Statistical Machine Learning for NLP

Xiaojin Zhu

jerryzhu@cs.wisc.edu
Department of Computer Sciences
University of Wisconsin–Madison, USA

CCF/ADL46 2013

Outline

Outline

Outline

Probability

- The probability of a discrete random variable A taking the value a is $P(A = a) \in [0, 1]$.
- Sometimes written as $P(a)$ when no danger of confusion.
- Normalization $\sum_{\text{all } a} P(A = a) = 1$.
- Joint probability $P(A = a, B = b) = P(a, b)$, the two events both happen at the same time.
- Marginalization $P(A = a) = \sum_{\text{all } b} P(A = a, B = b)$, "summing out B".
- Conditional probability $P(a|b) = \frac{P(a,b)}{P(b)}$, a happens given b happened.
- The product rule $P(a, b) = P(a)P(b|a) = P(b)P(a|b)$.

Bayes Rule

- Bayes rule $P(a|b) = \frac{P(b|a)P(a)}{P(b)}$.
- In general, $P(a|b,C) = \frac{P(b|a,C)P(a|C)}{P(b|C)}$ where C can be one or more random variables.
- Bayesian approach: when θ is model parameter, D is observed data, we have

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)},$$

  - $p(\theta)$ is the prior,
  - $p(D|\theta)$ the likelihood function (of θ, *not normalized*: $\int p(D|\theta)\,d\theta \neq 1$),
  - $p(D) = \int p(D|\theta)p(\theta)\,d\theta$ the evidence,
  - $p(\theta|D)$ the posterior.

Independence

- The product rule can be simplified as $P(a, b) = P(a)P(b)$ iff A and B are independent
- Equivalently, $P(a|b) = P(a)$, $P(b|a) = P(b)$.

Probability density

- A continuous random variable x has a probability density function (pdf) $p(x) \in [0, \infty]$.
- $p(x) > 1$ is possible! Integrates to 1.

$$\int_{-\infty}^{\infty} p(x)\, dx = 1$$

- $P(x_1 < X < x_2) = \int_{x_1}^{x_2} p(x)\, dx$
- Marginalization $p(x) = \int_{-\infty}^{\infty} p(x, y)\, dy$

Expectation and Variance

- The expectation ("mean" or "average") of a function f under the probability distribution P is

$$\mathbb{E}_P[f] = \sum_a P(a) f(a)$$

$$\mathbb{E}_p[f] = \int_x p(x) f(x)\, dx$$

- In particular if $f(x) = x$, this is the mean of the random variable x.
- The variance of f is

$$\mathrm{Var}(f) = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])^2] = \mathbb{E}[f(x)^2] - \mathbb{E}[f(x)]^2$$

- The standard deviation is $std(f) = \sqrt{\mathrm{Var}(f)}$.

Multivariate Statistics

- When x, y are vectors, $\mathbb{E}[x]$ is the mean vector
- $\mathrm{Cov}(x, y)$ is the covariance matrix with i, j-th entry being $\mathrm{Cov}(x_i, y_j)$.

$$\mathrm{Cov}(x, y) = \mathbb{E}_{x,y}[(x - \mathbb{E}[x])(y - \mathbb{E}[y])] = \mathbb{E}_{x,y}[xy] - \mathbb{E}[x]\mathbb{E}[y]$$

Some Discrete Distributions

- **Dirac** or point mass distribution $X \sim \delta_a$ if $P(X = a) = 1$
- **Binomial**. n (number of trials) and p (head probability)

$$f(x) = \begin{cases} \binom{n}{x} p^x (1-p)^{n-x} & \text{for } x = 0, 1, \ldots, n \\ 0 & \text{otherwise} \end{cases}$$

- **Bernoulli**. Binomial with $n = 1$.
- **Multinomial** $p = (p_1, \ldots, p_d)^\top$ (d-sided die)

$$f(x) = \begin{cases} \binom{n}{x_1, \ldots, x_d} \prod_{k=1}^d p_k^{x_k} & \text{if } \sum_{k=1}^d x_k = n \\ 0 & \text{otherwise} \end{cases}$$

More Discrete Distributions

- **Poisson**. $X \sim \text{Poisson}(\lambda)$ if

$$f(x) = e^{-\lambda} \frac{\lambda^x}{x!}$$

  for $x = 0, 1, 2, \ldots$.
- λ the rate or intensity parameter
- mean: λ, variance: λ
- If $X_1 \sim \text{Poisson}(\lambda_1)$ and $X_2 \sim \text{Poisson}(\lambda_2)$ then $X_1 + X_2 \sim \text{Poisson}(\lambda_1 + \lambda_2)$.
- This is a distribution on unbounded counts with a probability mass function "hump" (mode at $\lceil \lambda \rceil - 1$).

Some Continuous Distributions

- **Gaussian (Normal)**: $X \sim N(\mu, \sigma^2)$ with parameters $\mu \in \mathbb{R}$ (the mean) and σ^2 (the variance)

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

- σ is the standard deviation.
- If $\mu = 0, \sigma = 1$, X has a *standard normal distribution*.
- (Scaling) If $X \sim N(\mu, \sigma^2)$, then $Z = (X - \mu)/\sigma \sim N(0, 1)$
- (Independent sum) If $X_i \sim N(\mu_i, \sigma_i^2)$ are independent, then $\sum_i X_i \sim N\left(\sum_i \mu_i, \sum_i \sigma_i^2\right)$

Some Continuous Distributions

- **Multivariate Gaussian.** Let $x, \mu \in \mathbb{R}^d$, $\Sigma \in S_+^d$ a symmetric, positive definite matrix of size $d \times d$. Then $X \sim N(\mu, \Sigma)$ with PDF

$$f(x) = \frac{1}{|\Sigma|^{1/2}(2\pi)^{d/2}} \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right).$$

- μ is the mean vector, Σ is the covariance matrix, $|\Sigma|$ its determinant, and Σ^{-1} its inverse

Marginal and Conditional of Gaussian

- If two (groups of) variables x, y are jointly Gaussian:

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} A & C \\ C^\top & B \end{bmatrix} \right) \tag{1}$$

- (Marginal) $x \sim N(\mu_x, A)$
- (Conditional) $y|x \sim N(\mu_y + C^\top A^{-1}(x - \mu_x), B - C^\top A^{-1} C)$

More Continuous Distributions

- The *Gamma function* (not distribution) is $\Gamma(\alpha) = \int_0^\infty y^{\alpha-1} e^{-y} dy$ with $\alpha > 0$.
- Generalizes factorial: $\Gamma(n) = (n-1)!$ when n is a positive integer.
- $\Gamma(\alpha+1) = \alpha\Gamma(\alpha)$ for $\alpha > 0$.
- X has a *Gamma distribution* $X \sim \text{Gamma}(\alpha, \beta)$ with shape parameter $\alpha > 0$ and scale parameter $\beta > 0$

$$f(x) = \frac{1}{\beta^\alpha \Gamma(\alpha)} x^{\alpha-1} e^{-x/\beta}, \; x > 0.$$

- Conjugate prior for Poisson rate.

More Continuous Distributions

- **Beta**. $X \sim \text{Beta}(\alpha, \beta)$ with parameters $\alpha, \beta > 0$, if

$$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}, \; x \in (0,1).$$

  A draw from a beta distribution can be thought of as generating a (biased) coin.
- $\text{Beta}(1,1)$ is uniform in $[0,1]$.
- $\text{Beta}(\alpha < 1, \beta < 1)$ has a U-shape.
- $\text{Beta}(\alpha > 1, \beta > 1)$ is unimodal with mean $\alpha/(\alpha+\beta)$ and mode $(\alpha-1)/(\alpha+\beta-2)$.
- Beta distribution is conjugate to the binomial and Bernoulli distributions. A draw from the corresponding Bernoulli distribution can be thought of as a flip of that coin.

More Continuous Distributions

- **Dirichlet.** Multivariate version of beta. $X \sim \mathrm{Dir}(\alpha_1, \ldots, \alpha_d)$ with parameters $\alpha_i > 0$, if

$$f(x) = \frac{\Gamma(\sum_i^d \alpha_i)}{\prod_i^d \Gamma(\alpha_i)} \prod_i^d x_i^{\alpha_i - 1}$$

  where $x = (x_1, \ldots, x_d)$ with $x_i > 0, \sum_i^d x_i = 1$.
- The support is called the open $(d-1)$ dimensional simplex.
- Dirichlet is conjugate to multinomial.
- Dice factory (Dirichlet) and die rolls (multinomial)
- Modeling bag-of-word documents. Also in Dirichlet Processes.

Outline

Parametric Models

- A *statistical model* $\mathcal{H}$ is a set of distributions.
- In machine learning, we call $\mathcal{H}$ the hypothesis space.
- A *parametric model* can be parametrized by a finite number of parameters: $f(x) \equiv f(x;\theta)$ with parameter $\theta \in \mathbb{R}^d$:

$$\mathcal{H} = \left\{ f(x;\theta) : \theta \in \Theta \subset \mathbb{R}^d \right\}$$

where Θ is the *parameter space*.

Parametric Models

- We denote the expectation

$$\mathbb{E}_\theta(g) = \int_x g(x) f(x;\theta)\, dx$$

- $\mathbb{E}_\theta$ means $\mathbb{E}_{x\sim f(x;\theta)}$, not over different θ's.
- For parametric model $\mathcal{H} = \{N(\mu, 1) : \mu \in \mathbb{R}\}$, given iid data $x_1, \ldots, x_n$, the optimal estimator of the mean is $\widehat{\mu} = \frac{1}{n}\sum x_i$.
- All (parametric) models are wrong. Some are more useful than others.

Nonparametric model

- A *nonparametric model* cannot be parametrized by a fixed number of parameters.
- Model complexity grows indefinitely with sample size
- Example: $\mathcal{H} = \{P : Var_P(X) < \infty\}$.
- Given iid data $x_1, \ldots, x_n$, the optimal estimator of the mean is again $\widehat{\mu} = \frac{1}{n} \sum x_i$.
- Nonparametric makes weaker model assumptions and thus is preferred.
- But parametric models converge faster and are more practical.

Estimation

- Given $X_1 \dots X_n \sim F \in \mathcal{H}$, an *estimator* $\widehat{\theta}_n$ is any function of $X_1 \dots X_n$ that attempts to estimate a parameter θ.
- This is the "learning" in machine learning!
- Example: In classification $X_i = (x_i, y_i)$ and $\widehat{\theta}_n$ is the learned model.
- $\widehat{\theta}_n$ is a random variable because the training set is random.
- An estimator is *consistent* if $\widehat{\theta}_n \xrightarrow{P} \theta$.
- Consistent estimators learn the correct model with more training data eventually.

Bias

- Since $\widehat{\theta}_n$ is a random variable, it has an expectation $\mathbb{E}_\theta(\widehat{\theta}_n)$
- $\mathbb{E}_\theta$ is w.r.t. the joint distribution $f(x_1, \ldots, x_n; \theta) = \prod_{i=1}^n f(x_i; \theta)$.
- The *bias* of the estimator is

$$\text{bias}(\widehat{\theta}_n) = \mathbb{E}_\theta(\widehat{\theta}_n) - \theta$$

- An estimator is *unbiased* if $\text{bias}(\widehat{\theta}_n) = 0$.
- The *standard error* of an estimator is $\text{se}(\widehat{\theta}_n) = \sqrt{\text{Var}_\theta(\widehat{\theta}_n)}$
- Example: Let $\hat{\mu} = \frac{1}{n}\sum_i x_i$, where $x_i \sim N(0, 1)$. Then the standard deviation of x_i is 1 regardless of n. In contrast, $\text{se}(\hat{\mu}) = 1/\sqrt{n} = n^{-\frac{1}{2}}$ which decreases with n.

MSE

- The *mean squared error* of an estimator is

$$\mathrm{mse}(\widehat{\theta}_n) = \mathbb{E}_\theta \left((\widehat{\theta}_n - \theta)^2 \right)$$

- Bias-variance decomposition

$$\mathrm{mse}(\widehat{\theta}_n) = \mathrm{bias}^2(\widehat{\theta}_n) + \mathrm{se}^2(\widehat{\theta}_n) = \mathrm{bias}^2(\widehat{\theta}_n) + \mathrm{Var}_\theta(\widehat{\theta}_n)$$

- If $\mathrm{bias}(\widehat{\theta}_n) \to 0$ and $\mathrm{Var}_\theta(\widehat{\theta}_n) \to 0$ then $\mathrm{mse}(\widehat{\theta}_n) \to 0$.
- This implies $\widehat{\theta}_n \xrightarrow{P} \theta$, so that $\widehat{\theta}_n$ is consistent.

Maximum Likelihood

- Let $x_1, \ldots, x_n \sim f(x; \theta)$ where $\theta \in \Theta$.
- The *likelihood function* is

$$L_n(\theta) = f(x_1, \ldots, x_n; \theta) = \prod_{i=1}^{n} f(x_i; \theta)$$

- The *log likelihood function* is $\ell_n(\theta) = \log L_n(\theta)$.
- The maximum likelihood estimator (MLE) is

$$\widehat{\theta}_n = \text{argmax}_{\theta \in \Theta} L_n(\theta) = \text{argmax}_{\theta \in \Theta} \ell_n(\theta)$$

MLE examples

- The MLE for $p(head)$ from n coin flips is count(head)$/n$
- The MLE for $X_1, \ldots, X_N \sim N(\mu, \sigma^2)$ is $\widehat{\mu} = 1/n \sum_i X_i$ and $\widehat{\sigma}^2 = 1/n \sum (X_i - \widehat{\mu})^2$.
- The MLE does not always agree with intuition. The MLE for $X_1, \ldots, X_n \sim \text{uniform}(0, \theta)$ is $\widehat{\theta} = \max(X_1, \ldots, X_n)$.

Properties of MLE

- When $\mathcal{H}$ is identifiable, under certain conditions (see Wasserman Theorem 9.13), the MLE $\widehat{\theta}_n \xrightarrow{P} \theta^*$, where θ^* is the true value of the parameter θ. That is, the MLE is consistent.
- Asymptotic Normality: Let $se = \sqrt{Var_\theta(\widehat{\theta}_n)}$. Under appropriate regularity conditions, $se \approx \sqrt{1/I_n(\theta)}$ where $I_n(\theta)$ is the Fisher information, and
$$\frac{\widehat{\theta}_n - \theta}{se} \rightsquigarrow N(0,1)$$
- The MLE is asymptotically efficient (achieves the Cramér-Rao lower bound), "best" among unbiased estimators.

Frequentist statistics

- Probability refers to limiting relative frequency.
- Data are random.
- Estimators are random because they are functions of data.
- Parameters are fixed, unknown constants not subject to probabilistic statements.
- Procedures are subject to probabilistic statements, for example 95% confidence intervals trap the true parameter value 95
- Classifiers, even learned with deterministic procedures, are random because the training set is random.
- PAC bound is frequentist. Most procedures in machine learning are frequentist methods.

Bayesian statistics

- Probability refers to degree of belief.
- Inference about a parameter θ is by producing a probability distributions on it.
- Starts with *prior* distribution $p(\theta)$.
- *Likelihood function* $p(x \mid \theta)$, a function of θ not x.
- After observing data x, one applies the Bayes rule to obtain the *posterior*

$$p(\theta \mid x) = \frac{p(\theta)p(x \mid \theta)}{\int p(\theta')p(x \mid \theta')d\theta'} = \frac{1}{Z}p(\theta)p(x \mid \theta)$$

- $Z \equiv \int p(\theta')p(x \mid \theta')d\theta' = p(x)$ is the *normalizing constant* or *evidence*.
- Prediction by integrating parameters out:

$$p(x \mid Data) = \int p(x \mid \theta)p(\theta \mid Data)d\theta$$

Frequentist vs Bayesian in machine learning

- Frequentists produce a *point estimate* $\hat{\theta}$ from Data, and predict with $p(x \mid \hat{\theta})$.
- Bayesians keep the posterior distribution $p(\theta \mid Data)$, and predict by integrating over θs.
- Bayesian integration is often intractable, need either "nice" distributions or approximations.
- The *maximum a posteriori* (MAP) estimate

$$\theta^{MAP} = \text{argmax}_\theta p(\theta \mid x)$$

is a point estimate and not Bayesian.

Outline

Regularization for Maximum Likelihood

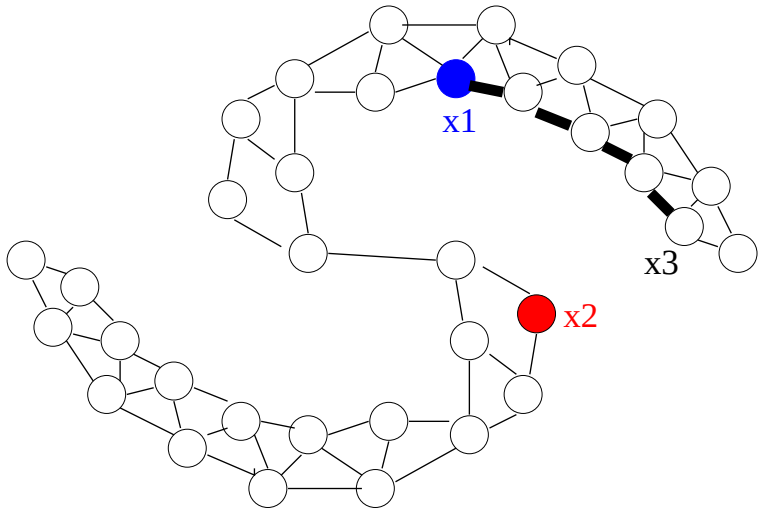
- Recall the MLE $\widehat{\theta}_n = \text{argmax}_{\theta \in \Theta} \ell_n(\theta)$
- Can overfit.
- Regularized likelihood

$$\widehat{\theta}_n = \text{argmin}_{\theta \in \Theta} - \ell_n(\theta) + \lambda \Omega(\theta)$$

- $\Omega(\theta)$ is the regularizer, for example $\Omega(\theta) = \|\theta\|^2$.
- Coincides with MAP estimate with prior distribution $p(\theta) \propto \exp(-\lambda \Omega(\theta))$

Graph-based regularization

- Nodes: $x_1 \dots x_n$, $\theta = \mathbf{f} = (f(x_1), \dots, f(x_n))$
- Edges: similarity weights computed from features, e.g.,
  - k-nearest-neighbor graph, unweighted (0, 1 weights)
  - fully connected graph, weight decays with distance $w = \exp\left(-\|x_i - x_j\|^2/\sigma^2\right)$
  - ϵ-radius graph
- Assumption Nodes connected by heavy edge tend to have the same value.



Graph energy

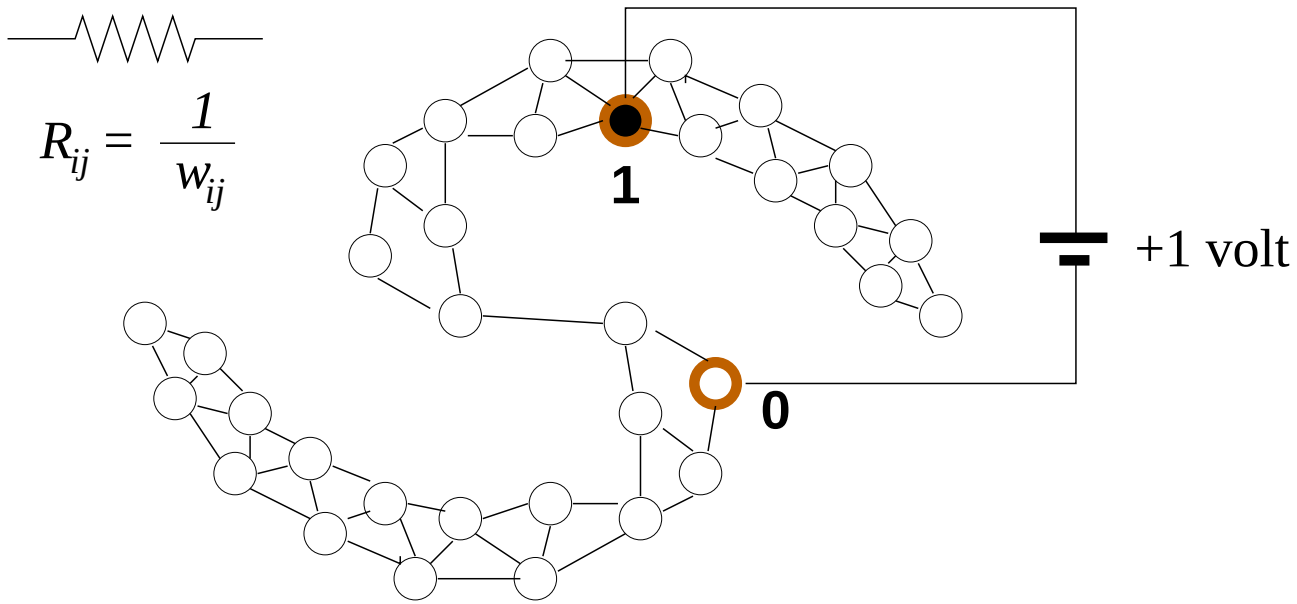
- f incurs the energy

$$\sum_{i \sim j} w_{ij} (f(x_i) - f(x_j))^2$$

- smooth f has small energy
- constant f has zero energy

An electric network interpretation

- Edges are resistors with conductance w_{ij}
- Nodes clamped at voltages specified by f
- Energy = heat generated by the network in unit time



The graph Laplacian

We can express the energy of f in closed-form using the graph Laplacian.

- $n \times n$ weight matrix W on $X_l \cup X_u$
  - symmetric, non-negative
- Diagonal degree matrix D: $D_{ii} = \sum_{j=1}^{n} W_{ij}$
- Graph Laplacian matrix Δ

$$\Delta = D - W$$

- The energy

$$\sum_{i \sim j} w_{ij}(f(x_i) - f(x_j))^2 = f^\top \Delta f$$

Graph Laplacian as a Regularizer

- Regression problem with training data $x_i \in \mathbb{R}^d, y_i \in \mathbb{R}$, $i = 1 \dots n$
- Allow $f(X_i)$ to be different from Y_i, but penalize the difference with a Gaussian log likelihood
- Regularizer $\Omega(\mathbf{f}) = f^\top \Delta f$

$$\min_f \sum_{i=1}^{n} (f(x_i) - y_i)^2 + \lambda f^\top \Delta f$$

- Equivalent to MAP estimate with
  - Gaussian likelihood $y_i = f(x_i) + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$, and
  - Gaussian Random Field prior $p(f) = \frac{1}{Z} \exp\left(-\lambda f^\top \Delta f\right)$

Graph Spectrum and Regularization

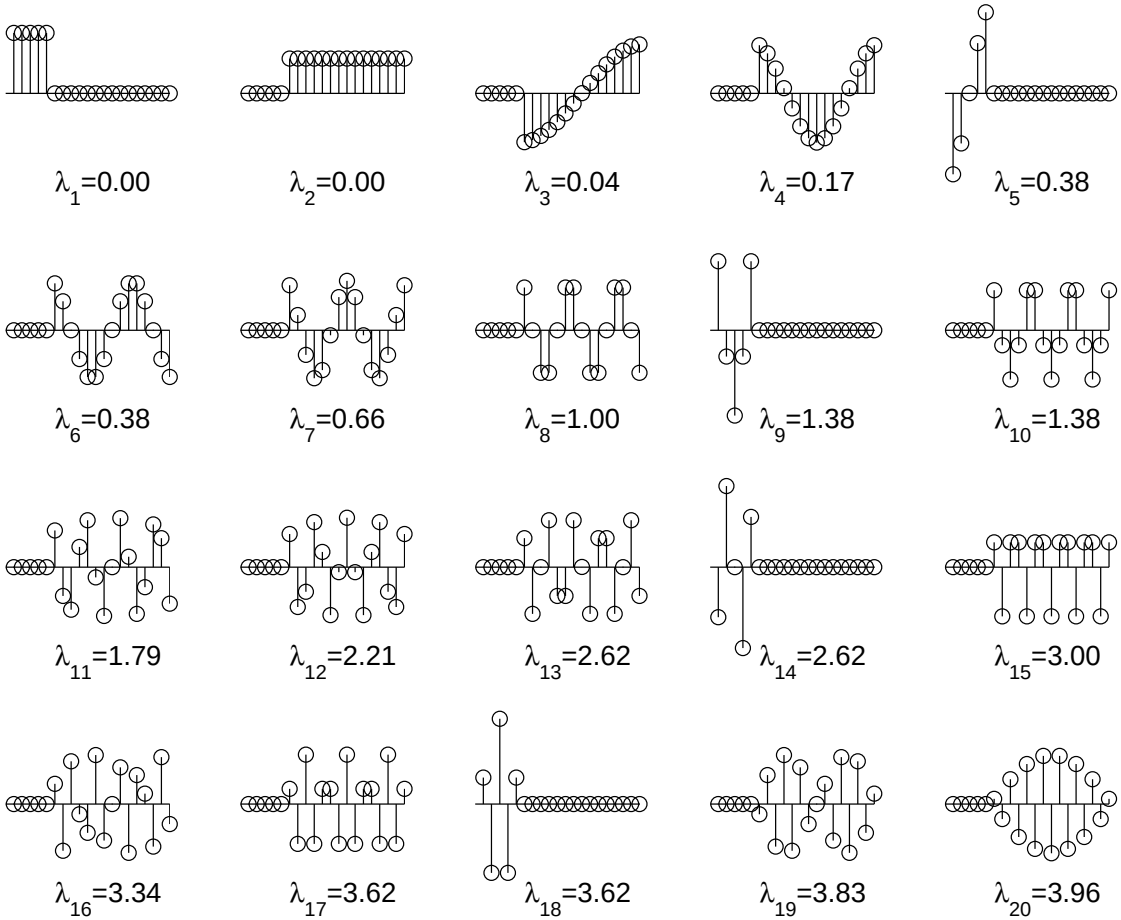
Assumption: labels are "smooth" on the graph, characterized by the graph spectrum (eigen-values/vectors $\{(\lambda_i, \phi_i)\}_{i=1}^n$ of the Laplacian L):

- $L = \sum_{i=1}^n \lambda_i \phi_i \phi_i^\top$
- a graph has k connected components if and only if $\lambda_1 = \ldots = \lambda_k = 0$.
- the corresponding eigenvectors are constant on individual connected components, and zero elsewhere.
- any $\mathbf{f}$ on the graph can be represented as $\mathbf{f} = \sum_{i=1}^n a_i \phi_i$
- graph regularizer $\mathbf{f}^\top L \mathbf{f} = \sum_{i=1}^n a_i^2 \lambda_i$
- smooth function $\mathbf{f}$ uses smooth basis (those with small λ_i)

Example graph spectrum

The graph

Eigenvalues and eigenvectors of the graph Laplacian



Outline

Comparing Estimators

- Training set $D = (x_1, \ldots, x_n) \sim p(x; \theta)$
- Learned model: $\hat{\theta} \equiv \hat{\theta}(D)$ an estimator of θ based on data D.
- *Loss function* $L(\theta, \hat{\theta}) : \Theta \times \Theta \mapsto \mathbb{R}_+$
- squared loss $L(\theta, \hat{\theta}) = (\theta - \hat{\theta})^2$
- 0-1 loss $L(\theta, \hat{\theta}) = \begin{cases} 0 & \theta = \hat{\theta} \\ 1 & \theta \neq \hat{\theta} \end{cases}$
- KL loss $L(\theta, \hat{\theta}) = \int p(x; \theta) \log \left(\frac{p(x;\theta)}{p(x;\hat{\theta})} \right) dx$
- Since D is random, both $\hat{\theta}(D)$ and $L(\theta, \hat{\theta})$ are random variables

Risk

- The *risk* $R(\theta, \hat{\theta})$ is the expected loss

$$R(\theta, \hat{\theta}) = \mathbb{E}_D[L(\theta, \hat{\theta}(D))]$$

- $\mathbb{E}_D$ averaged over training sets D sampled from the true θ
- The risk is the "average training set" behavior of a learning algorithm when the world is θ
- Not computable: we don't know which θ the world is in.
- Example: Let $D = X_1 \sim N(\theta, 1)$. Let $\hat{\theta}_1 = X_1$ and $\hat{\theta}_2 = 3.14$. Assume squared loss. Then $R(\theta, \hat{\theta}_1) = 1$ (hint: variance), $R(\theta, \hat{\theta}_2) = \mathbb{E}_D(\theta - 3.14)^2 = (\theta - 3.14)^2$.
- Smart learning algorithm $\hat{\theta}_1$ and a dumb one $\hat{\theta}_2$. However, for tasks $\theta \in (3.14 - 1, 3.14 + 1)$ the dumb algorithm is better.

Minimax Estimator

- *maximum risk*

$$R^{max}(\hat{\theta}) = \sup_{\theta} R(\theta, \hat{\theta})$$

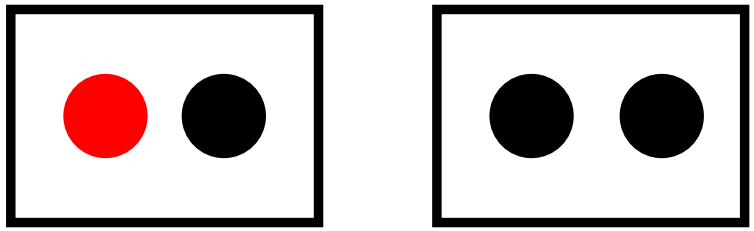
- The minimax estimator $\hat{\theta}^{minimax}$ minimizes the maximum risk

$$\hat{\theta}^{minimax} = \arg \inf_{\hat{\theta}} \sup_{\theta} R(\theta, \hat{\theta})$$

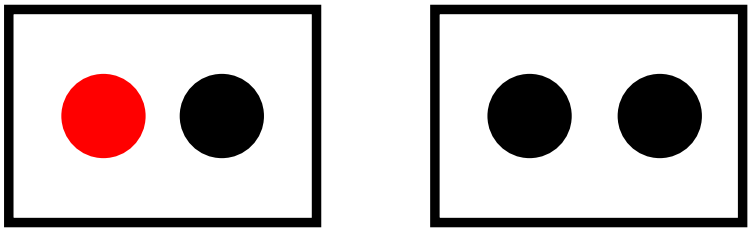
- The infimum is over all estimators $\hat{\theta}$.
- The minimax estimator is the "best" in guarding against the worst possible world.

Outline

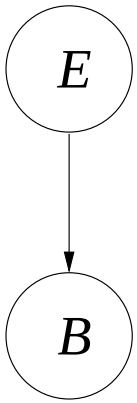
The envelope quiz



The envelope quiz



- Random variables $E \in \{1, 0\}, B \in \{r, b\}$
- $P(E = 1) = P(E = 0) = 1/2$
- $P(B = r \mid E = 1) = 1/2, P(B = r \mid E = 0) = 0$
- We ask: $P(E = 1 \mid B = b) \geq 1/2$?
- $P(E = 1 \mid B = b) = \frac{P(B=b|E=1)P(E=1)}{P(B=b)} = \frac{1/2 \times 1/2}{3/4} = 1/3$
- Switch.
- The graphical model:



Probabilistic Reasoning

- The world is reduced to a set of random variables $x_1, \ldots, x_n$
  - e.g. $(x_1, \ldots, x_{n-1})$ a feature vector, $x_n \equiv y$ the class label
- Inference: given joint distribution $p(x_1, \ldots, x_n)$, compute $p(X_Q \mid X_E)$ where $X_Q \cup X_E \subseteq \{x_1 \ldots x_n\}$
  - e.g. $Q = \{n\}$, $E = \{1 \ldots n-1\}$, by the definition of conditional

$$p(x_n \mid x_1, \ldots, x_{n-1}) = \frac{p(x_1, \ldots, x_{n-1}, x_n)}{\sum_v p(x_1, \ldots, x_{n-1}, x_n = v)}$$

- Learning: estimate $p(x_1, \ldots, x_n)$ from training data $X^{(1)}, \ldots, X^{(N)}$, where $X^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})$

It is difficult to reason with uncertainty

- joint distribution $p(x_1, \ldots, x_n)$
  - exponential naïve storage (2^n for binary r.v.)
  - hard to interpret (conditional independence)
- inference $p(X_Q \mid X_E)$
  - Often can't afford to do it by brute force
- If $p(x_1, \ldots, x_n)$ not given, estimate it from data
  - Often can't afford to do it by brute force

Graphical models

- Graphical models: efficient representation, inference, and learning on $p(x_1, \ldots, x_n)$, exactly or approximately
- Two main "flavors":
  - directed graphical models = Bayesian Networks (often frequentist instead of Bayesian)
  - undirected graphical models = Markov Random Fields
- Key idea: make conditional independence explicit

Outline

Bayesian Network

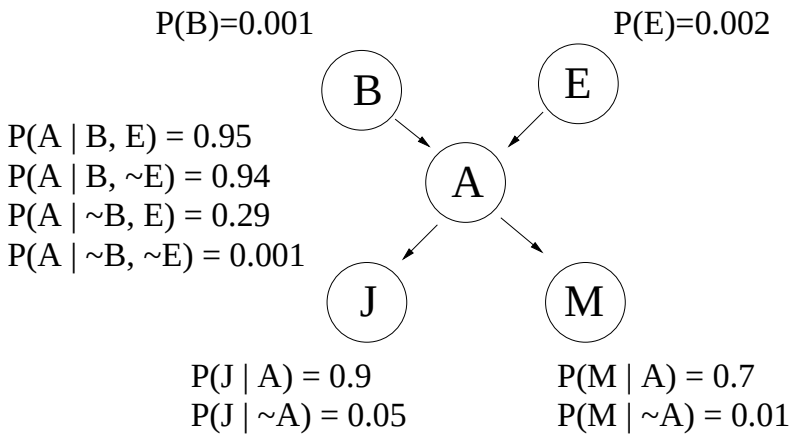
- Directed graphical models are also called Bayesian networks
- A directed graph has nodes $X = (x_1, \ldots, x_n)$, some of them connected by directed edges $x_i \to x_j$
- A cycle is a directed path $x_1 \to \ldots \to x_k$ where $x_1 = x_k$
- A directed acyclic graph (DAG) contains no cycles
- A Bayesian network on the DAG is a family of distributions satisfying

$$\{p \mid p(X) = \prod_i p(x_i \mid Pa(x_i))\}$$

  where $Pa(x_i)$ is the set of parents of x_i.
- $p(x_i \mid Pa(x_i))$ is the conditional probability distribution (CPD) at x_i
- By specifying the CPDs for all i, we specify a particular distribution $p(X)$

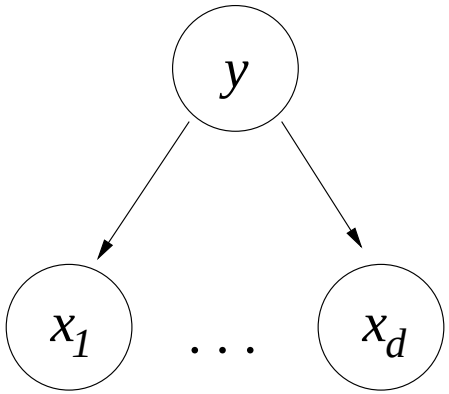
Example: Alarm

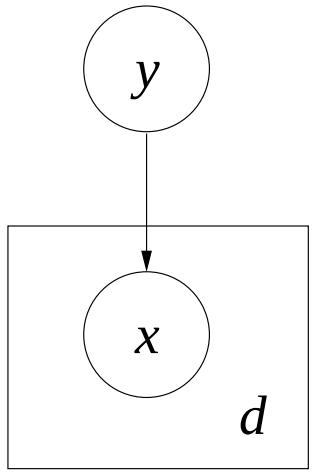
Binary variables



$$
\begin{aligned}
& P(B, \sim E, A, J, \sim M) \\
= \; & P(B)P(\sim E)P(A \mid B, \sim E)P(J \mid A)P(\sim M \mid A) \\
= \; & 0.001 \times (1 - 0.002) \times 0.94 \times 0.9 \times (1 - 0.7) \\
\approx \; & .000253
\end{aligned}
$$

Example: Naive Bayes

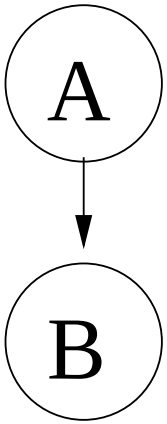


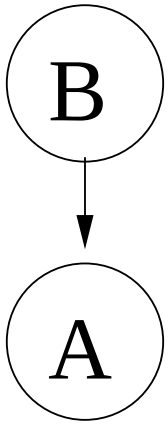


- $p(y, x_1, \ldots x_d) = p(y) \prod_{i=1}^{d} p(x_i \mid y)$
- Used extensively in natural language processing
- Plate representation on the right

No Causality Whatsoever

P(A)=a
P(B|A)=b
P(B|~A)=c



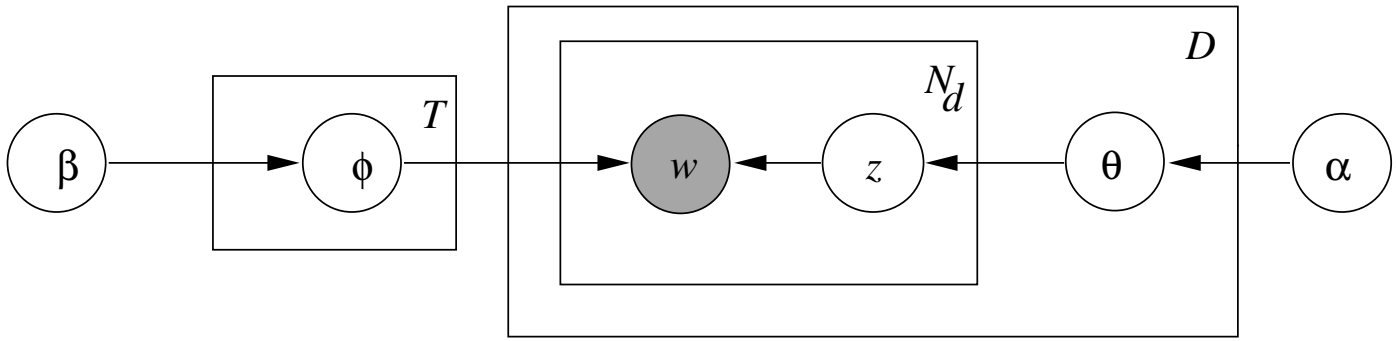


P(B)=ab+(1−a)c
P(A|B)=ab/(ab+(1−a)c)
P(A|~B)=a(1−b)/(1−ab−(1−a)c)

The two BNs are equivalent in all respects

- Bayesian networks imply no causality at all
- They only encode the joint probability distribution (hence correlation)
- However, people tend to design BNs based on causal relations

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

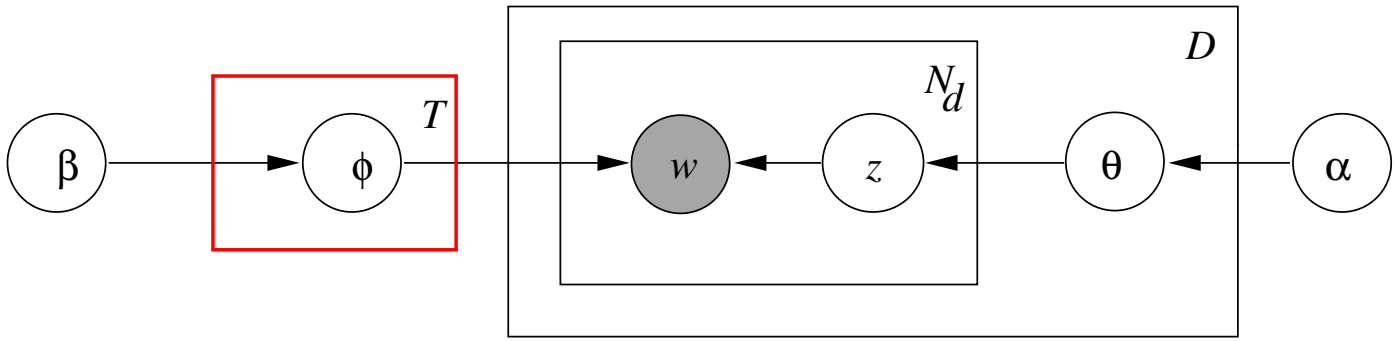
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

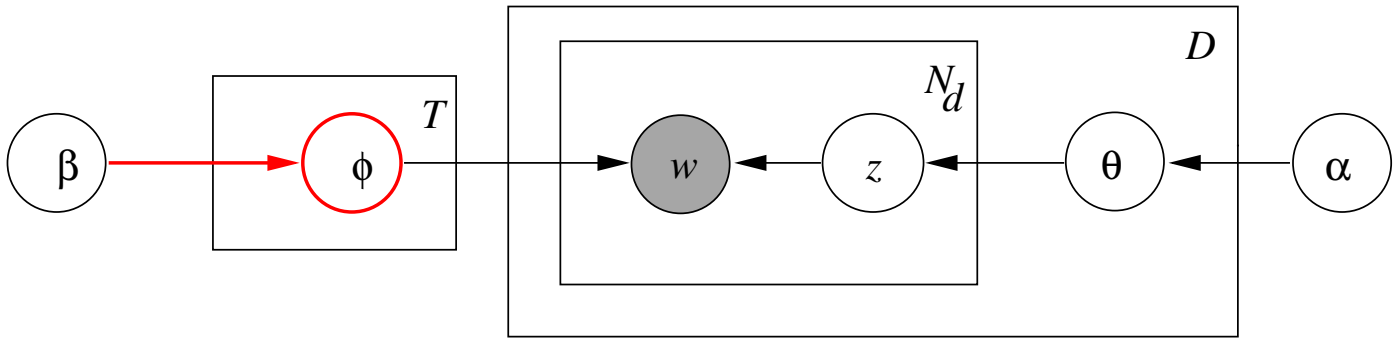
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

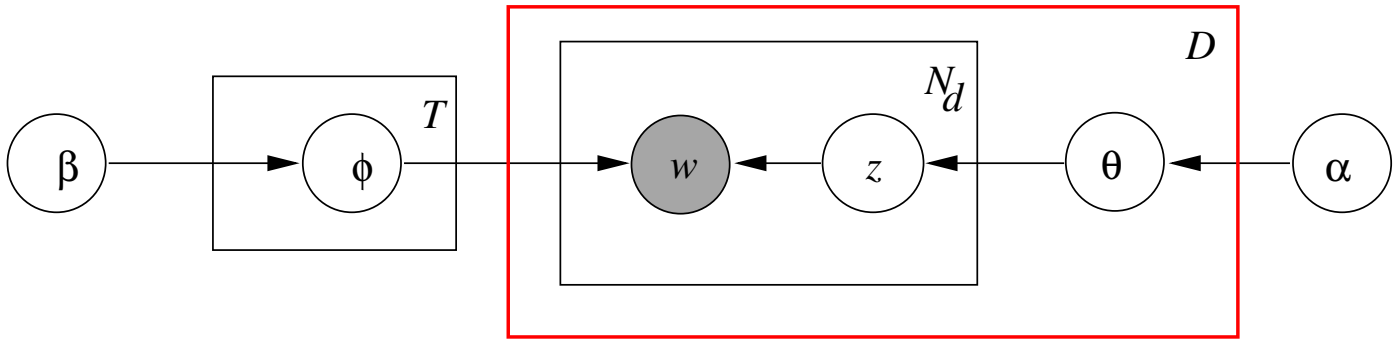
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

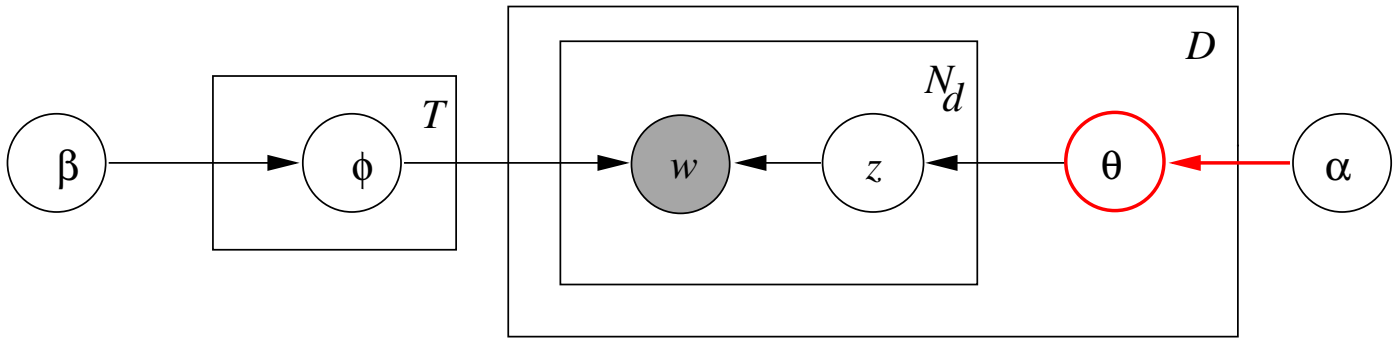
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

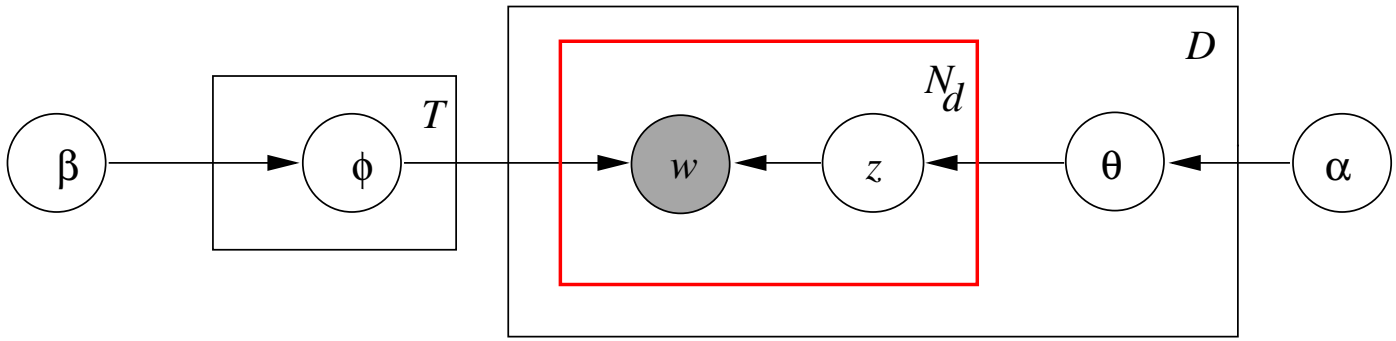
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

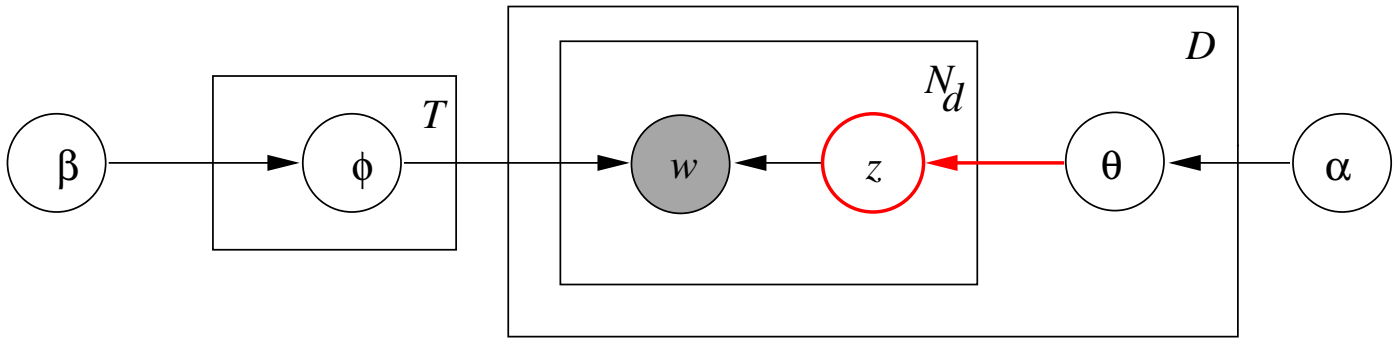
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

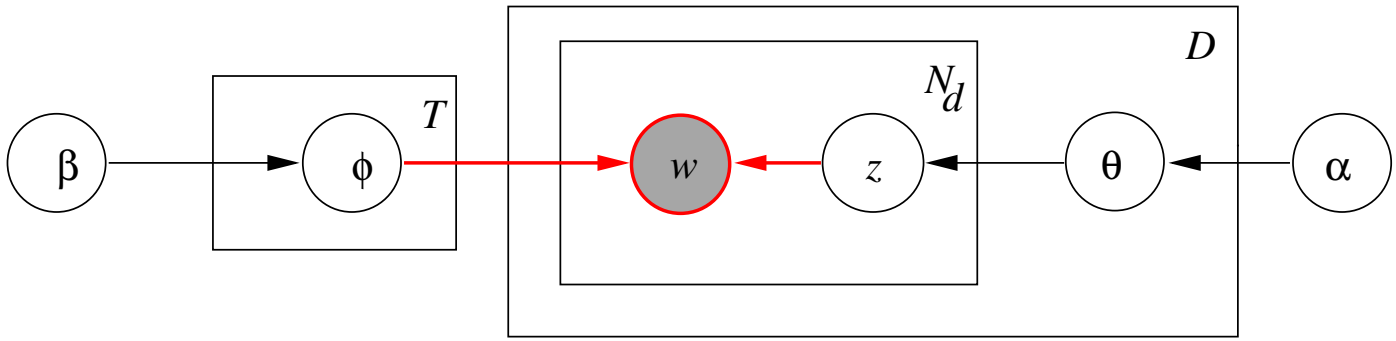
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

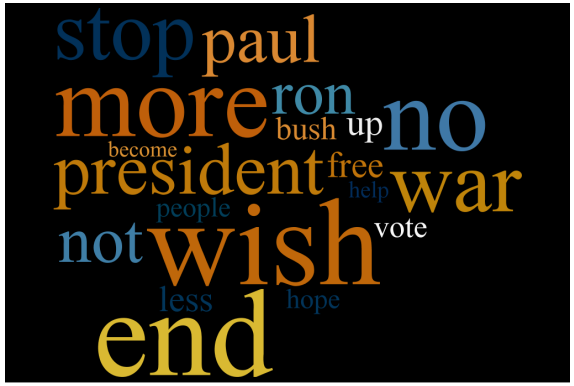
word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta)$, $\text{argmax}_{\phi,\theta} p(\phi, \theta \mid w, \alpha, \beta)$

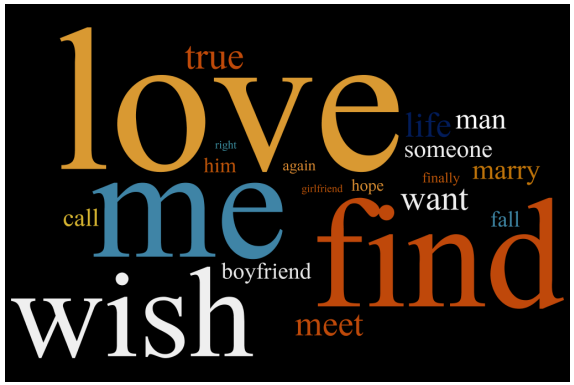
Some Topics by LDA on the Wish Corpus

$p(\text{word} \mid \text{topic})$

"troops"



"election"

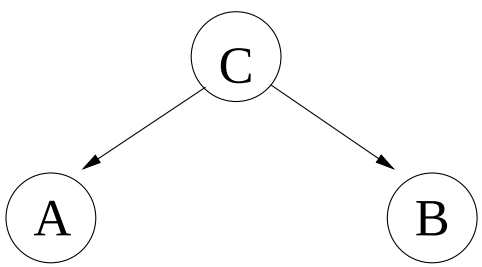


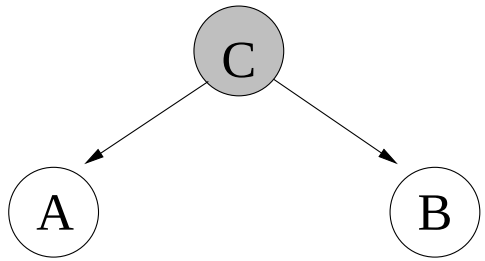
"love"

Conditional Independence

- Two r.v.s A, B are independent if $P(A, B) = P(A)P(B)$ or $P(A|B) = P(A)$ (the two are equivalent)
- Two r.v.s A, B are conditionally independent given C if $P(A, B \mid C) = P(A \mid C)P(B \mid C)$ or $P(A \mid B, C) = P(A \mid C)$ (the two are equivalent)
- This extends to groups of r.v.s
- Conditional independence in a BN is precisely specified by d-separation ("directed separation")

d-Separation Case 1: Tail-to-Tail



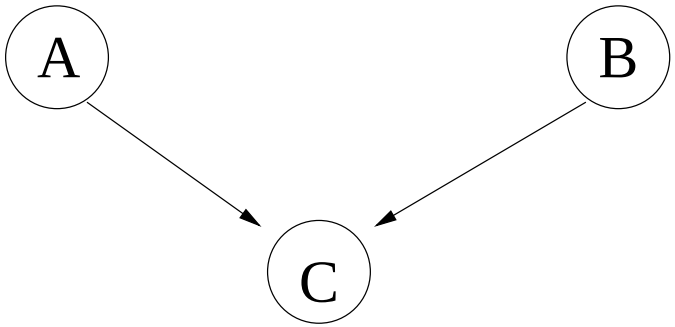


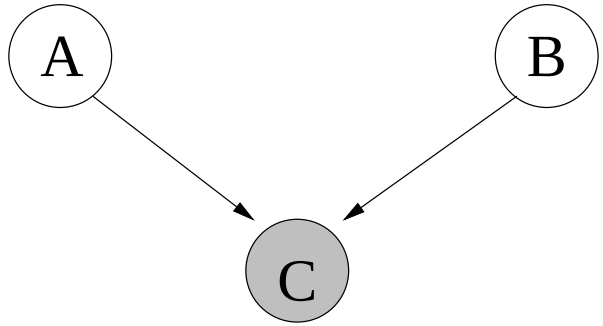
- A, B in general dependent
- A, B conditionally independent given C (observed nodes are shaded)
- An observed C is a tail-to-tail node, blocks the undirected path A-B

d-Separation Case 2: Head-to-Tail

- A, B in general dependent
- A, B conditionally independent given C
- An observed C is a head-to-tail node, blocks the path A-B

d-Separation Case 3: Head-to-Head





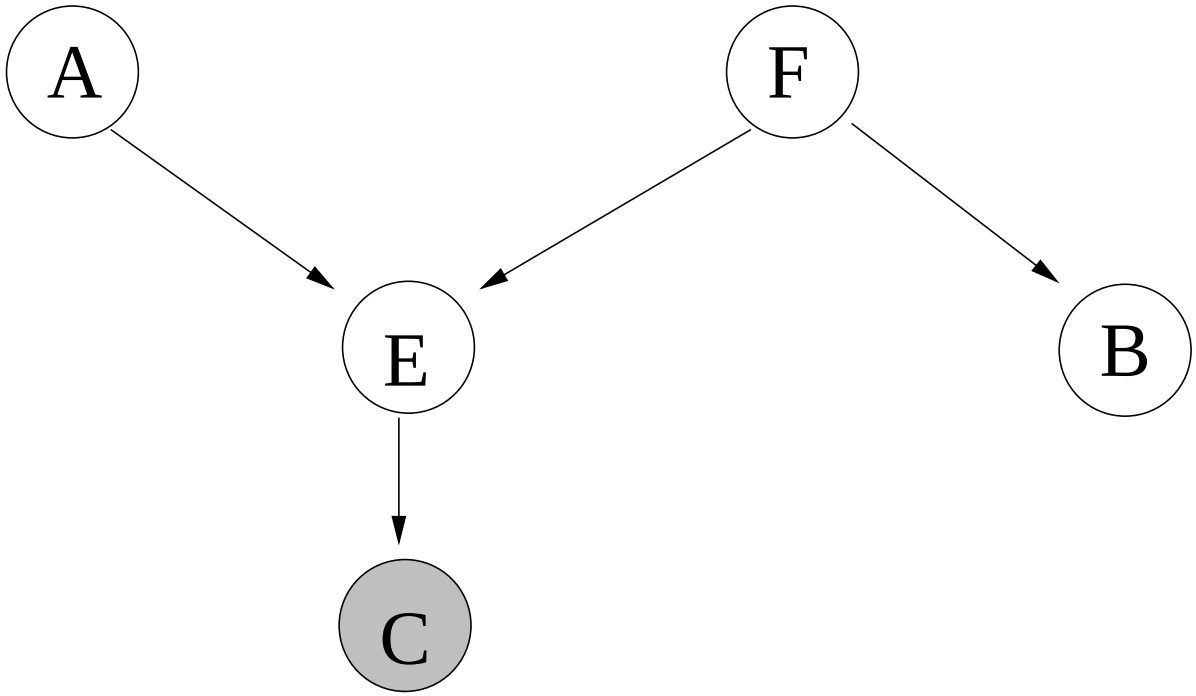
- A, B in general independent
- A, B conditionally dependent given C, or any of C's descendants
- An observed C is a head-to-head node, unblocks the path A-B

d-Separation

- Any groups of nodes A and B are conditionally independent given another group C, if all undirected paths from any node in A to any node in B are *blocked*
- A path is blocked if it includes a node x such that either
  - The path is head-to-tail or tail-to-tail at x and $x \in$ C, or
  - The path is head-to-head at x, and neither x nor any of its descendants is in C.

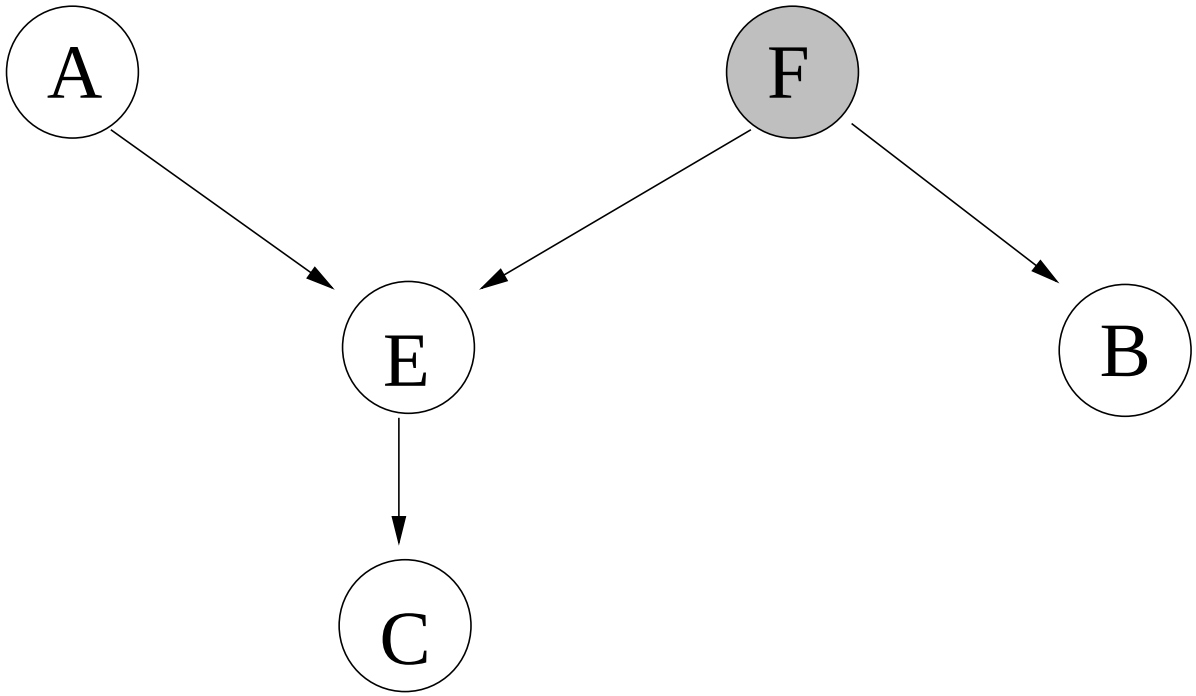
d-Separation Example 1

- The undirected path from A to B is unblocked by E (because of C), and is not blocked by F
- A, B dependent given C



d-Separation Example 2

- The path from A to B is blocked both at E and F
- A, B conditionally independent given F



Outline

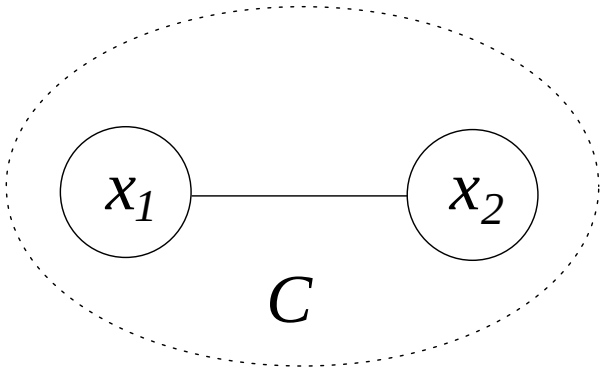
Markov Random Fields

- Undirected graphical models are also called Markov Random Fields
- The efficiency of directed graphical model (acyclic graph, locally normalized CPDs) also makes it restrictive
- A clique C in an undirected graph is a fully connected set of nodes (note: full of loops!)
- Define a nonnegative potential function $\psi_C : X_C \mapsto \mathbb{R}_+$
- An undirected graphical model is a family of distributions satisfying

$$\left\{ p \mid p(X) = \frac{1}{Z} \prod_C \psi_C(X_C) \right\}$$

- $Z = \int \prod_C \psi_C(X_C) dX$ is the partition function

Example: A Tiny Markov Random Field



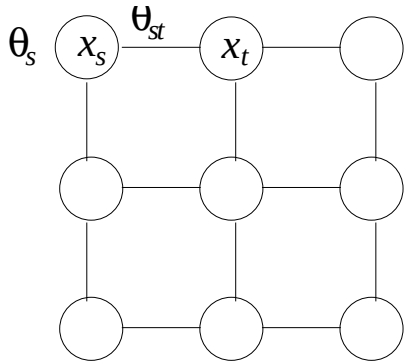
- $x_1, x_2 \in \{-1, 1\}$
- A single clique $\psi_C(x_1, x_2) = e^{ax_1x_2}$
- $p(x_1, x_2) = \frac{1}{Z} e^{ax_1x_2}$
- $Z = (e^a + e^{-a} + e^{-a} + e^a)$
- $p(1, 1) = p(-1, -1) = e^a/(2e^a + 2e^{-a})$
- $p(-1, 1) = p(1, -1) = e^{-a}/(2e^a + 2e^{-a})$
- When the parameter $a > 0$, favor homogeneous chains
- When the parameter $a < 0$, favor inhomogeneous chains

Log Linear Models

- Real-valued feature functions $f_1(X), \ldots, f_k(X)$
- Real-valued weights $w_1, \ldots, w_k$

$$p(X) = \frac{1}{Z} \exp\left(-\sum_{i=1}^{k} w_i f_i(X)\right)$$

Example: The Ising Model

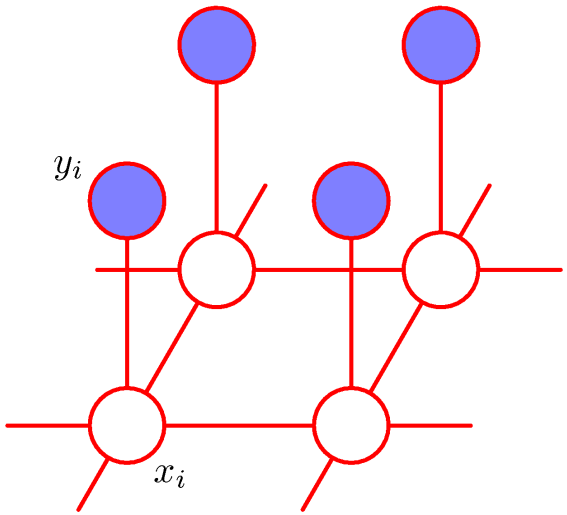


This is an undirected model with $x \in \{0, 1\}$.

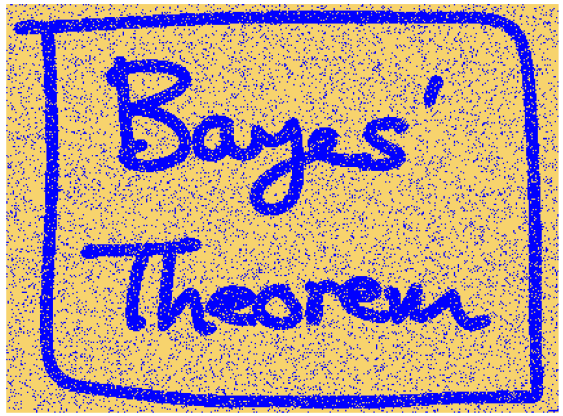
$$p_\theta(x) = \frac{1}{Z} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t\right)$$

- $f_s(X) = x_s$, $f_{st}(X) = x_s x_t$
- $w_s = -\theta_s$, $w_{st} = -\theta_{st}$

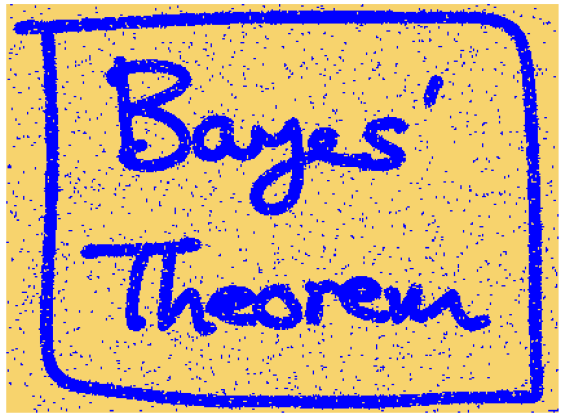
Example: Image Denoising



[From Bishop PRML]



noisy image



$\text{argmax}_X P(X|Y)$

$$p_\theta(X \mid Y) = \frac{1}{Z} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t\right)$$

$$\theta_s = \begin{cases} c & y_s = 1 \\ -c & y_s = 0 \end{cases}$$

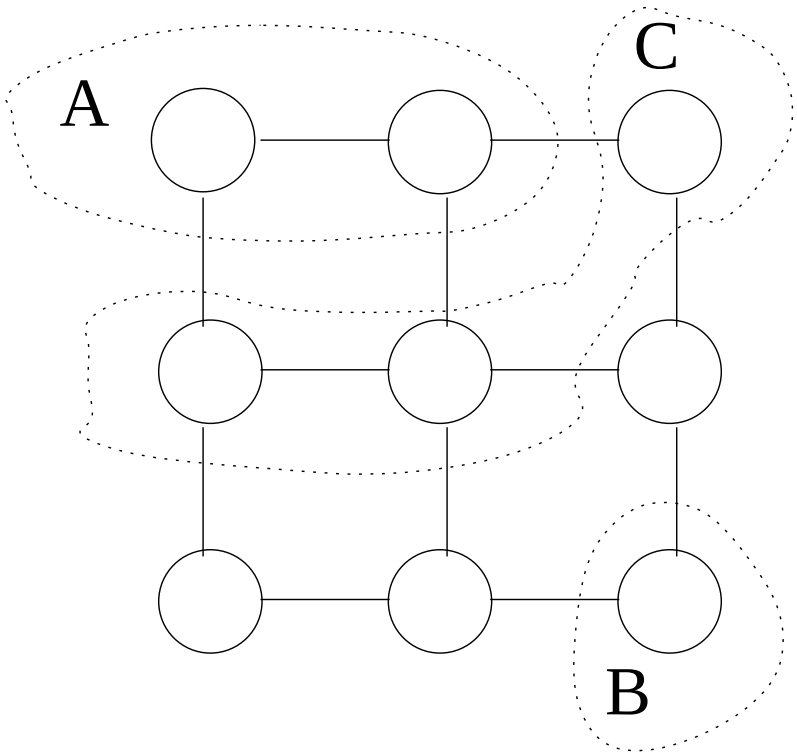
Example: Gaussian Random Field

$$p(X) \sim N(\mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(X-\mu)^\top \Sigma^{-1}(X-\mu)\right)$$

- Multivariate Gaussian
- The $n \times n$ covariance matrix Σ positive semi-definite
- Let $\Omega = \Sigma^{-1}$ be the precision matrix
- x_i, x_j are conditionally independent given all other variables, if and only if $\Omega_{ij} = 0$
- When $\Omega_{ij} \neq 0$, there is an edge between x_i, x_j

Conditional Independence in Markov Random Fields

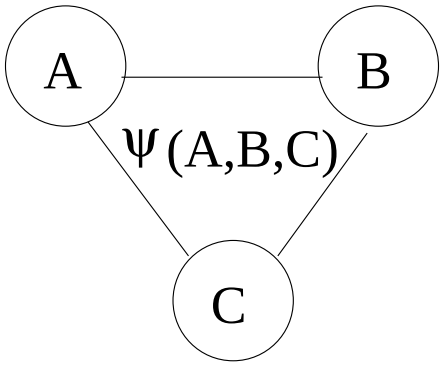
- Two group of variables A, B are conditionally independent given another group C, if:
- A, B become disconnected by removing C and all edges involving C

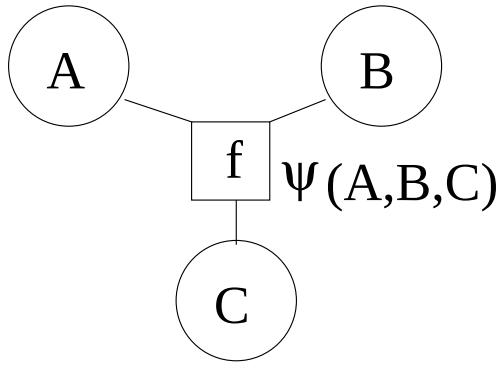


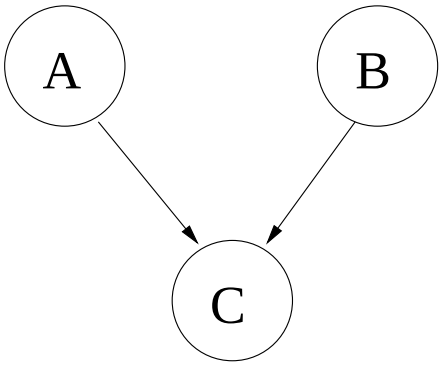
Outline

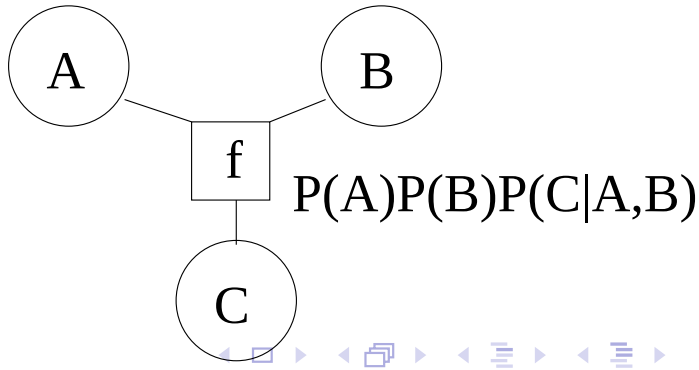
Factor Graph

- For both directed and undirected graphical models
- Bipartite: edges between a variable node and a factor node
- Factors represent computation









Outline

Inference by Monte Carlo

- Consider the inference problem $p(X_Q = c_Q \mid X_E)$ where $X_Q \cup X_E \subseteq \{x_1 \dots x_n\}$

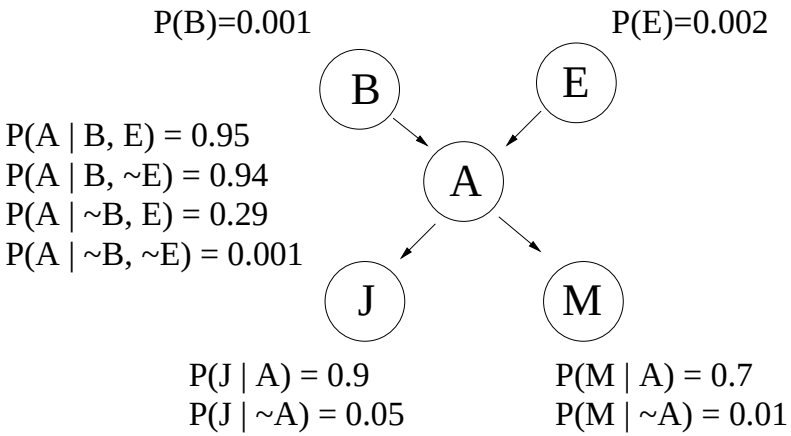
$$p(X_Q = c_Q \mid X_E) = \int 1_{(x_Q = c_Q)} p(x_Q \mid X_E) dx_Q$$

- If we can draw samples $x_Q^{(1)}, \dots x_Q^{(m)} \sim p(x_Q \mid X_E)$, an unbiased estimator is

$$p(X_Q = c_Q \mid X_E) \approx \frac{1}{m} \sum_{i=1}^{m} 1_{(x_Q^{(i)} = c_Q)}$$

- The variance of the estimator decreases as $O(1/m)$
- Inference reduces to sampling from $p(x_Q \mid X_E)$
- We discuss two methods: forward sampling and Gibbs sampling

Forward Sampling: Example



To generate a sample $X = (B, E, A, J, M)$:

1. Sample $B \sim \text{Ber}(0.001)$: $r \sim U(0, 1)$. If $(r < 0.001)$ then $B = 1$ else $B = 0$
2. Sample $E \sim \text{Ber}(0.002)$
3. If $B = 1$ and $E = 1$, sample $A \sim \text{Ber}(0.95)$, and so on
4. If $A = 1$ sample $J \sim \text{Ber}(0.9)$ else $J \sim \text{Ber}(0.05)$
5. If $A = 1$ sample $M \sim \text{Ber}(0.7)$ else $M \sim \text{Ber}(0.01)$

Inference with Forward Sampling

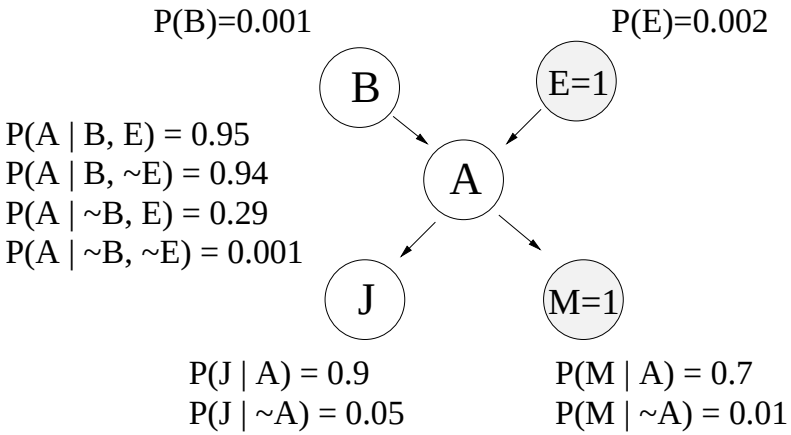
- Say the inference task is $P(B = 1 \mid E = 1, M = 1)$
- Throw away all samples except those with $(E = 1, M = 1)$

$$p(B = 1 \mid E = 1, M = 1) \approx \frac{1}{m} \sum_{i=1}^{m} 1_{(B^{(i)}=1)}$$

  where m is the number of surviving samples
- Can be highly inefficient (note $P(E = 1)$ tiny)
- Does not work for Markov Random Fields

Gibbs Sampler: Example $P(B = 1 \mid E = 1, M = 1)$

- Gibbs sampler is a Markov Chain Monte Carlo (MCMC) method.
- Directly sample from $p(x_Q \mid X_E)$
- Works for both graphical models
- Initialization:
  - Fix evidence; randomly set other variables
  - e.g. $X^{(0)} = (B = 0, E = 1, A = 0, J = 0, M = 1)$

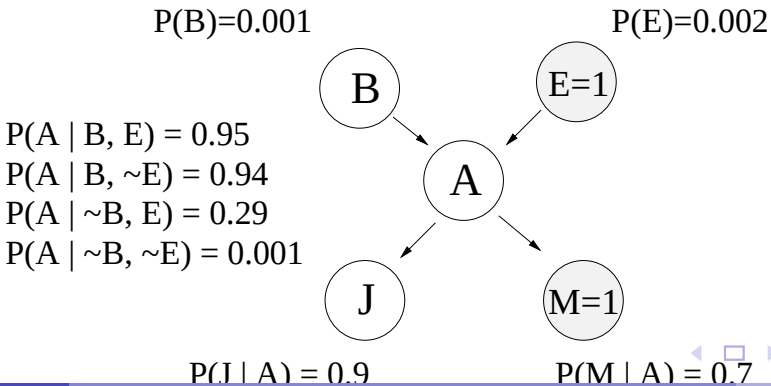


Gibbs Update

- For each non-evidence variable x_i, fixing all other nodes X_{-i}, resample its value $x_i \sim P(x_i \mid X_{-i})$
- This is equivalent to $x_i \sim P(x_i \mid \text{MarkovBlanket}(x_i))$
- For a Bayesian network MarkovBlanket(x_i) includes x_i's parents, spouses, and children

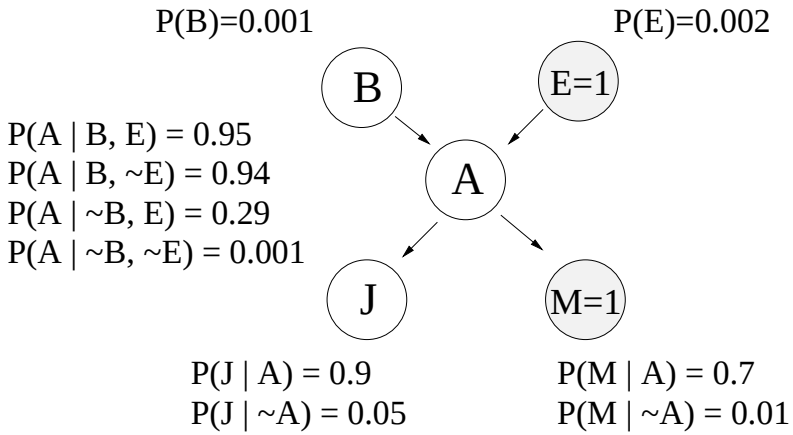
$$P(x_i \mid \text{MarkovBlanket}(x_i)) \propto P(x_i \mid Pa(x_i)) \prod_{y \in C(x_i)} P(y \mid Pa(y))$$

  where $Pa(x)$ are the parents of x, and $C(x)$ the children of x.
- For many graphical models the Markov Blanket is small.
- For example, $B \sim P(B \mid E = 1, A = 0) \propto P(B)P(A = 0 \mid B, E = 1)$

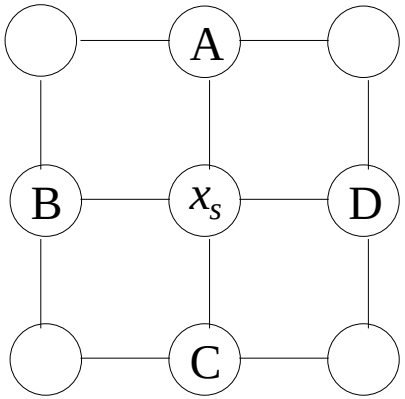


Gibbs Update

- Say we sampled $B = 1$. Then $X^{(1)} = (B = 1, E = 1, A = 0, J = 0, M = 1)$
- Starting from $X^{(1)}$, sample $A \sim P(A \mid B = 1, E = 1, J = 0, M = 1)$ to get $X^{(2)}$
- Move on to J, then repeat $B, A, J, B, A, J \ldots$
- Keep all *later* samples. $P(B = 1 \mid E = 1, M = 1)$ is the fraction of samples with $B = 1$.



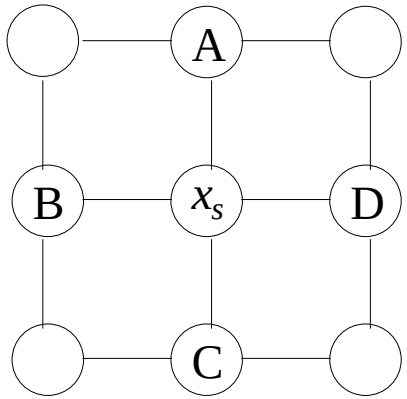
Gibbs Example 2: The Ising Model



This is an undirected model with $x \in \{0, 1\}$.

$$p_\theta(x) = \frac{1}{Z} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t\right)$$

Gibbs Example 2: The Ising Model



- The Markov blanket of x_s is A, B, C, D
- In general for undirected graphical models

$$p(x_s \mid x_{-s}) = p(x_s \mid x_{N(s)})$$

$N(s)$ is the neighbors of s.

- The Gibbs update is

$$p(x_s = 1 \mid x_{N(s)}) = \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} x_t)) + 1}$$

Gibbs Sampling as a Markov Chain

- A Markov chain is defined by a transition matrix $T(X' \mid X)$
- Certain Markov chains have a stationary distribution π such that $\pi = T\pi$
- Gibbs sampler is such a Markov chain with $T_i((X_{-i}, x_i') \mid (X_{-i}, x_i)) = p(x_i' \mid X_{-i})$, and stationary distribution $p(x_Q \mid X_E)$
- But it takes time for the chain to reach stationary distribution (mix)
  - Can be difficult to assert mixing
  - In practice "burn in": discard $X^{(0)}, \ldots, X^{(T)}$
  - Use all of $X^{(T+1)}, \ldots$ for inference (they are correlated); Do not thin

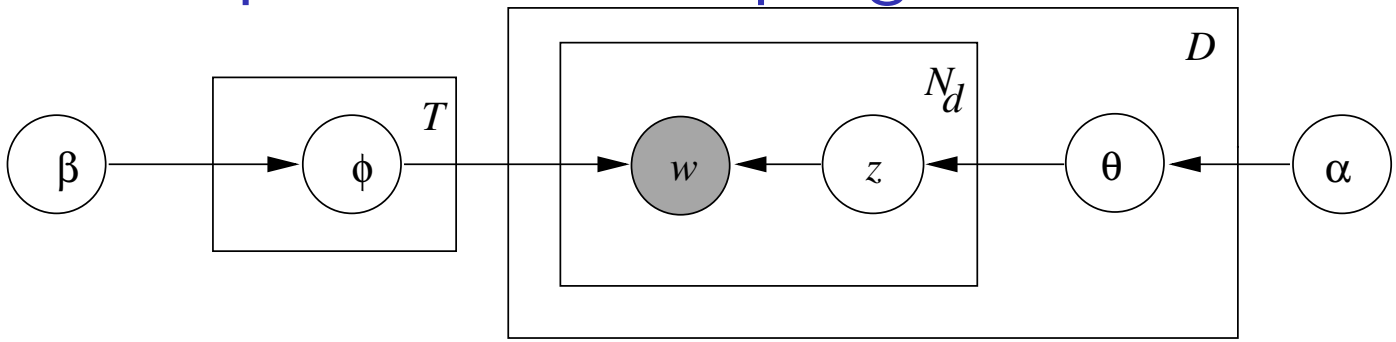
Collapsed Gibbs Sampling

- In general, $\mathbb{E}_p[f(X)] \approx \frac{1}{m}\sum_{i=1}^m f(X^{(i)})$ if $X^{(i)} \sim p$
- Sometimes $X = (Y, Z)$ where Z has closed-form operations
- If so,

$$\begin{aligned}\mathbb{E}_p[f(X)] &= \mathbb{E}_{p(Y)}\mathbb{E}_{p(Z|Y)}[f(Y,Z)] \\ &\approx \frac{1}{m}\sum_{i=1}^m \mathbb{E}_{p(Z|Y^{(i)})}[f(Y^{(i)},Z)]\end{aligned}$$

  if $Y^{(i)} \sim p(Y)$
- No need to sample Z: it is collapsed
- Collapsed Gibbs sampler $T_i((Y_{-i}, y_i') \mid (Y_{-i}, y_i)) = p(y_i' \mid Y_{-i})$
- Note $p(y_i' \mid Y_{-i}) = \int p(y_i', Z \mid Y_{-i})dZ$

Example: Collapsed Gibbs Sampling for LDA



Collapse θ, ϕ, Gibbs update:

$$P(z_i = j \mid \mathbf{z}_{-i}, \mathbf{w}) \propto \frac{n_{-i,j}^{(w_i)} + \beta n_{-i,j}^{(d_i)} + \alpha}{n_{-i,j}^{(\cdot)} + W\beta n_{-i,\cdot}^{(d_i)} + T\alpha}$$

- $n_{-i,j}^{(w_i)}$: number of times word w_i has been assigned to topic j, excluding the current position
- $n_{-i,j}^{(d_i)}$: number of times a word from document d_i has been assigned to topic j, excluding the current position
- $n_{-i,j}^{(\cdot)}$: number of times any word has been assigned to topic j, excluding the current position
- $n_{-i,\cdot}^{(d_i)}$: length of document d_i, excluding the current position

Summary: Markov Chain Monte Carlo

- Forward sampling
- Gibbs sampling
- Collapsed Gibbs sampling
- Not covered: block Gibbs, Metropolis-Hastings, etc.
- Unbiased (after burn-in), but can have high variance

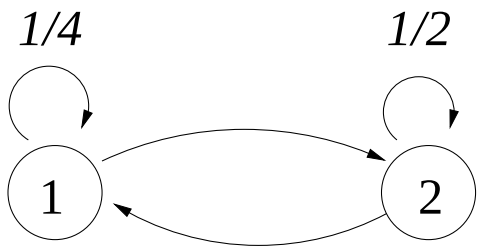
Outline

The Sum-Product Algorithm

- Also known as belief propagation (BP)
- Exact if the graph is a tree; otherwise known as "loopy BP", approximate
- The algorithm involves passing *messages* on the factor graph
- Alternative view: variational approximation (more later)

Example: A Simple HMM

- The Hidden Markov Model template (not a graphical model)



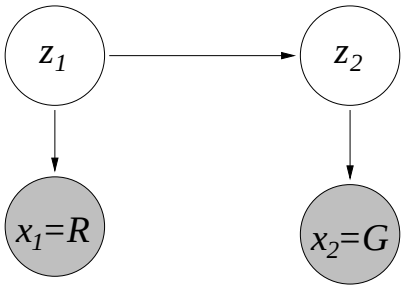
$P(x \mid z=1)=(1/2, 1/4, 1/4)$ $P(x \mid z=2)=(1/4, 1/2, 1/4)$

R G B R G B

$\pi_1 = \pi_2 = 1/2$

Example: A Simple HMM

- Observing $x_1 = R, x_2 = G$, the directed graphical model



- Factor graph

$P(z_1)P(x_1|\, z_1)$ $\quad$ $P(z_2|\, z_1)P(x_2|\, z_2)$

Messages

- A message is a vector of length K, where K is the number of values x takes.
- There are two types of messages:
  1. $\mu_{f \to x}$: message from a factor node f to a variable node x
     $\mu_{f \to x}(i)$ is the ith element, $i = 1 \dots K$.
  2. $\mu_{x \to f}$: message from a variable node x to a factor node f

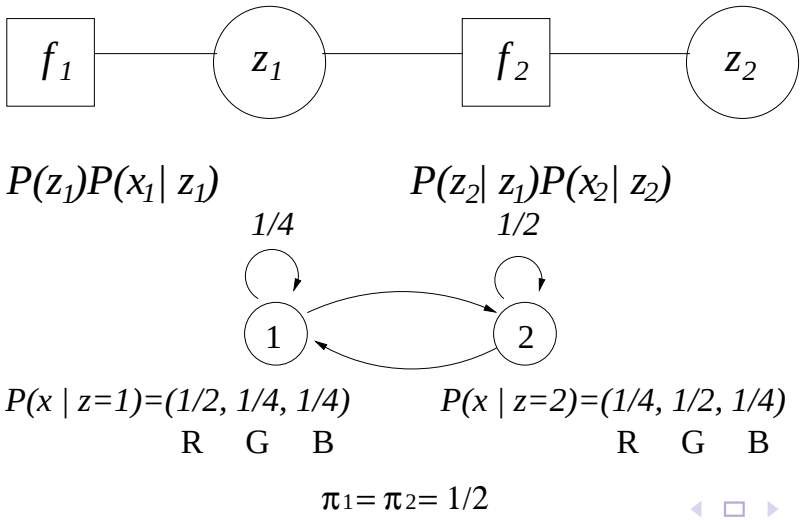
Leaf Messages

- Assume tree factor graph. Pick an arbitrary root, say z_2
- Start messages at leaves.
- If a leaf is a factor node f, $\mu_{f \to x}(x) = f(x)$

$$\mu_{f_1 \to z_1}(z_1 = 1) = P(z_1 = 1)P(R|z_1 = 1) = 1/2 \cdot 1/2 = 1/4$$

$$\mu_{f_1 \to z_1}(z_1 = 2) = P(z_1 = 2)P(R|z_1 = 2) = 1/2 \cdot 1/4 = 1/8$$

- If a leaf is a variable node x, $\mu_{x \to f}(x) = 1$



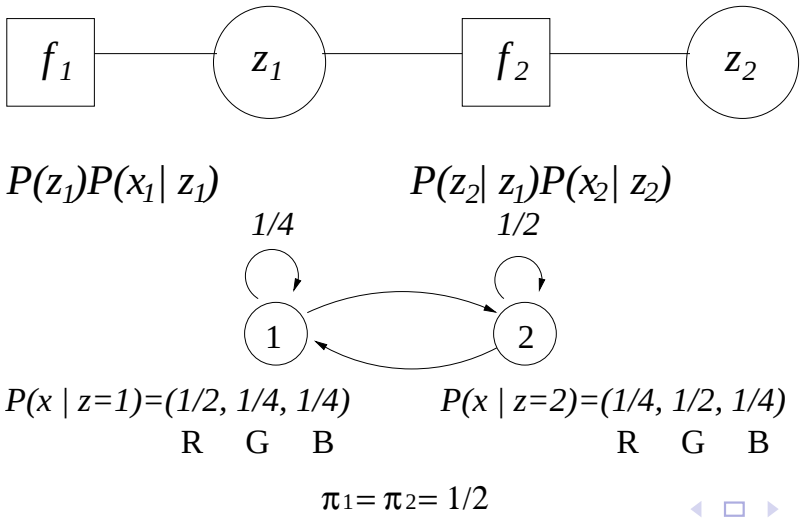
Message from Variable to Factor

- A node (factor or variable) can send out a message if all other incoming messages have arrived
- Let x be in factor f_s. $ne(x) \backslash f_s$ are factors connected to x excluding f_s.

$$\mu_{x \to f_s}(x) = \prod_{f \in ne(x) \backslash f_s} \mu_{f \to x}(x)$$

$$\mu_{z_1 \to f_2}(z_1 = 1) = 1/4$$

$$\mu_{z_1 \to f_2}(z_1 = 2) = 1/8$$



Message from Factor to Variable

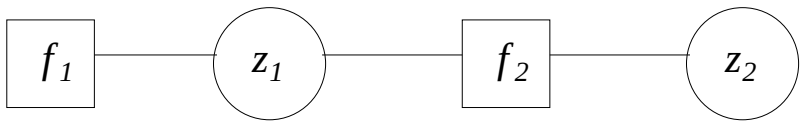
- Let x be in factor f_s. Let the other variables in f_s be $x_{1:M}$.

$$\mu_{f_s \to x}(x) = \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m=1}^{M} \mu_{x_m \to f_s}(x_m)$$

- In this example

$$\begin{aligned} \mu_{f_2 \to z_2}(s) &= \sum_{s'=1}^{2} \mu_{z_1 \to f_2}(s') f_2(z_1 = s', z_2 = s) \\ &= 1/4 P(z_2 = s | z_1 = 1) P(x_2 = G | z_2 = s) \\ &\quad + 1/8 P(z_2 = s | z_1 = 2) P(x_2 = G | z_2 = s) \end{aligned}$$

- We get $\mu_{f_2 \to z_2}(z_2 = 1) = 1/32$, $\mu_{f_2 \to z_2}(z_2 = 2) = 1/8$



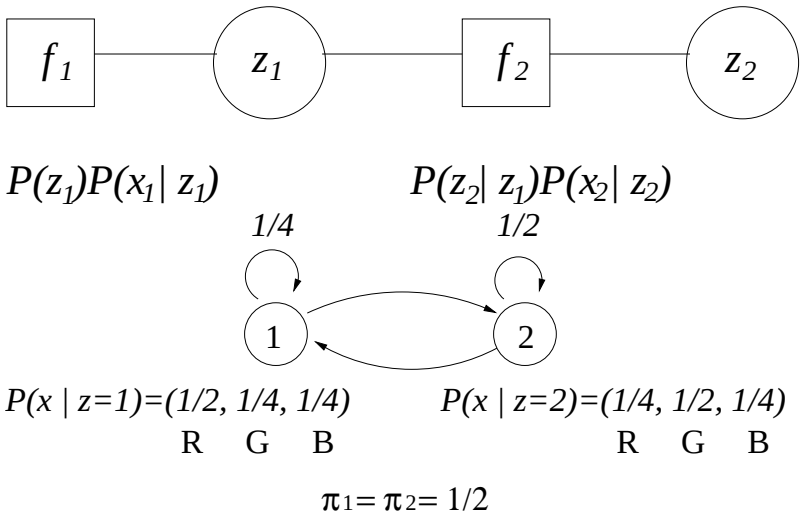
$P(z_1)P(x_1|\, z_1)$ $\quad$ $P(z_2|\, z_1)P(x_2|\, z_2)$

Up to Root, Back Down

- The message has reached the root, pass it back down

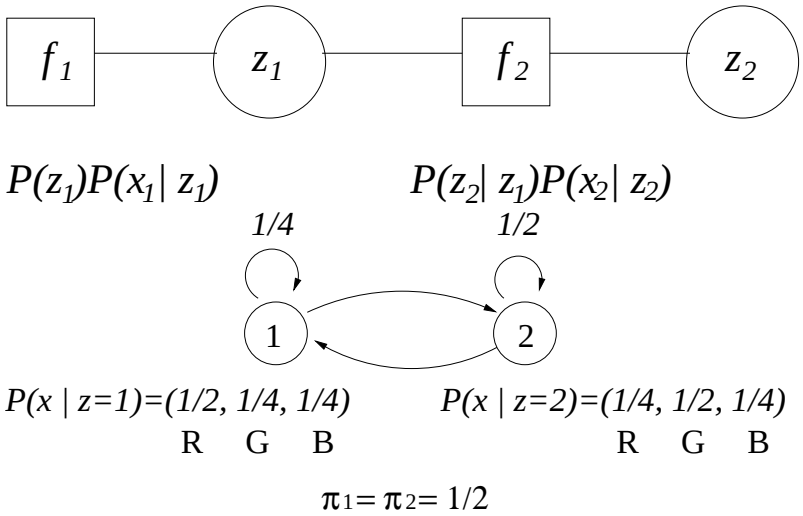
$$\mu_{z_2 \to f_2}(z_2 = 1) = 1$$

$$\mu_{z_2 \to f_2}(z_2 = 2) = 1$$



Keep Passing Down

- $\mu_{f_2 \to z_1}(s) = \sum_{s'=1}^{2} \mu_{z_2 \to f_2}(s') f_2(z_1 = s, z_2 = s')$
  $= 1P(z_2 = 1|z_1 = s)P(x_2 = G|z_2 = 1)$
  $+ 1P(z_2 = 2|z_1 = s)P(x_2 = G|z_2 = 2).$
- We get
  $\mu_{f_2 \to z_1}(z_1 = 1) = 7/16$
  $\mu_{f_2 \to z_1}(z_1 = 2) = 3/8$



From Messages to Marginals

- Once a variable receives all incoming messages, we compute its marginal as

$$p(x) \propto \prod_{f \in ne(x)} \mu_{f \to x}(x)$$

- In this example

$$P(z_1|x_1, x_2) \propto \mu_{f_1 \to z_1} \cdot \mu_{f_2 \to z_1} = \begin{pmatrix} 1/4 \\ 1/8 \end{pmatrix} \cdot \begin{pmatrix} 7/16 \\ 3/8 \end{pmatrix} = \begin{pmatrix} 7/64 \\ 3/64 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.7 \\ 0.3 \end{pmatrix}$$

$$P(z_2|x_1, x_2) \propto \mu_{f_2 \to z_2} = \begin{pmatrix} 1/32 \\ 1/8 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix}$$

- One can also compute the marginal of the *set of* variables x_s involved in a factor f_s

$$p(x_s) \propto f_s(x_s) \prod_{x \in ne(f)} \mu_{x \to f}(x)$$

Handling Evidence

- Observing $x = v$,
  - we can absorb it in the factor (as we did); or
  - set messages $\mu_{x \to f}(x) = 0$ for all $x \neq v$
- Observing X_E,
  - multiplying the incoming messages to $x \notin X_E$ gives the *joint* (not $p(x|X_E)$):

$$p(x, X_E) \propto \prod_{f \in ne(x)} \mu_{f \to x}(x)$$

  - The conditional is easily obtained by normalization

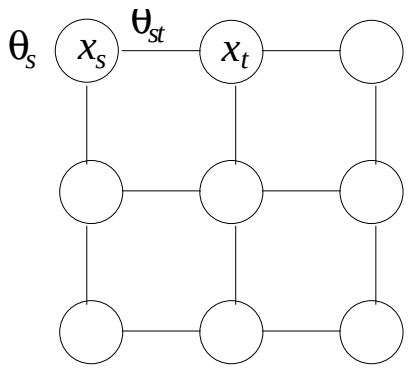
$$p(x|X_E) = \frac{p(x, X_E)}{\sum_{x'} p(x', X_E)}$$

Loopy Belief Propagation

- So far, we assumed a tree graph
- When the factor graph contains loops, pass messages indefinitely until convergence
- But convergence may not happen
- But in many cases loopy BP still works well, empirically

Outline

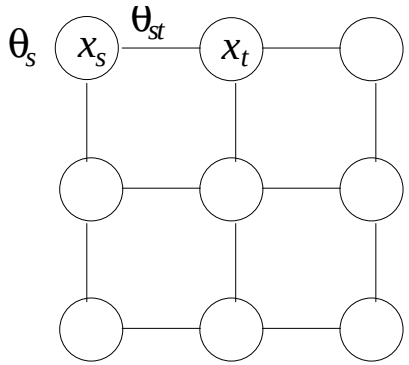
Example: The Ising Model



The random variables x take values in $\{0, 1\}$.

$$p_\theta(x) = \frac{1}{Z} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t\right)$$

The Conditional



- Markovian: the conditional distribution for x_s is

$$p(x_s \mid x_{-s}) = p(x_s \mid x_{N(s)})$$

$N(s)$ is the neighbors of s.

- This reduces to (recall Gibbs sampling)

$$p(x_s = 1 \mid x_{N(s)}) = \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} x_t)) + 1}$$

The Mean Field Algorithm for Ising Model

- Gibbs sampling would draw x_s from

$$p(x_s = 1 \mid x_{N(s)}) = \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} x_t)) + 1}$$

- Instead, let μ_s be the estimated marginal $p(x_s = 1)$
- Mean field algorithm:

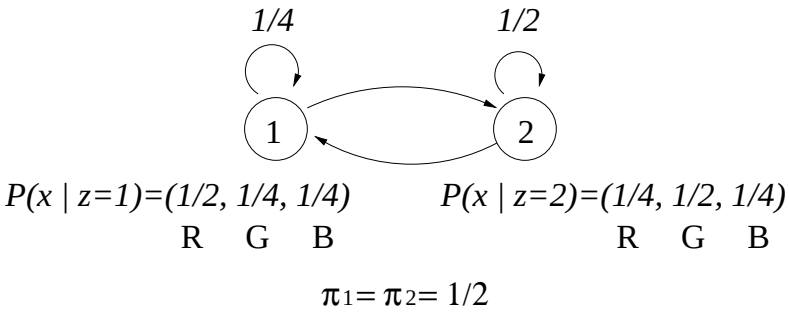
$$\mu_s \leftarrow \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} \mu_t)) + 1}$$

- The μ's are updated iteratively
- The Mean Field algorithm is coordinate ascent and guaranteed to converge to a local optimal (more later).

Outline

Maximizing Problems

Recall the HMM example



There are two senses of "best states" $z_{1:N}$ given $x_{1:N}$:

1. So far we computed the marginal $p(z_n|x_{1:N})$
   - We can define "best" as $z_n^* = \arg\max_k p(z_n = k|x_{1:N})$
   - However $z_{1:N}^*$ as a whole may not be the best
   - In fact $z_{1:N}^*$ can even have zero probability!
2. An alternative is to find

$$z_{1:N}^* = \arg\max_{z_{1:N}} p(z_{1:N}|x_{1:N})$$

   - finds the most likely *state configuration* as a whole
   - The max-sum algorithm solves this, generalizes the Viterbi algorithm for HMMs

Intermediate: The Max-Product Algorithm

Simple modification to the sum-product algorithm: replace $\sum$ with $\max$ in the factor-to-variable messages.

$$\begin{aligned}
\mu_{f_s \to x}(x) &= \max_{x_1} \ldots \max_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m=1}^{M} \mu_{x_m \to f_s}(x_m) \\
\mu_{x_m \to f_s}(x_m) &= \prod_{f \in ne(x_m) \setminus f_s} \mu_{f \to x_m}(x_m) \\
\mu_{x_{\text{leaf}} \to f}(x) &= 1 \\
\mu_{f_{\text{leaf}} \to x}(x) &= f(x)
\end{aligned}$$

Intermediate: The Max-Product Algorithm

- As in sum-product, pick an arbitrary variable node x as the root
- Pass messages up from leaves until they reach the root
- Unlike sum-product, do not pass messages back from root to leaves
- At the root, multiply incoming messages

$$p^{\max} = \max_x \left(\prod_{f \in ne(x)} \mu_{f \to x}(x) \right)$$

- This is the probability of the most likely state configuration

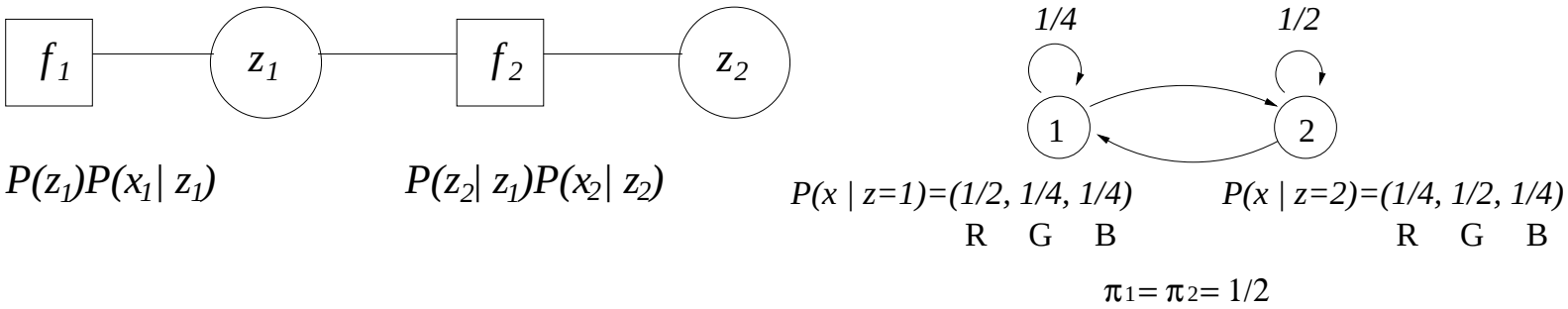
Intermediate: The Max-Product Algorithm

- To identify the configuration itself, keep *back pointers*:
- When creating the message

$$\mu_{f_s \to x}(x) = \max_{x_1} \dots \max_{x_M} f_s(x, x_1, \dots, x_M) \prod_{m=1}^{M} \mu_{x_m \to f_s}(x_m)$$

  for each x value, we separately create M pointers back to the values of $x_1, \dots, x_M$ that achieve the maximum.
- At the root, backtrack the pointers.

Intermediate: The Max-Product Algorithm



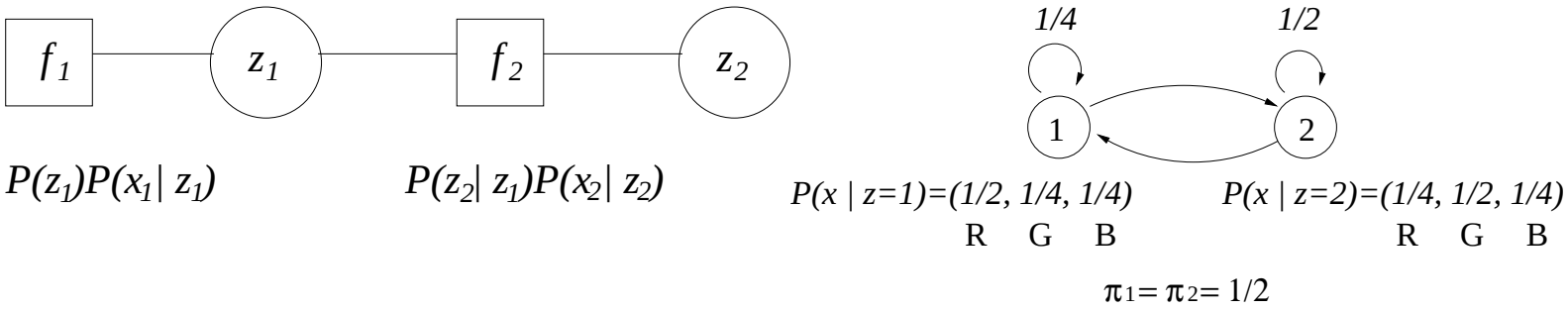
- Message from leaf f_1

$$\mu_{f_1 \to z_1}(z_1 = 1) = P(z_1 = 1)P(R|z_1 = 1) = 1/2 \cdot 1/2 = 1/4$$
$$\mu_{f_1 \to z_1}(z_1 = 2) = P(z_1 = 2)P(R|z_1 = 2) = 1/2 \cdot 1/4 = 1/8$$

- The second message

$$\mu_{z_1 \to f_2}(z_1 = 1) = 1/4$$
$$\mu_{z_1 \to f_2}(z_1 = 2) = 1/8$$

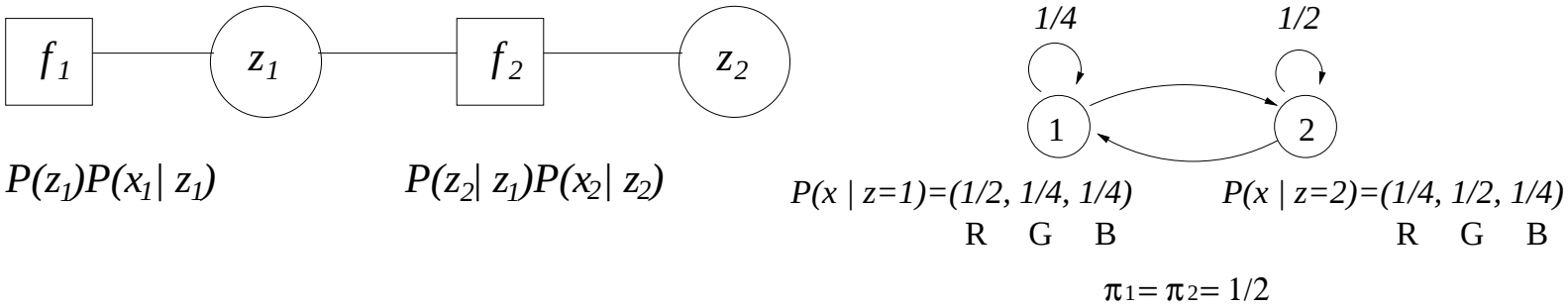
Intermediate: The Max-Product Algorithm



$$
\begin{aligned}
& \mu_{f_2 \to z_2}(z_2 = 1) \\
= & \max_{z_1} f_2(z_1, z_2) \mu_{z_1 \to f_2}(z_1) \\
= & \max_{z_1} P(z_2 = 1 \mid z_1) P(x_2 = G \mid z_2 = 1) \mu_{z_1 \to f_2}(z_1) \\
= & \max(1/4 \cdot 1/4 \cdot 1/4, 1/2 \cdot 1/4 \cdot 1/8) = 1/64
\end{aligned}
$$

Back pointer for $z_2 = 1$: either $z_1 = 1$ or $z_1 = 2$

Intermediate: The Max-Product Algorithm

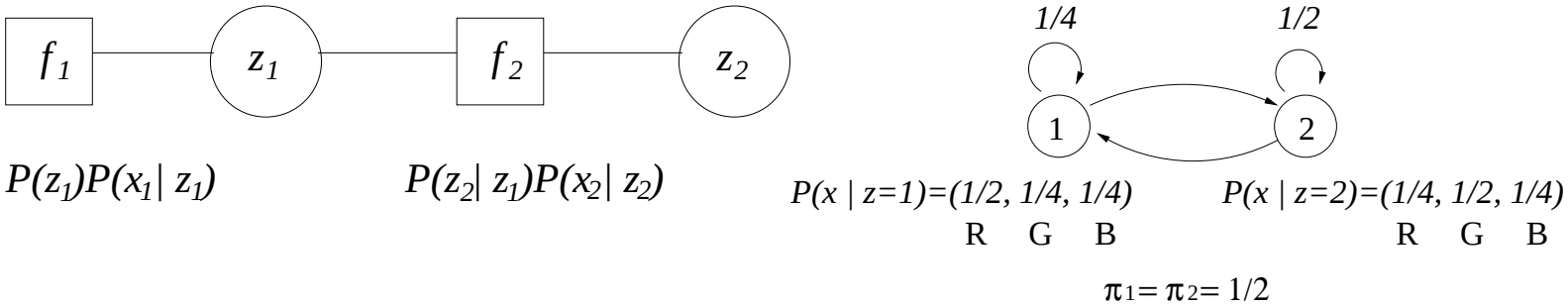


The other element of the same message:

$$
\begin{aligned}
& \mu_{f_2 \to z_2}(z_2 = 2) \\
= & \max_{z_1} f_2(z_1, z_2) \mu_{z_1 \to f_2}(z_1) \\
= & \max_{z_1} P(z_2 = 2 \mid z_1) P(x_2 = G \mid z_2 = 2) \mu_{z_1 \to f_2}(z_1) \\
= & \max(3/4 \cdot 1/2 \cdot 1/4, 1/2 \cdot 1/2 \cdot 1/8) = 3/32
\end{aligned}
$$

Back pointer for $z_2 = 2$: $z_1 = 1$

Intermediate: The Max-Product Algorithm



$$\mu_{f_2 \to z_2} = \begin{pmatrix} 1/64 & \to z_1 = 1,2 \\ 3/32 & \to z_1 = 1 \end{pmatrix}$$

At root z_2,

$$\max_{s=1,2} \mu_{f_2 \to z_2}(s) = 3/32$$

$$z_2 = 2 \to z_1 = 1$$

$$z^*_{1:2} = \arg\max_{z_{1:2}} p(z_{1:2}|x_{1:2}) = (1, 2)$$

In this example, sum-product and max-product produce the same best sequence; In general they differ.

From Max-Product to Max-Sum

The *max-sum algorithm* is equivalent to the max-product algorithm, but work in $\log$ space to avoid underflow.

$$\begin{aligned}
\mu_{f_s \to x}(x) &= \max_{x_1 \ldots x_M} \log f_s(x, x_1, \ldots, x_M) + \sum_{m=1}^{M} \mu_{x_m \to f_s}(x_m) \\
\mu_{x_m \to f_s}(x_m) &= \sum_{f \in ne(x_m) \setminus f_s} \mu_{f \to x_m}(x_m) \\
\mu_{x_{\text{leaf}} \to f}(x) &= 0 \\
\mu_{f_{\text{leaf}} \to x}(x) &= \log f(x)
\end{aligned}$$

When at the root,

$$\log p^{\max} = \max_x \left(\sum_{f \in ne(x)} \mu_{f \to x}(x) \right)$$

The back pointers are the same.

Outline

Stochastic Process

- Infinite collection of random variables indexed by a set $\{\mathbf{x}\}$.
- $\mathbf{x} \in \mathbb{R}$ for "time"
- More generally, $\mathbf{x} \in \mathbb{R}^d$ (e.g., space and time).

Outline

Base Distribution

- Let H be a *base distribution* over a probability space Θ.
- Example: $\Theta = \mathbb{R}^d$.
- An element $\theta \in \mathbb{R}^d$ is an index to the stochastic process
- $H = N(0, \Sigma)$ is a base distribution over Θ, but not a stochastic process.
- $H(\theta) = N(\theta; 0, \Sigma)$ is *not* a random variable (it is a fixed value for a given θ)

Stick-Breaking Construction of Dirichlet Process

$$
\begin{aligned}
\beta_k &\sim \text{Beta}(1, \alpha) \\
\pi_k &= \beta_k \prod_{i=1}^{k-1} (1 - \beta_i) \\
\theta_k^* &\sim H \\
G &= \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k^*}
\end{aligned}
$$

- δ_z is the point mass function on z
- $\pi_1, \pi_2, \ldots$ are stick fragments which tend to (but not always) get smaller. Sum to 1.
- Each fragment is associated with an index θ_k^* sampled from the base distribution H
- G is a sample from a Dirichlet Process $G \sim DP(\alpha, H)$

Properties of G

- G is a probability measure on Θ (naturally normalized), similar to the base distribution H.
- With probability one, G is a discrete measure (true even if H is a continuous measure, e.g. Gaussian).
- θ's drawn from G have *repeats*. Useful to model clusters.

More Properties of Dirichlet Process

$G \sim DP(\alpha, H)$

- Marginals of G are Dirichlet-distributed: Let $A_1, \ldots, A_r$ be any finite measurable partition of Θ, then

$$(G(A_1), \ldots, G(A_r)) \sim \text{Dirichlet}(\alpha H(A_1), \ldots, \alpha H(A_r))$$

- For any measurable $A \subseteq \Theta$,

$$\mathbb{E}[G(A)] = H(A) \quad \mathbb{V}[G(A)] = \frac{H(A)(1 - H(A))}{1 + \alpha}$$

- As $\alpha \to \infty$, $G(A) \to H(A)$ for any measurable A.

The Posterior of G

- Let $G \sim DP(\alpha, H)$ the prior.
- Suppose we observe $\theta_1, \ldots, \theta_n \sim G$.
- The posterior distribution of G given $\theta_1, \ldots, \theta_n$ is another DP:

$$G \mid \theta_1, \ldots, \theta_n \sim DP\left(\alpha + n, \frac{\alpha}{\alpha + n} H + \frac{1}{\alpha + n} \sum_{i=1}^{n} \delta_{\theta_i}\right)$$

- The predictive distribution of θ_{n+1} is

$$\theta_{n+1} \sim \frac{\alpha}{\alpha + n} H + \frac{1}{\alpha + n} \sum_{i=1}^{n} \delta_{\theta_i}$$

- There is a chance that $\theta_{n+1} = \theta_i$ for some $i \leq n$ (i.e. repeating).

The Blackwell-MacQueen Urn Scheme

- Assume samples from H do not repeat (e.g. Gaussian)
- Let $\theta_1^* \dots \theta_m^*$ be the *unique* values in $\theta_1 \dots \theta_n$
- Let $n_k = \sum_{i=1}^{n} 1_{\theta_i = \theta_k^*}$ for $k = 1 \dots m$.
- θ_{n+1} is generated with the following procedure:
  1. With probability $\alpha/(\alpha + n)$, draw a new value from H and assign it to θ_{n+1};
  2. Otherwise, reuse value θ_k^* with probability n_k/n.
  3. We add θ_{n+1} to the samples, and repeat this process.

The Chinese Restaurant Process

- The equality relationship in $\theta_1 \ldots \theta_n$ defines a *partition* of n items.
- The first customer sits at the first table.
- With probability $\alpha/(\alpha + n)$ the $(n+1)$-th customer sits at a new table; otherwise he joins an existing table with probability proportional to the number of people already sitting there.
- *Chinese Restaurant Process* (CRP) defines a distribution over partitions of items.
- CRP + (for a new table draw a dish $\theta \sim H$; all customers sitting on this table eat the dish) = DP

Dirichlet Process Mixture Models (DPMMs)

- Infinite mixture models: unlimited number of clusters

$$\begin{aligned} G &\sim DP(\alpha, H) \\ \theta_i &\sim G \\ \mathbf{x}_i &\sim F(\theta) \end{aligned}$$

  where $F(\theta)$ is an appropriate distribution parametrized by θ (e.g. multinomial).
- Each observation $\mathbf{x}_i$ has its own parameter θ_i.
- Many of the θ_i's are identical, naturally inducing a clustering structure over $\mathbf{x}$.
- Given $\mathbf{x}_1 \ldots \mathbf{x}_n, \alpha, H, F$, use MCMC to infer $\theta_1 \ldots \theta_n$

References

- Bishop, *Pattern Recognition and Machine Learning*. Springer 2006.
- Hastie, Tibshirani, Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Second Edition, 2009.
- Koller & Friedman, *Probabilistic Graphical Models*. MIT 2009.
- Murphy, *Machine Learning: a Probabilistic Perspective*, 2012.
- Wasserman, *All of Statistics: A Concise Course in Statistical Inference*. Springer 2003.