Gibbs Sampling

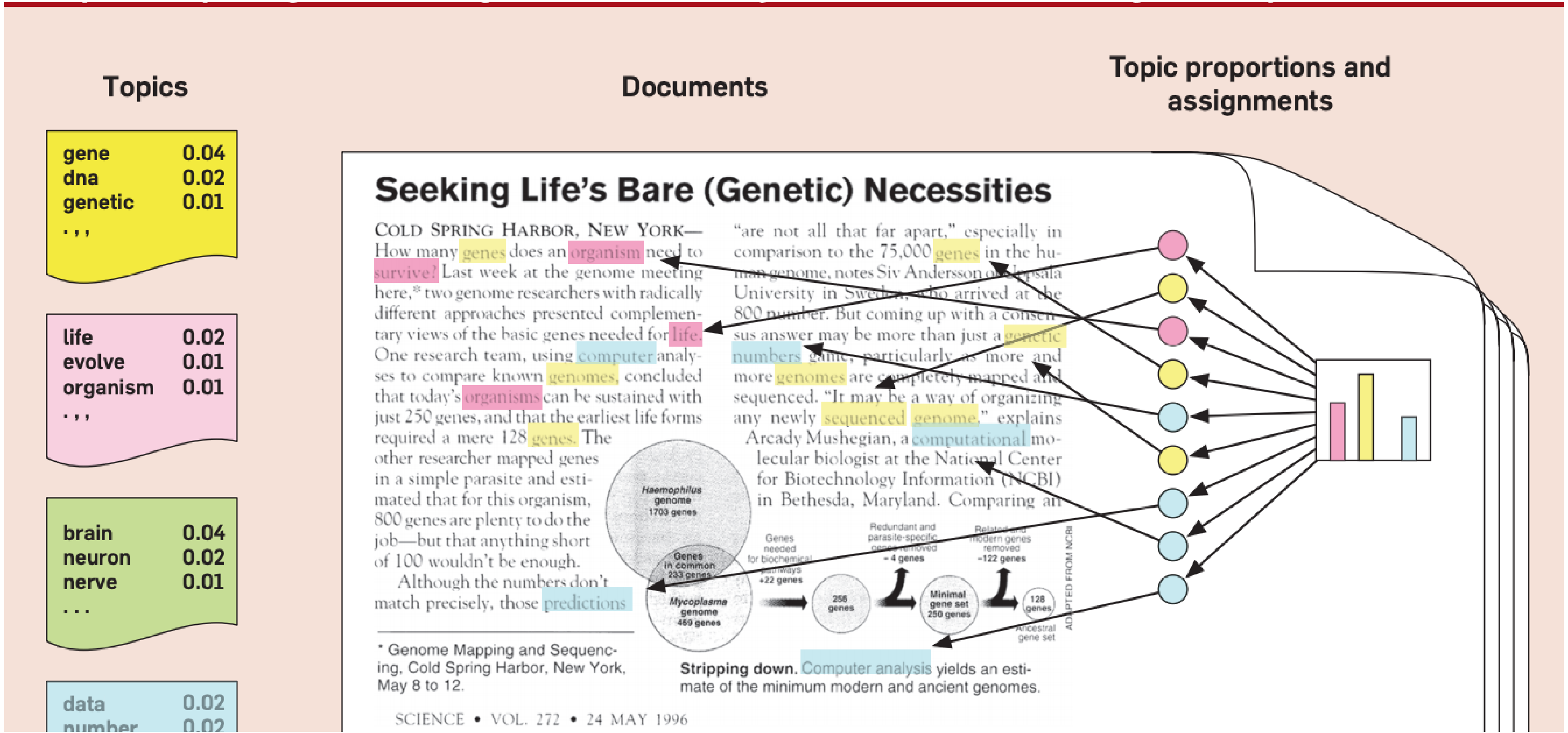
Héctor Corrada Bravo

University of Maryland, College Park, USA

CMSC 644: 2019-03-27

Latent semantic analysis

Documents as *mixtures* of topics (Hoffman 1999)



COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

SCIENCE • VOL. 272 • 24 MAY 1996

Latent semantic analysis

We have a set of documents D

Each document modeled as a bag-of-words (bow) over dictionary W.

$x_{w,d}$: the number of times word $w \in W$ appears in document $d \in D$.

Latent semantic analysis

Let's start with a simple model based on the frequency of word occurrences.

Each document is modeled as n_d draws from a *Multinomial* distribution with parameters $\theta_d = \{\theta_{1,d}, \ldots, \theta_{W,d}\}$

Note $\theta_{w,d} \geq 0$ and $\sum_w \theta_{w,d} = 1$.

Latent semantic analysis

Probability of observed corpus D

$$Pr(D|\{\theta_d\}) \propto \prod_{d=1}^{D} \prod_{w=1}^{W} \theta_{w,d}^{x_{w,d}}$$

Latent semantic analysis

Problem 1:

Prove MLE $\hat{\theta}_{w,d} = \frac{x_{w,d}}{n_d}$

Constrained optimization

We have a problem of type

$$\begin{array}{lc} \min_x & f_0(x) \\ \text{s.t.} & f_i(x) \leq 0 \; i = 1, \ldots, m \\ & h_i(x) = 0 \; i = 1, \ldots, p \end{array}$$

Note: This discussion follows Boyd and Vandenberghe, *Convex Optimization*

Constrained optimization

To solve these type of problems we will look at the *Lagrangian* function:

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i g_i(x)$$

Constrained optimization

We'll see these in more detail later, but there is a beautiful result giving *optimality conditions* based on the Lagrangian:

Suppose $\tilde{x}$, $\tilde{\lambda}$ and $\tilde{\nu}$ are *optimal*, then

$$
\begin{gathered}
f_i(\tilde{x}) \leq 0 \\
h_i(\tilde{x}) = 0 \\
\tilde{\lambda}_i \geq 0 \\
\tilde{\lambda}_i f_i(\tilde{x}) = 0 \\
\nabla L(\tilde{x}, \tilde{\lambda}, \tilde{\nu}) = 0
\end{gathered}
$$

Constrained optimization

We can use the gradient and feasibility conditions to prove the MLE result.

Probablistic Latent Semantic Analysis

Let's change our document model to introduce topics.

The key idea is that the probability of observing a *word* in a *document* is given by two pieces:

- The probability of observing a *topic* in a document, and
- The probability of observing a *word* given a *topic*

$$Pr(w,d) = \sum_{t=1}^{T} Pr(w|t)Pr(t|d)$$

Probablistic Latent Semantic Analysis

So, we rewrite corpus probability as

$$Pr(D|\{p_d\}\{\theta_t\}) \propto \prod_{d=1}^{D} \prod_{w=1}^{W} \left(\sum_{t=1}^{T} p_{t,d}\theta_{w,t} \right)^{x_{w,d}}$$

Probablistic Latent Semantic Analysis

So, we rewrite corpus probability as

$$Pr(D|\{p_d\}\{\theta_t\}) \propto \prod_{d=1}^{D} \prod_{w=1}^{W} \left(\sum_{t=1}^{T} p_{t,d}\theta_{w,t} \right)^{x_{w,d}}$$

Mixture of topics!!

Probablistic Latent Semantic Analysis

A fully observed model

Assume you know the *latent* number of occurences of word w in document d generated from topic t:

$\Delta_{w,d,t}$, such that $\sum_t \Delta_{w,d,t} = x_{w,d}$.

In that case we can rewrite corpus probability:

$$Pr(D|\{p_d\},\{\theta_t\}) \propto \prod_{d=1}^{D}\prod_{w=1}^{W}\prod_{t=1}^{T}(p_{t,d}\theta_{w,t})^{\Delta_{w,d,t}}$$

Probablistic Latent Semantic Analysis

Problem 2 Show MLEs given by

$$\hat{p}_{t,d} = \frac{\sum_{w=1}^{W} \Delta_{w,d,t}}{\sum_{t=1}^{T} \sum_{w=1}^{W} \Delta_{w,d,t}}$$

$$\hat{\theta}_{t,d} = \frac{\sum_{d=1}^{D} \Delta_{w,d,t}}{\sum_{w=1}^{W} \sum_{d=1}^{D} \Delta_{w,d,t}}$$

Probablistic Latent Semantic Analysis

Since we don't observe $\Delta_{w,d,t}$ we use the EM algorithm

At each iteration (given current parameters $\{p_d\}$ and $\{\theta_d\}$ find *responsibility*

$$\gamma_{w,d,t} = E[\Delta_{w,d,t} | \{p_d\}, \{\theta_t\}]$$

and maximize fully observed likelihood plugging in $\gamma_{w,d,t}$ for $\Delta_{w,d,t}$

Probablistic Latent Semantic Analysis

Problem 4: Show

$$\gamma_{w,d,t} = x_{w,d} \times \frac{p_{t,d}\theta_{w,t}}{\sum_{t'=1}^{T} p_{t',d}\theta_{w,t'}}$$

The EM Algorithm in General

So, why does that work?

Why does plugging in $\gamma_{w,d,t}$ for the latent variables $\Delta_{w,d,t}$ work?

Why does that maximize log-likelihood $\ell(\{p_d\}, \{\theta_t\}; D)$?

The EM Algorithm in General

Think of it as follows:

Z: observed data

Z^m: missing *latent* data $T = (Z, Z^m)$: complete data (observed and missing)

The EM Algorithm in General

Think of it as follows:

Z: observed data

Z^m: missing *latent* data $T = (Z, Z^m)$: complete data (observed and missing)

$\ell(\theta'; Z)$: log-likehood w.r.t. *observed* data

$\ell_0(\theta'; T)$: log-likelihood w.r.t. *complete* data

The EM Algorithm in General

Next, notice that

$$Pr(Z|\theta') = \frac{Pr(T|\theta')}{Pr(Z^m|Z, \theta')}$$

The EM Algorithm in General

Next, notice that

$$Pr(Z|\theta') = \frac{Pr(T|\theta')}{Pr(Z^m|Z,\theta')}$$

As likelihood:

$$\ell(\theta';Z) = \ell_0(\theta';T) - \ell_1(\theta';Z^m|Z)$$

The EM Algorithm in General

Iterative approach: given parameters θ take expectation of log-likelihoods

$$\begin{aligned} \ell(\theta'; Z) &= E[\ell_0(\theta'; T)|Z, \theta] - E[\ell_1(\theta'; Z^m|Z)|Z, \theta] \\ &\equiv Q(\theta', \theta) - R(\theta', \theta) \end{aligned}$$

The EM Algorithm in General

Iterative approach: given parameters θ take expectation of log-likelihoods

$$\begin{aligned} \ell(\theta'; Z) &= E[\ell_0(\theta'; T)|Z, \theta] - E[\ell_1(\theta'; Z^m|Z)|Z, \theta] \\ &\equiv Q(\theta', \theta) - R(\theta', \theta) \end{aligned}$$

In pLSA, $Q(\theta', \theta)$ is the log likelihood of complete data with $\Delta_{w,d,t}$ replaced by

$\gamma_{w,d,t}$

The EM Algorithm in General

The general EM algorithm

1. Initialize parameters $\theta^{(0)}$
2. Construct *function* $Q(\theta', \theta^{(j)})$
3. Find next set of parameters $\theta^{(j+1)} = \arg\max_{\theta'} Q(\theta', \theta^{(j)})$
4. Iterate steps 2 and 3 until convergence

The EM Algorithm in General

So, why does that work?

$$\begin{aligned} \ell(\theta^{(j+1)}; Z) - \ell(\theta^{(j)}; Z) = & \quad [Q(\theta^{(j+1)}, \theta^{(j)}) - Q(\theta^{(j)}, \theta^{(j)})] \\ & -[R(\theta^{(j+1)}, \theta^{(j)}) - R(\theta^{(j)}, \theta^{(j)})] \\ \geq & \quad 0 \end{aligned}$$

The EM Algorithm in General

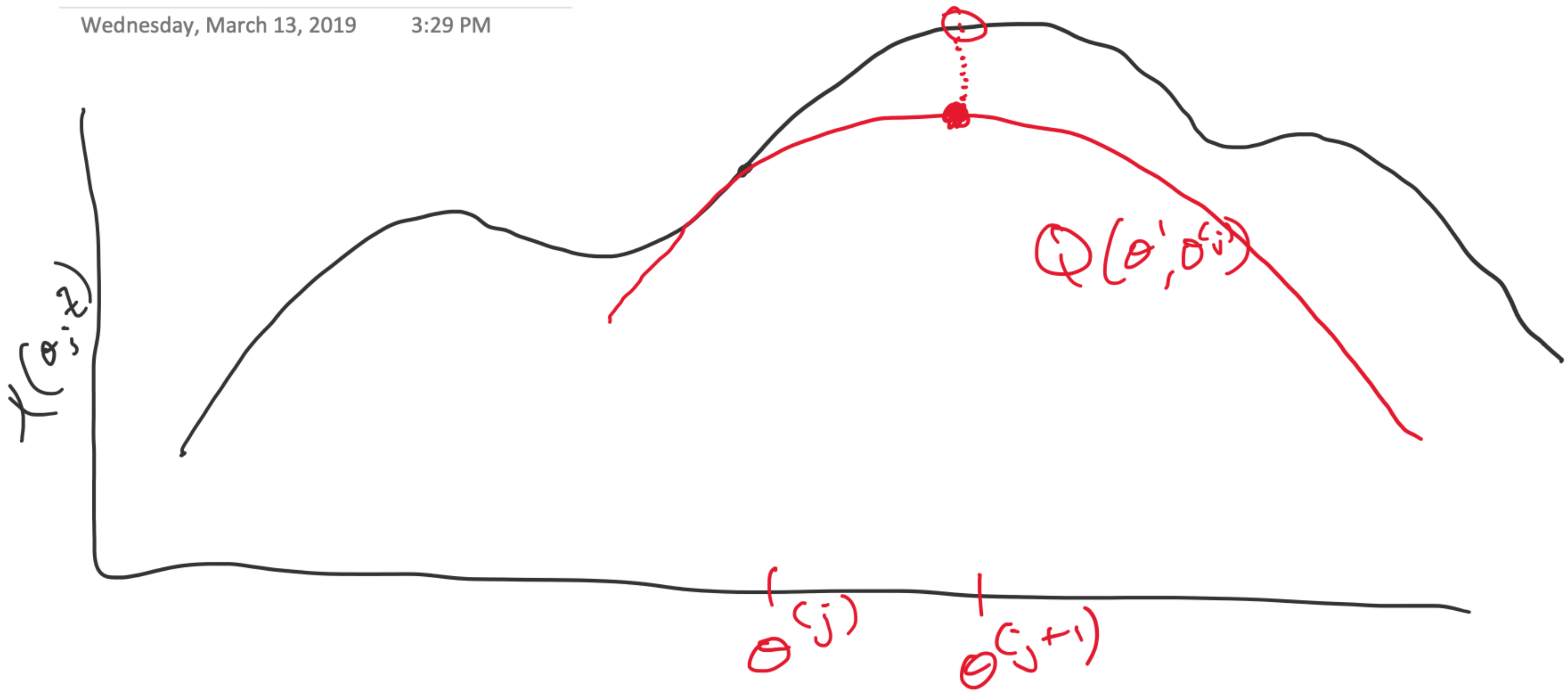
So, why does that work?

$$\begin{aligned}\ell(\theta^{(j+1)}; Z) - \ell(\theta^{(j)}; Z) = & \quad [Q(\theta^{(j+1)}, \theta^{(j)}) - Q(\theta^{(j)}, \theta^{(j)})] \\ & -[R(\theta^{(j+1)}, \theta^{(j)}) - R(\theta^{(j)}, \theta^{(j)})] \\ \geq & \quad 0\end{aligned}$$

I.E., every step makes log-likehood larger

The EM Algorithm in General

Why else does it work? $Q(\theta', \theta)$ *minorizes* $\ell(\theta'; Z)$



The EM Algorithm in General

General algorithmic concept:

Iterative approach:

- Initialize parameters
- Construct bound based on current parameters
- Optimize bound

The EM Algorithm in General

General algorithmic concept:

Iterative approach:

- Initialize parameters
- Construct bound based on current parameters
- Optimize bound

We will see this again when we look at *variational* methods

Approximate Inference by Sampling

Ultimately, what we are interested in is learning topics

Perhaps instead of finding parameters θ that maximize likelihood

Sample from a distribution $Pr(\theta|D)$ that gives us topic estimates

Approximate Inference by Sampling

Ultimately, what we are interested in is learning topics

Perhaps instead of finding parameters θ that maximize likelihood

Sample from a distribution $Pr(\theta|D)$ that gives us topic estimates

But, we only have talked about $Pr(D|\theta)$ how can we sample parameters?

Approximate Inference by Sampling

Like EM, the trick here is to expand model with *latent* data Z^m

And sample from distribution $Pr(\theta, Z^m|Z)$

Approximate Inference by Sampling

Like EM, the trick here is to expand model with *latent* data Z^m

And sample from distribution $Pr(\theta, Z^m|Z)$

This is challenging, but sampling from $Pr(\theta|Z^m, Z)$ and $Pr(Z^m|\theta, Z)$ is easier

Approximate Inference by Sampling

The *Gibbs Sampler* does exactly that

Property: After some rounds, samples from the conditional distributions

$Pr(\theta|Z^m, Z)$

Correspond to samples from marginal $Pr(\theta|Z) = \sum_{Z^m} Pr(\theta, Z^m|Z)$

Approximate Inference by Sampling

Quick aside, how to simulate data for pLSA?

- Generate parameters $\{p_d\}$ and $\{\theta_t\}$
- Generate $\Delta_{w,d,t}$

Approximate Inference by Sampling

Let's go backwards, let's deal with $\Delta_{w,d,t}$

Approximate Inference by Sampling

Let's go backwards, let's deal with $\Delta_{w,d,t}$

$$\Delta_{w,d,t} \sim \mathrm{Mult}_{\mathrm{x_{w,d}}}(\gamma_{w,d,1}, \ldots, \gamma_{w,d,T})$$

Where $\gamma_{w,d,t}$ was as given by E-step

Approximate Inference by Sampling

Let's go backwards, let's deal with $\Delta_{w,d,t}$

$$\Delta_{w,d,t} \sim \mathrm{Mult}_{\mathrm{x_{w,d}}}(\gamma_{w,d,1}, \ldots, \gamma_{w,d,T})$$

Where $\gamma_{w,d,t}$ was as given by E-step

```
for d in range(num_docs):

  delta[d,w,:] = np.random.multinomial(doc_mat[d,w],

      gamma[d,w,:])
```

Approximate Inference by Sampling

Hmm, that's a problem since we need $x_{w,d}$...

But, we know $Pr(w,d) = \sum_t p_{t,d}\theta_{w,t}$ so, let's use that to generate each $x_{w,d}$ as

$$x_{w,d} \sim \text{Mult}_{n_d}(Pr(1,d), \ldots, Pr(W,d))$$

Approximate Inference by Sampling

Hmm, that's a problem since we need $x_{w,d}$...

But, we know $Pr(w,d) = \sum_t p_{t,d}\theta_{w,t}$ so, let's use that to generate each $x_{w,d}$ as

$$x_{w,d} \sim \mathrm{Mult}_{n_d}(Pr(1,d), \ldots, Pr(W,d))$$

```
for d in range(num_docs):
  doc_mat[d,:] = np.random.multinomial(nw[d], np.sum(p[:,d] * theta), axis=0)
```

Approximate Inference by Sampling

Now, how about p_d? How do we generate the parameters of a Multinomial distribution?

Approximate Inference by Sampling

Now, how about p_d? How do we generate the parameters of a Multinomial distribution?

This is where the Dirichlet distribution comes in...

If $p_d \sim \mathrm{Dir}(\alpha)$, then

$$Pr(p_d) \propto \prod_{t=1}^{T} p_{t,d}^{\alpha_t - 1}$$

Approximate Inference by Sampling

Some interesting properties:

$$E[p_{t,d}] = \frac{\alpha_t}{\sum_{t'} \alpha_{t'}}$$

So, if we set all $\alpha_t = 1$ we will tend to have uniform probability over topics ($1/t$ each on average)

If we increase $\alpha_t = 100$ it will also have uniform probability but will have very little variance (it will almost always be $1/t$)

Approximate Inference by Sampling

So, we can say $p_d \sim \mathrm{Dir}(\alpha)$ and $\theta_t \sim \mathrm{Dir}(\beta)$

Approximate Inference by Sampling

So, we can say $p_d \sim \mathrm{Dir}(\alpha)$ and $\theta_t \sim \mathrm{Dir}(\beta)$

And generate data as (with $\alpha_t = 1$)

```
for d in range(num_docs):

  p[:,d] = np.random.dirichlet(1. * np.ones(num_topics))
```

Approximate Inference by Sampling

So what we have is a *prior* over parameters $\{p_d\}$ and $\{\theta_t\}$: $Pr(p_d|\alpha)$ and $Pr(\theta_t|\beta)$

And we can formulate a distribution for missing data $\Delta_{w,d,t}$:

$$Pr(\Delta_{w,d,t}|p_d, \theta_t, \alpha, \beta) =$$
$$Pr(\Delta_{w,d,t}|p_d, \theta_t) Pr(p_d|\alpha) Pr(\theta_t|\beta)$$

Approximate Inference by Sampling

However, what we care about is the *posterior* distribution $Pr(p_d|\Delta_{w,d,t}, \theta_t, \alpha, \beta)$

What do we do???

Approximate Inference by Sampling

Another neat property of the Dirichlet distribution is that it is *conjugate* to the Multinomial

If $\theta|\alpha \sim \mathrm{Dir}(\alpha)$ and $X|\theta \sim \mathrm{Multinomial}(\theta)$, then

$$\theta|X, \alpha \sim \mathrm{Dir}(X + \alpha)$$

Approximate Inference by Sampling

That means we can sample p_d from

$$p_{t,d} \sim \mathrm{Dir}(\sum_w \Delta_{w,d,t} + \alpha)$$

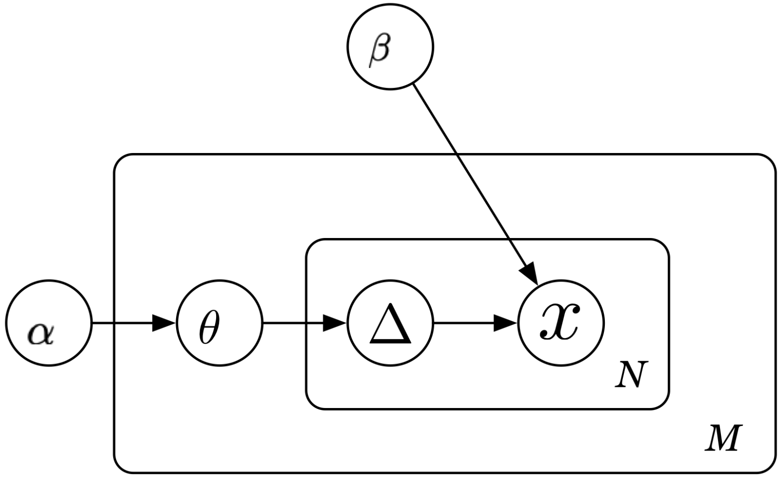
and

$$\theta_{w,t} \sim \mathrm{Dir}(\sum_d \Delta_{w,d,t} + \beta)$$

Approximate Inference by Sampling

Coincidentally, we have just specified the **Latent Dirichlet Allocation** method for topic modeling.

This is the most commonly used method for topic modeling



Blei, Ng, Jordan (2003), JMLR

Approximate Inference by Sampling

We can now specify a full Gibbs Sampler for an LDA mixture model.

Given:

- Word-document counts $x_{w,d}$
- Number of topics K
- Prior parameters α and β

Do: Learn parameters $\{p_d\}$ and $\{\theta_t\}$ for K topics

Approximate Inference by Sampling

Step 0: Initialize parameters $\{p_d\}$ and $\{\theta_t\}$

$$p_d \sim \mathrm{Dir}(\alpha)$$

and

$$\theta_t \sim \mathrm{Dir}(\beta)$$

Approximate Inference by Sampling

Step 1:

Sample $\Delta_{w,d,t}$ based on current parameters $\{p_d\}$ and $\{\theta_t\}$

$$\Delta_{w,d,\cdot} \sim \mathrm{Mult}_{x_{w,d}}(\gamma_{w,d,1}, \ldots, \gamma_{w,d,T})$$

Approximate Inference by Sampling

Step 2:

Sample parameters from

$$p_{t,d} \sim \mathrm{Dir}(\sum_w \Delta_{w,d,t} + \alpha)$$

and

$$\theta_{w,t} \sim \mathrm{Dir}(\sum_d \Delta_{w,d,t} + \beta)$$

Approximate Inference by Sampling

Step 3:

Get samples for a few iterations (e.g., 200), we want to reach a stationary distribution...

Approximate Inference by Sampling

Step 4:

Estimate $\hat{\Delta}_{w,d,t}$ as the average of the estimates from the last m iterations (e.g., m=500)

Approximate Inference by Sampling

Step 5:

Estimate parameters p_d and θ_t based on estimated $\hat{\Delta}_{w,d,t}$

$$\hat{p}_{t,d} = \frac{\sum_w \hat{\Delta}_{w,d,t} + \alpha}{\sum_t \sum_w \hat{\Delta}_{w,d,t} + \alpha}$$

$$\hat{\theta}_{w,t} = \frac{\sum_d \hat{\Delta}_{w,d,t} + \beta}{\sum_w \sum_d \hat{\Delta}_{w,d,t} + \beta}$$

Mixture models

We have now seen two different mixture models: soft k-means and topic models

Mixture models

We have now seen two different mixture models: soft k-means and topic models

Two inference procedures:

- Exact Inference with Maximum Likelihood using the EM algorithm
- Approximate Inference using Gibbs Sampling

Mixture models

We have now seen two different mixture models: soft k-means and topic models

Two inference procedures:

- Exact Inference with Maximum Likelihood using the EM algorithm
- Approximate Inference using Gibbs Sampling

Next, we will go back to Maximum Likelihood but learn about Approximate Inference using Variational Methods