1 Código utilizado na tarefa

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neighbors import NearestNeighbors
import operator

def ReadData(file):
    serie = np.genfromtxt(file, delimiter=',', dtype='str')
    X = serie[1:,1]
    X = X.astype(float)
    return X;
#-----------------------------------------------#

def computarN(X, threshold):
    bubbles_c = np.array([])
    bubbles_n = np.array([])
    firstBubble=0;
    firstIteration=True;

    for index in range(0, len(X)):

        if index == 0:
            bubbles_c = np.concatenate((bubbles_c,X[index:index+1]),axis=0)
            bubbles_n = np.concatenate((bubbles_n,[1]),axis=0)
            firstBubble = 0;

        else:
            closerBubble = abs(bubbles_c - X[index]);
            min_index, min_value = min(enumerate(closerBubble), key=operator.itemgetter(1));
            if (min_value <= threshold):
                if min_index == firstBubble  and (not(firstIteration)):
                    break
                else:
                    bubbles_c[min_index] = ((bubbles_c[min_index]*bubbles_n[min_index]) +  X[index])/(bubbles_n[min_index]+1)
                    bubbles_n[min_index] = bubbles_n[min_index] + 1

            else:
                bubbles_c = np.concatenate((bubbles_c, X[index:index + 1]), axis=0)
                bubbles_n = np.concatenate((bubbles_n, [1]), axis=0)
                firstIteration = False
    bubbles_c2 = np.array([])
    bubbles_n2 = np.array([])
    firstBubble=0;
    firstIteration=True;
```

```
    for iter in range(index, len(X)):

        if iter == index:
            bubbles_c2 = np.concatenate((bubbles_c2,X[iter:iter+1]),axis=0)
            bubbles_n2 = np.concatenate((bubbles_n2,[1]),axis=0)
            firstBubble = 0;
        else:
            closerBubble = abs(bubbles_c2 - X[iter]);
            min_index, min_value = min(enumerate(closerBubble), key=operator.itemgetter(1));
            if (min_value <= threshold):
                if min_index == firstBubble  and (not(firstIteration)):
                    break
                else:
                    bubbles_c2[min_index] = ((bubbles_c2[min_index]*bubbles_n2[min_index]) +  X[index])/(bubbles_n2[min_index]+1)
                    bubbles_n2[min_index] = bubbles_n2[min_index] + 1

            else:
                bubbles_c2 = np.concatenate((bubbles_c2, X[iter:iter + 1]), axis=0)
                bubbles_n2 = np.concatenate((bubbles_n2, [1]), axis=0)
                firstIteration = False

    return sum(bubbles_n2)

def acharAnomalia(X, threshold):
    N = computarN(X,threshold)
    N = int(N)
    print "N: " + str(N)

    p = []
    #computa a media e variancia para o primeiro intervalo
    p.append([np.mean(X[0:N]), np.std(X[0:N])])
    i = N
    #computa a media e variancia para o cada intervalo e concatena isso em p
    while(i <= len(X) - N):
        p.append([np.mean(X[i:i + N]), np.std(X[i:i + N])])
        i += N



    neighbours = NearestNeighbors(n_neighbors=2, algorithm='ball_tree').fit(p)
    dist, ind = neighbours.kneighbors(p)

    mediaDist = np.mean(dist[:,1:])
    desvioDist = np.std(dist[:,1:])

    for k in range(len(dist)):
```

```
        inicio = k*N
        if(inicio + N > len(X)):
            fim = len(X)
        else:
            fim = inicio + N - 1
        ran = list(range(inicio, fim))
        if(dist[k][1] > (mediaDist - desvioDist) and dist[k][1] < (mediaDist + desvioDist)):
            plt.plot(ran, X[inicio:fim],color=(0,0,1))
        else:
            plt.plot(ran, X[inicio:fim],color=(1,0,0))
    plt.show()

#encontra as anomalias
t1 = 15.0
s1 = ReadData('serie1.csv')
acharAnomalia(s1, t1)

s2 = ReadData('serie2.csv')
acharAnomalia(s2, t1)

s3 = ReadData('serie3.csv')
acharAnomalia(s3, t1)

s4 = ReadData('serie4.csv')
acharAnomalia(s4, t1)
t2 = 1.5
s5 = ReadData('serie5.csv')
acharAnomalia(s5, t2)
#-----------------------------------------------#
```

2 Explicando a abordagem usada

Para esse problema eu tive que fazer algumas considerações:

- A série temporal se repete a cada N pontos
- A anomalia da série não acontece no inicio

A ideia principal para detectar a anomalia é fazer uma janela de tamanho N pontos proporcional a um ciclo, tal que nessa janela será computado a média e o desvio padrão dos dados. Espera-se que a anomalia tenha media e/ou desvio padrão diferente(s) quando comparada a um ciclo normal da série temporal.

Basicamente, a parte dificil é encontrar o tamanho N para a janela. Para fazer isso utilizou-se uma estratégia que cria clusters à médida que os dados vão sendo lidos. Essa estratégia resumidamente funciona da seguinte maneira:

na leitura do primeiro dado já é criado um cluster com 1 único dado e média igual ao valor do dado, criando assim o cluster 1. Apartir dai, quando um dado novo é lido é avaliado se ele está dentro do perímetro do cluster ou se ele está fora do perímetro do cluster. O perímetro de um cluster é definido como o $C_i \pm t$, onde C_i é o centroide do cluster i, e t é o desvio padrão do cluster. Quando um dado está dentro do perímetro do cluster, então aquele dado é inserido ao cluster e é recomputado o centroide do cluster. Quando um ponto cai fora do perímetro de algum cluster, esse ponto imediatemente criará um novo cluster. Para computar o parâmetro N é analisado quando o primeiro cluster C_1 reaperce em uma sequência de cluster. Por exemplo, dada uma sequencia de cluster $C_1, C_2, C_3, ..., C_k, C_1$, então $N = \sum_{k=1}^{n} C_{n_k}$, onde C_{n_k} é o número de pontos no cluster k. Apesar, dessa abordagem ser falha em alguns aspectos, para esse trabalho ela foi suficiente.

2.1 Resultados obtidos

A figuras abaixo mostra os resultados obtidos e logo abaixo tem o N calculado para a série. Note que a região em vermelho se refere a região de anomalia nos dados

2.1.1 Série 1

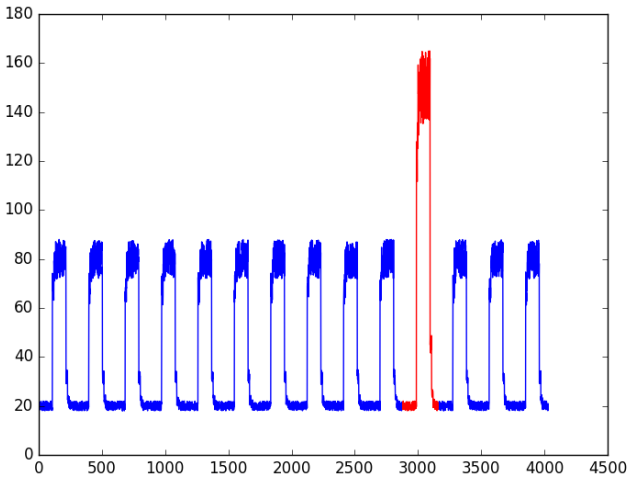

Figura 1: Resultado para a série 1. A região em vermelho mostra a anomalia.

$$N = 288$$

2.1.2 Série 2

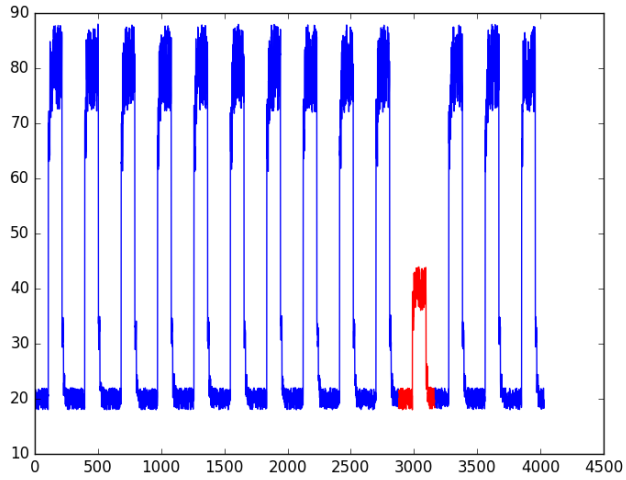

Figura 2: Resultado para a série 2. A região em vermelho mostra a anomalia.

$$N = 288$$

2.1.3 Série 3

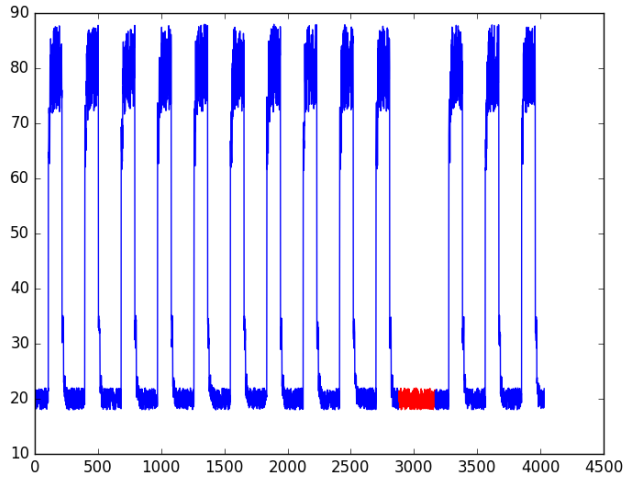

Figura 3: Resultado para a série 3. A região em vermelho mostra a anomalia.

$$N = 288$$

2.1.4 Série 4

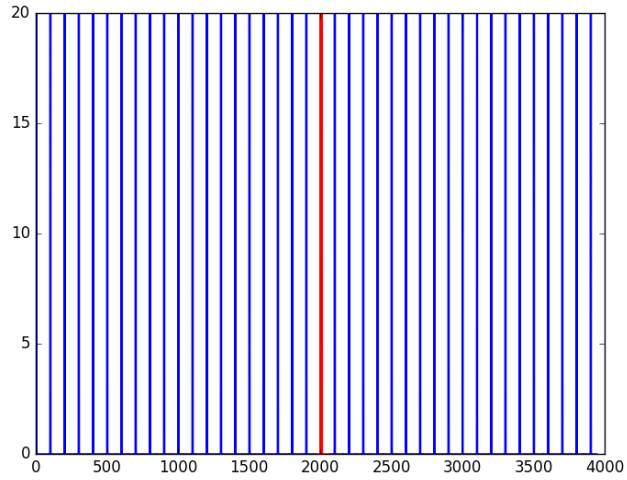

Figura 4: Resultado para a série 4. A região em vermelho mostra a anomalia.

$$N = 94$$

2.1.5 Série 5

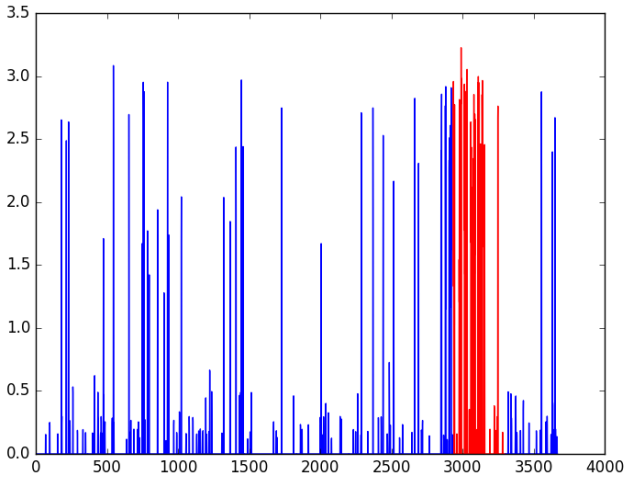

Figura 5: Resultado para a série 5. A região em vermelho mostra a anomalia.

$$N = 367$$