Exercício 02

Renan Salles de Freitas
CPE 723 - Otimização Natural

17 de março de 2018

Exercício 1.a. Temos que a distribuição de probabilidade de $X(1)$ é:

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

E ainda:

$$\mathbf{p}_2 = M\mathbf{p}_1 = M^2\mathbf{p}_0$$
$$\mathbf{p}_n = M^n\mathbf{p}_0$$

Portanto:

$$\mathbf{p}_3 = M^3\mathbf{p}_0$$
$$\mathbf{p}_3 = \begin{bmatrix} 0.3328 \\ 0.3344 \\ 0.3328 \end{bmatrix}$$

Exercício 1.b. Supondo que estamos no estado $X(t)$, construímos uma lista com os possíveis próximos estados, considerando a distribuição de probabilidade, conforme a matriz de transição de estados M: $X(t+1) = [X(t) \quad 0 \quad 1 \quad 2]$. Sorteamos um índice de zero a quatro com o MatLab e atualizamos $X(t+1)$. Observe que, dessa forma, a transsição para o estado o estado atual sempre possui probabilidade 0.5 e os outros estados possuem probabilidade 0.25.

$$\begin{aligned} X(0) &= 1 \\ \text{list} &= [0 \quad 1 \quad 2 \quad 1] \\ r &= \text{randi}(4) = 3 \\ X(1) &= \text{list}(r) = 2 \end{aligned} \tag{1}$$

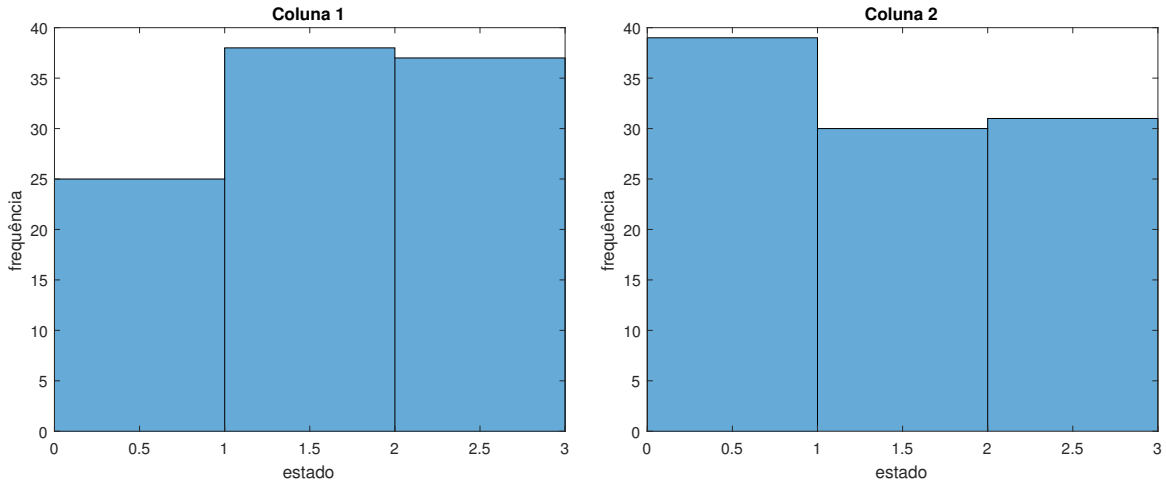
$$\begin{aligned} X(1) &= 2 \\ \text{list} &= [0 \quad 1 \quad 2 \quad 2] \\ r &= \text{randi}(4) = 3 \\ X(2) &= \text{list}(r) = 2 \end{aligned} \tag{2}$$

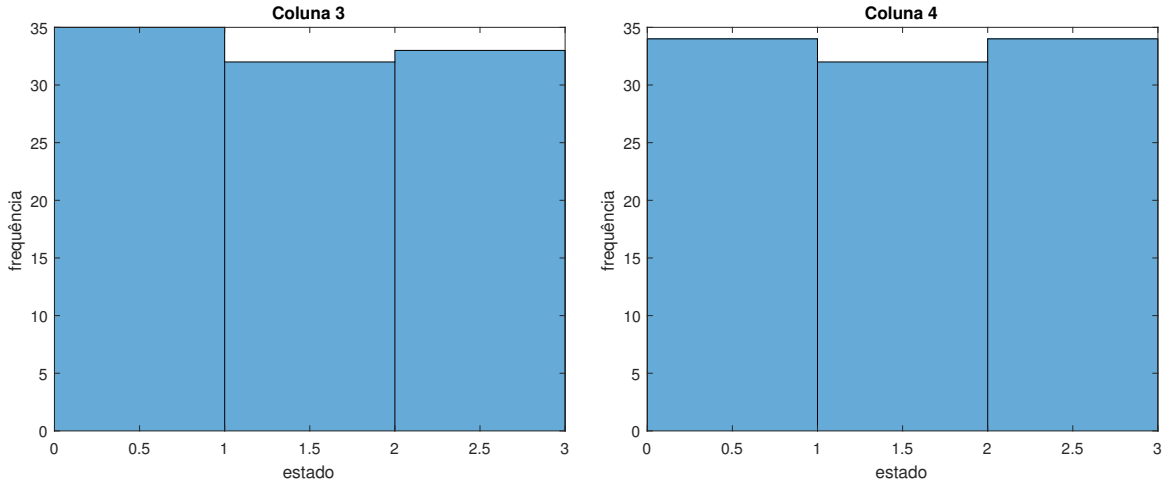
$$
\begin{aligned}
X(2) &= 2 \\
\text{list} &= [0 \quad 1 \quad 2 \quad 2] \\
r &= \text{randi}(4) = 4 \\
X(3) &= \text{list}(r) = 2
\end{aligned} \tag{3}
$$

Exercício 1.c. Código MatLab abaixo:

```
clear all
clc

init = [0 1 2];
n = 100;
x = zeros(n,4);
for i = 1:n
    x(i,1) = init(randi(3));
    for j = 2:4
        list = [0 1 2 x(i,j-1)];
        r = randi(4);
        x(i,j) = list(r);
    end
end
```

Exercício 1.d. Os histogramas estão representados abaixo:





O código MatLab para calcular as probabilidades está abaixo:

```
M = [ 0.5  0.25  0.25;
      0.25  0.5  0.25;
      0.25  0.25  0.5];

p0 = [sum(x(:,1)==0)/100  sum(x(:,1)==1)/100  sum(x(:,1)==2)/100]';
p1 = [sum(x(:,2)==0)/100  sum(x(:,2)==1)/100  sum(x(:,2)==2)/100]';
p2 = [sum(x(:,3)==0)/100  sum(x(:,3)==1)/100  sum(x(:,3)==2)/100]';
p3 = [sum(x(:,4)==0)/100  sum(x(:,4)==1)/100  sum(x(:,4)==2)/100]';

p0g = [1/3  1/3  1/3]';
p1g = M*p0g;
p2g = M*p1g;
p3g = M*p2g;
```

Sabemos que o estado inicial é equiprovável para os três estados :

$$\mathbf{p}_0 = [0.3333 \quad 0.3333 \quad 0.3333]$$

E ainda:

$$\mathbf{p}_1 = M\mathbf{p}_0 = [0.3333 \quad 0.3333 \quad 0.3333]$$
$$\mathbf{p}_n = M^n\mathbf{p}_0 = \mathbf{p}_0 = [0.3333 \quad 0.3333 \quad 0.3333]$$

Calculamos as probabilidades pela frequência do histograma e obtemos:

$$\mathbf{p}_0 = [0.25 \quad 0.38 \quad 0.37]$$
$$\mathbf{p}_1 = [0.39 \quad 0.30 \quad 0.31]$$
$$\mathbf{p}_2 = [0.35 \quad 0.32 \quad 0.33]$$
$$\mathbf{p}_3 = [0.34 \quad 0.32 \quad 0.34]$$

Vale observar que, conforme aumentamos o número de iterações, o estado se aproxima para o estado estacionário $\mathbf{p}_n = [0.333 \quad 0.333 \quad 0.333]$, autovetor da matriz M.

Exercício 2.a.

$$M = \begin{bmatrix} 0 & \frac{1}{4}e^{-3} & \frac{1}{4}e^{-2} & \frac{1}{4}e^{-4} & \frac{1}{4}e^{-1} \\ \frac{1}{4} & \frac{1}{4}(3 - e^{-1} - e^{-2} - e^{-3}) & \frac{1}{4} & \frac{1}{4}e^{-1} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4}e^{-1} & \frac{1}{4}(2 - e^{-1} - e^{-2}) & \frac{1}{4}e^{-2} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4}(4 - e^{-1} - e^{-2} - e^{-3} - e^{-4}) & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4}e^{-2} & \frac{1}{4}e^{-1} & \frac{1}{4}e^{-3} & \frac{1}{4}(1 - e^{-1}) \end{bmatrix}$$

O código MatLab para calcular as probabilidades está abaixo:

```
% Exerc[icio 2a
k = (1/4);
ex1 = exp(-1);
ex2 = exp(-2);
ex3 = exp(-3);
ex4 = exp(-4);

M = [0 k*ex3 k*ex2 k*ex4 k*ex1;
     k k*(3-ex1-ex2-ex3) k k*ex1 k;
     k k*ex1 k*(2-ex1-ex2) k*ex2 k;
     k k k k*(4-ex1-ex2-ex3-ex4) k;
     k k*ex2 k*ex1 k*ex3 k*(1-ex1)];

%{
M =

         0    0.0124    0.0338    0.0046    0.0920
    0.2500    0.6117    0.2500    0.0920    0.2500
    0.2500    0.0920    0.3742    0.0338    0.2500
    0.2500    0.2500    0.2500    0.8572    0.2500
    0.2500    0.0338    0.0920    0.0124    0.1580

%}
```

Exercício 2.b. Coonstruímos uma lista com os possíveis estados: list = [1 2 3 4 5]. Utilizamos o MatLab para fornecer um número inteiro randômico (distribuição uniforme) entre 1 e 4: randi(4). Removemos o estadoo atual da lista: list = [2 3 4 5]. O número randômico será o índice do elemento que utilizaremos como próximo candidato para estado $X(1)$:

$$\begin{aligned} \text{randi}(4) &= 4 \\ \text{list} &= [2 \quad 3 \quad 4 \quad 5] \\ X(1) &= \text{list}(4) = 5 \\ J(5) &< J(1) \\ X(1) &= 5 \end{aligned} \tag{4}$$

$$\begin{aligned}
\text{randi}(4) &= 4 \\
\text{list} &= [1 \quad 2 \quad 3 \quad 4] \\
X(2) &= \text{list}(4) = 4 \\
J(4) &< J(5) \\
X(2) &= 4
\end{aligned} \tag{5}$$

$$\begin{aligned}
\text{randi}(4) &= 3 \\
\text{list} &= [1 \quad 2 \quad 3 \quad 5] \\
X(3) &= \text{list}(3) = 3 \\
\text{rand} &= 0.0975 > \frac{1}{4}e^{-0.2/0.1} = 0.0338 \\
X(3) &= 4
\end{aligned} \tag{6}$$

$$\begin{aligned}
\text{randi}(4) &= 2 \\
\text{list} &= [1 \quad 2 \quad 3 \quad 5] \\
X(4) &= \text{list}(2) = 2 \\
\text{rand} &= 0.5469 > \frac{1}{4}e^{-0.1/0.1} = 0.092 \\
X(4) &= 4
\end{aligned} \tag{7}$$

Exercício 2.c. Fazemos $[v, a] = \text{eig}(M)$. O vetor invariante é autovetor correspondente ao autovalor 1, dividido pela soma de seus elemento:

$$v = \begin{bmatrix} 0.0117 \\ 0.2341 \\ 0.0861 \\ 0.6364 \\ 0.0317 \end{bmatrix}$$

Exercício 2.d. Calculando os fatores de Boltzmann, obtemoos:

$$\begin{aligned}
B &= \begin{bmatrix} e^{-5} & e^{-2} & e^{-3} & e^{-1} & e^{-4} \end{bmatrix} \\
B &= \begin{bmatrix} 0.0067 & 0.1353 & 0.0498 & 0.3679 & 0.0183 \end{bmatrix}
\end{aligned}$$

Dividindo o vetor por sua soma, obtemos:

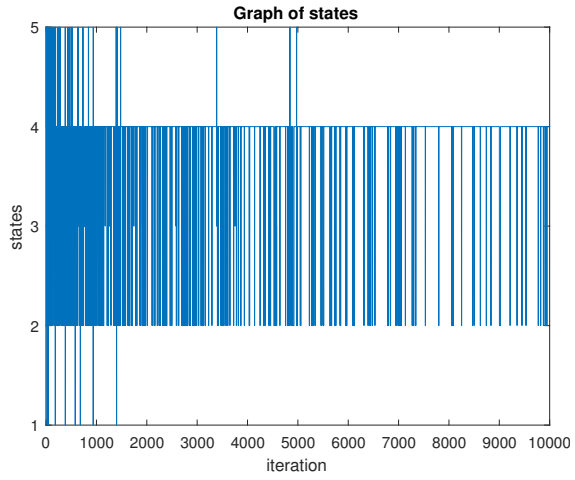
$$B = \begin{bmatrix} 0.0117 & 0.2341 & 0.0861 & 0.6364 & 0.0317 \end{bmatrix}$$

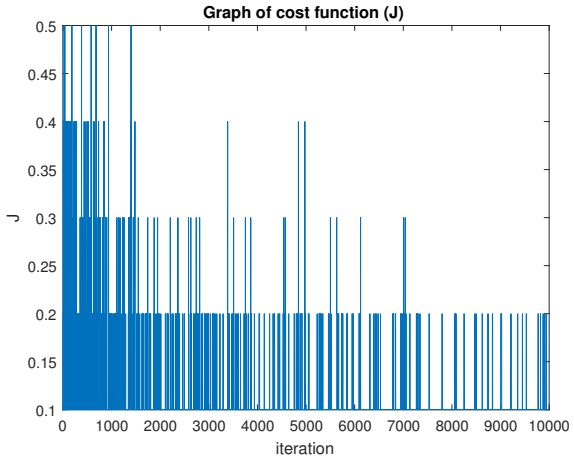
Que é o mesmo vetor invariante.

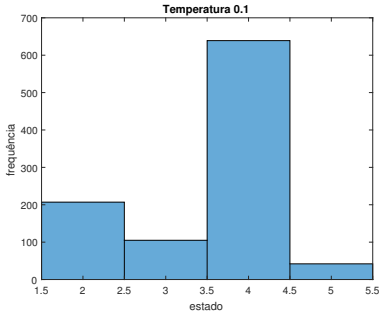
Exercício 2.e. O código MatLab para SA está abaixo:

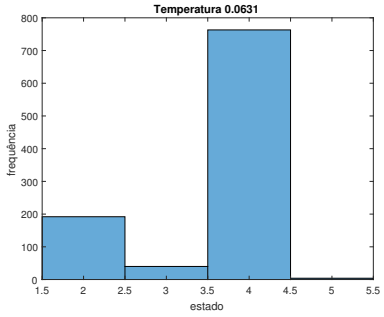
```
% Exercicio 2e

clear all
clc

x0 = 1;
number_of_iteration = 1000;
number_of_temperature_iteration = 10;
T = [0.1 0.0631 0.05 0.0431 0.0387 0.0356 0.0333 0.0315 0.0301
   0.0289];

x = zeros(number_of_iteration * number_of_temperature_iteration, 1);
x(1) = x0;

costs = [0.5 0.2 0.3 0.1 0.4];
J = @(n) costs(n);

B = @(n,t) exp(-n/t);

counter = 1;
counter_temperatures = 1;
Jmin = J(x(counter));
xmin = x(counter);
while counter_temperatures <= number_of_temperature_iteration
    while counter < number_of_iteration * counter_temperatures
        states = 1:5;
        states(x(counter)) = [];
        r = randi(4);
        xk = states(r);
        Jxk = J(xk);
        dJ = Jxk - J(x(counter));
        counter = counter + 1;
        if dJ < 0
            x(counter) = xk;
        else
            a = rand;
            if B(dJ,T(counter_temperatures)) > a
                x(counter) = xk;
            else
                x(counter) = x(counter - 1);
            end
        end
        if Jxk < Jmin
            Jmin = Jxk;
            xmin = xk;
        end
```

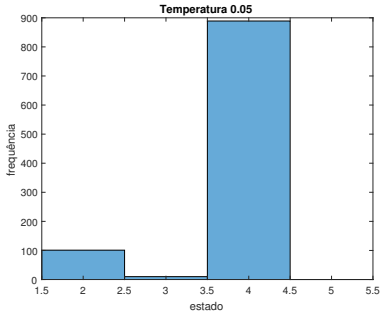
```
    end
    counter_temperatures = counter_temperatures + 1;
end

histogram(x(1:1000),[1 2 3 4 5])
```

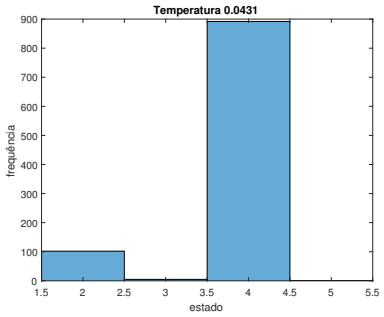


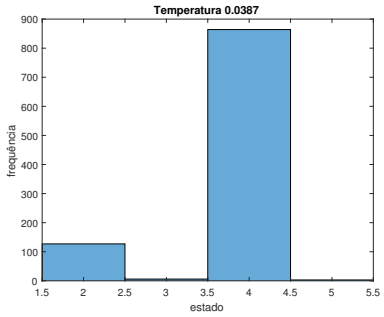


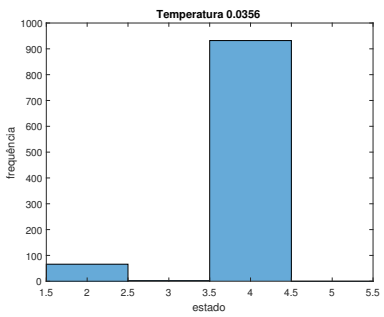


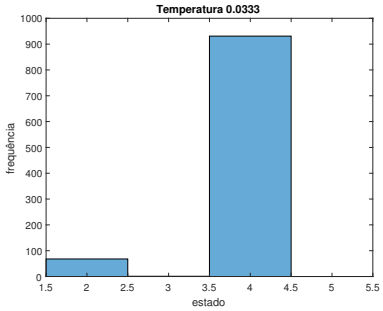


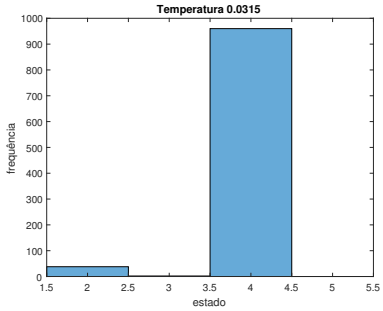


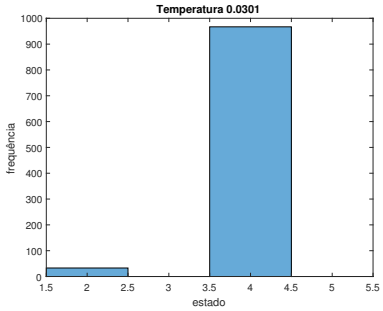


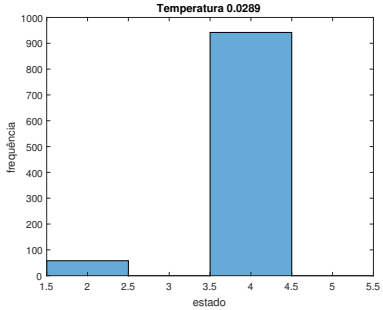












As probabilidades foram calculadas e estão abaixo:

$$
\begin{aligned}
v[1] &= \begin{bmatrix} 0.007 & 0.207 & 0.105 & 0.639 & 0.042 \end{bmatrix} \\
v[2] &= \begin{bmatrix} 0.001 & 0.192 & 0.04 & 0.763 & 0.04 \end{bmatrix} \\
v[3] &= \begin{bmatrix} 0 & 0.101 & 0.01 & 0.889 & 0 \end{bmatrix} \\
v[4] &= \begin{bmatrix} 0 & 0.102 & 0.005 & 0.892 & 0.001 \end{bmatrix} \\
v[5] &= \begin{bmatrix} 0 & 0.127 & 0.006 & 0.864 & 0.003 \end{bmatrix} \\
v[6] &= \begin{bmatrix} 0 & 0.066 & 0.002 & 0.932 & 0 \end{bmatrix} \\
v[7] &= \begin{bmatrix} 0 & 0.068 & 0.001 & 0.931 & 0 \end{bmatrix} \\
v[8] &= \begin{bmatrix} 0 & 0.038 & 0.002 & 0.96 & 0 \end{bmatrix} \\
v[9] &= \begin{bmatrix} 0 & 0.033 & 0 & 0.967 & 0 \end{bmatrix} \\
v[10] &= \begin{bmatrix} 0 & 0.058 & 0 & 0.942 & 0 \end{bmatrix}
\end{aligned}
$$

Inicialmente, a distribuição $v[1]$ se assemelha à probabilidade estacionária (vetor invariante). Porém, conforme resfriamos o sistema (diminuímos a temperatura), o estado converge para o mínimo global.

Exercício 3. Para este exercício vamos utilizar a função Rosenbrock para N = 10:

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$$

O mínimo está em $\mathrm{x_{min}} = \mathbf{1}$ e $\mathrm{J_{min}} = 0$. O código MatLab para SA está abaixo:

```
% Exercicio 3

clear all
clc

number_of_variables = 10;
number_of_iteration = 5000;
number_of_temperature_iteration = 30;
epsilon = 0.05;
T0 = 1;

T = T0;
x0 = zeros(1, number_of_variables);
x = zeros(number_of_iteration * number_of_temperature_iteration, 10);
x(1,:) = x0;

B = @(n,t) exp(-n/t);

counter = 1;
counter_temperatures = 1;
Jmin = J(x(counter,:));
xmin = x(counter,:);
while counter_temperatures <= number_of_temperature_iteration
    while counter < number_of_iteration * counter_temperatures
        r = randn(1,number_of_variables);
        xk = x(counter,:) + epsilon * r;
        Jxk = J(xk);
        dJ = Jxk - J(x(counter,:));
        counter = counter + 1;
        if dJ < 0
            x(counter,:) = xk;
        else
            a = rand;
            if B(dJ,T) > a
                x(counter,:) = xk;
            else
                x(counter,:) = x(counter - 1,:);
            end
        end
        if Jxk < Jmin
            Jmin = Jxk;
            xmin = xk;
        end
    end
    counter_temperatures = counter_temperatures + 1;
    T = T0 / log2(counter_temperatures + 1);
end
```

A função custo:

```
function y = J(x)
n = size(x);
y = zeros(n(1),1);
for i=1:n(2)-1
    y = y + 100*(x(:,i+1) - x(:,i).^2).^2 + (1 - x(:,i)).^2;
end
```

Encontramos:

$$x_{min} = \begin{bmatrix} 0.9887 \\ 0.9877 \\ 0.9884 \\ 0.9997 \\ 1.0082 \\ 1.0121 \\ 1.0005 \\ 0.9981 \\ 1.0034 \\ 1.0511 \end{bmatrix}$$

e $J_{min} = 0.3474$.