CPE723 Otimização Natural:
Lista de Exercícios #2

Data de entrega: Quinta, 28 de Março, 2019

Prof. José Gabriel Rodríguez Carneiro Gomese, Terças e Quintas: 08:00-10:00

Vinicius Mesquita de Pinho

Questão 1

Considere um processo de Markov $X(t)$ que tem três estados possíveis: 0, 1 e 2. A evolução temporal deste processo é dada pela matriz de transição a seguir:

$$M = \begin{bmatrix} 0.5 & 0.25 & 0.25 \\ 0.25 & 0.50 & 0.25 \\ 0.25 & 0.25 & 0.50 \end{bmatrix}$$

a) Considerando que a distribuição de probabilidade de $X(0)$ é dada pelo vetor $\mathbf{p}_0 = (0.3\ 0.4\ 0.3)^{\mathrm{T}}$, calcule a distribuição de probabilidade de $X(3)$ (ou seja, do processo de Markov no instante $t = 3$).

A distribuição de probabilidade de $X(1)$ pode ser calculada a partir da cadeia de Markov da seguinte maneira:

$$\mathbf{p}_1 = \mathbf{M}\mathbf{p}_0.$$

A distribuição de probabilidade de $X(2)$, seguindo a mesma ideia:

$$\mathbf{p}_2 = \mathbf{M}^2\mathbf{p}_1.$$

Logo, a distribuição de probabilidade de $X(3)$, pode ser obtida por:

$$\mathbf{p}_3 = \mathbf{M}^3\mathbf{p}_2.$$

Usando os valores dados pela questão, obtemos o seguinte vetor $\mathbf{p}_3^{\mathrm{T}} = (0.3593\ 0.3489\ 0.3463)$.

b)Iniciando em $X(0) = 1$, e usando um gerador de números aleatórios (são necessários apenas três números aleatórios equiprováveis), calcule manualmente uma amostra do processo X(t) até $t = 3$.

O código usado para a questão.

```
M = np.matrix('0.50 0.25 0.25; 0.25 0.50 0.25; 0.25 0.25 0.50')
estado_atual = 1
estados = np.array(estado_atual)
tempos_possiveis = np.array([1, 2, 3])
soma_coluna = 0

iterador = 0

for t in tempos_possiveis:
    uniforme_sorteado = np.random.uniform(0,1)
    while uniforme_sorteado > soma_coluna:
        soma_coluna = soma_coluna + M[estado_atual,iterador]
        iterador += 1
    print(f'no tempo {t} => mudança de estado: de {estado_atual} para
    ... {iterador}. uniforme sorteado foi {uniforme_sorteado:.3}')
    estados = np.append(estados, iterador)
    estado_atual = iterador
```

As saídas encontradas:
No tempo 1 => mudança de estado: de 1 para 2. Uniforme sorteado foi 0.593.
No tempo 2 => mudança de estado: de 2 para 2. Uniforme sorteado foi 0.497.
No tempo 3 => mudança de estado: de 2 para 2. Uniforme sorteado foi 0.597.

c) Usando um computador, execute 100 repetições do item (b). Em cada uma das 100 repetições, comece a simulação com um valor diferente de $X(0)$, assumindo que os eventos $X(0) = 0$, $X(0) = 1$ e $X(0) = 2$ são equiprováveis. Armazene as 100 cadeiras obtidas em uma matriz X, com 4 colunas ($t = 0$ até $t = 3$) e 100 linhas.

d) Fazendo histogramas de cada uma das 4 colunas, calcule as distribuições de probabilidade do processo $X(t)$ em cada um dos 4 instantes: $t = 0,\ 1\,2\,3$. Comente os resultados obtidos.

Questão 2

Considere um sistema em que só há 5 estados possíveis: $x = 1$, $x = 2$, $x = 3$, $x = 4$ e $x = 5$. Os custos de $J(x)$ De cada um dos estados são indicados na tabela abaixo:

x	$J(x)$
1	0.5
2	0.2
3	0.3
4	0.1
5	0.5

a) Considere um processo de Markov gerado pela aplicação do algoritmo de Metropolis aos dados da tabela acima, com temperatura fixa $T = 0.1$. Calcule a matriz de transição M que define o processo de $X(t)$. Obs: note que o estado $X(t)$ é unidimensional, e portanto a matriz M é 5×5.

b) Iniciando em $X(0) = 1$, calcule manualmente 4 amostras do processo X(t).

c) Qual é o vetor invariante da matriz M do item (a)? Obs: para facilitar os cálculos, pode-se usar o computador neste item.

d) Calcule os fatores de Boltzmann (ou seja, $e^{-(J(x)/T)}$) associados aos dados da tabela acima, e compare-os com o resultado do item (c). Use $T = 0.1$.

e) Simulated Annealing: Usando um computador, execute 1000 iterações do algoritmo de Metropolis em cada uma das 10 temperaturas a seguir. Na passagem de uma temperatura para a outra, use o estado atual. Comente as distribuições de probabilidade obtidas no final de cada temperatura.

Questão 3

Proponha uma função $J(\mathbf{x})$, sendo $\mathbf{x}$ um vetor com 10 dimensões, cujo ponto mínimo você conheça. Evite propor funções que tenham um só ponto mínimo. Encontre o ponto mínimo global utilizando S.A. Obs: Neste exercício, entregue o código utilizando e alguns comentários sobre o resultado obtido.