Sensation and percetion
Assignement1

Sami Sellami

September 12, 2018

TASK 1: Case 4 Confidence interval and linear regression

Calculating the confidence intervel:

We have a set of data points x and y representing the x-acceleration of the human CoM (Center of mass) during the walking:
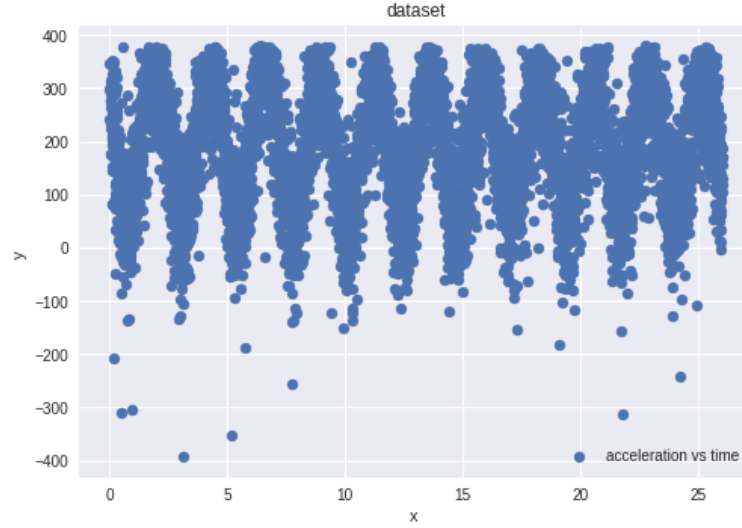

Figure 1: Acceleration in respect of time

We calculate the mean and standard deviation of t and x:

$$\bar{t} = \frac{\sum_{i=0}^{N} x_i}{N} = 12.99 \quad \sigma_t = \frac{\sum_{i=0}^{N}(x_i - \bar{x})^2}{N} = 7.50$$

$$\bar{x} = \frac{\sum_{i=0}^{N} y_i}{N} = 197.85 \quad \sigma_x = \frac{\sum_{i=0}^{N}(y_i - \bar{y})^2}{N} = 126.55$$

Now we have to remvove the outliers in the dataset, we use the instruction:

```
data= data[data.y<3*sd_y]
```
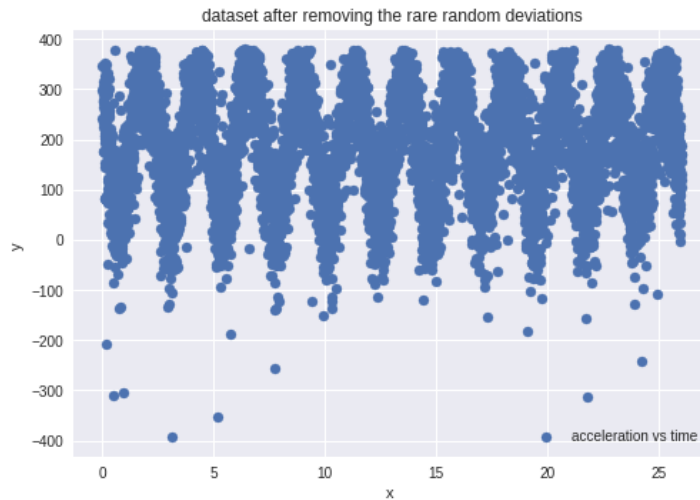
and we obtain the following graph:

Figure 2: Acceleration in respect of time after outliers elimination

Now we will compute the confidence interval; $\alpha = 1 - CL = 1 - 0.95 = 0.05$
The standard error is : $\sigma_{\bar{x}} = \frac{\sigma_x}{\sqrt{N}} = 1.628$
The z value for a confidence level corresponding to $CL = 95$ is calculated like this:

$$P(0.95 + 0.05/2) = P(0.975)$$

The z value for probability 0.9750 is 1.96 (using the standard normal table) The margin of error would be then:
$ME = z * \sigma_{\bar{x}} = 3.1932$ And finally we can write the expression of the confidence interval :

$$CI = [\bar{x} - ME, \bar{x} + ME] = [194.661, 201.045]$$

It represent the interval in which the mean value of acceleration will be in 95 per cent of the time (the experiments)

Performing the linear regression:

The graph of the data shows that the response follow a sinusoide in respect of time thus the true output can be written like this

$$y(t) = A + B_0 \cos(\omega t + \phi) = A + B \cos(\omega t) + C \sin(\omega t)$$

So for N observation we can write in matrix form:

$$\begin{pmatrix} y(1) \\ y(2) \\ ... \\ y(n) \end{pmatrix} = \begin{pmatrix} 1 & \cos(\omega t(1)) & \sin(\omega t(1)) \\ 1 & \cos(\omega t(2)) & \sin(\omega t(2)) \\ ... & ... & ... \\ 1 & \cos(\omega t(N)) & \sin(\omega t(N)) \end{pmatrix} \begin{pmatrix} A \\ B \\ C \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ ... \\ \epsilon_N \end{pmatrix}$$

In the form: $Y = X\beta + \epsilon$, and by minimizing the sum of squared erros we can find an estimate of the parameters :

$$\beta = (X^T X)^{-1} X^T Y$$

But first we have to estimate the period of our signal; we use for that the following code which estimate the distance between the mean value of the data and then compute the mean period at the end

```
def periode (data):
  x=data.x[0]
  p_data=data.y > mean_y
  T=[]
  k=1
  for i in range(size(data-10)):
    if (not(p_data[i])):
      for j in range (3):
        if (p_data[i+j]):
          k=0
      if (k==1):
        T.append(data.x[i])
        x=data.x[i]
```

We find $T = 2.28 \implies \omega = 2\pi/T = 2.75$ And finally

$$\beta = \begin{pmatrix} 185.926 \\ 101.021 \\ -40.583 \end{pmatrix}$$

The corresponding graph of the linear regression is as follows:

Figure 3: Linear regression and dataset

TASK 2 Case 17: RANSAC

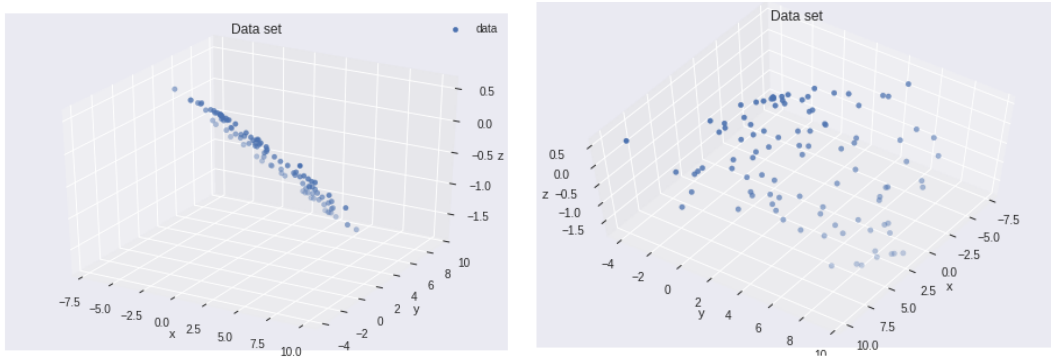We have a set of data set in 3D as shown in the graph below:

Figure 4: Dataset in 3D

After rotating the graph we clearly see that the dataset represent a plane in 3D and since we need 4 parameters to define a plane (model), the minimal sample set $n = 4$

The number of iterations is calculated with the formula :

$$k = \frac{\log(1 - P(success))}{\log(1 - \omega^n)}$$

With a $P(success) = 99$ and $\omega = 0.6$ we obtain: $k = 18.9243 = 18$

Then we implement a function that estimate the parameters of the plane that best fits the datas using RANSAC algorithm:

```
def RANSAC(data2, k, threshold, d):
  iteration=0
  while (iteration<k):
    r=[0, 0, 0]
    m_inlier=1.0*np.array([[0, 0, 0],[0, 0, 0], [0, 0, 0]])

    #we randomly select three points in the dataset
    for i in range(3):
      r[i]= np.random.choice(range(99))
      m_inlier[i]=data2.iloc[r[i]]
```

```
parameters=[0, 0, 0]
parameters= plan(m_inlier[0, :], m_inlier[1, :], m_inlier[2, :])
j=0
i=0
n=np.array([parameters[0],parameters[1], parameters[2]])
inlier=1.0*np.array([0, 0, 0])

for i in range(99):
  #we verify if the distance between the plane and the point is less the threshold
  if(abs(parameters[0]*data2.x[i]+parameters[1]*data2.y[i]+parameters[2]*data2.z[i])/np.
    #np.concatenate((inlier, np.array(data2.iloc[1])), axis=1)
    j=j+1

if j>d:
  best_parameters=parameters
  print(best_parameters)
  d=j
iteration=iteration+1
return best_parameters
```

By using the algorithm with a value of threshold equal to $t =$, We find the parameters of the plane that best fits our dataset $\beta = \begin{bmatrix} a & b & c & d \end{bmatrix} = \begin{bmatrix} 0.7 & 1.41 & 6.35 & 3.53 \end{bmatrix}$
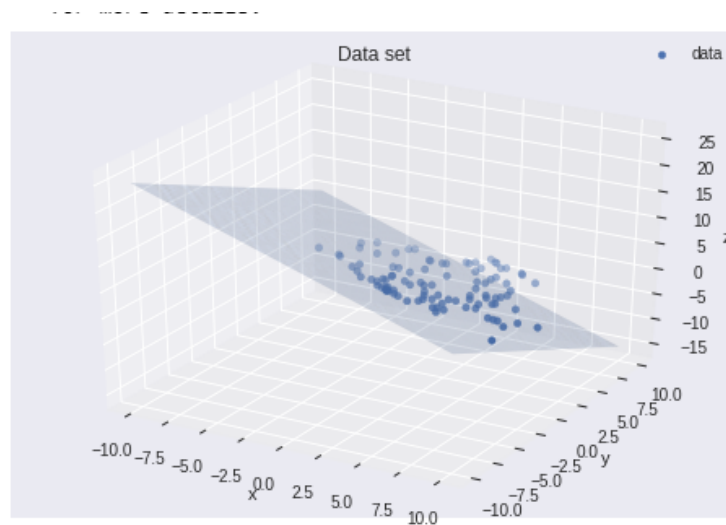
And finally we can plot the plane obtained along with our dataset:

Figure 5: Dataset and regression plane in 3D

4