**CPE-723 – Otimização Natural**

**Lista de Exercícios #2**

**Amanda Isabela de Campos (DRE 120074842)**

1. **Considere um processo de Markov *X*(*t*) que tem trˆes estados poss´ıveis: 0, 1, e 2. A evolu¸c˜ao temporal deste processo ´e dada pela matriz de transi¸c˜ao a seguir:**
   1. **Considerando que a distribui¸c˜ao de probabilidade de *X*(0) ´e dada pelo vetor p0 = [0*.*3 0*.*4 0*.*3]*T*, calcule a distribui¸c˜ao de probabilidade de *X*(3) (ou seja, do processo de Markov no instante *t* = 3).**

import numpy as np

M = np.array([[0.5,0.25,0.25],[0.25,0.50,0.25], [0.25, 0.25, 0.50]])

p0 = np.array([[0.3],[0.4], [0.3]])

M2 = M.dot(M)

M3 = M2.dot(M)

print(M3.dot(p0))

[[0.3328125], [0.334375 ], [0.3328125]]

* 1. **Iniciando em X(0) = 1, e usando um gerador de números aleatórios (são necessários apenas três números aleatórios, sorteados de PDF uniforme entre 0 e 1), calcule manualmente uma amostra do processo *X*(*t*) at´e *t* = 3.**

a = np.random.uniform(0.00,0.25,1) #0.135726

b = np.random.uniform(0.25,0.75,1) #0.743673

c = np.random.uniform(0.75,1.00,1) #0.900077

p0 = np.array([a,b,c])

M = np.array([[0.5,0.25,0.25],[0.25,0.50,0.25], [0.25, 0.25, 0.50]])

M2 = M.dot(M)

M3 = M2.dot(M)

print(M3.dot(p0))

[[0.5860114 ], [0.59551058], [0.59795439]]

* 1. **Usando um computador, execute 100 repeti¸c˜oes do item (b). Em cada uma das 100 repeti¸c˜oes, comece a simula¸c˜ao com um valor diferente de *X*(0), assumindo que os eventos *X*(0) = 0, *X*(0) = 1, e *X*(0) = 2 s˜ao equiprovaveis. Armazene as 100 cadeias obtidas em uma matriz X, com 4 colunas (*t* = 0 at´e *t* = 3) e 100 linhas.**

M = np.array([[0.5,0.25,0.25],[0.25,0.50,0.25], [0.25, 0.25, 0.50]])

M2 = M.dot(M)

M3 = M2.dot(M)

C = np.zeros((100,4))

for i in range(100):

prob = np.random.choice([0,1,2])

if prob == 0:

a = np.random.uniform(0.00,0.50,1)

b = np.random.uniform(0.50,0.75,1)

c = np.random.uniform(0.75,1.00,1)

p0 = np.array([a,b,c])

if prob == 1:

a = np.random.uniform(0.00,0.25,1)

b = np.random.uniform(0.25,0.75,1)

c = np.random.uniform(0.75,1.00,1)

p0 = np.array([a,b,c])

if prob == 2:

a = np.random.uniform(0.00,0.25,1)

b = np.random.uniform(0.25,0.50,1)

c = np.random.uniform(0.50,1.00,1)

p0 = np.array([a,b,c])

C[i,0] = np.linalg.norm(p0)

C[i,1] = np.linalg.norm(M.dot(p0))

C[i,2] = np.linalg.norm(M2.dot(p0))

C[i,3] = np.linalg.norm(M3.dot(p0))

* 1. **Fazendo histogramas de cada uma das 4 colunas, calcule as distribui¸c˜oes de probabilidade do processo *X*(*t*) em cada um dos 4 instantes: *t* = 0, 1, 2, 3. Comente os resultados obtidos.**

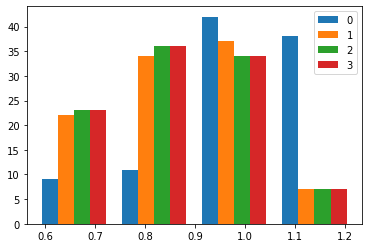
import matplotlib.pyplot as plt

plt.figure(1)

plt.hist(C, 4)

plt.legend('0123')

plt.show()

****

1. **Considere um sistema em que s´o h´a 5 estados poss´ıveis: *x* = 1, *x* = 2, *x* = 3, *x* = 4, *x* = 5. Os custos *J* (*x*) de cada um dos estados s˜ao indicados na tabela abaixo:**

|  |  |
| --- | --- |
| ***x*** | ***J* (*x*)** |
| **1** | **0*.*5** |
| **2** | **0*.*2** |
| **3** | **0*.*3** |
| **4** | **0*.*1** |
| **5** | **0*.*4** |

* 1. **Considere um processo de Markov gerado pela aplica¸c˜ao do algoritmo de Metropolis aos dados da tabela acima, com temperatura fixa *T* = 0*.*1. Calcule a matriz de transi¸c˜ao *M* que define o processo *X*(*t*). Obs.: note que o estado *X*(*t*) ´e unidimensional, e portanto a matriz *M* ´e 5 *×* 5.**

(prob. de sortear 5 a partir de 1) x (prob. de aceitar 5 a partir de 1) =

* 1. **Iniciando em *X*(0) = 1, calcule manualmente 4 amostras do processo *X*(*t*).**
  2. **Qual ´e o vetor invariante da matriz *M* do item (a) ?**

**Obs.: para facilitar os c´alculos, pode-se usar o computador neste item.**

* 1. **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.**

|  |  |  |
| --- | --- | --- |
| *x* | *J* (*x*) | *e-(J(x))/T* |
| 1 | 0*.*5 | 0.00673795 |
| 2 | 0*.*2 | 0.13533528 |
| 3 | 0*.*3 | 0.04978707 |
| 4 | 0*.*1 | 0.367879441 |
| 5 | 0*.*4 | 0.018315639 |

* 1. ***Simulated Annealing:* Usando um computador, execute 1000 itera¸c˜oes 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 distribui¸c˜oes de probabilidade obtidas no final de cada temperatura.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***T*0** | ***T*1** | ***T*2** | ***T*3** | ***T*4** | ***T*5** | ***T*6** | ***T*7** | ***T*8** | ***T*9** |
| **0*.*1000** | **0*.*0631** | **0*.*0500** | **0*.*0431** | **0*.*0387** | **0*.*0356** | **0*.*0333** | **0*.*0315** | **0*.*0301** | **0*.*0289** |

import numpy as np

import matplotlib.pyplot as plt

##

def Custo(x):

if x == 1:

J = 0.5

if x == 2:

J = 0.2

if x == 3:

J = 0.3

if x == 4:

J = 0.1

if x == 5:

J = 0.4

return J

N = 1000

M = 0.1\*N

T = np.zeros(10)

T[0] = 0.1; T[1] = 0.0631; T[2] = 0.05; T[3] = 0.0431; T[4] = 0.0387

T[5] = 0.0356; T[6] = 0.0333; T[7] = 0.0315; T[8] = 0.0301; T[9] = 0.0289

x = np.zeros(N)

n = 0

x[n] = np.random.choice([1,2,3,4,5])

J0 = Custo(x[0]); Xatual = x[0]; Jatual = J0;

i = 0; fim = 0

k = 1; K=7

while not(fim):

T = T[i]

i += 1

for n in range(N):

perturbacao = np.random.choice([-1,1])

if x[n] == 5:

if perturbacao == 1:

x\_hat = 1

elif x[n] == 1:

if perturbacao == -1:

x\_hat= 5

else:

x\_hat = x[n-1] + perturbacao

if np.random.uniform(0,1)< np.exp(((x[n-1])\*\*2 - (x\_hat)\*\*2)/T):

x[n] = x\_hat

else:

x[n] = x[n-1]

if np.remainder(n+1,100)==0:

print([k,n+1,Xmin,Jmin])

k+=1

if k==K: fim=1

y = np.linspace(0,6,1000)

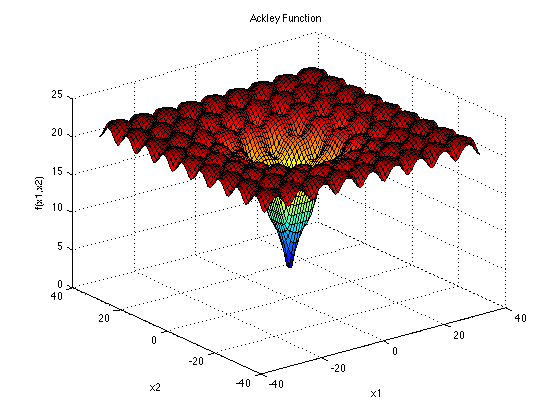
plt.figure(1)

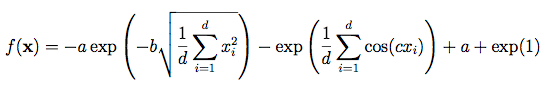
plt.hist(x[M:], 5)

plt.show()

1. **Proponha uma fun¸c˜ao *J* (x), sendo x um vetor com 10 dimens˜oes, cujo ponto m´ınimo vocˆe conhe¸ca. Evite propor fun¸c˜oes que tenham um s´o ponto m´ınimo. Encontre o ponto m´ınimo global utilizando S.A. Obs.: neste exerc´ıcio, entregue o c´odigo utilizado e alguns coment´arios sobre o resultado obtido.**

Função escolhida ACKLEY FUNCTION (Referência: <https://www.sfu.ca/~ssurjano/ackley.html>) que segue a expressão a seguir e nesse caso, será adotado d = 10 para representar uma função de 10 dimensões.





Com valores recomendados para as constantes como: a = 20, b = 0.2 e c = 2π. Tem-se que o ponto de mínimo global é: f(x) = 0 em x = (0,...,0). A seguir está o código adotando o Simulated Annealing para a busca desse ponto de mínimo global.

1. **Prova de 2009 - Questão 2, itens (a) e (c).**

**(*Simulated Annealing*) Considere um problema de otimização representado pela função custo a seguir:**

|  |  |
| --- | --- |
| **x** | **J(x)** |
| **1** | **0.3** |
| **2** | **0.1** |
| **3** | **0.1** |
| **4** | **0.2** |

**a) Calcule os fatores de Boltzmann *eJ*(x)*/T* , para *T* = 1.0 e para *T* = 0*.*1.**

**c) Calcule as matrizes de transição M para *T* = 1*.*0 e para *T* = 0*.*1. Calcule os vetores invariantes destas matrizes e compare-os com os resultados do item (a).**

1. **Prova de 2011 - Questão 2, itens (a), (b), e (e).**

**(*Simulated Annealing*) Considere a fun¸c˜ao custo *J*(*x*1*; x*2) definida pela tabela a seguir:**

|  |  |  |
| --- | --- | --- |
| ***x*1** | ***x*2** | ***J*(*x*)** |
| **0** | **0** | **0.2** |
| **0** | **1** | **0.3** |
| **1** | **0** | **0.3** |
| **1** | **1** | **0.1** |

**a) A aplicação do Algoritmo de Metropolis a um vetor inicial x(0) qualquer, alterando uma componente (*x*1 ou *x*2) de cada vez, define um processo de Markov com duas matrizes de transição: M1 e M2. Calcule estas matrizes de transição, considerando *T* = 0.5. Note que o número de estados possíveis é 4.**

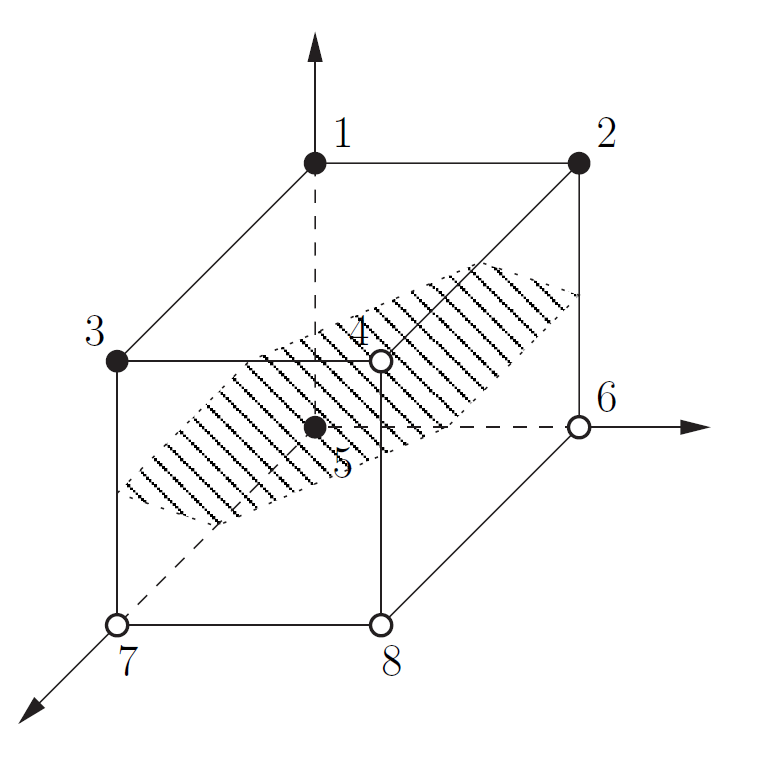
**b) Calcule, para temperatura *T* = 0*:*5, a distribuição de Boltzmann/Gibbs do vetor aleatório *X*. Verifique que esta distribuição de probabilidades define um vetor invariante para ambas as matrizes de transição calculadas no item (a).**

**c) Utilizando um pseudo-código, descreva um algoritmo de *Simulated Annealing* para minimizar esta função *J*(*x*1*; x*2). Defina e use quaisquer parâmetros (temperatura inicial, método de resfriamento etc.) que você julgar necessários.**

**e) Quando um número suficientemente grande de iterações do algoritmo do item (c) tiver sido calculado à temperatura *T* = 0*,*1, com que probabilidade teremos a ocorrência do evento *J* = 0*.*3?**

1. **(Opcional/Desafio) Prova de 2012 - Questão 3.**

**(*Simulated Annealing*) Considere uma situa¸c˜ao em que gostar´ıamos de dividir os oito v´ertices de um cubo unit´ario em dois agrupamentos, de modo que o erro quadr´atico total entre os centros dos agrupamentos e os membros dos agrupamentos seja minimizado. Considere que os v´ertices s˜ao numerados da seguinte forma:**

****

**e tamb´em que os agrupamentos s˜ao definidos atrav´es de um vetor x bin´ario de comprimento oito. A *i*-´esima componente de x ´e igual a zero se o v´ertice *i* pertencer ao agrupamento zero. E ´e igual a um no caso contr´ario. Por exemplo, o estado x = 11101000 corresponde ao caso da figura, em que os centr´oides s˜ao (3*/*4*,* 3*/*4*,* 1*/*4) (agrupamento 0) e (1*/*4*,* 1*/*4*,* 3*/*4) (agrupamento 1) e o custo *J* (x) ´e 4.50.**

* 1. **Baseando-se em um esquema de perturba¸c˜ao que consiste em sortear uma posi¸c˜ao do vetor x e invertˆe-la, escreva um algoritmo SA b´asico para a minimiza¸c˜ao do erro quadr´atico total. Defina quaisquer parˆametros que vocˆe julgar necess´arios.**
  2. **Assumindo *T* = 1*.*0 e o mesmo esquema de perturba¸c˜ao do item (a), calcule a probabilidade de transi¸c˜ao do estado 11101000 para o estado 11100000, e tamb´em a probabilidade de transi¸c˜ao do estado 11100000 para o estado 11110000.**
  3. **A contagem dos estados conforme os seus custos ´e dada pela tabela a seguir:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Custo** | **4.00** | **4.50** | **4.53** | **4.67** | **5.00** | **5.14** | **5.33** | **5.50** | **5.60** | **6.00** |
| **Nu´mero de Estados** | **6** | **8** | **48** | **24** | **24** | **16** | **24** | **24** | **64** | **18** |

**Calcule qual ´e a probabilidade de x = 00001111 ser gerado, quando o SA b´asico atinge *T* = 0*.*1.**

1. **Prova de 2016 - Questão 2.**

**(*Algoritmo de Metropolis*) Considere uma execu¸c˜ao do algoritmo de Metropolis `a temperatura fixa *T* = 1, com estados [*X*1*X*2] (*X*1 e *X*2 s˜ao vari´aveis aleat´orias bin´arias) e as duas matrizes de transi¸c˜ao dadas a seguir. A matriz M1, `a esquerda, modela as probabilidades de transi¸c˜ao entre estados no caso em que a perturba¸c˜ao, sempre diferente de zero, ´e feita sobre *X*1. A matriz M2, `a direita, ´e para o caso em que a perturba¸c˜ao, sempre diferente de zero, ´e feita sobre *X*2.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **M1** | **00** | **01** | **11** | **10** | **M2** | **00** | **01** | **11** | **10** |
| **00** | **2*/*3** | **0** | **0** | **1** | **00** | **2*/*3** | **1** | **0** | **0** |
| **01** | **0** | **2*/*3** | **1** | **0** | **01** | **1*/*3** | **0** | **0** | **0** |
| **11** | **0** | **1*/*3** | **0** | **0** | **11** | **0** | **0** | **0** | **1*/*3** |
| **10** | **1*/*3** | **0** | **0** | **0** | **10** | **0** | **0** | **1** | **2*/*3** |

* 1. **Considerando *J* (00) = 1, calcule os valores de *J* (01), *J* (11) e *J* (10) de forma que M1 e M2 tenham os valores dados acima.**
  2. **Calcule uma matriz de transi¸c˜ao M que modele transi¸c˜oes de qualquer um dos quatro estados para qualquer um dos quatro estados.**
  3. **Calcule o vetor invariante da matriz M do item (b). Verifique que ele ´e um vetor invariante tamb´em de M1 e M2, apesar de estas matrizes terem diferentes autovetores correspondentes aos autovalores que tˆem valor igual a 1.**

1. **Prova de 2016 - Questão 3.**

**(*Simulated Annealing*) Considere uma fun¸c˜ao custo dada pela tabela a seguir:**

*x* 1 2 3 4

*J* (*x*) 7 1 10 4

**a) Descreva, usando pseudo-c´odigo, a implementa¸c˜ao do algoritmo S.A. b´asico aplicado `a minimiza¸c˜ao da fun¸c˜ao custo acima. Na sua descri¸c˜ao, leve em considera¸c˜ao os seguintes parˆametros: temperatura inicial *T*0, temperatura m´ınima *Tmin*, e o nu´mero de itera¸c˜oes *N* a serem executadas em temperatura fixa.**

**b) Calcule as matrizes de transi¸c˜ao do processo de Markov que corresponde ao S.A. `a temperatura *T* = 10 e à temperatura *T* = 5 (chamadas de M10 e M5) e os seus respectivos vetores invariantes.**

**c) (0.25 ponto extra) Observe o menor dos nu´meros em M10 e o menor dos nu´meros em M5. Qual ´e a rela¸c˜ao entre estes nu´meros e *T* , *Jmax*, *Jmin* e o nu´mero de estados poss´ıveis?**

1. **Prova de 2017 - Questão 3, itens (b) e (c).**

**(*Simulated Annealing*) Considere uma função custo definida sobre cinco estados discretos, chamados de estados 1*,* 2*, ...* 5, com os seguintes valores: *J*(1) = *J*(5) = 4, *J*(2) = 1, *J*(3) = 3 e *J*(4) = 2.**

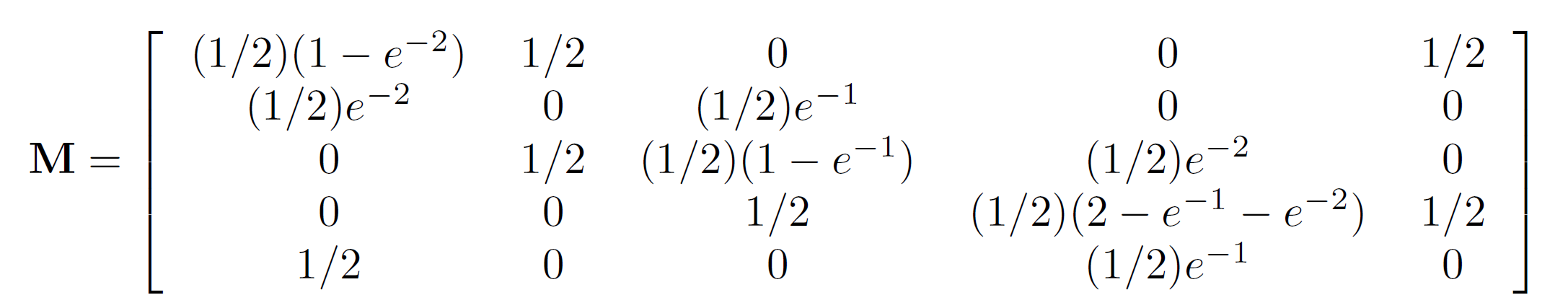
**a) Apresente, usando pseudo-código, uma implementação do algoritmo Simulated Annealing básico, usada para encontrar o estado para o qual o valor da função custo \_e mínimo. Defina e utilize todos os parâmetros que você considerar necessários.**

**b) Calcule uma matriz de transição entre estados à temperatura *T*1 = 1/ln2 e uma matriz de transição entre estados à temperatura *T*2 = 1*/* ln 3.**

**c) Calcule os vetores invariantes das matrizes encontradas no item (b).**

1. **Prova de 2018 - Questão 3.**

**(*Simulated Annealing*) Considere a matriz de transi¸c˜ao dada a seguir (calculada usando *T* = *T*0 = 0*.*1):**

****

1. **Considerando um grafo com cinco estados (estado 1 conectado aos estados 2 e 5; 2 conectado a 1 e 3; 3 conectado a 2 e 4; 4 conectado a 3 e 5; 5 conectado a 4 e 1) e considerando que o custo associado ao estado 1 ´e igual a 0.2, calcule os custos associados aos estados 2, 3, 4 e 5.**
2. **Calcule o vetor invariante da matriz M.**
3. **Escreva a menor das probabilidades da matriz M em fun¸c˜ao dos valores m´aximo e m´ınimo de *J* (*x*), da temperatura *T* e do nu´mero *N* das transi¸c˜oes poss´ıveis para cada estado. Recalcule esta probabilidade para *T* = *T*0*/* log2 4.**