

1. Introduction
2. Model Definition
3. Parameter Estimation and Inference
4. Example Output and Simulation
5. References



As more information becomes available, it becomes more difficult to find and discover what we need.

We need tools to help us organize, search and understand these vast amount of information.

Topic modeling provides methods for automatically organizing, understanding, searching, and summarizing large electronic archives:

1. Discover the hidden themes in the collection
2. Annotate the documents according to these themes
3. Use annotations to organize, summarize, search, and form predictions

1. Introduction

Today, the large collection of data calls for **unsupervised** probabilistic models.

Example Applications:

1. Summarizing Collections of Images



SKY WATER TREE
MOUNTAIN PEOPLE



SCOTLAND WATER
FLOWER HILLS TREE

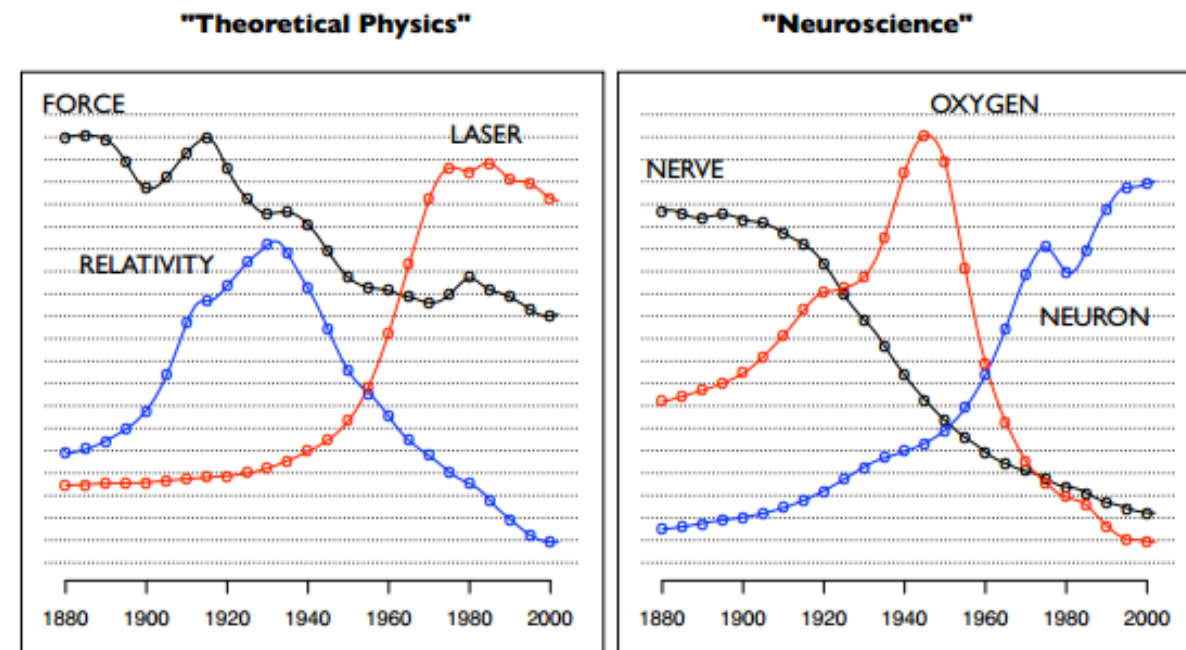


FISH WATER OCEAN
TREE CORAL



PEOPLE MARKET PATTERN
TEXTILE DISPLAY

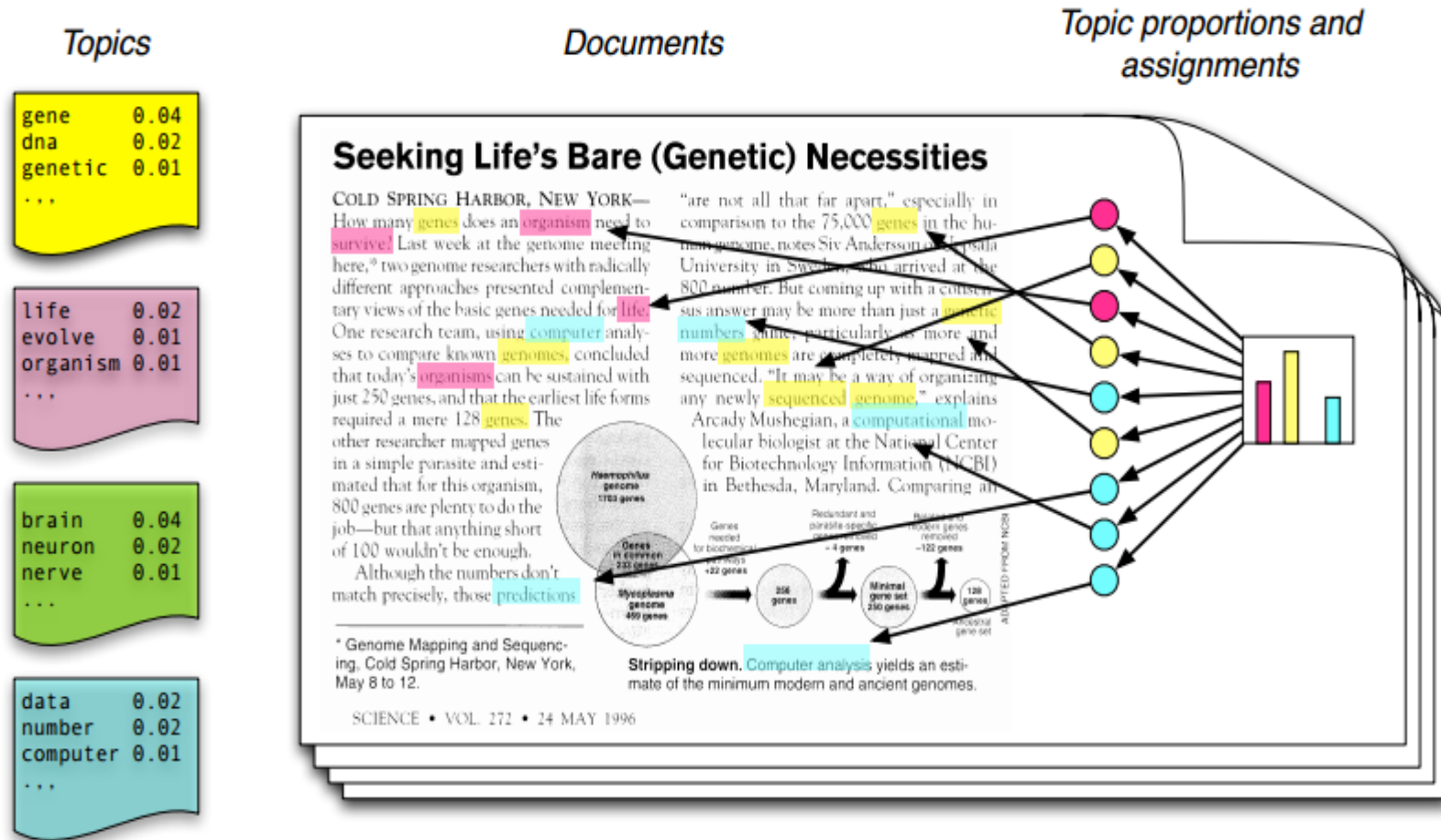
2. Evolution of Pervasive Topics by Time



Some Assumptions:

- We have a set of documents D_1, D_2, \dots, D_D .
- Each document is just a collection of words or a “bag of words”. Thus, the order of the words and the grammatical role of the words (subject, object, verbs, ...) are **not** considered in the model.
- Words like am/is/are/of/a/the/but/... can be eliminated from the documents as a preprocessing step since they don't carry any information about the “topics”.
- In fact, we can eliminate words that occur in at least %80 ~ %90 of the documents!
- Each document is composed of N “important” or “effective” words, and we want K topics.

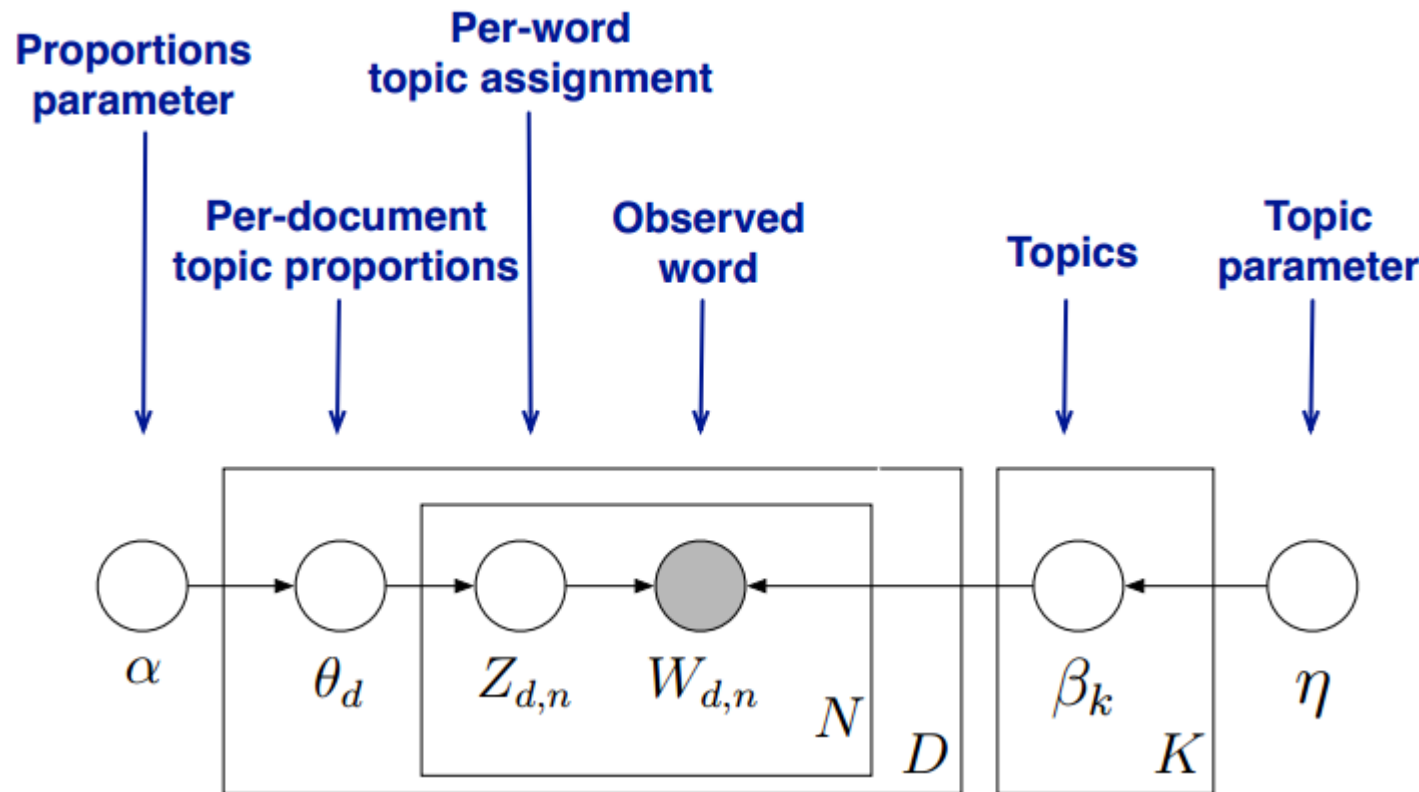
2. Model Definition



- Each **topic** is a distribution over words
- Each **document** is a mixture of corpus-wide topics
- Each **word** is drawn from one of these topics
- We only observe the words within the documents and the other structure are **hidden variables**.

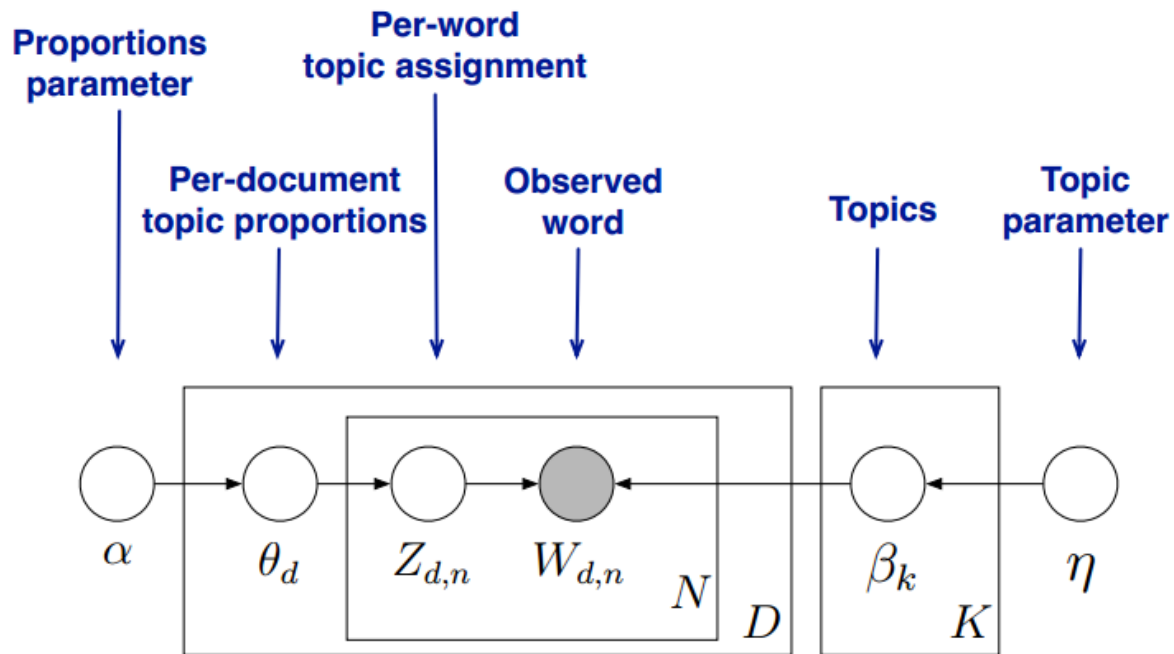
2. Model Definition

Our goal is to **infer** or **estimate** the hidden variables, i.e. computing their distribution conditioned on the documents. $\longrightarrow p(\text{topics}, \text{proportions}, \text{assignments} \mid \text{documents})$



- Nodes are RVs; edges indicate dependence.
- Shaded nodes are observed, and unshaded nodes are hidden.
- Plates indicate replicated variables.

2. Model Definition



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

1) Draw each topic $\beta_i \sim \text{Dir}(\eta)$ for $i = 1, \dots, K$

2) For each document:

First, Draw topic proportions $\theta_d \sim \text{Dir}(\alpha)$

For each word within the document:

a) Draw $Z_{d,n} \sim \text{Multi}(\theta_d)$

b) Draw $W_{d,n} \sim \text{Multi}(\beta_{z_{d,n}})$

This joint distribution defines a posterior $p(\theta, z, \beta | w)$.

From a collection of documents we have to infer:

1. Per-word topic assignment $z_{d,n}$.
2. Per-document topic proportions θ_d .
3. Per-corpus topic distributions β_k .

Then use posterior expectations ($E\{\beta|w\}$ for the corpus, $E\{\theta_d|w\}$ for each document) to perform the task at hand: information retrieval, document similarity, exploration, and others.

Formal definition of the model:

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

$$(\beta_d | \eta) \sim \text{Dir}(\beta)$$

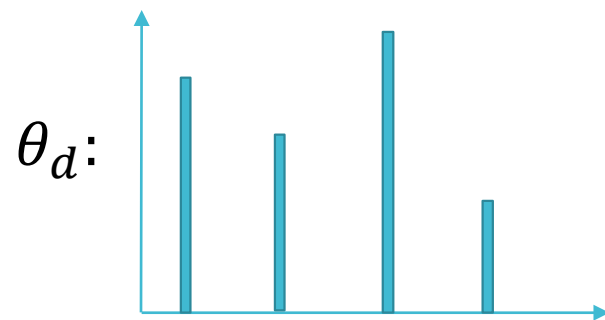
$$(\theta_d | \alpha) \sim \text{Dir}(\alpha)$$

$$Z_{d,n} \sim \text{Multi}(\theta_d)$$

$$W_{d,n} \sim \text{Multi}(\beta_{z_{d,n}})$$

$$p(z_{d,n} | \theta_d) = \theta_{d,z_{d,n}}$$

$$p(w_{d,n} | z_{d,n}, \beta_{1:K}) = \beta_{z_{d,n}, w_{d,n}}$$



β :

		Word probabilities for each topic		
Topics	Topic 1			
	Topic 2			
	Topic 3			

Review of Multinomial and Dirichlet distributions:

1. Multinomial:

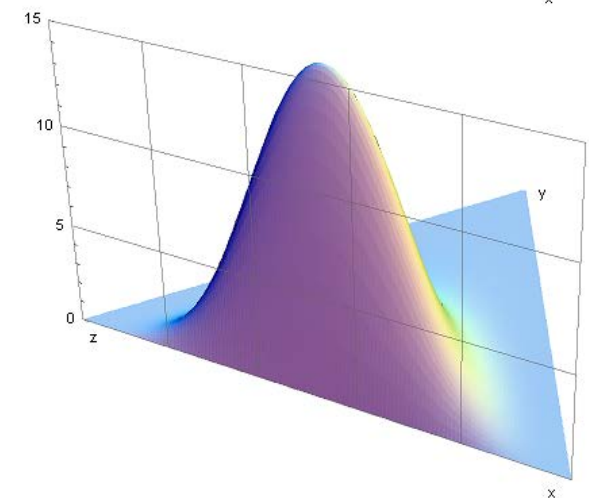
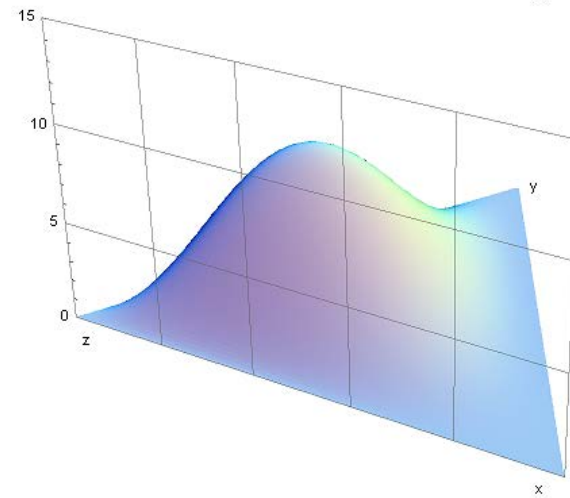
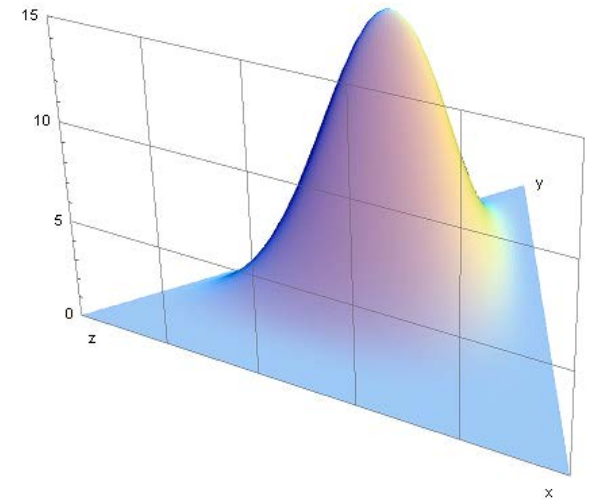
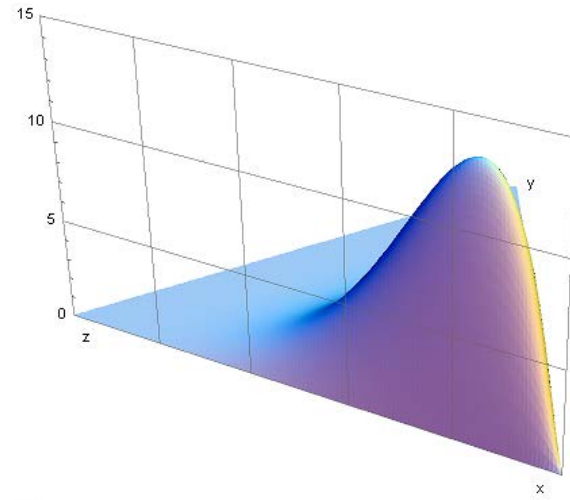
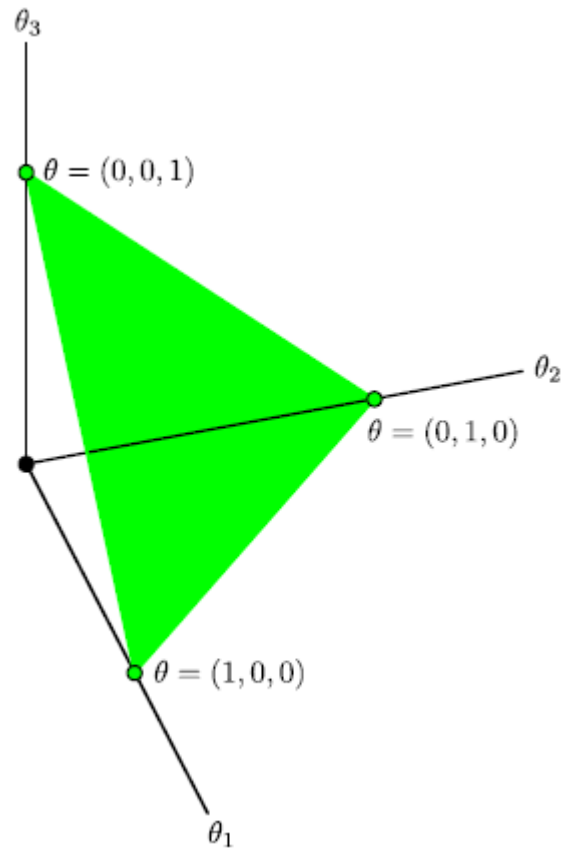
$$P(X_1 = x_1, \dots, X_K = x_K) = \frac{n!}{x_1! \dots x_K!} p_1^{x_1} \dots p_K^{x_K} \quad X_i \in \{0, \dots, n\} \quad \sum_{i=1}^K X_i = n$$

2. Dirichlet: Good for modeling a distribution over distributions

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^K \alpha_i)}{\prod_{i=1}^K \Gamma(\alpha_i)} \theta_1^{\alpha_1-1} \dots \theta_K^{\alpha_K-1} \quad \alpha = k - \text{dimensional vector} \quad \alpha_i > 0$$

variable θ can take values in the $(k - 1)$ simplex: $\theta_i > 0$ and $\sum_{i=1}^K \theta_i = 1$

2. Model Definition

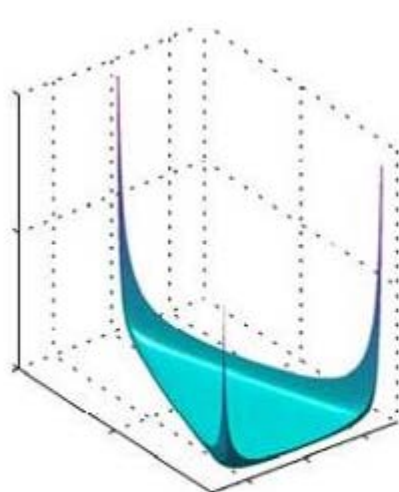


2. Model Definition

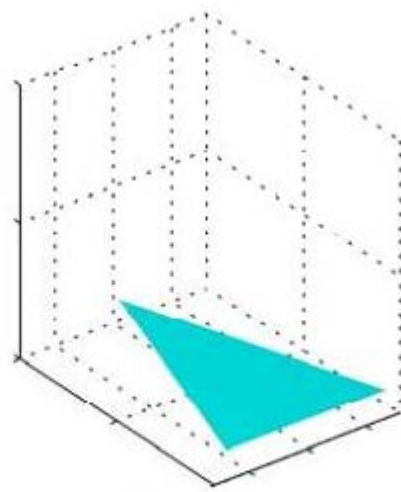
Role of parameter $\vec{\alpha} = (\alpha_1, \dots, \alpha_K)$:

$$E\{\theta_i | \alpha\} = \frac{\alpha_i}{\sum \alpha_i}$$

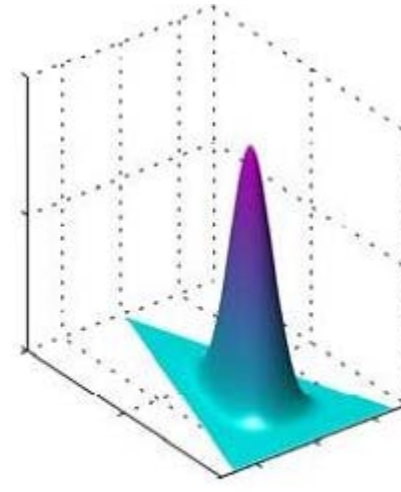
Note that here we are working with symmetric (exchangeable) Dirichlet distributions meaning $\alpha_1 = \dots = \alpha_K$



$\{\alpha_k\} = 0.1$



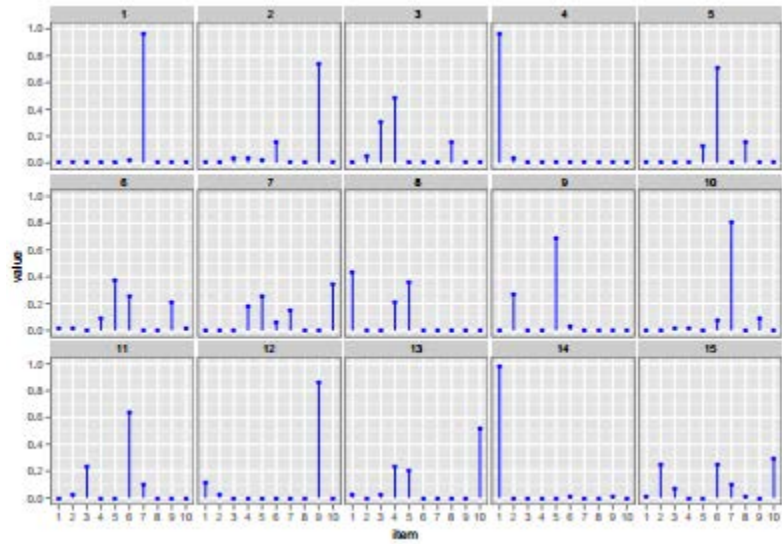
$\{\alpha_k\} = 1$



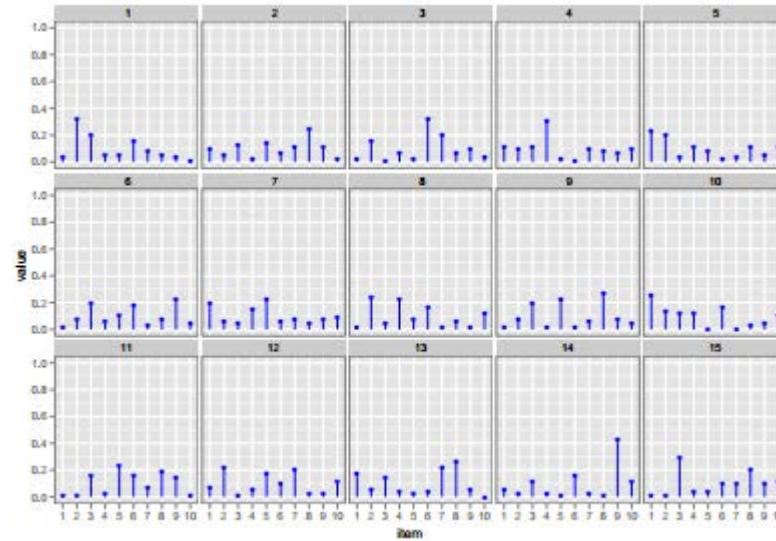
$\{\alpha_k\} = 10$

2. Model Definition

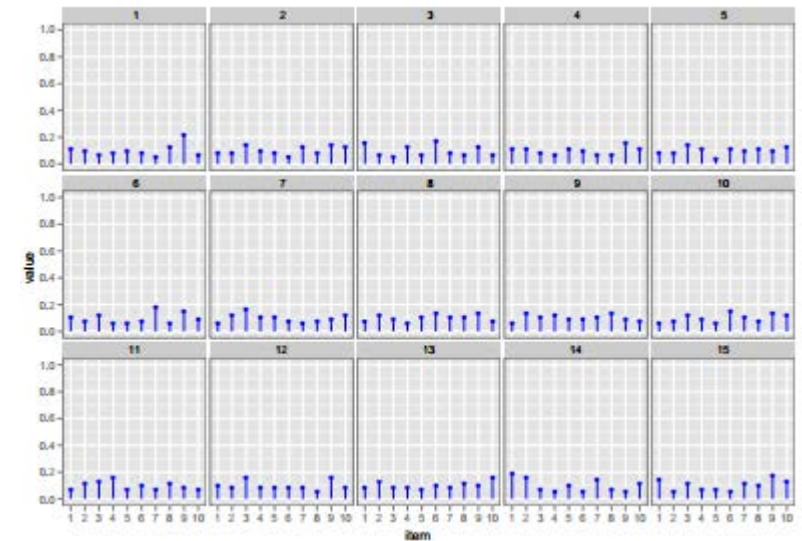
$$\alpha_i = 0.1$$



$$\alpha_i = 1$$



$$\alpha_i = 10$$



As mentioned earlier, from a set of N_d documents and the observed words within each document, we want to infer the posterior distribution $p(\theta, z, \beta | w)$ (Bayesian Inference)

There are many approximate posterior inference algorithms for this! We will briefly review Gibbs sampling here as an example.

Gibbs Sampling for Bayesian Inference:

Denote ϕ as the collection of model parameters and X as the observed data. For Bayesian Inference:

$$p(\phi|X) = \frac{p(X|\phi)p(\phi)}{p(X)} = \frac{p(X|\phi)p(\phi)}{\int p(X|\phi)p(\phi)d\phi}$$

Computing the integral in the denominator is impractical \longrightarrow Gibbs Sampling

Simple Gibbs Sampling:

Suppose you wish to sample $\theta_1, \theta_2 \sim p(\theta_1, \theta_2)$ but cannot use direct simulation or some other methods.

But, you can sample from $p(\theta_1|\theta_2)$ and $p(\theta_2|\theta_1)$. Then you can use Gibbs sampling.

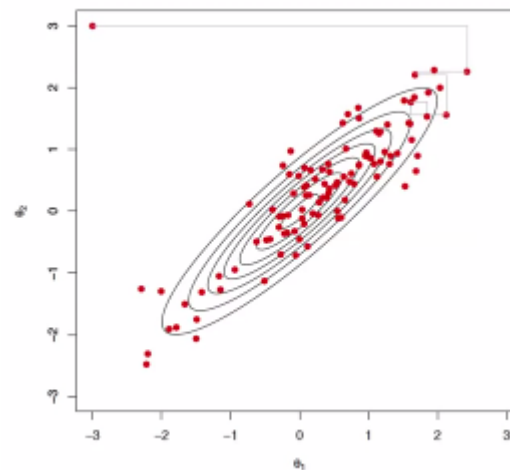
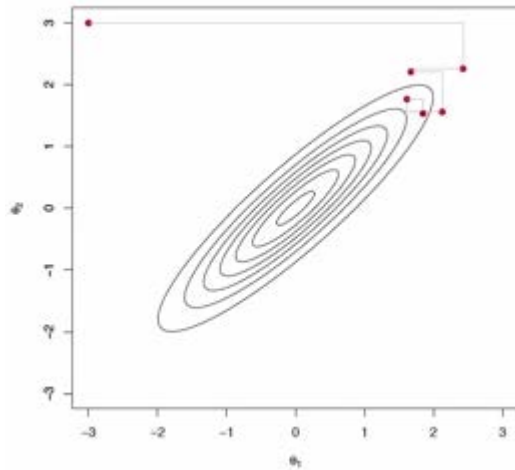
Algorithm:

1. Initialize $(\theta_1^{(0)}, \theta_2^{(0)})$
2. Repeat the following steps consecutively to compute $(\theta_1^{(j)}, \theta_2^{(j)})$:
 - a) Sample $\theta_1^{(j)} \sim p(\theta_1 | \theta_2^{(j-1)})$
 - b) Sample $\theta_2^{(j)} \sim p(\theta_2 | \theta_1^{(j)})$

Theorem: $\theta^{(j)} = (\theta_1^{(j)}, \theta_2^{(j)})^T$ converges in distribution to $\theta = (\theta_1, \theta_2)^T \sim p(\theta_1, \theta_2)$

Example: Bivariate Normal $\theta \sim N_2(0, \Sigma)$ where $\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$

We can show that $\theta_1 | \theta_2 \sim N(\rho\theta_2, 1 - \rho^2)$ and $\theta_2 | \theta_1 \sim N(\rho\theta_1, 1 - \rho^2)$



Applying this technique to estimating $p(\theta, z|w, \beta)$, we can form a simple Gibbs sampler:

1. $p(\theta|z_{1:N}, w_{1:N}) = Dir(\alpha + n(z_{1:N}))$ where $n()$ is a vector of the count of each topic (this is because of the choice of conjugate priors!)
2. $p(z_i|z_{-i}, \theta, w_{1:N}) \propto p(z_i|\theta)p(w_i|\beta_{1:K}, z_i)$

We have assumed $\beta_{1:K}$ is fixed in the above iterations!!! We can use a more complex Gibbs sampling to infer $\beta_{1:K}$ as well, or we can use other more efficient methods like the mean-field variational inference!

Now you have samples from the inferred posterior distribution for Bayesian Inference! Enjoy!

Text Mining package in R \longrightarrow `library("tm")`

Topic Models library in R \longrightarrow `library("topicmodels")`

We can perform preprocessing steps using function `tm_map` in the `tm` package such as removing unimportant words (stopwords, ...)

LDA function in the “`topicmodels`” package can fit the LDA model for a specific number of topics K .

R > `LDA(x, k, method = "VEM", control = NULL, model = NULL, ...)`

4.Example Output and Simulation

Data: Collection of Science from 1990-2000

17K documents

11M words

20K unique terms (redundant words removed)

Model: 100-topic LDA model using variational inference

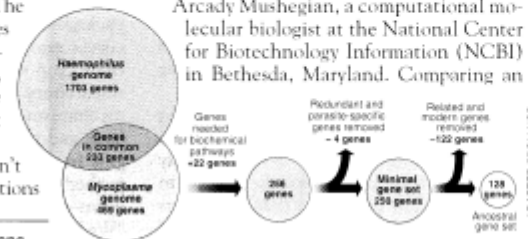
Seeking Life's Bare (Genetic) Necessities

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

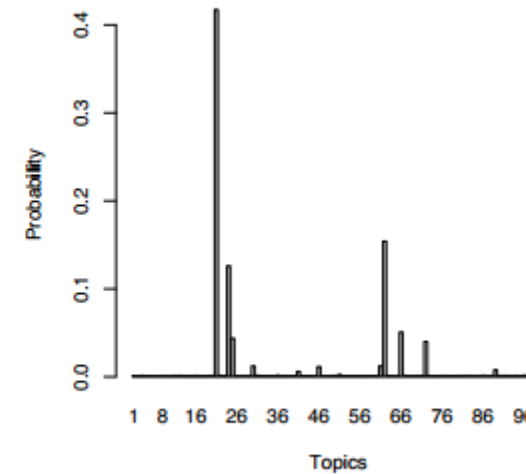
* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

"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



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

SCIENCE • VOL. 272 • 24 MAY 1996



Most probable words in four of the topics:

	Topic 1	Topic 2	Topic 3	Topic 4
Probability ↑	human	evolution	disease	computer
	genome	evolutionary	host	models
	dna	species	bacteria	information
	genetic	organisms	diseases	data
	genes	life	resistance	computers
	sequence	origin	bacterial	system
	gene	biology	new	network
	molecular	groups	strains	systems
	sequencing	phylogenetic	control	model
	map	living	infectious	parallel
	information	diversity	malaria	methods
	genetics	group	parasite	networks
	mapping	new	parasites	software
	project	two	united	new
	sequences	common	tuberculosis	simulations

- [1] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." *the Journal of machine Learning research* 3 (2003): 993-1022.
- [2] Homepage of David Blei, Associate Professor of Computer Science at Princeton University:
<http://www.cs.princeton.edu/~blei/topicmodeling.html>
- [3] Video Lectures of David Blei on videolectures.net:
http://videolectures.net/mlssoguk_blei_tm/

Questions?!