CPE-723 – Otimização Natural

Lista de Exercícios #3

Amanda Isabela de Campos (DRE 120074842)

01) Prova de 2012 - Questão 4;
(*Deterministic Annealing*) Considere um problema de *soft clustering* com cinco vetores de dados x = ($x1$, $x2$) e três centroides iniciais y = (y_1, y_2), definidos segundo a tabela a seguir. Considere também que a distância entre dois vetores é quadrática, ou seja, $d(x, y) = (x_1 - y_1)^2 + (x_2 - y_2)^2$.

$\mathbf{x}_1$	5	4
$\mathbf{x}_2$	4	5
$\mathbf{x}_3$	5	5
$\mathbf{x}_4$	−5	−4
$\mathbf{x}_5$	−4	−5
$\mathbf{y}_1$	0	0
$\mathbf{y}_2$	1	1
$\mathbf{y}_3$	−1	−1

a) Calcule a matriz de probabilidades $p(y|x)$ que minimiza $J = D - TH$ com $T = 10$. Calcule também os valores dos centróides atualizados segundo esta matriz.

```
import numpy as np
X = np.array([[5,4,5,-5,-4],[4,5,5,-4,-5]])
M,N=np.shape(X)
K=2 #número de clusters
Y=np.array([[0,1,-1],[0,1,-1]])
d=np.zeros([K,N])
T = 10
p_ygivenx=np.zeros([K,N])

# Condição da Partição
for n in range(0,N):
    for k in range(0,K):
        d[k,n]=np.sum(np.power(X[:,n]-Y[:,k],2))
        p_ygivenx[k,n]=np.exp(-d[k,n]/T)
Zx=np.sum(p_ygivenx,axis=0)
p_ygivenx=p_ygivenx/np.tile(Zx,(K,1))

# Condição do centróide
Y=np.zeros([M,K])
for k in range(0,K):
    y=np.zeros(M)
    w=0
    for n in range(0,N):
        y+=p_ygivenx[k,n]*X[:,n]
        w+=p_ygivenx[k,n]
    Y[:,k]=y/w
```

$p_{y|x}$ = 0.164248 0.164248 0.139656 0.164248 0.164248

0.813524 0.813524 0.84487 0.0222285 0.0222285
0.0222285 0.0222285 0.0154743 0.813524 0.813524

Y = 0.876524 4.50887 -4.17568
0.876524 4.50887 -4.17568

b) Repita o item (a) para $T = 0.1$ e comente sobre qual é a diferença.

$p_{y|x}$ = 3.25749e-70 3.25749e-70 6.71418e-79 3.25749e-70 3.25749e-70

1 1 1 4.50803e-157 4.50803e-157

4.50803e-157 4.50803e-157 1.91517e-174 1 1

Y = 2.57644e-09 4.66667 -4.5

2.57644e-09 4.66667 -4.5

Neste caso, para uma temperatura menor a matriz de probabilidades $p_{y|x}$ tem em cada linha praticamente todos os valores iguais a zero e apenas um valor igual a um, o que indica um probabilidade de acerto igual a 100%, ou seja o processo convergiu.

02) Prova de 2017 - Questão 4;

4. (*Deterministic Annealing*) Considere um conjunto de dados $\mathbf{X}$ composto por quatro vetores equiprováveis e com coordenadas $(0,0)$, $(0,2)$, $(2,2)$ e $(2,0)$. Considere uma partição suave do $\mathbb{R}^2$ realizada através de dois centróides $\mathbf{y}_1$ e $\mathbf{y}_2$. As distâncias quadráticas entre cada vetor $\mathbf{x}$ e cada vetor $\mathbf{y}$ são dadas a seguir:

	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$
$\mathbf{y}_1$	1	5	5	1
$\mathbf{y}_2$	5	1	1	5

a) Considerando $T = 5$, calcule a matriz de probabilidades condicionais $p_{\mathbf{Y}|\mathbf{X}}$ que minimiza $J = D - TH = \sum_x p_{\mathbf{X}}(x) \sum_y p_{\mathbf{Y}|\mathbf{X}}(y|x) + T \sum_x p_{\mathbf{X}}(x) \sum_y p_{\mathbf{Y}|\mathbf{X}}(y|x) \log p_{\mathbf{Y}|\mathbf{X}}(y|x)$.

b) Também considerando $T = 5$, calcule vetores $\mathbf{y}_1$ e $\mathbf{y}_2$ atualizados. Compare o erro médio quadrático D_2, associado aos vetores $\mathbf{y}_1$ e $\mathbf{y}_2$ atualizados, com o erro médio quadrático D_1, associado aos vetores $\mathbf{y}_1$ e $\mathbf{y}_2$ anteriores à atualização.

a) $p_{y|x}$ = 0.689974 0.310026 0.310026 0.689974
0.310026 0.689974 0.689974 0.310026

Calculos obtidos com o seguinte código:

```
X = np.array([[0,0,2,2],[0,2,2,0]])
M,N=np.shape(X)
K=2 #número de clusters
d=np.zeros([K,N])
T = 5
p_ygivenx=np.zeros([K,N])
d = np.array([[1,5,5,1],[5,1,1,5]])
# Partition Condition
for n in range(0,N):
    for k in range(0,K):
        # d[k,n]=np.sum(np.power(X[:,n]-Y[:,k],2))
        p_ygivenx[k,n]=np.exp(-d[k,n]/T)
Zx=np.sum(p_ygivenx,axis=0)
```

```
p_ygivenx=p_ygivenx/np.tile(Zx,(K,1))

J=-T/N*np.sum(np.log(Zx))
D=np.mean(np.sum(p_ygivenx*d,axis=0))
```

b) Y = 1 1
0.620051 1.37995

O erro médio quadrático D2 é 2.240102 (atualizado)

Com as coordenadas dos pontos e as distâncias quadráticas é possível calcular as coordenadas dos dois centróides y_1 e y_2 : y_1 (1,0) e y_2 (1,-2), portanto pode-se calcular o erro médio quadrático.

O erro médio quadrático D1 é 4.76766 (anterior). O erro médio quadrático diminuiu depois da atualização, isso porque esse processo de atualização é um passo do algoritmo de Deterministic Annealing em busca do clusters com mínimo erro médio quadrático.

O código adotado está reproduzido a seguir.

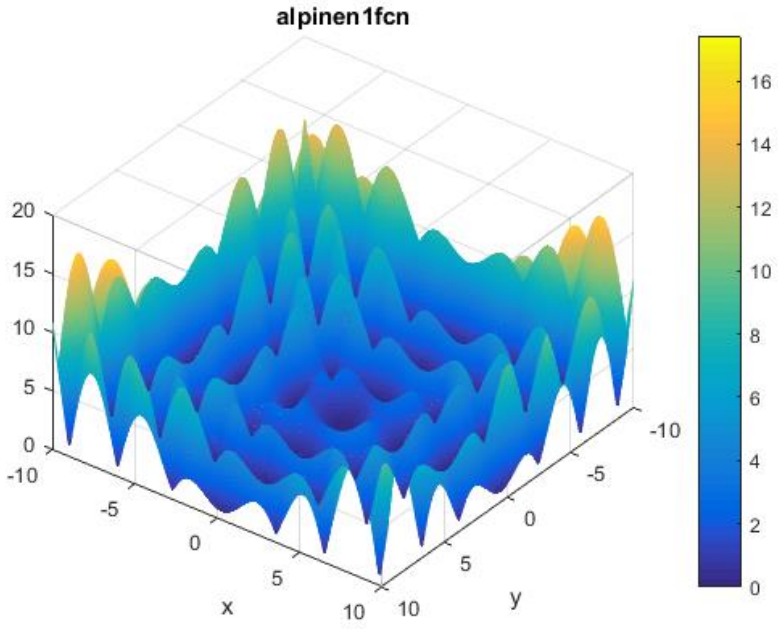
```
X = np.array([[0,0,2,2],[0,2,2,0]])
M,N=np.shape(X)
K = 2 #número de clusters
Y=np.array([[0,1],[1,-2]])
d=np.zeros([K,N])
T = 10
p_ygivenx=np.zeros([K,N])

# Partition Condition
for n in range(0,N):
    for k in range(0,K):
        d[k,n]=np.sum(np.power(X[:,n]-Y[:,k],2))
        p_ygivenx[k,n]=np.exp(-d[k,n]/T)
Zx=np.sum(p_ygivenx,axis=0)
p_ygivenx=p_ygivenx/np.tile(Zx,(K,1))

J=-T/N*np.sum(np.log(Zx))
D=np.mean(np.sum(p_ygivenx*d,axis=0))
```

03) Proponha uma função J (x), sendo x um vetor com 20 dimensões, cujo ponto mínimo você conheça. Evite propor funções que tenham um só ponto mínimo. Encontre o ponto mínimo global utilizando S.A. Entregue o código utilizado e alguns comentários sobre o resultado obtido;

A função adotada como exemplo é a Alpine N. 1 Function definida no espaço n-dimensional. A função tem mínimo global $f(x^*) = 0$ localizado em $x^*=(0,\ldots,0)$ e segue a expressão:

$$f(\mathbf{x}) = f(x_1, \ldots, x_n) = \sum_{i=1}^{n} |x_i sin(x_i) + 0.1x_i|$$

O gráfico da função Alpine em duas dimensões está indicado a seguir:



Referencias:

http://benchmarkfcns.xyz/benchmarkfcns/alpinen1fcn.html

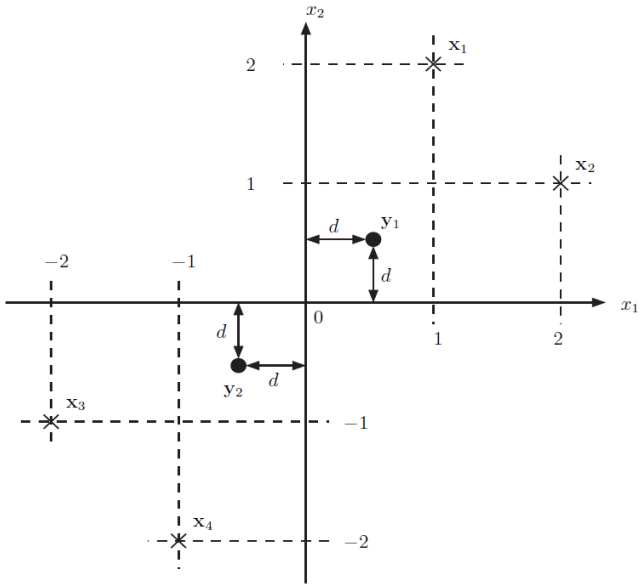
O código implementado para a busca do mínimo global desta função com o Simulated Annealing e o Fast Simulated Annealing está reproduzido a seguir. Vale observar que em ambos os casos o mínimo global foi encontrado, porem com o FSA a convergência foi mais rápida.

```
import numpy as np
import math

numero_variaveis = 20
b = 10.0  ## Limite superior
a = -10.0  ## Limite inferior    limites = [a,b)
def Custo(X):
    numero_variaveis = 20
    d = numero_variaveis
    sum = 0;
    for ii in range(0,d):
        xi = X[ii]
        sum = sum + (np.abs(xi * math.sin(xi) + 0.1 * xi))
    y = sum

    return y

#-------------------------------------------------------------------------
# Simulated Annealing
X0 = (b-a)*np.random.rand(numero_variaveis) + a

N=int(1e5);  K=100; T0=5e-1; e=1e-1
X = X0
Xmin = X0
np.random.seed(0);
fim=0; n=0; k=0; Jmin=Custo(X); Xmin=X; T=T0;
```

```
while not(fim):
    T = T0/(1+k)
    # T = T0/np.log2(2+k)
    for n in range(N):
        X_hat = X + e*(np.random.standard_cauchy(np.shape(X)))
        # X_hat = X + e*(np.random.normal(0,1,np.shape(X)))

        if np.random.uniform()<np.exp((Custo(X)-Custo(X_hat))/T):
            X = X_hat
            if Custo(X) < Jmin:
                Jmin = Custo(X)
                Xmin = X

    print([k,Jmin])
    k=k+1
    if k == K: fim =1
print(Jmin)
```

04) Questão extra (opcional): Prova de 2009 - Questão 4.

4. (*Deterministic Annealing*) A figura a seguir representa quatro vetores reais, $\mathbf{x}_1$ a $\mathbf{x}_4$, marcados sobre o plano $\mathbb{R}^2$. Estes quatro vetores serão agrupados em duas "classes", conforme a posição dos centróides de cada classe: $\mathbf{y}_1$ e $\mathbf{y}_2$. A distância quadrática entre dois vetores é definida como $d_{\mathbf{xy}} = (x_1 - y_1)^2 + (x_2 - y_2)^2$ e os vetores $\mathbf{x}_i$ são atribuídos às classes $\mathbf{y}_j$ conforme as probabilidades condicionais $p(\mathbf{y}_j|\mathbf{x}_i) = \exp(-d_{\mathbf{x}_i\mathbf{y}_j}/T)$, onde $i = 1, 2, 3, 4$ e $j = 1, 2$. O parâmetro T ("temperatura") controla a incerteza com a qual a partição é feita. O objetivo deste problema é caracterizar a dependência entre a posição relativa (d) dos centróides e a temperatura T.



a) Usando as posições iniciais de $\mathbf{y}_1$ e $\mathbf{y}_2$ indicadas na figura, complete a matriz de probabilidades condicionais a seguir. Neste item, considere $T = 5$ e $d = 1$.

$p(\mathbf{y}\|\mathbf{x})$	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$
$\mathbf{y}_1$				
$\mathbf{y}_2$				

Usando a matriz acima, calcule os novos vetores $\mathbf{y}_1$ e $\mathbf{y}_2$, atualizados para a iteração seguinte.

b) As probabilidades do item (a) e as novas posições dos centróides podem ser escritas como funções de d e T. Escreva a expressão atualizada de $\mathbf{y}_1$ em função de $d(k)$ e T genéricos. Mostre então que $d(k+1) = f(d(k), T)$, descrevendo a função $f(.)$.

c) (Item opcional) Encontre a faixa de temperaturas de transição (T_1,T_2), acima das quais tem-se $d \simeq 0$ e abaixo das quais tem-se $d \simeq 1.5$.

a)

$$d_{xy} = \begin{bmatrix} d^2 - 2d + 1 & d^2 + 2d + 1 \\ d^2 + 2d + 1 & d^2 - 2d + 1 \end{bmatrix}$$

Partição:

$$p_{y|x} = \begin{bmatrix} \frac{e^{\frac{-d^2+2d-1}{T}}}{e^{\frac{-d^2+2d-1}{T}} + e^{\frac{-d^2-2d-1}{T}}} & \frac{e^{\frac{-d^2-2d-1}{T}}}{e^{\frac{-d^2+2d-1}{T}} + e^{\frac{-d^2-2d-1}{T}}} \\ \frac{e^{\frac{-d^2-2d-1}{T}}}{e^{\frac{-d^2+2d-1}{T}} + e^{\frac{-d^2-2d-1}{T}}} & \frac{e^{\frac{-d^2+2d-1}{T}}}{e^{\frac{-d^2+2d-1}{T}} + e^{\frac{-d^2-2d-1}{T}}} \end{bmatrix}$$

Substituindo $e^{\frac{2d}{T}}$ por α em $p_{x|y}$, temos:

$$p_{y|x} = \begin{bmatrix} \frac{\alpha}{\alpha+\alpha^{-1}} & \frac{\alpha^{-1}}{\alpha+\alpha^{-1}} \\ \frac{\alpha^{-1}}{\alpha+\alpha^{-1}} & \frac{\alpha}{\alpha+\alpha^{-1}} \end{bmatrix}$$

b) Condição da partição: novo y_1 (sendo que $y_2 = -y_1$)

$$y_1 = \frac{1}{1}\left(\frac{\alpha}{\alpha+\alpha^{-1}} \times 1 + \frac{\alpha^{-1}}{\alpha+\alpha^{-1}} \times 2\right)$$

$$y_1 = \frac{\alpha - \alpha^{-1}}{\alpha + \alpha^{-1}} = \frac{e^{2d/T} - e^{-2d/T}}{e^{2d/T} + e^{-2d/T}} = tanh(\frac{2d}{T})$$

c) Atualizando d(n), temos: d(n+1) = tanh(2d(n)/T). Numericamente:

$$\lim_{n\to\infty} d(n) = \begin{cases} 1, se T < 0.5 \\ 0, se T > 2 \\ \epsilon\,[0,1]\,, se T \epsilon\,[0.5,2] \end{cases}$$