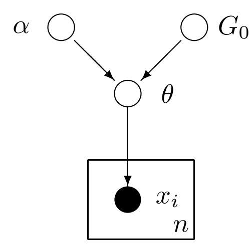
**CS281B/Stat241B: Advanced Topics in Learning & Decision Making**

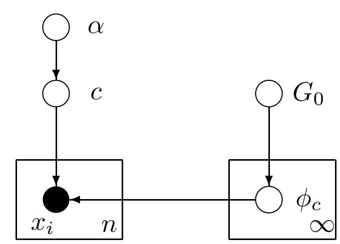
# DP Mixtures - Gibbs Sampling and Some Applications

*Lecturer: Michael I. Jordan* *Scribe: Michael Maire*

In the last lecture, we discussed two algorithms for Gibbs sampling in Dirichlet Process mixture models. These algorithms marginalized over different variables, yielding different submodels, as shown below. In the first algorithm, we marginalized out $G$ and sampled each $\theta_i$ given the other $\theta's$ (sampled $\theta_i|\theta_{-i}$). This was not computationally efficient as the algorithm had to update cluster information for one data point at a time. In the second algorithm, we introduced variables $c_i$ to represent the cluster assignments explicitly. Integrating out the $\theta_i$'s resulted in a relatively fast algorithm that mixes well. This is the standard Gibbs sampling algorithm for DP mixtures.



Graphical Model for Algorithm 1



Graphical Model for Algorithm 2

However, what if we don't need to recover explicit parameter values, but just want a clustering of the data points? Starting with the model for algorithm 2 shown above, one could integrate out the $\phi$'s and just update the $c_i$'s (cluster assignments). This is analogous to just keeping track of the table each data point is assigned to in the Chinese Restaurant process. Algorithm 3 below gives the details of this process.

**Algorithm 3**

- for $i = 1$ to $n$
  draw $c_i|c_{-i}, x$ from
  if $c = c_j$ for some $j \neq i$

  $$p(c_i = c|c_{-i}, x) \propto \frac{n_{-i,c}}{n-1+\alpha} \int F(x_i, \phi) dH_{-i,c}(\phi)$$

  where $H_{-i,c}$ is the posterior based on $G_0$ and those $x_j$ such that $j \neq i$ and $c_j = c$. This probability is that for joining an existing table. We can take advantage of conjugacy to compute $H$. Note that $F(x_i, \phi)$ represents the probability of the left-out data point $x_i$.

  else

  $$p(c_i \neq c_j \ \forall j \neq i | c_i, x) \propto \frac{\alpha}{n-1+\alpha} \int F(x_i, \phi) dG_0(\phi)$$

  (the probability for starting a new table)

- repeat, output $c_i$'s when done

**Application - Haplotype Phasing**

DP mixture models have applications to real clustering problems in which the number of clusters is unknown and it is not practical to guess and verify many different choices for the number of clusters. Haplotype phasing [1] is one such problem in biology.

A chromosome is a sequence of nucleotides (A, C, T, or G). Chromosomes are paired together, with individuals receiving one from each parent. Most genetic differences between individuals are explained by single nucleotide polymorphisms (SNPs). SNPs are almost always binary. This is because (1) Its hard for mutations to get into the population (most are deadly) and (2) Its unlikely that random mutations will occur next to each other. A haplotype is a listing of the SNPs along a chromosome.

Therefore, considering a section of length $k$ of a pair of chromosomes (a sequence of $k$ nucleotide pairs), there are $2^k$ possible haplotypes (allowing for only a binary SNP at each point). In practice, $k$ may be on the order of 10000, but not all $2^{10000}$ haplotypes occur in the human population (there aren't even $2^{10000}$ people, but moreover only a few mutations actually survive). In reality, let's say there are $K$ haplotypes. How big is $K$ for the human population?

If it were easy to sequence each of the two chromosomes, then we could do this by inspection. Unfortunately, this is an expensive process. A new chip has recently been developed to efficiently recover the genotype from chromosomes. The genotype is a sequence of nucleotide pairs. However, for each pair in the sequence, there is an ambiguity in terms of the chromosome from which each of the two nucleotides originated. Haplotype phasing is the process of recovering haplotypes from genotype data. Medical applications include discovering the locations of disease genes.

Taking a mixture model approach, given a genotype $g$,

$$p(g) = \sum_{h_1,h_2 \in H} p(h_1, h_2) 1(g = h_1 \oplus h_2)$$

where $p(h_1, h_2)$ is the probability of obtaining the two haplotypes $h_1$ and $h_2$ ($h_1$ is the haplotype for the first chromosome and $h_2$ the haplotype for the corresponding chromosome in the pair), and $1(g = h_1 \oplus h_2)$ is an indicator for whether the genotype $g$ is consistent with $h_1$ and $h_2$.

For this application, assuming independence between haplotypes turns out to be a reasonable approximation. This is equivalent to assuming that it is equally probable that any two people will mate. With this simplifying assumption, we have

$$p(g) = \sum_{h_1,h_2 \in H} p(h_1) p(h_2) 1(g = h_1 \oplus h_2)$$

A typical algorithm would fix the number of haplotypes $K$. Its easy to overfit the data with this approach as there is not a strong prior on the correct value of $K$.

A better strategy is to use a Dirichlet process. The data points $x$ are genotypes and the tables in the Chinese Restaurant analogy are haplotypes. In addition, we turn the indicator $1(g = h_1 \oplus h_2)$ into a probability $p(g = h_1 \oplus h_2)$ in order to account for possible errors.

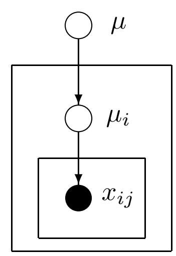
Question: Do we need to assume $p(h_1, h_2) = p(h_1)p(h_2)$?
Answer: No, you just need more data without the assumption as you have more parameters.

Question: What about chromosome crossovers?

Answer: We assumed we were taking a local region of the chromosome (length $k$) so we didn't have to deal with crossovers (they are unlikely to get into the population). A more elaborate model could handle them.

**Hierarchical Bayes**

Suppose we want to estimate the mean height of people in several different cities. A naive approach would be to estimate the mean height separately for each city by taking the sample mean of heights observed for that city. But this method fails to use the fact that individuals in all of the samples share the characteristic of being a person. Hence, they are exchangeable. Transferring information from one city to another results in a bias-variance trade off and can improve the estimate. A graphical model is shown below. The naive approach would lack the $\mu$ node, which is a prior on the mean height of a person.




$$\mu_i|\mu \sim N(\mu, \tau^2)$$

$$x_{ij}|\mu_i \sim N(\mu_i, \sigma^2)$$

Hierarchical Gaussian Means

Applying empirical Bayes to the above models yields the posterior estimate of the means

$$\hat{\mu}_i = \frac{\sigma^2}{n+\sigma^2}\hat{\mu}_{ML} + \frac{n}{n+\sigma^2}\bar{x}_i$$

where

$$\hat{\mu}_{ML} = \frac{1}{N}\sum_{i,j} x_{ij} \qquad \bar{x}_i = \frac{1}{n}\sum_j x_{ij}$$

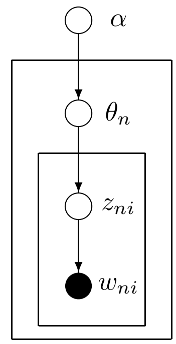
and $n$ is the number of data points in city $i$ and $N$ is the total number of data points.

**Application - Document Corpora**

A common assumption (made by Google, for example) used when modeling documents is the "bag-of-words" model. Specifically, there is a universal vocabulary $W = \{a_1, a_2, ..., a_{|W|}\}$ and each document is a set of words $w_i$ such that $w_i \in W$ for all $i$. This assumption is equivalent to exchangeability, and by De Finetti's Theorem, we can think in terms of an underlying $\theta$ parameter for each document.

In a "topic" document model, a topic is a distribution on words (a point inside a simplex where the corners are the words in the vocabulary). A document is then a distribution on topics, as shown in the graphical model below (for the case in which there are $K$ possible topics). This yields a mixture model for word probabilities,

$$p(w|\theta) = \sum_z p(z|\theta)p(w|z)$$




$\theta_n \sim Dir(\alpha)$

$z_{ni}$ - multinomial over $K$ topics

$w_{ni}$ - multinomial over $|W|$ words

Document Topic Model

Question: Why does a single topic only generate one word?
Answer: An alternative model could move the $z$'s out of the innermost box, but this is a clustering model in which a document is assigned to a single topic. In most applications, documents reflect multiple topics, so this is a less realistic model.

We would like to avoid the need to fix the number of topics in advance. This could be done by using a Dirichlet process to generate topics within each document. But this leads to a setup in which we have several Dirichlet processes, one per document. We would like the topics that are generated in the course of processing one document to also be available for other documents. I.e., we want some form of hierarchical Bayesian model in which multiple Dirichlet processes are linked. See [2, 3] for a description of such a Hierarchical Dirichlet process formalism.

# References

[1] E. P. Xing, R. Sharan and M. I. Jordan. Bayesian haplotype inference via the Dirichlet process. *Technical Report CSD-03-1275*, Division of Computer Science, University of California, Berkeley, 2003.

[2] D. M. Blei, T. Griffiths, M. I. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. *In press: Advances in Neural Information Processing Systems (NIPS)* 16, 2003.

[3] Y. W. Teh, M. I. Jordan, M. J. Beal and D. M. Blei. Hierarchical Dirichlet processes. *Technical Report 653*, Department of Statistics, University of California, Berkeley, 2004.