Statistical modeling with stochastic processes

Alexandre Bouchard-Côté

Lecture 7, Monday March 21

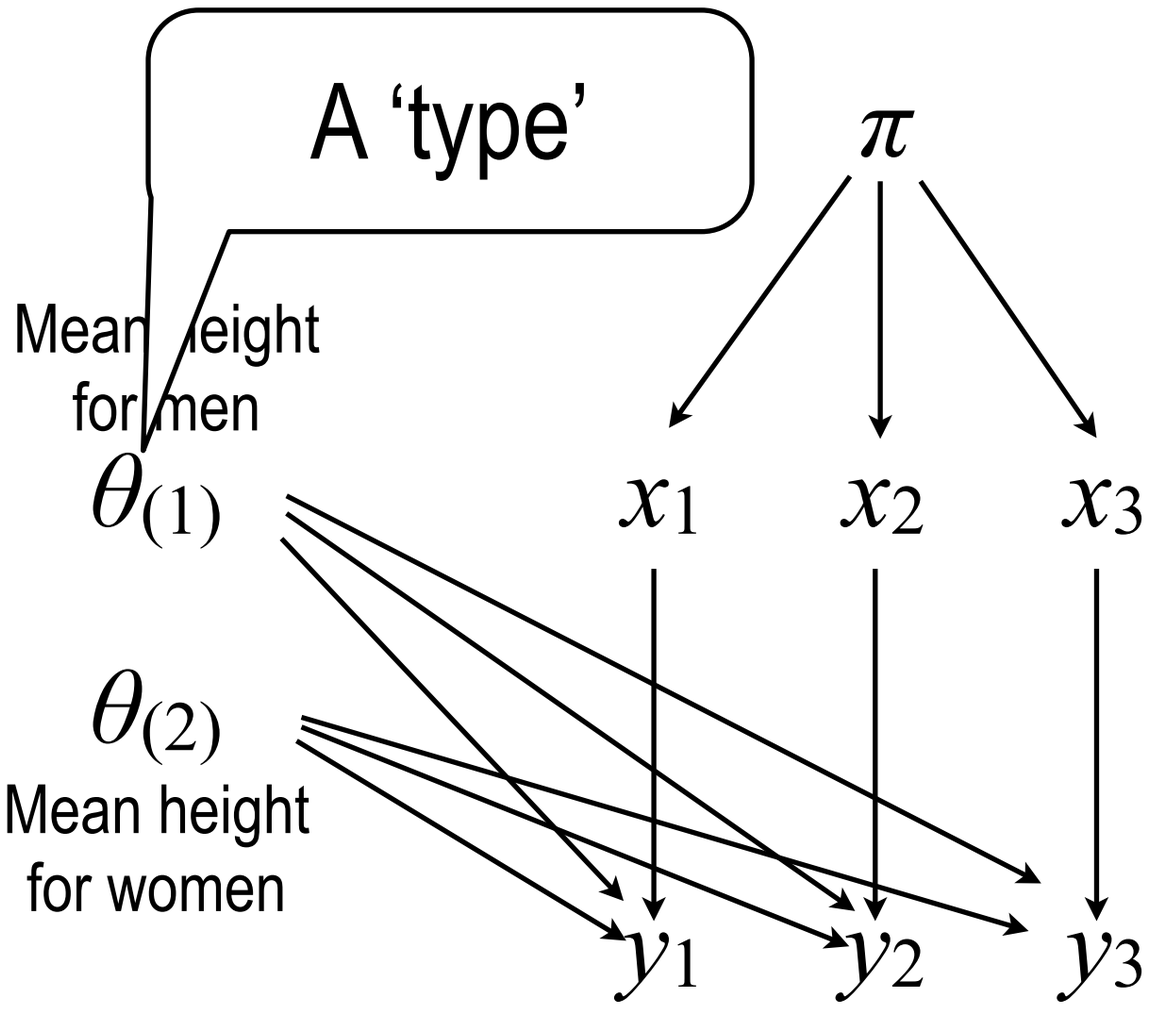
Program for today

- Introduction to Bayesian non-parametrics
  - Chinese Restaurant
- Basic probabilistic inference
  - Collapsed sampler
  - Slice sampler

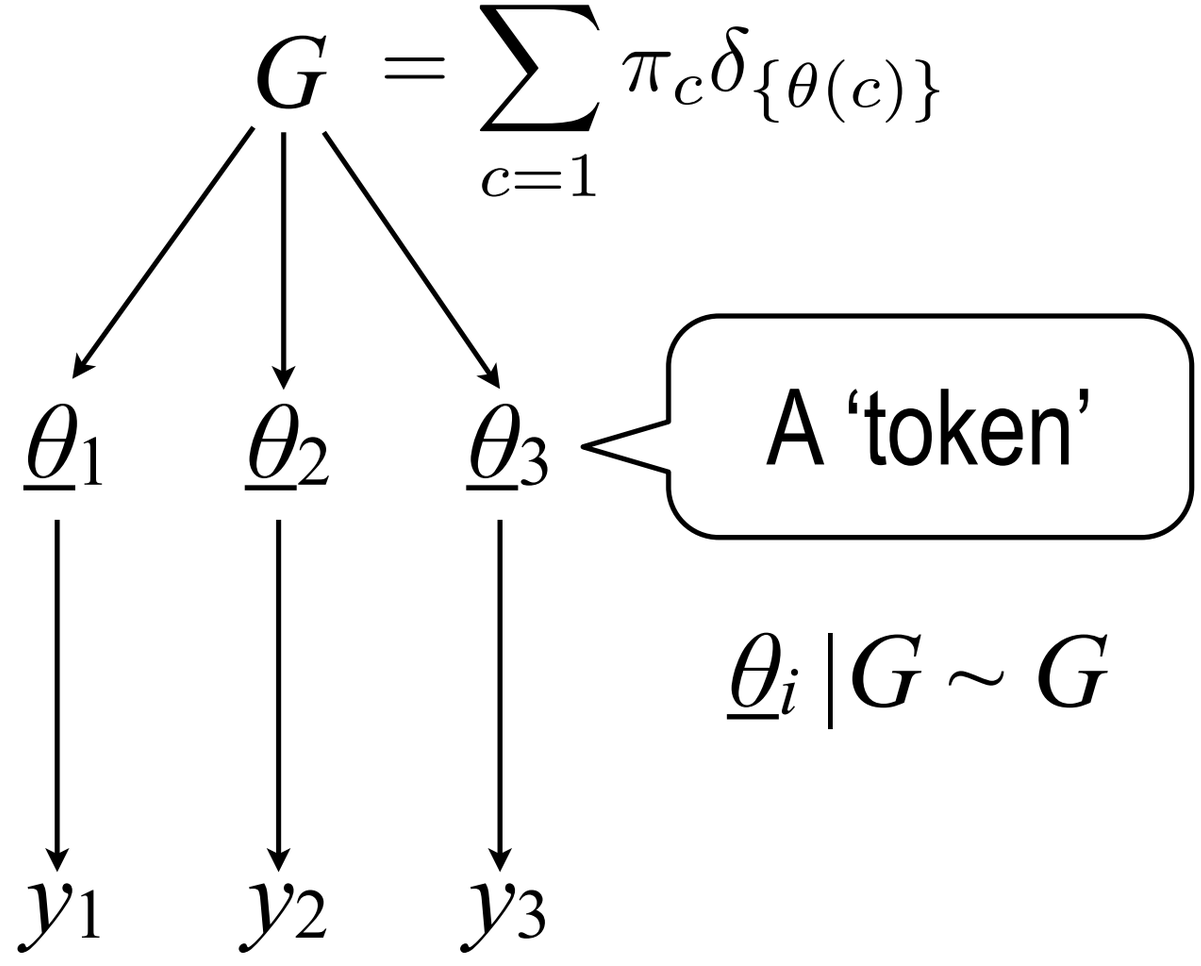
Review

Equivalent notation

Mixture model: (UBC student height with 2 components) say we have only 3 observations



$$G = \sum_{c=1}^{2} \pi_c \delta_{\{\theta(c)\}}$$

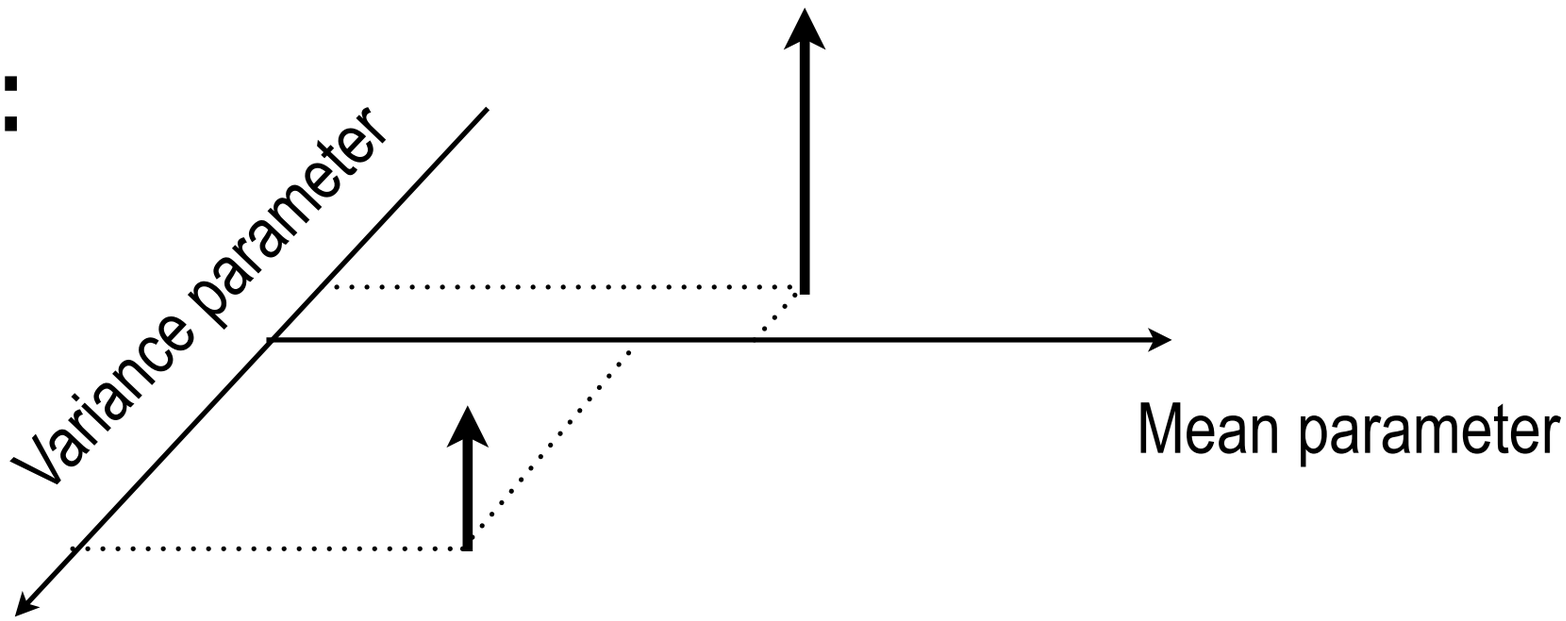


$$\underline{\theta}_i \,|\, G \sim G$$

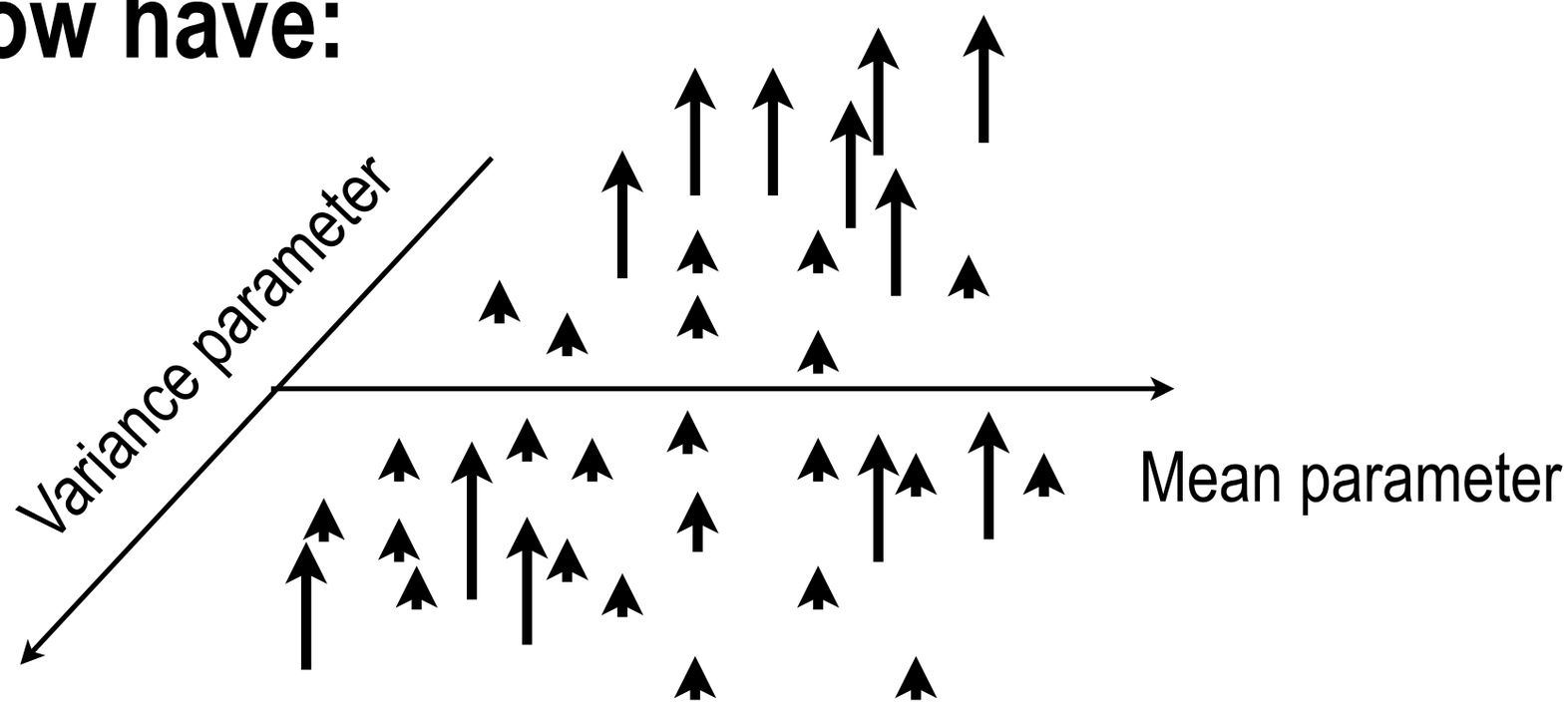
$$y_i \mid \underline{\theta}_i \sim \mathrm{N}(\underline{\theta}_i, \text{variance})$$

Samples from G

What we had:



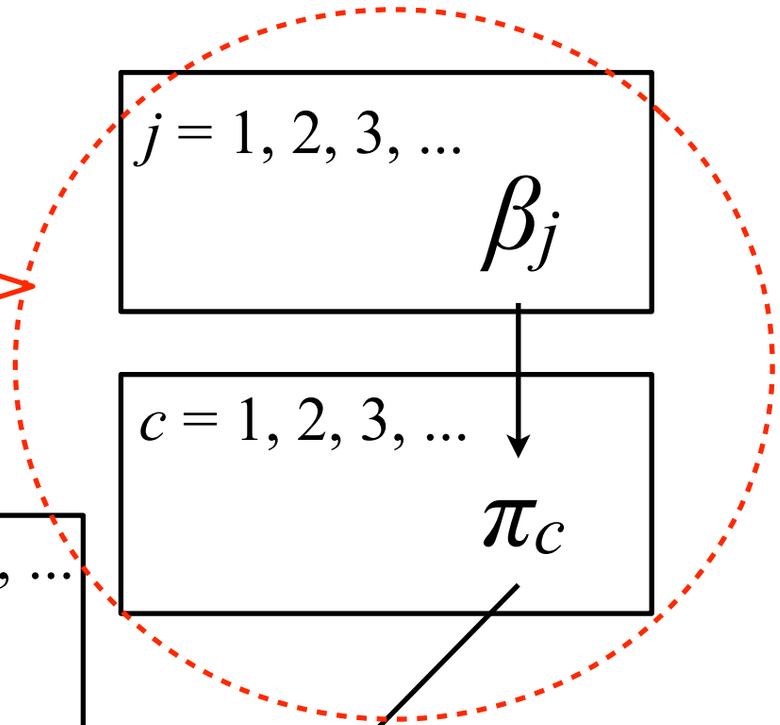
What we now have:



Constructive argument

$$\beta_j \overset{\text{iid}}{\sim} \text{Beta}(1, \alpha_0)$$

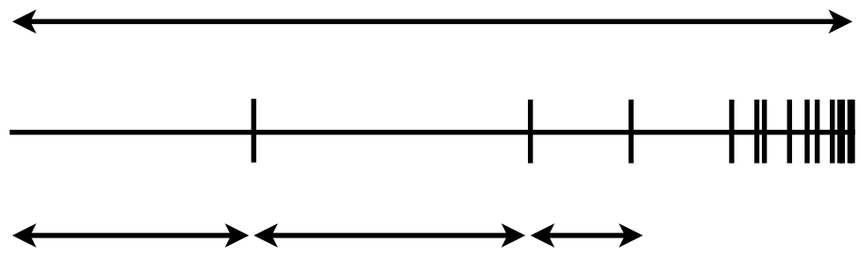
We will denote this distribution over π by GEM(α_0)



$$\theta_c \overset{\text{iid}}{\sim} G_0$$

Likelihood mixture component parameters

$$G' = \sum_{c=1}^{\infty} \pi_c \delta_{\{\theta(c)\}}$$



1

β_1

$\beta_2(1\text{-}\beta_1)$

$\beta_3(1\text{-}\beta_2)\ (1\text{-}\beta_1)$

Moments

Let $G \sim \mathrm{DP}(\alpha_0, G_0)$ and A be a measurable set

The first and second moments of $G(A)$:

- Mean: $G_0(A)$
- Variance: $G_0(A)\ G_0(A^c) / (\alpha_0 + 1)$

Conjugacy

Suppose now we have several draws from G:

$$\big(G(A_1),\ldots,G(A_k)\big)\,\big|\,\underline{\theta}_1,\ldots,\underline{\theta}_n \sim \mathrm{Dir}\big(\alpha_0 G_0(A_1)+n_1,\ldots,\alpha_0 G_0(A_k)+n_k\big)$$

where: $\quad n_j = \sum_{i=1}^{n} \delta_{\{\underline{\theta}_i\}}(A_j)$

Therefore the posterior parameters are:

$$\alpha_0' = \alpha_0 + n$$

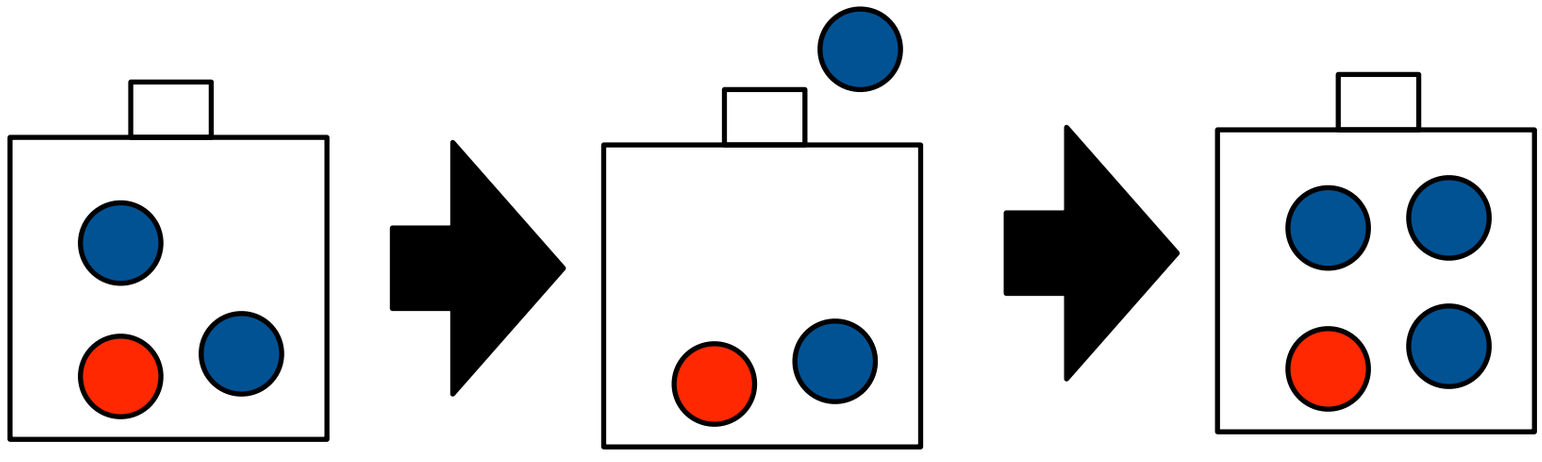
$$G_0' = \frac{\alpha_0}{\alpha_0+n} G_0 + \frac{1}{\alpha_0+n}\sum_{i=1}^{n} \delta_{\{\underline{\theta}_i\}}$$

More on Polya Urns

Review: Pólya Urn

Thought experiment: Consider an urn, with initially R red marbles and B blue marbles.

At each step, draw one marble at random, and put it back in the urn after adding another one of the same color



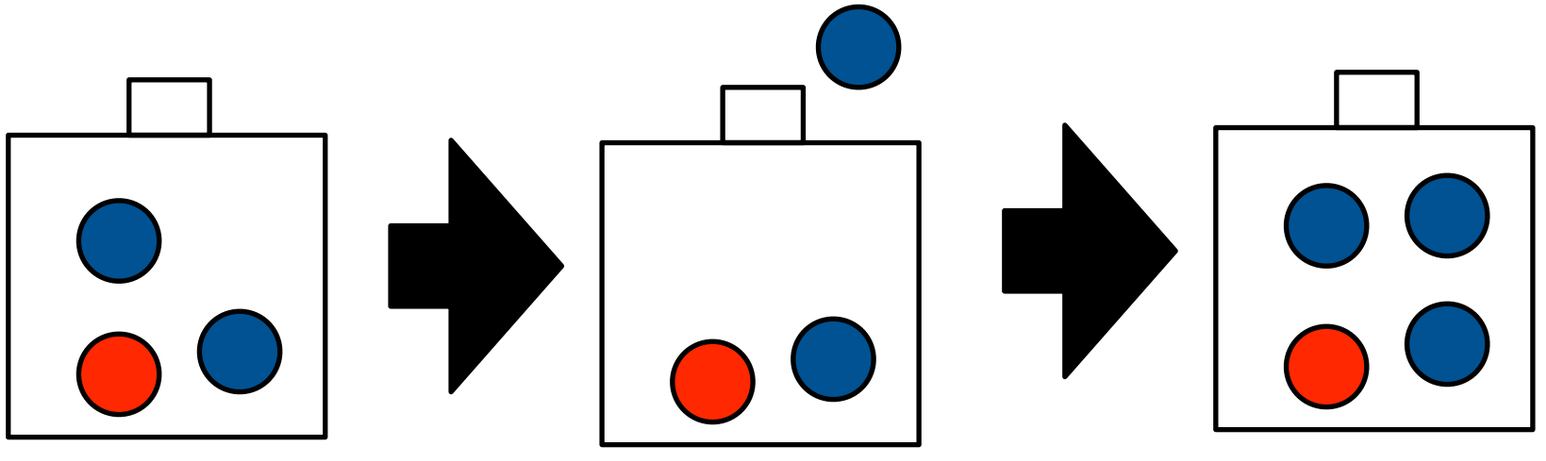
Question: does this process converge to a certain red:blue ratio? What is this ratio?

Review: Pólya Urn

Thought experiment: Consider an urn, with initially R red marbles and B blue marbles.

Question: does this process converge to a certain red:blue ratio? What is this ratio?

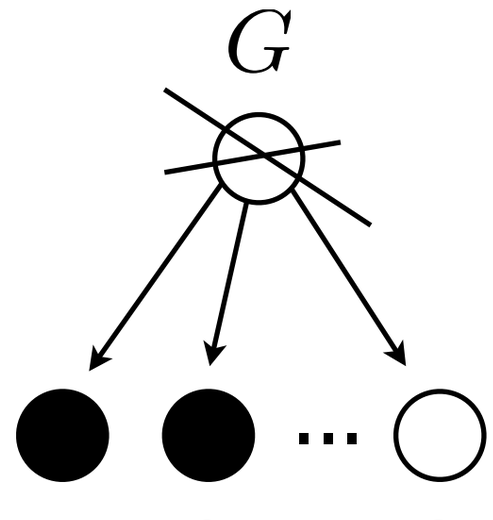
Soln: Let $\alpha_0 = R + B$ $\quad$ $G_0 = \text{Bern}\left(\frac{R}{B+R}\right)$



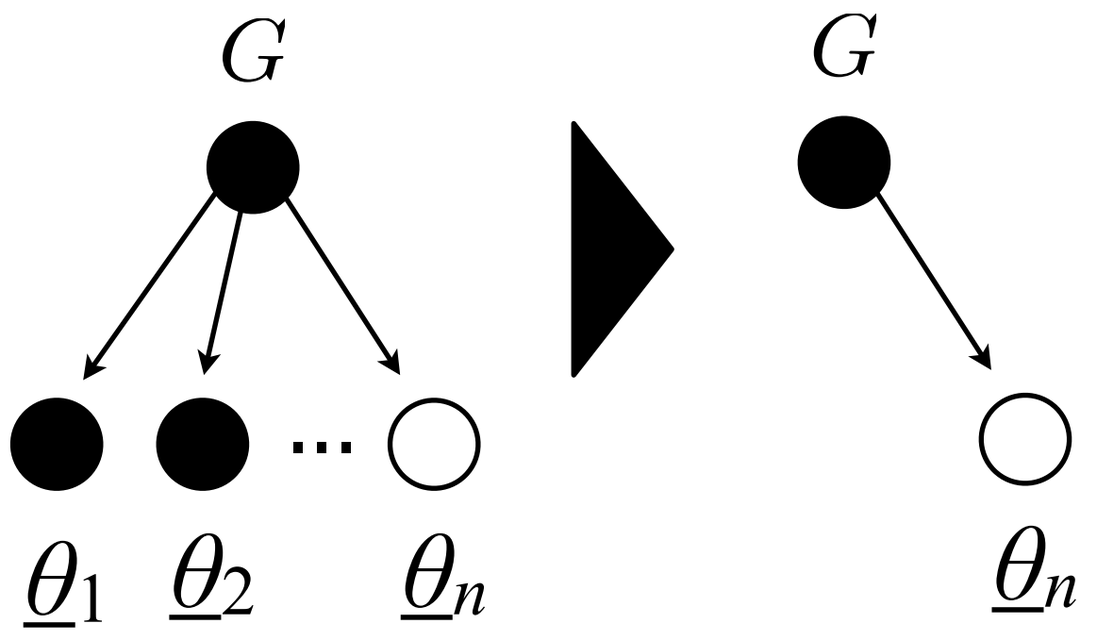
(Solution described on the board)

Another way of simulating Pólya Urns

Original description/sampling scheme:
An example of a collapsed (marginalized) sampler



Question: is it possible to describe an alternative algorithm that samples G (using the stick breaking representation) and then samples the observations $\underline{\theta}_1, \underline{\theta}_2, \ldots$?



Another way of simulating Pólya Urns

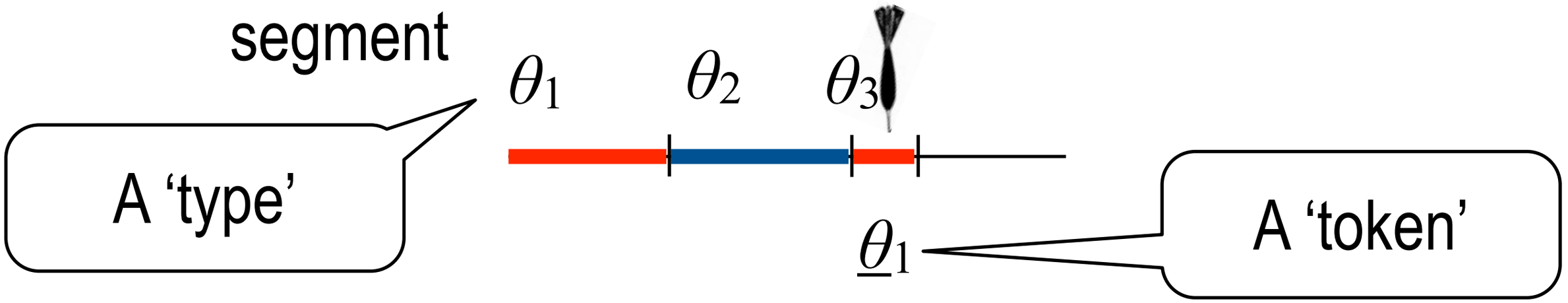
Description of the naive 'algorithm':

1. Sample a list of stick lengths $\pi_1, \pi_2, \ldots \sim \mathrm{GEM}(R+B)$
2. Sample one stick 'color' $\theta_1, \theta_2, \ldots$ for each stick segment using $G_0 = \mathrm{Bin}(\mathrm{R/R+B})$
3. To sample a ball, throw a dart on the stick and return the color $\underline{\theta}_1$ of the stick segment

Precision on notation

Description of the naive 'algorithm':

1. Sample a list of stick lengths $\pi_1, \pi_2, \ldots \sim \mathrm{GEM}(R+B)$
2. Sample one stick 'color' $\theta_1, \theta_2, \ldots$ for each stick segment using $G_0 = \mathrm{Bin}(\mathrm{R/R+B})$
3. To sample a ball, throw a dart on the stick and look at the index x of the stick
4. Return the color $\underline{\theta}_1 = \theta_x$ of the sampled stick segment



Another way of simulating Pólya Urns

Problem: running time (before getting first draw $\underline{\theta}_1$) is infinite

Solution: lazy computation

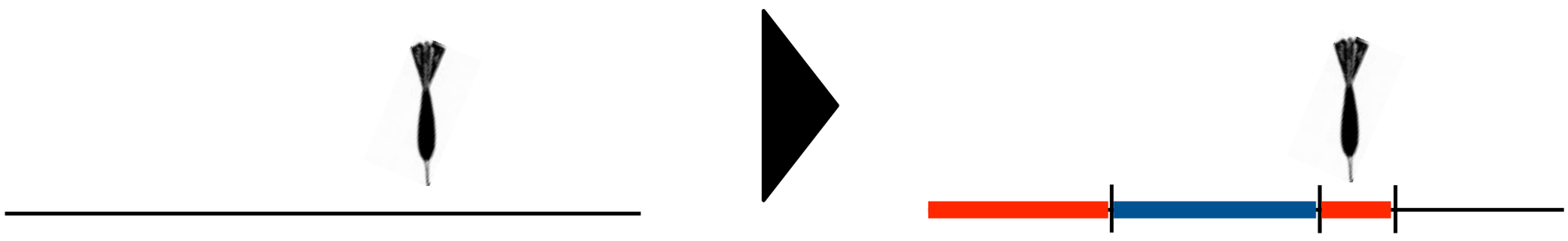
1. Sample a list of stick lengths $\pi_1, \pi_2, \ldots \sim \mathrm{GEM}(R+B)$
2. Sample one stick 'color' for each stick segment using $G_0 = \mathrm{Bin}(\mathrm{R}/\mathrm{R}+\mathrm{B})$
3. To sample an urn draw, throw a dart on the stick and return the color of the stick segment

Do this part only on demand (when it's absolutely required)

Another way of simulating Pólya Urns

Solution: lazy computation

1. Throw a dart
2. Sample the minimum number of
   π_i's ~ $\mathrm{GEM}(R+B)$ needed, and their colors
3. Return the color of the sampled segment



Another way of simulating Pólya Urns

Solution: lazy computation

1. Throw a dart
2. Sample the minimum number of π_i's ~ $\mathrm{GEM}(R+B)$ needed, and their color
3. Return the color of the sampled segment
4. Throw a second dart
5. Check if there is enough π_i's, and sample some more if needed
6. ...

Related metaphor: Chinese Restaurant Process

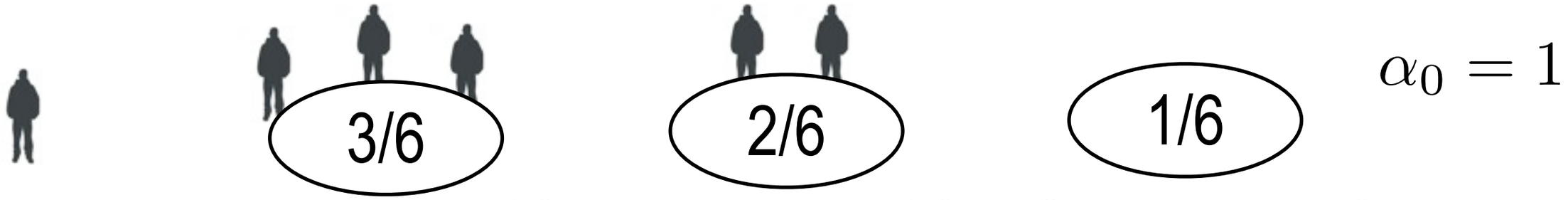
Chinese Restaurant Process (CRP)

Idea: Instead of balls sharing colors, think about customers sharing tables in an infinite restaurant

Initialization: The first customer sits in the first empty table.

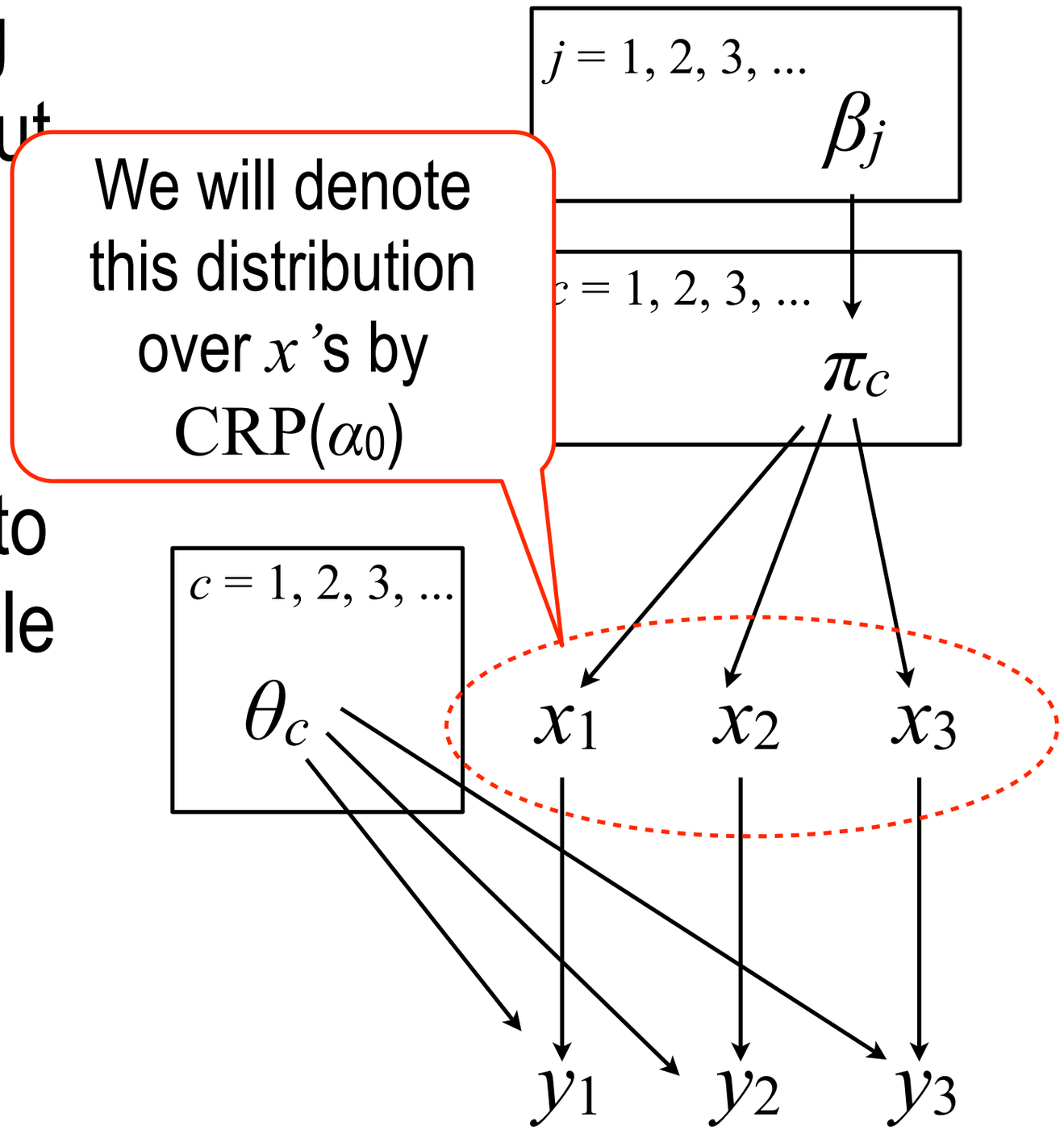
Iterate: If n **customers** are already sitting in the restaurant, the next customer starts a new table with probability $\alpha_0 / (\alpha_0 + n)$; otherwise the customer joins an existing table with probability proportional to the number of people already at the table

E.g.:



How the CRP relates to the DP & Polya urn

Consider the stick breaking representation of the DP, but only looking at the cluster indicators x_i



Variant: you can also add to the metaphor that each table sample a dish θ from G_0

- When G_0 is over two atoms, equivalent to Polya urn

Chinese Restaurant Process (CRP)

Notation: let x_n denote the table index for customer n. If there are t tables and customer n creates a new table, set $x_n = t + 1$, and suppose we have sampled according to the CRP:

$$x_1, x_2, ..., x_n \sim \text{CRP}(\alpha_0)$$

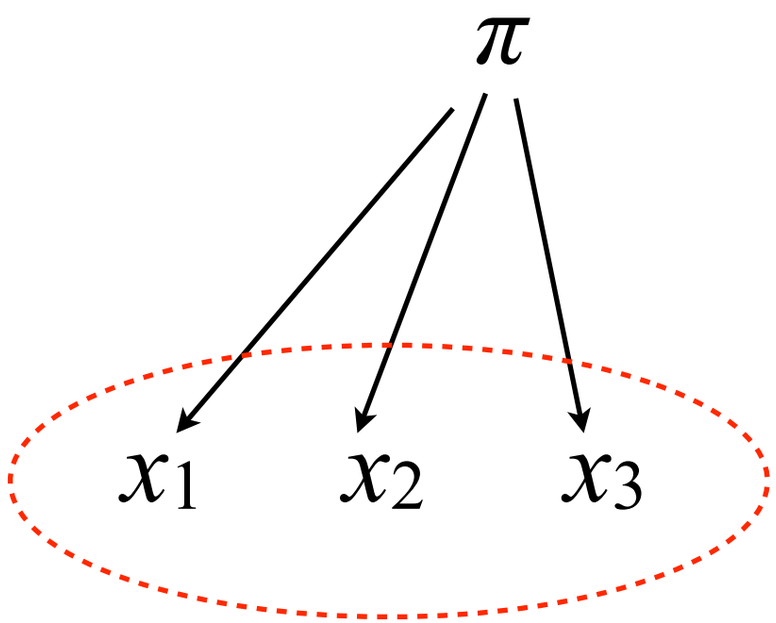
But for MCMC, we will need:

$$x_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$$

This *seems* hard: x_{i+1}, x_{i+2}, ..., x_n all seem to depend on the table x_i picked...

Exchangeability

Key observation: we know there is a prior π (the $\mathrm{GEM}(\alpha_0)$ stick breaking distribution) such that the x_i are iid conditionally on π



In other words: the fact that CRP emerges as the predictive distribution of the GEM means CRP is exchangeable: for all permutation σ,

$$x_1, x_2, \ldots, x_n \stackrel{d}{=} x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)}$$

Consequence of exchangeability

To compute

$$x_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$$

simply treat x_i as if it was the last customer to enter the restaurant! We know how to do this already!

Consequence of exchangeability

This also means that $\mathrm{CRP}(\alpha_0)$ can be viewed as a distribution over partitions ρ of {1, 2, ..., n}, $\mathrm{CRP}(\rho, \alpha_0)$

Labeled vs. unlabeled partitions

For (labeled) partitions: {1, 2}, {3} ≠ {1}, {2, 3}

For unlabeled partitions: {1, 2}, {3} = {1}, {2, 3}

Today: we will need labels so let's keep them for now

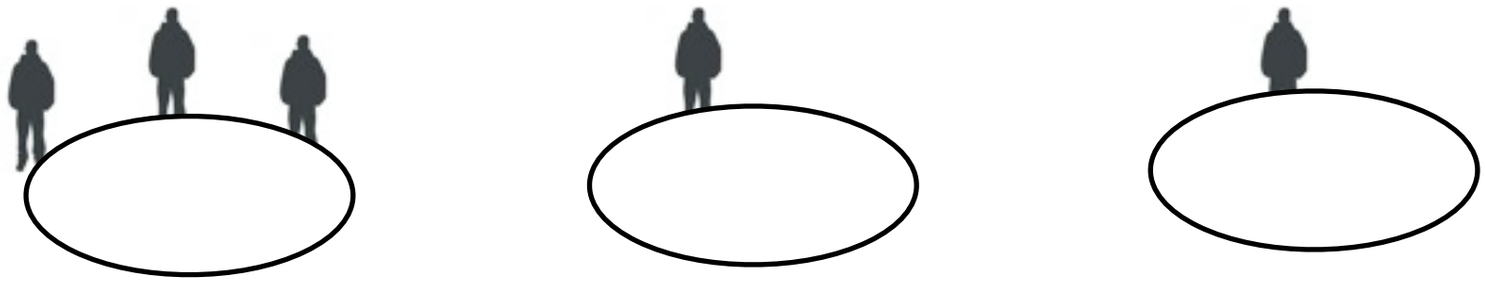
Later: we will see what we get when we remove the labels (Ewen's formula)

Consequence of exchangeability

This also means that $\mathrm{CRP}(\alpha_0)$ can be viewed as a distribution over *labeled partitions* ρ of {1, 2, ..., n}, $\mathrm{CRP}(\rho, \alpha_0)$

This is because the only information needed for computing $\mathrm{CRP}(x, \alpha_0)$ is the number of tables of each sizes

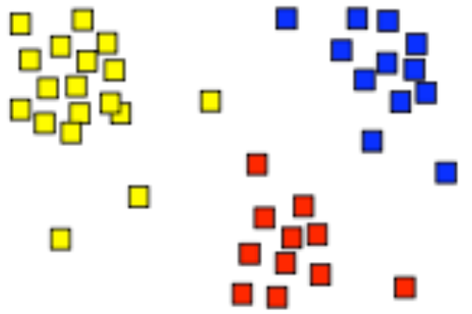
E.g.: all you need to know to compute the CRP(1) probability of the seating arrangement below is that there is one table with 3 customers, and 2 tables with 1 customer

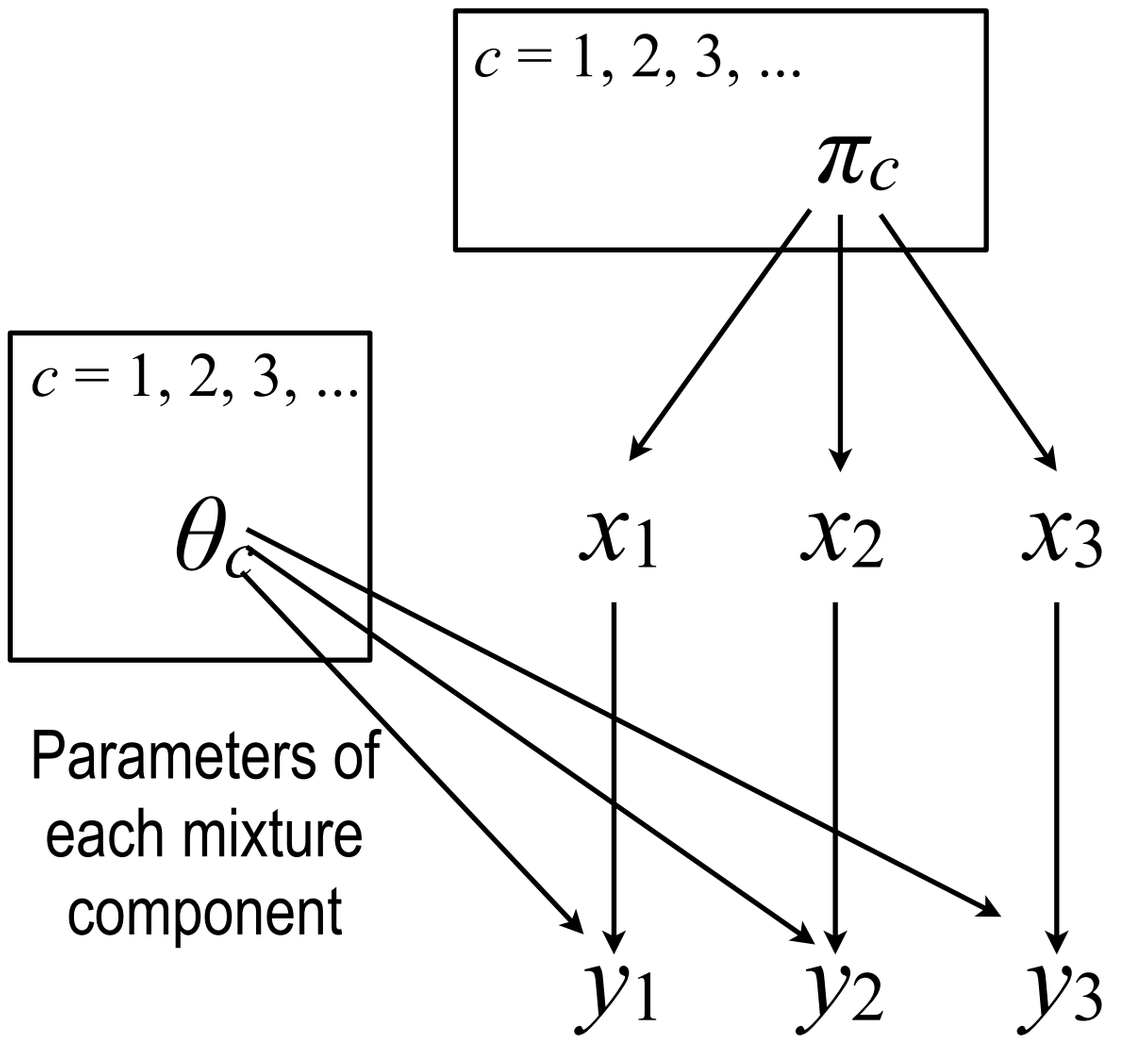


Dirichlet Processes applied to a statistical problem: cluster analysis

Clustering

Informally: Find 'clusters' or groups of related data points





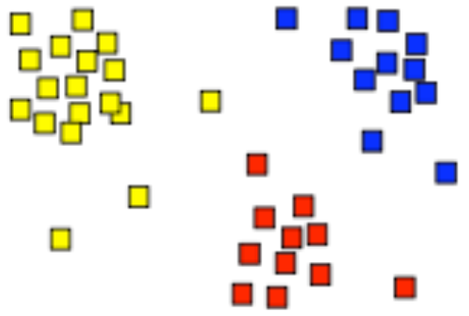
To make this formal, let's assume the data was generated from a *DP mixture model*. The task is to partition the data points into clusters.

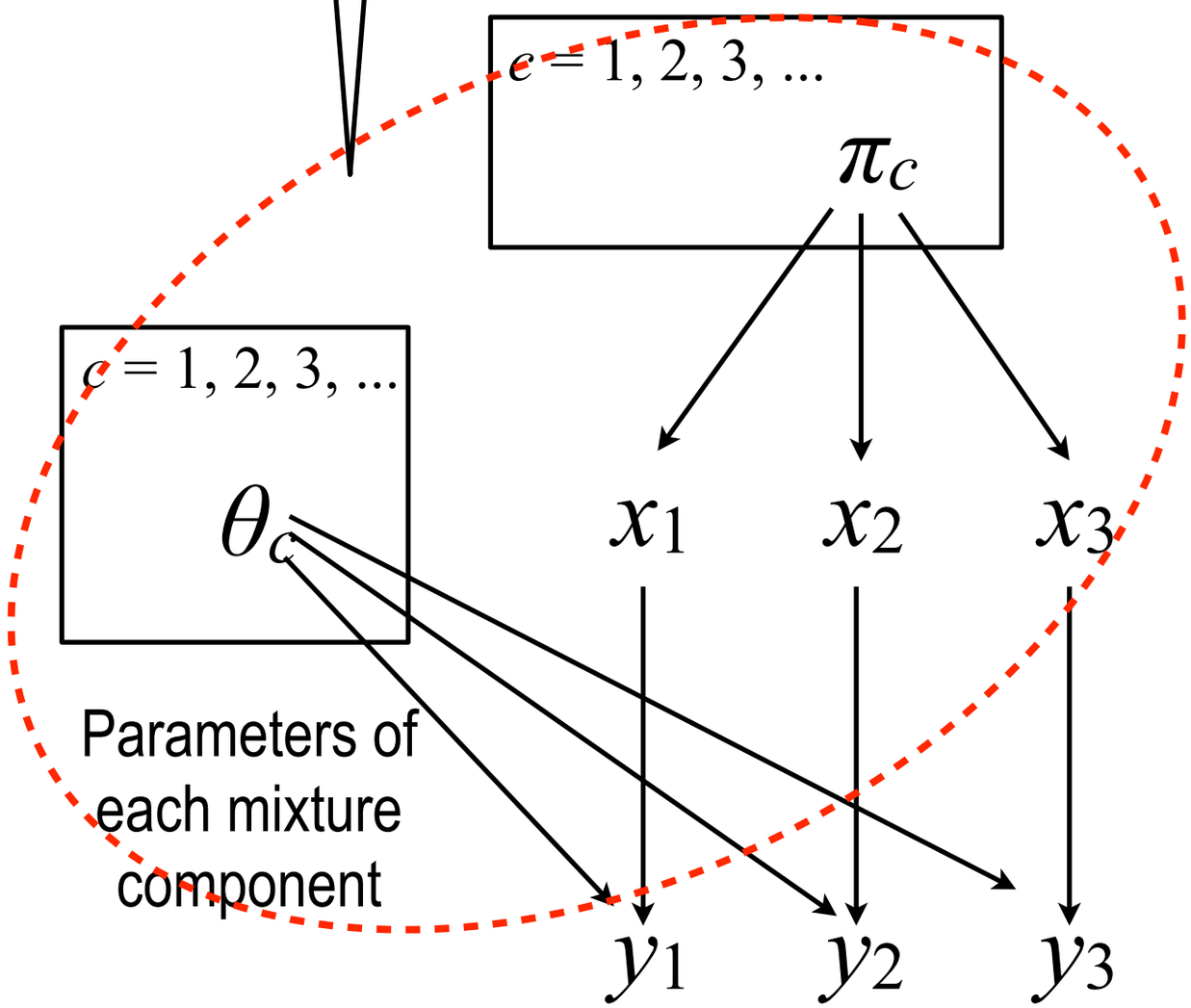
Let's say the task here is to partition into two clusters (DP still useful in this situation!)

Clustering

The part in red is the part called a DP; the whole model is called a DP mixture model

s' or groups





To make this formal, let's assume the data was generated from a *DP mixture model*. The task is to partition the data points into clusters.

Let's say the task here is to partition into two clusters (DP still useful in this situation!)

Clustering

Loss function: Rand loss $\mathrm{Rand}(x, x')$ between the true partition x' and a putative partition x'

Rand loss: The number of pairs of data points i, j such that:

$$\mathbf{1}[x_i = x_j] \neq \mathbf{1}[x'_i = x'_j]$$

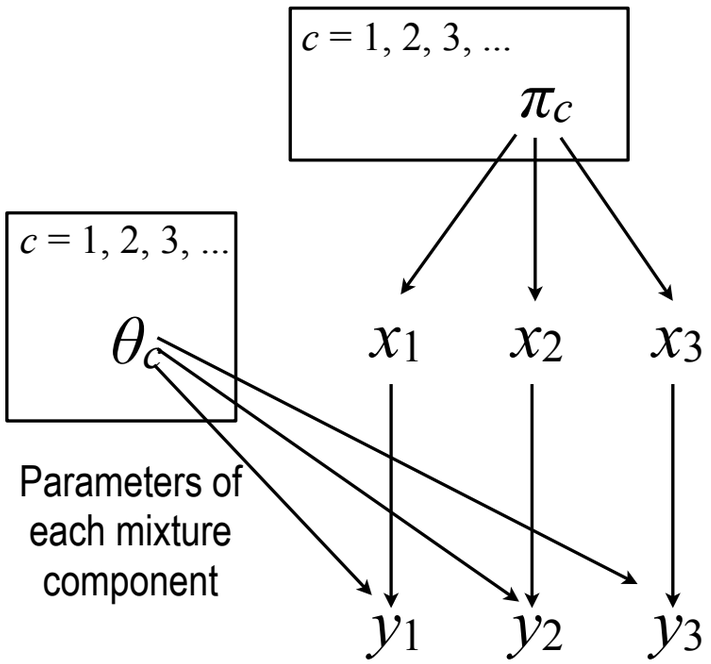
In other words, you incur a loss of one each time you either (1) put together two points that did not belong together, or (2) did not put together two points that are in the same cluster in the true partition

$$\sum_{i<j} \mathbf{1}\Big[\mathbf{1}[x_i = x_j] \neq \mathbf{1}[x'_i = x'_j]\Big]$$

Let's call this $\rho_{i,j}$

We have all the ingredients for the Bayes estimator

Joint probability over knowns and unknowns



Loss function

$$\mathrm{Rand}(x, \rho) = \sum_{i<j} \mathbf{1}\Big[\mathbf{1}[x_i = x_j] \neq \rho_{i,j}\Big]$$

Bayes estimator: $\mathrm{argmin}_\rho \left\{ \mathbb{E}\left[\mathrm{Rand}(x, \rho) \middle| y\right] : \rho \text{ is a partition} \right\}$

(Simplifications of the Bayes estimator on the board)

Computing the Bayes estimator

Two steps:

1. Computing the posterior that pairs of data points come from the same cluster: MCMC
2. Plug-in these numbers into the minimization problem and solve it (in this case this can be done efficiently using a min-flow algorithm*)

*Optional: If you are interested, you can read about the second step in Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2009). Introduction to Algorithms (3rd ed.). MIT Press.

Computing the posterior distribution of Dirichlet Process mixture models

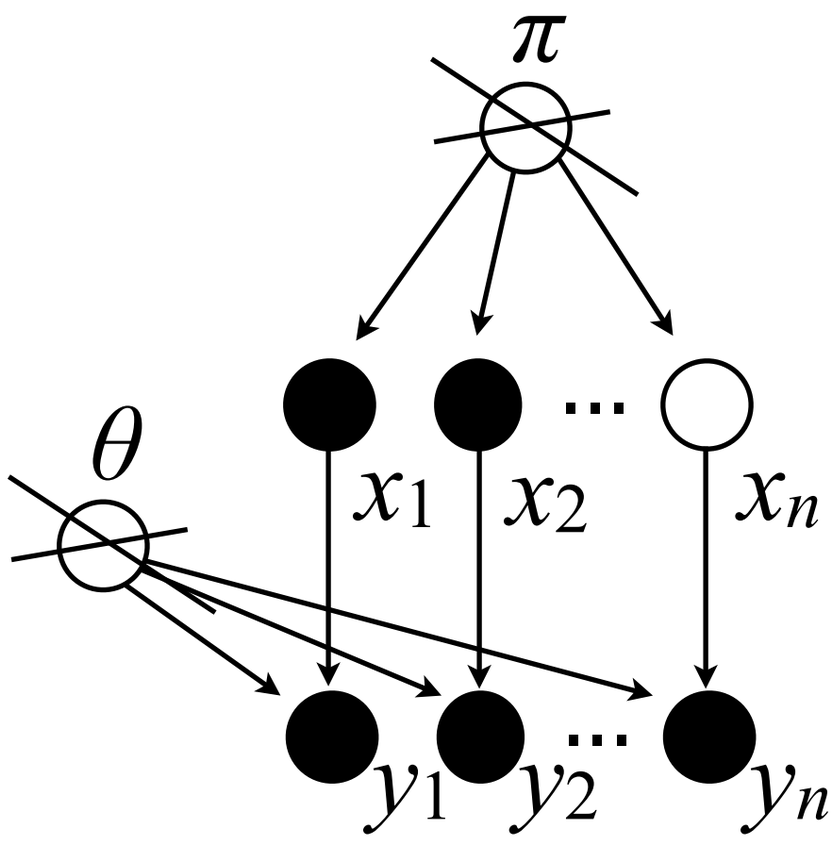
Overview

Goal: computing a conditional expectation (e.g. for a Bayes estimator)

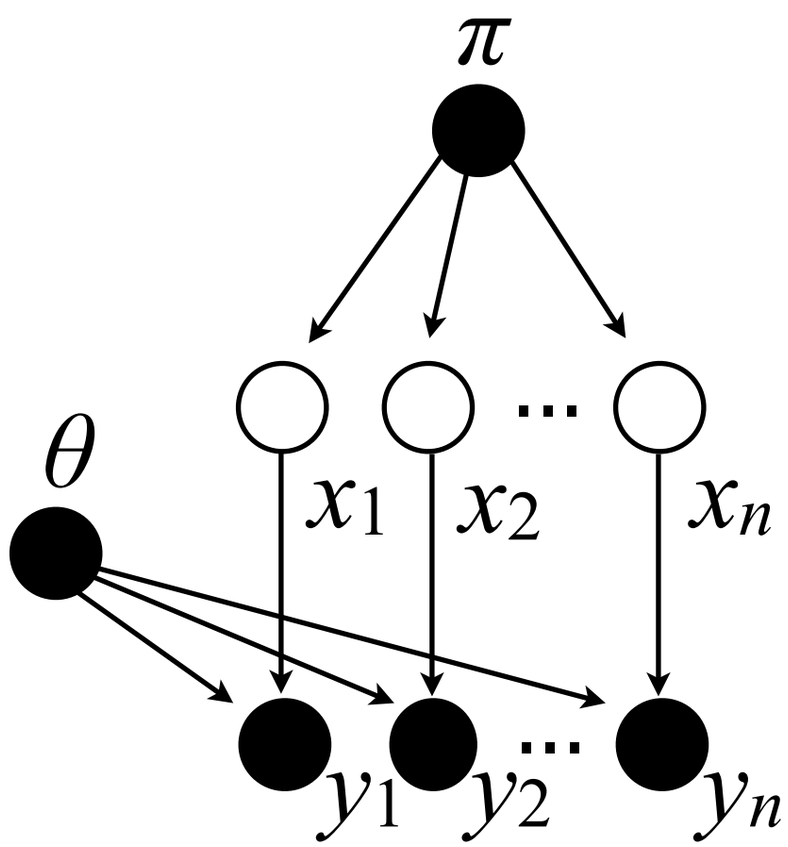
$$\mathbb{E}[f(\pi, \theta, x, y)|y]$$

We will cover two samplers:

Collapsed sampler



Slice sampler



Overview

Goal: computing a conditional expectation (e.g. for a Bayes estimator)

$$\mathbb{E}[f(\pi, \theta, x, y)|y]$$

We will cover two samplers:

	Collapsed sampler	**Slice sampler**
Pros	+ Easy to implement + Rao-Blackwellized	+ Flexible conditions on loss + Easy to parallelize
Cons	- Restrictions on the loss - Restr. on the likelihood - No easy parallelization	- Harder to implement - Aux. variables: less efficient - Memory needs

Overview

	Collapsed sampler	Slice sampler
Pros	+ Easy to implement + Rao-Blackwellized	+ Flexible conditions on loss + Easy to parallelize
Cons	- **Restrictions on the loss** - **Restr. on the likelihood** - No easy parallelization	- Harder to implement - Aux. variables: less efficient - Memory needs

Collapsed Gibbs: restrictions on the loss

Goal: computing a conditional expectation (e.g. for a Bayes estimator)

$$\mathbb{E}[f(\pi, \theta, x, y)|y]$$

Special case: sometimes, f depends only on the cluster indicators,

$$f(\pi, \theta, x, y) = f(x)$$

Example: clustering, where we only care about the posterior fraction of the time each pair of points is in the same mixture component

Note: can be made a bit less restrictive
(will come back to this point later)

Collapsed Gibbs: conjugacy restriction

Let $L(\mathrm{d}y|\theta)$ denote the likelihood, and for a subset of the data points $B \subseteq \{1, 2, ..., n\}$, let $L(\mathrm{d}y_B)$ denote the cluster marginal likelihood

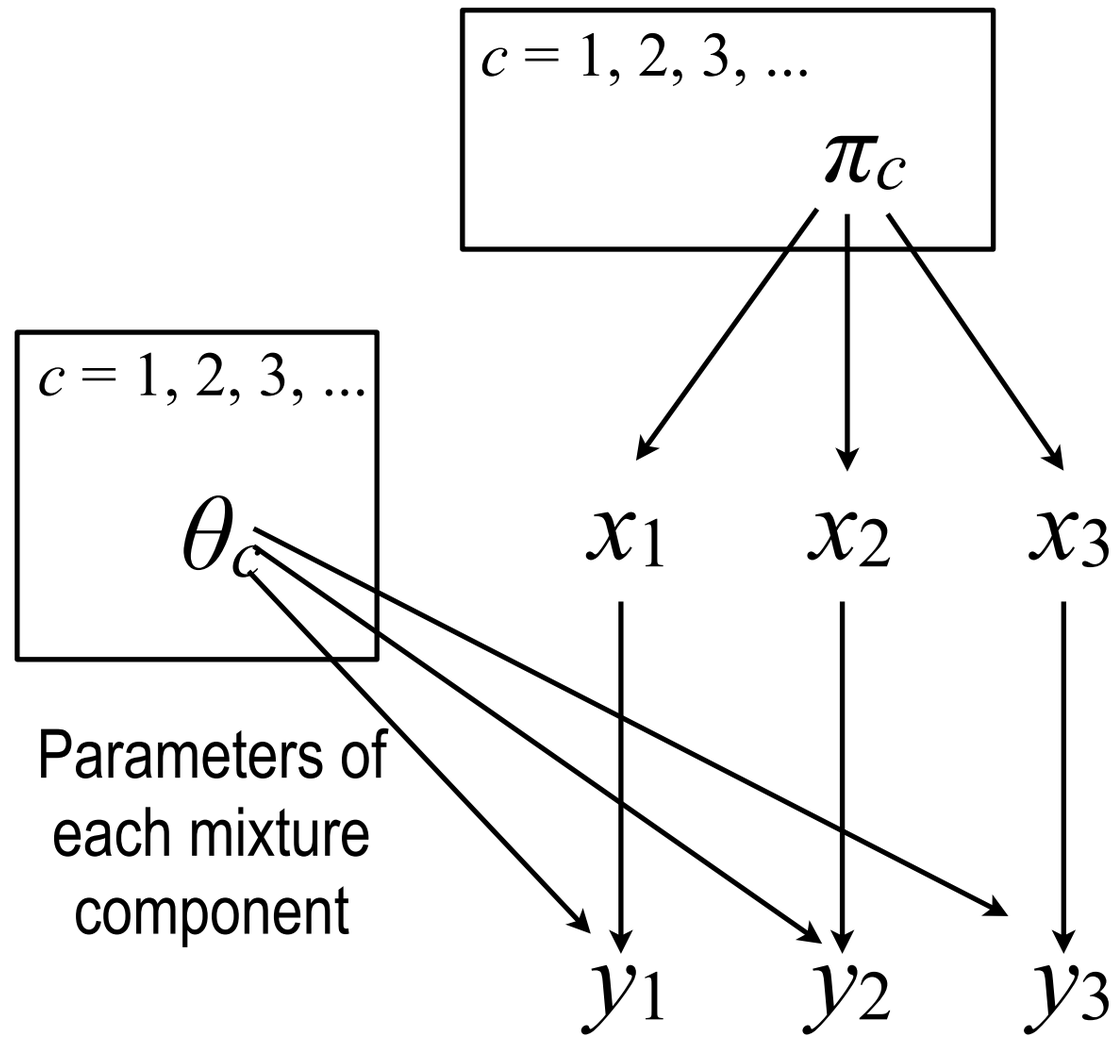
$$L(\mathrm{d}y_B) = \int \prod_{i \in B} L(\mathrm{d}y_i|\theta) G_0(\mathrm{d}\theta)$$

Assumption: This integral can be computed analytically (i.e. the likelihood L is conjugate with G_0, with a tractable normalization $A(\theta)$)

Collapsed Gibbs sampler: representation

Consequence of the two assumptions: we can use a MCMC sampler on a finite state space: the space of partitions of the data points

Joint distribution after marginalization of the *sticks* π and *dishes* θ:



(Derivation on the board)

Collapsed Gibbs sampler: proposals

Basic idea: propose local changes to the current partition of the data points

Standard way: take one of the customer out of the restaurant, compute the unnormalized density p_i for each possible table assignment, normalize the p_i's and sample from a multinomial with these parameters

Practical notes on the standard collapsed sampler

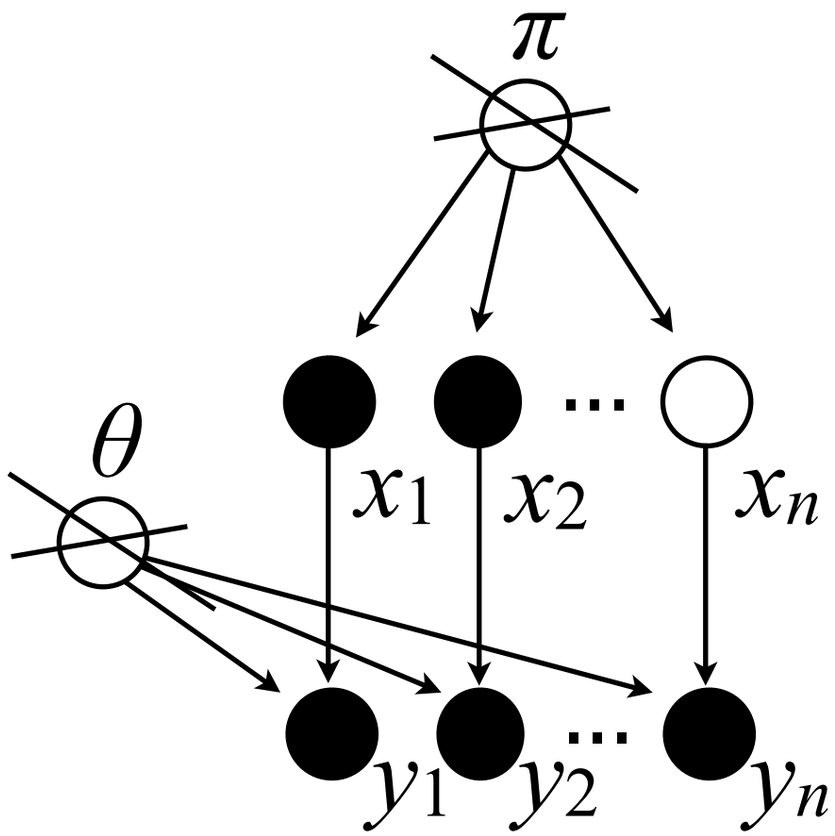
In theory: it is irreducible (can break all the clusters and reform some new ones with positive probability)

In practice: breaking clusters have small probabilities, so it's important to initialize the chain with every datapoint in its own cluster (so first sweep takes $\mathrm{O}(n^2)$)

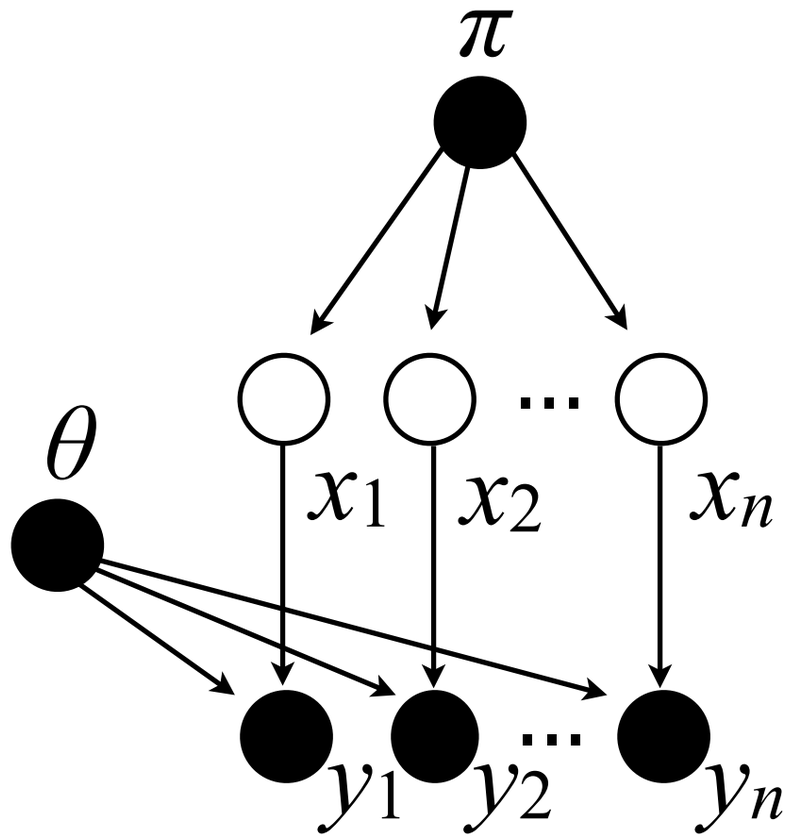
Large-move techniques: can remove the need to use such initialization (potential project here---come talk to me at office hours for more info)

Next: slice sampler

Collapsed sampler



Slice sampler



Slice sampling for DP inference

Idea: use lazy computation

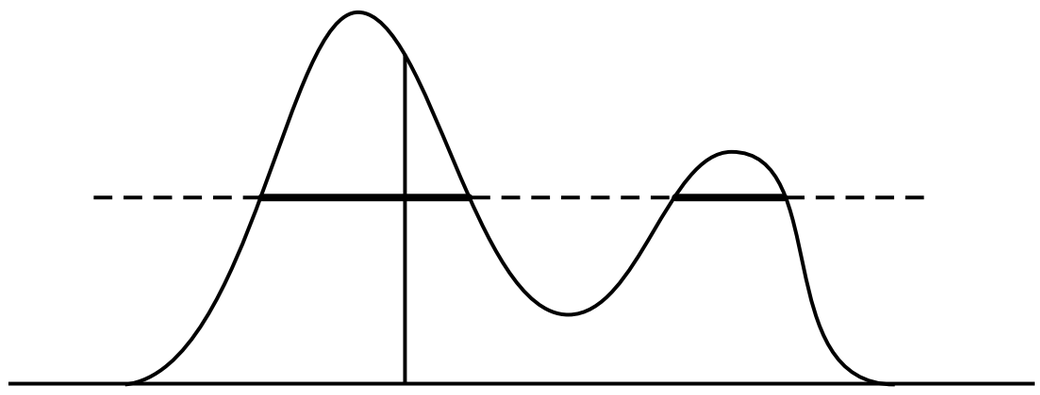
Problem: for posterior inference, even the small sticks have an impact on the posterior

Solution: using an auxiliary variable similar to the one used for slice sampling in univariate distributions

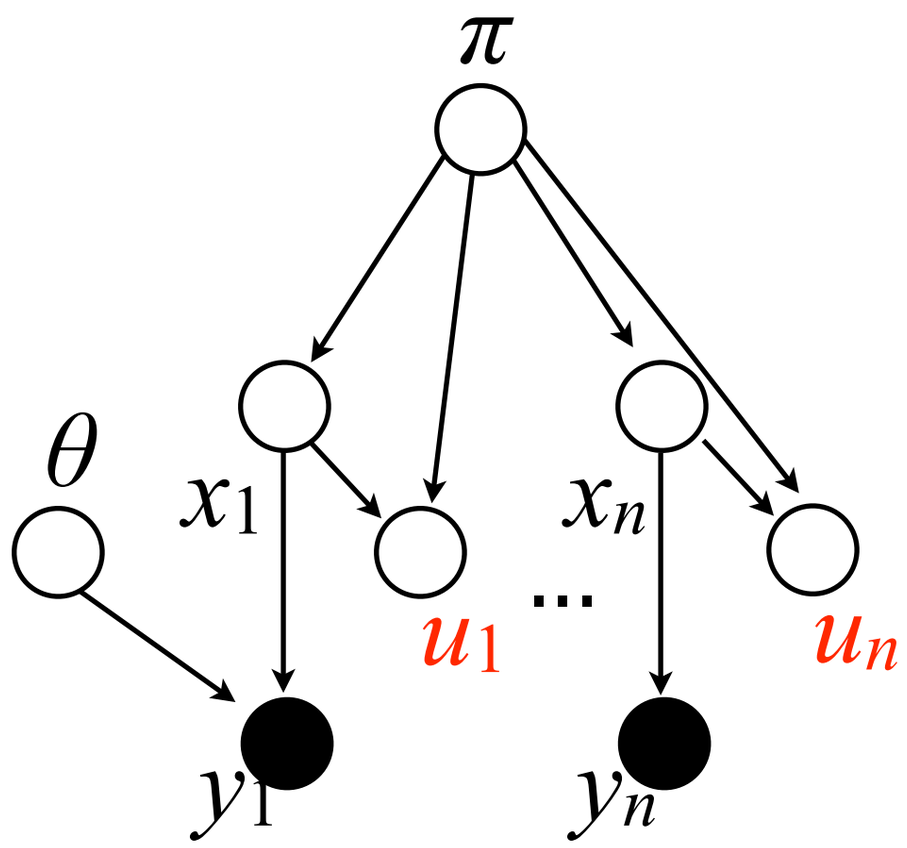
$$X \qquad X \sim f(x)/Z$$

$$\downarrow$$

$$U \qquad U \mid X \sim \mathrm{Uni}[0, f(X)]$$



Auxiliary variable

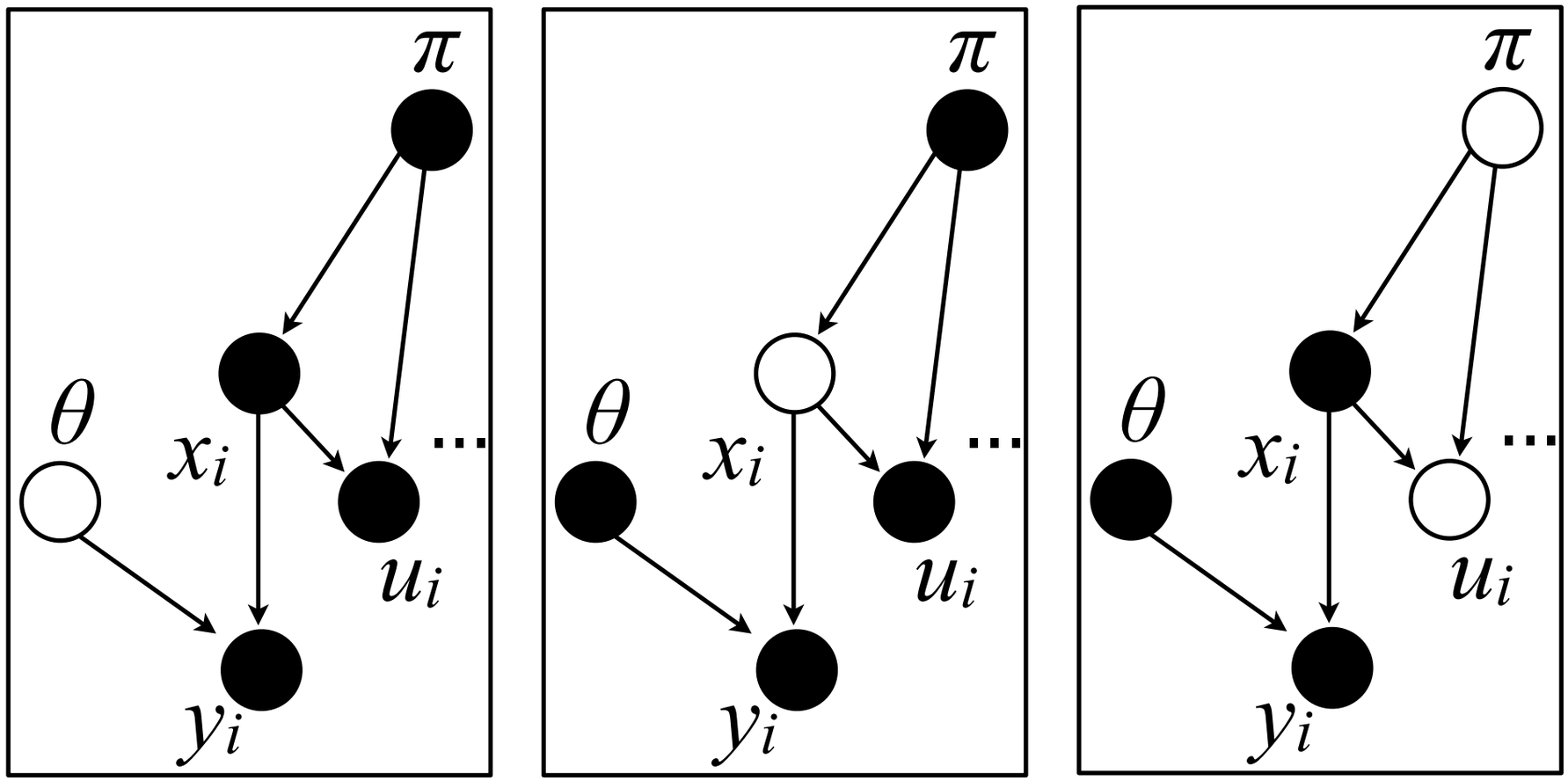


$$u_i | x_i, \pi \sim \mathrm{Uni}[0, w_{x_i}]$$

As before, adding a node downstream in a directed graphical model does not change the marginal of the original nodes

(Derivation of the joint density on the board)

Sampling

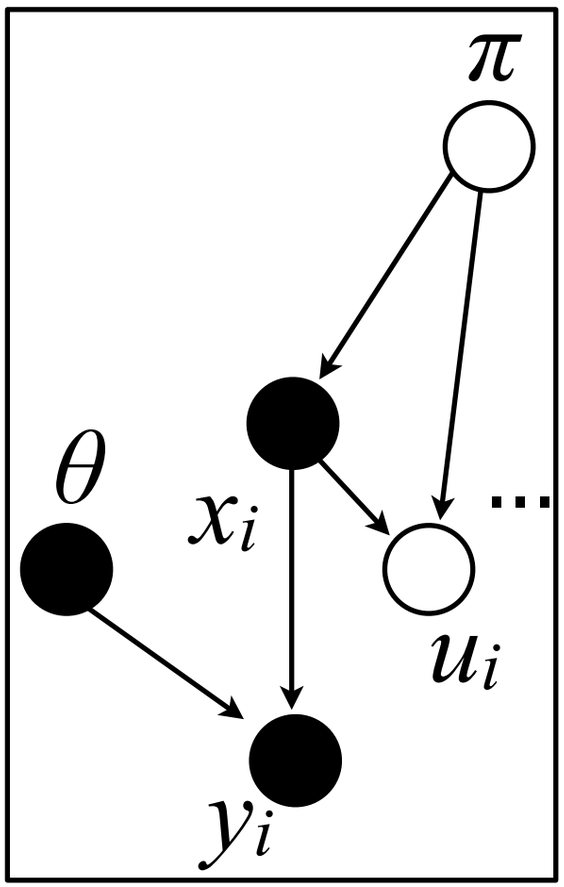


Dishes Assignments Aux. var. & sticks

(Derivation of the sampling steps on the board)

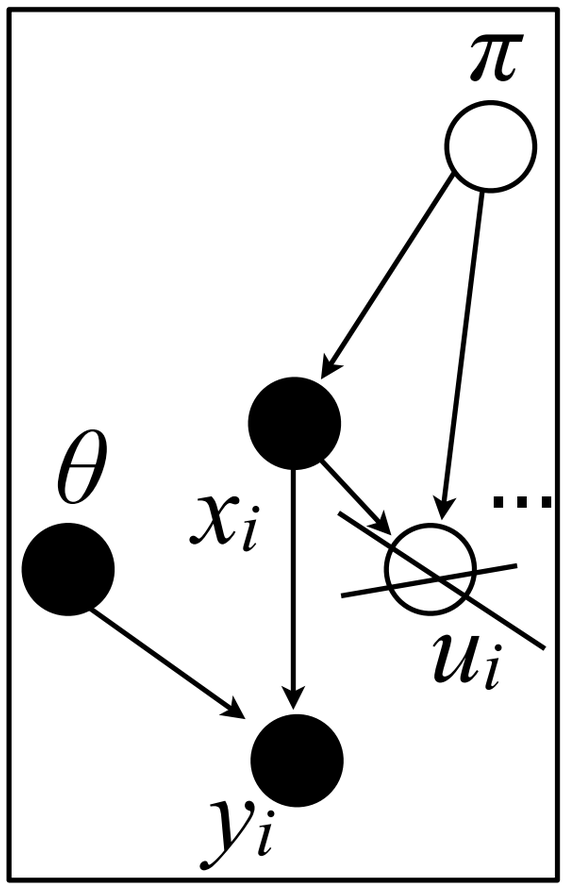
Sampling auxiliary variables and sticks

By the chain rule: (derivation of each distribution on the board)

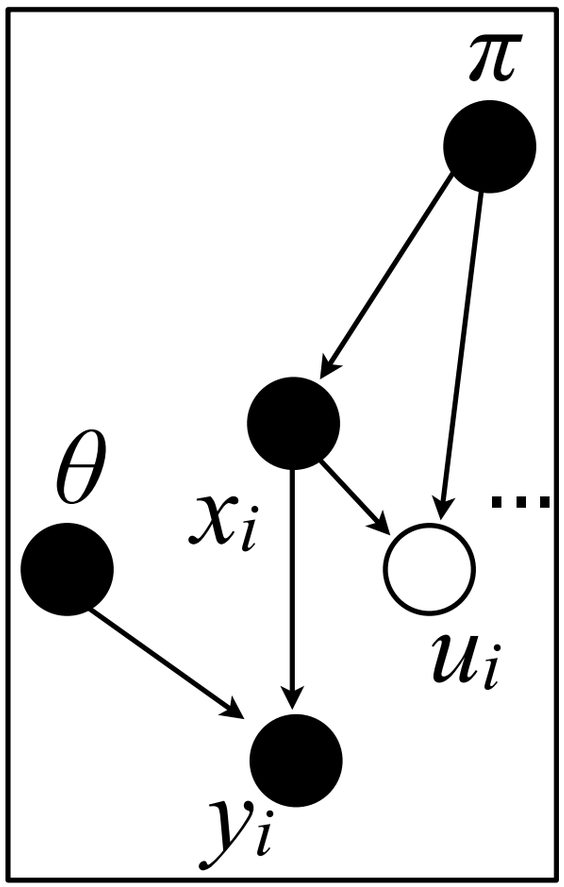


Aux. var. & sticks

~



Sticks



Aux. var. given sticks

Sequential alternative to dart throwing

$$\beta_j \overset{\text{iid}}{\sim} \text{Beta}(1, \alpha_0)$$