to be learnt: $\theta = \{ \theta_A, \theta_B \}$ probabilities of landing on head of A and B, resp.

  When the results are fully observable, by MLE:

$$\theta_A^{\ *} = \frac{N_{A=1}}{N_A} \qquad \theta_B^{\ *} = \frac{N_{B=1}}{N_B}$$

Expectation Maximization: a preliminary example



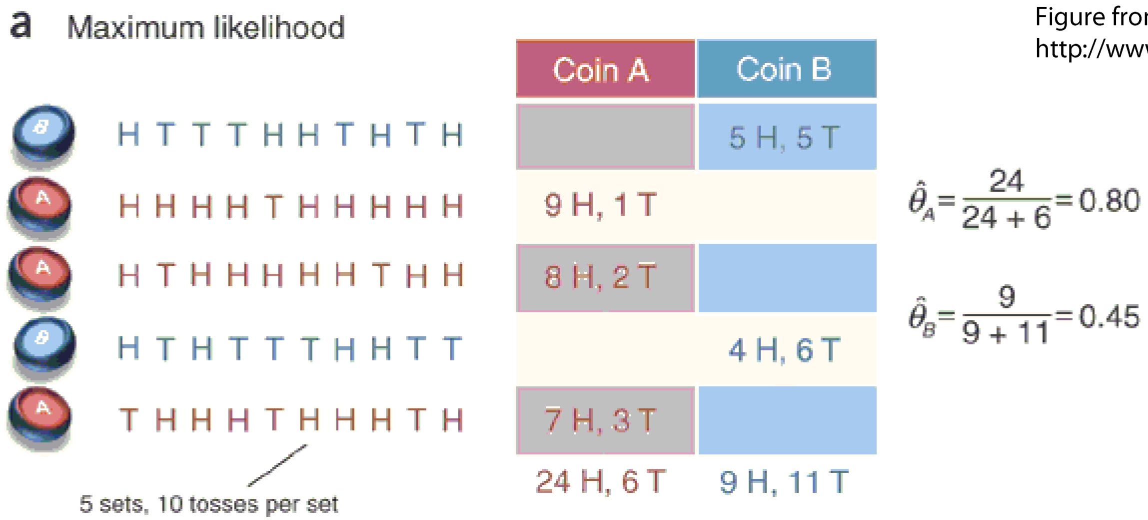


Figure from
http://www.nature.com/nbt/journal/v26/n8/full/nbt1406.html

- An experiment with two coins

  At each step, one coin is selected at random (*with equal probability*) and then tossed ten times

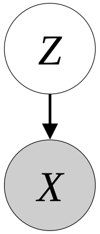
  Random variables: X number of heads, Z selected coin (i.e A or B)

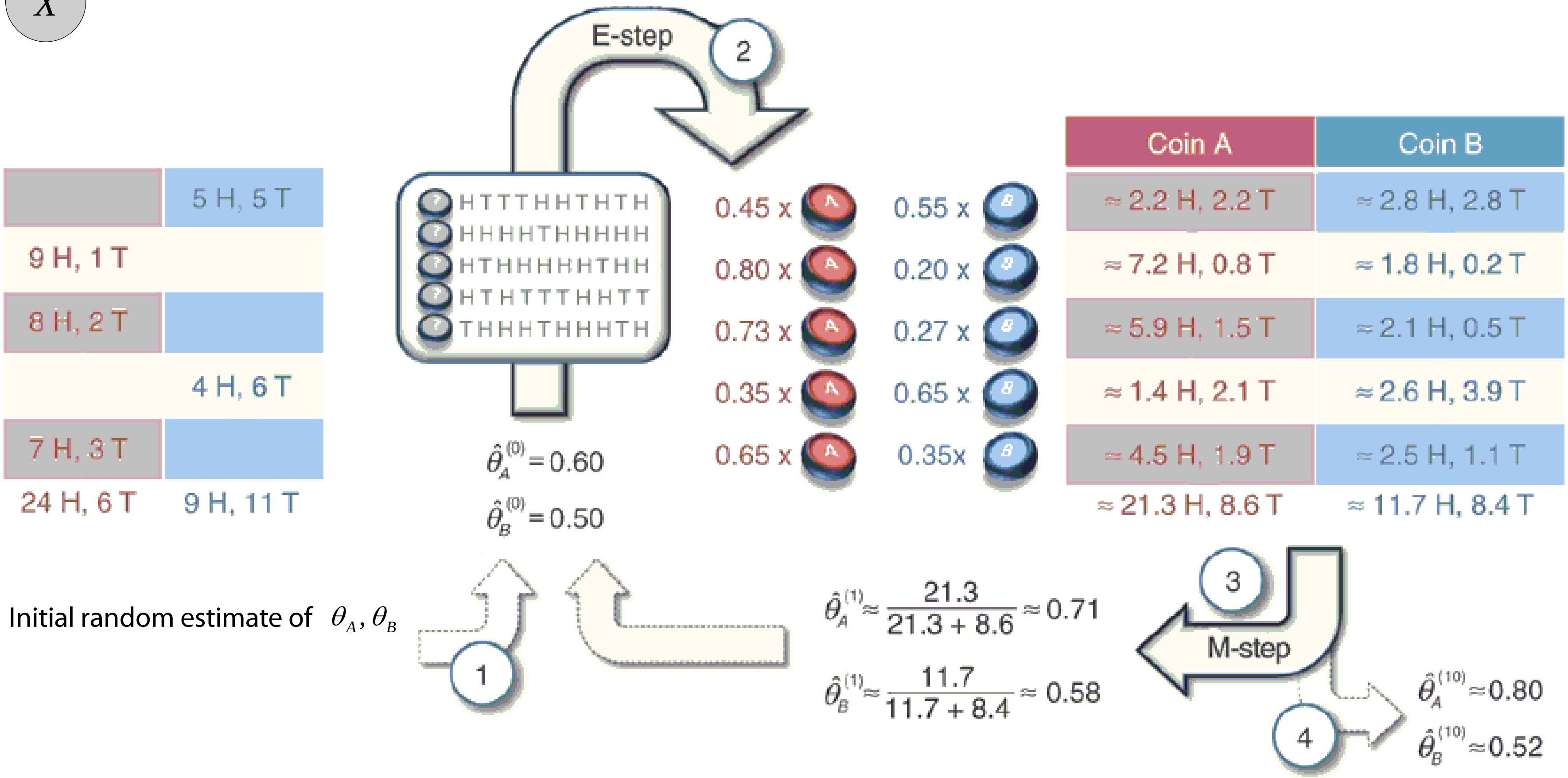
  Parameters to be learnt: $\theta = \{ \theta_A, \theta_B \}$ probabilities of landing on head of A and B, resp.

- What if Z is *hidden* (= *latent*, = *unobserved*)?

  The results of each sequence of coin tosses are known, but not the coin selected

Expectation Maximization: a preliminary example





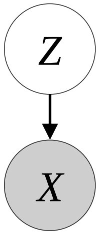
Initial random estimate of θ_A, θ_B

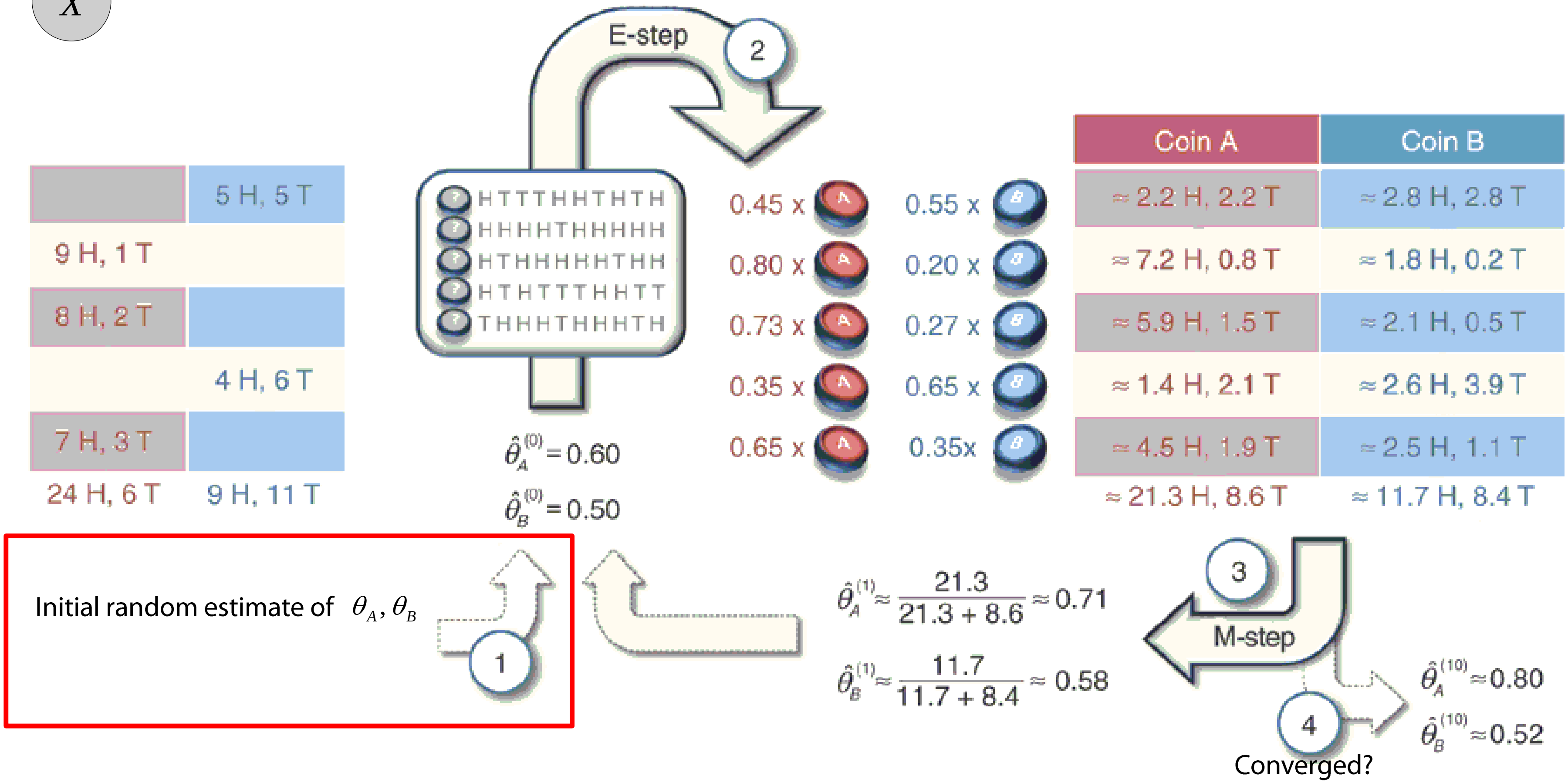
Converged?

MLE given expected observations

Figure from
http://www.nature.com/nbt/journal/v26/n8/full/nbt1406.html

Expectation Maximization: a preliminary example

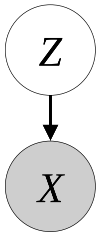




MLE given expected observations

Figure from
http://www.nature.com/nbt/journal/v26/n8/full/nbt1406.html

Expectation Maximization: a preliminary example



Compute the probability distribution of hidden observations

$$P(Z=z \mid X^{(i)}=x,\theta) = \frac{P(X^{(i)}=x \mid Z=z,\theta)P(Z=z \mid \theta)}{\sum_z P(X^{(i)}=x \mid Z=z,\theta)P(Z=z \mid \theta)}$$

where

$$P(X^{(i)}=x \mid Z=z,\theta) = \binom{10}{x}\,\theta_z^x\,(1-\theta_z)^{10-x}$$

$$P(Z=z \mid \theta) = \theta_z$$

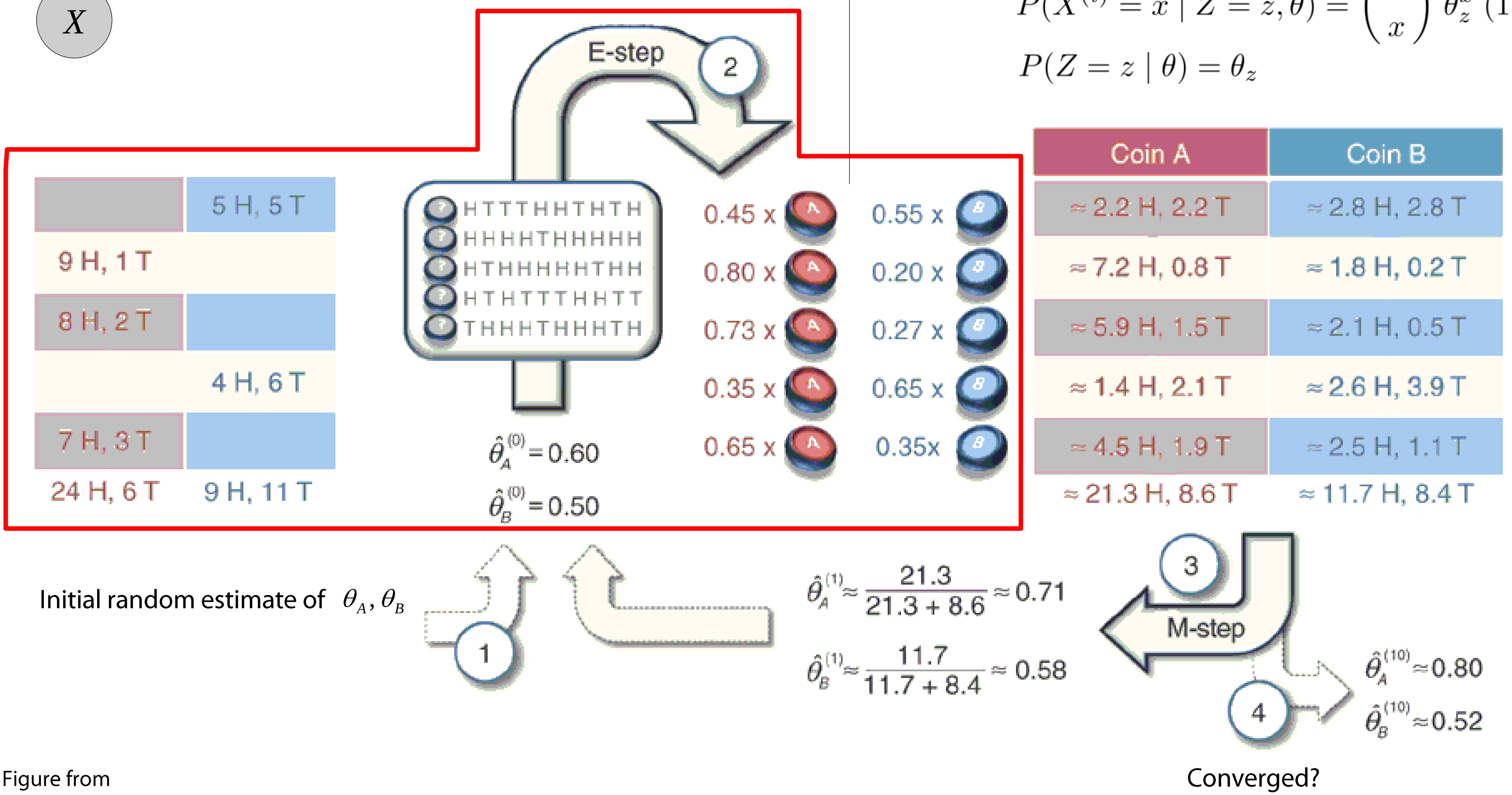
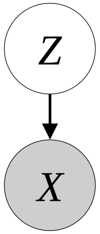


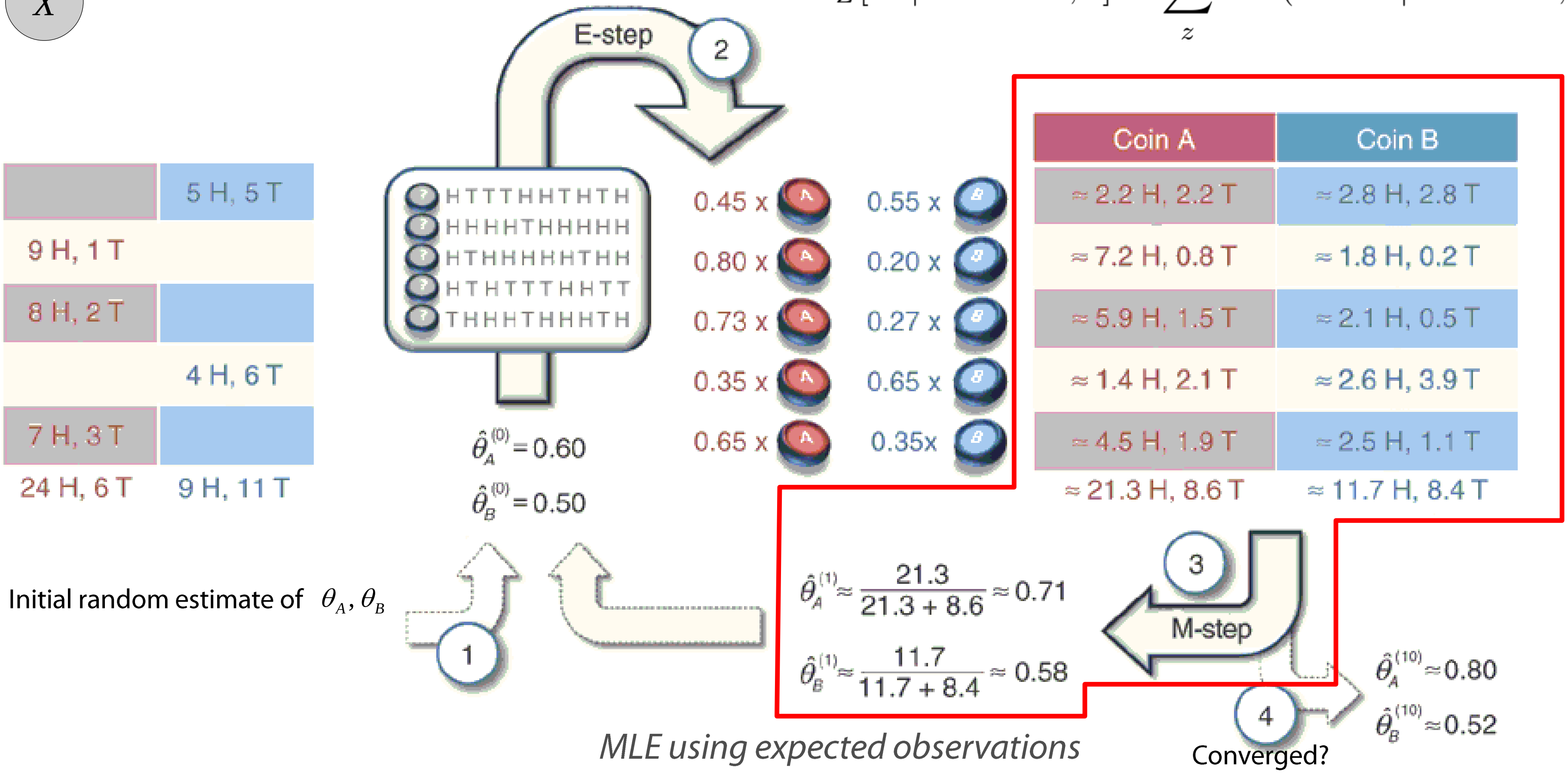
Figure from http://www.nature.com/nbt/journal/v26/n8/full/nbt1406.html

Expectation Maximization: a preliminary example



Use expected observations instead of actual observations to update ML estimations

$$\mathbb{E}_Z[\, x \mid X^{(i)} = x, \theta] = \sum_z x\, P(Z = z \mid X^{(i)} = x, \theta) = x$$



Initial random estimate of θ_A, θ_B

MLE using expected observations

Converged?

Figure from
http://www.nature.com/nbt/journal/v26/n8/full/nbt1406.html

An aside: Jensen's inequality

A relationship between probability and geometry

When f is *convex function*

$$f(E[\{X_i\}]) \leq E[f(\{X_i\})]$$

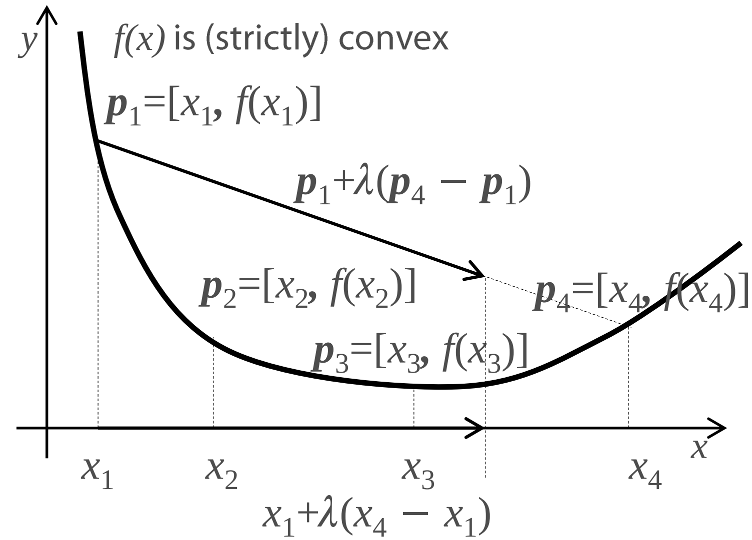
*f is **convex** when for any two points $\mathbf{p}_i$ and $\mathbf{p}_j$ the segment ($\mathbf{p}_i - \mathbf{p}_j$) is not below f*

That is, when

$$\lambda f(x_i) + (1-\lambda) f(x_j) \geq f(\lambda x_i + (1-\lambda) x_j) \quad \forall \lambda \in [0,1]$$

*Furthermore, f is **strictly convex** when*

$$\lambda f(x_i) + (1-\lambda) f(x_j) > f(\lambda x_i + (1-\lambda) x_j) \quad \forall \lambda \in (0,1)$$



Corollary:

when f is *strictly* convex, if and only if all the variables in $\{X_i\}$ are <u>constant</u> it is true that

$$f(E[\{X_i\}]) = E[f(\{X_i\})]$$

Dual results also hold for <u>*concave*</u> functions

An aside: Jensen's inequality

A relationship between probability and geometry

When f is *convex function*

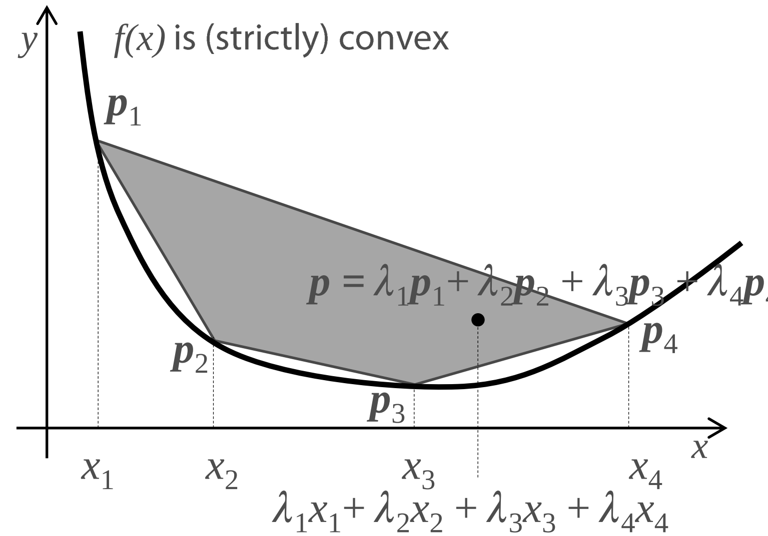
$$f(E[\{X_i\}]) \le E[f(\{X_i\})]$$

To see this, consider

$$\boldsymbol{p} = \lambda_1\boldsymbol{p}_1 + \lambda_2\boldsymbol{p}_2 + \lambda_3\boldsymbol{p}_3 + \lambda_4\boldsymbol{p}_4$$

i.e. a ***linear combination*** of $\boldsymbol{p}_i$ points

This is an ***affine*** combination if $\sum \lambda_i = 1$
and it is a ***convex*** combination if also $\lambda_i \ge 0, \forall i$



When the λ_i define a probability, then $\boldsymbol{p}$ is a *convex combination* of $\boldsymbol{p}_i$ points

Any convex combination of $\boldsymbol{p}_i$ points lies inside their ***convex hull*** (*see figure*) and therefore above f :

$$\sum_i \lambda_i f(x_i) \ \ge \ f\left(\sum_i \lambda_i x_i\right)$$

Corollary: the only way to make the convex hull be <u>on</u> f is to shrink it to a single point (i.e. the Jensen's corollary)

Incomplete observations

Likelihood function with hidden random variables

$$L(\theta \mid D) = P(D \mid \theta) = \prod_m P(D_m \mid \theta)$$

$$\ell(\theta \mid D) = \sum_m \log P(D_m \mid \theta) = \sum_m \log \sum_{\{Z_i\}} P(D_m, \{Z_i\} \mid \theta_k)$$

Arbitrary probability distributions

$$= \sum_m \log \sum_{\{Z_i\}} Q_m(\{Z_i\}) \frac{P(D_m, \{Z_i\} \mid \theta)}{Q_m(\{Z_i\})}$$

Jensen's inequality: log is concave

$$= \sum_m \log E_{Q_m(\{Z_i\})}\left[\frac{P(D_m, \{Z_i\} \mid \theta)}{Q_m(\{Z_i\})}\right] \geq \sum_m E_{Q_m(\{Z_i\})}\left[\log \frac{P(D_m, \{Z_i\} \mid \theta)}{Q_m(\{Z_i\})}\right]$$

$$= \sum_m \sum_{\{Z_i\}} Q_m(\{Z_i\}) \log \frac{P(D_m, \{Z_i\} \mid \theta)}{Q_m(\{Z_i\})}$$

Expectation- Maximization (EM) Algorithm

Alternate optimization (coordinate ascent)

Log-likelihood function:

$$\ell(\theta \mid D) \geq \sum_m \sum_{\{Z_i\}} Q_m(\{Z_i\}) \log \frac{P(D_m, \{Z_i\} \mid \theta)}{Q_m(\{Z_i\})}$$

This inequality becomes equality *when this term is <u>constant</u> (see Jensen's corollary)*

Keep θ constant, define $Q_m(\{Z_i\})$ so that the right side of the inequality is maximized

$$Q_m(\{Z_i\}) := \frac{P(D_m, \{Z_i\} \mid \theta)}{\sum_{\{Z_i\}} P(D_m, \{Z_i\} \mid \theta)} = \frac{P(D_m, \{Z_i\} \mid \theta)}{P(D_m \mid \theta)} = P(\{Z_i\} \mid D_m, \theta) =: p^{(m)}_{\{Z_i\}}$$

These <u>numbers</u> can be computed from the graphical model (i.e. as an <u>inference</u> step)

Then maximize the log-likelihood while keeping $Q_m(\{Z_i\})$ constant

$$\theta^* = \arg\max_\theta \sum_m \sum_{\{Z_i\}} p^{(m)}_{\{Z_i\}} \log \frac{P(D_m, \{Z_i\} \mid \theta)}{p^{(m)}_{\{Z_i\}}}$$

$$= \arg\max_\theta \sum_m \left(\sum_{\{Z_i\}} p^{(m)}_{\{Z_i\}} \log P(D_m, \{Z_i\} \mid \theta) - \sum_{\{Z_i\}} p^{(m)}_{\{Z_i\}} \log p^{(m)}_{\{Z_i\}} \right)$$

This is also called the <u>entropy</u> of $Q_m(\{Z_i\})$ (i.e. a constant measure of the distribution)

$$= \arg\max_\theta \sum_m \sum_{\{Z_i\}} p^{(m)}_{\{Z_i\}} \log P(D_m, \{Z_i\} \mid \theta)$$

Expectation- Maximization (EM) Algorithm

Alternate optimization (coordinate ascent)

Log-likelihood function and its estimator:

$$\ell(\theta \mid D) \;\geq\; \sum_{m} \sum_{\{Z_i\}} Q_m(\{Z_i\}) \log \frac{P(D_m, \{Z_i\} \mid \theta)}{Q_m(\{Z_i\})}$$

Algorithm:

1) Assign the θ at random
2) (*E-step*) Compute the probabilities

$$p^{(m)}_{\{Z_i\}} \;=\; Q_m(\{Z_i\}) \;=\; P(\{Z_i\} \mid D_m, \theta)$$

3) (*M-step*) Compute a new estimate of θ

$$\theta^* = \arg\max_{\theta} \sum_{m} \sum_{\{Z_i\}} p^{(m)}_{\{Z_i\}} \log P(D_m, \{Z_i\} \mid \theta)$$

4) Go back to step 2) until some convergence criterion is met

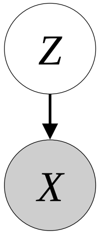
The algorithm converges to a local maximum of the log-likelihood

The effectiveness of algorithm depends on the form of the distribution (see step3):

$$P(D_m, \{Z_i\} \mid \theta)$$

In particular, when this distribution is exponential… (e.g. Gaussian – see next slide)

EM Algorithm: mixture of Gaussians



Model:

The hidden variable Z *has* k *possible values, the observable variable* X *is a point in* $\mathbb{R}^d$

$P(Z = k) := \phi_k$ — *Multivariate normal distribution*

$$P(X = x \mid Z = k) = N(x; \mu_k, \Sigma_k) := (2\pi)^{-d/2} (\det \Sigma_k)^{-1/2} \exp\left(-\frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) \right)$$

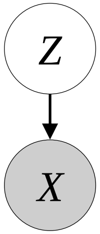
i.e. the condition probabilities are <u>*normal*</u> *distributions*

The observations are a set $D = \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$ *of points in* $\mathbf{R}^d$

Algorithm:

1) For each value *k*, assign ϕ_k, μ_k and Σ_k at random
2) (*E-step*) For all the x_i in D compute the probabilities
   $$p_k^{(m)} = P(Z = k \mid x^{(m)}, \phi_k, \mu_k, \Sigma_k) = \phi_k \cdot N(x^{(m)}; \mu_k, \Sigma_k)$$
3) (*M-step*) Compute the new estimates for the parameters
   $$\phi_k = \frac{1}{n} \sum_m p_k^{(m)}$$
   $$\mu_k = \frac{\sum_m p_k^{(m)} x^{(m)}}{\sum_m p_k^{(m)}} \qquad \Sigma_k = \frac{\sum_m p_k^{(m)} (x - \mu_k)(x - \mu_k)^T}{\sum_m p_k^{(m)}}$$
4) Go back to step 2) until some convergence criterion is met

EM Algorithm: mixture of Gaussians



Model:

The hidden variable Z has k possible values, the variable X is a point in $\mathbb{R}^d$

$$P(Z = k) := \phi_k$$

$$P(X = x \mid Z = k) = N(x; \mu_k, \Sigma_k) := (2\pi)^{-d/2} (\det \Sigma_k)^{-1/2} \exp\left(-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) \right)$$

i.e. the condition probabilities are *<u>normal</u>* *distributions*

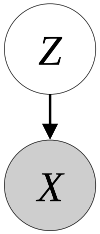
The observations are a set $D = \{x^{(1)}, x^{(2)}, \dots, x^{(n)}\}$ of points in $\mathbb{R}^d$

Proof (of the M-step):

$$\sum_m \sum_k p_k^{(m)} \log P(X^{(m)}, Z = k \mid \phi_k, \mu_k, \Sigma_k) = \sum_m \sum_k p_k^{(m)} \log P(X^{(m)} \mid Z = k, \mu_k, \Sigma_k) P(Z = k \mid \phi_k)$$

$$= \sum_m \sum_k p_k^{(m)} \left(\log\left(2\pi^{-d/2} (\det \Sigma_k)^{-1/2}\right) + \left(-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) \right) + \log \phi_k \right)$$

EM Algorithm: mixture of Gaussians



Model:

The hidden variable Z has k possible values, the variable X is a point in $\mathbb{R}^d$

$$P(Z=k) := \phi_k$$

$$P(X=x \mid Z=k) = N(x;\mu_k,\Sigma_k) := (2\pi)^{-d/2}(\det\Sigma_k)^{-1/2}\exp\left(-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)\right)$$

i.e. the condition probabilities are <u>normal</u> distributions

The observations are a set $D = \{x^{(1)}, x^{(2)}, \dots, x^{(n)}\}$ *of points in* $\mathbb{R}^d$

Proof (of the M-step):

$$\frac{\partial}{\partial\mu_j}\sum_m\sum_k p_k^{(m)}\left(\log\left((2\pi)^{-d/2}(\det\Sigma_k)^{-1/2}\right)+\left(-\frac{1}{2}(x^{(m)}-\mu_k)^T\Sigma_k^{-1}(x^{(m)}-\mu_k)\right)+\log\phi_k\right)$$

$$=\frac{\partial}{\partial\mu_j}\sum_m\sum_k p_k^{(m)}\left(-\frac{1}{2}(x^{(m)}-\mu_k)^T\Sigma_k^{-1}(x^{(m)}-\mu_k)\right)$$

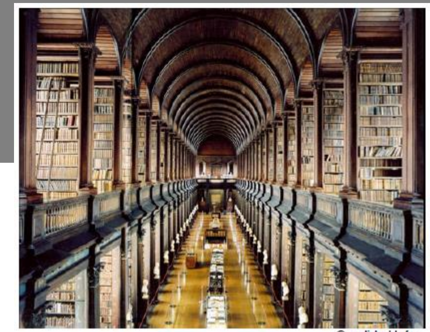
$$=\frac{\partial}{\partial\mu_j}\sum_m\sum_k p_k^{(m)}\left(-\frac{1}{2}(x^{(m)T}\Sigma_k^{-1}x^{(m)}+\mu_k^T\Sigma_k^{-1}\mu_k-2+x^{(m)T}\Sigma_k^{-1}\mu_k)\right) = \sum_m p_j^{(m)}\left(x^T\Sigma_j^{-1}-\mu_j^T\Sigma_j^{-1}\right)$$

By imposing: $\sum_m p_j^{(m)}\left(x^T\Sigma_j^{-1}-\mu_j^T\Sigma_j^{-1}\right)=0$

$$\mu_j = \frac{\sum_m p_j^{(m)}x^{(m)}}{\sum_m p_j^{(m)}}$$

See the link in the web page for the derivations of other parameters …

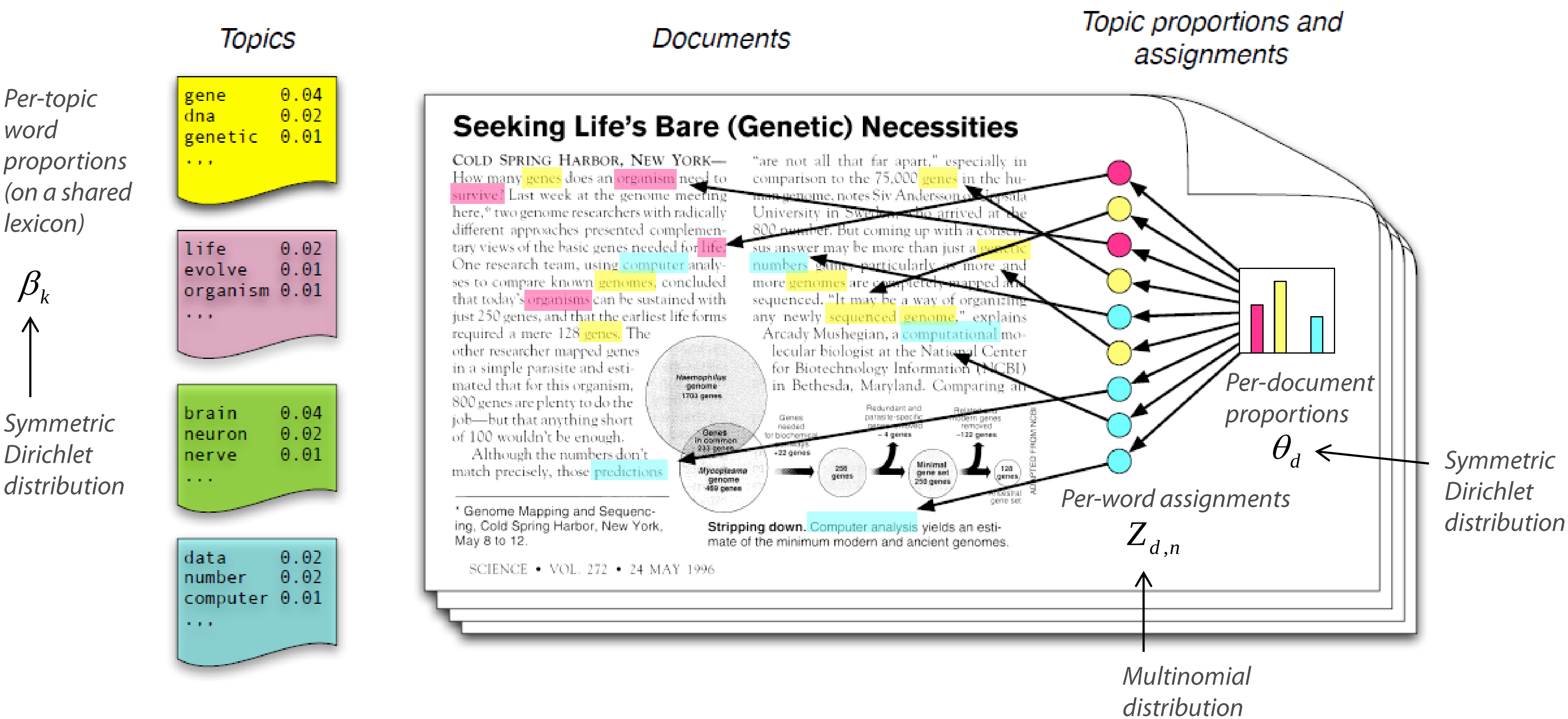
Topic modeling



Candida Hofer

Topic modeling

Classifying a (large) corpus of digital documents relying on word counting only



Multinomial distribution

- Bernoulli

  Head or Tail?

  $$P(X=1)=\theta, \quad P(X=0)=1-\theta$$

- Binomial

  n heads out of N coin tosses

  $$P(X=n)=\binom{N}{n}\theta^{n}(1-\theta)^{(N-n)}$$

- Categorical

  The result of throwing a dice with k *faces*

  $$P(X=1)=\theta_1, \quad P(X=k)=\theta_k, \qquad \sum_{i=1}^{k}\theta_i=1$$

- Multinomial

  Obtaining an outcome combination $x_1, \dots, x_k$ in N throws of a k*–faced dice*, *with* $\sum_{i=1}^{k} x_i = N$

  $$P(X_1=x_1, \dots, X_k=x_k)=\frac{N!}{x_1! \dots x_k!}\prod_{i=1}^{k}\theta_i^{x_i}$$

Dirichlet distribution

- Beta distribution

  What do you think about a coin after obtaining $(\alpha_1 - 1)$ *heads and* $(\alpha_2 - 1)$ *tails?*

$$\mathrm{Beta}(x_1, x_2; \alpha_1, \alpha_2) := \frac{x_1^{\alpha_1 - 1} \cdot x_2^{\alpha_2 - 1}}{\mathrm{B}(\alpha_1, \alpha_2)}, \qquad x_1 + x_2 = 1$$

This is just a re-writing of the 'standard' formula:

$$\mathrm{Beta}(x; \alpha, \beta) := \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha, \beta)}$$

- Dirichlet distribution

  What *do you think about a* k*-faced dice after obtaining* $(\alpha_1 - 1), (\alpha_2 - 1) \ldots (\alpha_k - 1)$ *outcomes?*

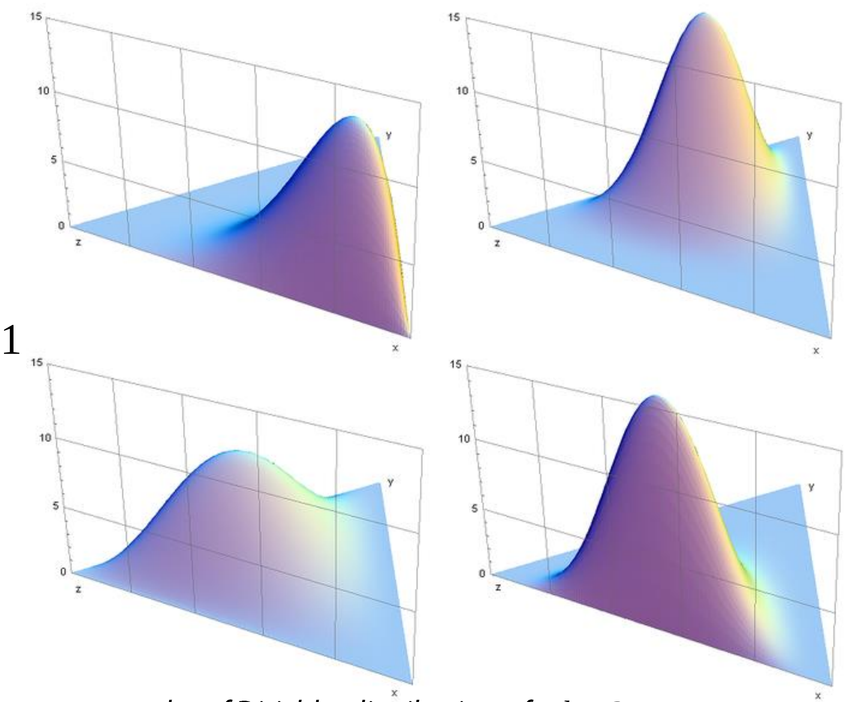
$$\mathrm{D}(x_1, \ldots, x_k; \alpha_1, \ldots, \alpha_k) := \frac{\prod_{i=1}^{k} x_i^{\alpha_i - 1}}{\mathrm{B}(\alpha_1, \ldots, \alpha_k)}, \qquad \sum_{i=1}^{k} x_i = 1$$

where

$$\mathrm{B}(\alpha_1, \ldots, \alpha_k) := \frac{\prod_{i=1}^{k} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{k} \alpha_i\right)}$$

is the *multivariate Beta function.*

The Dirichlet distribution is the ***conjugate prior*** of the Multinomial distribution



examples of Dirichlet distributions, for k = 3
(from Wikipedia)

Dirichlet distribution

- Symmetric Beta distribution

  i.e. when $\alpha = \beta$

$$\mathrm{Beta}(x_1, x_2; \alpha) := \frac{x_1^{\alpha-1} \cdot x_2^{\alpha-1}}{\mathrm{B}(\alpha, \alpha)}, \qquad x_1 + x_2 = 1$$

- Symmetric Dirichlet distribution

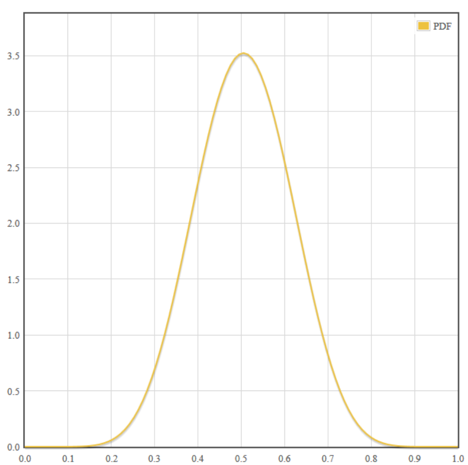
  i.e. when $\alpha_1 = \alpha_2 = \ldots = \alpha_k$

$$\mathrm{D}(x_1, \ldots, x_k; \alpha) := \frac{\prod_{i=1}^{k} x_i^{\alpha-1}}{\mathrm{B}(\alpha, \ldots, \alpha)}, \qquad \sum_{i=1}^{k} x_i = 1$$

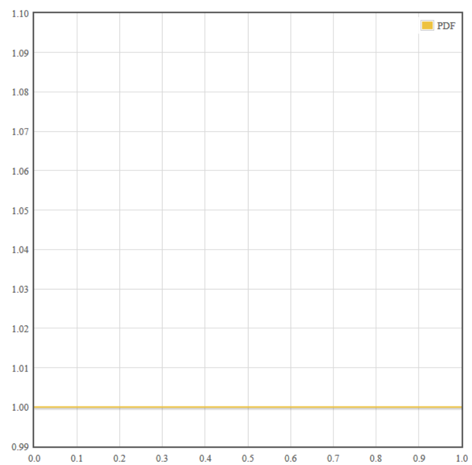
Note: in both distributions, the parameters can be < 1

(this is true of the non-symmetric versions as well)

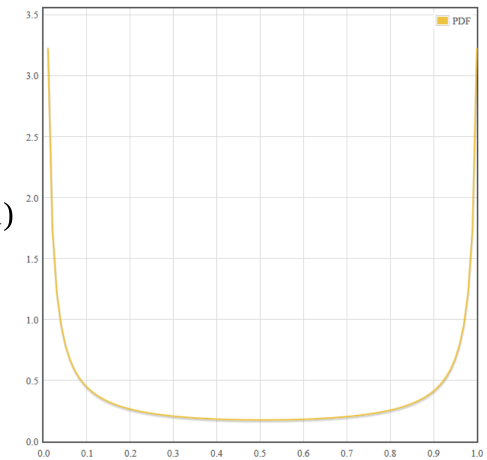
$\mathrm{Beta}(x_1,x_2;10)$



$\mathrm{Beta}(x_1,x_2;1)$

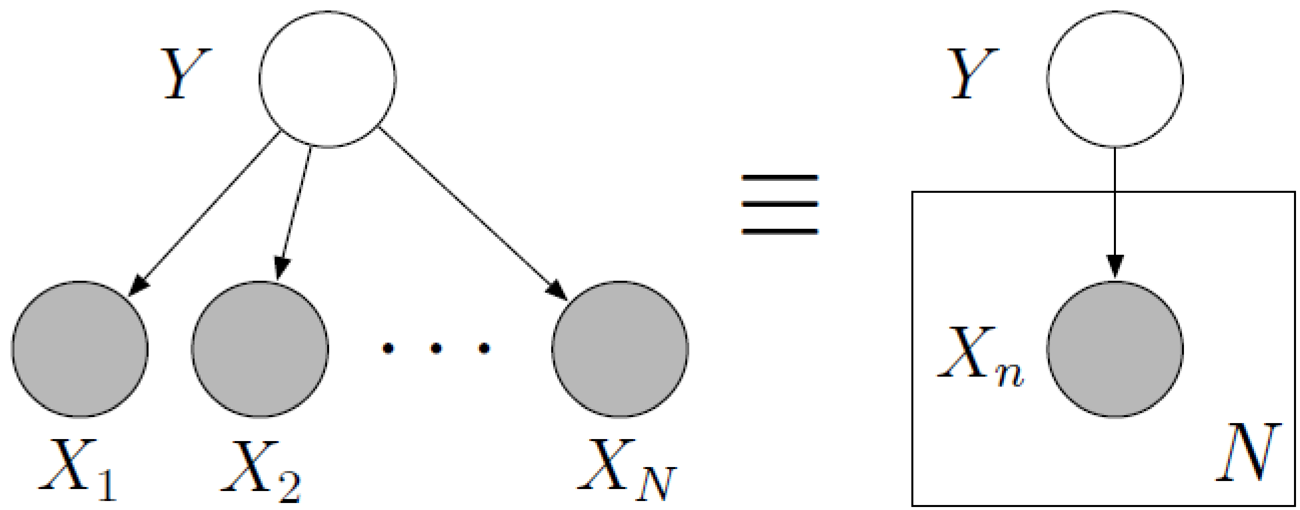


$\mathrm{Beta}(x_1,x_2;0.1)$



An aside: plate notation

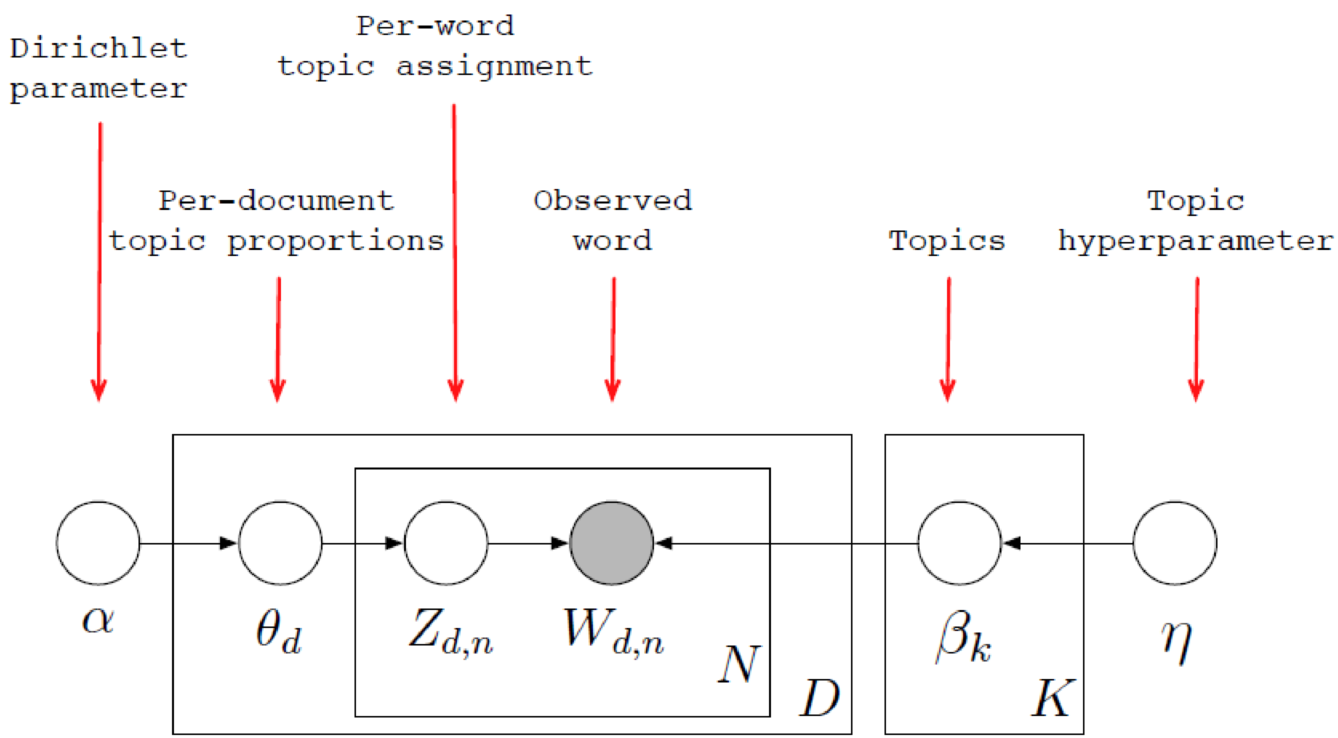
A shorthand notation for graphical models



An example: Probabilistic Topic Models (Blei & Lafferty, 2009)

Classifying a corpus of documents with k (unknown) topics
when the only observable variables is the multiple occurrence of words

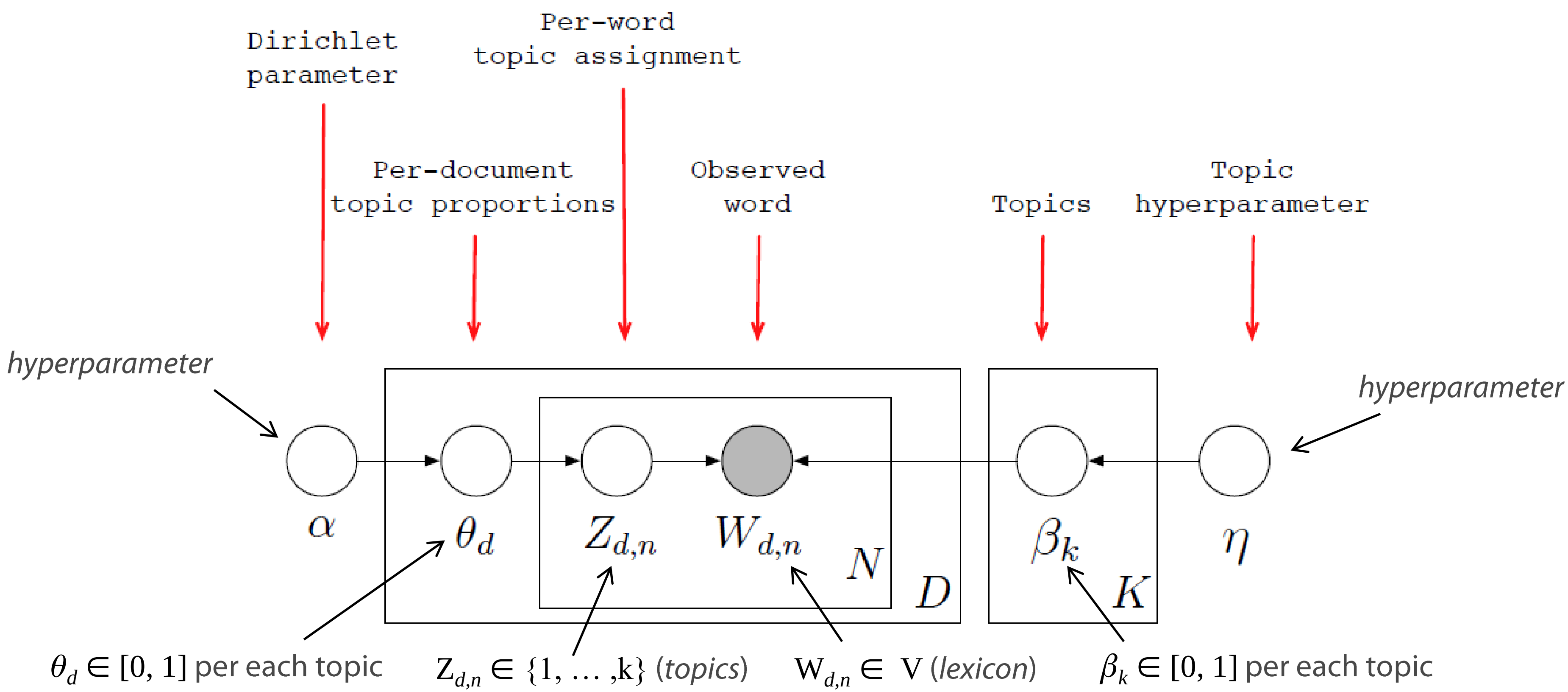
A <u>mixture</u> model:
each document belongs to <u>multiple topics</u>, with different probabilities



An example: Probabilistic Topic Models (Blei & Lafferty, 2009)

Classifying a corpus of documents with K (unknown) topics
when the only observable variables is the multiple occurrence of words

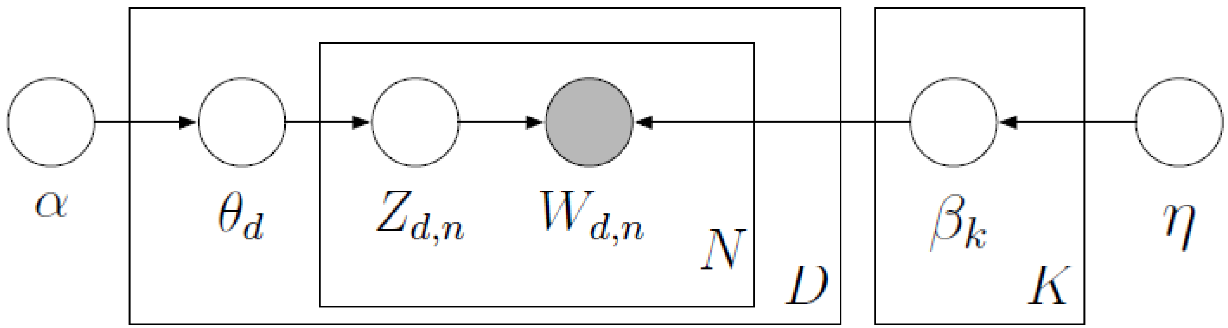
A mixture model:
each document belongs to multiple topics, with different probabilities



An example: Probabilistic Topic Models (Blei & Lafferty, 2009)

Classifying a corpus of documents as *mixtures* of K (unknown) topics when the only observable variables is the multiple occurrence of words

A three-level, mixture model:
each document belongs to multiple topics, with different probabilities



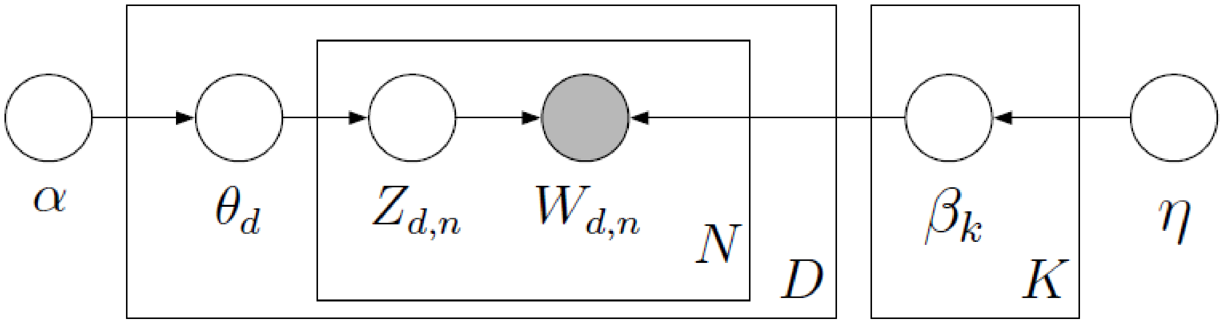
$$\prod_{i=1}^{K} p(\beta_i \mid \eta) \prod_{d=1}^{D} p(\theta_d \mid \alpha) \left(\prod_{n=1}^{N} p(z_{d,n} \mid \theta_d) p(w_{d,n} \mid \beta_{1:K}, z_{d,n}) \right)$$

Symmetric Dirichlet distributions — *Multinomial distributions*

Latent Dirichlet Allocation (LDA)

Classifying a corpus of documents as *mixtures* of K (unknown) topics when the only observable variables is the multiple occurrence of words

A three-level, mixture model:
each document belongs to multiple topics, with different probabilities



A *generative* procedure:

1. Draw each topic $\beta_i \sim \mathrm{Dir}(\eta)$, for $i \in \{1, \ldots, K\}$.
2. For each document:
   1. Draw topic proportions $\theta_d \sim \mathrm{Dir}(\alpha)$.
   2. For each word:
      1. Draw $Z_{d,n} \sim \mathrm{Mult}(\theta_d)$.
      2. Draw $W_{d,n} \sim \mathrm{Mult}(\beta_{z_{d,n}})$.

LDA: which results?

Identifying topics:
relative frequencies
of words that define a class

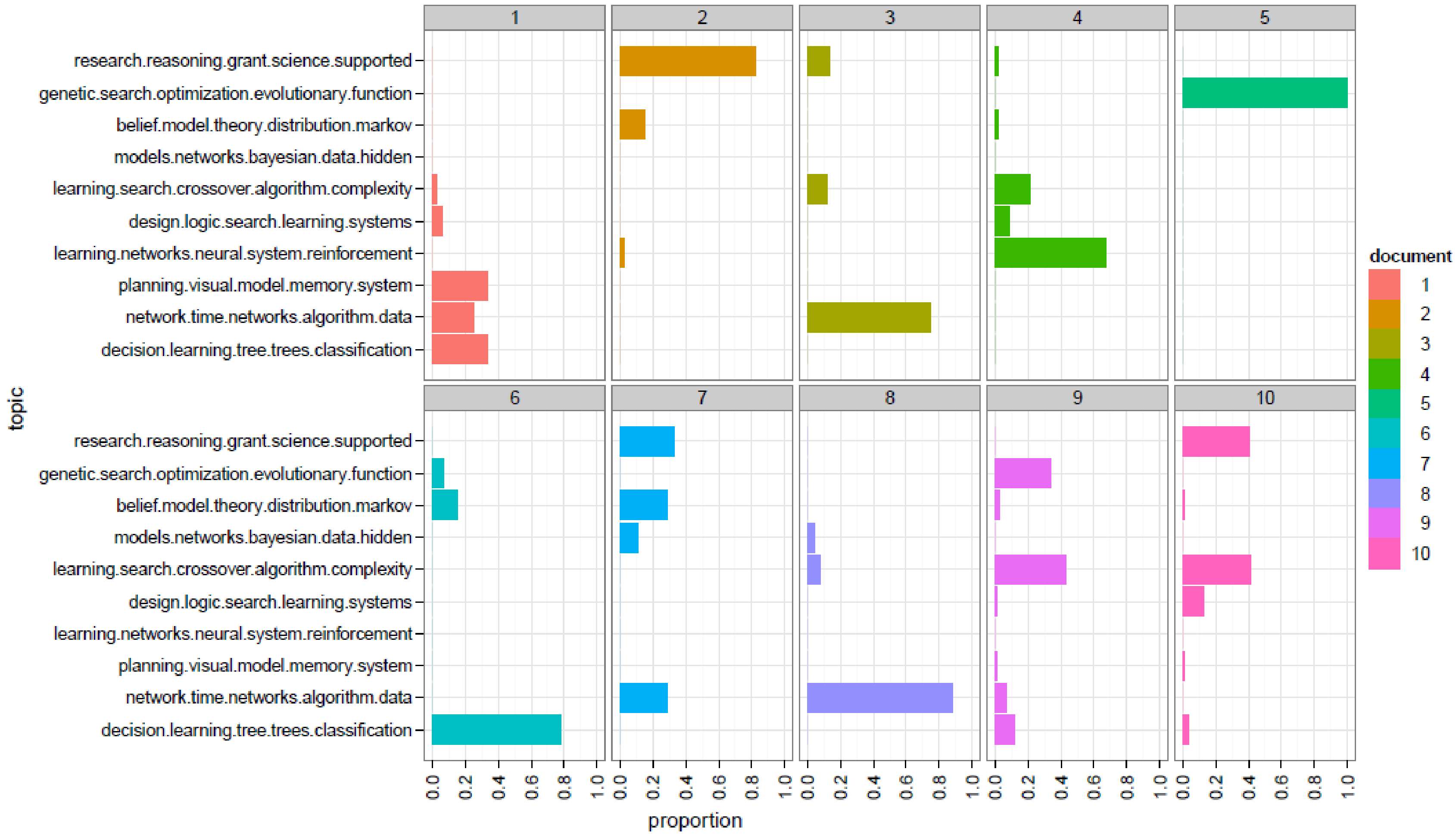
Each box represents a topic

The size of words in a box represents its relative proportion

LDA: which results?

Classifying documents: *relative topic assignment proportions*

Each topic is represented by a list of most relevant words



LDA in practice

There exist multiple methods

Mean-Field Variational Inference (Blei et al. 2003)

(not discussed here – see links to the literature)
(It is a sort of generalization of the EM algorithm)

Many software implementations around: e.g. Apache Mahout

Real-world examples

The OCR'ed collection of Science from 1990-2000 [2009]

- *17K documents*
- *11M words*
- *20K unique terms (stop words and rare words removed)*

Model: 100 Topics

The New York Times online recommender system [2015]

See http://open.blogs.nytimes.com/2015/08/11/building-the-next-new-york-times-recommendation-engine/