Lecture 22
Exploratory Text Analysis & Topic Models

Intro to NLP, CS585, Fall 2014

http://people.cs.umass.edu/~brenocon/inlp2014/

Brendan O'Connor

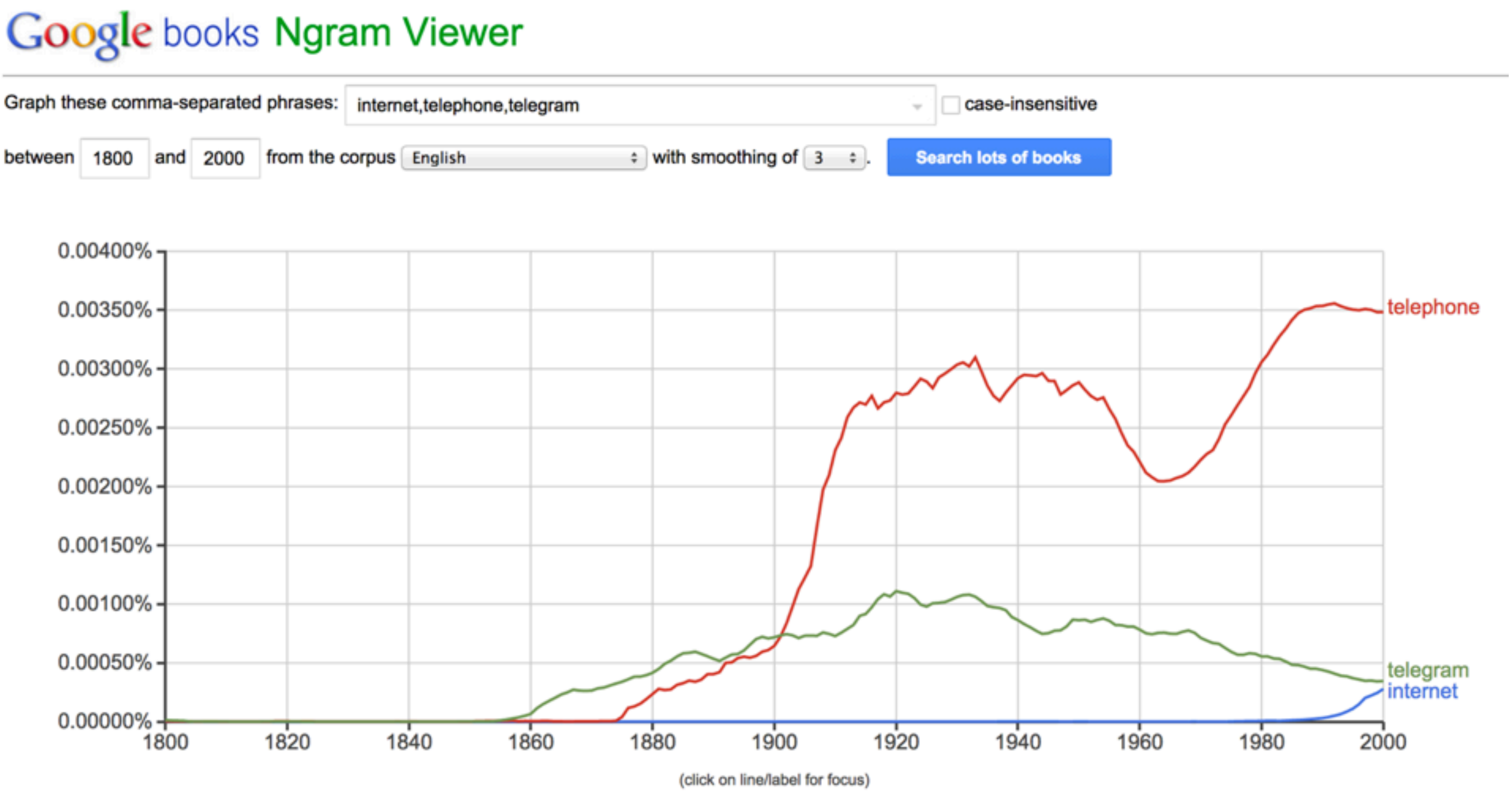
[Some slides borrowed from Michael Paul]

Text Corpus Exploration

- You have a big pile of text documents. What's going on inside?
  - Comparisons to document covariates
  - Clustering and topic models

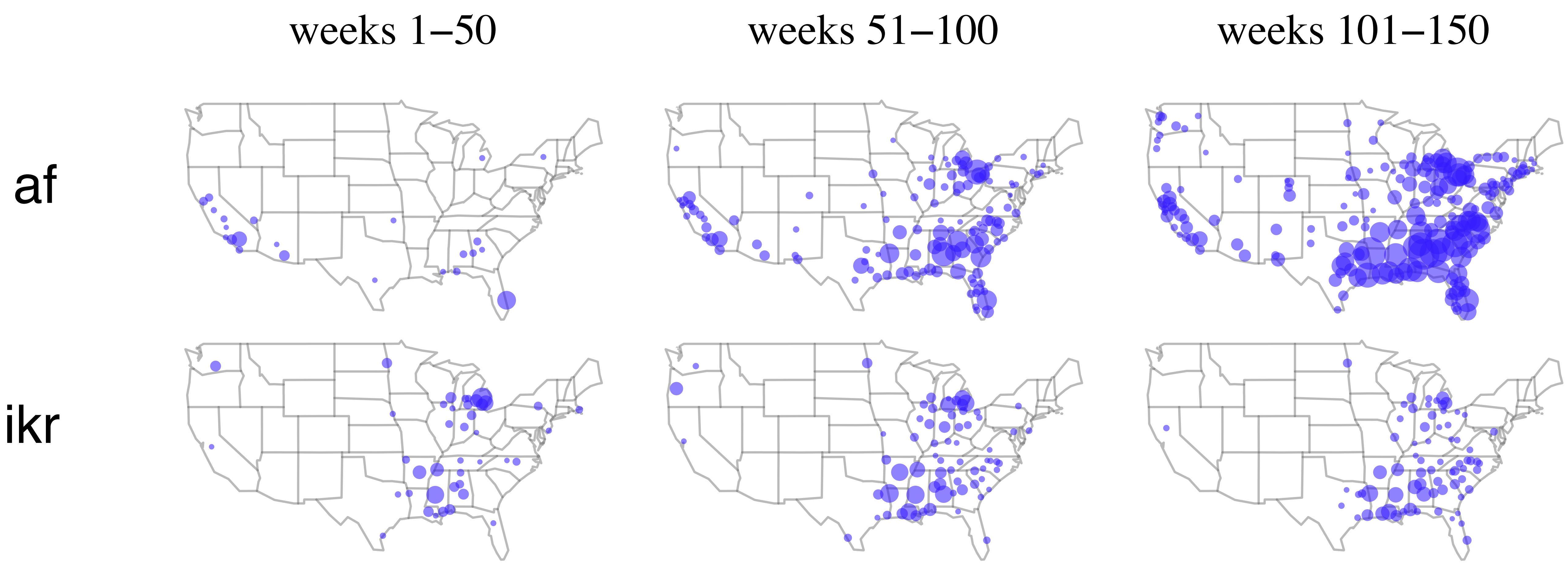
Word-covariate analysis

- Documents have metadata. How do individual words correlate?
- Words against time



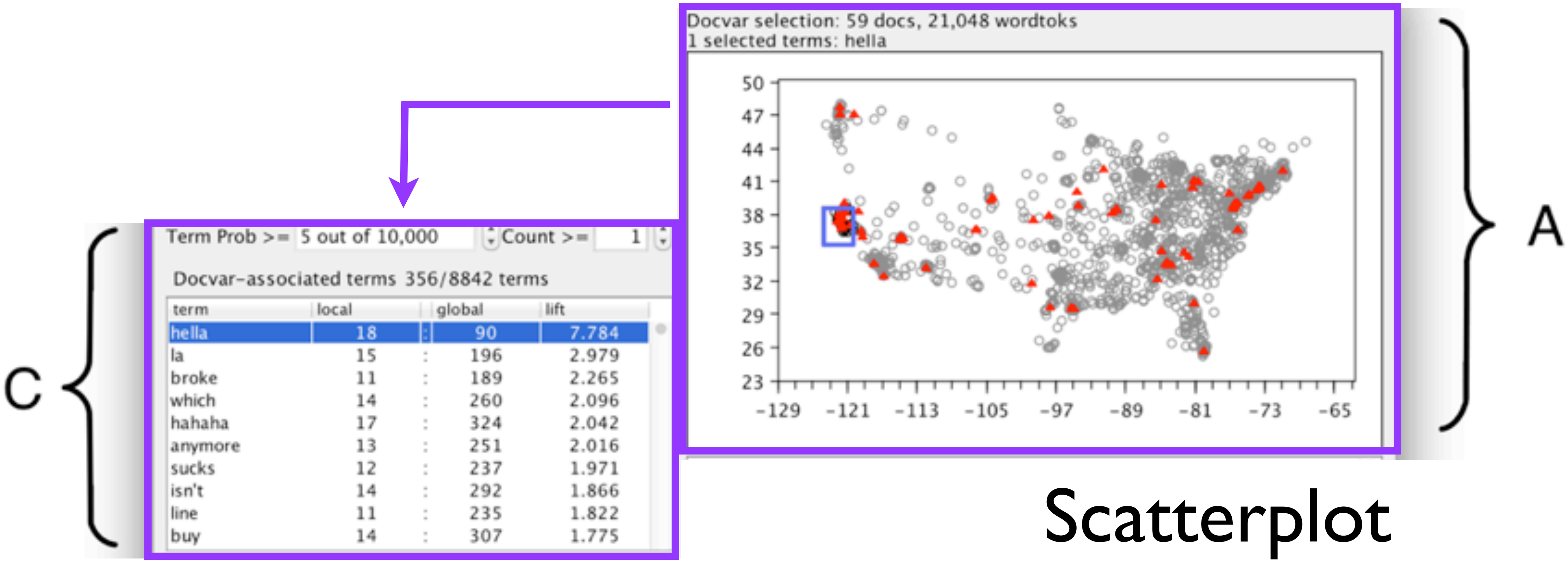
Word-covariate analysis

- Documents have metadata. How do individual words correlate?
- Words against time and space



[Eisenstein, O'Connor, Smith, Xing 2014]

Word-covariate correlations



term	local		global	lift
hella	18	:	90	7.784
la	15	:	196	2.979
broke	11	:	189	2.265
which	14	:	260	2.096
hahaha	17	:	324	2.042
anymore	13	:	251	2.016
sucks	12	:	237	1.971
isn't	14	:	292	1.866
line	11	:	235	1.822
buy	14	:	307	1.775

Scatterplot

Ranked list

$$\text{rank}_w \frac{p(w|Q)}{p(w)} \qquad \text{where} \qquad \begin{aligned} p(w|Q) &\geq \text{TermProbThresh} \\ \text{count}_Q(w) &\geq \text{TermCountThresh} \end{aligned}$$

(Exponentiated) Pointwise Mutual Information (a.k.a. *lift*)

Text Corpus Exploration

- You have a big pile of text documents. What's going on inside?
  - ~~Comparisons to document covariates~~
  - Clustering and topic models

Making sense of text

Suppose you want to learn something about a corpus that's too big to read

- What topics are trending today on Twitter?
- What research topics receive grant funding (and from whom)?
- What issues are considered by Congress (and which politicians are interested in which topic)?
- Are certain topics discussed more in certain languages on Wikipedia?

need to make sense of…

- **half a billion** tweets daily
- **80,000** active NIH grants
- **hundreds** of bills each year
- **Wikipedia** (it's big)

Making sense of text

Suppose you want to learn something about a corpus that's too big to read

Why don't we just throw all these documents at the computer and see what interesting patterns it finds?

need to make sense of…

- **half a billion** tweets daily
- **80,000** active NIH grants
- **hundreds** of bills each year
- **Wikipedia** (it's big)

Preview

- Topic models can help you automatically discover patterns in a corpus
  - **unsupervised** learning

- Topic models automatically…
  - group topically-related words in “topics”
  - associate tokens and documents with those topics

Demo

http://mimno.infosci.cornell.edu/jsLDA/

- by David Mimno
  PhD, 2012, UMass Amherst
  Now professor at Cornell
  - MALLET: open-source topic model software in Java

The "document"

- Topic models assume the "document", or unit of text, has topical specificity.
- Mimno's demo: assume every SOTU paragraph has a distribution over topics.
  - It's a discourse model...
- Below: topic-word results from Twitter data, assuming every Twitter user has a distribution over topics.
  - It's a user model...

"basketball"	"popular music"	"daily life"	"emoticons"	"chit chat"
PISTONS KOBE LAKERS game DUKE NBA CAVS STUCKEY JETS KNICKS	album music beats artist video #LAKERS ITUNES tour produced vol	tonight shop weekend getting going chilling ready discount waiting iam	:) haha :d :(;) :p xd :/ hahaha hahah	lol smh jk yea wyd coo ima wassup somethin jp

So what is “topic”?

- Loose idea: a grouping of words that are likely to appear in the same document-level **context**
- A hidden structure that helps determine what words are likely to appear in a corpus
  - but the underlying structure is different from what you’ve seen before – it’s not syntax
  - e.g. if “war” and “military” appear in a document, you probably won’t be surprised to find that “troops” appears later on

    why? it’s not because they’re all nouns

    …though you might say they all belong to the same *topic*
  - *long-range* context (versus local dependencies like n-grams, syntax)

You’ve seen these ideas before

Most of NLP is about inferring hidden structures that we assume are behind the observed text

- parts of speech, syntax trees

You’ve already seen a model that can capture hidden lexical semantics

- HMMs: based on sequential structure
- Topic models: based on document grouping of words (unordered!)

Syntax (HMM) vs Topics

HMM is a reasonable model of part-of-speech:

Stocks mixed after long holiday weekend

Microsoft codename 'Threshold': The next major Windows

Apple iPads beat early holiday expectations

- coloring corresponds to value of hidden state (POS)

Syntax (HMM) vs Topics

HMM is a reasonable model of part-of-speech:

Stocks mixed after long holiday weekend

Microsoft codename 'Threshold': The next major Windows

Apple iPads beat early holiday expectations

but you might imagine modeling topic associations instead:

Stocks mixed after long holiday weekend

Microsoft codename 'Threshold': The next major Windows

Apple iPads beat early holiday expectations

Topic models

Take an HMM, but give every document its own transition probabilities (rather than a global parameter of the corpus)

- This let's you specify that certain topics are more common in certain documents
  - whereas with parts of speech, you probably assume this doesn't depend on the specific document

Topic models

Take an HMM, but give every document its own transition probabilities (rather than a global parameter of the corpus)

- This let's you specify that certain topics are more common in certain documents
  - whereas with parts of speech, you probably assume this doesn't depend on the specific document
- We'll also assume the hidden state of a token doesn't actually depend on the previous tokens
  - "0th order"
  - individual documents probably don't have enough data to estimate full transitions
  - plus our notion of "topic" doesn't care about local interactions

Topic models

- The probability of a token is the joint probability of the word and the topic label

P(word=Apple, topic=1 | θ_d , β_1)

= P(word=Apple | topic=1, β_1) P(topic=1 | θ_d)

Topic models

- The probability of a token is the joint probability of the word and the topic label

$$P(\text{word}=\text{Apple}, \text{topic}=1 \mid \theta_d, \beta_1)$$

$$= P(\text{word}=\text{Apple} \mid \text{topic}=1, \beta_1)\, P(\text{topic}=1 \mid \theta_d)$$

each topic has distribution over *words*
(the emission probabilities)

- **global** across all documents

each document has distribution over *topics*
(the 0th order "transition" probabilities)

- **local** to each document

Topic models

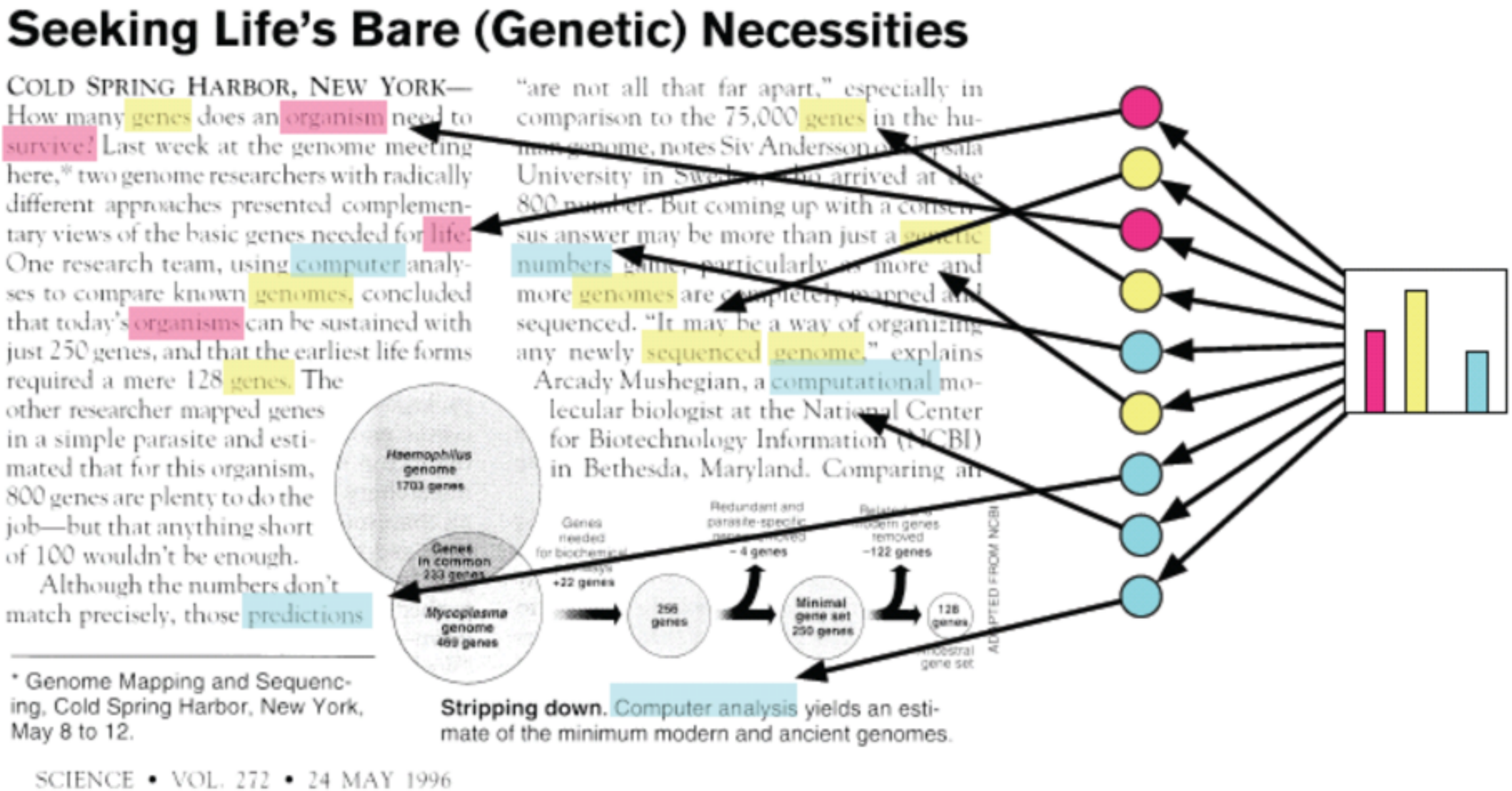
- The probability of a token is the joint probability of the word and the topic label

$$P(\text{word}=\text{Apple}, \text{topic}=1 \mid \theta_d, \beta_1)$$

$$= P(\text{word}=\text{Apple} \mid \text{topic}=1, \beta_1)\, P(\text{topic}=1 \mid \theta_d)$$

- The probability of a document is the product of all of its token probabilities
  - the tokens are independent because it's a 0th order model
- The probability of a corpus is the product of all of its document probabilities

Topic models



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 "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

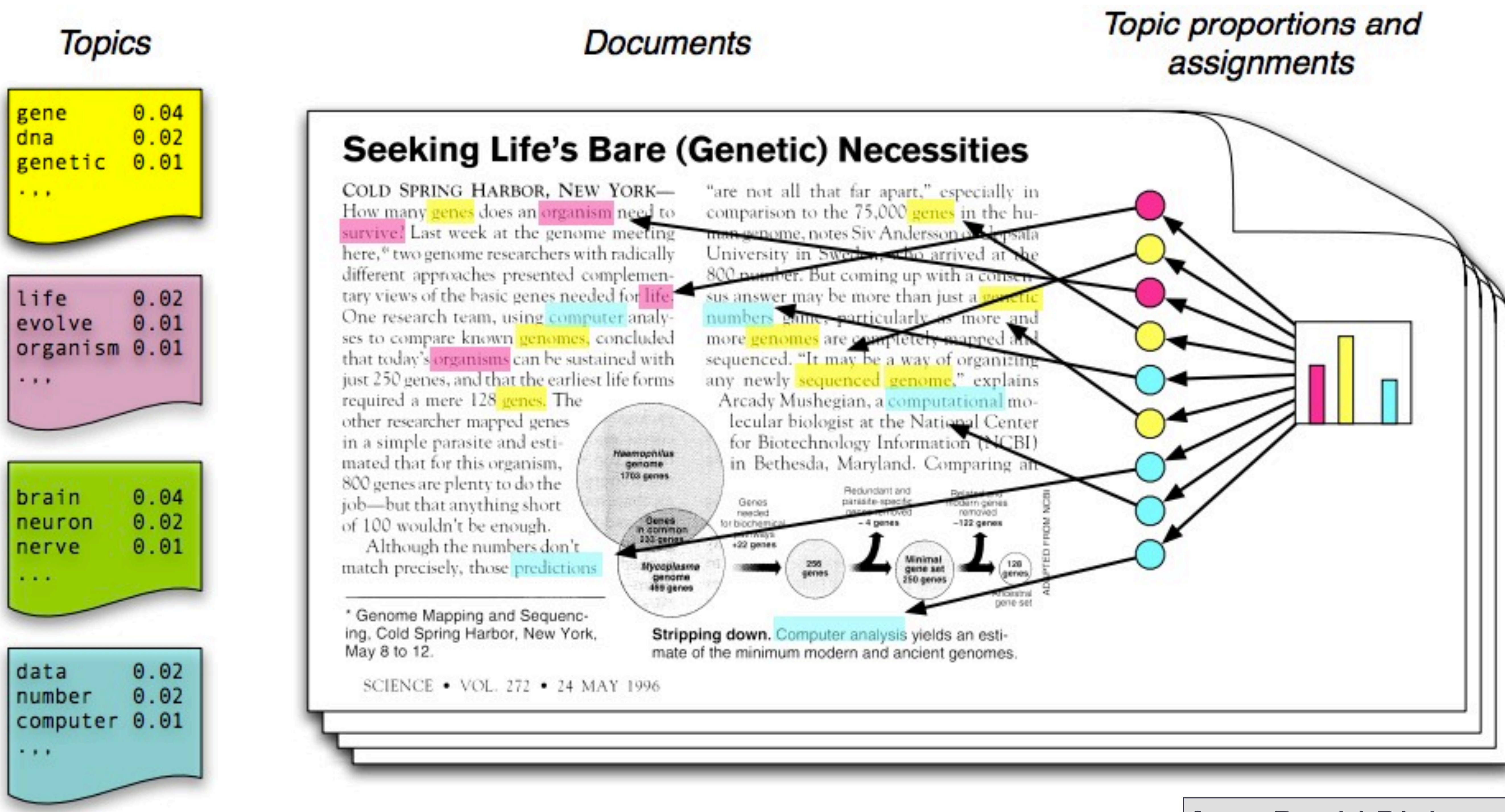
Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

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

SCIENCE • VOL. 272 • 24 MAY 1996

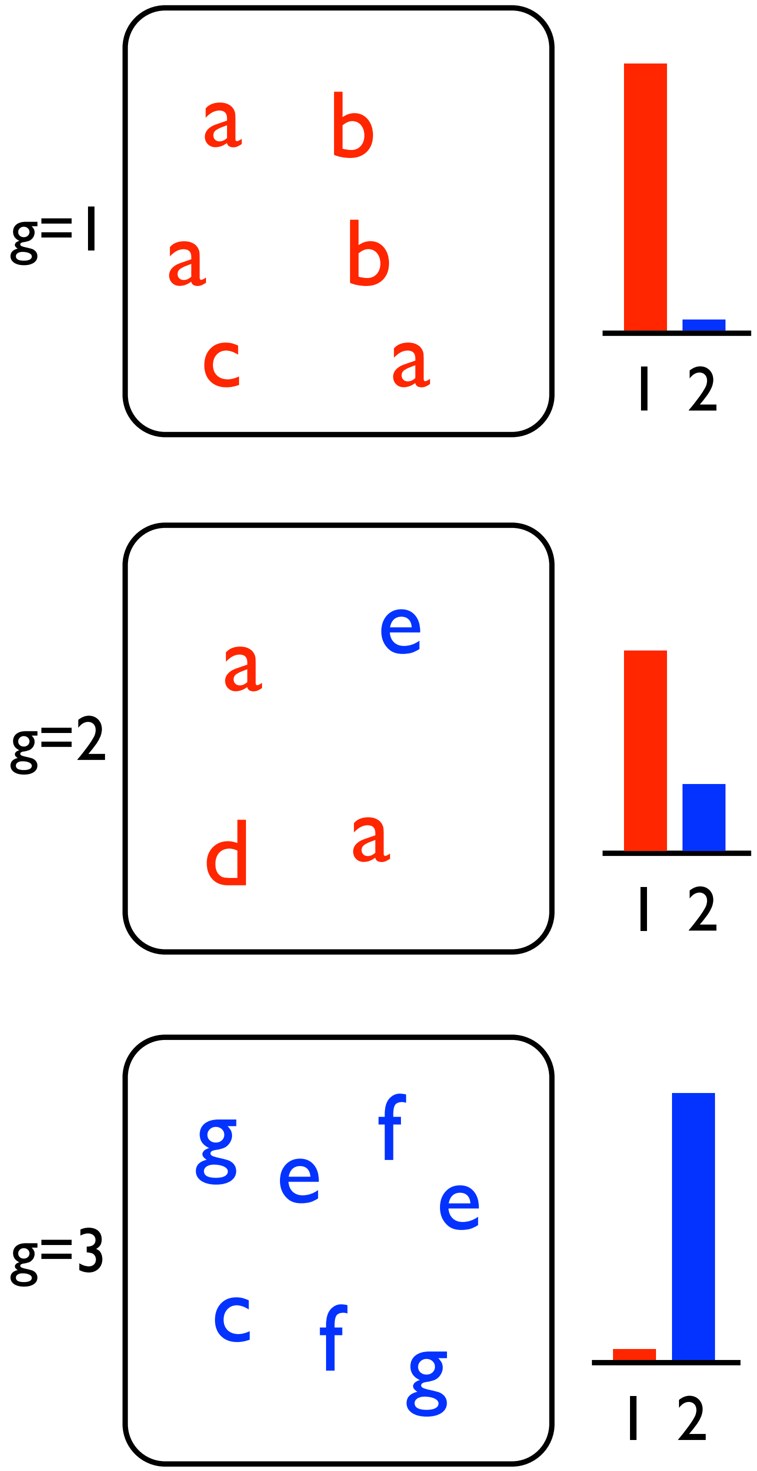
from David Blei

Topic models



from David Blei

Why is it possible to learn topics at all?



Doc-Topic dist:
over *K* classes
A document is about a small set of topics.

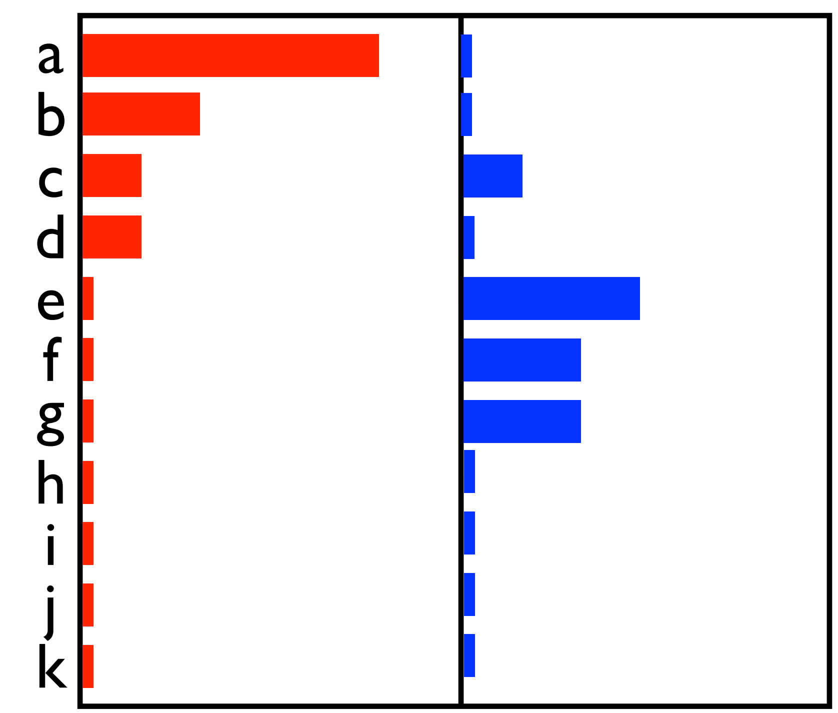
$$\theta_g \sim Dir$$

Topic-Word dist:
over vocabulary
A topic is about a small set of words.

$$\phi_k \sim Dir$$

Word distrib. is mixture

$$w \sim \mathrm{Mult}(\Phi\ \theta)$$



$$z \sim \mathrm{Mult}(\theta_g)$$
$$w \sim \mathrm{Mult}(\phi_z)$$

Latent Dirichlet allocation

[Pritchard et al. 2000, Blei et al. 2003]

Estimating the parameters

- Need to estimate the parameters θ, β
  - want to pick parameters that maximize the likelihood of the observed data
- This is easy if all the tokens were labeled with topics (observed variables)

Data: Apple iPads beat early holiday expectations

  - just counting
- But we don't actually know the (hidden) topic assignments

Data: Apple iPads beat early holiday expectations

  - sound familiar?

Estimating the parameters

Expectation Maximization (EM) to the rescue!

1. Compute the expected value of the variables, given the current model parameters

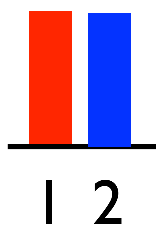
2. Pretend these *expected* counts are real and update the parameters based on these
   - now parameter estimation is back to “just counting”

3. Repeat until convergence

Example: political verb phrase clustering based on arguments

subj=Israel*, *obj*=Palestin*

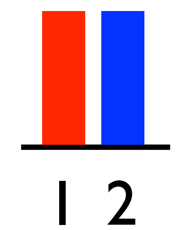
θ =



- say <-ccomp be to
- release to
- take control of
- occupy
- wound in
- scuffle with
- be <-xcomp meet
- meet with
- meet with
- arrest
- commit to
- strike
- carry in
- continue in
- reject
- fire at target in
- start around
- ratchet pressure on
- shell
- hit

subj=US/American, *obj*=France/French

θ =



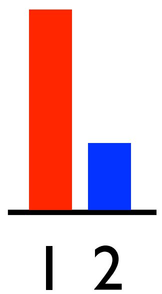
- travel <-xcomp meet with
- consider
- meet with
- meet with
- meet with
- release with
- welcome
- welcome by
- win
- agree with
- indict
- win from
- concern over
- win
- indict

Topics (phrase probs/dictionaries) ϕ_1 ϕ_2

agree with, arrest, be <-xcomp meet, carry in, commit to, concern over, consider, continue in, fire at target in, hit, indict, meet with, occupy, ratchet pressure on, reject, release to, release with, say <-ccomp be to, scuffle with, shell, start around, strike, take control of, travel <-xcomp meet with, welcome, welcome by, win, win from, wound in

Example: political verb phrase clustering based on arguments

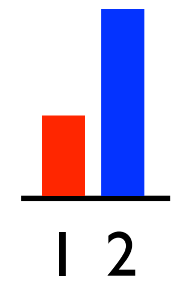
subj=Israel*, *obj*=Palestin*

θ = 

say <-ccomp be to
release to
take control of
occupy
wound in
scuffle with
be <-xcomp meet
meet with
meet with
arrest

commit to
strike
carry in
continue in
reject
fire at target in
start around
ratchet pressure on
shell
hit

subj=US/American, *obj*=France/French

θ = 

travel <-xcomp meet with
consider
meet with
meet with
meet with

release with
welcome
welcome by
win
agree with
indict
win from
concern over
win
indict

Topics (phrase probs/dictionaries) ϕ_1 ϕ_2

agree with, arrest, be <-xcomp meet, carry in, commit to, concern over, consider, continue in, fire at target in, hit, indict, meet with, occupy, ratchet pressure on, reject, release to, release with, say <-ccomp be to, scuffle with, shell, start around, strike, take control of, travel <-xcomp meet with, welcome, welcome by, win, win from, wound in

Estimating the parameters

Expectation Maximization (EM) to the rescue!

E-step

P(topic=1 | word=Apple, θ_d, β_1)

$$= \frac{P(\text{word=Apple, topic=}1 \mid \theta_d, \beta_1)}{\Sigma_k\, P(\text{word=Apple, topic=}k \mid \theta_d, \beta_k)}$$

Estimating the parameters

Expectation Maximization (EM) to the rescue!

M-step

new θ_{d1}

$$= \frac{\text{\# tokens in } d \text{ with topic label 1}}{\text{\# tokens in } d}$$

if *the topic labels were observed!*

- just counting

Estimating the parameters

Expectation Maximization (EM) to the rescue!

M-step

new θ_{d1}

$$= \frac{\Sigma_{i \in d} \text{P(topic } i\text{=}1 \mid \text{word } i, \theta_d, \beta_1)}{\Sigma_k \Sigma_{i \in d} \text{P(topic } i\text{=}k \mid \text{word } i, \theta_d, \beta_k)}$$

just the number of tokens in the doc

sum over each token *i* in document *d*

- numerator: "the expected number of tokens with topic 1"
- denominator: "the (expected) number of tokens"

Estimating the parameters

Expectation Maximization (EM) to the rescue!

M-step

new β_{1w}

$$= \frac{\text{\# tokens with topic label 1 and word type } w}{\text{\# tokens with topic label 1}}$$

if *the topic labels were observed!*

- just counting

Estimating the parameters

Expectation Maximization (EM) to the rescue!

M-step

new β_{1w}

$$= \frac{\sum_i \overbrace{\mathrm{I}(\text{word } i=w)}^{\text{1 if word=w, 0 otherwise}} \mathrm{P}(\text{topic } i=1 \mid \text{word } i=w, \theta_d, \beta_1)}{\sum_v \sum_i \mathrm{I}(\text{word } i=v)\, \mathrm{P}(\text{topic } i=1 \mid \text{word } i=v, \theta_d, \beta_1)}$$

sum over vocabulary

sum over each token *i* in the entire corpus

- numerator: “the expected number of times word *w* belongs to topic 1”
- denominator: “the expected number of all tokens belonging to topic 1”

Smoothing revisited

- Topics are just language models
- Can use standard smoothing techniques for the topic parameters (the word distributions)
  - most commonly, pseudocount smoothing
- Can also smooth the topic proportions in each document

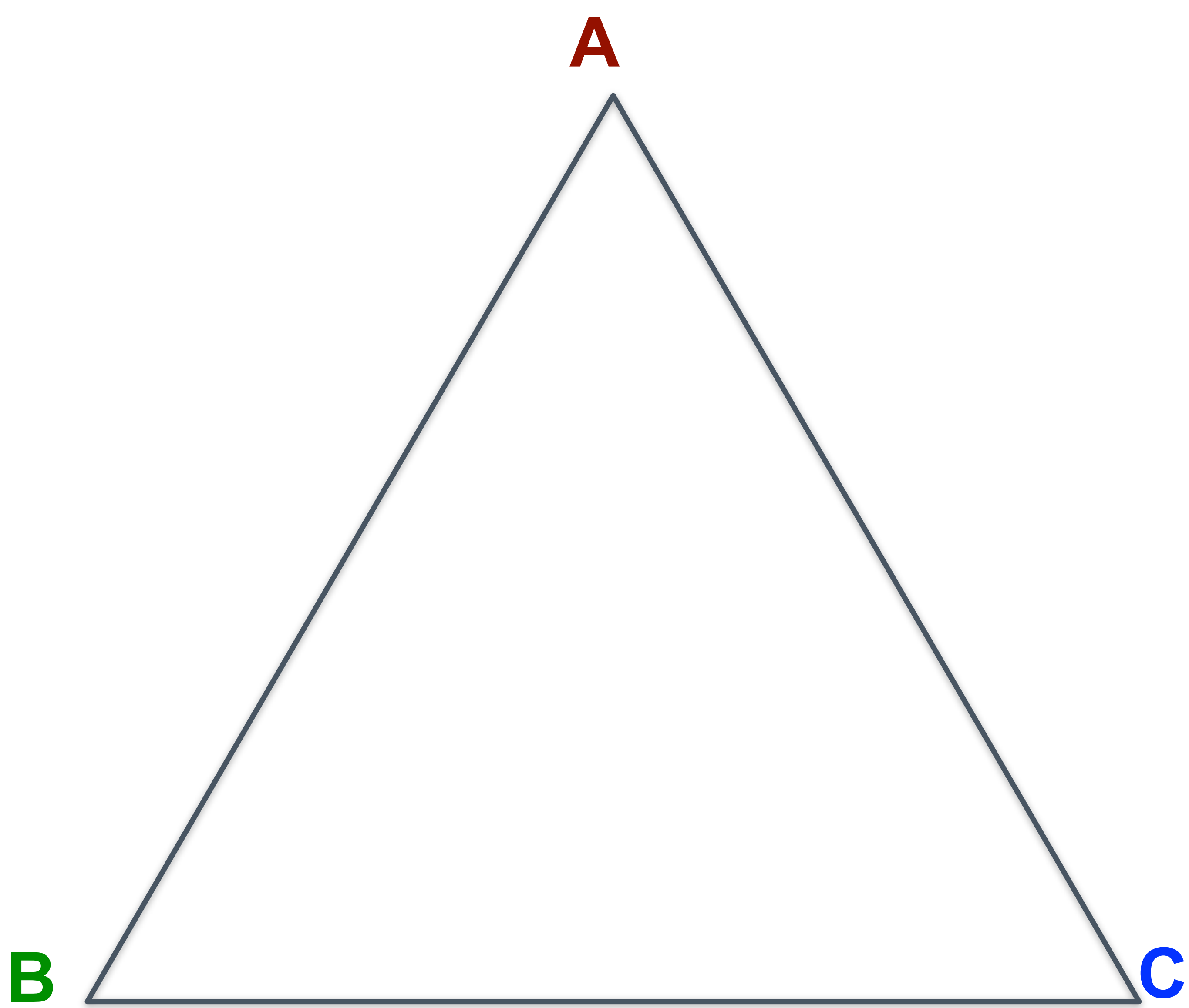
Smoothing: A Bayesian perspective

- The parameters themselves are random variables
  - $P(\theta \mid \alpha)$
  - $P(\beta \mid \eta)$
- Some parameters are more likely than others
  - as defined by a **prior** distribution
- You'll see that pseudocount smoothing is the result when the parameters have a prior distribution called the **Dirichlet** distribution
  - (in fact, pseudocount smoothing is also called "Dirichlet prior smoothing")

Geometry of probability distributions

A distribution over K elements is a point on a K-1 **simplex**

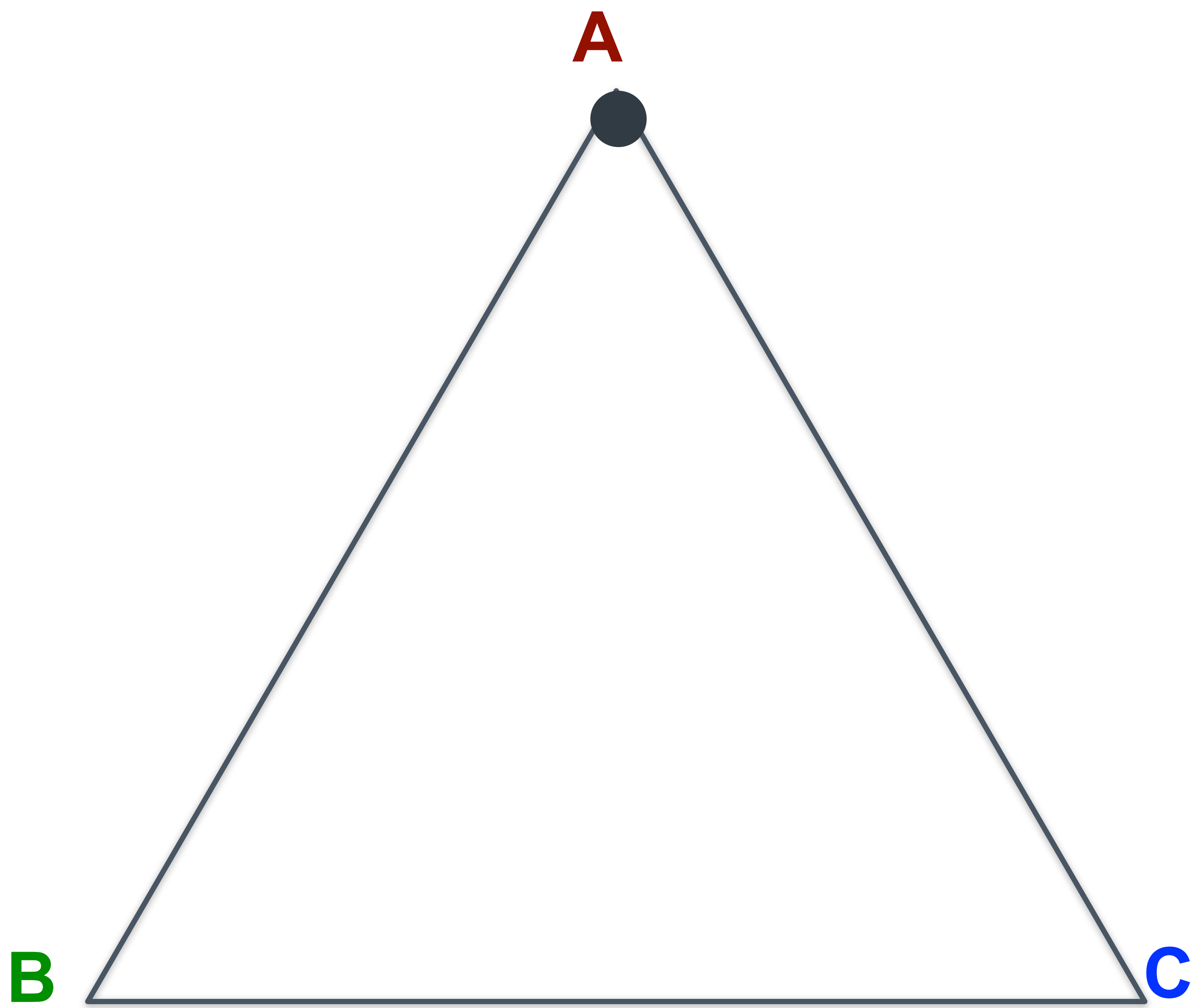
- a 2-simplex is called a triangle



Geometry of probability distributions

A distribution over K elements is a point on a K-1 **simplex**

- a 2-simplex is called a triangle

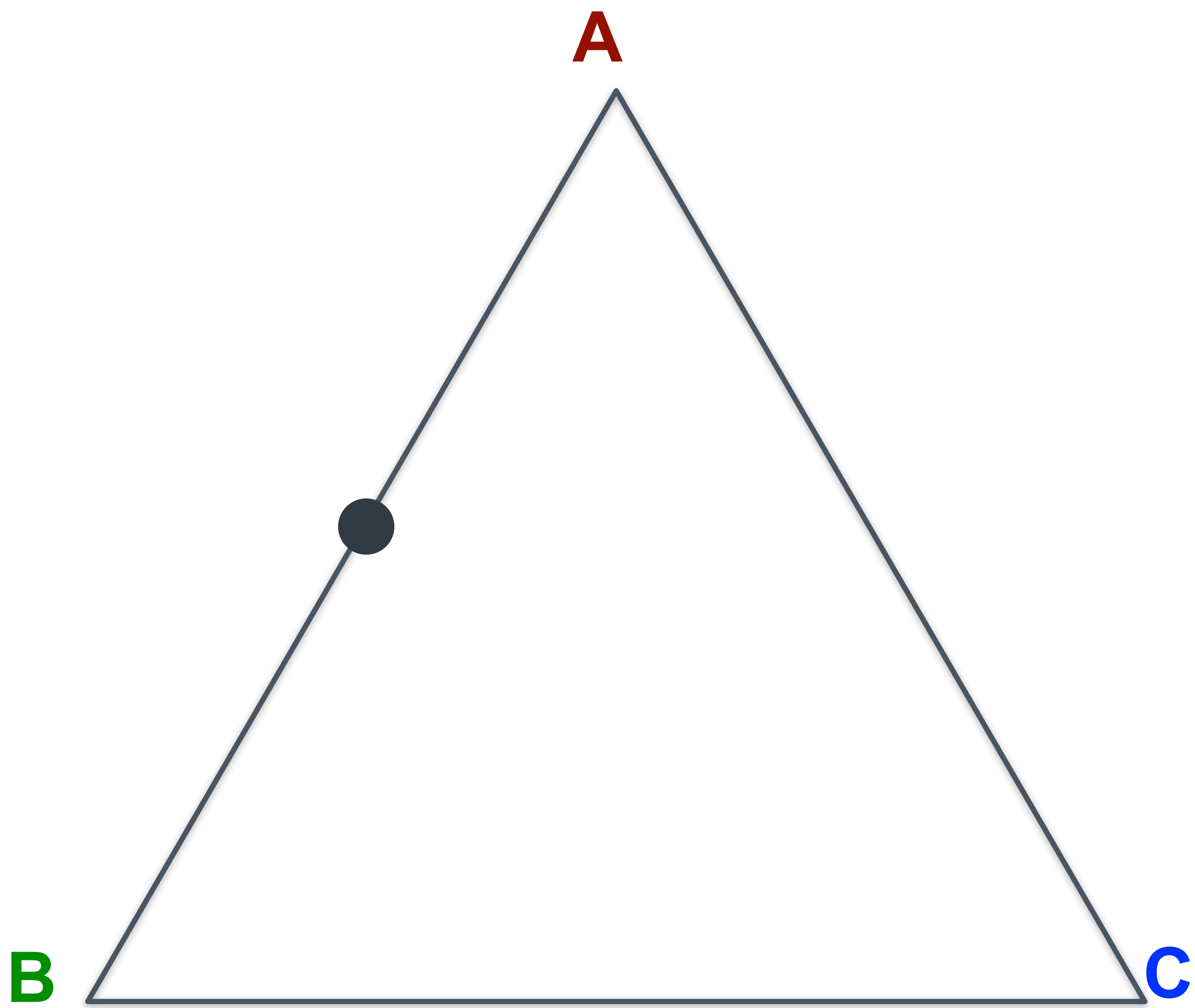


P(**A**) = 1
P(**B**) = 0
P(**C**) = 0

Geometry of probability distributions

A distribution over K elements is a point on a K-1 **simplex**

- a 2-simplex is called a triangle

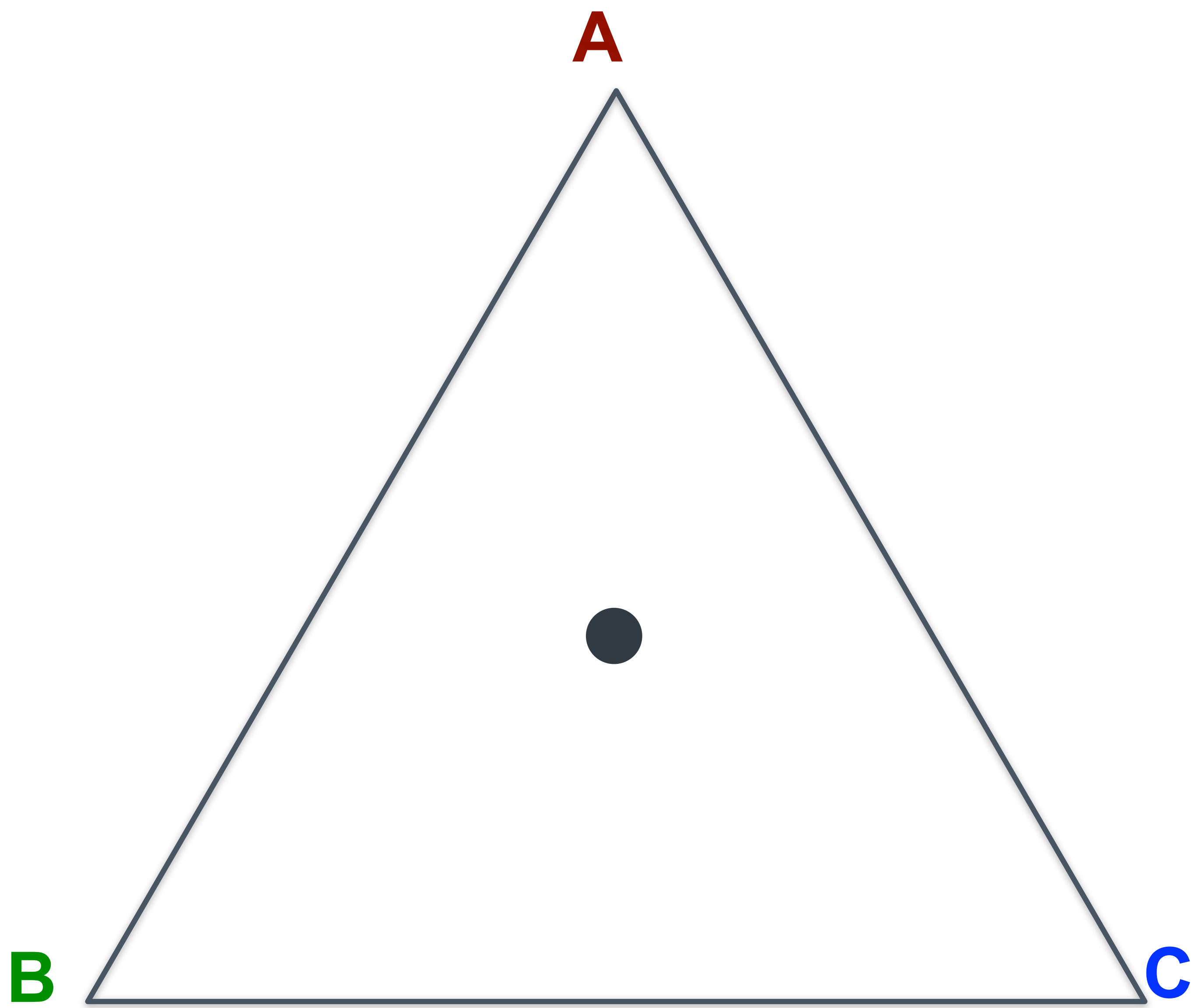


P(**A**) = 1/2
P(**B**) = 1/2
P(**C**) = 0

Geometry of probability distributions

A distribution over K elements is a point on a K-1 **simplex**

- a 2-simplex is called a triangle

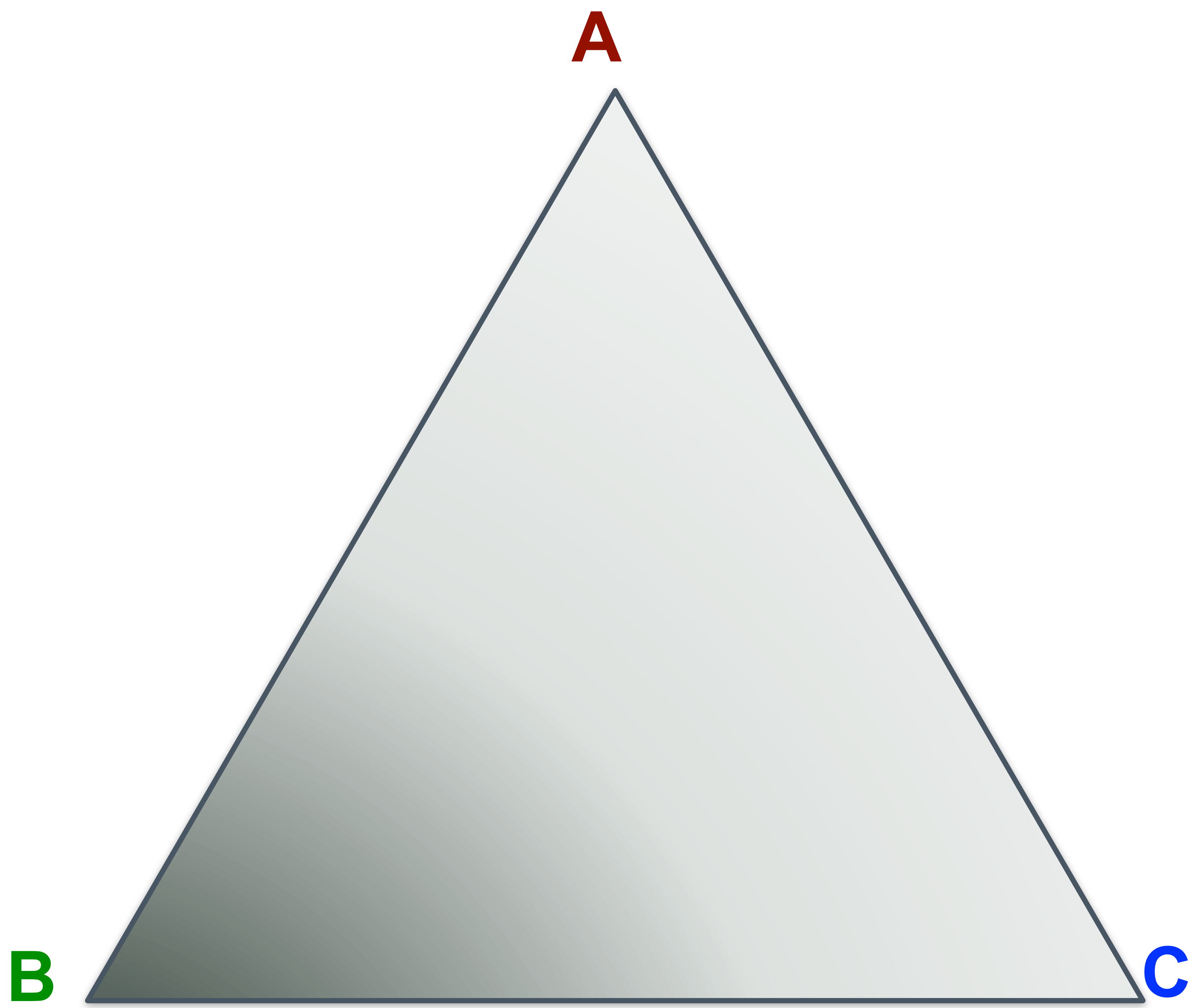


P(**A**) = 1/3
P(**B**) = 1/3
P(**C**) = 1/3

The Dirichlet distribution

Continuous distribution (probability density) over points in the simplex

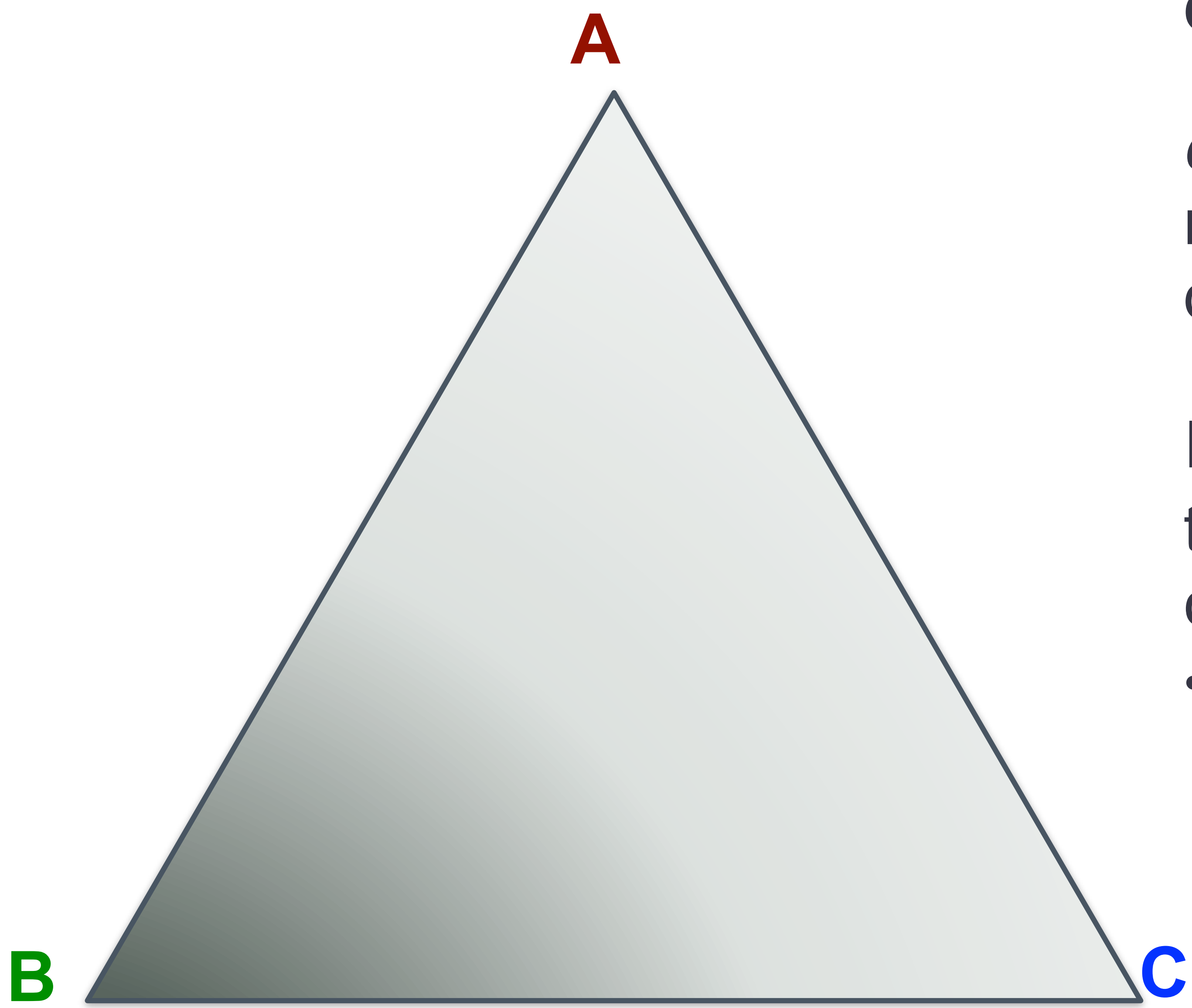
- “distribution of distributions”



The Dirichlet distribution

Continuous distribution (probability density) over points in the simplex

- "distribution of distributions"



denoted Dirichlet($\boldsymbol{\alpha}$)

$\boldsymbol{\alpha}$ is a vector that gives the mean/variance of the distribution

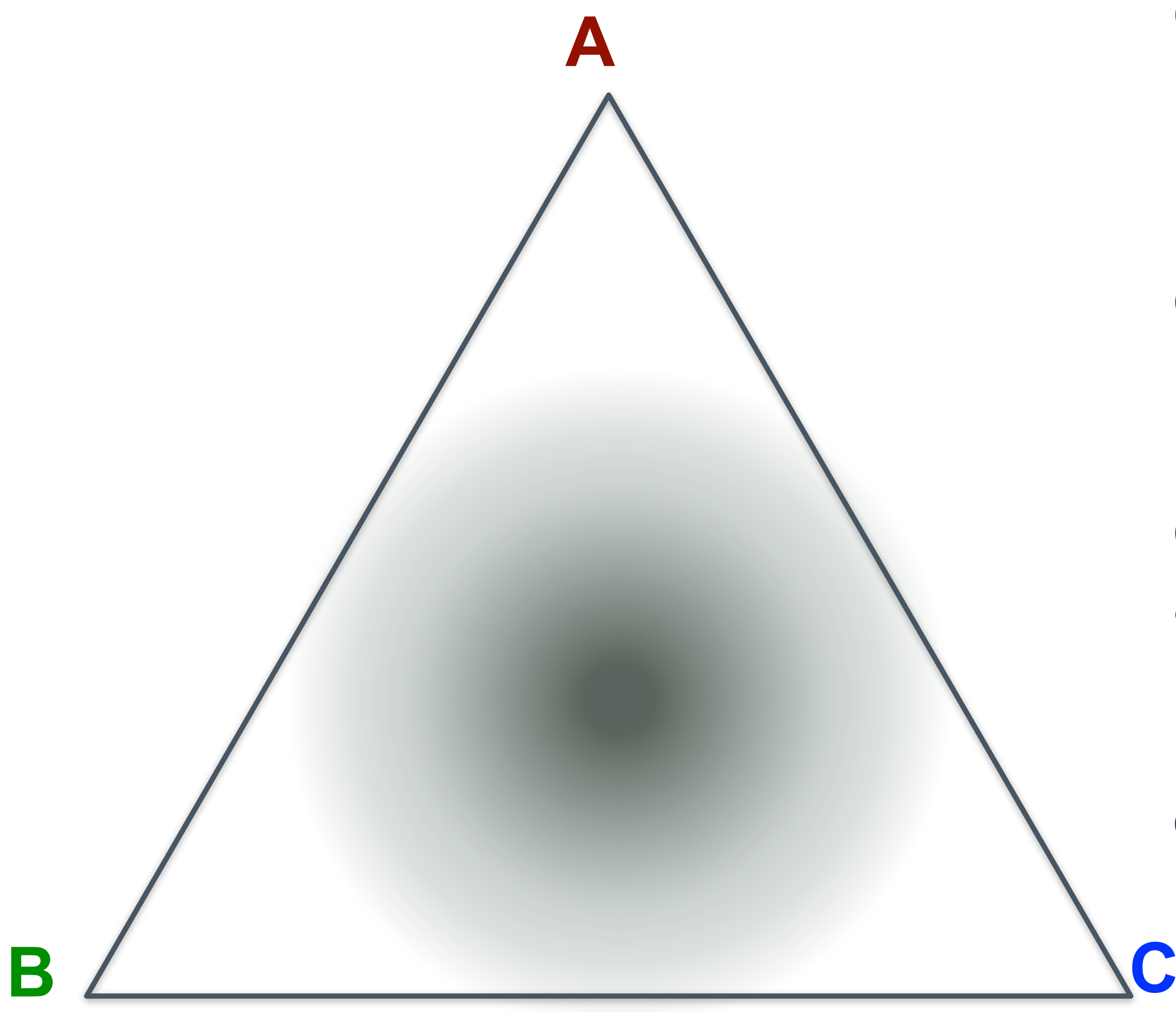
In this example, α_B is larger than the others, so points closer to *B* are more likely

- distributions that give **B** high probability are more likely than distributions that don't

The Dirichlet distribution

Continuous distribution (probability density) over points in the simplex

- "distribution of distributions"



denoted Dirichlet(α)

α is a vector that gives the mean/variance of the distribution

In this example, $\alpha_A=\alpha_B=\alpha_C$, so distributions close to uniform are more likely

Larger values of α mean higher density around mean
(lower variance)

Latent Dirichlet allocation (LDA)

LDA is the basic topic model you saw earlier, but with Dirichlet priors on the parameters θ and β

- $P(\theta \mid \alpha)$ = Dirichlet(α)
- $P(\beta \mid \eta)$ = Dirichlet(η)

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})$$
$$= \prod_{i=1}^{K} p(\beta_i) \prod_{d=1}^{D} p(\theta_d) \left(\prod_{n=1}^{N} p(z_{d,n} \mid \theta_d) p(w_{d,n} \mid \beta_{1:K}, z_{d,n}) \right)$$

The posterior distribution

- Now we can reason about the probability of the hidden variables and parameters, given the observed data

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D} \mid w_{1:D}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})}$$

MAP estimation

- Earlier we saw how to use EM to find parameters that maximize the likelihood of the data, given the parameters

- EM can also find the **maximum a posteriori (MAP)** value
  - the parameters that maximum the posterior probability

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D} \mid w_{1:D}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})}$$

constant

- This is basically maximum likelihood estimation, but with additional terms for the probability of θ and β

MAP estimation

- E-step is the same
- M-step is modified

new θ_{d1}

$$= \frac{\overbrace{\alpha_1 - 1}^{\text{pseudocounts}} + \sum_{i \in d} P(\text{topic } i{=}1 \mid \text{word } i, \theta_d, \beta_1)}{\sum_k \left(\alpha_k - 1 + \sum_{i \in d} P(\text{topic } i{=}k \mid \text{word } i, \theta_d, \beta_k)\right)}$$

This amounts to pseudocount smoothing!

"pseudocount-minus-1 smoothing"

Where do the pseudocounts come from?

The probability of observing the *k*th topic *n* times given the parameter θ_k is proportional to:

$$\theta_k^{\,n}$$

The probability density of the parameter θ_k given the Dirichlet parameter α_k is proportional to:

$$\theta_k^{\,\alpha_k\text{-}1}$$

So the product of these probabilities is proportional to:

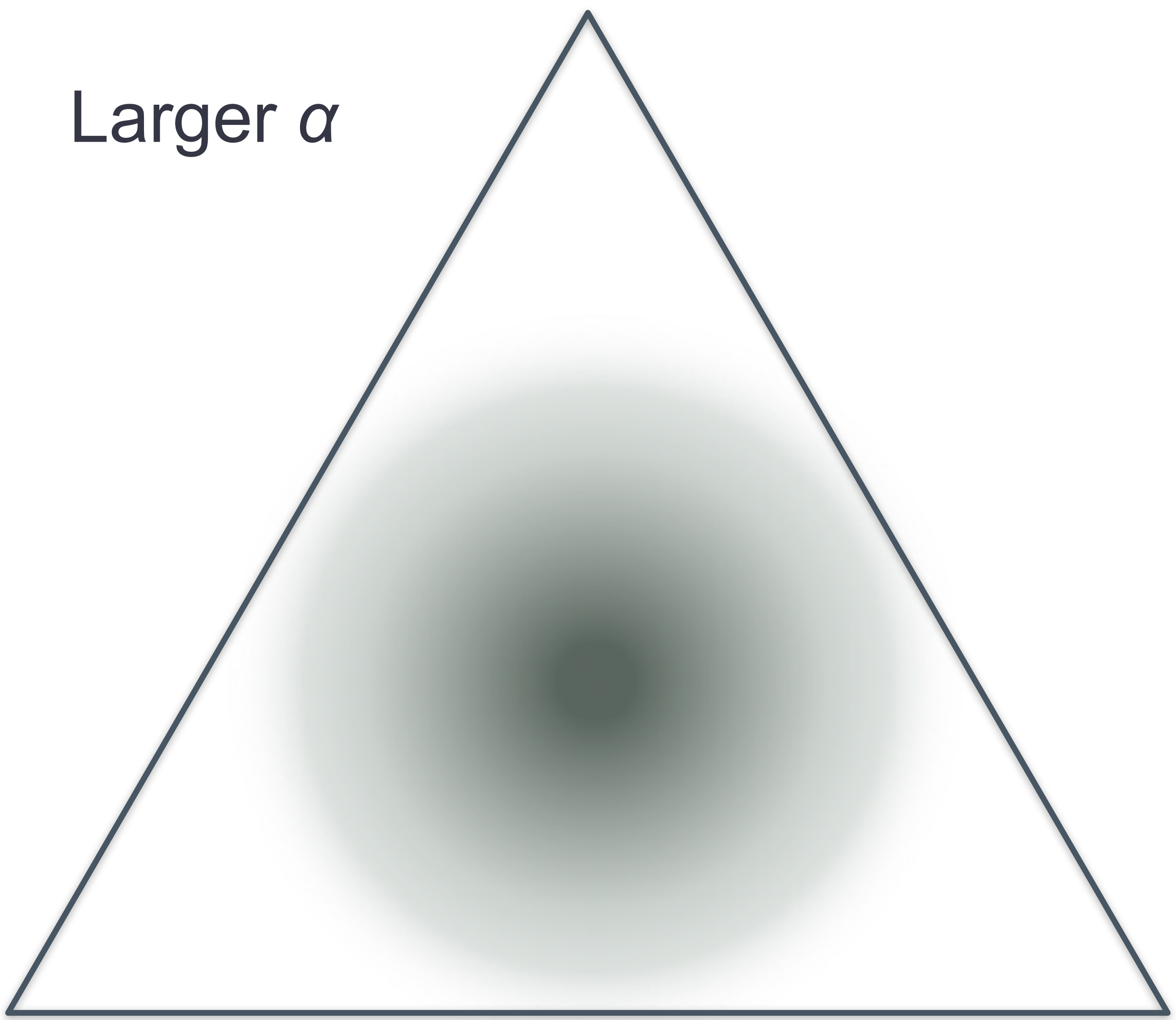
$$\theta_k^{\,n+\alpha_k\text{-}1}$$

Smoothing: A Bayesian perspective

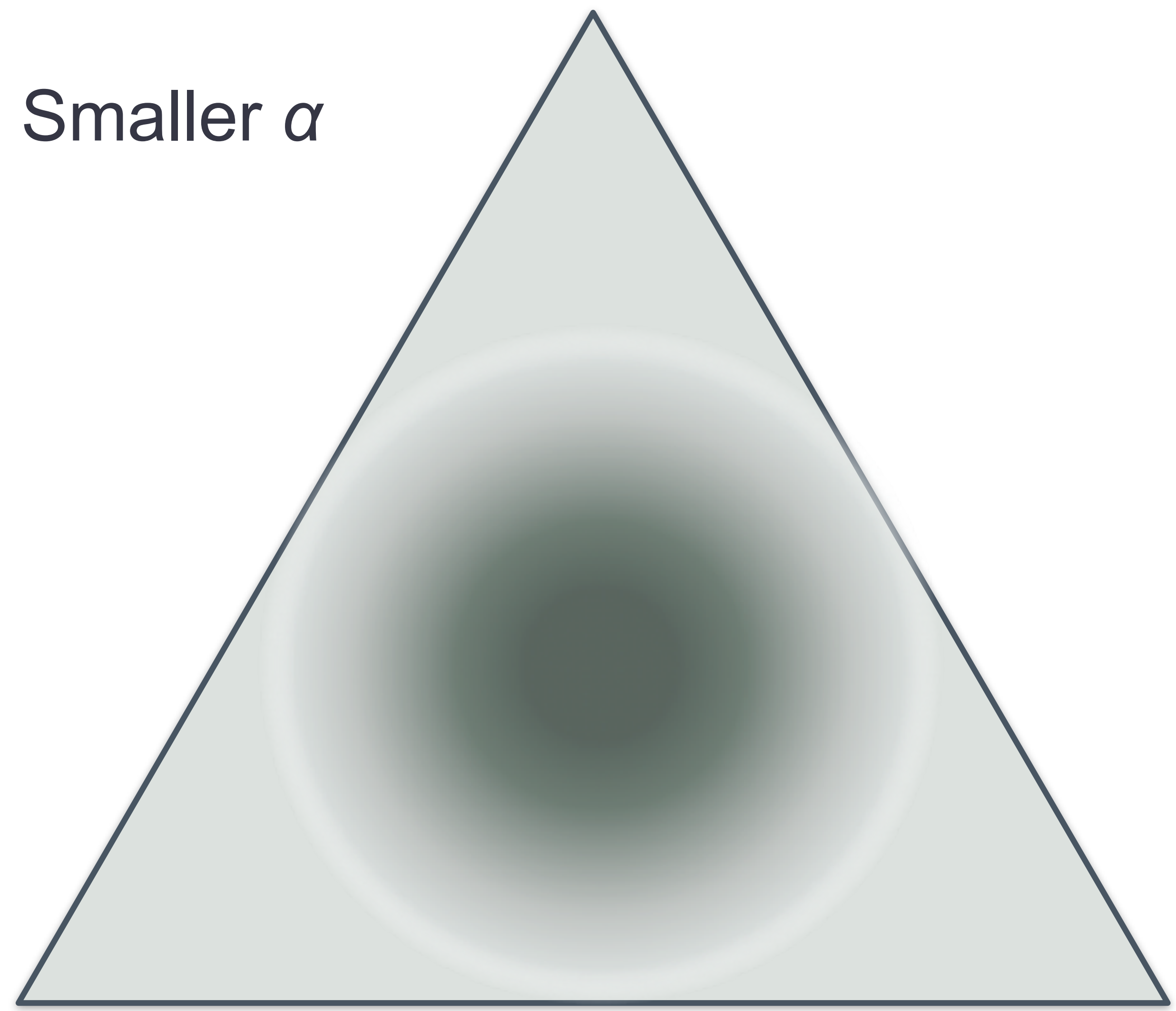
Larger pseudocounts will bias the MAP estimate more heavily

Larger Dirichlet parameters concentrate the density around the mean

Larger α

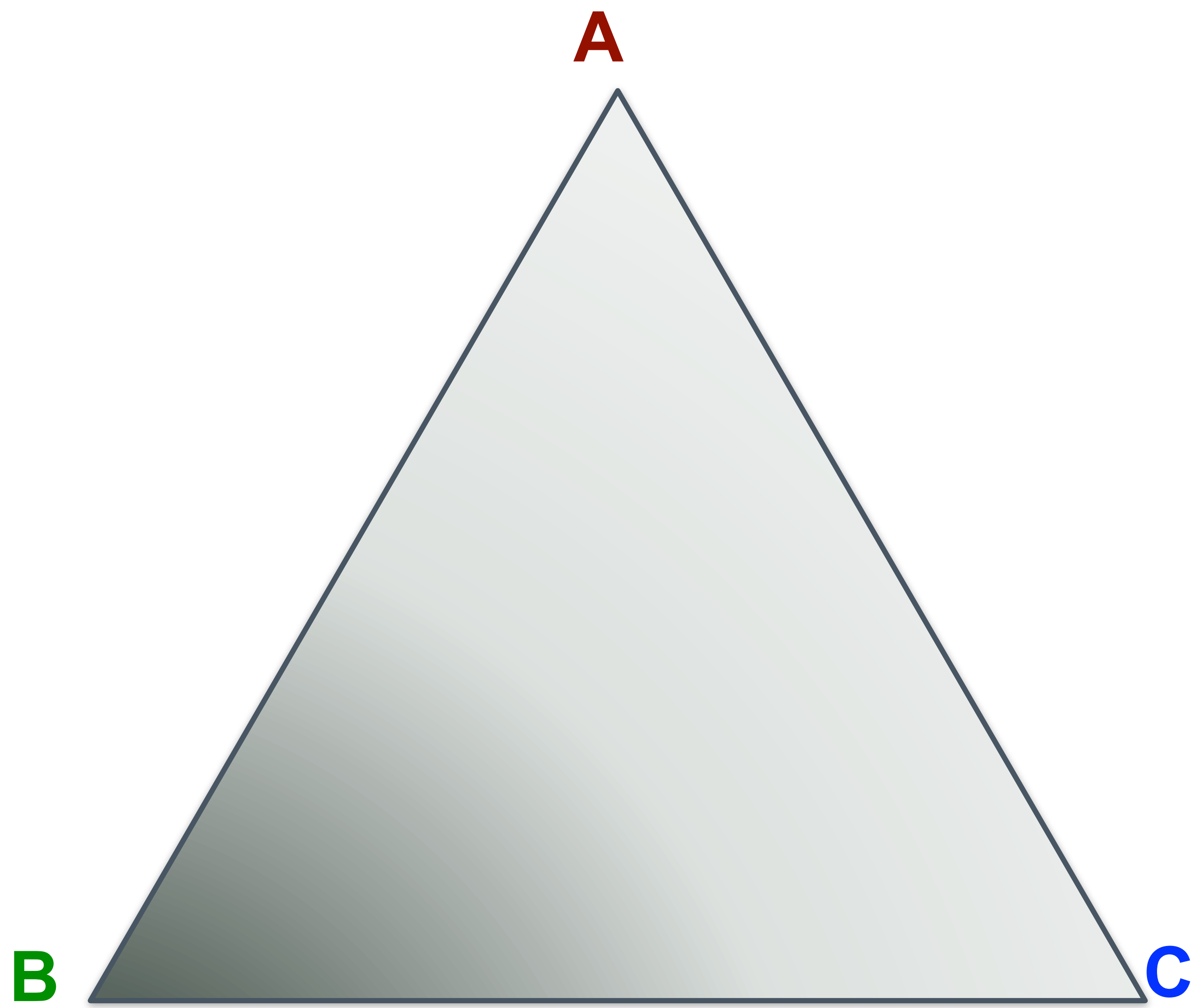


Smaller α



Asymmetric smoothing

We don’t have to smooth toward the uniform distribution



Asymmetric smoothing

We don't have to smooth toward the uniform distribution

- You might expect one topic to be very common in all documents

Symmetric α	0.080	a **field emission** an **electron** the
	0.080	a the **carbon** and **gas** to an
	0.080	the of a to and about at
	0.080	of a **surface** the with in **contact**
	0.080	the a and to is of **liquid**
Asymmetric α	0.895	the a of to and is in
	0.187	**carbon nanotubes nanotube catalyst**
	0.043	sub is c or and n sup
	0.061	**fullerene compound fullerenes**
	0.044	**material particles coating inorganic**

from Hanna Wallach, David Mimno, Andrew McCallum. NIPS 2009.

Posterior inference

What if we don't just want the parameters that maximize the posterior?

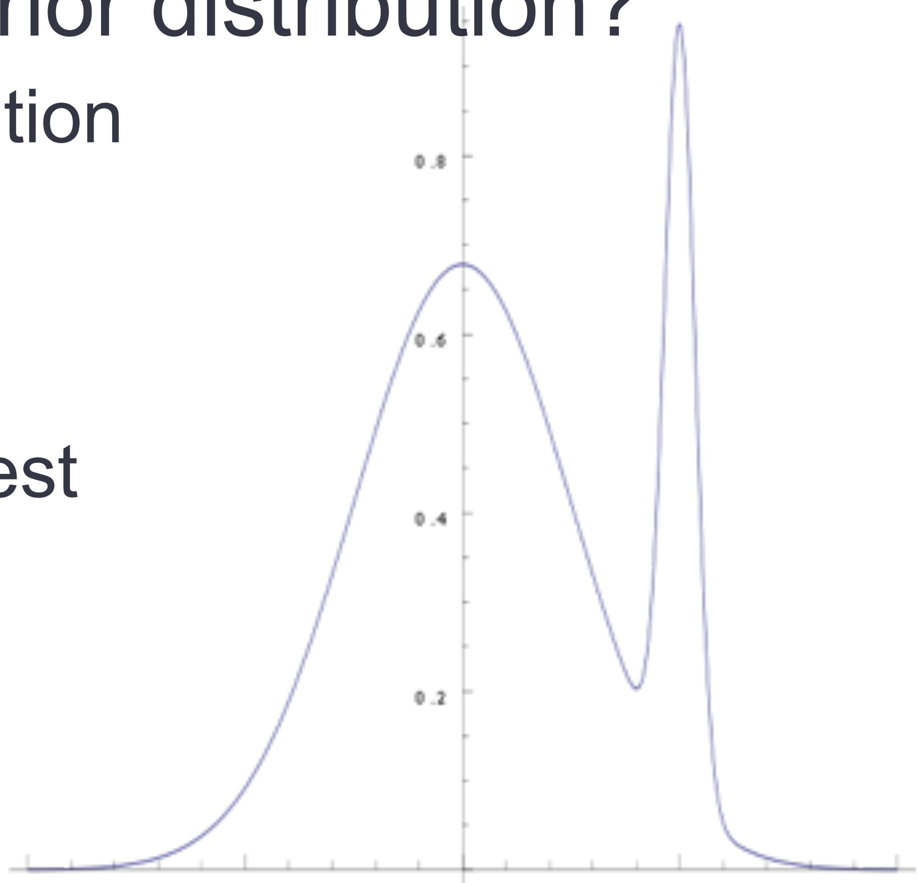
$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D} \mid w_{1:D}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})}$$

What if we care about the entire posterior distribution?

- or at least the mean of the posterior distribution

Why?

- maybe the maximum doesn't look like the rest
- other points of the posterior more likely to generalize to data you haven't seen before



Posterior inference

What if we don't just want the parameters that maximize the posterior?

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D} \mid w_{1:D}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})}$$

This is harder

- Computing the denominator involves marginalizing over all possible configurations of the hidden variables/parameters

Posterior inference: approximations

- Random sampling
  - Monte Carlo methods
- Variational inference
  - Optimization using EM-like procedure
  - MAP estimation is a simple case of this

I didn't tell you…

- where the number of topics K comes from
- where the Dirichlet parameters α and η come from

What are topic models good for?

- Extremely useful for exploratory data analysis. But,
  - Did you get the right topics/concepts for what you care about?
  - Did you find them at the right granularity?
  - How to evaluate?

- For downstream applications
  - Topic model gives dimension reduction compared to full vocab
  - e.g. doc classification, with doc-topic theta as features
  - My opinions:
    - When labeled data is small, doc-topics can be useful
    - When labeled data is plentiful (>1000's of examples), discriminative models based on word count features always seem to do better
    - Many have found that combining topics with word features can do better

Extensions

- n-grams
- topic hierarchies
- supervision

- can you think of other ideas?