b) We implement iteratively-reweighted least squares on noisy data and compare with oridnary least squares. Our code and output are as follows:

Code:

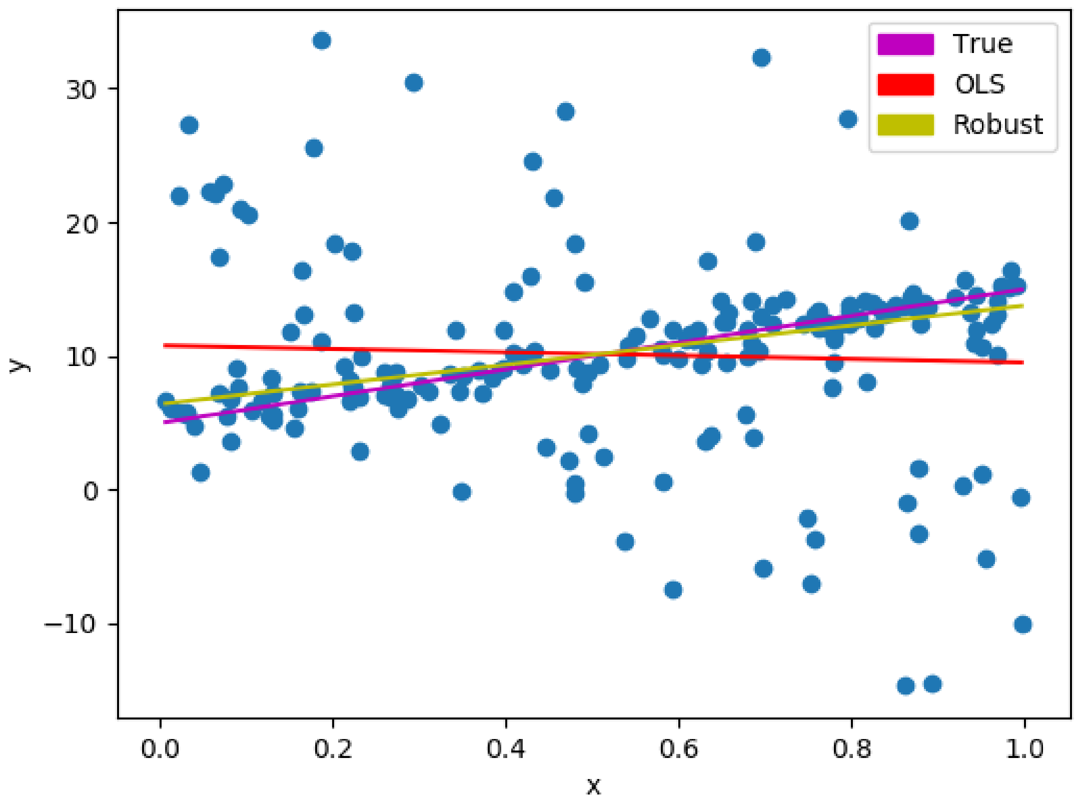
```
“import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches

epsilon = 1e-10
error = 1000000000

df=pd.read_csv('synthetic.csv', sep=',',header=None)
datavals=df.values
data=datavals.astype(np.float)

x = data[:,0]
y = data[:,1]
xtilde = np.hstack((np.ones((len(x),1)), x.reshape((len(x),1))))

def weight(r):
 return 1/ np.sqrt(1+r*r)

def WLS(X, C, y):
 return np.linalg.inv(X.transpose() @ np.diag(C) @ X) @ X.transpose() @ np.diag(C) @ y

w = np.zeros(xtilde.shape[1])
iter = 0

print("IRWLS iter, error, weight is")
while error > epsilon:
 iter += 1
 wold = w
 r = y - (xtilde @ w)
 C = weight(r)
 w= WLS(xtilde, C, y)
 error = np.linalg.norm(w-wold)
 print(iter, w, error)

plt.scatter(x, y)
xest = np.array([[1, x.min()], [1,x.max()]])
w_corr = [5, 10]
w_ols = WLS(xtilde, np.ones_like(C), y)
print("OLS weight is")
print(w_ols)

y_corr = xest @ w_corr
y_ols = xest @ w_ols
y_robust = xest @ w
```

```
plt.plot(xest[:,1], y_corr, label = "True Line", color = 'm')
plt.plot(xest[:,1], y_ols, label = "OLS Regression", color = 'r')
plt.plot(xest[:,1], y_robust, label = "Robust Regression", color = 'y')
plt.xlabel('x')
plt.ylabel('y')
green_patch = mpatches.Patch(color='m', label='True')
red_patch = mpatches.Patch(color='red', label='OLS')
blue_patch = mpatches.Patch(color='y', label='Robust')
plt.legend(handles=[green_patch, red_patch,blue_patch])
plt.show()”
```

Output:

```
IRWLS iter, error, weight is
1 [ 7.39804707 -2.97374443] 7.97334661995
2 [ 7.08857373  2.34424684] 5.32698835508
3 [ 6.76345936  5.15646072] 2.83094440806
4 [ 6.60159904  6.31548823] 1.17027497695
5 [ 6.5069146  6.8703754] 0.56290755617
6 [ 6.45711931  7.13159628] 0.265924650565
7 [ 6.43235617  7.25184294] 0.122769994193
8 [ 6.42042875  7.30635394] 0.0558006517927
9 [ 6.41479825  7.33084234] 0.0251273650228
10 [ 6.41217573  7.34178678] 0.0112542559642
11 [ 6.41096586  7.34666291] 0.00502398358147
12 [ 6.41041176  7.3488309 ] 0.00223768601854
13 [ 6.41015944  7.34979335] 0.00099497601713
14 [ 6.41004509  7.3502201 ] 0.000441804543342
15 [ 6.40999347  7.35040913] 0.00019594966468
16 [ 6.40997024  7.35049279] 8.68206962558e-05
17 [ 6.40995982  7.35052978] 3.84343764083e-05
18 [ 6.40995516  7.35054613] 1.70011653292e-05
19 [ 6.40995307  7.35055335] 7.51515388058e-06
20 [ 6.40995215  7.35055654] 3.31994218578e-06
21 [ 6.40995173  7.35055795] 1.46583708249e-06
22 [ 6.40995155  7.35055857] 6.46888017529e-07
23 [ 6.40995147  7.35055884] 2.85353666345e-07
24 [ 6.40995143  7.35055896] 1.25825542701e-07
25 [ 6.40995142  7.35055901] 5.54630494311e-08
26 [ 6.40995141  7.35055904] 2.44400880514e-08
27 [ 6.40995141  7.35055905] 1.07666495071e-08
28 [ 6.4099514   7.35055905] 4.74189166322e-09
29 [ 6.4099514   7.35055905] 2.08796871203e-09
30 [ 6.4099514   7.35055905] 9.19217227267e-10
31 [ 6.4099514   7.35055906] 4.04583906014e-10
32 [ 6.4099514   7.35055906] 1.78042770187e-10
33 [ 6.4099514   7.35055906] 7.83659417164e-11
OLS weight is
[ 10.78394203  -1.25342659]
```

c) We plot the coefficients of the true line, OLS, and iteratively-reweighted least squares on the same graph.


True
OLS
Robust
y
x