Computational Political Science

Session 8

David Broska
Zeppelin University
March 23, 2021

Outline for today

1. **Building blocks**
    - Probability distributions
2. **Latent Dirichlet Allocation (LDA)**
    - Estimation intuition
    - Results
3. **Extensions**
    - Correlated, dynamic, and structural topic model
4. **Coding examples**
    - Uncovering themes in New York Times articles with LDA
    - Using the correlated topic model for Newton's Principia
    - Structural topic model: Tweets of members of congress
5. **Coding exercise**
    - Themes in Trump's tweets using LDA

Course schedule

Session	Date	Topic	Assignment	Due date
1	Feb 02	Overview and key concepts	-	-
2	Feb 09	Preprocessing and descriptive statistics	Formative	Feb 22 23:59:59
3	Feb 16	Dictionary methods	-	-
4	Feb 23	Machine learning for texts: Classification I	Summative 1	Mar 08 23:59:59
5	Mar 02	Machine learning for texts: Classification II	-	-
6	Mar 09	Supervised and unsupervised scaling	Summative 2	Mar 22 23:59:59
7	Mar 16	Similarity and clustering	-	-
8	Mar 23	*Topic models*	Summative 3	Apr 12 23:59:59
-	-	*Break*	-	-
9	Apr 13	Retrieving data from the web	-	-
10	Apr 20	Published applications	-	-
11	Apr 27	Project Presentations	-	-

Mixed membership models

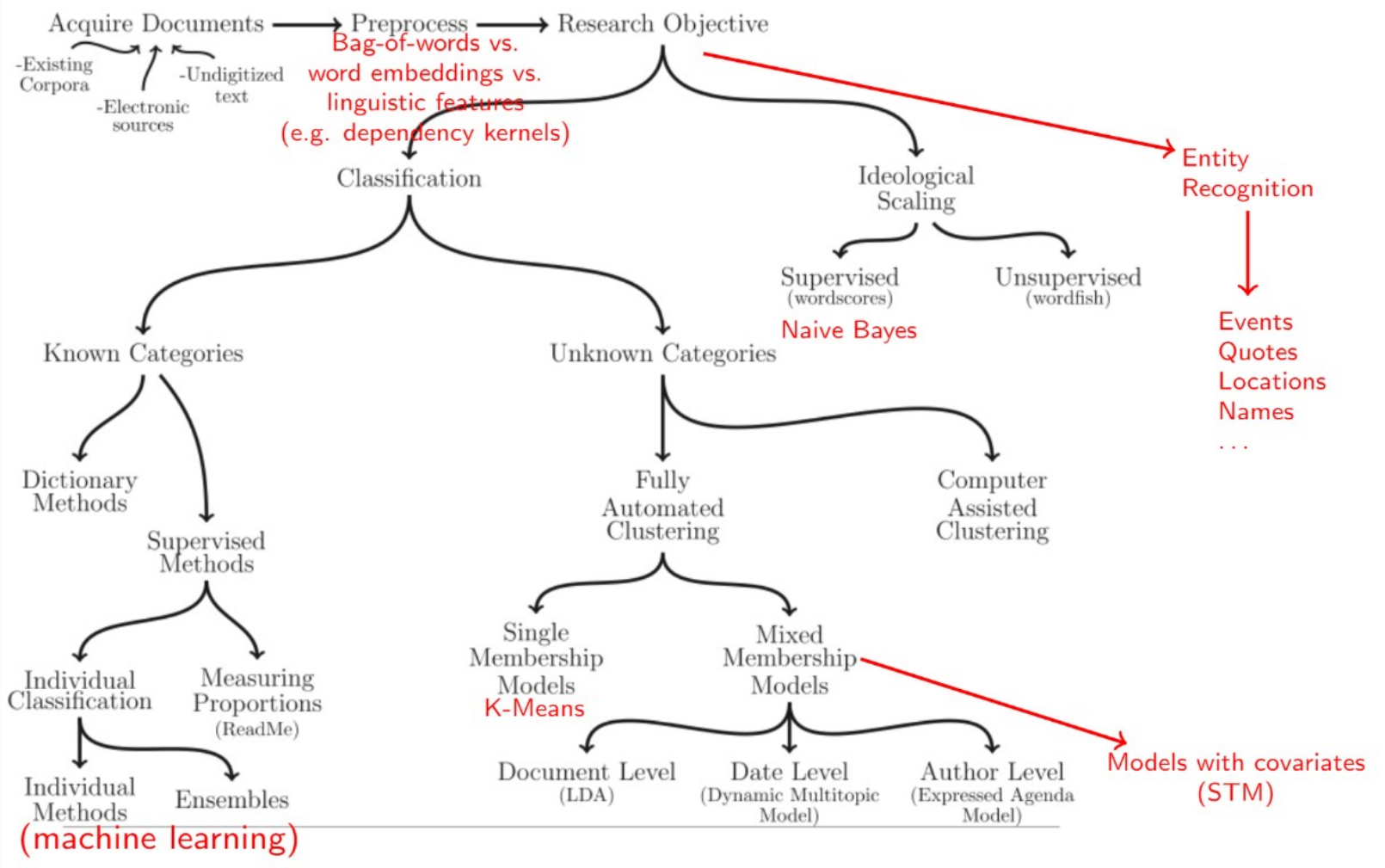


Fig. 1 in Grimmer and Stuart (2013)

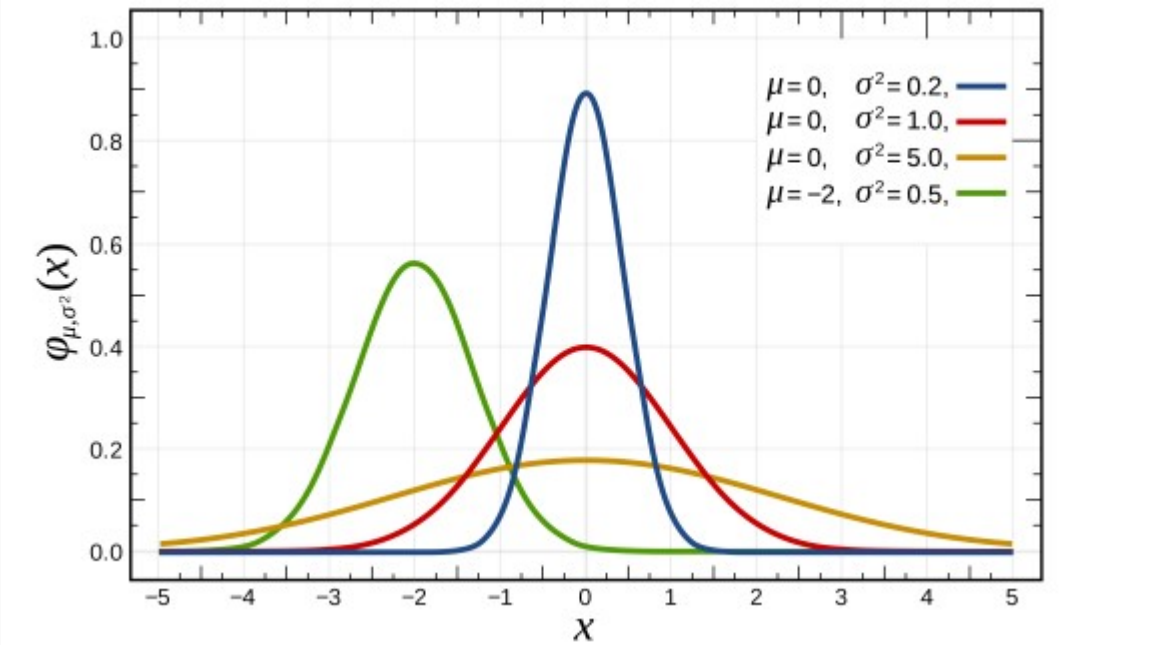
Topic models: motivation

- Topic models offer an automated procedure to discover the main "themes" in an unstructured corpus of texts
- Can be used to understand and organize large collections of documents according to the discovered themes (e.g. New York Times articles from 2019-2020)

- Unsupervised learning: topic models require no labeled data - only a set of documents
- Mixture model: Documents can contain multiple topics; words can belong to multiple topics (as opposed to k-means for example)

Distributions review

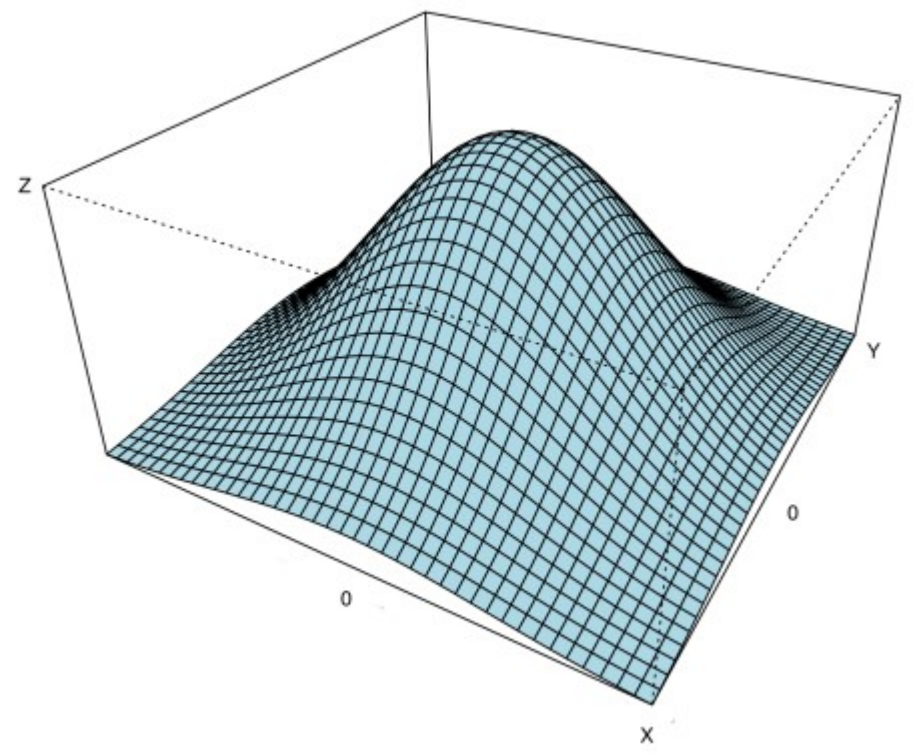
Univariate probability density function



Different parameter values (e.g. μ and σ for the normal distribution) change the distributions' shape

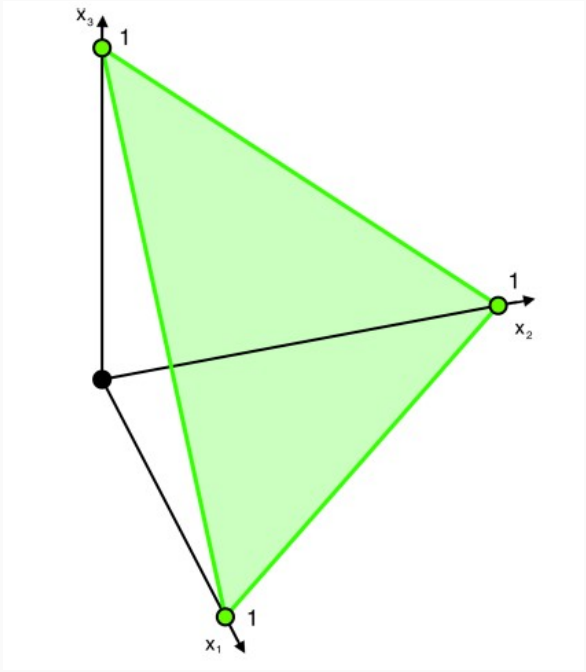
The notation $\mathrm{x} \sim \mathrm{N}(0, 1)$ denotes to sample or draw x from a standard normal distribution. This draw could e.g. return x = −1.124.

Multivariate probability density function



A draw $\mathbf{a} \sim \mathbf{N}(\mu, \Sigma)$ from this multivariate normal distribution could e.g. return a = (−0.12, 1.2)

Standard simplex

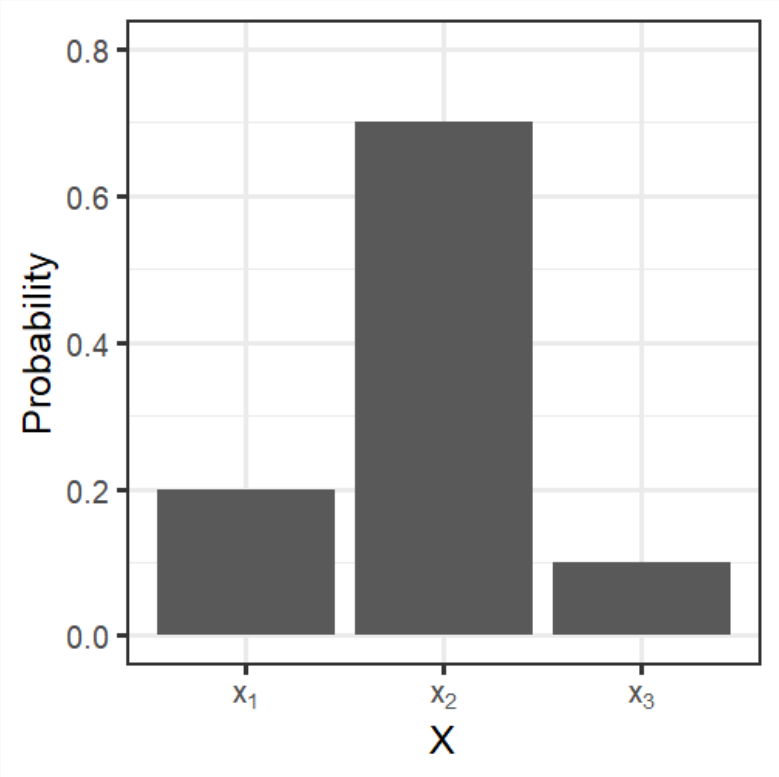


Graphical intuition of a standard simplex

A point on the triangle is x_1, x_2, x_3, with $x_1, x_2, x_3 \in [0, 1]$ and $x_1 + x_2 + x_3 = 1$. For example, (0.05, 0.8, 0.15)

Generalizes to higher dimensions with $x_1, \ldots, x_k \in [0, 1]$ and $\sum x_k = 1$

Key distribution 1: Multinomial



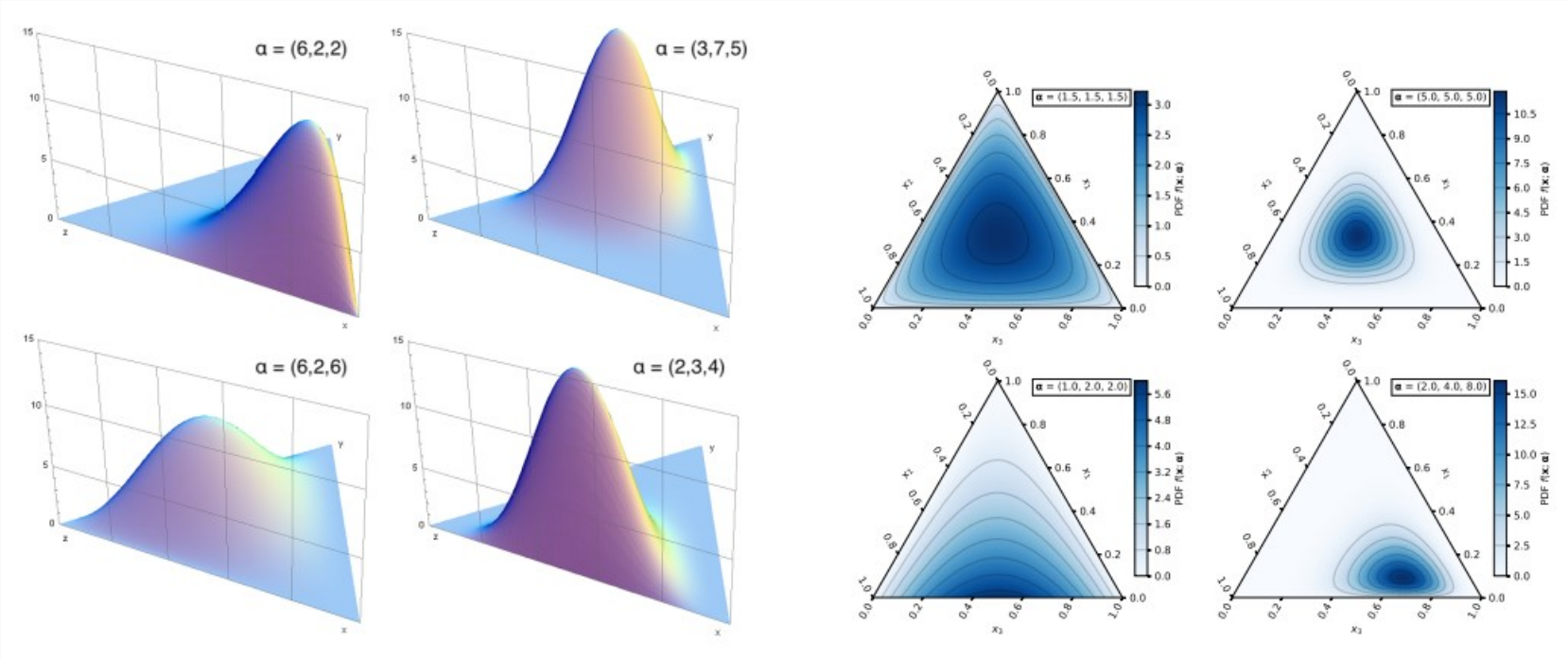
X is a discrete random variable with three categories x_1, x_2, and x_3

The figure shows a multinomial distribution with probabilities [0.2, 0.7, 0.1].

A draw X ~ Multinomial([0.2, 0.7, 0.1]) is most likely to return the second category

Key distribution 2: Dirichlet distribution

The Dirichlet distribution is a probability distribution over a simplex



We can draw a multinomial distribution from a Dirichlet distribution

- A draw b ~ Dir(α) from this distribution could e.g. return b = (0.2, 0.7, 0.1)
- Hence, we can think of the draw from a Dirichlet distribution being itself a multinomial distribution (previous slide)

Topic models

Topic models: main idea

Intuition

Topic models reverse-engineer the writing process of documents with a sequence of probabilistic steps

Generative probabilistic model

Topic models models assume a *fixed number of topics* which are *distributions over words*, and each document has a *distribution over topics*

Each document is "written" by randomly sampling a topic from a document-topic distribution, then sampling a word from that topic distribution. Repeat that process until you have all words in the text.

Latent Dirichlet Allocation (LDA) by Blei et al. (2003) is a common approach but there are extensions such as Structural Topic models (STM) by Roberts et al (2014)

Latent Dirichlet Allocation (LDA)

Figure 1. The intuitions behind latent Dirichlet allocation. We assume that some number of "topics," which are distributions over words, exist for the whole collection (far left). Each document is assumed to be generated as follows. First choose a distribution over the topics (the histogram at right); then, for each word, choose a topic assignment (the colored coins) and choose the word from the corresponding topic. The topics and topic assignments in this figure are illustrative—they are not fit from real data. See Figure 2 for topics fit from data.

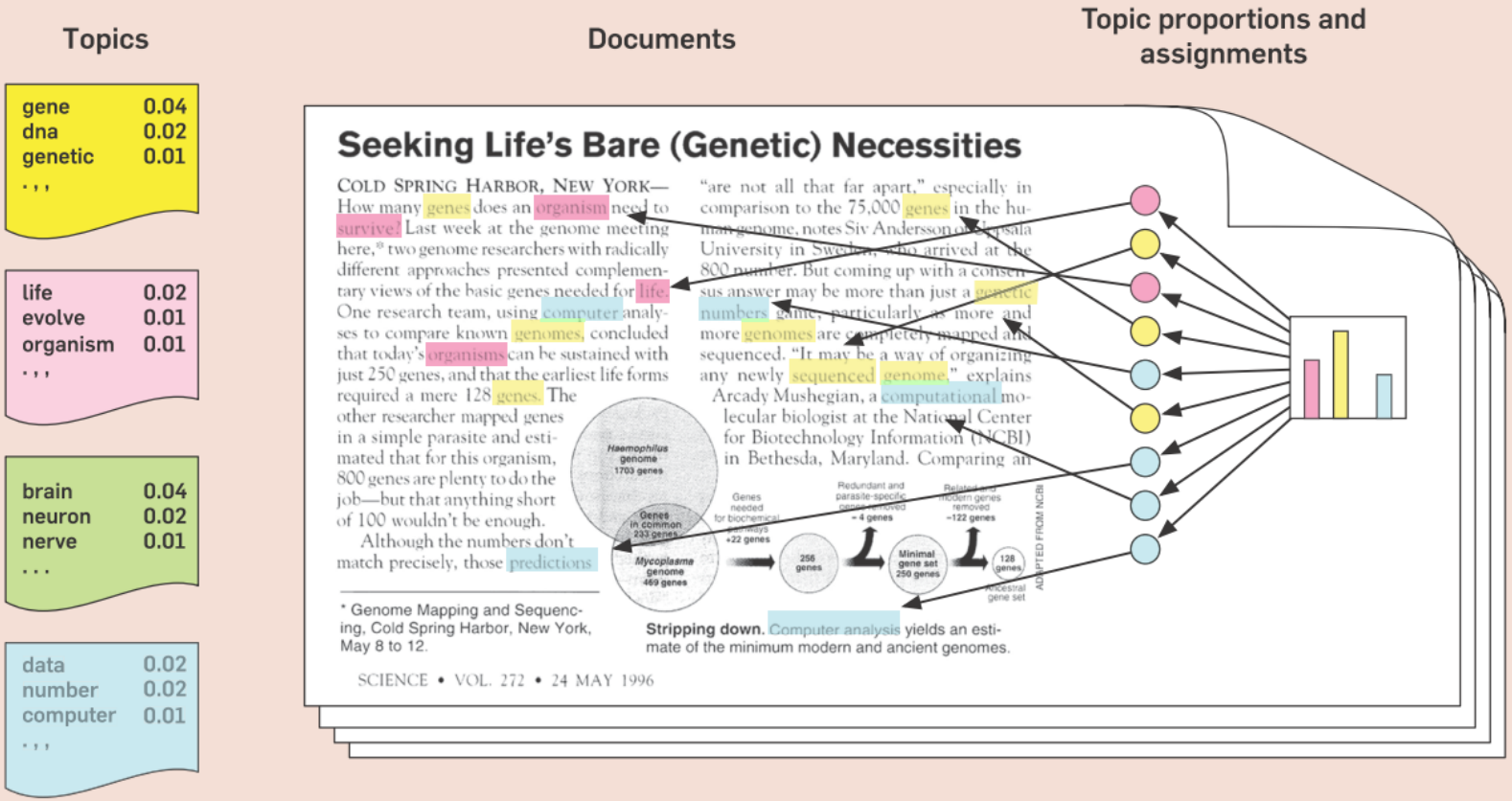
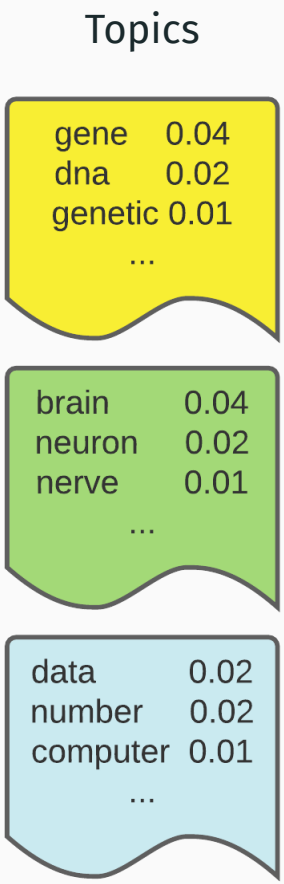


Figure by Blei (2012)

Topic model results I

From an input corpus and a predefined number of topics k, we get words related to topics

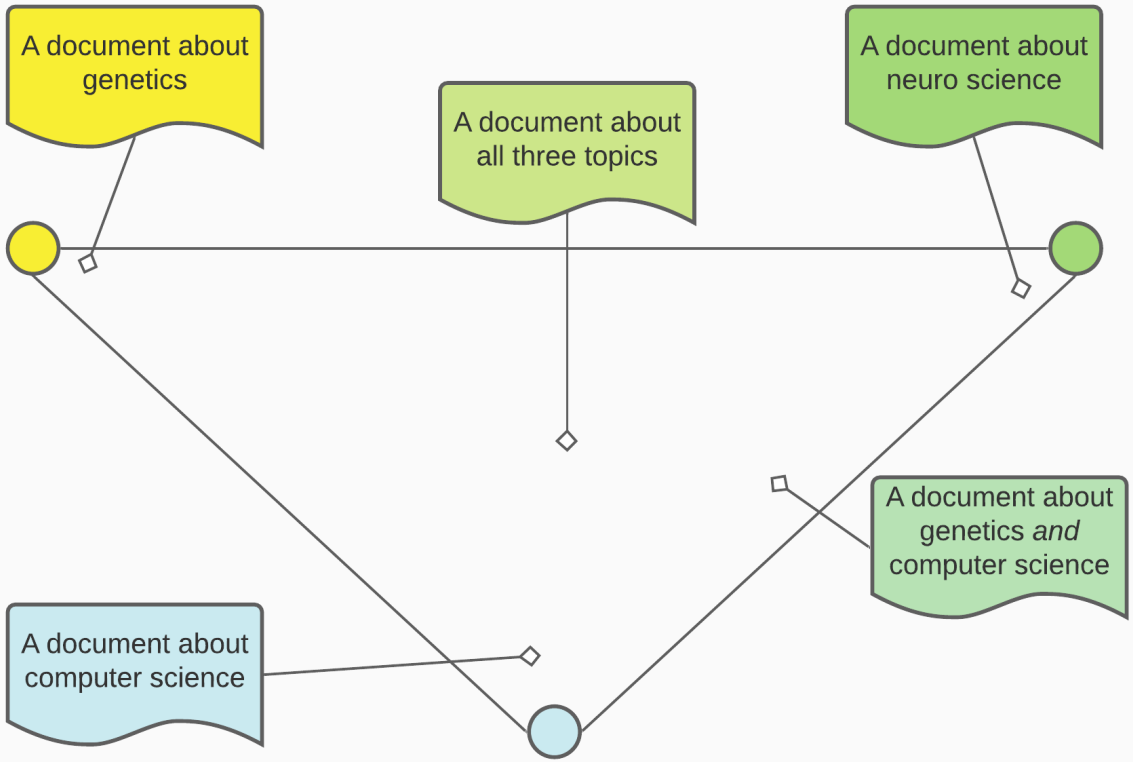


- Each topic is a multinomial distribution over words
- Each topic comes from a Dirichlet distribution
- The degree of association of words with a topic is quantified with a probability

Topic model results II

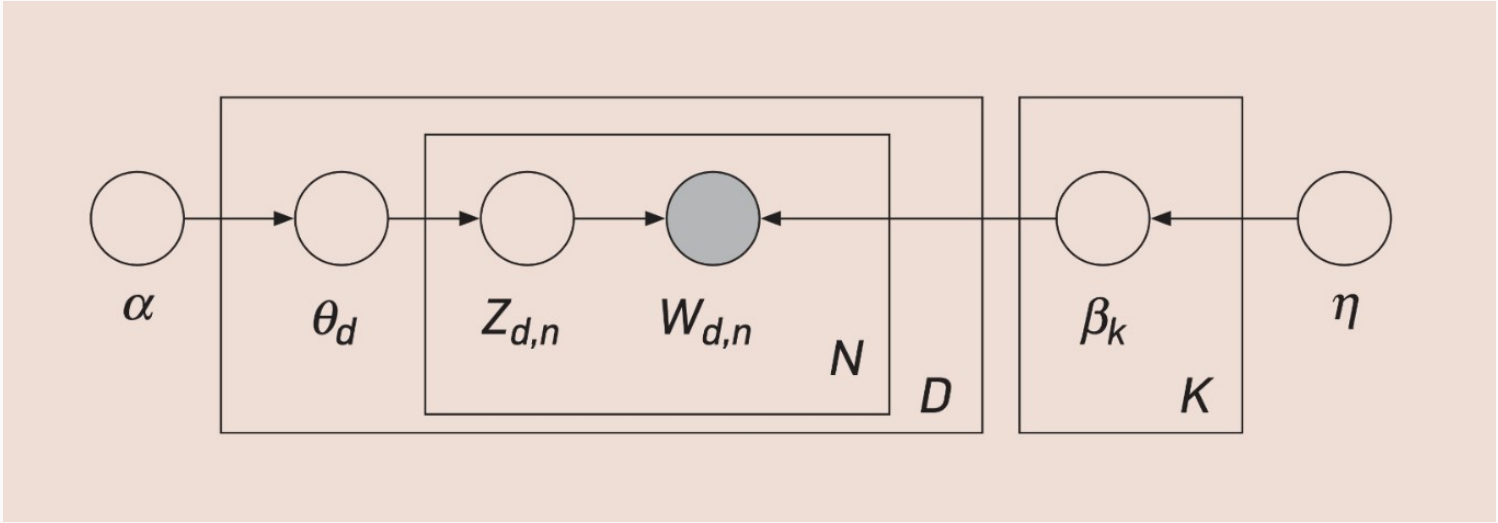
Each topic is multinomial distribution that is drawn from a Dirichlet distribution

LDA quantifies the extent to which a document belongs to a topic



Depiction of the association of 3 topics with documents in a simplex

LDA: plate notation



- For each topic $k \in \{1, ..., K\}$, draw a multinomial distribution β_k from a Dirichlet distribution with parameter η
- For each document $d \in \{1, ..., D\}$, draw a multinomial distribution θ_d from a Dirichlet distribution with parameter α
- For each word position in the document $n \in \{1, ..., N\}$, select a hidden topic assignment z_n from the multinomial distribution parametrized by θ
- Choose the observed word w_n from the distribution β_{z_n}

LDA: General case

Let's look more closely at how LDA assumes our document have been generated

- We consider a corpus of D document, each with $\mathrm{N_d}$ words
- For each document $\mathrm{d} \in 1, \ldots, \mathrm{D}$, draw its *topic shares* from a Dirichlet distribution $\theta_\mathrm{d} \sim \mathrm{Dir}(\alpha)$
- For each topic $\mathrm{k} \in 1, \ldots, \mathrm{K}$, draw its *word shares* from a second Dirichlet distribution $\beta_\mathrm{k} \sim \mathrm{Dir}(\eta)$
- Fill each document with words. For each document-word position d, n in $\mathrm{d} \in 1, \ldots, \mathrm{D}$ and $\mathrm{n} \in 1, \ldots, \mathrm{N_d}$
  - Choose the topic $\mathrm{z_{d,n}}$ of the document-word position by taking a draw from document d's topic distribution: $\mathrm{z_{d,n}} \sim \mathrm{Multinomial}(\theta_\mathrm{d})$
  - Choose the word $\mathrm{w_{d,n}}$ by taking a draw from the corresponding topic distribution: $\mathrm{w_{d,n}} \sim \mathrm{Multinomial}(\beta_{\mathrm{z_{d,n}}})$

LDA: stylized example

Let us first imagine a stylized corpus with only $D = 5$ documents, $N = 8$ words per document (and also 8 unique words in the corpus in total), and $K = 3$ topics

- For each of the 5 documents, draw its topic shares from a Dirichlet distribution $\theta_d \sim \text{Dir}(\alpha)$. For document 1 this could e.g. yield $\theta_1 = (0.1, 0.8, 0.1)$, i.e. document 1 consists predominantly of topic 2
- For each of the 3 topics, draw its word shares from a second Dirichlet distribution $\beta_k \sim \text{Dir}(\eta)$. For topic 2 this e.g. yield $\beta_2 = (0, 0, 0, 0.1, 0, 0, 0.4, 0.5)$, i.e. topic consists predominantly of words 7 and 8.
- Now it is possible to fill each document with words. For each document-word position d, n
    - Choose the topic $z_{d,n}$ of the document-word position by talking a draw from document d's topic distribution: $z_{d,n} \sim \text{Multinomial}(\theta_d)$
    - Choose the word $w_{d,n}$ by taking a draw from the corresponding topic distribution: $w_{d,n}$

Estimation output

Say we have a corpus with $\mathrm{D} = 1,000$ documents, a vocabulary of $\mathrm{V} = 10,000$ total unique words/n-grams, and choose $\mathrm{K} = 3$ topics. The final output after estimating the topic model could be

Topic shares for each document

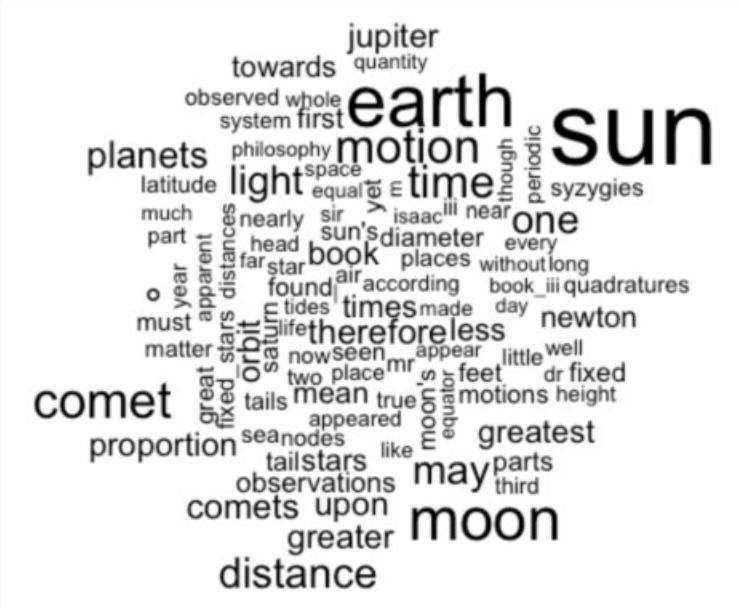
$$\theta = \underbrace{\begin{pmatrix} \theta_{1,1} & \theta_{1,2} & \theta_{1,3} \\ \theta_{2,1} & \theta_{2,2} & \theta_{2,3} \\ \dots & \dots & \dots \\ \theta_{1000,1} & \theta_{1000,2} & \theta_{1000,3} \end{pmatrix}}_{1000 \times 3} = \underbrace{\begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.3 & 0.1 & 0.6 \\ \dots & \dots & \dots \\ 0.1 & 0.8 & 0.1 \end{pmatrix}}_{1000 \times 3}$$

Word shares for each topic

$$\beta = \underbrace{\begin{pmatrix} \beta_{1,1} & \beta_{1,2} & \dots & \beta_{1,10000} \\ \beta_{2,1} & \beta_{2,2} & \dots & \beta_{2,10000} \\ \beta_{3,1} & \beta_{3,2} & \dots & \beta_{3,10000} \end{pmatrix}}_{3 \times 10000} = \underbrace{\begin{pmatrix} 0.04 & 0.01 & \dots & 0.0001 \\ 0.00002 & 0.001 & \dots & 0.05 \\ 0.00001 & 0.03 & \dots & 0.0001 \end{pmatrix}}_{3 \times 10000}$$

Topics

- What is referred to as a "topic" is a row vector in β with a probability for every word in the corpus to belong to that topic
- For example, the second topic is the second row vector $\beta_2 = [\beta_{2,1}\ \beta_{2,2} \ldots \beta_{2,10000}] = [0.00002\ 0.001 \ldots 0.05]$
- Put differently, a topic is a collection of word probabilities, names which summarizes their meaning are given by humans



- A topic us often visualized via a word cloud where the size of words corresponds to their frequency in the vector
- The figures shows a topic which could be termed "astronomy".

Estimation intuition

- Estimation the topic models is done in a Bayesian framework
- Our $\mathrm{Dir}(\alpha)$ and $\mathrm{Dir}(\eta)$ are the so called *prior distributions* of the θ_d and β_k
- With Bayes' theorem, with the help of our data, and the model, we update these prior distributions to obtain a so called *posterior distribution* for each θ_d and β_k
- The means of these posterior distributions are the rows in the above matrices commonly given by statistical packages and referred to as θ and β

Obtain posterior distributions

- Algorithms to obtain posterior distributions are not the focus of this lecture
- Common algorithm to obtain posterior distributions for the θ_d and β_k are from two main groups

    1. Sampling algorithms such as **Gibbs sampling** approximate the posterior distribution with an empirical distribution by drawing a sequence of samples in a way that ensure its limit distribution is the true posterior distribution
    2. **Variational methods** approximate the posterior distribution with a parametrized family of distributions. The error of this approximation is minimized and obtaining posterors thus becomes an optimization problem

See *Probabilistic Topic Models* by Blei (2012) for links to further resources

Extensions

Correlated topic model

- Blei and Lafferty (2005, 2007) developed the correlated topic model (CTM)
- It swaps the Dirichlet distribution of topic shares in documents (θ) with a logistic normal normal distribution
- The logistic normal distribution is also defined over a simplex, however, unlike the Dirichlet distribution, it also allows to model correlations (between topics) through a covariance matrix
- This can return estimates of correlations between the topics and can also lead to a better fit

Structural topic model

The structural topic model was introduced by Roberts et al. (2013)

Topic prevalence covariates

Topic proportions within documents can vary through covariates

E.g. social media posts by Republican politicians might have different topic proportions than those posted by Democrats

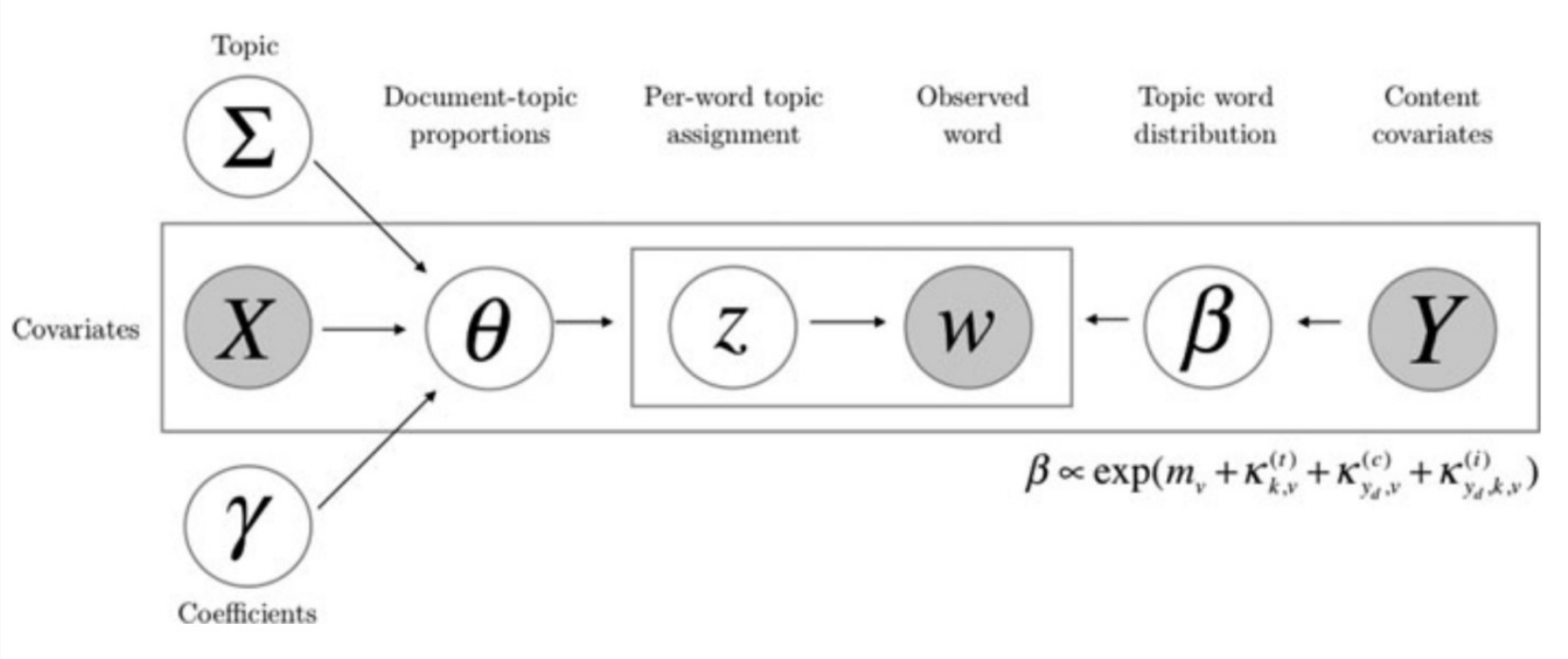
Topical content covariates

Word proportions within topics can vary through covariates

E.g. when talking about a health care topic, Republican politicians might use different words than Democrats

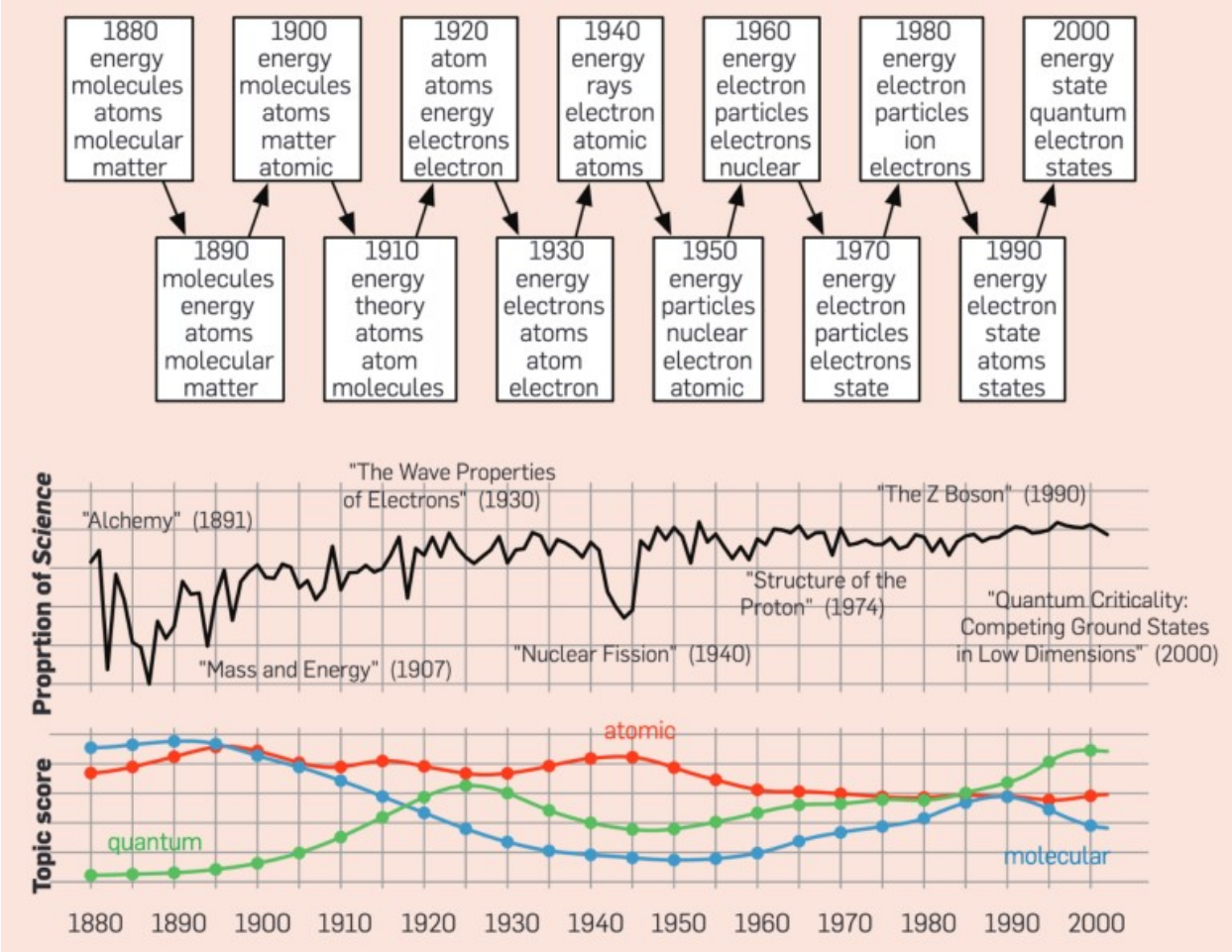
Without any covariates supplied, the model reduce to the correlated topic model

Structural topic model



From: “A Model of Text for Experimentation in the Social Sciences” by Roberts, Stewart, and Airoldi, 2016

Dynamic topic model



A physics topics from a dynamic topic model that was fit to articles in *Science* from 1880 to 2002 from by Blei (2012)

Selecting K and beyond

Challenge

In particular selecting the amount of topics K, but also parameters, covariates, and potentially initializations is a very challenging exercise without a single solution

What to do?

Researchers typically consult a combination of quantitative metrics and human judgment

Quantitative metrics

Held-out likelihood or perplexity

For some held-out documents, how likely would the model have generated/predicted these documents

Semantic coherence

For example, how likely do the most common words from a topic also co-occur in the same document?

Exclusivity

Do words with high probability in one topic have low probabilities in others?

Many automated metrics exist, see e.g. Grimmer and Stewart (2013), Mimno et al. (2011), Taddy (2012)

Quantitative metrics

According to Roberts et al. (2014), a semantically interpretable topic has two qualities:

1. It is **cohesive** in the sense that high-probability words for the topic tend to co-occur within documents

2. It is **exclusive** in the sense that the top words for that topic are unlikely to appear with in top words of other topics.

Semantic coherence and exclusivity, with many other quantitative metrics, are outputs of the function ‘searchK‘ in the ‘stm’ package

For a discussion of model selection for (structural) topic models (e.g. different choices of K, initializations, and covariates) and evaluation, see e.g. Section “3.4. Evaluate: Model selection and search” in the package vignette or the Section Model Specification and Selection in “Structural Topic Models for Open-Ended Survey Responses” by Roberts et al. (2014)

Takeaways

No quantitative metric can replace human judgement when selecting K or other model parameters, and evaluating the fit of a particular topic model more generally

- “The most effective method for assessing model fit is to carefully read documents that are closely associated with particular topics to verify that the semantic concept covered by the topic is reflected in the text.” from “A Model of Text for Experimentation in the Social Sciences” by Roberts et al. (2016)
- The ‘plotQuote‘ function in `stm` allows to plot documents highly associated with certain topics
- Key consideration: What is the goal of the current model?
- To generate coherent topics which describe themes? Combine careful human reading with quantitative metrics
- To use topic models e.g. for document embeddings to find similar documents? As input in a predictive model? Try to select K and other parameters such that they maximize whatever the objective function

Example

A carefully documented project with very good average topic coherence is e.g. “Leaders or Followers? Measuring Political Responsiveness in the U.S. Congress Using Social Media Data” by Barberá et al. (2014)

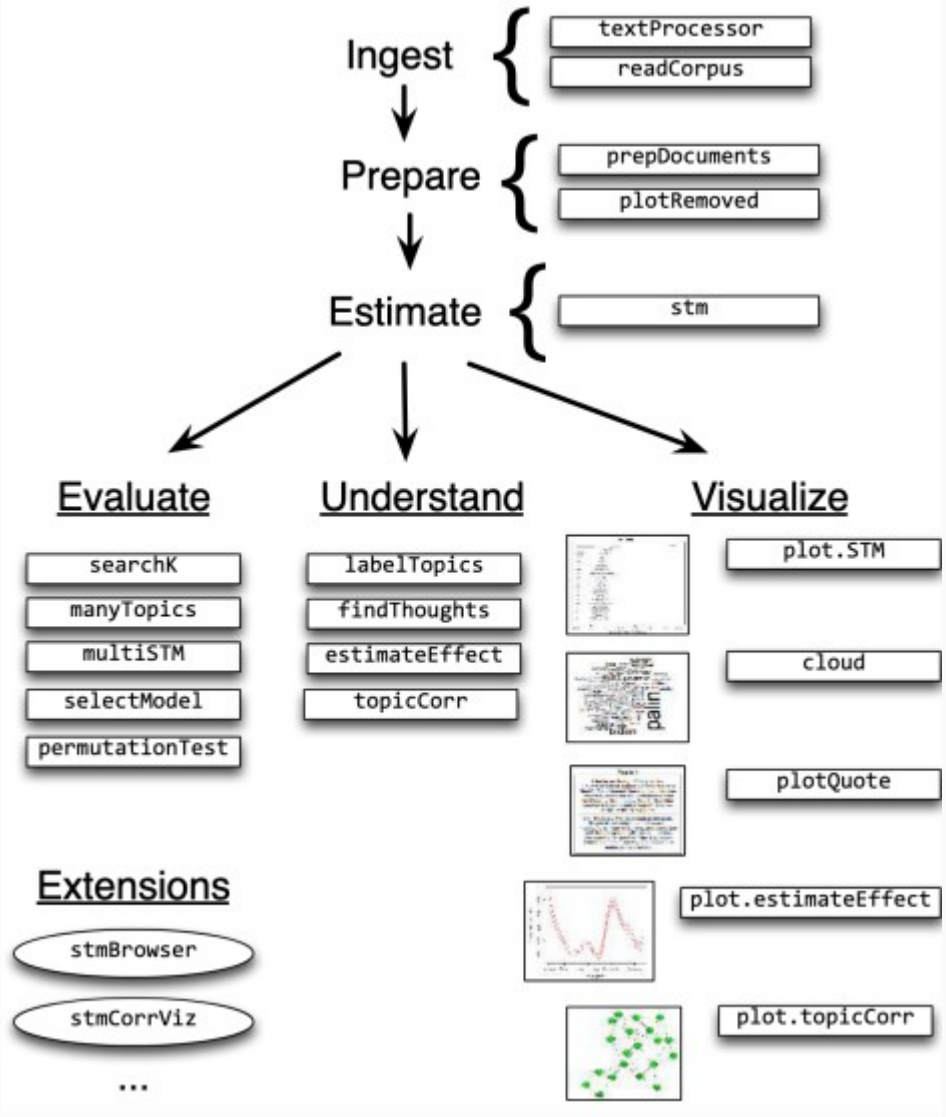
- Data: 651,116 tweets sent by US legislators from January 2013 to December 2014
- 2,920 documents = 730 days × 2 chambers × 2 parties
- Why aggregating? Applications that aggregate by author or day outperform tweet-level analyses (Hong and Davidson, 2010)
- Sidenote: For short texts, such as also e.g. survey responses, topic models such as sparse additive generative models (SAGE) might create more coherent topics than e.g. LDA (see e.g. Bauer et al, Political Behavior, 2017)
- K = 100 topics
- Validation: http://j.mp/lda-congress-demo

Implementations in R

For an implementation of the LDA and CTM, see e.g. the package `topicmodels`

- We will focus on the `stm` package which offers a range of helpful functionalities
- Without added covariates, the `stm` function also estimates a standard CTM, with covariates a structural topic model
- Further helpful package to visualize topic models are `LDAvis` or, the stm-specific, `stminsights`

Functionalities of stm package



Before we move on

- If you are interested in using the Twitter API in the next weeks, please apply for a developer account
- Applications can be done via https://developer.twitter.com/en/apps and take some time to be processed
- After your application is approved, you can create an app within a project or a standalone app
- This will create a 1) key and a 2) key secret
- Afterwards you can create 3) an access token and 4) an access token secret

These four strings/characters will be used to access the API, e.g. via the package `rtweet`

Computer exercises