A Permutation-Augmented Sampler for DP Mixture Models

ICML 2007 Corvallis, Oregon

June 21, 2007

Percy Liang
UC Berkeley

Michael I. Jordan
UC Berkeley

Ben Taskar
U Penn

Introduction

Dirichlet process mixture models:

- Clustering applications:
  - natural language processing, e.g. [Blei, et. al, 2004; Daume, Marcu, 2005; Goldwater, et. al, 2006; Liang, et. al, 2007]
  - vision, e.g. [Sudderth, et. al, 2006]
  - bioinformatics, e.g. [Xing, et. al, 2004]
- Nonparametric: number of clusters adapts to data
- Current inference based on local moves

Introduction

Dirichlet process mixture models:

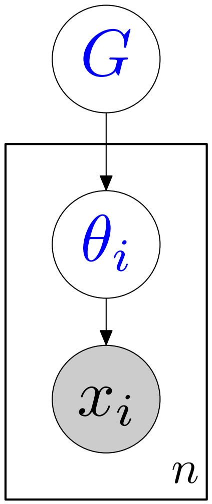
- Clustering applications:
  - natural language processing, e.g. [Blei, et. al, 2004; Daume, Marcu, 2005; Goldwater, et. al, 2006; Liang, et. al, 2007]
  - vision, e.g. [Sudderth, et. al, 2006]
  - bioinformatics, e.g. [Xing, et. al, 2004]
- Nonparametric: number of clusters adapts to data
- Current inference based on local moves

Outline:

- DP mixture model
- Permutation-augmented model $\Rightarrow$ global moves
- Experiments

[Ferguson, 1973; Antoniak, 1974]

Dirichlet processes



DP mixture model

$G \sim \text{DP}(\alpha_0, G_0)$

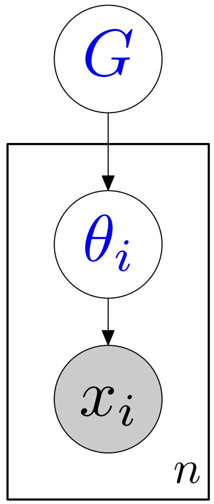
For each data point $i = 1, \ldots, n$:

  $\theta_i \sim G$

  $x_i \sim F(\theta_i)$

[Ferguson, 1973; Antoniak, 1974]

Dirichlet processes



DP mixture model

$G \sim \text{DP}(\alpha_0, G_0)$

For each data point $i = 1, \ldots, n$:

$\theta_i \sim G$

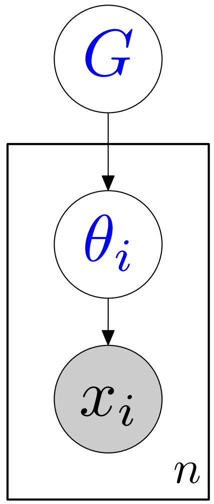
$x_i \sim F(\theta_i)$

Definition: G_0 = a distribution on Θ, α_0 = concentration parameter.

G is a draw from a Dirichlet process, denoted $G \sim \text{DP}(\alpha_0, G_0)$

[Ferguson, 1973; Antoniak, 1974]

Dirichlet processes



DP mixture model

$G \sim \text{DP}(\alpha_0, G_0)$

For each data point $i = 1, \ldots, n$:

  $\theta_i \sim G$

  $x_i \sim F(\theta_i)$

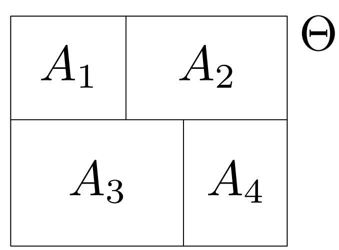
Definition: G_0 = a distribution on Θ, α_0 = concentration parameter.

G is a draw from a Dirichlet process, denoted $G \sim \text{DP}(\alpha_0, G_0)$

$$\Updownarrow$$

$$(G(A_1), \ldots, G(A_K)) \sim \text{Dirichlet}(\alpha_0 G_0(A_1), \ldots, \alpha_0 G_0(A_K))$$

for all partitions $(A_1, \ldots, A_K)$ of Θ.



Inference

Representations:

- Chinese restaurant process: marginalize G
- Stick-breaking representation: explicitly represent G

Inference

Representations:

- Chinese restaurant process: marginalize G
- Stick-breaking representation: explicitly represent G

Previous algorithms:

- Collapsed Gibbs sampling [Escobar, West, 1995]
- Blocked Gibbs sampling [Ishwaran, James, 2001]
- Split-merge sampling [Jain, Neal, 2000; Dahl, 2003]
- Variational [Blei, Jordan, 2005; Kurihara, et. al, 2007]
- A-star search [Daume, 2007]

Chinese restaurant process

$G \sim \text{DP}(\alpha_0, G_0)$ is discrete (with probability 1)

Marginalize out $G \Rightarrow$ induces clustering $\mathbf{C}$

Each cluster $c \in \mathbf{C}$ is a subset of $\{1, \dots, n\}$

Example: $\mathbf{C} = \{\{1\}, \{2, 3, 5\}, \{4\}\}$

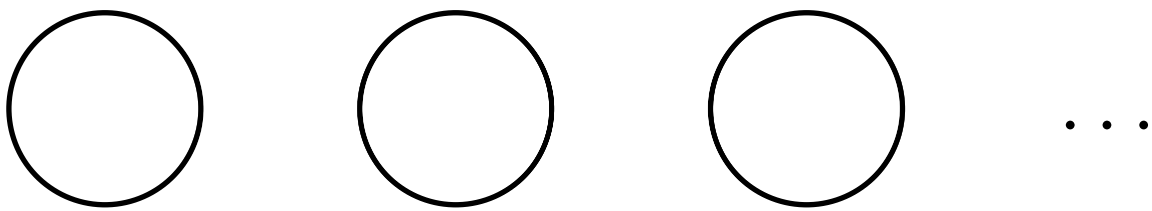
Chinese restaurant process

$G \sim \text{DP}(\alpha_0, G_0)$ is discrete (with probability 1)

Marginalize out $G \Rightarrow$ induces clustering $\mathbf{C}$

Each cluster $c \in \mathbf{C}$ is a subset of $\{1, \dots, n\}$

Example: $\mathbf{C} = \{\{1\}, \{2, 3, 5\}, \{4\}\}$



$$p(i \in c) = \begin{cases} \frac{|c|}{i-1+\alpha_0} & \text{if } c \text{ old} \\ \frac{\alpha_0}{i-1+\alpha_0} & \text{if } c \text{ new} \end{cases}$$

probability:

[Pitman, 2002]

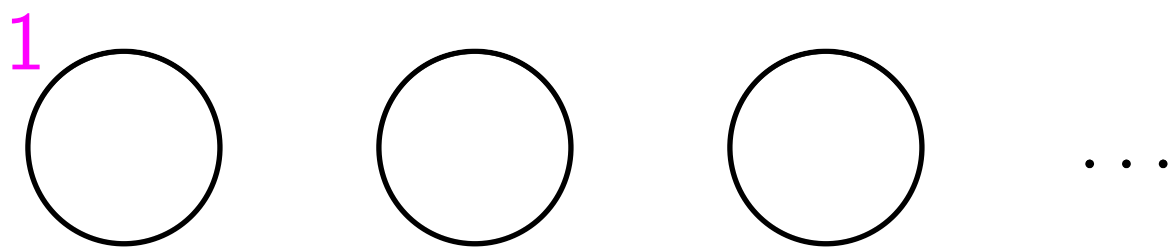
Chinese restaurant process

$G \sim \text{DP}(\alpha_0, G_0)$ is discrete (with probability 1)

Marginalize out $G \Rightarrow$ induces clustering $\mathbf{C}$

Each cluster $c \in \mathbf{C}$ is a subset of $\{1, \ldots, n\}$

Example: $\mathbf{C} = \{\{1\}, \{2, 3, 5\}, \{4\}\}$



$$p(i \in c) = \begin{cases} \frac{|c|}{i-1+\alpha_0} & \text{if } c \text{ old} \\ \frac{\alpha_0}{i-1+\alpha_0} & \text{if } c \text{ new} \end{cases}$$

probability: $\frac{\alpha_0}{0+\alpha_0}$

[Pitman, 2002]

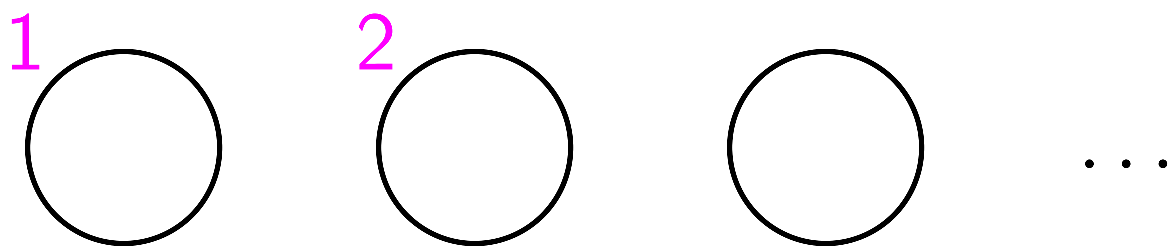
Chinese restaurant process

$G \sim \mathrm{DP}(\alpha_0, G_0)$ is discrete (with probability 1)

Marginalize out $G \Rightarrow$ induces clustering $\mathbf{C}$

Each cluster $c \in \mathbf{C}$ is a subset of $\{1, \dots, n\}$

Example: $\mathbf{C} = \{\{1\}, \{2, 3, 5\}, \{4\}\}$



$$p(i \in c) = \begin{cases} \frac{|c|}{i-1+\alpha_0} & \text{if } c \text{ old} \\ \frac{\alpha_0}{i-1+\alpha_0} & \text{if } c \text{ new} \end{cases}$$

probability: $\frac{\alpha_0}{0+\alpha_0} \frac{\alpha_0}{1+\alpha_0}$

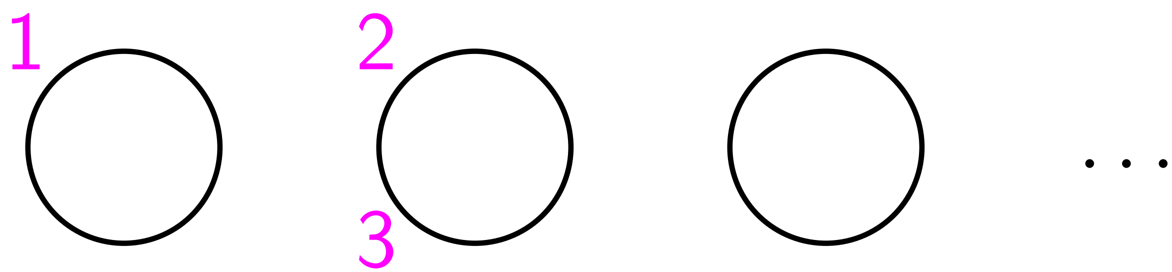
Chinese restaurant process

$G \sim \text{DP}(\alpha_0, G_0)$ is discrete (with probability 1)

Marginalize out $G \Rightarrow$ induces clustering $\mathbf{C}$

Each cluster $c \in \mathbf{C}$ is a subset of $\{1, \dots, n\}$

Example: $\mathbf{C} = \{\{1\}, \{2, 3, 5\}, \{4\}\}$



$$p(i \in c) = \begin{cases} \frac{|c|}{i-1+\alpha_0} & \text{if } c \text{ old} \\ \frac{\alpha_0}{i-1+\alpha_0} & \text{if } c \text{ new} \end{cases}$$

probability: $\frac{\alpha_0}{0+\alpha_0} \frac{\alpha_0}{1+\alpha_0} \frac{1}{2+\alpha_0}$

[Pitman, 2002]

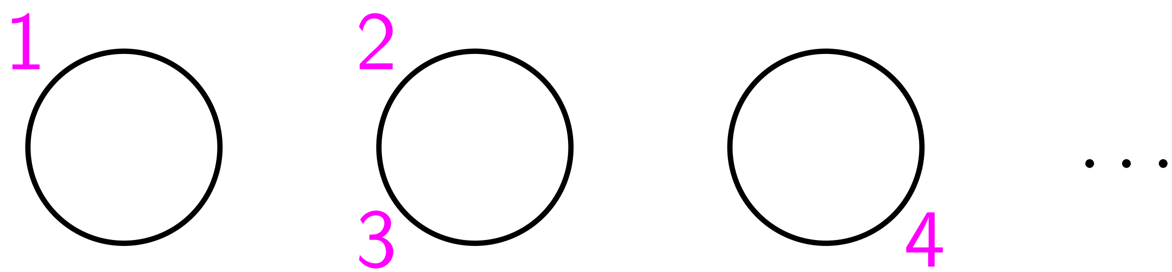
Chinese restaurant process

$G \sim \text{DP}(\alpha_0, G_0)$ is discrete (with probability 1)

Marginalize out $G \Rightarrow$ induces clustering $\mathbf{C}$

Each cluster $c \in \mathbf{C}$ is a subset of $\{1, \dots, n\}$

Example: $\mathbf{C} = \{\{1\}, \{2, 3, 5\}, \{4\}\}$



$$p(i \in c) = \begin{cases} \frac{|c|}{i-1+\alpha_0} & \text{if } c \text{ old} \\ \frac{\alpha_0}{i-1+\alpha_0} & \text{if } c \text{ new} \end{cases}$$

probability: $\frac{\alpha_0}{0+\alpha_0} \frac{\alpha_0}{1+\alpha_0} \frac{1}{2+\alpha_0} \frac{\alpha_0}{3+\alpha_0}$

[Pitman, 2002]

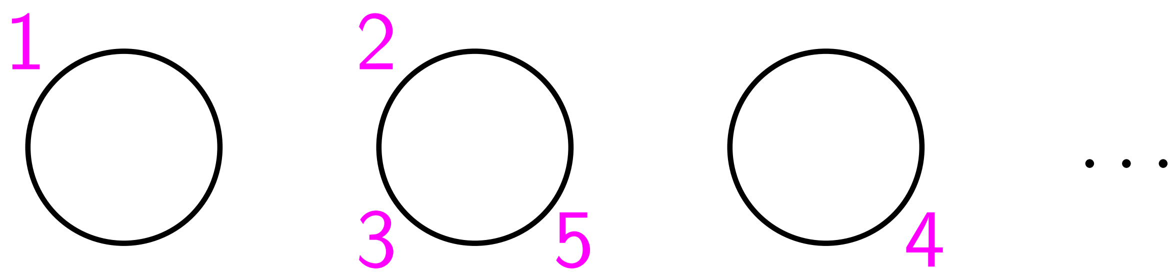
Chinese restaurant process

$G \sim \mathrm{DP}(\alpha_0, G_0)$ is discrete (with probability 1)

Marginalize out $G \Rightarrow$ induces clustering $\mathbf{C}$

Each cluster $c \in \mathbf{C}$ is a subset of $\{1, \dots, n\}$

Example: $\mathbf{C} = \{\{1\}, \{2, 3, 5\}, \{4\}\}$



$$p(i \in c) = \begin{cases} \frac{|c|}{i-1+\alpha_0} & \text{if } c \text{ old} \\ \frac{\alpha_0}{i-1+\alpha_0} & \text{if } c \text{ new} \end{cases}$$

probability: $\frac{\alpha_0}{0+\alpha_0} \frac{\alpha_0}{1+\alpha_0} \frac{1}{2+\alpha_0} \frac{\alpha_0}{3+\alpha_0} \frac{2}{4+\alpha_0}$

[Antoniak, 1974]

CRP prior over clusterings

Previous example: $p(\mathbf{C}) = \frac{\alpha_0}{0+\alpha_0} \frac{\alpha_0}{1+\alpha_0} \frac{1}{2+\alpha_0} \frac{\alpha_0}{3+\alpha_0} \frac{2}{4+\alpha_0}$

[Antoniak, 1974]

CRP prior over clusterings

Previous example: $p(\mathbf{C}) = \frac{\alpha_0}{0+\alpha_0} \frac{\alpha_0}{1+\alpha_0} \frac{1}{2+\alpha_0} \frac{\alpha_0}{3+\alpha_0} \frac{2}{4+\alpha_0}$

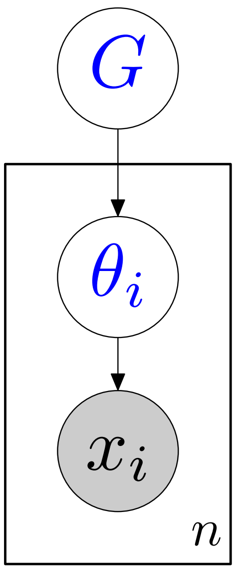
In general:

$$p(\mathbf{C}) = \frac{1}{\mathcal{AF}(\alpha_0, n)} \prod_{c \in \mathbf{C}} \alpha_0(|c| - 1)!$$

$\mathcal{AF}(\alpha_0, n) = \alpha_0(\alpha_0 + 1) \cdots (\alpha_0 + n - 1)$ is ascending factorial

Key: $p(\mathbf{C})$ decomposes over clusters c

DP mixture model via the CRP

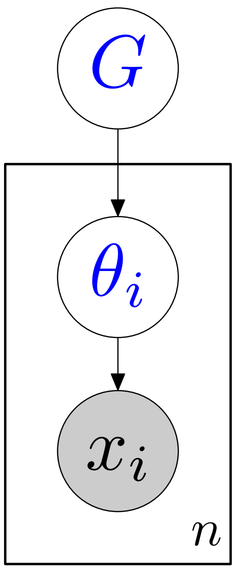


DP mixture model via the CRP

Each cluster (table) c has a dish θ.

Data points (customers) generated i.i.d. given dish.

Assuming conjugacy, we can marginalize out θ.

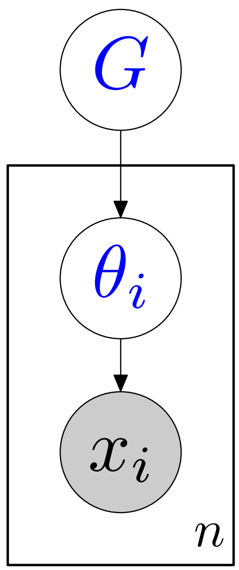


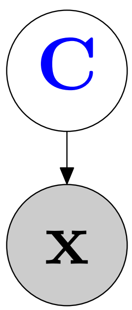
DP mixture model via the CRP

Each cluster (table) c has a dish θ.

Data points (customers) generated i.i.d. given dish.

Assuming conjugacy, we can marginalize out θ.



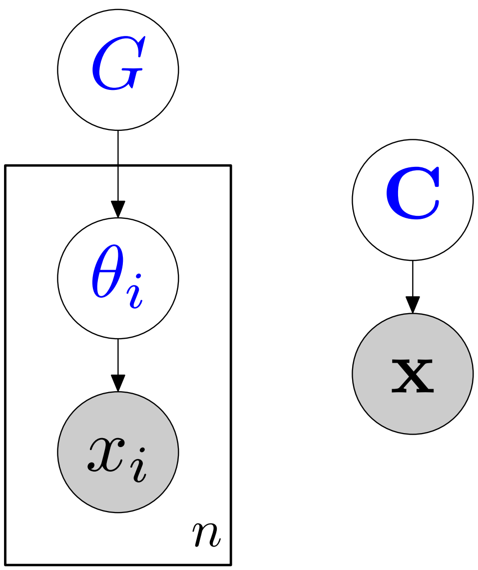


DP mixture model via the CRP

Each cluster (table) c has a dish θ.

Data points (customers) generated i.i.d. given dish.

Assuming conjugacy, we can marginalize out θ.

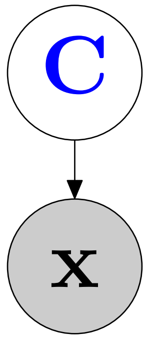


$$p(\mathbf{C}) = \frac{1}{\mathcal{AF}(\alpha_0, n)} \prod_{c \in \mathbf{C}} \alpha_0 (|c| - 1)!$$

$$p(\mathbf{x} \mid \mathbf{C}) = \prod_{c \in \mathbf{C}} \underbrace{\int \prod_{i \in c} F(x_i; \theta) G_0(d\theta)}_{\stackrel{\text{def}}{=} p(\mathbf{x}_c)}$$

Key: $p(\mathbf{C})$ and $p(\mathbf{x} \mid \mathbf{C})$ decompose over clusters c

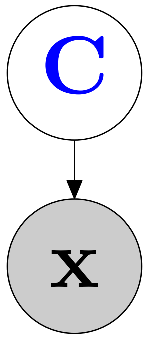
Posterior inference



Goal: compute $p(\mathbf{C} \mid \mathbf{x})$

- Exact inference: sum over exponential number of clusterings

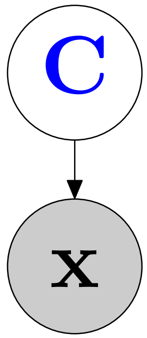
Posterior inference



Goal: compute $p(\mathbf{C} \mid \mathbf{x})$

- Exact inference: sum over exponential number of clusterings
- Collapsed Gibbs sampler: change $\mathbf{C}$ one assignment at a time

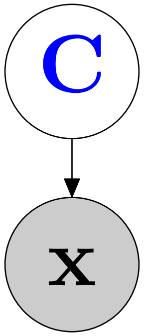
Posterior inference



Goal: compute $p(\mathbf{C} \mid \mathbf{x})$

- Exact inference: sum over exponential number of clusterings
- Collapsed Gibbs sampler: change $\mathbf{C}$ one assignment at a time
- Split-merge sampler: change $\mathbf{C}$ two clusters at a time

Posterior inference

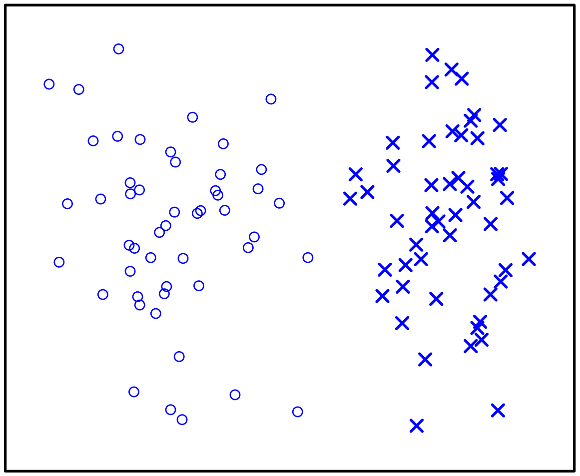


Goal: compute $p(\mathbf{C} \mid \mathbf{x})$

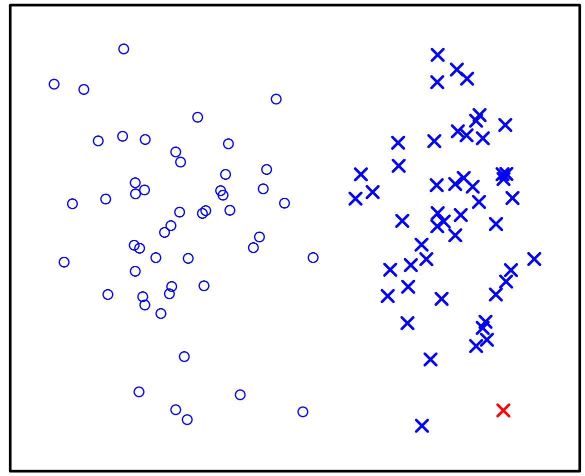
- Exact inference: sum over exponential number of clusterings
- Collapsed Gibbs sampler: change $\mathbf{C}$ one assignment at a time
- Split-merge sampler: change $\mathbf{C}$ two clusters at a time
- Permutation-augmented sampler: can change all of $\mathbf{C}$

Local optima

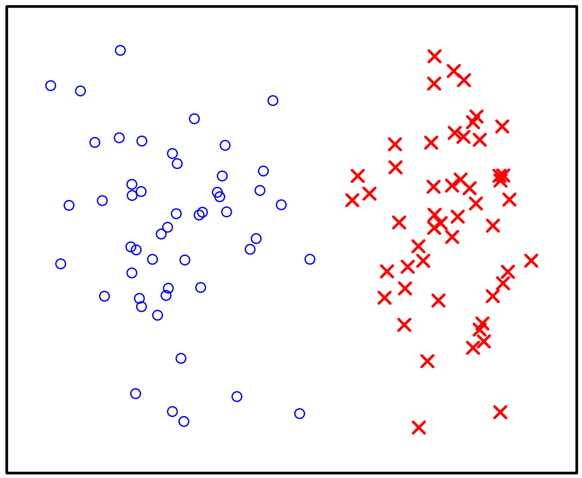
Collapsed Gibbs can get stuck in local optima



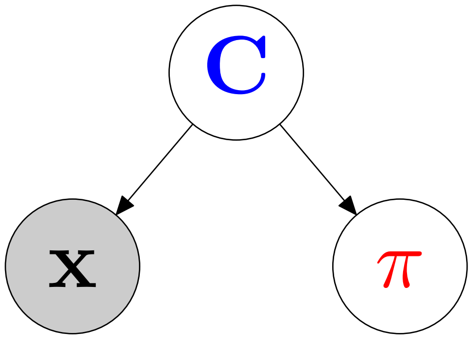
one collapsed Gibbs move →



Hard to reach this state:

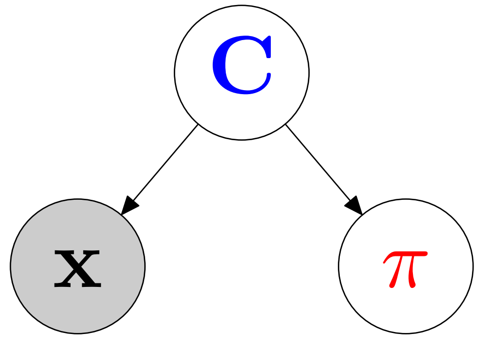


Augmenting with a permutation



Sampler: alternate between sampling $\mathbf{C}$ and π

Augmenting with a permutation

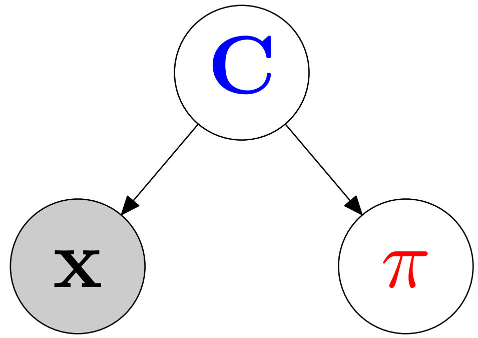


Sampler: alternate between sampling $\mathbf{C}$ and π

Why augment?

- Conditioned on π, can use dynamic programming to efficiently sample all of $\mathbf{C}$

Augmenting with a permutation

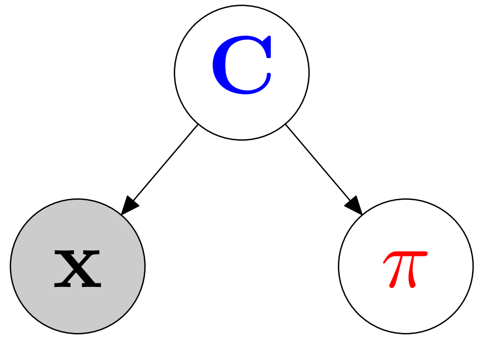


Sampler: alternate between sampling $\mathbf{C}$ and π

Why augment?

- Conditioned on π, can use dynamic programming to efficiently sample all of $\mathbf{C}$
- If sample in augmented model, can marginalize out (ignore) π to recover original model

Augmenting with a permutation



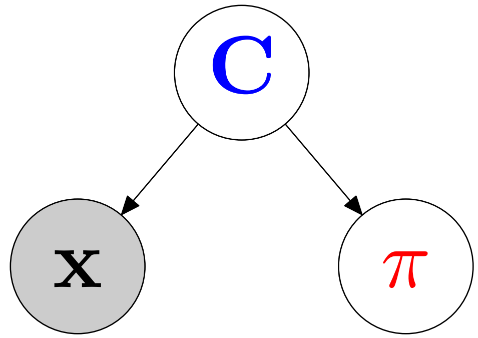
Sampler: alternate between sampling $\mathbf{C}$ and π

Why augment?

$\{\{1\}, \{2, 3, 5\}, \{4\}\}$

- Conditioned on π, can use dynamic programming to efficiently sample all of $\mathbf{C}$
- If sample in augmented model, can marginalize out (ignore) π to recover original model

Augmenting with a permutation



Sampler: alternate between sampling $\mathbf{C}$ and π

Why augment?

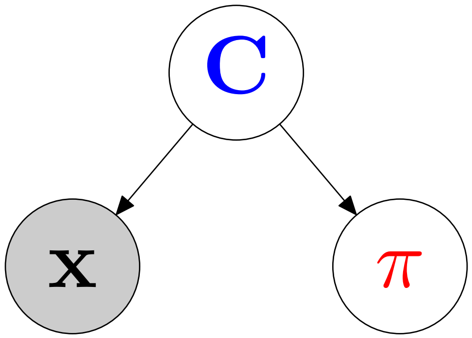
- Conditioned on π, can use dynamic programming to efficiently sample all of $\mathbf{C}$
- If sample in augmented model, can marginalize out (ignore) π to recover original model

$\{\{1\}, \{2, 3, 5\}, \{4\}\}$

sample $\pi \mid \mathbf{C}$

4 1 5 2 3

Augmenting with a permutation



Sampler: alternate between sampling $\mathbf{C}$ and π

Why augment?

- Conditioned on π, can use dynamic programming to efficiently sample all of $\mathbf{C}$
- If sample in augmented model, can marginalize out (ignore) π to recover original model

$\{\{1\},\{2,3,5\},\{4\}\}$

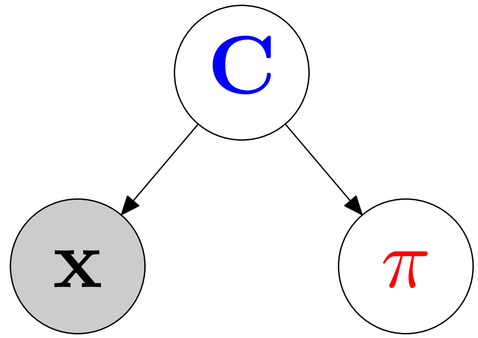
sample $\pi \mid \mathbf{C}$

4 1 5 2 3

sample $\mathbf{C} \mid \pi, \mathbf{x}$

$\{\{4,1\},\{5\},\{2,3\}\}$

Augmenting with a permutation



Sampler: alternate between sampling $\mathbf{C}$ and π

Why augment?

- Conditioned on π, can use dynamic programming to efficiently sample all of $\mathbf{C}$
- If sample in augmented model, can marginalize out (ignore) π to recover original model

$\{\{1\},\{2,3,5\},\{4\}\}$

sample $\pi \mid \mathbf{C}$

4 1 5 2 3

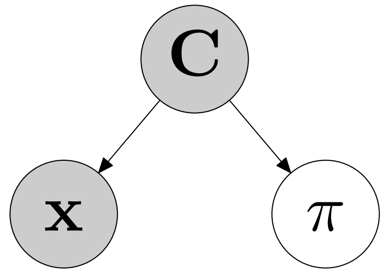
sample $\mathbf{C} \mid \pi, \mathbf{x}$

$\{\{4,1\},\{5\},\{2,3\}\}$

sample $\pi \mid \mathbf{C}$

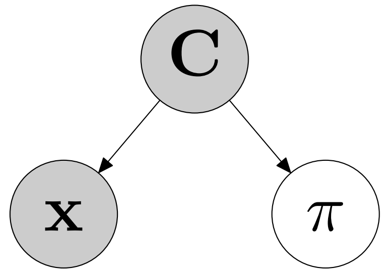
5 4 1 3 2

Sampling the permutation



$$p(\pi \mid \mathbf{C}, \mathbf{x})$$

Sampling the permutation



$p(\pi \mid \mathbf{C}, \mathbf{x})$

What's $p(\pi \mid \mathbf{C})$?

Let $\Pi(\mathbf{C})$ = permutations consistent with $\mathbf{C}$ (all clusters contiguous in permutation)

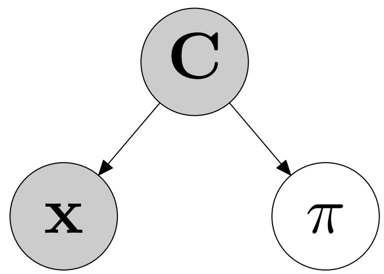
Example:

Clustering $\mathbf{C} = \{\{1,3\},\{2\}\}$

Consistent permutations:

$$132 \quad 312 \quad 213 \quad 231 \quad \sout{123} \quad \sout{321}$$

Sampling the permutation



$p(\pi \mid \mathbf{C}, \mathbf{x})$

What's $p(\pi \mid \mathbf{C})$?

Let $\Pi(\mathbf{C})$ = permutations consistent with $\mathbf{C}$ (all clusters contiguous in permutation)

Example:

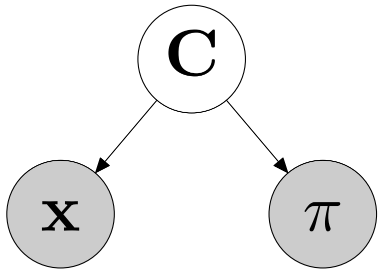
Clustering $\mathbf{C} = \{\{1,3\},\{2\}\}$

Consistent permutations:

$$132 \quad 312 \quad 213 \quad 231 \quad \cancel{123} \quad \cancel{321}$$

$$p(\pi \mid \mathbf{C}) = \text{uniform over } \Pi(\mathbf{C})$$
$$= \frac{1}{|\mathbf{C}|! \prod_{c \in \mathbf{C}} |c|!} \text{ if } \pi \in \Pi(\mathbf{C}), \text{ else } 0.$$

Sampling the clustering



$$p(\mathbf{C} \mid \pi, \mathbf{x}) \propto p(\mathbf{C})p(\mathbf{x} \mid \mathbf{C})p(\pi \mid \mathbf{C})$$

Number of consistent clusterings $\mathbf{C}$: 2^{n-1}

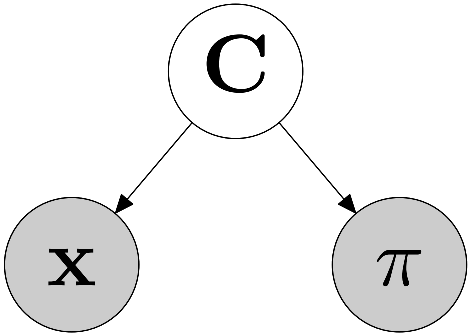
Example:

Permutation $\pi = 312$

Consistent clusterings $\mathbf{C}$:

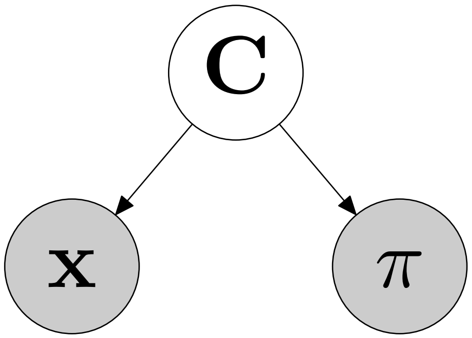
$$\{3\}, \{1\}, \{2\}$$
$$\{3, 1\}, \{2\}$$
$$\{3\}, \{1, 2\}$$
$$\{3, 1, 2\}$$

Sampling the clustering



$$p(\mathbf{C} \mid \pi, \mathbf{x}) \propto p(\mathbf{C})p(\mathbf{x} \mid \mathbf{C})p(\pi \mid \mathbf{C})$$

Sampling the clustering



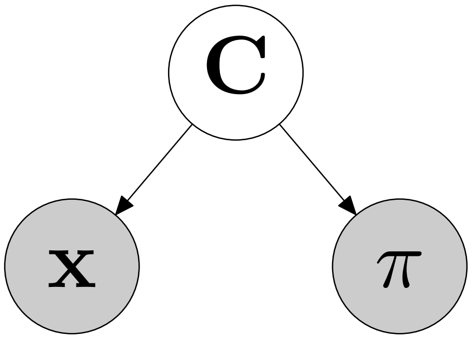
$$p(\mathbf{C} \mid \pi, \mathbf{x}) \propto p(\mathbf{C})p(\mathbf{x} \mid \mathbf{C})p(\pi \mid \mathbf{C})$$

$$p(\mathbf{C}) = \frac{1}{\mathcal{AF}(\alpha_0, n)} \prod_{c \in \mathbf{C}} \alpha_0(|c| - 1)!$$

$$p(\mathbf{x} \mid \mathbf{C}) = \prod_{c \in \mathbf{C}} p(\mathbf{x}_c)$$

$$p(\pi \mid \mathbf{C}) = \frac{1[\pi \in \Pi(C)]}{|\mathbf{C}|! \prod_{c \in \mathbf{C}} |c|!}$$

Sampling the clustering



$$p(\mathbf{C} \mid \pi, \mathbf{x}) \propto p(\mathbf{C})p(\mathbf{x} \mid \mathbf{C})p(\pi \mid \mathbf{C})$$

$$p(\mathbf{C}) = \frac{1}{\mathcal{AF}(\alpha_0, n)} \prod_{c \in \mathbf{C}} \alpha_0(|c| - 1)!$$

$$p(\mathbf{x} \mid \mathbf{C}) = \prod_{c \in \mathbf{C}} p(\mathbf{x}_c)$$

$$p(\pi \mid \mathbf{C}) = \frac{1[\pi \in \Pi(C)]}{|\mathbf{C}|! \prod_{c \in \mathbf{C}} |c|!}$$

$$p(\mathbf{C}, \pi, \mathbf{x}) = \underbrace{\frac{1[\pi \in \Pi(C)]}{\mathcal{AF}(\alpha_0, n)|\mathbf{C}|!}}_{\stackrel{\text{def}}{=} A(|\mathbf{C}|)} \prod_{c \in \mathbf{C}} \underbrace{\frac{\alpha_0 p(\mathbf{x}_c)}{|c|}}_{\stackrel{\text{def}}{=} B(c)}$$

DPDP

$$p(\mathbf{C}, \pi, \mathbf{x}) = A(|\mathbf{C}|) \prod_{c \in \mathbf{C}} B(c)$$

Goal: $p(\pi, \mathbf{x}) = \sum_{K=1}^{n} A(K) \underbrace{\sum_{\mathbf{C}: \pi \in \Pi(\mathbf{C}), |\mathbf{C}|=K} \prod_{c \in \mathbf{C}} B(c)}_{\stackrel{\text{def}}{=} g(n,K)}$

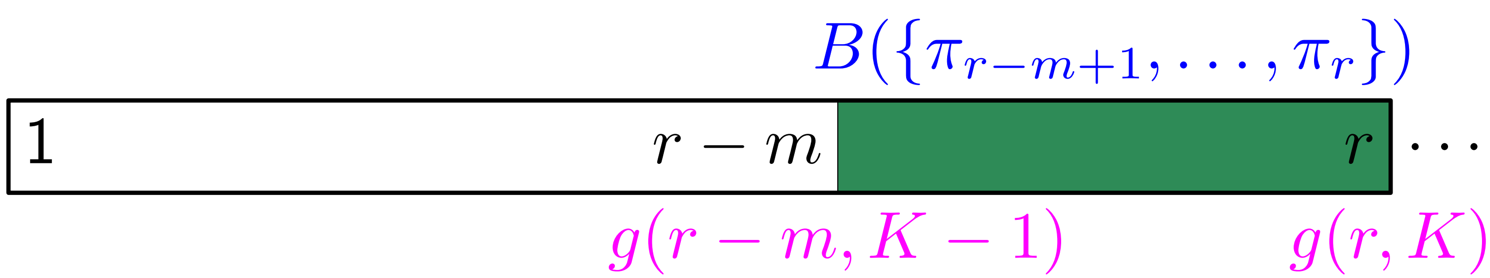
DPDP

$$p(\mathbf{C}, \pi, \mathbf{x}) = A(|\mathbf{C}|) \prod_{c \in \mathbf{C}} B(c)$$

Goal: $p(\pi, \mathbf{x}) = \sum_{K=1}^{n} A(K) \underbrace{\sum_{\mathbf{C}:\pi \in \Pi(\mathbf{C}), |\mathbf{C}|=K} \prod_{c \in \mathbf{C}} B(c)}_{\stackrel{\text{def}}{=} g(n,K)}$

$g(r, K) =$ sum over clusterings of $1 \dots r$ with K clusters

$$g(r, K) = \sum_{m=1}^{r} g(r-m, K-1) B(\{\pi_{r-m+1}, \dots, \pi_r\})$$



Running time: $O(n^3)$, space: $O(n^2)$

Optimizations

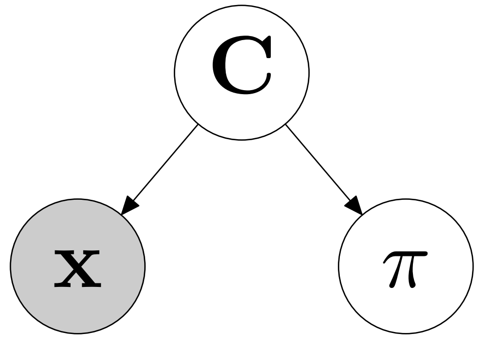
Current running time: $O(n^3)$, space: $O(n^2)$

$$p(\mathbf{C}, \pi, \mathbf{x}) = A(|\mathbf{C}|) \prod_{c \in \mathbf{C}} B(c)$$

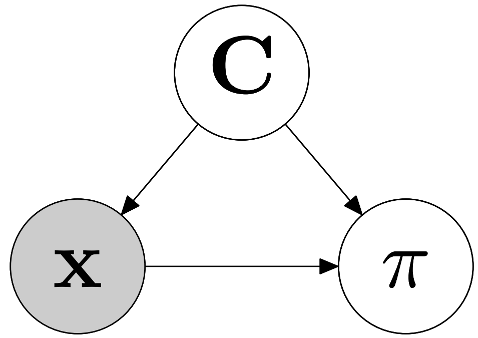
- Remove dependence on $|\mathbf{C}|$ to get MH proposal $\Rightarrow$ $O(n^2)$ dynamic program
- Use a beam $\Rightarrow$ $O(n)$ time

Final running time: empirically $O(n)$, space: $O(n)$

Data-dependent permutations

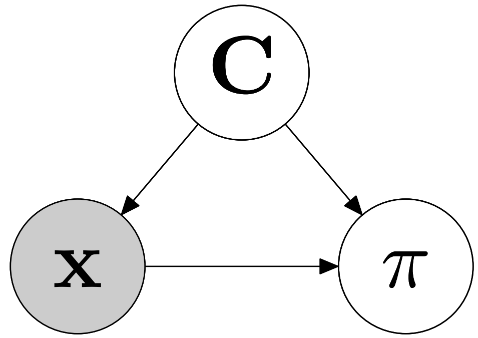


Data-dependent permutations



Goal: use data $\mathbf{x}$ to guide permutation—place similar points near each other

Data-dependent permutations



Goal: use data $\mathbf{x}$ to guide permutation—place similar points near each other

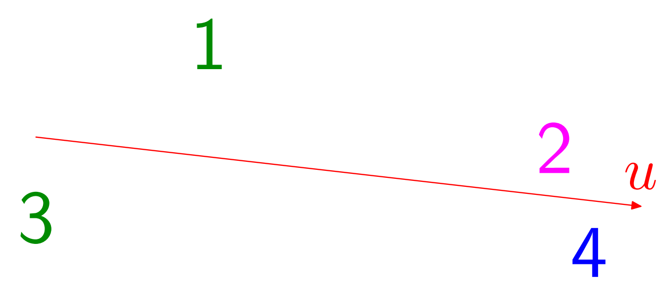
Two possible $p(\pi \mid \mathbf{C}, \mathbf{x})$:

- Markov Gibbs scans
- Random projections

Random projections

How to sample from $p(\pi \mid \mathbf{C}, \mathbf{x})$:

- Choose a random direction u
- Project points onto $u \Rightarrow$ induces permutation
- Note: keep clusters contiguous in permutation

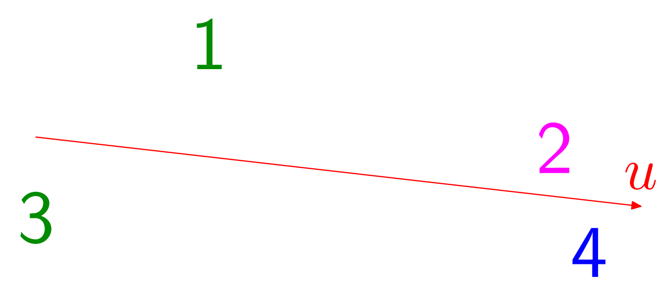


Permutation induced by projection u: 3 1 2 4

Random projections

How to sample from $p(\pi \mid \mathbf{C}, \mathbf{x})$:

- Choose a random direction u
- Project points onto $u \Rightarrow$ induces permutation
- Note: keep clusters contiguous in permutation



Permutation induced by projection u: 3 1 2 4

Computing $p(\pi \mid \mathbf{C}, \mathbf{x})$ is hard; ignore it $\Rightarrow$ stochastic hill-climbing algorithm

Experimental setup

Interleave different moves to form hybrid samplers:

GIBBS	Collapsed Gibbs [Escobar, West, 1995]
GIBBS+SPLITMERGE	With split-merge [Dahl, 2003]
GIBBS+PERM	With permutation (this paper)
GIBBS+SPLITMERGE+PERM	With all three moves

Experimental setup

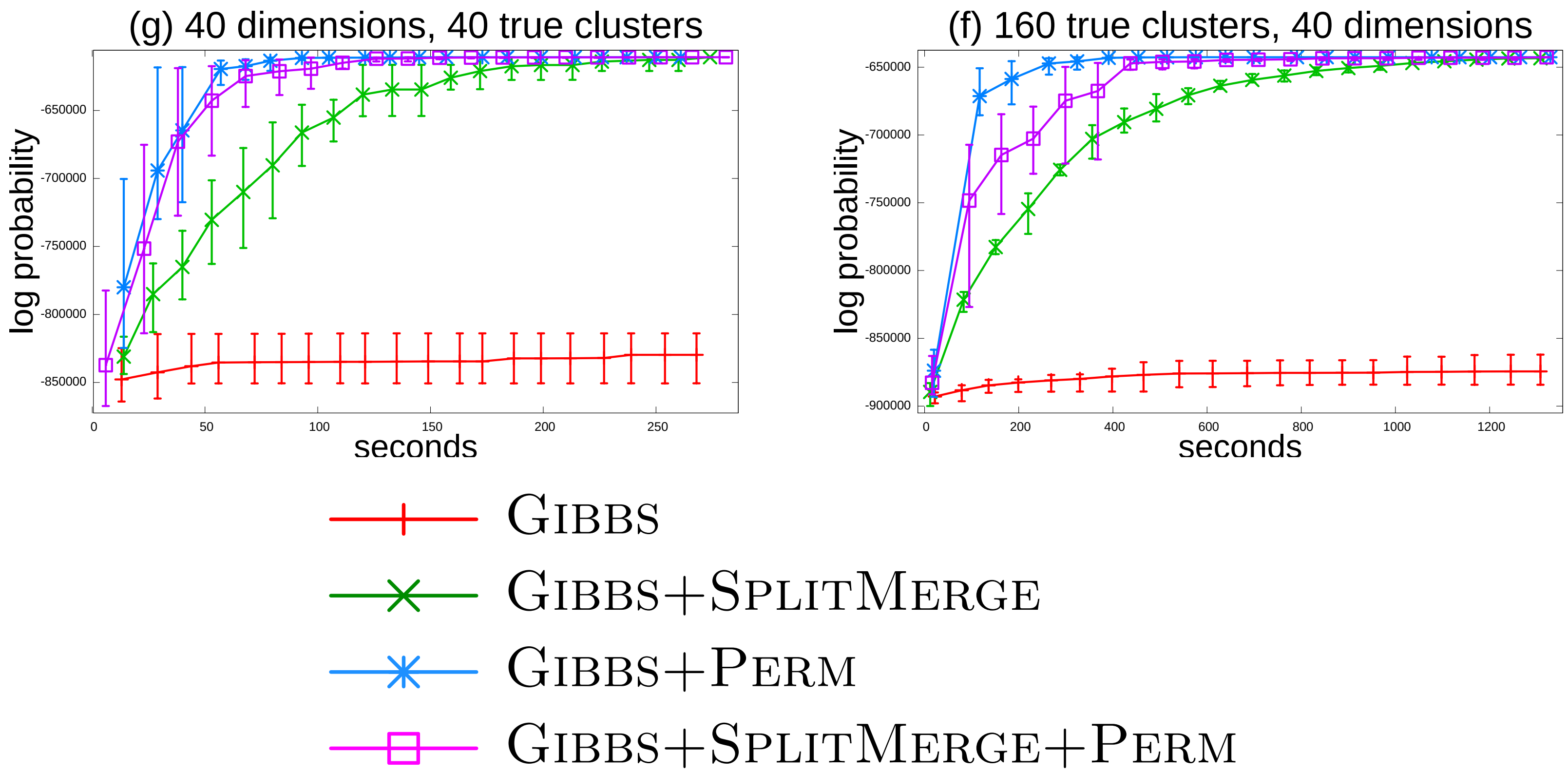
Interleave different moves to form hybrid samplers:

GIBBS	Collapsed Gibbs [Escobar, West, 1995]
GIBBS+SPLITMERGE	With split-merge [Dahl, 2003]
GIBBS+PERM	With permutation (this paper)
GIBBS+SPLITMERGE+PERM	With all three moves

- Run on synthetic Gaussians and two real-world datasets
- Evaluate on log-probability of clustering

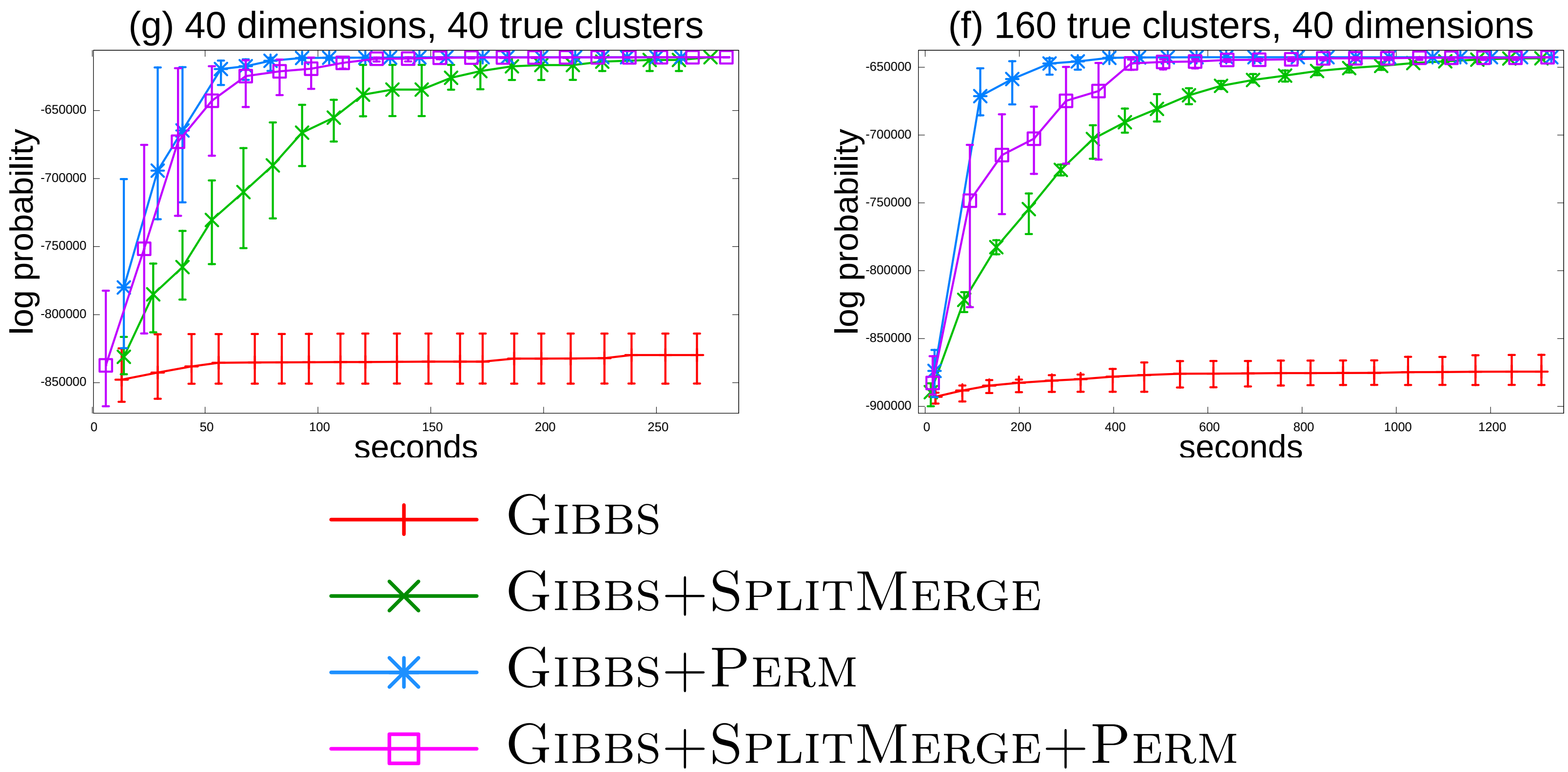
Synthetic Gaussians

Setup: generate mixture of Gaussians: 10,000 points, 10–80 dimensions, 20–160 true clusters



Synthetic Gaussians

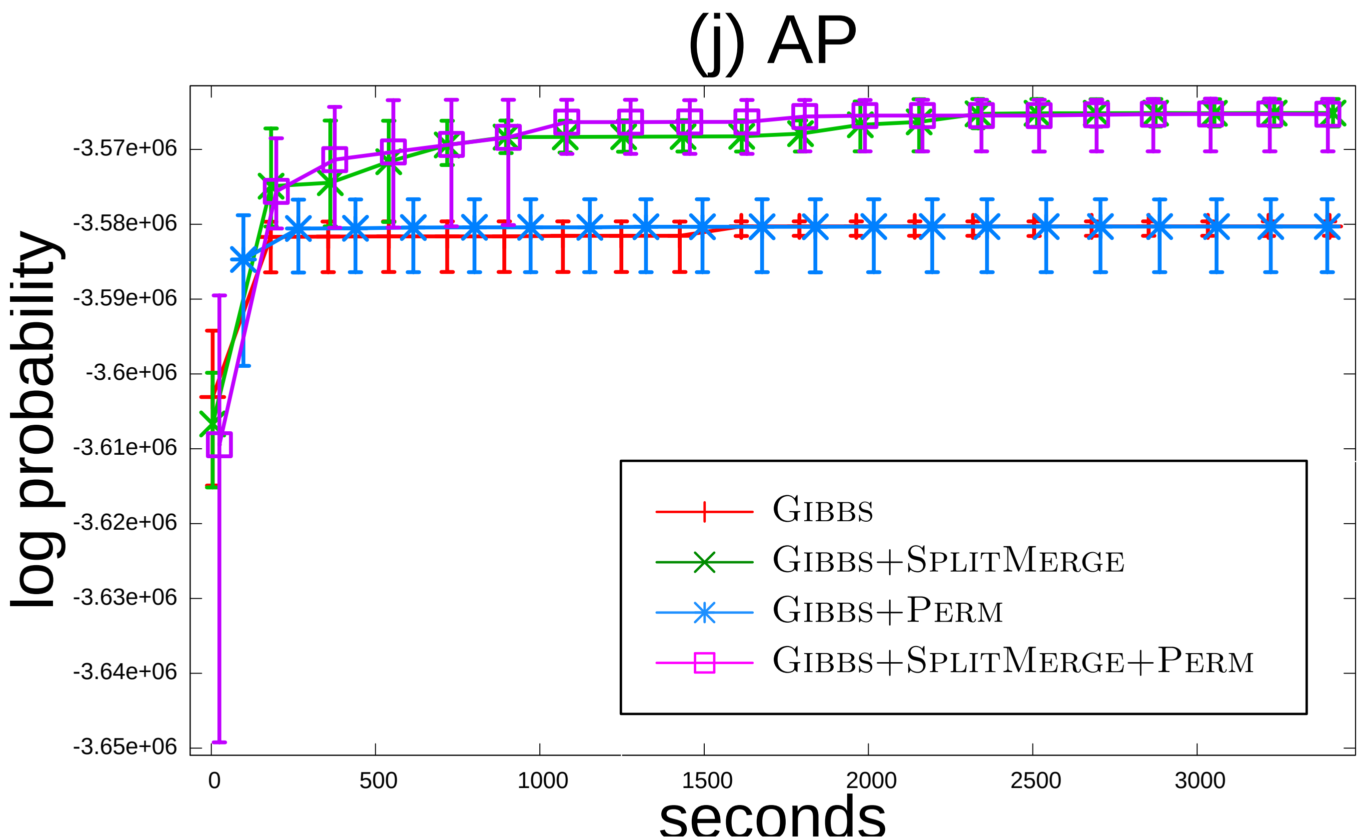
Setup: generate mixture of Gaussians: 10,000 points, 10–80 dimensions, 20–160 true clusters



- Gibbs+Perm significantly outperforms Gibbs
- Gibbs+Perm outperforms Gibbs+SplitMerge, especially when there are many clusters

AP dataset

2246 points, 10,473 dimensions [multinomial model]

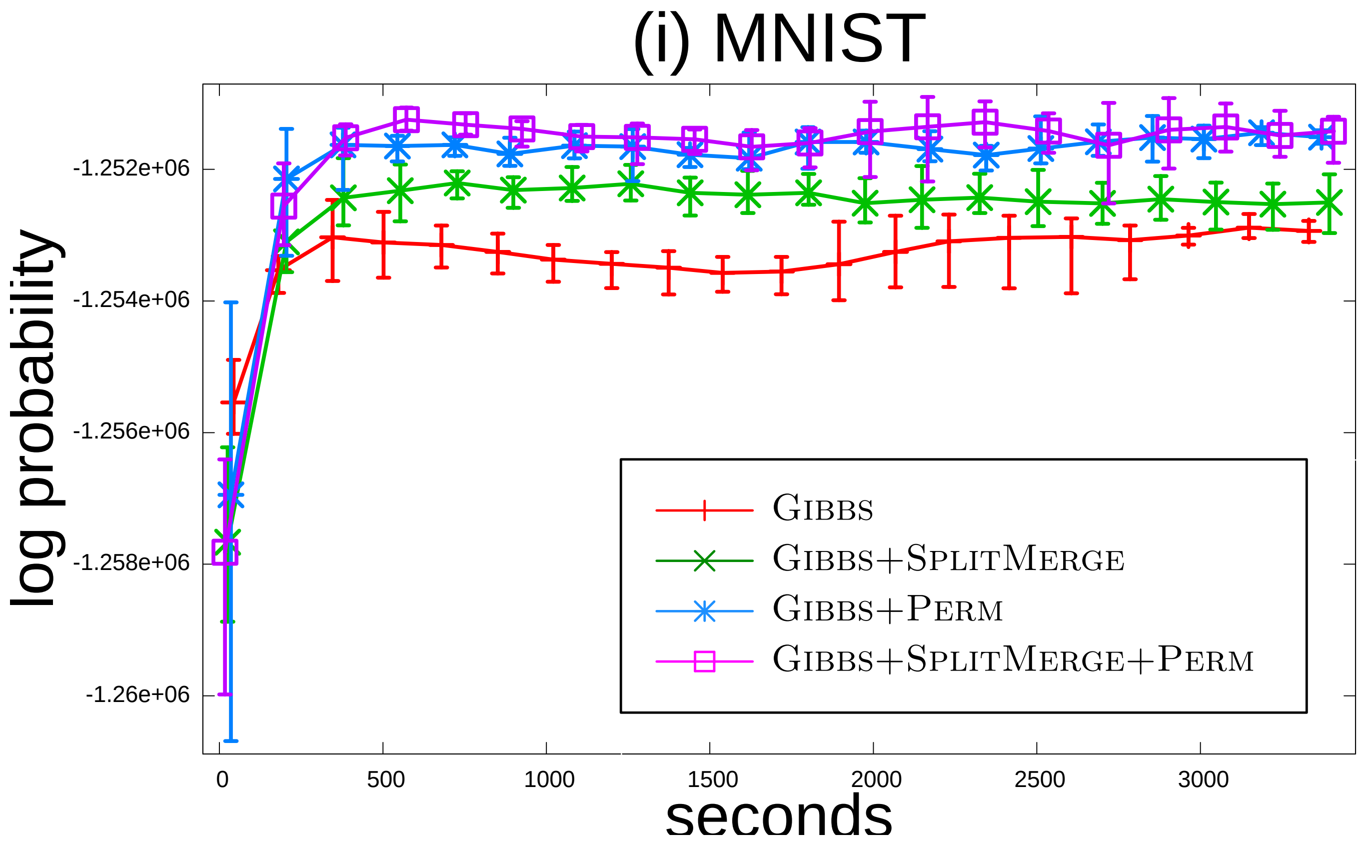


Gibbs+SplitMerge outperforms Gibbs+Perm

Gibbs+SplitMerge+Perm performs best

MNIST dataset

10,000 points, 50 dimensions (obtained via PCA on pixels) [Gaussian model]



GIBBS+PERM outperforms GIBBS+SPLITMERGE

GIBBS+SPLITMERGE+PERM performs best

Conclusions

- Inference algorithms for DP mixtures suffer from local minima when they make small moves
- Key idea: can use dynamic programming to sum over all clusterings consistent with a permutation
- Random projections yields effective stochastic hill-climbing algorithm

Conclusions

- Inference algorithms for DP mixtures suffer from local minima when they make small moves
- Key idea: can use dynamic programming to sum over all clusterings consistent with a permutation
- Random projections yields effective stochastic hill-climbing algorithm

What sampler should I use for my data?

- Gibbs is good at refining clusterings
- Split-merge is good when there are few clusters
- Permutation-augmented is good at changing many clusters at once

Conclusions

- Inference algorithms for DP mixtures suffer from local minima when they make small moves
- Key idea: can use dynamic programming to sum over all clusterings consistent with a permutation
- Random projections yields effective stochastic hill-climbing algorithm

What sampler should I use for my data?

- Gibbs is good at refining clusterings
- Split-merge is good when there are few clusters
- Permutation-augmented is good at changing many clusters at once

Combining all three often leads to best performance.