Group Meeting - August 28, 2020

Paper review & Research progress

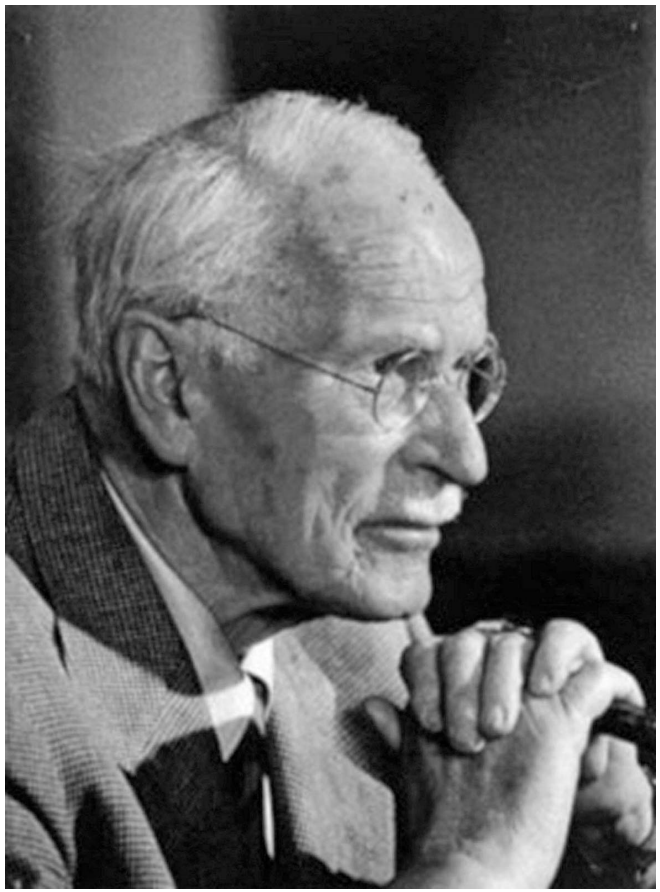
Truong Son Hy *

*Department of Computer Science
The University of Chicago

Ryerson Physical Lab

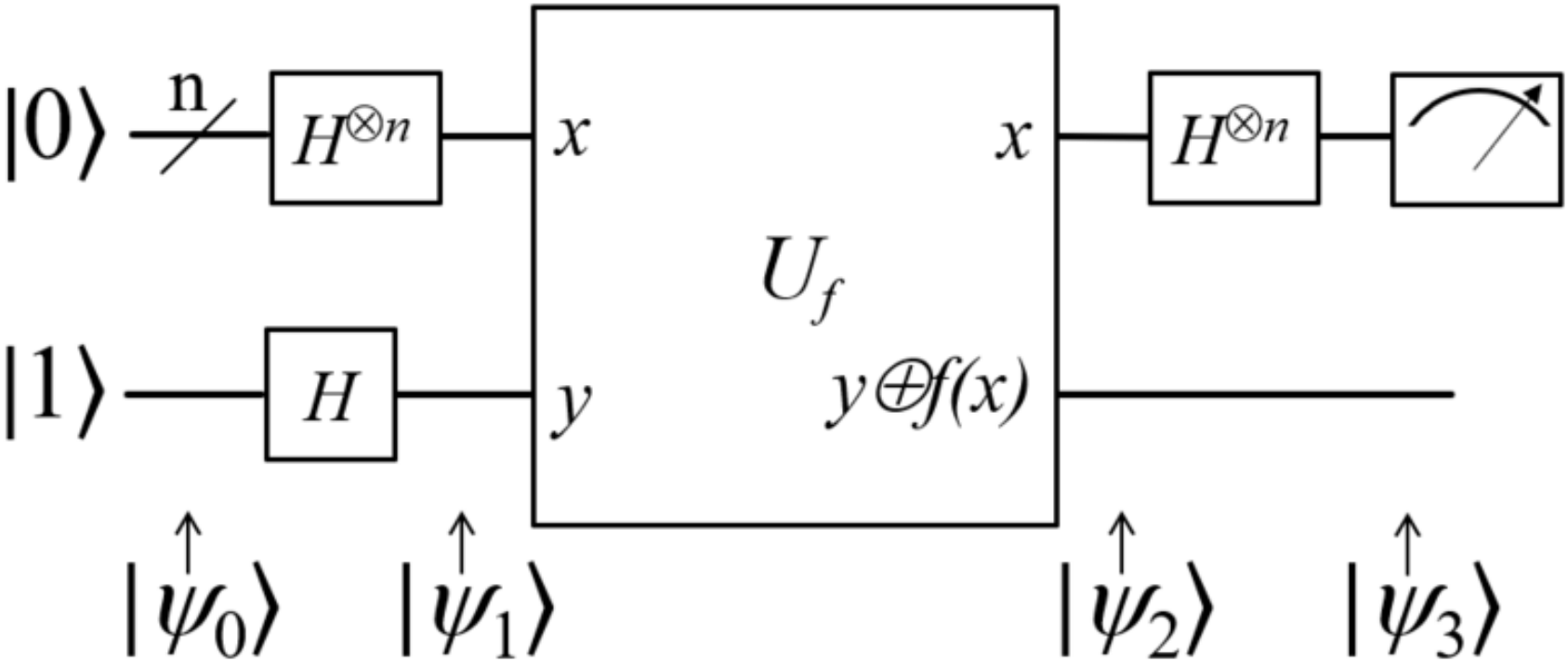
Carl Jung

As far as we can discern, the sole purpose of human existence is to kindle **a light of meaning** in the darkness of mere being.



Alan Turing

We can only see a short distance ahead, but we can see plenty there that needs to be done.



Source code

Source code:

`https://github.com/HyTruongSon/Dirichlet_Process`

1. Gibbs sampling on 2D Ising model: `ising_gibbs.py`
2. Stick-breaking process: `stick_breaking.py`
3. Gibbs sampling for fitting Dirichlet Process Gaussian Mixture Model (DPGMM): `gibbs_dirichlet_process.py`
4. Collapsed Gibbs sampling for fitting DPGMM: `collapsed_gibbs_dirichlet_process.py`

Paper 1

Machine Learning: A Probabilistic Perspective, Kevin P. Murphy

1. Chapter 24. Markov chain Monte Carlo (MCMC) inference
2. Chapter 25. Clustering
3. Chapter 27. Latent variable models for discrete data

I highly recommend this textbook!

Paper 2

Latent dirichlet allocation, David M. Blei, Andrew Y. Ng, Michael I. Jordan (2003) – 33,218 citations!

Some useful video tutorials:

1. (David Blei) `https://www.youtube.com/watch?v=FkckgwMHP2s`
2. `https://www.youtube.com/watch?v=T05t-SqKArY`
3. `https://www.youtube.com/watch?v=BaM1uiCpj_E`

Background – Gamma function (1)

The gamma function Γ is a commonly used extension of the factorial to complex numbers. The gamma function is defined for all complex numbers except non-positive integers. For any positive integer n:

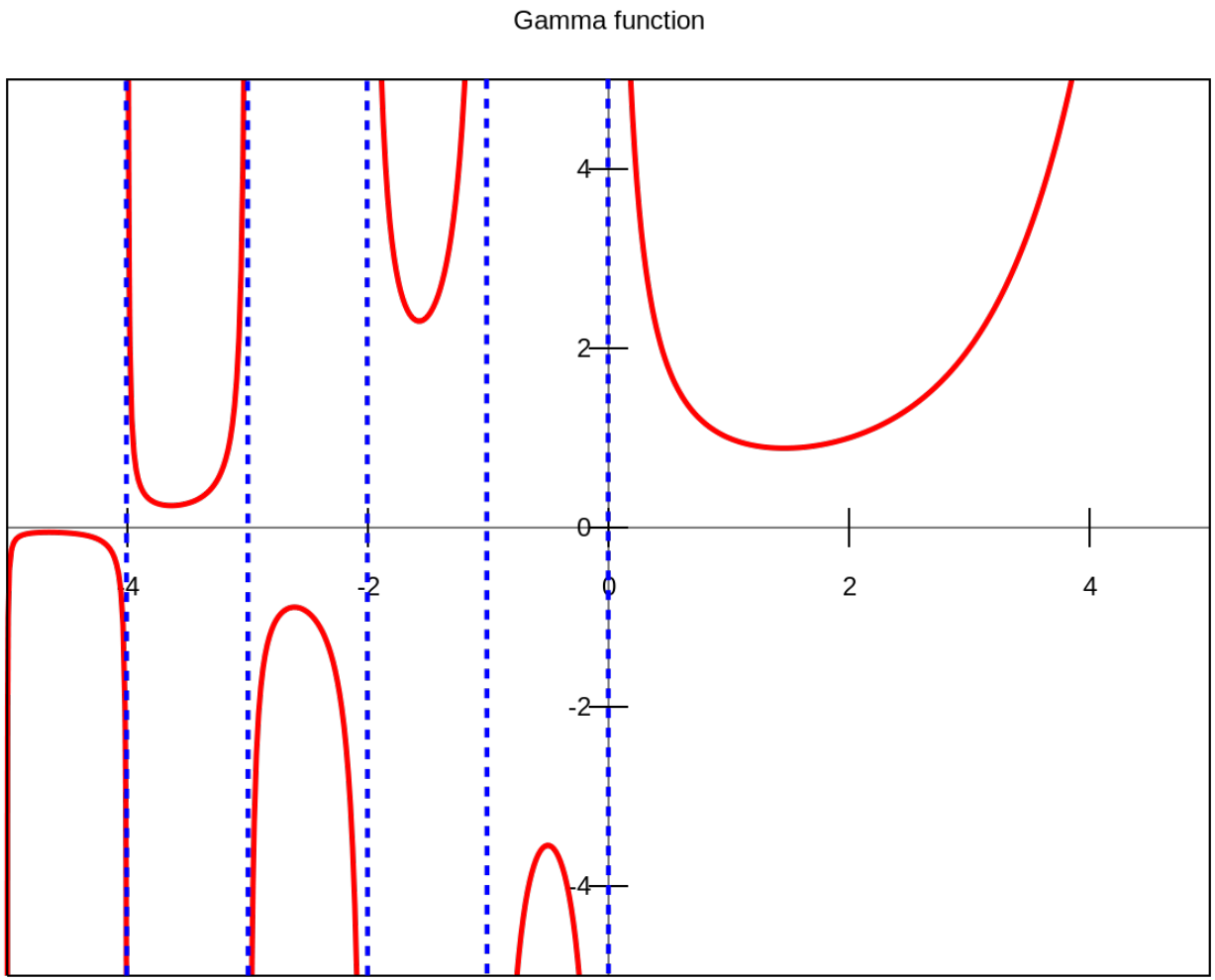
$$\Gamma(n) = (n-1)!$$

For complex numbers z with a positive real part $\Re(z) > 0$, the gamma function is defined as:

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx$$

Exercise: Prove that $\int_0^\infty x^{z-1} e^{-x} dx = (z-1)!$ for $\forall z \in \mathrm{N}_+$ (by induction).

Background – Gamma function (2)



Background – Beta distribution (1)

The **Beta distribution** is a family of continuous probability distributions defined on the interval $[0, 1]$ parameterized by two positive shape parameters α and β, that appear as exponents of the random variable and control the shape of the distribution. The generalization to multiple variables is called a **Dirichlet distribution**.

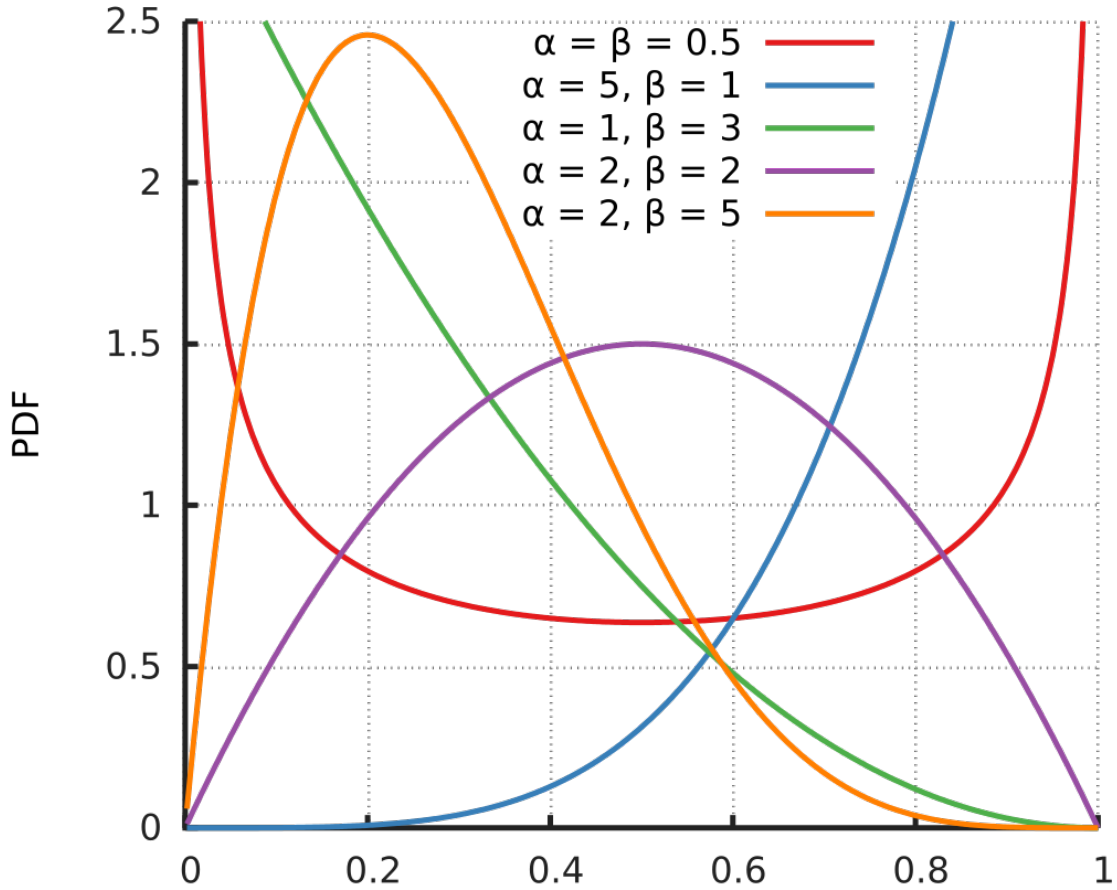
The probability density function (PDF) of Beta(α, β) for $0 \leq x \leq 1$, and shape parameters α, β:

$$f(x; \alpha, \beta) = \frac{x^{\alpha-1} \cdot (1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1} du} = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \cdot x^{\alpha-1} \cdot (1-x)^{\beta-1}$$

$$= \frac{1}{B(\alpha, \beta)} \cdot x^{\alpha-1} \cdot (1-x)^{\beta-1}$$

where Beta function $B(\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}$.

Background – Beta distribution (2)



$$X \sim \text{Beta}(\alpha, \beta): \qquad \mathrm{E}[X] = \frac{\alpha}{\alpha+\beta}, \qquad \text{Mode}(X) = \frac{\alpha-1}{\alpha+\beta-2}$$

Background – Inverse-Wishart distribution (1)

The **inverse Wishart distribution** (denoted by IW) is a probability distribution defined on real-valued positive-definite matrices. The PDF of the inverse Wishart is:

$$f(\boldsymbol{x}; \boldsymbol{S}, \nu) = \frac{|\boldsymbol{S}|^{\nu/2}}{2^{\nu p/2}\Gamma_p(\nu/2)}|\boldsymbol{x}|^{-(\nu+p+1)/2}e^{-\frac{1}{2}\mathrm{tr}(\boldsymbol{S}\boldsymbol{x}^{-1})}$$

where $\boldsymbol{x}$ and $\boldsymbol{S}$ are $p \times p$ positive definite matrices, and $\Gamma_p(.)$ is the **multivariate gamma function** that can be defined as integrating over the $p \times p$ positive-definite real matrices:

$$\Gamma_p(a) = \int_{\boldsymbol{S} \succ 0} \exp(-\mathrm{tr}(\boldsymbol{S})) \cdot |\boldsymbol{S}|^{a-(p+1)/2} d\boldsymbol{S}$$

or

$$\Gamma_p(a) = \pi^{p(p-1)/4} \prod_{j=1}^{p} \Gamma_1(a + (1-j)/2)$$

where $\Gamma_1(a)$ is the ordinary gamma function.

Background – Inverse Wishart vs. Wishart

We say $\boldsymbol{X}$ follows an **inverse** Wishart distribution, $\boldsymbol{X} \sim IW(\boldsymbol{S}, \nu)$ if its inverse $\boldsymbol{X}^{-1}$ has a Wishart distribution $W(\boldsymbol{S}^{-1}, \nu)$. Suppose $\boldsymbol{G}$ is a $p \times \nu$ matrix, each column independently drawn from a p-variate normal distribution with zero mean:

$$G_i = (G_{1,i}, .., G_{p,i})^T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{S}^{-1})$$

Then the Wishart distribution is the probability distribution of the $p \times p$ random matrix (called scatter matrix):

$$\boldsymbol{Y} = GG^T = \sum_{i=1}^{n} G_i G_i^T$$

One can write $\boldsymbol{Y} \sim W(\boldsymbol{S}^{-1}, \nu) \leftarrow$ I use the precision matrix instead!

Background – Dirichlet distribution (1)

The **Dirichlet distribution** is often denoted as Dir(α), a family of continuous multivariate probability distributions parameterized by a vector $\boldsymbol{\alpha}$ of positive reals. It is a multivariate generalization of the **Beta distribution**. The infinite-dimensional generalization of the **Dirichlet distribution** is the **Dirichlet Process**.

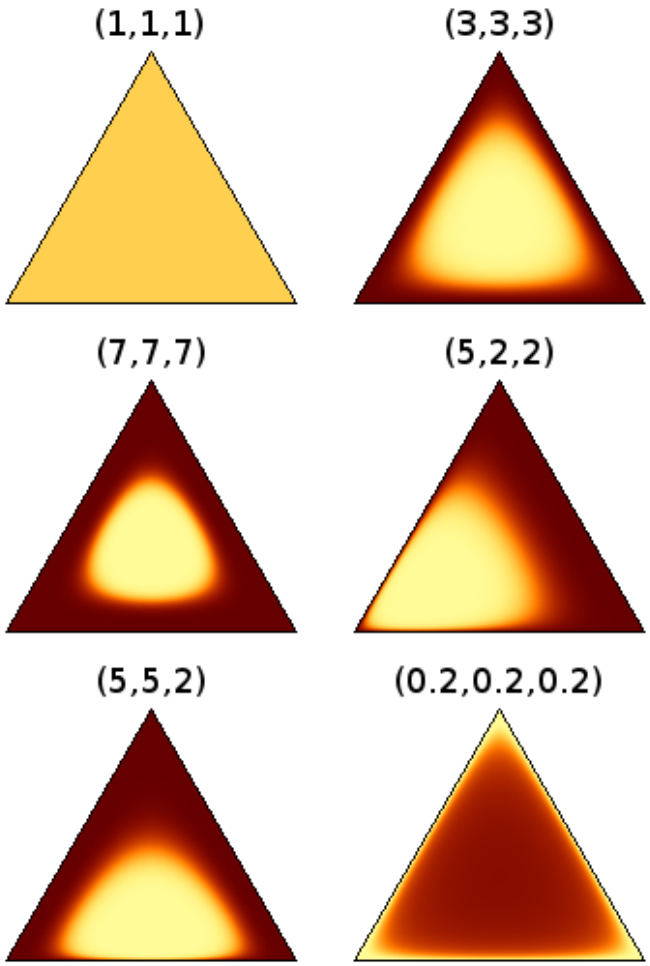
Parameters $\alpha_1, .., \alpha_K > 0$. PDF:

$$f(x_1, .., x_K; \alpha_1, .., \alpha_K) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$

where $\{x_k\}_{k=1}^{K}$ belongs to the standard $K-1$ simplex, in other words:

$$\sum_{i=1}^{K} x_i = 1, \qquad x_i \geq 0 \qquad \forall i \in [1, K]$$

Background – Dirichlet distribution (2)



Normalizing constant – Beta function:

$$B(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{K} \alpha_i)}$$

Mean:

$$\mathrm{E}[X] = \frac{\alpha_i}{\sum_{k=1}^{K} \alpha_k}$$

Mode:

$$x_i = \frac{\alpha_i - 1}{\sum_{k=1}^{K} \alpha_k - K}$$

Background – Gibbs sampling (1)

Gibbs sampling

To fit a Dirichlet Process (DP) mixture model, we need to learn Gibbs sampling.

Background – Gibbs sampling (1)

Gibbs sampling

To fit a Dirichlet Process (DP) mixture model, we need to learn Gibbs sampling.

The basic idea behind Gibbs sampling is that we sample each variable in turn, conditioned on the values of all the other variables in the distribution. Given a joint sample $\boldsymbol{x}^s$ of all the variables, we generate a new sample $\boldsymbol{x}^{s+1}$ by sampling each component in turn, based on the most recent values of the other variables.

Background – Gibbs sampling (1)

Gibbs sampling

To fit a Dirichlet Process (DP) mixture model, we need to learn Gibbs sampling.

The basic idea behind Gibbs sampling is that we sample each variable in turn, conditioned on the values of all the other variables in the distribution. Given a joint sample $\boldsymbol{x}^s$ of all the variables, we generate a new sample $\boldsymbol{x}^{s+1}$ by sampling each component in turn, based on the most recent values of the other variables.

Gibbs sampling for graphical models

Gibbs sampling can be used for **Permutation Synchronization** as I mentioned last week.

Background – Gibbs sampling (2)

For example, we apply Gibbs sampling for pairwise Markov Random Field (MRF) taking the form:

$$p(x_t|x_{-t}, \boldsymbol{\theta}) \propto \prod_{s \in \mathcal{N}(t)} \psi_{st}(x_s, x_t)$$

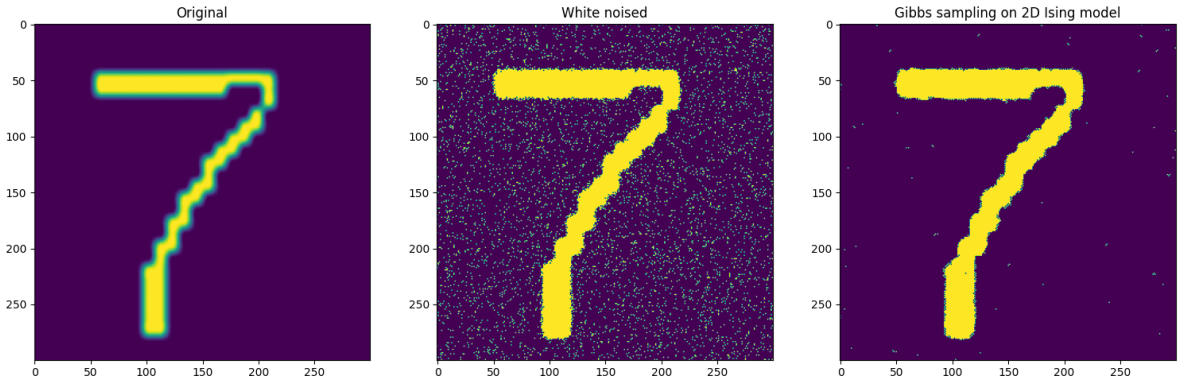
In the case of an **Ising model** with edge potentials $\psi(x_s, x_t) = \exp(J x_s x_t)$, where $x_t \in \{-1, +1\}$, the full conditional becomes:

$$p(x_t = +1|x_{-t}, \boldsymbol{\theta}) = \frac{\prod_{s \in \mathcal{N}(t)} \psi_{st}(x_s, x_t = +1)}{\prod_{s \in \mathcal{N}(t)} \psi_{st}(x_s, x_t = +1) + \prod_{s \in \mathcal{N}(t)} \psi_{st}(x_s, x_t = -1)}$$

$$= \frac{\exp(J \sum_{s \in \mathcal{N}(t)} x_s)}{\exp(J \sum_{s \in \mathcal{N}(t)} x_s) + \exp(-J \sum_{s \in \mathcal{N}(t)} x_s)} = \frac{\exp(J\eta_t)}{\exp(J\eta_t) + \exp(-J\eta_t)}$$

Background – Gibbs sampling (3)

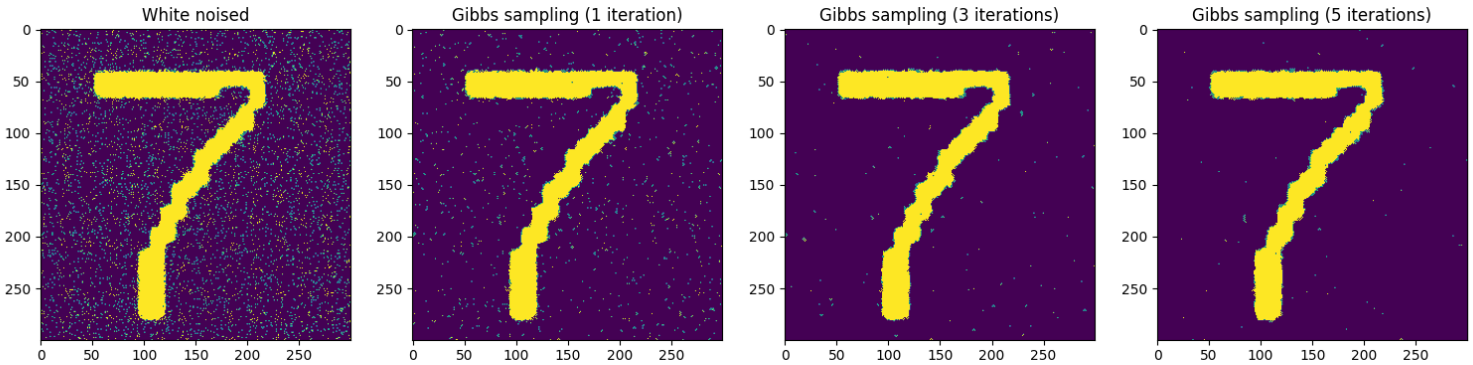
Thus $p(x_t = +1|x_{-t}, \boldsymbol{\theta}) = \text{sigmoid}(2J\eta_t)$ where $\eta_t = x_t(a_t - d_t)$, where a_t is the number of neighbors that agree with t (have the same sign), and d_t is the number of neighbors that disgree with t (opposite sign).



Code: `ising_gibbs.py`

Background – Gibbs sampling (4)

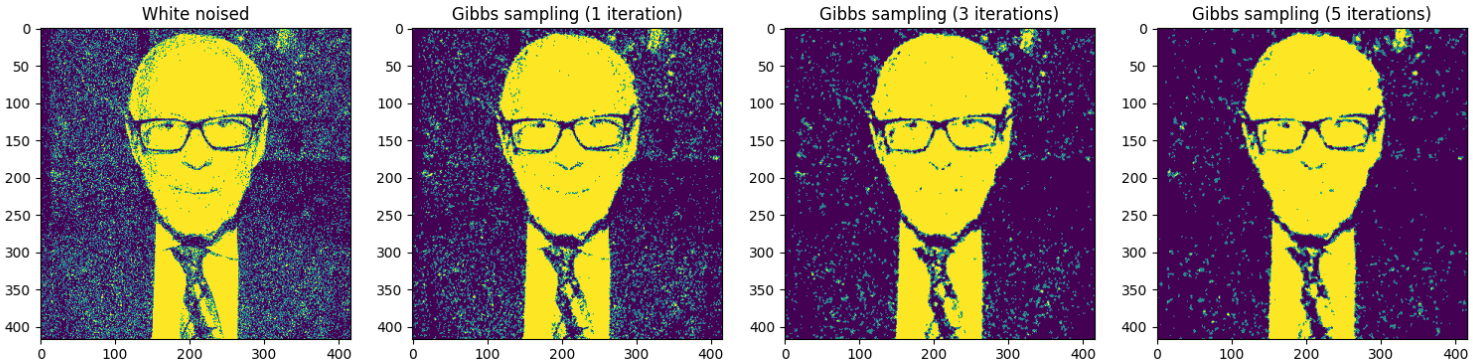
Gibbs sampling is an iterative algorithm:



Code: `ising_gibbs.py`

Background – Gibbs sampling (5)

For color images, we cannot assign each pixel values of $x_t \in \{-1, 1\}$ that is just black and white. We need to extend the 2D Ising model for $x_t \in \{0, .., 255\}$ and change the edge potential functions $\psi(x_s, x_t) = \exp(J|x_s - x_t|)$ with $J < 0$ (to punish the neighboring difference). This is **binarized** James Simions:



Background – Gibbs sampling (6)

Gibbs sampler (complexity $O(NKD)$) for fitting mixture model, given the number of clusters K. Later, we will need to extend this algorithm for Dirichlet Process with unknown number of clusters.

1. **Naive Gibbs**: As presented above.
2. **Collapsed Gibbs** (more efficient): Analytically integrate out some of the unknown quantities and just sample the rest. Suppose we sample $\boldsymbol{z}$ and integrate out $\boldsymbol{\theta}$. We can draw conditionally independent samples $\boldsymbol{\theta}^s \sim p(\boldsymbol{\theta}|\boldsymbol{z}^s, \mathcal{D})$ which has a lower variance than samples drawn from the joint state space, as the result of **Rao-Blackwell's theorem**.

Background – Collapsed Gibbs sampling (1)

Rao-Blackwell's theorem

Let $\boldsymbol{z}$ and $\boldsymbol{\theta}$ be dependent random variables, and $f(\boldsymbol{z}, \boldsymbol{\theta})$ be some scalar function. Then:

$$\text{var}_{\boldsymbol{z},\boldsymbol{\theta}}[f(\boldsymbol{z}, \boldsymbol{\theta})] \geq \text{var}_{\boldsymbol{z}}[\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{z}, \boldsymbol{\theta})|\boldsymbol{z}]]$$

Algorithm 24.1: Collapsed Gibbs sampler for a mixture model

for *each* $i = 1 : N$ *in random order* **do**
  Remove $\mathbf{x}_i$'s sufficient statistics from old cluster z_i ;
  for *each* $k = 1 : K$ **do**
    Compute $p_k(\mathbf{x}_i) \triangleq p(\mathbf{x}_i | \{\mathbf{x}_j : z_j = k, j \neq i\})$;
  Compute $p(z_i = k | \mathbf{z}_{-i}, \mathcal{D}) \propto (N_{k,-i} + \alpha/K) p_k(\mathbf{x}_i)$;
  Sample $z_i \sim p(z_i | \cdot)$;
  Add $\mathbf{x}_i$'s sufficient statistics to new cluster z_i

Background – Collapsed Gibbs sampling (2)

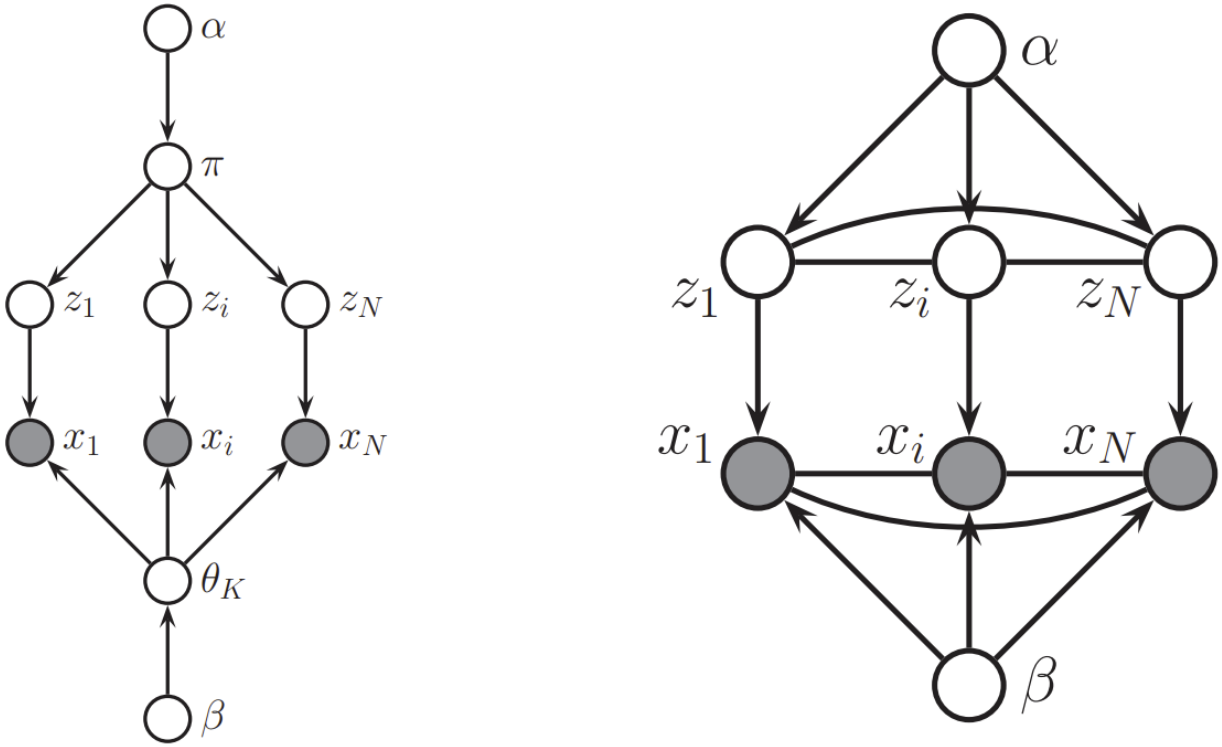


Figure 24.2 (a) A mixture model. (b) After integrating out the parameters.

Introduction

The simplest approach to clustering is to use a **finite mixture model**, sometimes called **model-based clustering**: we define a probabilistic model of the data, and optimize a well-defined objective (e.g. likelihood or posterior).

Introduction

The simplest approach to clustering is to use a **finite mixture model**, sometimes called **model-based clustering**: we define a probabilistic model of the data, and optimize a well-defined objective (e.g. likelihood or posterior).

Finite in which sense?

The principle problem with finite mixture model is how to choose the number of components K. It would be much better if we did not have to choose K at all.

Introduction

The simplest approach to clustering is to use a **finite mixture model**, sometimes called **model-based clustering**: we define a probabilistic model of the data, and optimize a well-defined objective (e.g. likelihood or posterior).

Finite in which sense?

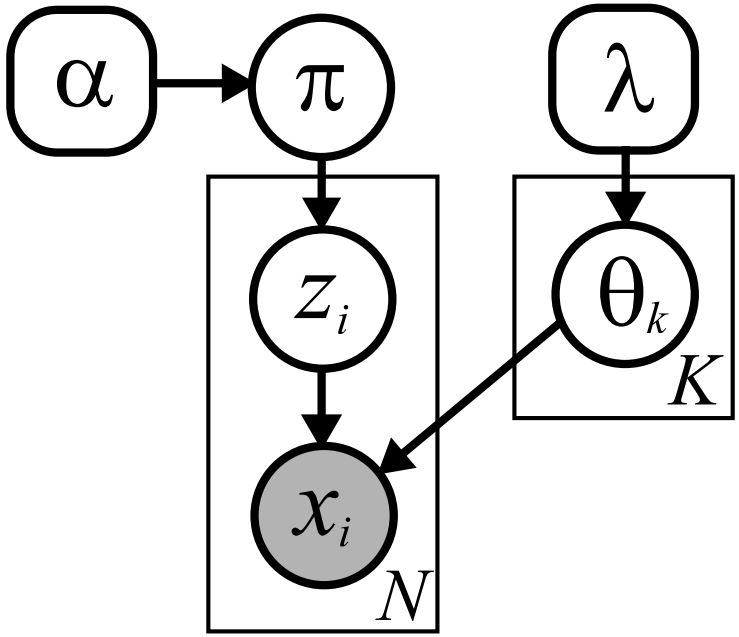
The principle problem with finite mixture model is how to choose the number of components K. It would be much better if we did not have to choose K at all.

Infinite mixture models

Infinite mixture models are proposed in which we do not impose any priori bound on K. We will discuss a **non-parametric prior** based on the **Dirichlet process** (DP): the number of clusters grows as the amount of data increases $\rightarrow$ useful for hierarchical clustering.

Finite mixture model (1)

The usual representation of a finite mixture model:

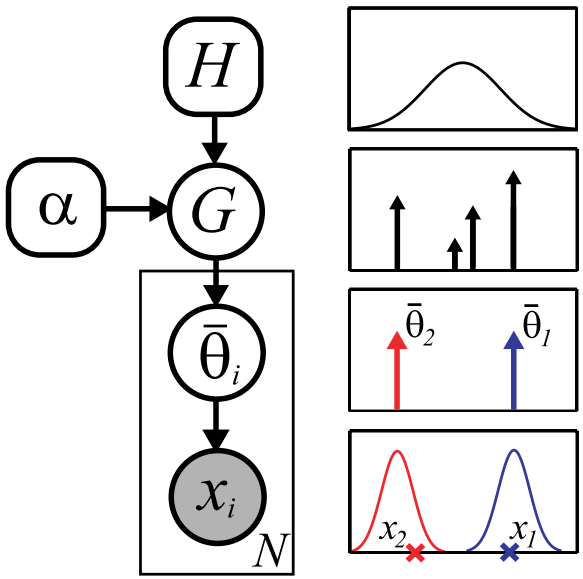


$$p(\mathbf{x}_i|z_i = k, \theta) = p(\mathbf{x_i}|\theta_k)$$

$$p(z_i = k|\pi) = \pi_k$$

$$p(\pi|\alpha) = \text{Dir}(\pi|(\alpha/K)\mathbf{1}_K)$$

Finite mixture model (2)

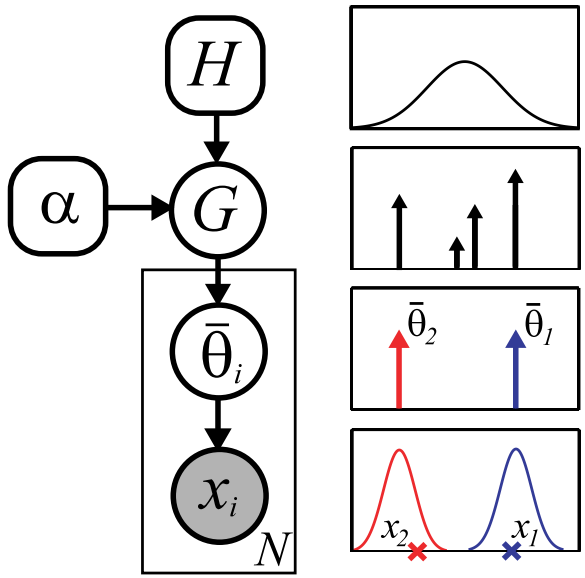


This is an equivalent representation. Here, θ_i is the parameter used to generate observation $\mathbf{x}_i$; these parameters are sampled from distribution G which has the form:

$$G(\theta) = \sum_{k=1}^{K} \pi_k \delta_{\theta_k}(\theta)$$

where $\pi \sim \text{Dir}(\frac{\alpha}{K}\mathbf{1})$ and $\theta_k \sim H(\lambda)$.

Finite mixture model (3)



G is a finite mixture of delta functions, centered on the cluster parameters θ_k. The probability that θ_i is equal to θ_k is exactly π_k (the prior probability for that cluster). Because the more data we generate, the more likely we should see a new cluster, we replace G with a **random probability measure**. One way to do so is **Dirichlet process**: $G \sim \mathrm{DP}(\alpha, H)$.

Dirichlet Process (1)

A **Dirichlet Process** is a distribution over probability measures $G : \Theta \to \mathrm{R}^+$, where we require $G(\theta) \geq 0$ and $\int_\Theta G(\theta) d\theta = 1$. The DP is defined implicitly by the requirement that $(G(T_1), .., G(T_K))$ has a joint Dirichlet distribution:

$$\mathrm{Dir}(\alpha H(T_1), .., \alpha H(T_K))$$

for any finite partition $(T_1, .., T_K)$ of Θ. If this is the case, we write $G \sim \mathrm{DP}(\alpha, H)$, where α is called the **concentration parameter** and H is called the **base measure**.

Dirichlet Process (2)

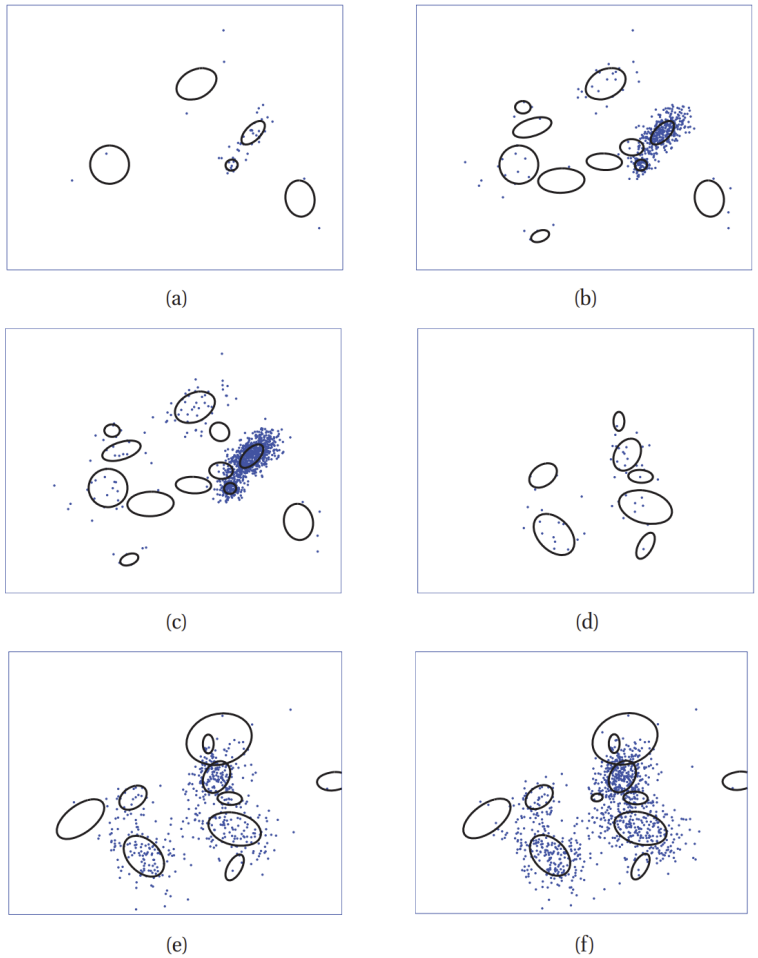


Figure 25.3 Some samples from a Dirichlet process mixture model of 2D Gaussians, with concentration parameter $\alpha = 1$. From left to right, we show $N = 50$, $N = 500$ and $N = 1000$ samples. Each row is a different run. We also show the model parameters as ellipses, which are sampled from a vague NIW base distribution. Based on Figure 2.25 of (Sudderth 2006). Figure generated by `dpmSampleDemo`, written by Yee-Whye Teh.

Dirichlet Process (3)

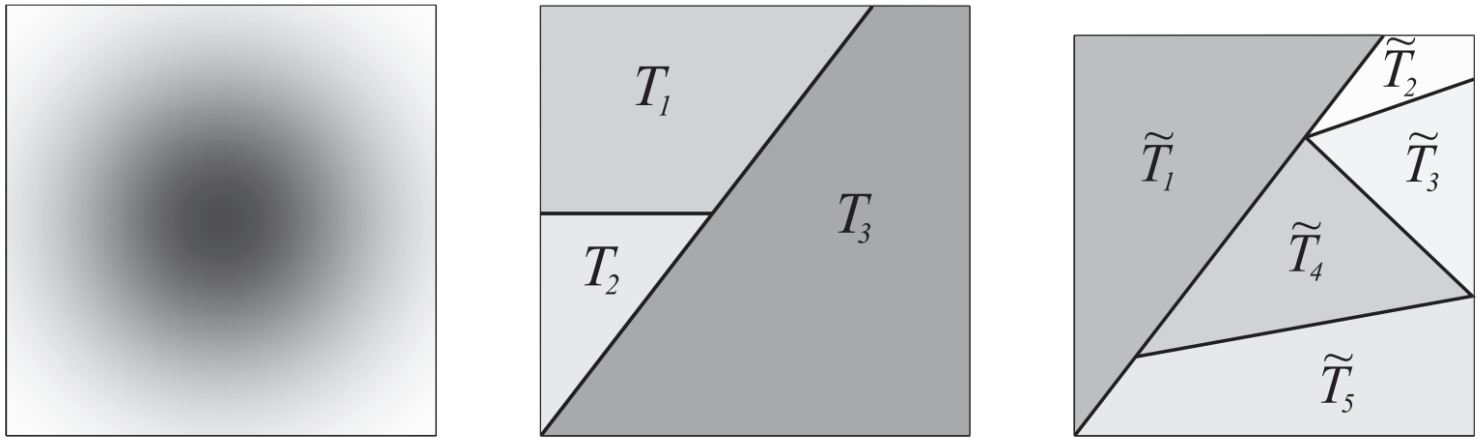


Figure 25.4 (a) A base measure H on a 2d space Θ. (b) One possible partition into $K = 3$ regions, where the shading of cell T_k is proportional to $\mathbb{E}[G(T_k)] = H(T_k)$. (c) A refined partition into $K = 5$ regions. Source: Figure 2.21 of (Sudderth 2006). Used with kind permission of Erik Sudderth.

Stick breaking construction of the DP (1)

Stick-breaking construction gives us a constructive definition for the DP. Let $\pi = \{\pi_k\}_{k=1}^{\infty}$ be an infinite sequence of mixing weights derived from the following process:

$$\beta_k \sim \text{Beta}(1, \alpha)$$

$$\pi_k = \beta_k \prod_{l=1}^{k-1}(1 - \beta_l) = \beta_k \left(1 - \sum_{l=1}^{k-1} \pi_l\right)$$

This is often denoted by $\pi \sim \text{GEM}(\alpha)$ where GEM stands for Griffiths, Engen and McCloskey.

1. **This process terminates with probability 1.**
2. The number of elements it generates increases with α.
3. The size of the π_k components decreases on average.

Stick breaking construction of the DP (2)

Define:

$$G(\boldsymbol{\theta}) = \sum_{k=1}^{\infty} \pi_k \delta_{\boldsymbol{\theta}_k}(\boldsymbol{\theta})$$

where $\pi \sim \text{GEM}(\alpha)$ and $\boldsymbol{\theta}_k \sim H$. One can show that $G \sim \text{DP}(\alpha, H)$. If the base measure H is Gaussian, then most data comes from the Gaussians with large π_k values, this suggests that DP might be useful for clustering.

Stick breaking construction of the DP (3)

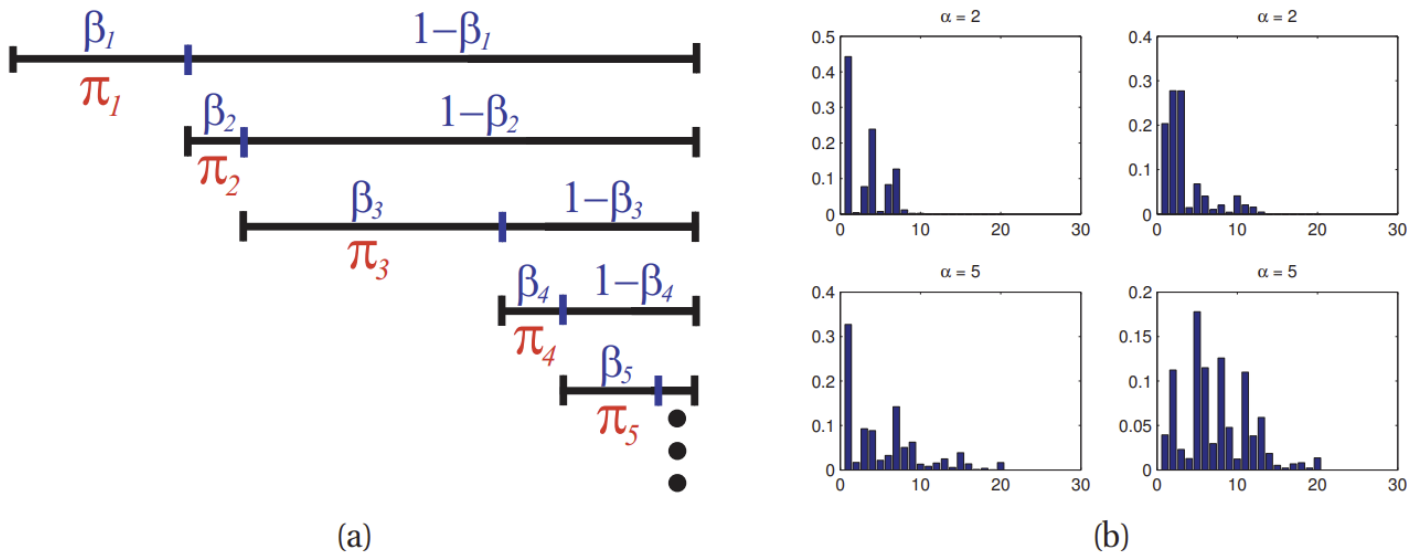
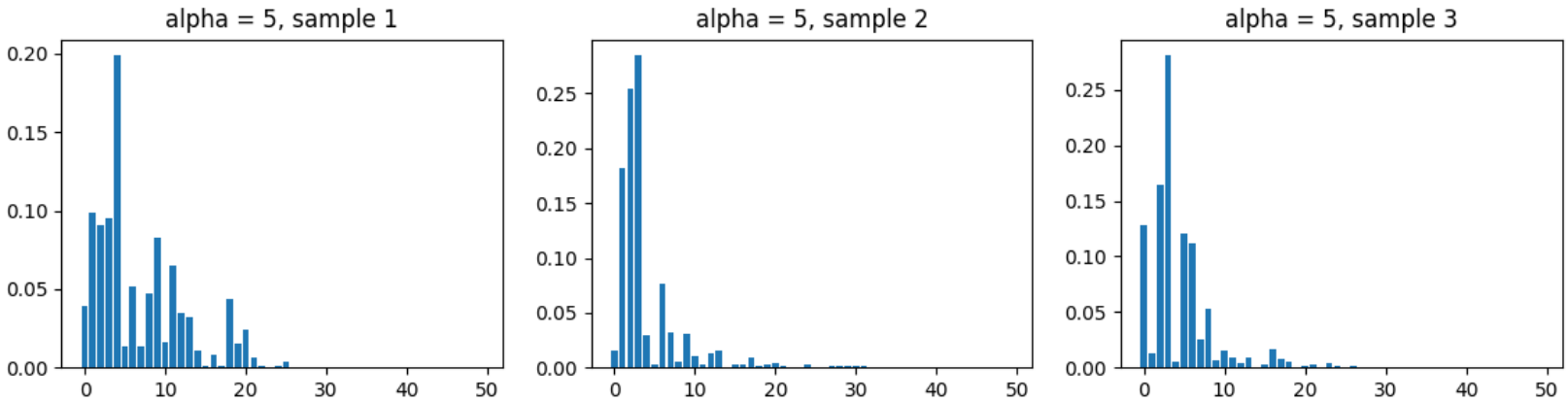


Figure 25.5 Illustration of the stick breaking construction. (a) We have a unit length stick, which we break at a random point β_1; the length of the piece we keep is called π_1; we then recursively break off pieces of the remaining stick, to generate $\pi_2, \pi_3, \ldots$. Source: Figure 2.22 of (Sudderth 2006). Used with kind permission of Erik Sudderth. (b) Samples of π_k from this process for $\alpha = 2$ (top row) and $\alpha = 5$ (bottom row). Figure generated by `stickBreakingDemo`, written by Yee-Whye Teh.

Stick breaking construction of the DP (4)



Code: `stick_breaking.py`

The Chinese Restaurant Process (CRP)

If $\bar{\boldsymbol{\theta}}_i \sim G$ are N observations from $G \sim \text{DP}(\alpha, H)$, taking on K distinct values $\boldsymbol{\theta}_k$, then the predictive distribution of the next observation is given by:

$$p(\bar{\boldsymbol{\theta}}_{N+1} = \boldsymbol{\theta} | \bar{\boldsymbol{\theta}}_{1:N}, \alpha, H) = \frac{1}{\alpha + N} \left(\alpha H(\boldsymbol{\theta}) + \sum_{k=1}^{K} N_k \delta_{\bar{\boldsymbol{\theta}}_k}(\boldsymbol{\theta}) \right)$$

where N_k is the number of previous observations equal to $\boldsymbol{\theta}_k$. It is more convenient to work with discrete variables z_i which specify which value of $\boldsymbol{\theta}_k$ to use, that is to define $\bar{\boldsymbol{\theta}}_i = \boldsymbol{\theta}_{z_i}$:

$$p(z_{N+1} = z | \boldsymbol{z}_{1:N}, \alpha) = \frac{1}{\alpha + N} \left(\alpha \mathbb{I}(z = k^*) + \sum_{k=1}^{K} N_k \mathbb{I}(z = k) \right)$$

where k^* represents a new cluster index that has not been yet used.

Applying Dirichlet processes to mixture modeling

Stochastic data generating mechanism:

$$\boldsymbol{\pi} \sim \text{GEM}(\alpha)$$

$$z_i \sim \boldsymbol{\pi}$$

$$\boldsymbol{\theta}_k \sim H(\boldsymbol{\lambda})$$

$$\boldsymbol{x}_i \sim F(\boldsymbol{\theta}_{z_i})$$

Too much abstract so far?

Let's get real!

Gibbs – Fitting a DP mixture model (1)

Stochastic data generating mechanism:

$$\pi \sim \text{GEM}(\alpha), \quad z_i \sim \pi, \quad \theta_k \sim H(\lambda), \quad x_i \sim F(\theta_{z_i})$$

Let's fit a Dirichlet Process **Gaussian** Mixture Model:

- $\theta_k = \{\mu_k, \Sigma_k\}$
- $\lambda = \{m_0, V_0, S_0, \nu_0\}$
- $p(\theta_k|\lambda) = p(\mu_k|m_0, V_0) \cdot p(\Sigma_k|S_0, \nu_0)$
- $p(\mu_k|m_0, V_0) = \mathcal{N}(\mu_k|m_0, V_0)$
- $p(\Sigma_k|S_0, \nu_0) = IW(\Sigma_k|S_0, \nu_0)$

Later, we will discuss 2-level Dirichlet Process Multinomial Mixture Model (that is also called LDA).

Gibbs – Fitting a DP mixture model (2)

Full joint distribution (finite):

$$p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) p(\boldsymbol{z}|\boldsymbol{\pi}) p(\boldsymbol{\pi}) \prod_{k=1}^{K} p(\boldsymbol{\mu}_k) p(\boldsymbol{\Sigma}_k)$$

$$= \left(\prod_{i=1}^{N} \prod_{k=1}^{K} (\pi_k \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))^{\mathbb{I}(z_i=k)} \right) \times$$

$$\text{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}) \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{\mu}_k|\boldsymbol{m}_0, \boldsymbol{V}_0) IW(\boldsymbol{\Sigma}_k|\boldsymbol{S}_0, \nu_0)$$

Gibbs – Fitting a DP mixture model (3)

Step 1. Sample for the mixing weights:

- Finite:

$$p(\boldsymbol{\pi}|\boldsymbol{z}) = \text{Dir}(\{\alpha_k + \sum_{i=1}^{N} \mathbb{I}(z_i = k)\}_{k=1}^{K})$$

- Infinite: Use the stick-breaking process $\boldsymbol{\pi} \sim \text{GEM}(\alpha)$ until $1 - \sum_{k=1}^{K} \pi_k < \epsilon$

Step 2. Sample for the discrete indicators:

$$p(z_i = k|\boldsymbol{x}_i, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) \propto \pi_k \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Gibbs – Fitting a DP mixture model (4)

Step 3. For the means:

$$p(\boldsymbol{\mu}_k|\boldsymbol{\Sigma}_k, \boldsymbol{z}, \boldsymbol{x}) = \mathcal{N}(\boldsymbol{\mu}_k|\boldsymbol{m}_k, \boldsymbol{V}_k)$$

$$\boldsymbol{V}_k^{-1} = \boldsymbol{V}_0^{-1} + N_k\boldsymbol{\Sigma}_k^{-1}$$

$$\boldsymbol{m}_k = \boldsymbol{V}_k(\boldsymbol{\Sigma}_k^{-1}N_k\bar{\boldsymbol{x}}_k + \boldsymbol{V}_0^{-1}\boldsymbol{m}_0)$$

$$N_k = \sum_{i=1}^{N}\mathbb{I}(z_i = k)$$

$$\bar{\boldsymbol{x}}_k = \frac{\sum_{i=1}^{N}\mathbb{I}(z_i = k)\boldsymbol{x}_i}{N_k}$$

Make Kalman Filter great again!

Markov chain of reasoning: Kalman Filter → Control theory → Probabilistic Robotics (suggestions: Sebastian Thrun's book) → SLAM → Permutation (group) synchronization → ...

Gibbs – Fitting a DP mixture model (5)

Step 4. For the covariances:

$$p(\Sigma_k|\boldsymbol{\mu}_k, \boldsymbol{z}, \boldsymbol{x}) = IW(\Sigma_k|\boldsymbol{S}_k, \nu_k)$$

$$\boldsymbol{S}_k = \boldsymbol{S}_0 + \sum_{i=1}^{N} \mathbb{I}(z_i = k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T, \qquad \nu_k = \nu_0 + N_k$$

Python doesn't have support for taking samples from the Inverse Wishart distribution, so I just take the mode of it, basically:

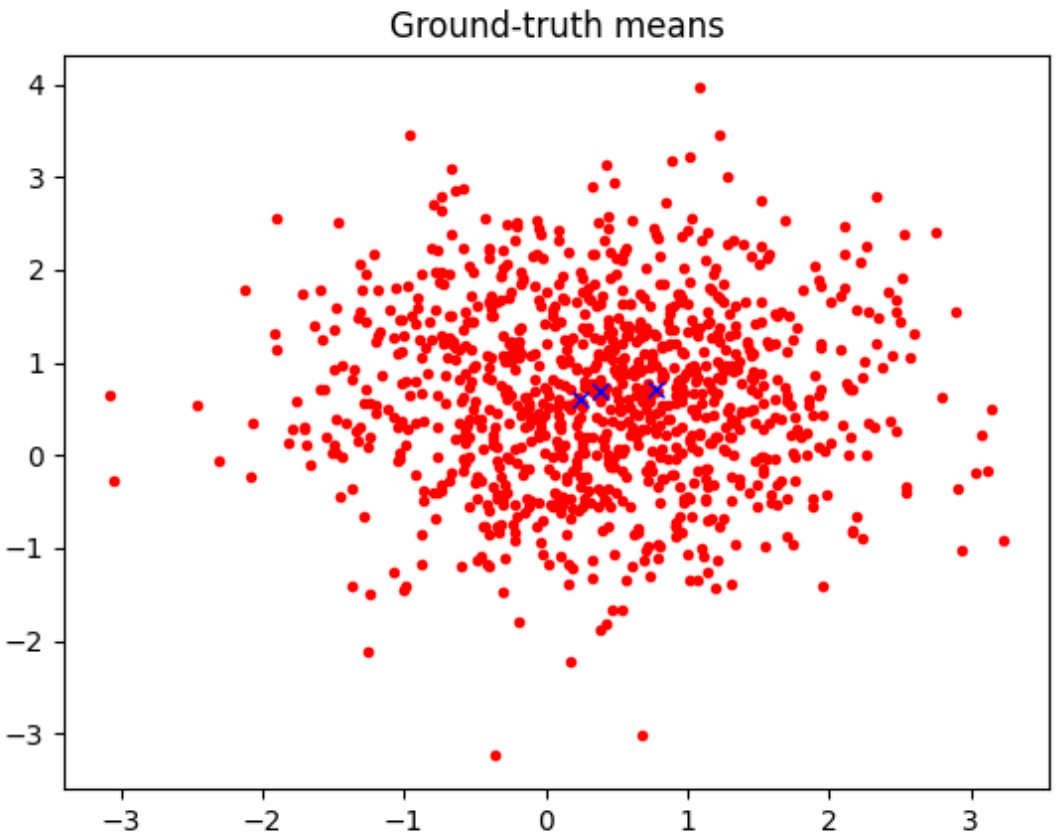
$$\Sigma_k \leftarrow \frac{\boldsymbol{S}_k}{\nu_k + d + 1}$$

Step 5. Shrink the components without any assignments ($\forall k : N_k = 0$). Repeat step 2.

Code: `gibbs_dirichlet_process.py`

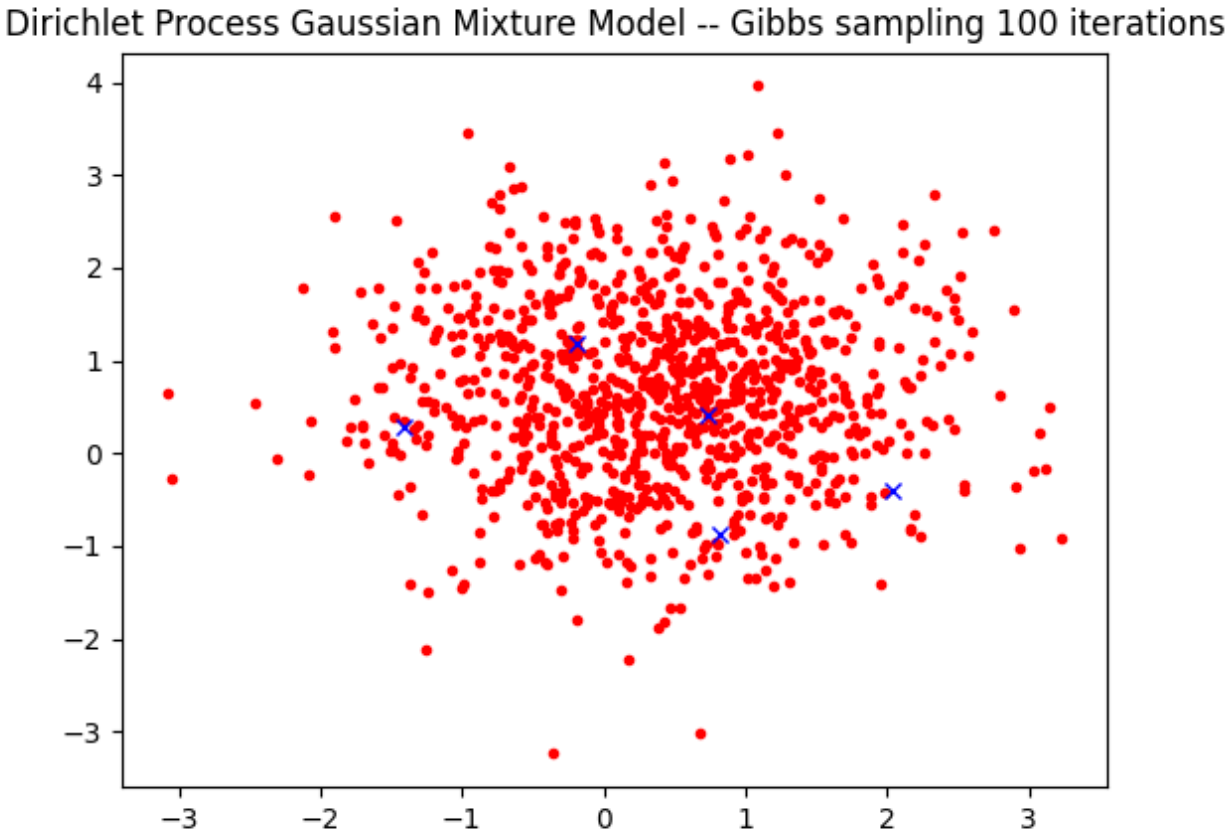
Gibbs – Fitting a DP mixture model (6)

Example 1: Generate data by a mixture of 3 Gaussians (**random** μ, **identity** Σ). Red dots are data. Blue crosses are the means.



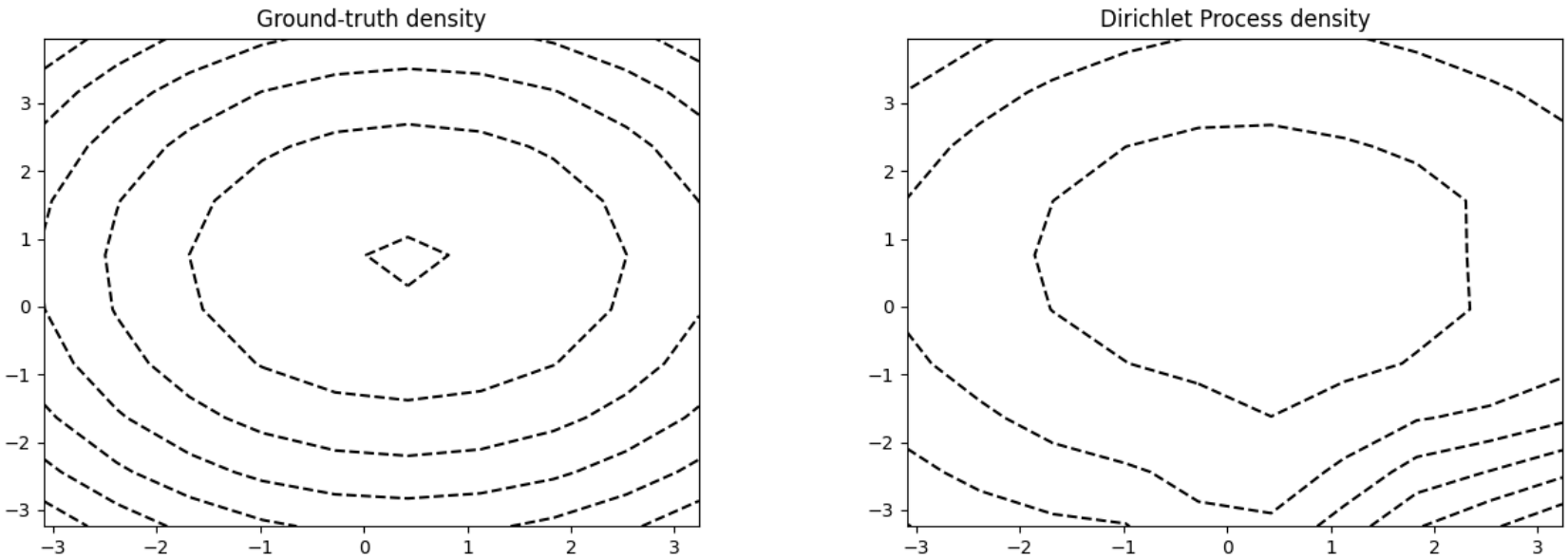
Gibbs – Fitting a DP mixture model (7)

Dirichlet Process Gaussian Mixture Model: Initial stick-breaking process with $\alpha = 1$ and stoping $\epsilon = 0.05$. Gibbs sampling runs 100 iterations.



Gibbs – Fitting a DP mixture model (8)

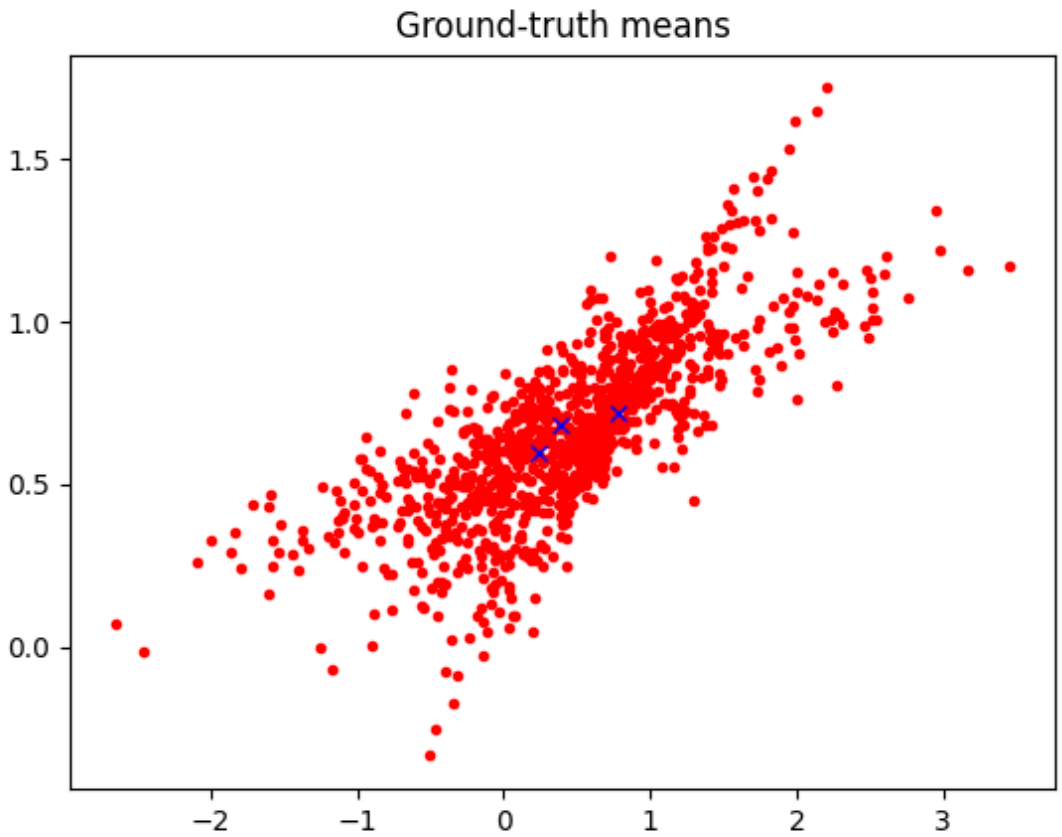
Countour plots in log scale



Check the code `gibbs_dirichlet_process.py` for details.

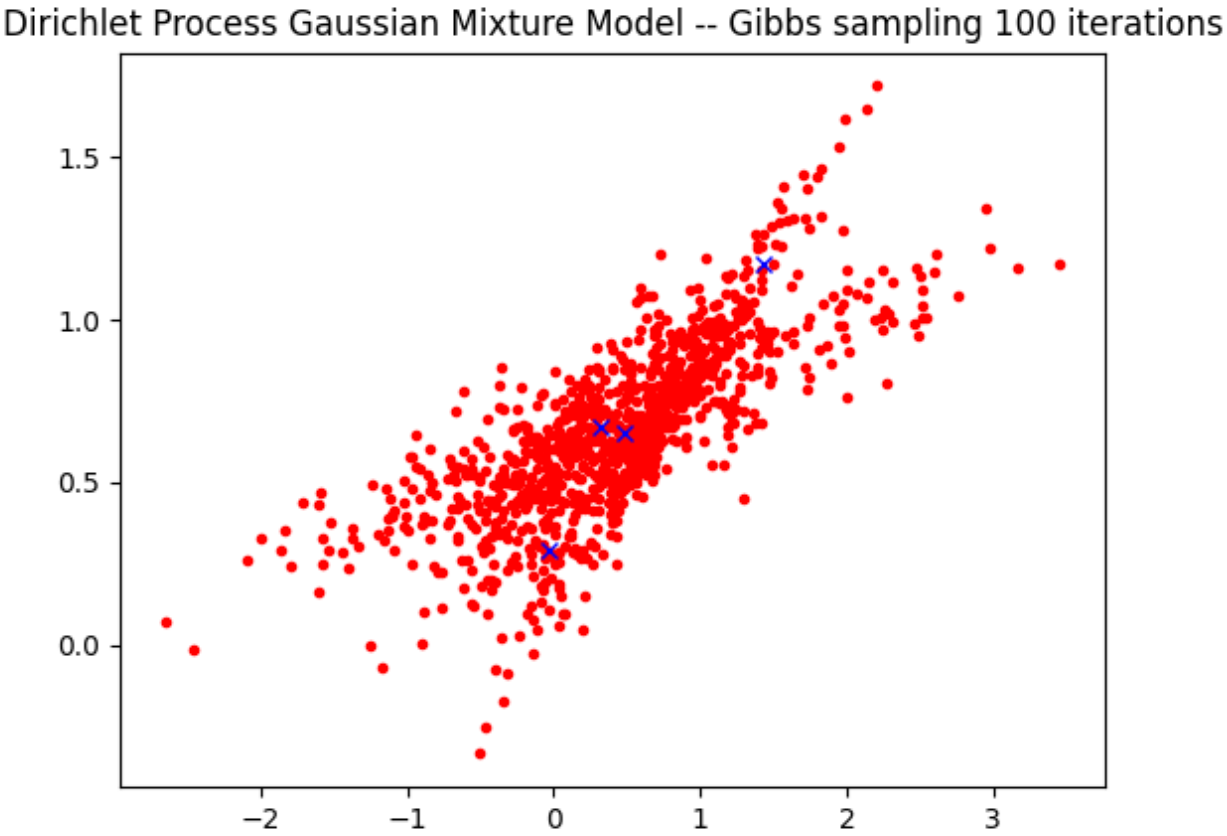
Gibbs – Fitting a DP mixture model (9)

Example 2: Generate data by a mixture of 3 Gaussians (**random** μ, **random** Σ). Red dots are data. Blue crosses are the means.



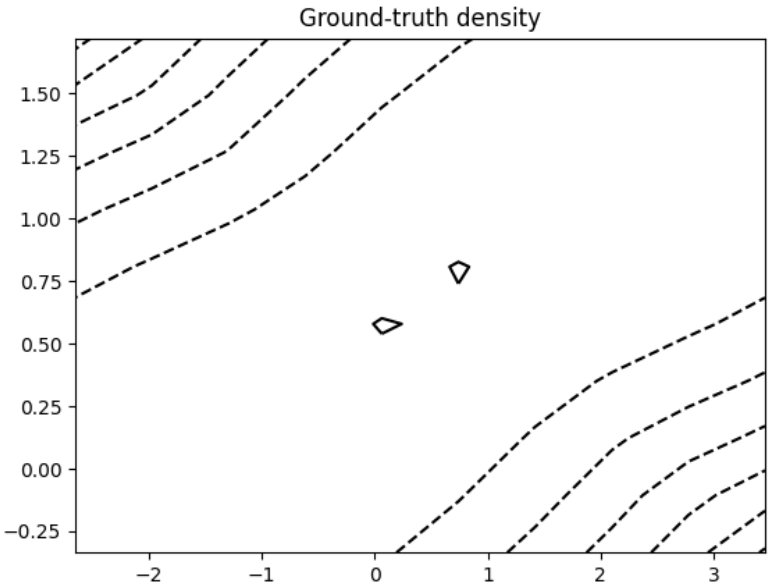
Gibbs – Fitting a DP mixture model (10)

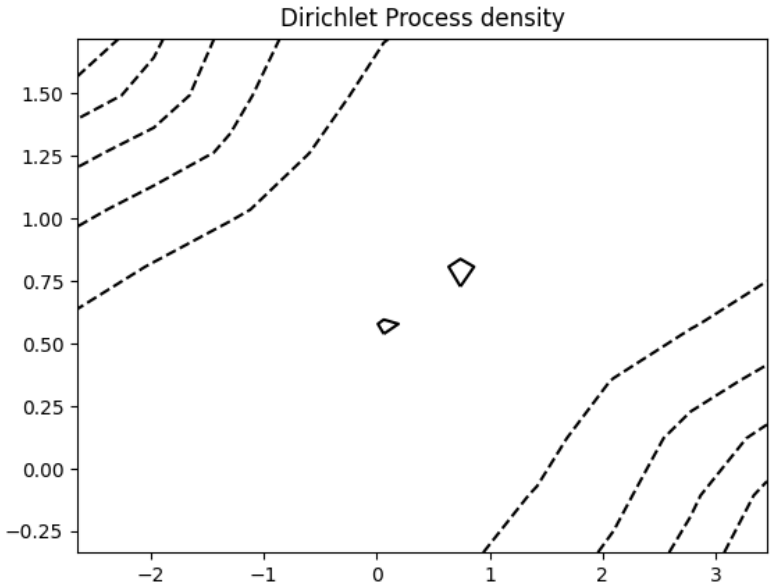
Dirichlet Process Gaussian Mixture Model: Initial stick-breaking process with $\alpha = 1$ and stoping $\epsilon = 0.05$. Gibbs sampling runs 100 iterations. Initially, there are $K = 5$ components from the stick-breaking process. After 100 iterations, number of components shrinked into 4.



Gibbs – Fitting a DP mixture model (11)

Countour plots in log scale





Check the code `gibbs_dirichlet_process.py` for details.

Gibbs sampling vs. Expectation Maximization (EM)

The goal of EM is to maximize the log likelihood of the observed data ($\boldsymbol{x}_i$ visible, $\boldsymbol{z}_i$ hidden) that is **hard** to optimize:

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(\boldsymbol{x}_i|\boldsymbol{\theta}) = \sum_{i=1}^{N} \log \left[\sum_{\boldsymbol{z}_i} p(\boldsymbol{x}_i, \boldsymbol{z}_i|\boldsymbol{\theta}) \right]$$

EM gets around the problem. The E step is to compute the **expected complete data log likelihood**:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{t-1}) = \mathbb{E}[\ell_c(\boldsymbol{\theta})|\mathcal{D}, \boldsymbol{\theta}^{t-1}]$$

where the **complete data log likelihood** is defined as:

$$\ell_c(\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(\boldsymbol{x}_i, \boldsymbol{z}_i|\boldsymbol{\theta})$$

In the M step, we optimize Q function wrt $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^t = \text{argmax}_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{t-1})$$

Collapsed Gibbs – Fitting a DP mixture model (1)

The simpler way to fit a DPMM is to use **collapsed** Gibbs sampler. We have:

$$p(z_i = k|\boldsymbol{z}_{-i}, \boldsymbol{x}, \alpha, \boldsymbol{\lambda}) \propto p(z_i = k|\boldsymbol{z}_{-i}, \alpha)p(\boldsymbol{x}_i|\boldsymbol{x}_{-i}, z_i = k, \boldsymbol{z}_{-i}, \boldsymbol{\lambda})$$

By exchangeability, we can assume that z_i is the last customer to enter the restaurant. The first term is given by:

$$p(z_i = k|\boldsymbol{z}_{-i}, \alpha) = \frac{1}{\alpha + N - 1}\left(\alpha\mathbb{I}(z_i = k^*) + \sum_{k=1}^{K} N_{k,-i}\mathbb{I}(z_i = k)\right)$$

where K is the number of clusters used by $\boldsymbol{z}_{-i}$, and k^* is a new cluster.

- If k has been seen before:

$$p(z_i = k|\boldsymbol{z}_{-i}, \alpha) = \frac{N_{k,-i}}{\alpha + N - 1}$$

- If k is a new cluster:

$$p(z_i = k|\boldsymbol{z}_{-i}, \alpha) = \frac{\alpha}{\alpha + N - 1}$$

Collapsed Gibbs – Fitting a DP mixture model (2)

To compute the second term $p(\boldsymbol{x}_i|\boldsymbol{x}_{-i}, z_i = k, \boldsymbol{z}_{-i}, \boldsymbol{\lambda})$, let us partition the data $\boldsymbol{x}_{-i}$ into clusters based on $\boldsymbol{z}_{-i}$. Let $\boldsymbol{x}_{-i,c} = \{\boldsymbol{x}_j : z_j = c, j \neq i\}$ be the data assigned to cluster c. If $z_i = k$, then $\boldsymbol{x}_i$ is conditionally independent of all the data points except those assigned to cluster k:

$$p(\boldsymbol{x}_i|\boldsymbol{x}_{-i}, z_i = k, \boldsymbol{z}_{-i}, \boldsymbol{\lambda}) = p(\boldsymbol{x}_i|\boldsymbol{x}_{-i,k}, \boldsymbol{\lambda}) = \frac{p(\boldsymbol{x}_i, \boldsymbol{x}_{-i,k}|\boldsymbol{\lambda})}{p(\boldsymbol{x}_{-i,k}|\boldsymbol{\lambda})}$$

where:

$$p(\boldsymbol{x}_i, \boldsymbol{x}_{-i,k}|\boldsymbol{\lambda}) = \int p(\boldsymbol{x}_i|\boldsymbol{\theta}_k) \left[\prod_{j \neq i : z_j = k} p(\boldsymbol{x}_j|\boldsymbol{\theta}_k) \right] H(\boldsymbol{\theta}_k|\boldsymbol{\lambda}) d\boldsymbol{\theta}_k$$

If $z_i = k^*$, corresponding to a new cluster, we have:

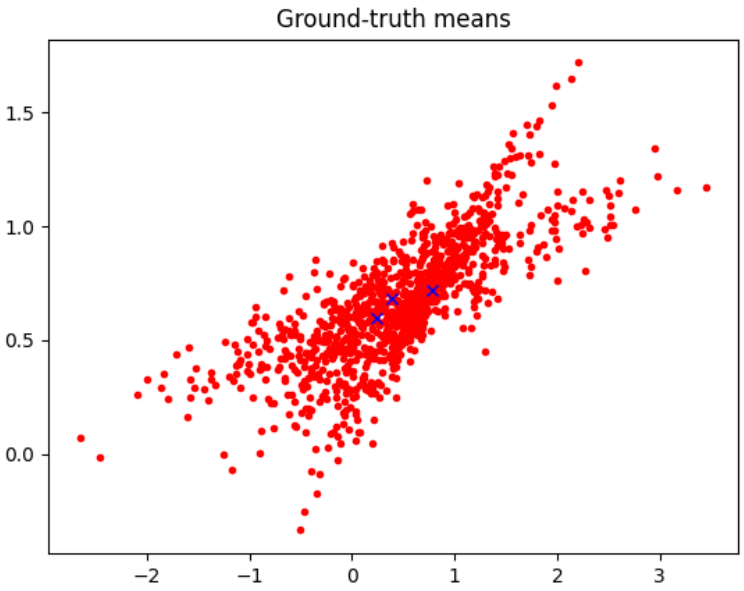
$$p(\boldsymbol{x}_i|\boldsymbol{x}_{-i}, z_i = k^*, \boldsymbol{z}_{-i}, \boldsymbol{\lambda}) = p(\boldsymbol{x}_i|\boldsymbol{\theta}) = \int p(\boldsymbol{x}_i|\boldsymbol{\lambda}) H(\boldsymbol{\theta}|\boldsymbol{\lambda}) d\boldsymbol{\theta}$$

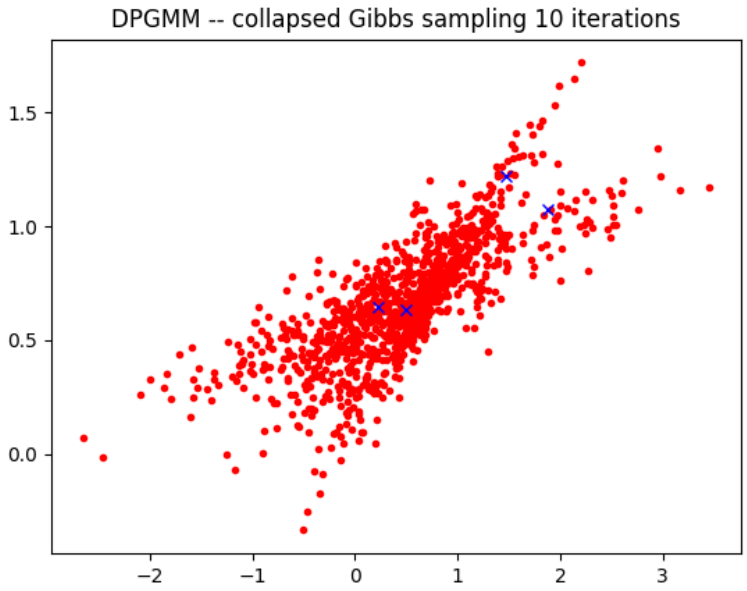
Collapsed Gibbs – Fitting a DP mixture model (3)

Algorithm 25.1: Collapsed Gibbs sampler for DP mixtures

for *each* $i = 1 : N$ *in random order* **do**
- Remove $\mathbf{x}_i$'s sufficient statistics from old cluster z_i ;
- **for** *each* $k = 1 : K$ **do**
  - Compute $p_k(\mathbf{x}_i) = p(\mathbf{x}_i|\mathbf{x}_{-i}(k))$;
  - Set $N_{k,-i} = \dim(\mathbf{x}_{-i}(k))$;
  - Compute $p(z_i = k|\mathbf{z}_{-i}, \mathcal{D}) = \frac{N_{k,-i}}{\alpha+N-1}$;
- Compute $p_*(\mathbf{x}_i) = p(\mathbf{x}_i|\boldsymbol{\lambda})$;
- Compute $p(z_i = *|\mathbf{z}_{-i}, \mathcal{D}) = \frac{\alpha}{\alpha+N-1}$;
- Normalize $p(z_i|\cdot)$;
- Sample $z_i \sim p(z_i|\cdot)$;
- Add $\mathbf{x}_i$'s sufficient statistics to new cluster z_i ;
- If any cluster is empty, remove it and decrease K;

Collapsed Gibbs – Fitting a DP mixture model (4)



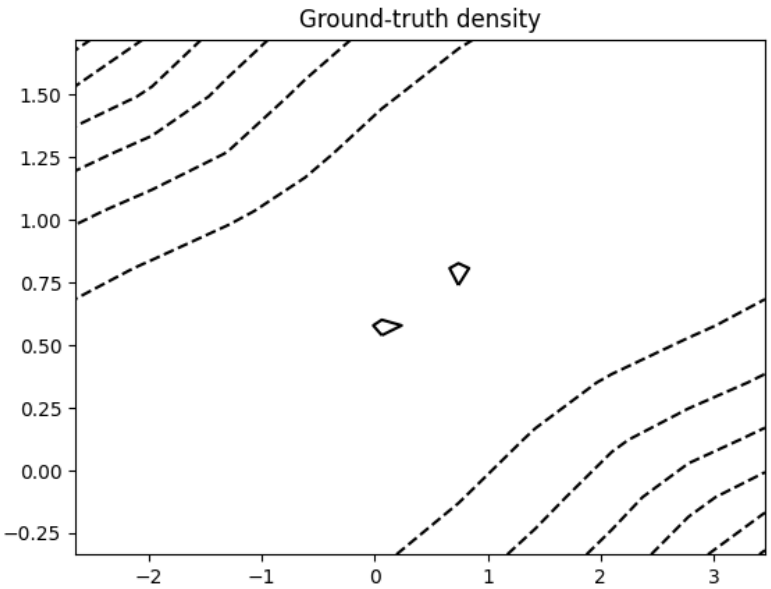


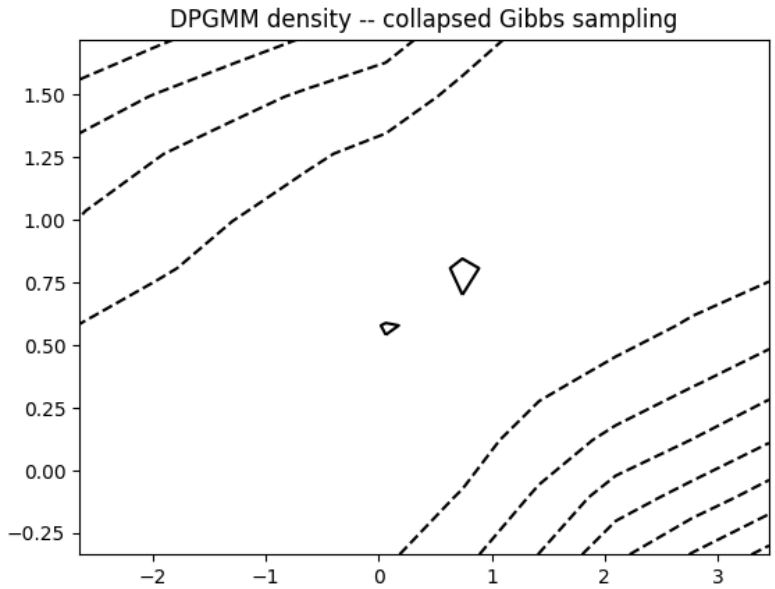
Initial number of components: 5

Final number of components: 4

Number of iterations: 10 ($<<$ 100 of naive Gibbs)

Collapsed Gibbs – Fitting a DP mixture model (5)





Code: `collapsed_gibbs_dirichlet_process.py`

Topic modeling – Introduction

Mission

Unsupervised classification of a large corpus into topics (not known). Last time, we talked about another work of David Blei's hierarchical Chinese Restaurant Process that is an extension of LDA.

Terminology:

- A **word** is the basic unit of discrete data, defined to be an item from a vocabulary indexed by $\{1, .., V\}$. We represent words using unit-basis vectors that have a single component equal to one and all other components equal to zero. For example, the v-th word is represented by a V-vector w such that $w^v = 1$ and $w^u = 0$ for $u \neq v$.
- A **document** is a sequence of N words denoted by $\mathbf{w} = (w_1, w_2, .., w_N)$, where w_n is the n-th word in the sequence.
- A **corpus** is a collection of M documents denoted by $\boldsymbol{D} = \{\boldsymbol{w}_1, .., \boldsymbol{w}_M\}$.

Topmic modeling – Related work: Probabilistic approaches

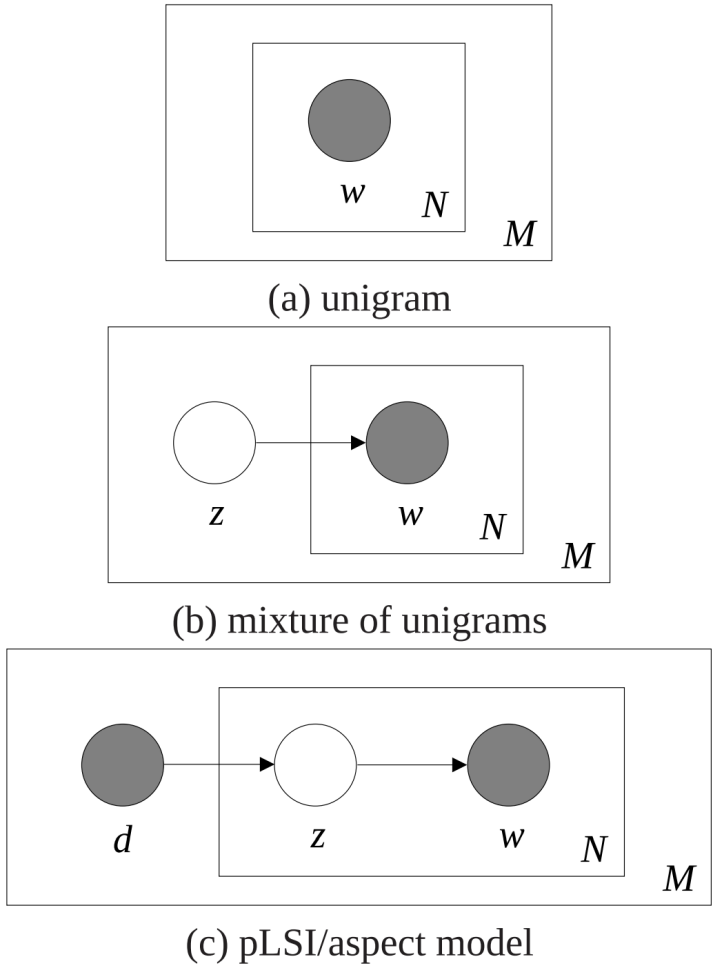


Figure 3: Graphical model representation of different models of discrete data.

Topic modeling – Related work: Probabilistic approaches

1. **Unigram model**: The words of every document are drawn independently from a single multinomial distribution.

$$p(\boldsymbol{w}) = \prod_{n=1}^{N} p(w_n)$$

2. **Mixture of unigrams**: Each document is generated by first choosing a topic z (only a single one!), and then generating N words independently.

$$p(\boldsymbol{w}) = \sum_{z} p(z) \prod_{n=1}^{N} p(w_n|z)$$

3. **Probabilistic latent semantic indexing (pLSI)**: A document label d and a word w_n are conditionally independent given an unobserved topic z.

$$p(d, w_n) = p(d) \sum_{z} p(w_n|z)p(z|d)$$

Topic modeling – Related work: Probabilistic approaches

Mixture of unigrams: Each document is generated by first choosing a topic z (only a single one!), and then generating N words independently.

$$p(\boldsymbol{w}) = \sum_{z} p(z) \prod_{n=1}^{N} p(w_n|z)$$

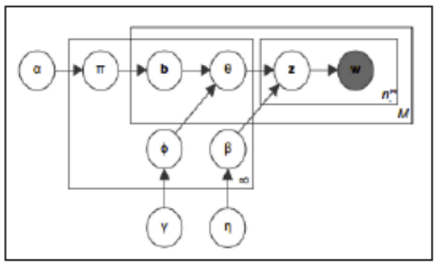
My opinion

LDA is just a relaxed version of mixture of unigrams with a flexible partition of words (each partition belongs to a topic) in a document.

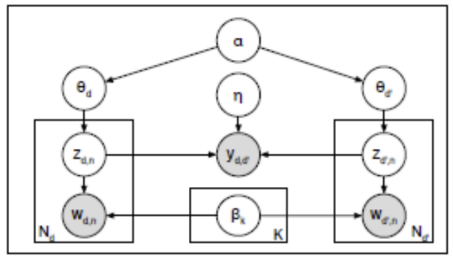
Topic modeling – Related work: Probabilistic approaches

Topic models zoo

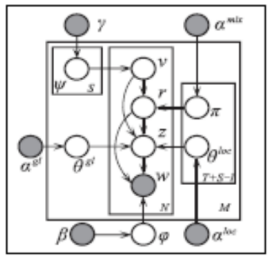
Williamson et al. 2010



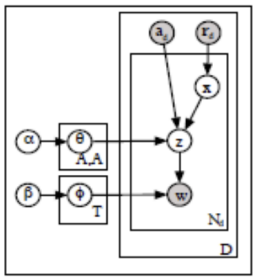
Chang & Blei, 2009



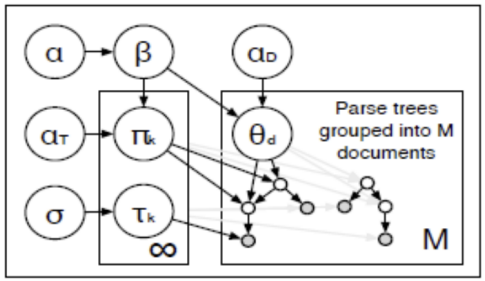
Titov & McDonald, 2008



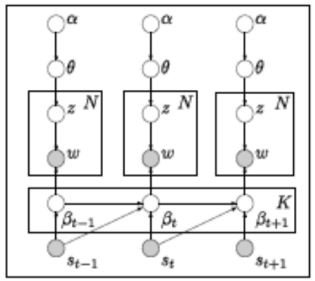
McCallum et al. 2007



Boyd-Graber & Blei, 2008



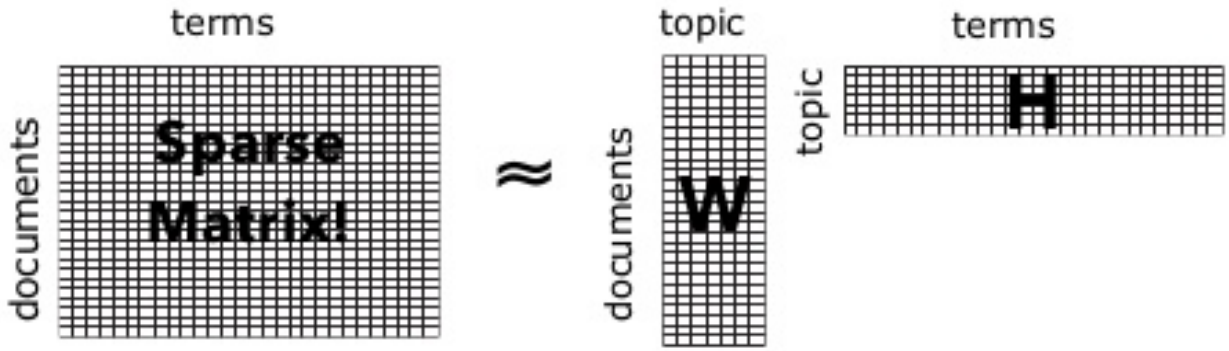
Wang & Blei, 2008



Reference: https://www.cs.cmu.edu/~epxing/Class/10708-17/notes-17/10708-scribe-lecture13.pdf

Topic modeling – Related work: NMF approach

TOPIC MODELING / LATENT SEMANTIC ANALYSIS



$$X \approx WH$$

Non-negative Matrix Factorization (NMF):

$$\arg\min_{\mathbf{W},\mathbf{H}} \|\mathbf{X} - \mathbf{WH}\| \quad \text{s. t.} \quad \mathbf{W}, \mathbf{H} \geq 0$$

(~1970 Lawson, ~1995 Paatero, ~2000 Lee & Seung)

2005 Gaussier et al. "Relation between PLSA and NMF and implications."

Topic modeling – Latent Dirichlet Allocation (LDA)

Latent Dirichlet allocation (LDA) is a generative probabilistic model of a corpus. The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words.

LDA assumes that following generative process for each document $\boldsymbol{w}$ in a corpus $\boldsymbol{D}$:

1. Choose $N \sim \text{Poisson}(\lambda)$
2. Choose $\theta \sim \text{Dir}(\alpha)$
3. For each word $w_n, n \in \{1, .., N\}$:
   - Choose a topic $z_n \sim \text{Multinomial}(\theta)$
   - Choose a word w_n from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

Topic Modeling – Latent Dirichlet Allocation (LDA)

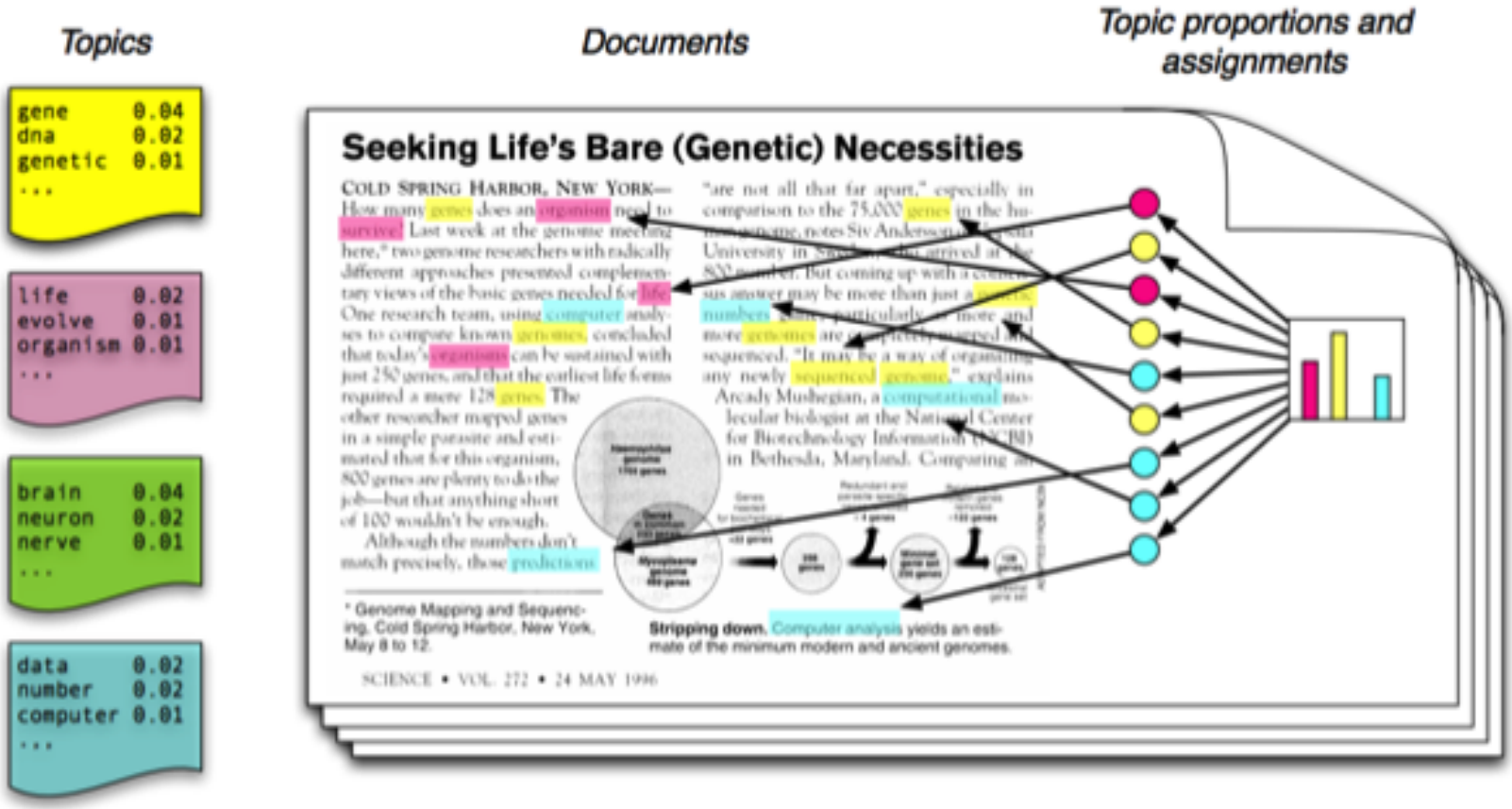


Figure source: Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77-84.

Topic Modeling – Latent Dirichlet Allocation (LDA)

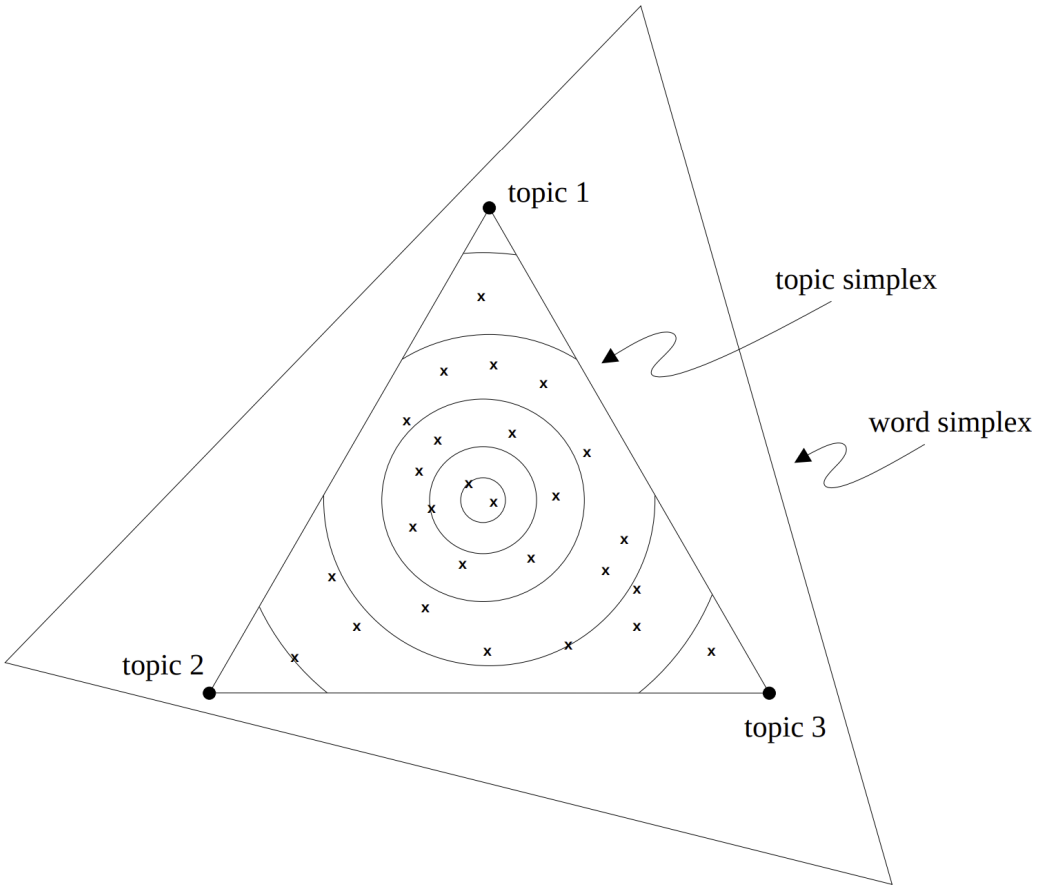


Figure 4: The topic simplex for three topics embedded in the word simplex for three words.

Notation change – Sorry about that!

A k-dimensional Dirichlet random variable θ can take values in the $(k-1)$-simplex (a k-vector θ lies in the $(k-1)$-simplex if $\theta_i \geq 0$, $\sum_{i=1}^{k} \theta_i = 1$), and has the following probability density on the simplex:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{k} \theta_i^{\alpha_i - 1}$$

where the parameter α is a k-vector with components $\alpha_i > 0$ and Γ is the Gamma function.

LDA – Overview (1)

Given the parameters α and β, the joint distribution of a topic mixture θ, a set of N topics $\boldsymbol{z}$, and a set of N words $\boldsymbol{w}$ is given by:

$$p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n, \beta)$$

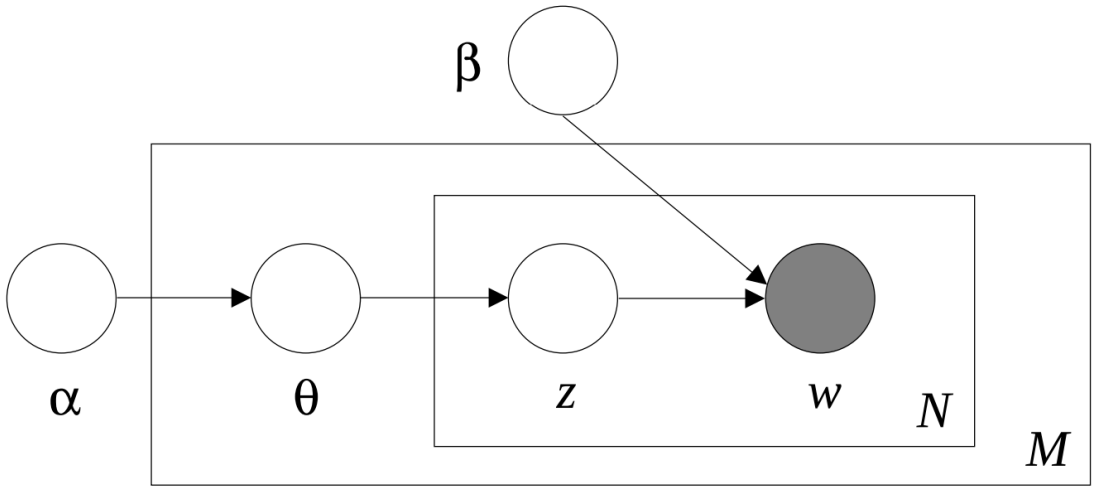


Figure 1: Graphical model representation of LDA. The boxes are "plates" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.

LDA – Overview (2)

Integrating over θ and summing over z, we obtain the marginal distribution of a document:

$$p(\boldsymbol{w}|\alpha,\beta) = \int p(\theta|\alpha)\left(\prod_{n=1}^{N}\sum_{z_n} p(z_n|\theta)p(w_n|z_n,\beta)\right)d\theta$$

Taking the product of the marginal probabilities of single documents, we obtain the probability of a corpus:

$$p(\boldsymbol{D}|\alpha,\beta) = \prod_{d=1}^{M}\int p(\theta_d|\alpha)\left(\prod_{n=1}^{N_d}\sum_{z_{dn}} p(z_{dn}|\theta_d)p(w_{dn}|z_{dn},\beta)\right)d\theta_d$$

LDA – overview (3)

LDA representation has three levels:

1. The parameters α and β are corpus-level parameters, assumed to be sampled once in the process.
2. The variables θ_d are document-level variables, sampled once per document.
3. The variables z_{dn} and w_{dn} are word-level variables and are sampled once for each word in each document.

Few assumptions:

- The dimensionality k of the Dirichlet distribution (or dimensionality of the topic variable z) is assumed to be known and **fixed**.
- The word probabilities are parameterized by a $k \times V$ matrix β where $\beta_{ij} = p(w^j = 1 | z^i = 1)$, fixed and to be estimated.

LDA – Inference (1)

Computing the posterior distribution of the hidden variables given a document is **intractable**:

$$p(\theta, \boldsymbol{z}|\boldsymbol{w}, \alpha, \beta) = \frac{p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta)}{p(\boldsymbol{w}|\alpha, \beta)}$$

$$p(\boldsymbol{w}|\alpha, \beta) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) d\theta$$

Idea 1

The exact/approximate **Variational Principles** (that I presented that last time):

- If something is too hard to compute, cast it as an optimization problem and the dual form.
- After a bit of algebra, it turns out to be a KL divergence minimization.

LDA – Inference (2)

Idea 2

Decouple θ and β as in collapsed Gibbs sampling.

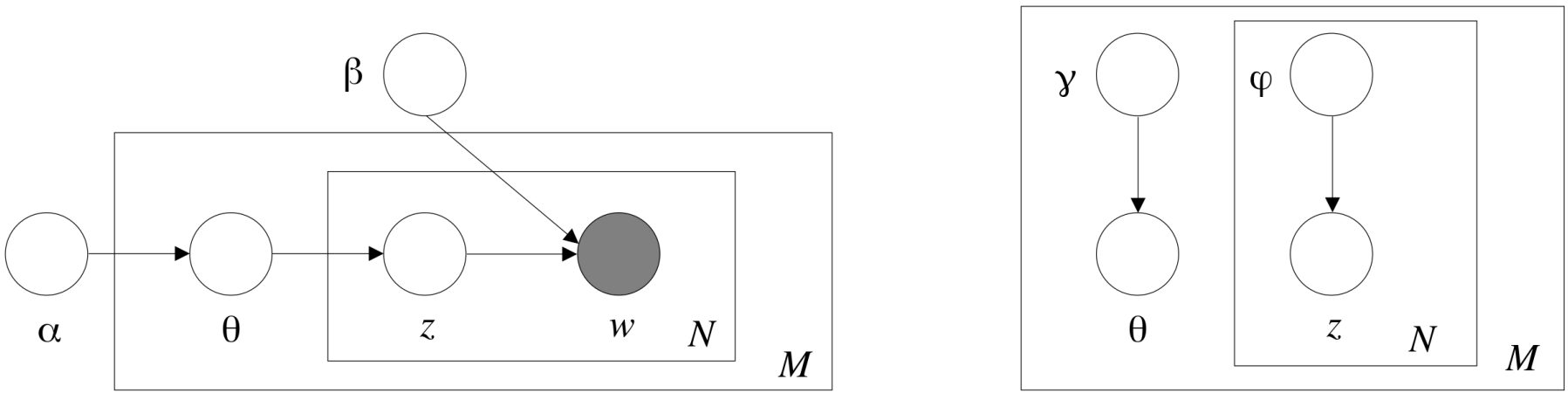


Figure 5: (Left) Graphical model representation of LDA. (Right) Graphical model representation of the variational distribution used to approximate the posterior in LDA.

LDA – Variational Principles (1)

The optimal values of the variational parameters are found by minimizing the Kullback-Leibler (KL) divergence between the variational distribution and the true posterior $p(\theta, \boldsymbol{z}|\boldsymbol{w}, \alpha, \beta)$:

$$(\gamma^*, \phi^*) = \arg\min_{(\gamma,\phi)} \mathcal{D}(q(\theta, \boldsymbol{z}|\gamma, \phi)||p(\theta, \boldsymbol{z}|\boldsymbol{w}, \alpha, \beta))$$

where the variational distribution:

$$p(\theta, \boldsymbol{z}|\gamma, \phi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\phi_n)$$

and the Dirichlet parameter γ and the multinomial paramters $(\phi_1, .., \phi_N)$ are the free variational parameters. The minimization can be achieved by an **iterative fixed-point method** ← I think it is just mean-field algorithm.

LDA – Variational Principles (2)

Given a corpus of documents $\boldsymbol{D} = \{\boldsymbol{w}_1, .., \boldsymbol{w}_M\}$, find parameters α and β that maximize the (marginal) log likelihood of the data:

$$\ell(\alpha, \beta) = \sum_{d=1}^{M} \log p(\boldsymbol{w}_d | \alpha, \beta)$$

Variational EM:

1. (E-step) For each document, find the optimizing values of the variational parameters $\{\gamma_d^*, \phi_d^* : d \in \boldsymbol{D}\}$.
2. (M-step) Maximize the lower bound of the log likelihood with respect to the model parameters α and β:

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

Metrics – Natural Language Processing (NLP)

In information theory, the perplexity of a descrete probability distribution p is defined as:

$$\text{perplexity}(p) = 2^{H(p)} = 2^{-\sum_x p(x) \log_2 p(x)}$$

where the exponent $H(p)$ is the entropy. Given an unknown probability distribution p, a propose probability model q, samples $\{x_1, .., x_N\}$ drawn from p. The perplexity of the model q to measure how well q predicts the samples is defined as:

$$2^{H(\hat{p},q)} = 2^{-\sum_x \hat{p}(x) \log_2 q(x)} = 2^{-\frac{1}{N}\sum_{i=1}^{N} \log_2 q(x_i)}$$

By convention in language modeling, perplexity is the metric to measure the generalization performance (the lower score, the better), that is equivalent to the inverse of the geometric mean per-word likelihood:

$$\text{perplexity}(\boldsymbol{D}_{test}) = \exp\left\{ -\frac{\sum_{d=1}^{M} \log p(\boldsymbol{w}_d)}{\sum_{d=1}^{M} N_d} \right\}$$