Aula 16

Aula passada

- Caminho amostral
- Teorema Ergódico
- Estimando π
- Simulação de CM
- Gerando amostras
- Markov Chain Monte Carlo (caso simétrico)
- Exemplo

Aula de hoje

- Metropolis-Hasting
- Amostrando vértices
- Modelo Hardcore
- Gibbs Sampling (ou Glauber Dynamics)
- Gerando q-colorações

O que é MCMC? – uma técnica para gerar amostras de uma dist.

Como? – Montar uma MC e a simular para amostrar os objetos

MH → a prob. depende apenas dos estados vizinhos

Markov Chain Monte Carlo

- **Problema:** gerar amostras de um espaço *S* grande e complicado com distribuição π
  - ex: amostrar grafos conexos com *n* vértices com probabilidade proporcional ao número de arestas
- **Solução:** Markov chain Monte Carlo
  - construir uma CM com espaço de estado *S* tal que π seja sua distribuição estacionária
  - gerar amostras através de caminhos amostrais de comprimento τ_ε
- **Dificuldade:** construir CM que seja eficiente em termos de calcular transições e tempo de mistura
  - arte com dicas de engenharia

Construindo a CM para MCMC

- **Ideia 1:** partindo de uma CM aperiódica e irredutível, transformar cadeia em outra mudando valores de transições não nulas, tq distribuição estacionária seja π
- **Ideia 2:** não "aceitar" todas as transições da CM original, continuar no mesmo estado para induzir π qualquer
  - para cada transição não-nula em P, temos probabilidade de rejeição $a(i, j)$
- **Problema:** encontrar $a(i, j)$ tq nova CM tenha distribuição π

Algoritmo (ou cadeia) de Metropolis-Hasting

- Aula passada: P simétrica, ou seja $P_{ij} = P_{ji}$
- P qualquer no próximo slide → não necessariamente simétrica

Metropolis-Hasting

- Considere CM no estado *i* e uma proposta de transição da CM original (ou seja, em *P*) para o estado *j*
  - aceita transição com probabilidade a(*i*, *j*), rejeita com complemento
- Matriz de transição *P'* da nova CM

$$P'_{ij} = \begin{cases} P_{ij}\, a(i,j), & se\, i \neq j \\ 1 - \sum_{k:k\neq i} P_{ik}\, a(i,k), & se\, i = j \end{cases} \rightarrow \text{LOOP}$$

- Temos que escolher a(i, j) tal que

$$\underbrace{\pi_i P_{ij}\, a(i,j)}_{P'_{ij}} = \underbrace{\pi_j P_{ji}\, a(j,i)}_{P'_{ji}} \rightarrow \text{1 equação, 2 variáveis}$$

  - garante que π é distribuição estacionária de *P'*
  - garante que *P'* é reversível

Metropolis-Hasting

- Como a(i,j) e a(j,i) são probabilidades, temos

$$\pi_i P_{ij} a(i,j) \leq \pi_i \qquad \pi_i P_{ij} a(i,j) = \pi_j P_{ji} a(j,i) \leq \pi_j$$

- Como *a(i, j)* deve ser o maior possível para evitar desperdício, temos

Fluxo de $i \to j$

Se o fluxo de $i \to j$ é menor que $j \to i$ então sempre aceito a prob. de transição

$$a(i,j) = 1, \quad se\; \pi_i P_{ij} \leq \underbrace{\pi_j P_{ji}}$$

Fluxo de $j \to i$

$$a(i,j) = \frac{\pi_j P_{ji}}{\pi_i P_{ij}}, \quad se\; \pi_i P_{ij} > \pi_j P_{ji}$$

Caso 1: $a_{ij} = 1 \Rightarrow$

$\pi_i \cdot P_{ij} = \pi_j \cdot P_{ji} \cdot a_{ji}$ → é o menor

- Ou seja,

$$a(i,j) = min\left\{1, \frac{\pi_j P_{ji}}{\pi_i P_{ij}}\right\}$$

$\{a(i,j)$ e $a(j,i)\}$ → um deles é 1

$$a_{ji} = min\left\{1, \frac{\pi_i \cdot P_{ij}}{\pi_j \cdot P_{ji}}\right\} = \frac{\pi_i \cdot P_{ij}}{\pi_j \cdot P_{ji}}$$

FAZER CASO 2

- Entradas de *P'* calculadas desta forma

Amostrando Vértices

- Considere grafo grande e desconhecido (não conhecemos os vértices ou as arestas, a priori)
  - ex. grafo de amizades do facebook
- Podemos percorrer o grafo: a partir de um vértice, descobrir seus vizinhos
- Como gerar amostras de vértices uniformemente?
  - ex. estimar fração de brasileiros no FB

→ pega os vizinhos, depois os vizinhos dos vizinhos, e retorna um vértice aleatório

- **Ideia 1:** BFS de raio *k*, amostrar uniforme no vértices descobertos

A amostra é enviesada pela quantidade de amigos que um vértice tem

- **Ideia 2:** Passeio aleatório de comprimento *k*, retornar vértice X_k

→ Posso transformar uma MC em outra MC com distribuição π uniforme

Como garantir que amostra é uniforme?

- Ideias 1 e 2 geram amostras enviesadas (não uniformes)

Figueiredo 2018

MCMC to the Rescue

- Construir CM usando algoritmo de Metropolis-Hasting tq π seja uniforme nos vértices π_v é a distribuição estacionária da cadeia que queremos gerar.
  - $\pi_v = 1/Z$ onde Z é o número de vértices da rede (desconhecido)
    ↳ poderia não ser uniforme
- Modificar CM induzida por um passeio aleatório (que não é simétrica)
- Definindo probabilidade de aceite → depende apenas de onde estou partindo
  - CM original, temos: $P_{ij} = 1/d_i$, $P_{ji} = 1/d_j$

→ NÃO precisamos de Z

CM desejada

$$a(i,j) = min\left\{1, \frac{\pi_j P_{ji}}{\pi_i P_{ij}}\right\} = min\left\{1, \frac{d_i}{d_j}\right\}$$

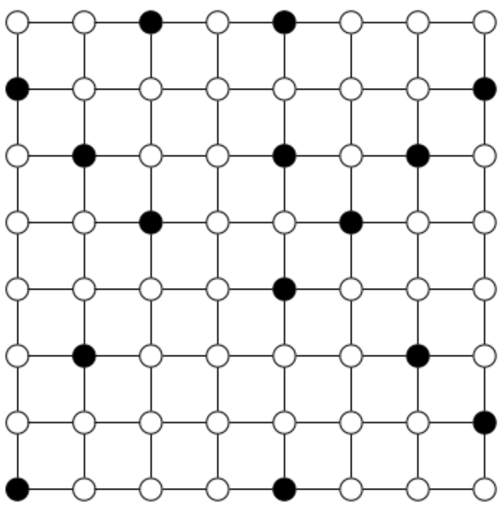
$$\boxed{\pi_i = \pi_j = \frac{1}{Z}}$$

Se d_j grande, há uma menor probabilidade de aceite. Logo, remove o viés dos vértices com grau alto no passeio aleatório

- Logo $P'_{ij} = 1/d_i * \min\{ 1, d_i / d_j \}$
- Enviesa o passeio contra vértices de grau alto, dando distribuição estacionária uniforme
- Método usado na prática, hoje em dia!

Exemplo – *Hardcore Model*

- Modelo *hardcore*: atribuir bolinhas aos vértices de um grafo tal que vértices vizinhos não possuam bolinhas
  - vértice pode ter 0 ou 1 bolinha



- Dado um grafo G=(V, E), *S* é o conjunto de todas as configurações, e *S'* o subconjunto de configurações válidas
  - $|S| = 2^n$, para grafo com n vértices

- **Problema:** qual é o valor esperado do número de bolinhas de uma configuração?
  - distribuição uniforme sobre o espaço *S'*

O número máximo de bolinhas ligadas é $n/2$

Em geral, a dificuldade em aplicar Metropolis-Hastings está em definir a CM inicial (seus estados e transições).

Exemplo – *Hardcore Model*

- Seja n(C) o número de bolinhas de uma configuração C, escolhida ao acaso (uniformemente)
- Então

quantas bolas pretas há na configuração C

$$E[n(C)] = \frac{1}{Z} \sum_{c \in S'} n(c) \qquad Z = |S'| = \sum_{c \in S} 1(c\ \acute{e}\ v\acute{a}lida)$$

prob. de escolher uma config. válida

→ conjunto de config. válida

Quantas configurações existem

- **Problemas:**
  (1) como enumerar *S'* de forma eficiente?

  (2) como lidar com o tamanho de *S'* (exponencial)?

- **Solução:** Markov Chain Monte Carlo

Construir uma MC onde os estados são configurações válidas.

MCMC → técnica para gerar amostras com qualquer distribuição

MCMC para *Hardcore Model*

- Construir CM onde estados são elementos de *S'*
  - configurações válidas
- Fazer com que distribuição estacionária desta CM seja uniforme, $\pi_c = 1/Z$ para toda configuração *c*
- Gerar caminho amostral pela CM e usar configurações
  - X_k estado da CM no passo *k* → O problema aqui é o tempo de mistura. Gerar 1 amostra não é um problema, mas estimar a média requer muitas amostras
- Monte Carlo para estimar valor esperado

bolas pretas no k-ésimo estado

$$\hat{n}(k) = \frac{1}{k}\sum_{l=1}^{k} n(X_k) \xrightarrow{\text{converge quando } k \to \text{infinito (teorema ergódico)}} E[n(C)]$$

k-ésima config válida

na distribuição $\pi_c = 1/Z$ (uniforme)

Estimador do número médio de bolinhas após *k* amostras

Andaremos pela cadeia e utilizaremos o teorema ergódico para gerar amostras independentes

MCMC para *Hardcore Model*

- Construir a CM é equivalente a definir as transições para cada estado Dada uma configuração válida, se eu apagar uma bola eu vou para outra config. válida
  - cada configuração leva a outras configurações
- **Algoritmo** → Só terei transições entre vértices que diferem de 1 vértice

  1) Escolher vértice v do grafo aleatoriamente

  Se houver ao menos 1 vizinho preto, então o vértice vai p/ branco

  2) Se todos os vizinhos de v estão em 0, colocar v em 1 com probabilidade ½; caso colocar v em 0

  3) Manter configuração de todos os outros vértices
- CM acima é aperiódica e irredutível, e possui distribuição estacionária uniforme
  - usar equações de reversibilidade para mostrar π

$$\pi_i P_{ij} = \pi_j P_{ji}$$

i e j são duas configurações válidas

Mostrando

no vértice que faz $d(i,j)=1$, todos os vizinhos do vértice branco são brancos (pois o vértice correspondente na outra configuração é preto)

$$\pi_i P_{ij} = \pi_j P_{ji}$$

- Seja i e j duas configurações quaisquer em S'
- Seja d(i,j) o número de vértices em que i e j diferem
- d(i,j) = 0 → as duas configurações são idênticas (vale a relação)
- d(i,j) > 1 → não temos transições, pois apenas um vértice troca de valor por vez (vale a relação)

estou interessado nesta onde apenas 1 vértice difere

- d(i,j) = 1 → diferem na configuração de um único vértice v

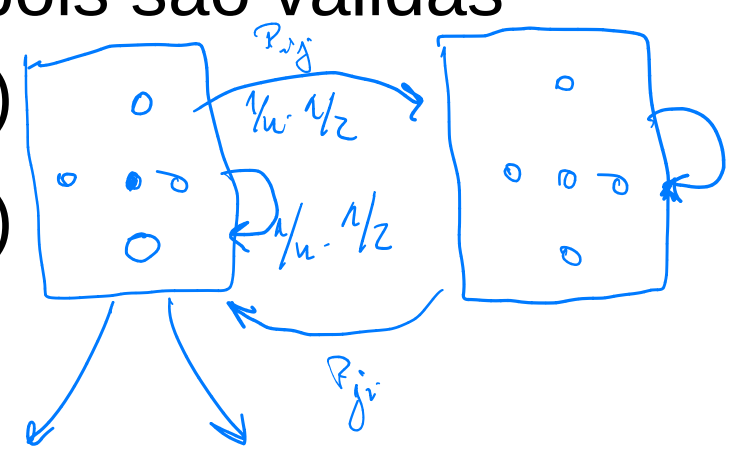
1 vértice só possui n transições de saída neste caso

  - vizinhos de v são todos zero, pois são válidas

prob de escolher um vértice

  - P_{ij} = 1/n*1/2 (v em 1 → v em 0)
  - P_{ji} = 1/n*1/2 (v em 0 → v em 1)
  - Logo, $\pi_i = \pi_j$ = 1/Z

para todos os vértices conectados



Amostragem de Gibbs

- Problema anterior é um exemplo de *Gibbs Sampling* ou *Glauber Dynamics*
  - algoritmo genérico para construção de CM para MCMC
- Considere espaço amostral S do tipo K^V onde K e V são conjuntos finitos
  - cada elemento de V assume possíveis valores de K
  - ex. anterior, V = vértices do grafo, $K = \{0,1\}$
- Considere uma distribuição qualquer π sobre S
- CM construída assim a partir de X_t para o tempo $t+1$

1) Escolher um elemento v de V uniformemente

2) Escolher $X_{t+1}(v)$ com distribuição π condicionada em todos os outros elementos de V assumirem valor em X_t

3) $X_{t+1}(w) = X_t(w)$ para todo w diferente de v

PAROU AQUI

Amostragem de Gibbs

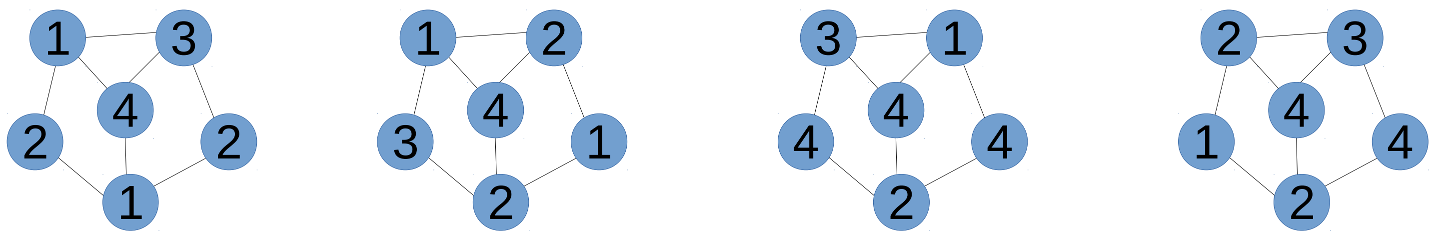
- CM de amostragem de Gibbs é aperiódica e reversível com distribuição estacionária dada por π , ou seja

$$\pi_i P_{ij} = \pi_j P_{ji}$$

- Podemos verificar propriedade acima (exercício)
- Usada para gerar amostras de *S* com distribuição π
- Usada para aproximar valor esperado de uma função sobre *S* com distribuição π
- O que é escolher novo valor a_k para $v = v_i$ com distribuição π condicionada?
  - $P[X(v_i) = a_k \mid X(v_1) = a_1, X(v_2) = a_2, \dots X(v_n) = a_n]$

q-Coloração

- Uma coloração dos vértices de um grafo que usa exatamente *q* cores
  - vértices adjacentes tem cores diferentes
- Exemplos de 4-colorações



- **Problema:** gerar uma *q*-coloração para um grafo qualquer uniformemente
  - conexão com problema de coloração mínima, que é NP-Completo

Gibbs Sampler para *q*-coloração

- V = vértices do grafo, K = cores $\{1,\ldots,q\}$
- $S = V_K$ todas possíveis colorações (incluindo não-válidas)
- *Lema*: dado uma coloração válida, a distribuição condicional das cores que um determinado vértice pode assumir é uniforme nas possíveis cores
- Algoritmo para amostragem de Gibbs, dada uma coloração válida no tempo t, X_t

  1) Escolher vértice v uniformemente

  2) Escolher $X_{t+1}(v)$ (nova cor para v) uniformemente entre as cores válidas para v em X_t

  3) Não modificar nenhuma outra cor: $X_{t+1}(w) = X_t(w)$

- Gerar amostra uniforme gerando caminho amostral longo o suficiente e retornando X_τ

Gibbs Sampler para *q*-coloração

- Gibbs Sampler anterior tem distribuição uniforme
- Considere duas colorações i e j que diferem em apenas um vértice v (único com cor diferente)
  - então $P_{ij} = P_{ji}$, pois cores válidas para *v* em *i* é igual as cores válidas para *v* em *j*
- Se *i* e *j* diferem em mais cores, então $P_{ij} = P_{ji} = 0$
- Logo, temos que

$$\pi_i P_{ij} = \pi_j P_{ji} \longrightarrow \pi_i = \pi_j = \frac{1}{Z}$$

Número de colorações válidas

Convergência deste Gibbs Sampler

- Considere variação onde vértice a ter cor alterada é escolhido sequencialmente (ao invés de aleatoriamente)
  - v_1, v_2, v_3, …, v_n, v_1, ….
- Considere que $q > 2d^2$, onde d é o grau máximo de G (q é número de cores), e considere $\varepsilon > 0$
- Então, para CM definida acima, existe uma constante C (que não depende de n ou ε), tal que

$$\tau_\epsilon = C\, n \log\left(n + \log 1/\epsilon\right)$$

- CM é *fast mixing*, pois tempo de mistura é polylog em n apesar de número de estados ser exponencial
- Na prática, funciona!