The Gibbs Sampler

Bayesian Statistics Explorations

M Loecher

Motivation

Rejection Sampling

Last week we learned about a "general workhorse" MCMC algorithm, the *Metropolis Hastings Algorithm*. While it is a powerful and versatile "hill climbing" sampling scheme, there are a few challenges:

- Designing a proposal distribution can be tricky
- Tradeoff between bandwidth and sampling efficiency (number of rejected samples)
- Hyperparameter tuning

Hierarchical Models

Gibbs sampling is another MCMC method which works if all conditional posterior densities can be explicitly derived.

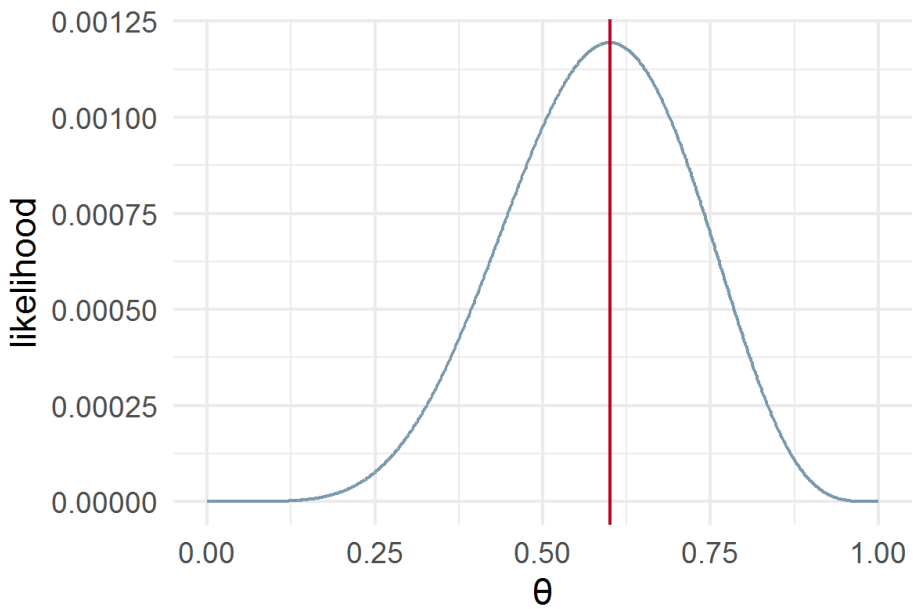
- No specially designed proposal distributions
- No rejected samples
- No hyperparameter tuning
- Conditionals eays for most hierarchical models
- Very popular in machine learning, e.g. LDA, Bayesian Networks, NLP

Estimation

Maximum Likelihood

Let's say I've flipped a coin 10 times and got 6 heads, 4 tails. Each coin flip is independent so the likelihood of a value of θ given our observed data is:

$$P(X_1 = x_1, \ldots, X_{10} = x_{10}) = L(\theta) = \prod_{i=1}^{10} \theta^{x_i}(1-\theta)^{1-x_i} = \theta^6(1-\theta)^4$$



Beta Binomial 1

$$\underbrace{p(\theta|D)}_{posterior} = \frac{\overbrace{p(D|\theta)}^{likelihood}\overbrace{p(\theta)}^{prior}}{\underbrace{p(D)}_{evidence}} \tag{1}$$

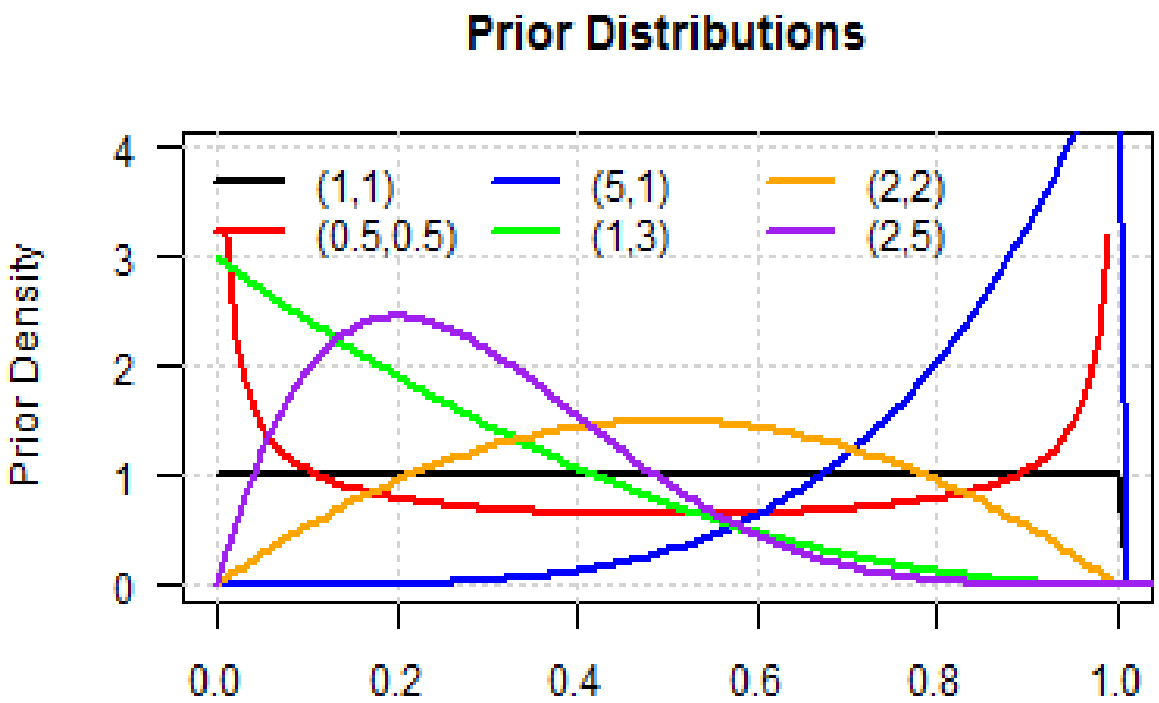
$$\underbrace{p(\theta|D)}_{posterior} \propto \overbrace{p(D|\theta)}^{likelihood}\overbrace{p(\theta)}^{prior} \tag{2}$$

To estimate the posterior of the bernoulli distribution we plug in our likelihood and prior (Beta distribution)

Bayesian Estimate

The Beta distribution is a conjugate distribution of the binomial distribution.

$$\pi(\theta|\alpha,\beta) = Beta(\alpha,\beta) = \theta^{\alpha-1} \cdot (1-\theta)^{\beta-1}/B(\alpha,\beta)$$



Beta Binomial 2

$$p(\theta|z,N) \propto \overbrace{\theta^{z}(1-\theta)^{(N-z)}}^{likelihood} \frac{\overbrace{\theta^{(\alpha-1)}(1-\theta)^{(\beta-1)}}^{prior}}{B(\alpha,\beta)} \tag{3}$$

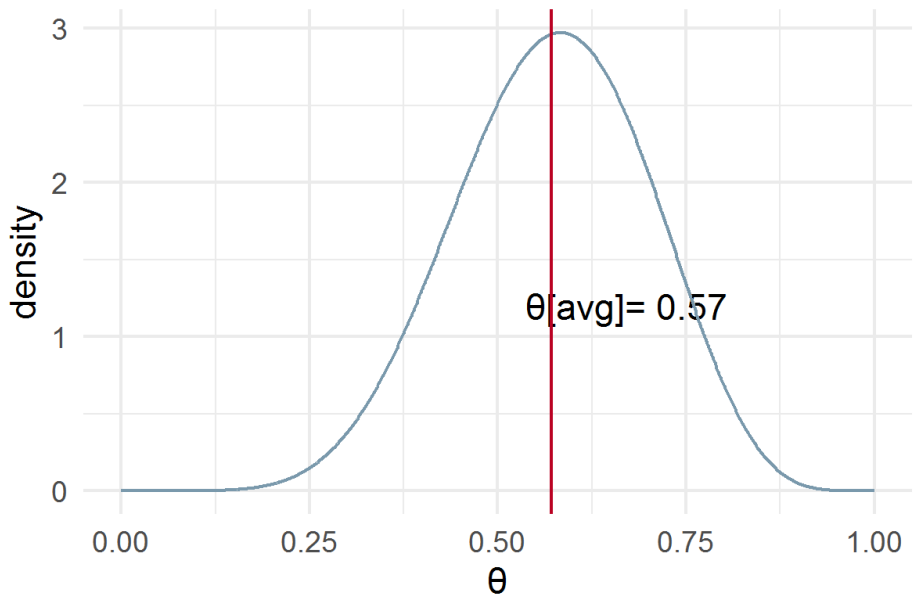
$$p(\theta|z,N) \propto \frac{\overbrace{\theta^{(\alpha+z-1)}(1-\theta)^{(N-z+\beta-1)}}^{Same\ Pattern\ as\ Prior}}{B(\alpha,\beta)} \tag{4}$$

$$p(\theta|z,N) \propto Beta(\alpha+z, N-z+\beta)$$

Posterior Density

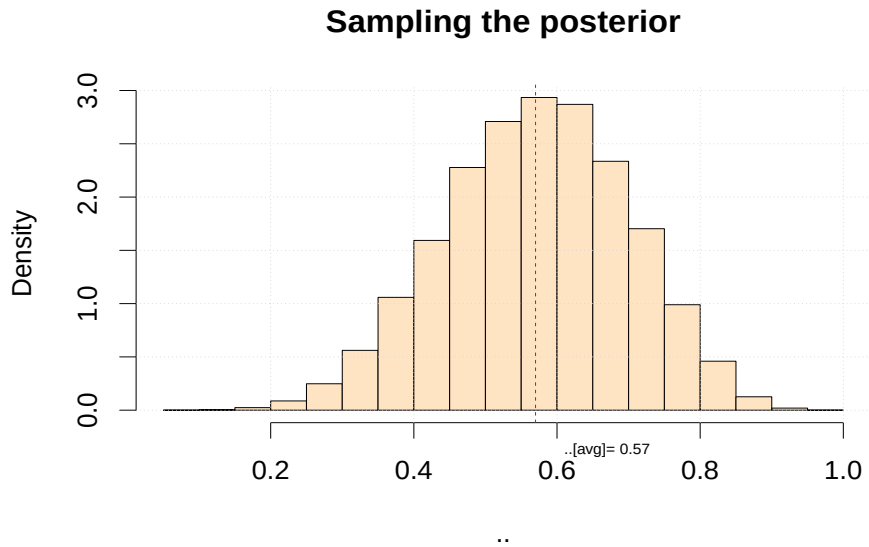
Choose a "weakly informative" prior: $\alpha = \beta = 2$; Weighted Average of prior and data based likelihood!

$$\hat{\theta} = \frac{6+2}{10+4}$$



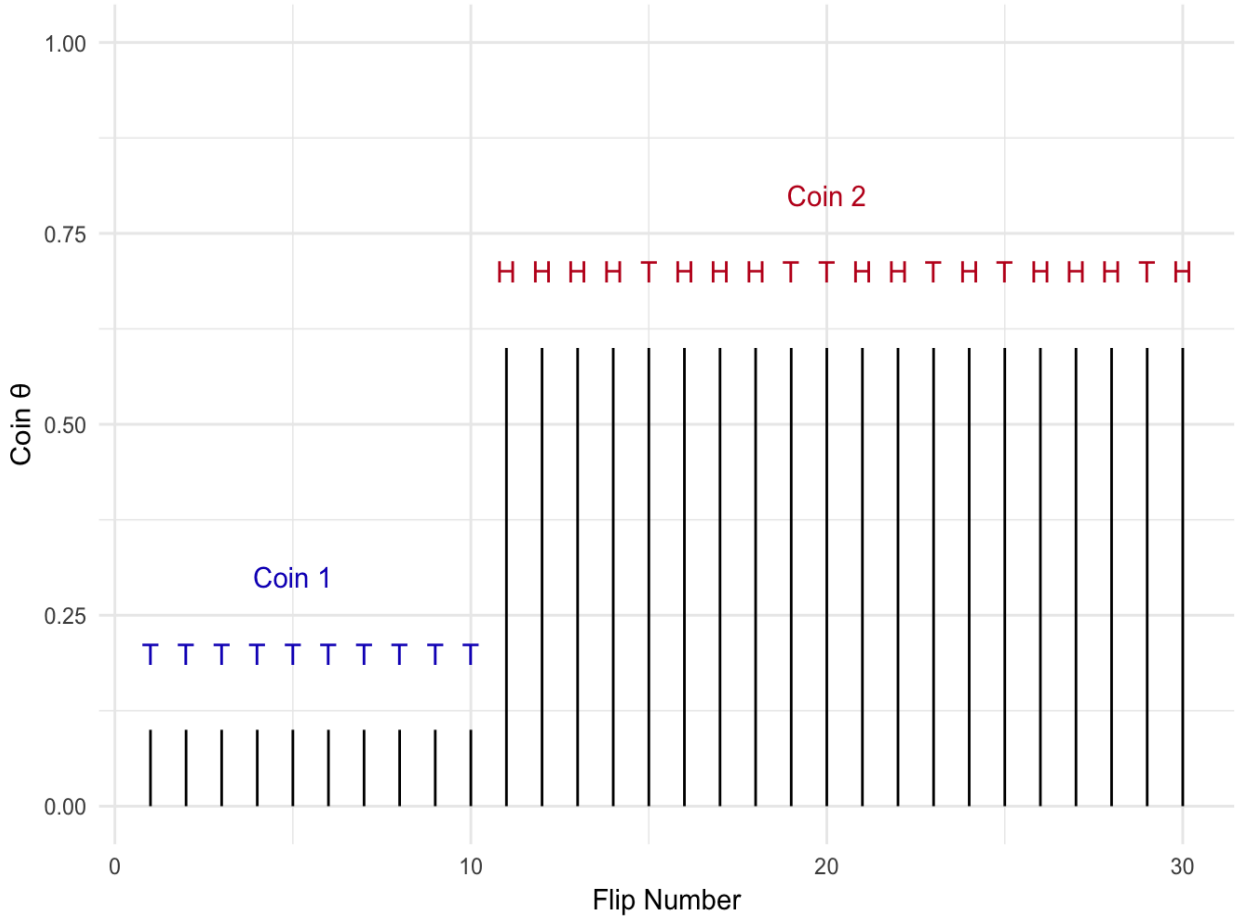
Sampling = Estimating

The coin flip example solved via Bayesian inference was capable of being solved analytically. However in many cases of Bayesian inference this is not possible due to the intractability of solving for the marginal likelihood (evidence term). Imagine we could sample from the posterior:



Change Point Example

What if the problem we are solving is more complicated? Let's say I flip a coin repeatedly, but at some point I switch to another coin with a different bias (θ). I want to detect the point in time when coin 1 was swapped out for coin 2.



Generative Model

In this case we have 3 variables that we need to estimate:

1. Coin bias for coin 1: θ_1
2. Coin bias for coin 2: θ_2
3. The point in time, i.e. on which flip, the coin was swapped: n

We assume that all values of n are equally probable and therefore can be modeled as a uniform distribution.

$$
\begin{aligned}
x &\sim \begin{cases} Bern(x_i; \theta_1) & 1 \leq i \leq n \\ Bern(x_i; \theta_2) & n < i < N \end{cases} \\
n &\sim Uniform(2...N) \\
\theta_i &\sim Beta(\theta_i, \alpha_i, \beta_i)
\end{aligned}
\tag{5}
$$

Joint Distribution

$$p(\theta_1, \theta_2, n|x_{1:N}) \propto \overbrace{p(x_{1:n}|\theta_1, n)p(x_{n+1:N}|\theta_2, n)}^{\textit{Likelihoods}} \overbrace{p(\theta_1)p(\theta_2)p(n)}^{\textit{Priors}} \tag{6}$$

$$\begin{aligned} p(\theta_1, \theta_2, n|x_{1:N}) &\propto \prod_{1}^{n} p(x_i|\theta_1) \prod_{n+1}^{N} p(x_i|\theta_2)p(\theta_1)p(\theta_2)p(n) \\ &\propto [\theta_1^{z_1}(1-\theta_1)^{n-z_1}][\theta_2^{z_2}(1-\theta_2)^{N-(n+1)-z_2}]p(\theta_1)p(\theta_2)p(n) \\ &\propto [\theta_1^{z_1}(1-\theta_1)^{n-z_1}][\theta_2^{z_2}(1-\theta_2)^{N-(n+1)-z_2}]\cdot \\ &\frac{\theta_1^{(\alpha_1-1)}(1-\theta_1)^{(\beta_1-1)}}{B(\alpha_1,\beta_1)} \frac{\theta_2^{(\alpha_2-1)}(1-\theta_2)^{(\beta_2-1)}}{B(\alpha_2,\beta_2)} \end{aligned} \tag{7}$$

Conditionals

Not clear how to sample from (7) at all ! We see that only the likelihood terms contain n. Using these terms we can solve for the posterior conditional.

$$p(n|x_{1:n}, \theta_1, \theta_2) \propto [\theta_1^{z_1}(1-\theta_1)^{n-z_1}][\theta_2^{z_2}(1-\theta_2)^{N-(n+1)-z_2}] \tag{8}$$

We utilize the conjugate prior relationship between the likelihoods and the priors and collapse the priors and likelihoods for the θ values.

$$\begin{aligned} p(\theta_1, \theta_2, n|x_{1:N}) &\propto [\theta_1^{(z_1+\alpha_1-1)}(1-\theta_1)^{(n-z_1+\beta_1-1)}][\theta_2^{(z_2+\alpha_2-1)}(1-\theta_2)^{(N-n-1-z_2+\beta_2-1)}] \\ &\propto Beta(\alpha_1+z_1, n-z_1+\beta_1)Beta(z_2+\alpha_2, N-n-1-z_2+\beta_2) \end{aligned} \tag{9}$$

Conditional Posteriors

We can now solve for each of the θ's posterior conditionals.

$$p(\theta_1|x_{1:n}, \theta_2, n) \propto Beta(a_1 + z_1, n - z_1 + b_1) \tag{10}$$

$$p(\theta_2|x_{1:N}, \theta_1, n) \propto Beta(z_2 + a_2, N - n - 1 - z_2 + b_2) \tag{11}$$

So what is this good for ?

Gibbs Sampler

Gibbs Sampling

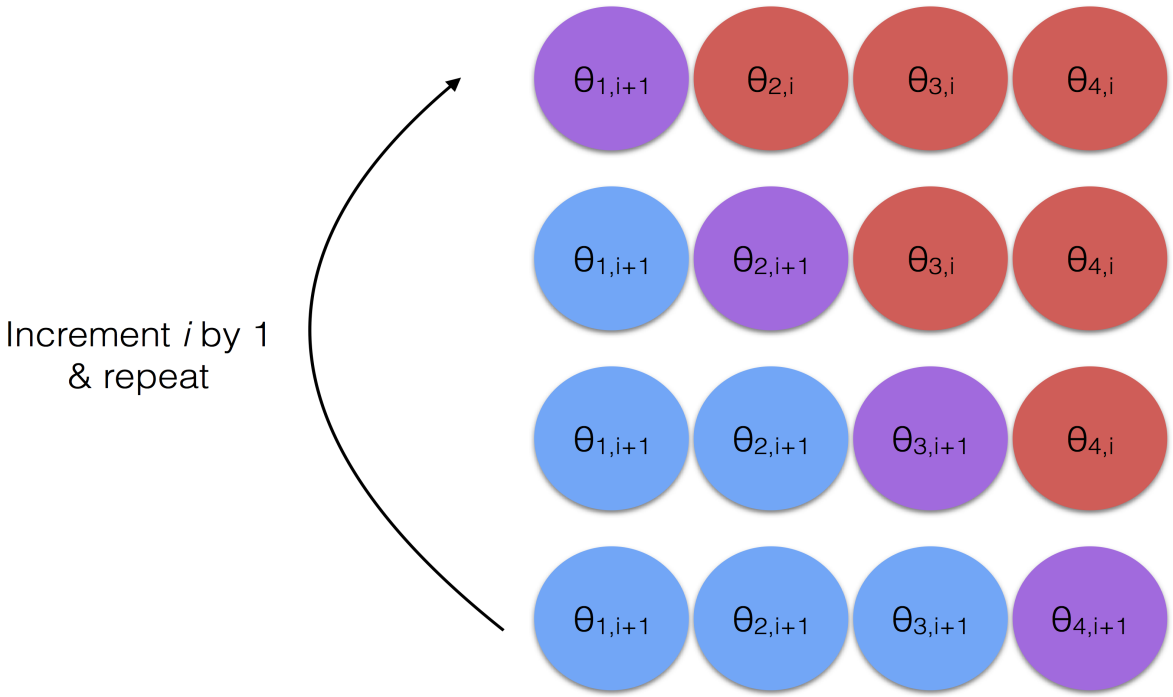
Gibbs sampling works by estimating all parameters via the posterior conditional iteratively for a set number of iterations or a distinct stopping criteria/convergence measure.

For i in iterations :

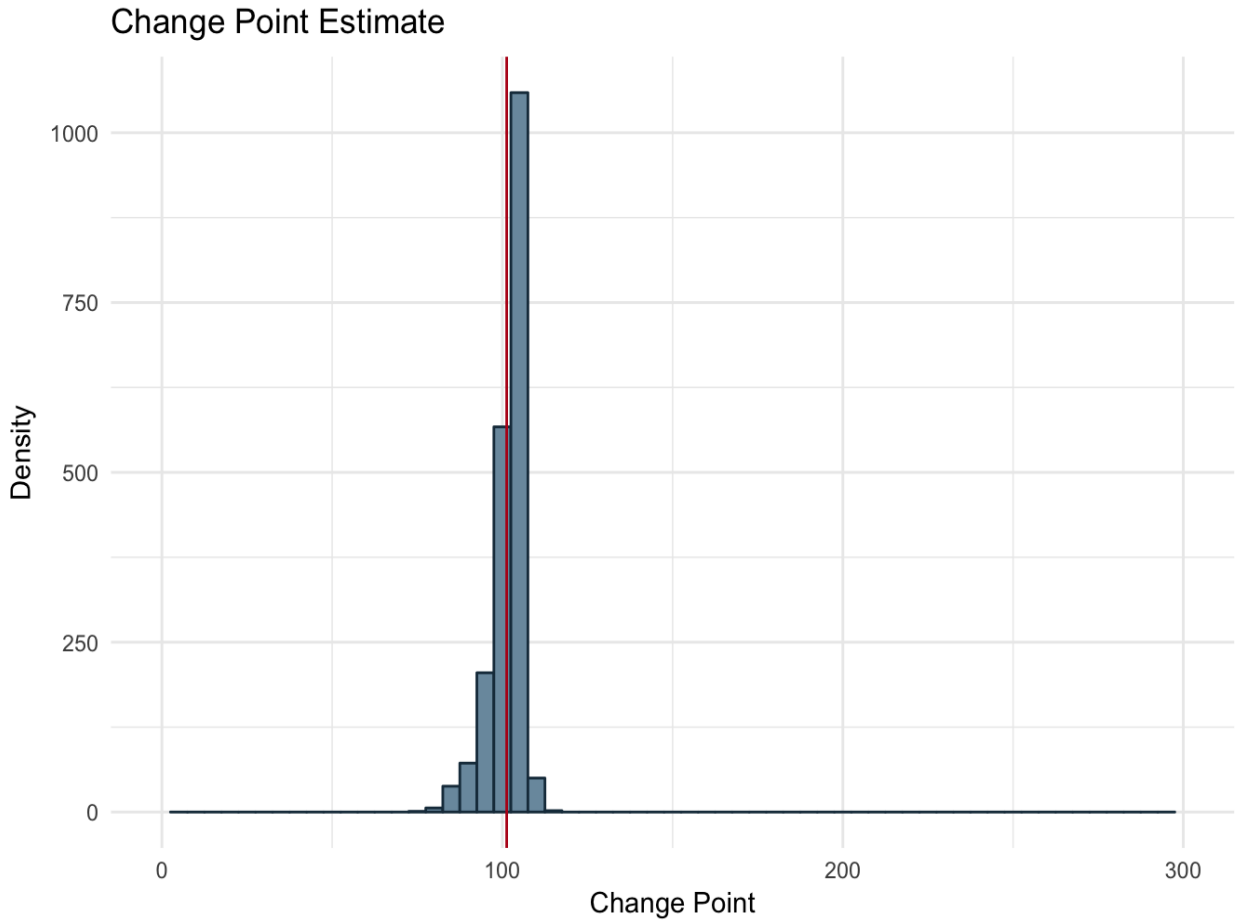
$$
\begin{aligned}
p(\theta_1^{i+1}) &\sim p(\theta_1^i|\theta_2^i, \theta_3^i, ..., \theta n^i) \\
p(\theta_2^{i+1}) &\sim p(\theta_2^i|\theta_1^{i+1}, \theta_3^i, ..., \theta n^i) \\
p(\theta_3^{i+1}) &\sim p(\theta_3^i|\theta_1^{i+1}, \theta_2^{i+1}, ..., \theta n^i) \\
&..... \\
p(\theta_n^{i+1}) &\sim p(\theta_n^i|\theta_1^{i+1}, \theta_2^{i+1}, ..., \theta_{n-1}^{i+1})
\end{aligned}
\tag{12}
$$

Gibbs Sampling

The red circles represent the parameters yet to be estimated in this iteration where the blue represent those that have been previously estimated during the current iteration. Note that the purple circle in each row is the parameter currently being estimated in that step (the current row) and that it takes into account all the available info, i.e. all the red and blue circles in that row.



Base R Code


Change Point Estimate
1000
750
500
250
0
Density
0
100
200
300
Change Point

Latent Variables

The above task is trivial if one knew the value of n beforehand. In fact, a human would eye-ball the change point and quickly compute the 2 separate means.

In machine learning, we would say that there is a *latent variable* (n) which is missing but knowledge of which greatly simplifies the problem!

Instead of viewing the above as a change point detection, the task can be phrased as a clustering problem: find the best partition into 2 groups.

Let us briefly review the most basic clustering algorithm.

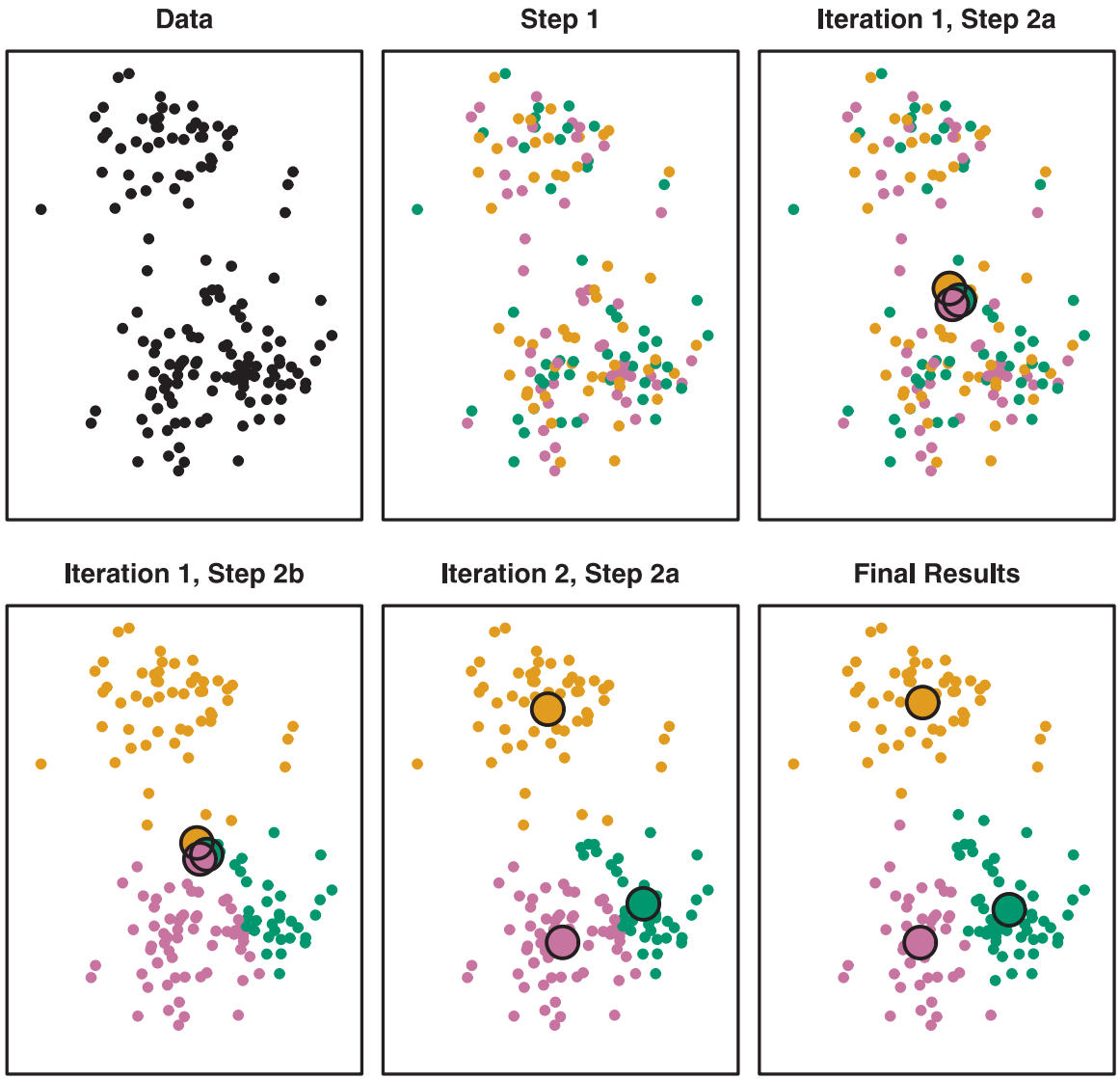
K-means Clustering

One chooses the desired number of cluster centers, say R, and the K-means procedure iteratively moves the centers to minimize the total within cluster variance. Given an initial set of centers, the K-means algorithm alternates the two steps:

1. for each center we identify the subset of training points (its cluster) that is closer to it than any other center;
2. the means of each feature for the data points in each cluster are computed, and this mean vector becomes the new center for that cluster.

These two steps are iterated until convergence. Typically the initial centers are R randomly chosen observations from the training data

Iterative Procedure



The EM Algorithm

Gaussian Mixtures as Soft K-means Clustering

Generative Model

$$
\begin{aligned}
Y_1 &\propto N(\mu_1, \sigma_1) \\
Y_2 &\propto N(\mu_2, \sigma_2) \\
Y &= \Delta \cdot Y_1 + (1 - \Delta) \cdot Y_2
\end{aligned}
\tag{13}
$$

Direct maximization of likelihood is difficult numerically, but if we knew the **unobserved latent** variables Δ, it would be simple.

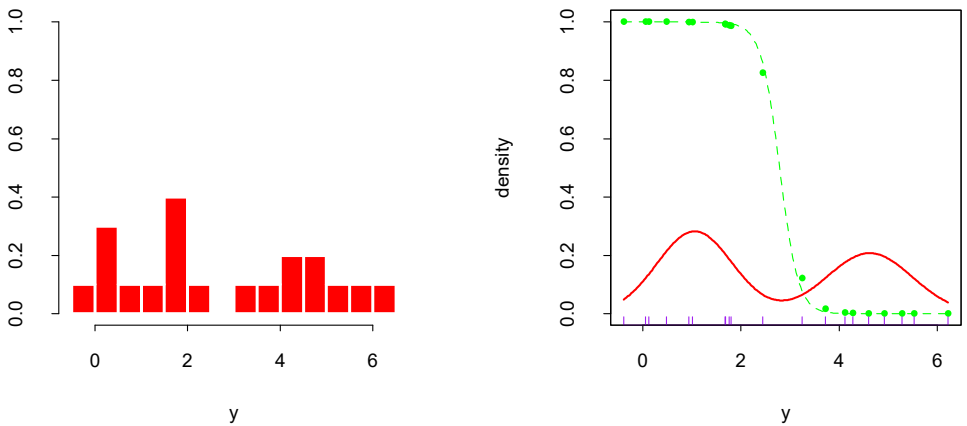


FIGURE 8.5. *Mixture example. (Left panel:) Histogram of data. (Right panel:) Maximum likelihood fit of Gaussian densities (solid red) and responsibility (dotted green) of the left component density for observation y, as a function of y.*

The EM algorithm

Algorithm 8.1 *EM Algorithm for Two-component Gaussian Mixture.*

1. Take initial guesses for the parameters $\hat{\mu}_1, \hat{\sigma}_1^2, \hat{\mu}_2, \hat{\sigma}_2^2, \hat{\pi}$ (see text).

2. *Expectation Step*: compute the responsibilities

$$\hat{\gamma}_i = \frac{\hat{\pi}\phi_{\hat{\theta}_2}(y_i)}{(1-\hat{\pi})\phi_{\hat{\theta}_1}(y_i) + \hat{\pi}\phi_{\hat{\theta}_2}(y_i)}, \ i = 1, 2, \ldots, N. \tag{8.42}$$

3. *Maximization Step*: compute the weighted means and variances:

$$\hat{\mu}_1 = \frac{\sum_{i=1}^N (1-\hat{\gamma}_i) y_i}{\sum_{i=1}^N (1-\hat{\gamma}_i)}, \qquad \hat{\sigma}_1^2 = \frac{\sum_{i=1}^N (1-\hat{\gamma}_i)(y_i - \hat{\mu}_1)^2}{\sum_{i=1}^N (1-\hat{\gamma}_i)},$$
$$\hat{\mu}_2 = \frac{\sum_{i=1}^N \hat{\gamma}_i y_i}{\sum_{i=1}^N \hat{\gamma}_i}, \qquad \hat{\sigma}_2^2 = \frac{\sum_{i=1}^N \hat{\gamma}_i (y_i - \hat{\mu}_2)^2}{\sum_{i=1}^N \hat{\gamma}_i},$$

   and the mixing probability $\hat{\pi} = \sum_{i=1}^N \hat{\gamma}_i / N$.

4. Iterate steps 2 and 3 until convergence.

Gibbs and EM

Gibbs sampling is closely related to the EM algorithm: the main difference is that it samples from the conditional distributions rather than maximizing over them.

Algorithm 8.4 *Gibbs sampling for mixtures.*

1. Take some initial values $\theta^{(0)} = (\mu_1^{(0)}, \mu_2^{(0)})$.
2. Repeat for $t = 1, 2, \ldots, .$
   (a) For $i = 1, 2, \ldots, N$ generate $\Delta_i^{(t)} \in \{0, 1\}$ with $\Pr(\Delta_i^{(t)} = 1) = \hat{\gamma}_i(\theta^{(t)})$, from equation (8.42).
   (b) Set

$$\begin{aligned} \hat{\mu}_1 &= \frac{\sum_{i=1}^N (1 - \Delta_i^{(t)}) \cdot y_i}{\sum_{i=1}^N (1 - \Delta_i^{(t)})}, \\ \hat{\mu}_2 &= \frac{\sum_{i=1}^N \Delta_i^{(t)} \cdot y_i}{\sum_{i=1}^N \Delta_i^{(t)}}, \end{aligned}$$

   and generate $\mu_1^{(t)} \sim N(\hat{\mu}_1, \hat{\sigma}_1^2)$ and $\mu_2^{(t)} \sim N(\hat{\mu}_2, \hat{\sigma}_2^2)$.
3. Continue step 2 until the joint distribution of $(\boldsymbol{\Delta}^{(t)}, \mu_1^{(t)}, \mu_2^{(t)})$ doesn't change

Gibbs and EM, details

The key is to consider the latent data Z^m from the EM procedure to be another parameter for the Gibbs sampler. To make this explicit for the Gaussian mixture problem, we take our parameters to be (θ, Z^m). For simplicity we fix the variances σ_1^2, σ_2^2 and mixing proportion π at their maximum likelihood values so that the only unknown parameters in θ are the means μ_1 and μ_2. The Gibbs sampler for the mixture problem is given in Algorithm 8.4.

We see that steps 2(a) and 2(b) are the same as the E and M steps of the EM procedure, except that we sample rather than maximize. In step 2(a), rather than compute the maximum likelihood responsibilities $\gamma_i = E(\Delta_i|\theta, Z^m)$, the Gibbs sampling procedure simulates the latent data Δ_i from the distributions $Pr(\Delta_i|\theta, Z^m)$. In step 2(b), rather than compute the maximizers of the posterior $Pr(\mu_1, \mu_2, \Delta|Z)$ we simulate from the conditional distribution $Pr(\mu_1, \mu_2, \Delta|Z)$.

Multinomial

Multinomial <-> Dirichlet

How Multinomial and Bernoulli Relate

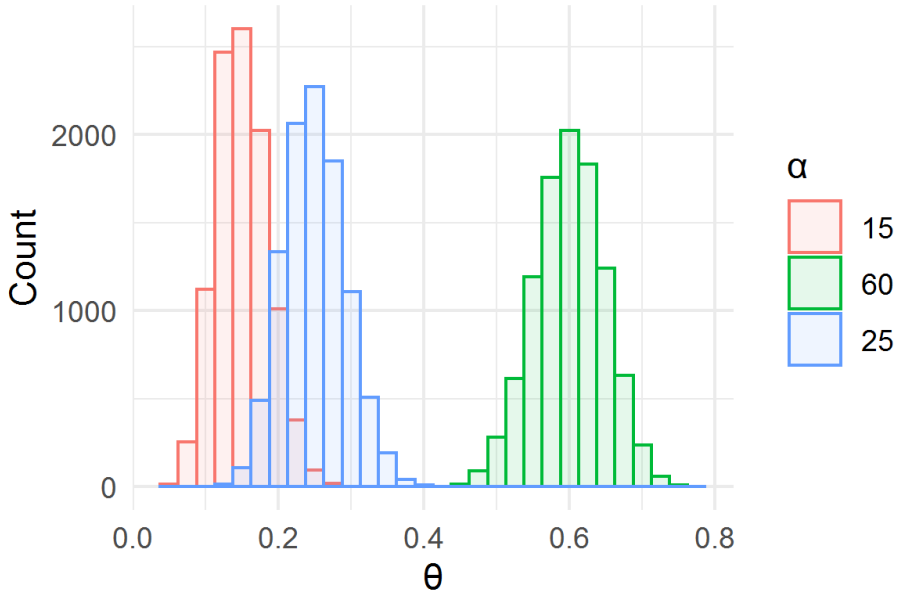
$$f(x) = \frac{n!}{x_1!x_2!\cdots x_K!}\theta_1^{x_1}\theta_2^{x_2}\cdots\theta_k^{x_K} \tag{14}$$

The **conjugate prior** for the multinomial distribution is the **Dirichlet** distribution. Similar to the beta distribution, Dirichlet can be thought of as a distribution of distributions. Also note that the beta distribution is the special case of a Dirichlet distribution where the number of possible outcomes is 2. This is similar to the relationship between the binomial and multinomial distributions.

$$Dir(\overrightarrow{\theta}\,|\,\overrightarrow{\alpha}) = \frac{\Gamma\left(\sum_{i=1}^{K}\alpha_i\right)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\prod_{i=1}^{K}\theta_i^{\alpha_i-1} = \frac{1}{B(\alpha)}\prod_{i=1}^{K}\theta_i^{\alpha_i-1} \tag{15}$$

Sampling from Dirichlet

The distribution of samples for each category (α_i value), are approximately centered at the ratio of the α_i value to the sum of all α values.



This is similar to the shift of the beta distribution when using hyperparameters that are unequal.

Dirichlet - Multinomial

The multinomial distribution with parameters $\overrightarrow{\theta} = \theta_1, \theta_2, ...\theta_n$ for words 1 to n would be capable of generating a bag of words (BOW) representation of a document. The model will be used to generate a document using a limited vocabulary, only 3 distinct words: Δ, Ω, Ψ., in particular using the word counts as our α values for the dirichlet prior. $\Delta\ \Delta\ \Delta\ \Delta\ \Delta\ \Delta\ \Delta\ \Delta\ \Omega\ \Psi$

Then we use the θ values generated by the dirichlet prior as the parameters for a multinomial distribution to generate the next term in the document.

1 $\Delta\ \Delta\ \Psi\ \Delta\ \Delta\ \Omega\ \Delta\ \Delta\ \Delta\ \Delta$
2 $\Psi\ \Delta\ \Omega\ \Delta\ \Delta\ \Psi\ \Delta\ \Delta\ \Omega\ \Delta$
3 $\Delta\ \Delta\ \Delta\ \Delta\ \Omega\ \Delta\ \Delta\ \Omega\ \Delta\ \Delta$
4 $\Delta\ \Psi\ \Delta\ \Psi\ \Delta\ \Delta\ \Omega\ \Omega\ \Delta\ \Delta$
5 $\Delta\ \Delta\ \Delta\ \Psi\ \Delta\ \Psi\ \Delta\ \Delta\ \Delta\ \Delta$

Inference I

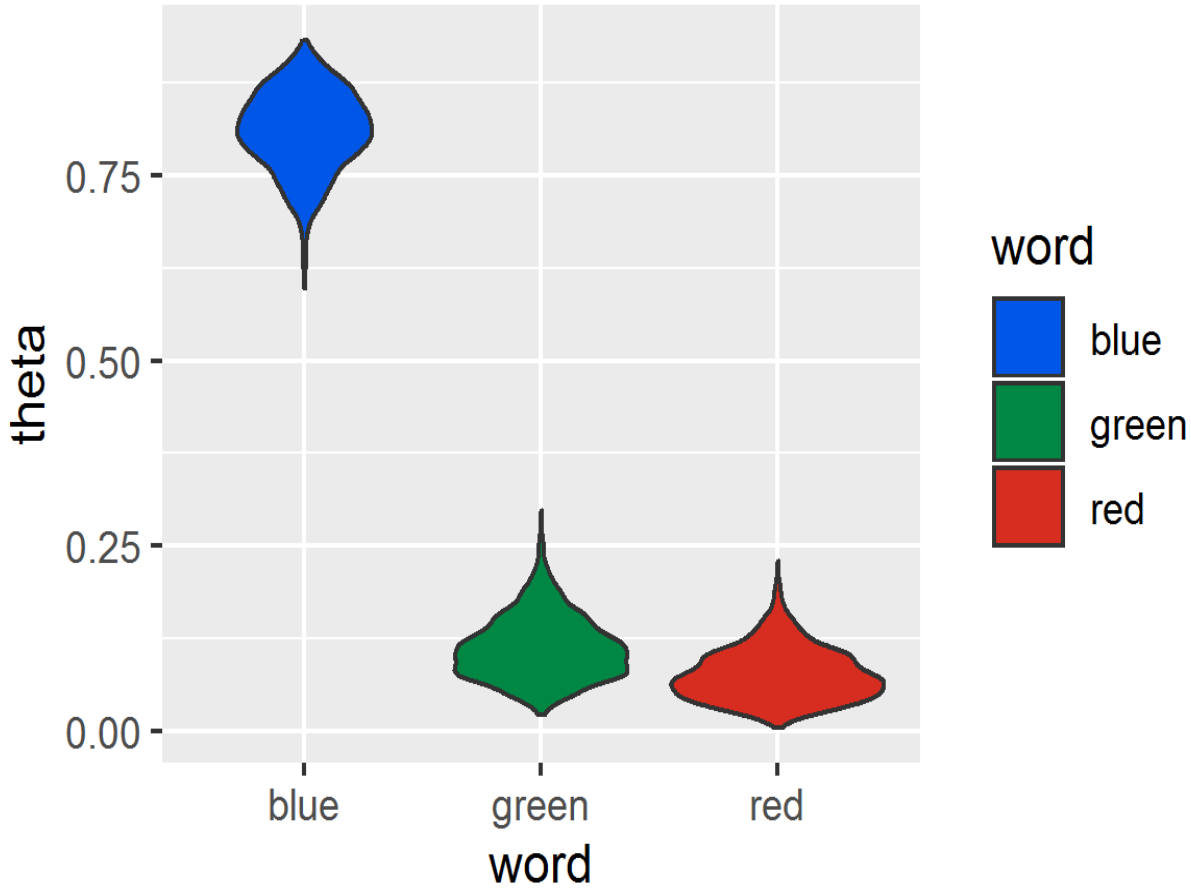
The process above is a **generative model**. Now we are going to take a series of pre-existing documents and infer what model created them

Once we plug in the prior and likelihood and simplify, we find that we are left with a Dirichlet PDF with the input parameters of $\overrightarrow{\alpha} + \overrightarrow{n}$ where n are the observed word counts.

$$
\begin{aligned}
p(\theta|D) &\propto p(D|\theta)p(\theta) \\
&\propto \prod_{i=1}^{K} \theta^{n(k)} \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} \theta_i^{\alpha_i - 1} \\
&\propto \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} \theta_i^{\alpha_i + n_k - 1} \\
&\propto Dir(\overrightarrow{\alpha} + \overrightarrow{n})
\end{aligned}
\tag{16}
$$

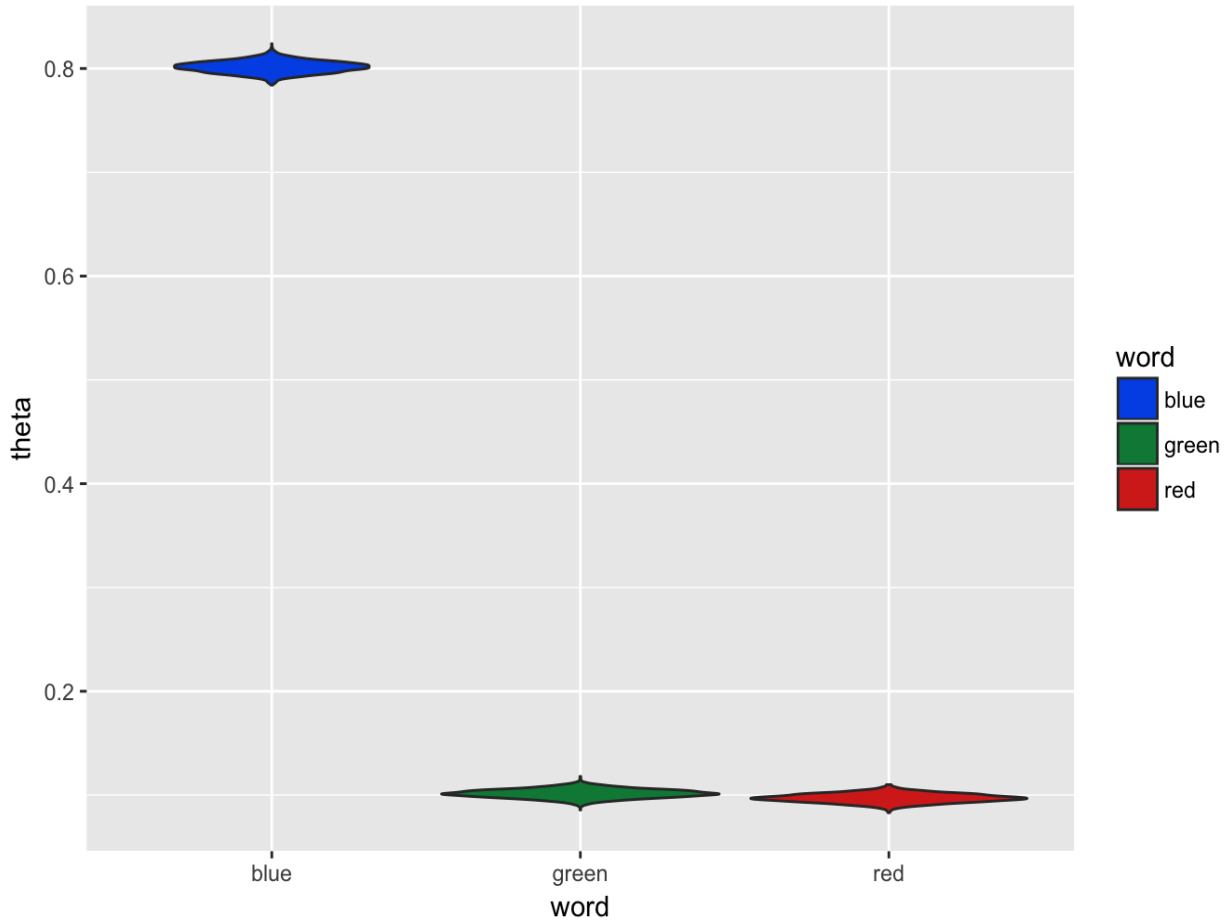
Inference II

Let's use the 5 documents we previously generated as our basis and infer the parameters used to generate them via Gibbs sampling.



Inference III

Instead of just 5 documents we will generate 500 and use this as our sample to infer the word mixtures from.



LDA

Latent Dirichlet allocation

Latent Dirichlet allocation is one of the most common algorithms for topic modeling. It is guided by two principles.

- **Every document is a mixture of topics.** We imagine that each document may contain words from several topics in particular proportions. For example, in a two-topic model we could say "Document 1 is 90% topic A and 10% topic B, while Document 2 is 30% topic A and 70% topic B."
- **Every topic is a mixture of words.** For example, we could imagine a two-topic model of American news, with one topic for "politics" and one for "entertainment." The most common words in the politics topic might be "President", "Congress", and "government", while the entertainment topic may be made up of words such as "movies", "television", and "actor". Importantly, words can be shared between topics; a word like "budget" might appear in both equally.

Mixture of Words **and** Topics

- **Documents (d in D):**, that we want to identify the topic structures of.
- **Words (w in W):** We have a collection of words and word counts for each document.
- **Hyperparameters:**
  - $\overrightarrow{\alpha}$: Our prior assumption about the topic distribution of our documents. We will be supplying the α value for inference.
    - Higher $\overrightarrow{\alpha}$ - We assume documents will have a similar and close to uniform distribution of topics.
    - Lower $\overrightarrow{\alpha}$ - We assume document topic distributions vary more drastically.
  - $\overrightarrow{\beta}$: Our prior assumption about the word distribution of each topic.
    - Higher $\overrightarrow{\beta}$: Word distributions in each topic are closer to uniform, i.e. each word is equally as likely in each topic.
    - Lower $\overrightarrow{\beta}$: Word distributions vary more from topic to topic.

Simplified View

- Go through each document d, and randomly assign each word in the document to one of the K topics.
- For each document d,
  - For each word w in d, and each topic t, compute:
    1. $p(t|d)$ = proportion of words in document d that are currently assigned to topic t, and
    2. $p(w|t)$ = proportion of assignments to topic t over all documents that come from this word w.
    3. Reassign w a new topic t with the probability that topic t generated word w: $p(t|d) \cdot p(w|t)$

In the last step, we're assuming that all topic assignments except for the current word in question are correct, and then updating the assignment of the current word using our model of how documents are generated.

Gibbs Sampling

The main goal of inference in LDA is to determine the topic of each word w, z_i (topic of word i), in each document: $p(z_i|z_{\neg i}, \alpha, \beta, w)$.

$$p(z_i|z_{\neg i}, w) \propto \overbrace{\frac{n^k_{d,\neg i} + \alpha_k}{\sum_k n^k_{d,\neg i} + \alpha_k}}^{document-topic} \cdot \overbrace{\frac{n^w_{k,\neg i} + \beta_w}{\sum_w n^w_{k,\neg i} + \beta_w}}^{topic-word} \tag{17}$$

where

- $n^k_{d,\neg i}$: Number of times document d use topic k
- $n^w_{k,\neg i}$: Number of times topic k uses the given word

Software & References

Software

- The **OpenBUGS** software (Bayesian inference Using Gibbs Sampling) does a Bayesian analysis of complex statistical models using Markov chain Monte Carlo.
- **JAGS** (Just another Gibbs sampler) is a GPL program for analysis of Bayesian hierarchical models using Markov Chain Monte Carlo.
- **PyMC3** is an open source Python library for Bayesian learning of general Probabilistic Graphical Model with advanced features and easy to use interface.
- **Turing** is a Julia package that allows multiple sampler types to be run as components of Gibbs sampling.

RJAGS

Closing Remarks

- Generally, samples from the beginning of the chain (the burn-in period) may not accurately represent the desired distribution and are usually discarded.
- A **blocked Gibbs sampler** groups two or more variables together and samples from their joint distribution conditioned on all other variables, rather than sampling from each one individually.
- A **collapsed Gibbs sampler** integrates out (marginalizes over) one or more variables when sampling for some other variable.
- Failure modes:
  - islands of high-probability states, with no paths between them
  - all states have nonzero probability and there is only a single island of high-probability states

Further References

- Jordan Boyd Graber at UMD:
  - Videos, e.g. Topic Models: Gibbs Sampling or Beta and Dirichlet Distributions or Clustering: Gaussian Mixture Models
  - book on Applications of Topic Models
- The Little LDA book
- The Elements of Statistical Learning
- Uni Koblenz, e.g. Animation Polya urn
- Tidy Text Mining, chapter 6