Table of Contents

Exercise 1: Solve a simple linear regression problem

Farzad Rezazadeh Pilehdarboni, 25.05.2022

```
% In this excercise you will create some simulated data and will fit simple
% linear regression models to it. Make sure to use rng(1998) prior to
% staring part a) to ensure consistent results.

clear all
close all
clc
rng(1000);                          % specify seed
i = 0;

sdAll = [40 60 100];
for li = 1:length(sdAll)

i = i + 2;
```

a)

```
% Using the linspace() function, create a vector, x, containing N=100
% equidistant observations between 1 and 100. This represents a feature, X.
% By default, linspace() generates a row vector. Use the transpose X = X'
% to obtain a column vector.

N = 50;
X = linspace(1,100,N);              % generate equidistant vector of length N
X = X';
```

b)

```
% Using the randn() function, create a vector, eps, containing 100
```

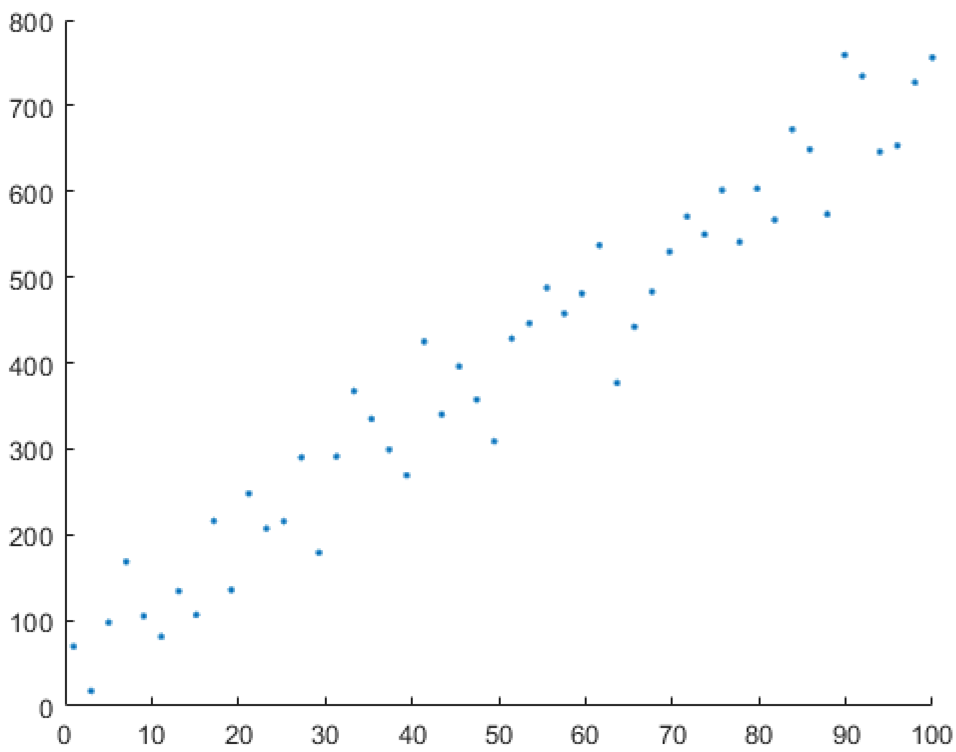
```
% observations drawn from N(0,30^2) distribution, i.e., a normal distribution
% with mean zero and standard deviation 30.

sd = sdAll(li);
eps = sd*randn(N,1);
```

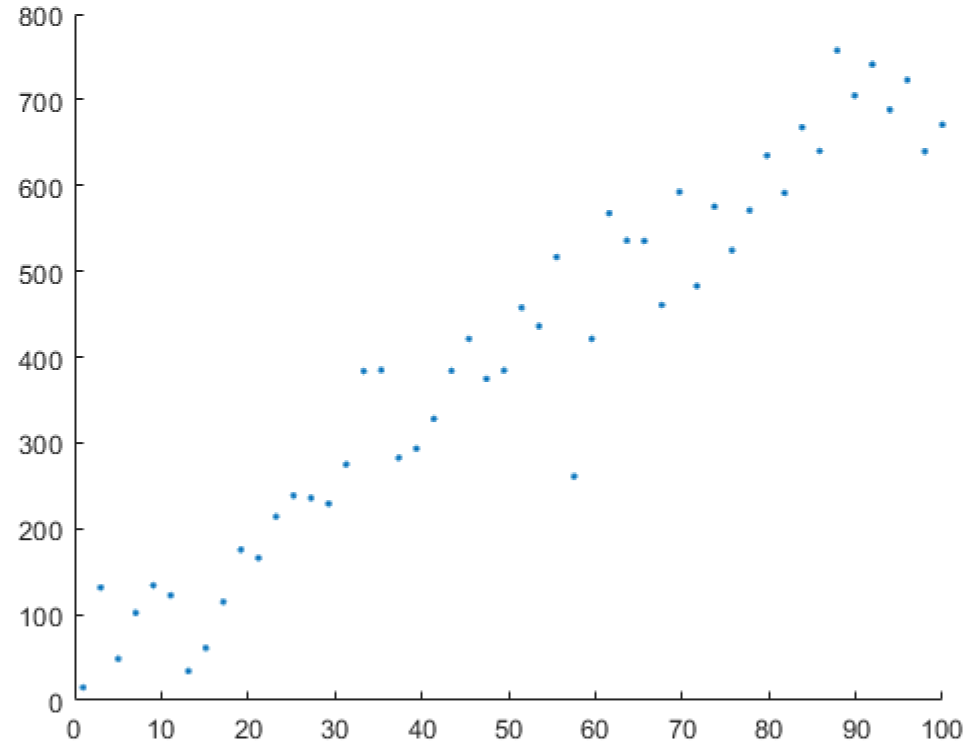
c)

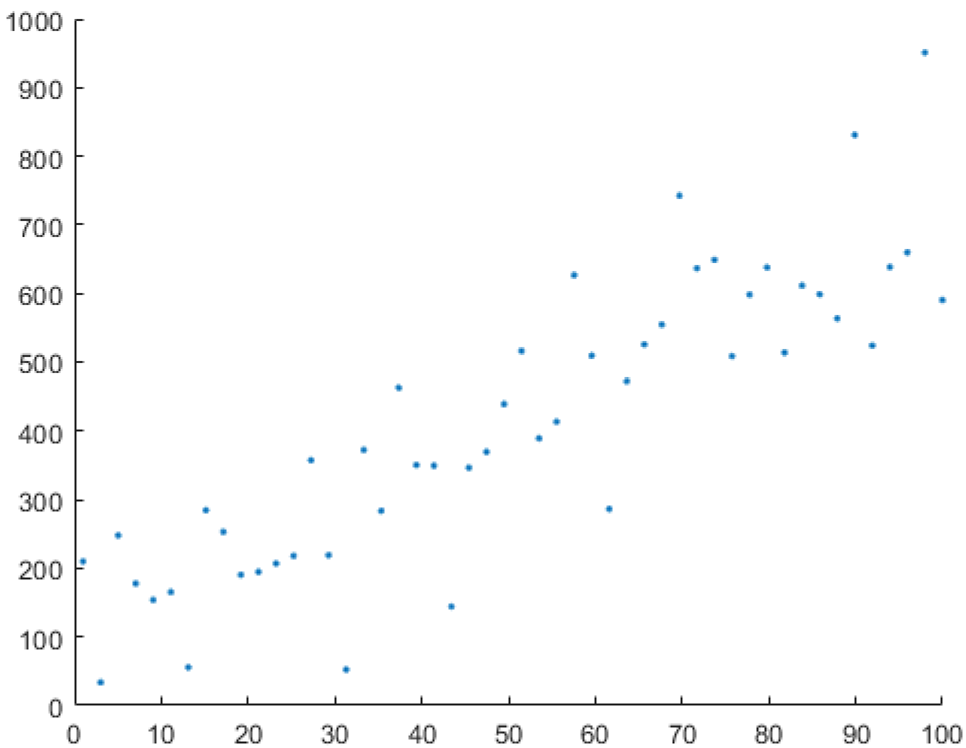
```
% Using x and eps, generate a vector y according to the model
% Y0 = 50 + 7*X
% and add the noise to obtain the disturbed output Y = Y0 + eps.
% What is the length of the vector y? What are the values of \beta_0 and
% \beta_1 in this linear model?

beta_0 = 50;
beta_1 = 7;
Y0 = beta_0 + beta_1*X;
Y = Y0 + eps;       % generate disturbed output
```

d)

```
% Using scatter(), create a scatterplot displaying the relationship
% between x and y. Comment on what you observe.
figure(i)
scatter(X,Y, '.')
```




800
700
600
500
400
300
200
100
0
0
10
20
30
40
50
60
70
80
90
100

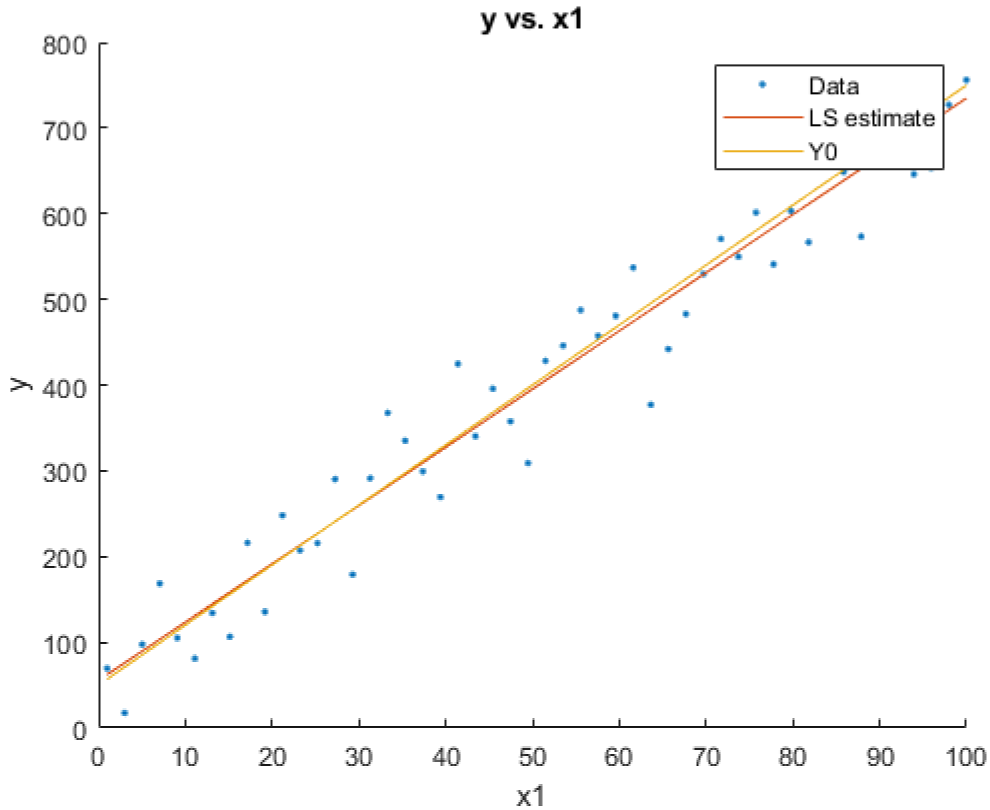

1000
900
800
700
600
500
400
300
200
100
0
0
10
20
30
40
50
60
70
80
90
100

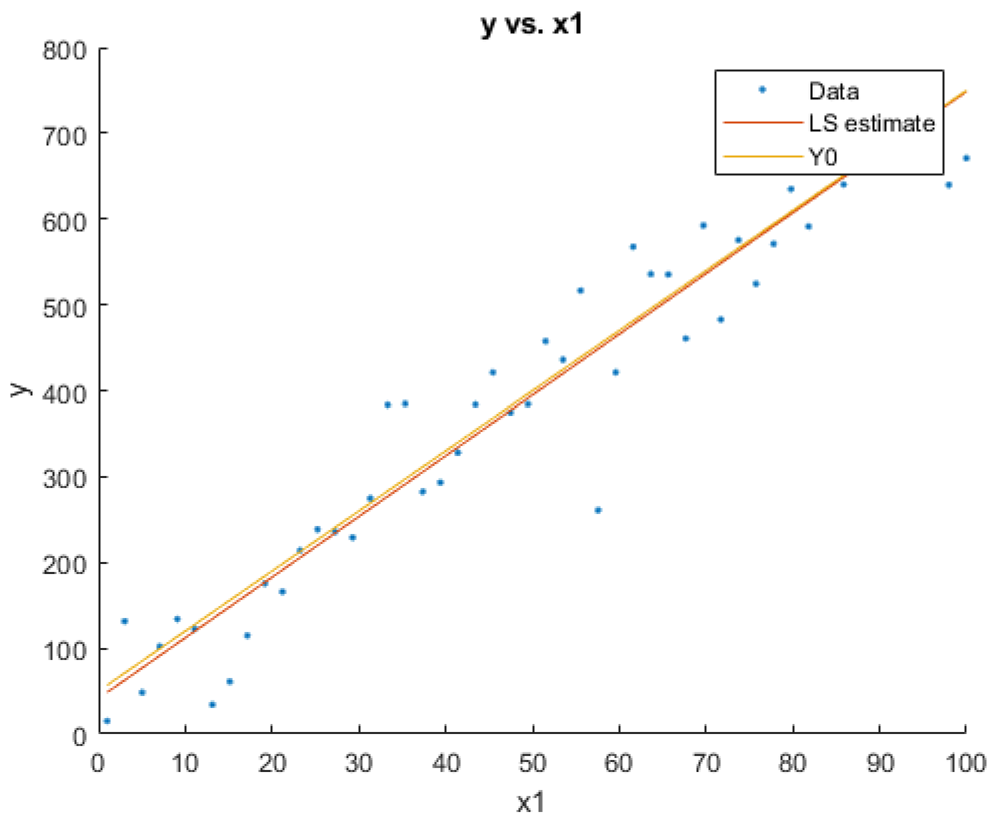
e)

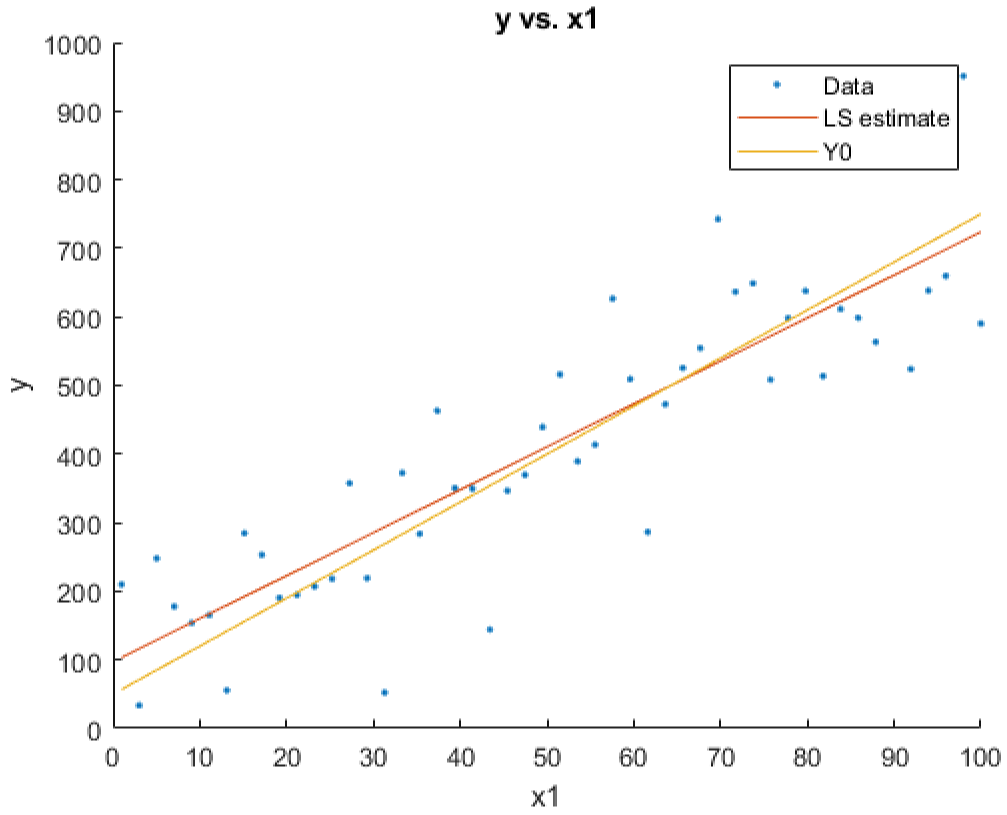
```
% Using the backslash operator, fit a least squares linear model to
% predict y using x. First, create the regression matrix using
% Phi=[ones(size(x)) x]. Comment on the model obtained. How do
% \hat{\beta}_0 and \hat{\beta}_1 compare to \beta_0 and \beta_1?

Phi = [ones(size(X)) X];
theta = Phi\Y;
```

f)

```
% Calculate the predictions \hat{Y} on the training data set. Display the
 least
% squares line on the scatterplot obtained in d) using hold all. Additonally,
 draw the
% population regression line on the plot.

Y_hat = Phi*theta;

hold all
plot(X,Y_hat)
plot(X,Y0)
legend('Data','LS estimate','Y0')
title('y vs. x1')
xlabel('x1')
ylabel('y')
```


y vs. x1
Data
LS estimate
Y0
800
700
600
500
400
300
200
100
0
y
0
10
20
30
40
50
60
70
80
90
100
x1


y vs. x1
Data
LS estimate
Y0
800
700
600
500
400
300
200
100
0
y
0
10
20
30
40
50
60
70
80
90
100
x1

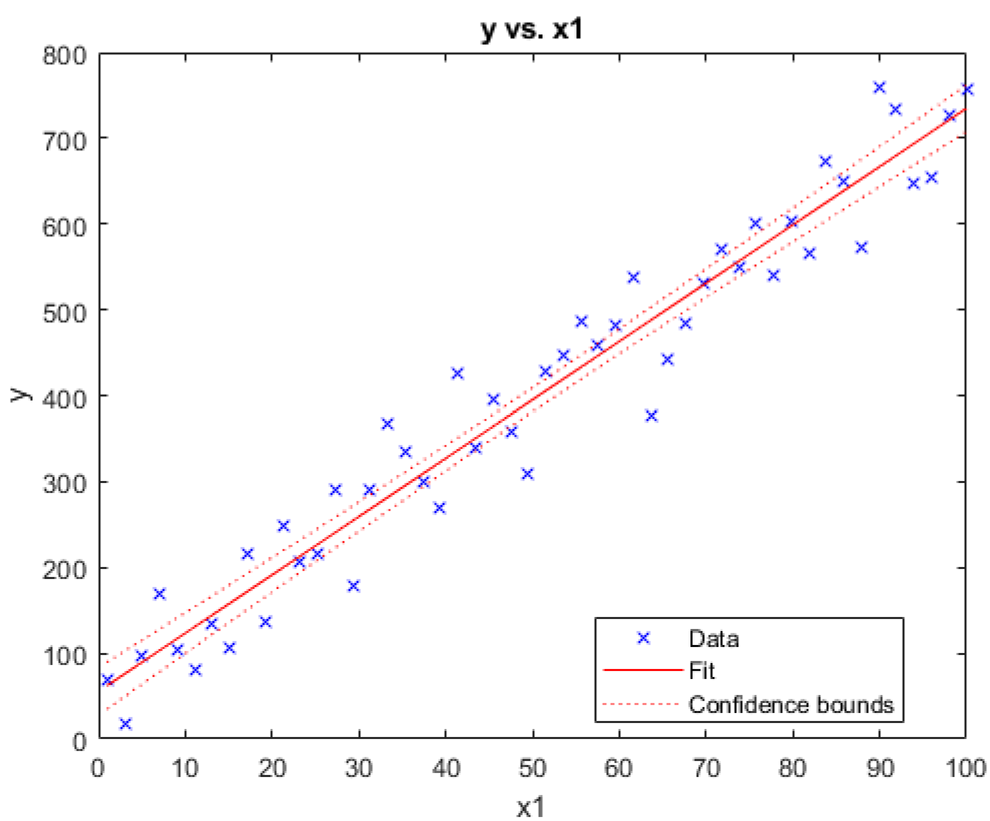


g)

```
% Alternatively, use the Matlab function fitlm to generate a linear
% regression model object. Plot the results by the use of plot(myModel) in a
 new figure.
myModel = fitlm(X,Y)

% By default, LinearModel assumes that you want to model the relationship
% as a straight line with an intercept term. The expression "y ~ 1 + x1"
% describes this model. Formally, this expression translates as "Y is
% modeled as a linear function which includes an intercept and a variable".
% Once again note that we are representing a model of the form Y = mX + B...
% The next block of text includes estimates for the coefficients, along
% with basic information regarding the reliability of those estimates.
% Finally, we have basic information about the goodness-of-fit including
% the R-square, the adjusted R-square and the Root Mean Squared Error.

figure(i+1)
plot(myModel)

% Notice that this simple command creates a plot with a wealth of information
 including
%
%    - A scatter plot of the original dataset
%    - A line showing our fit
```

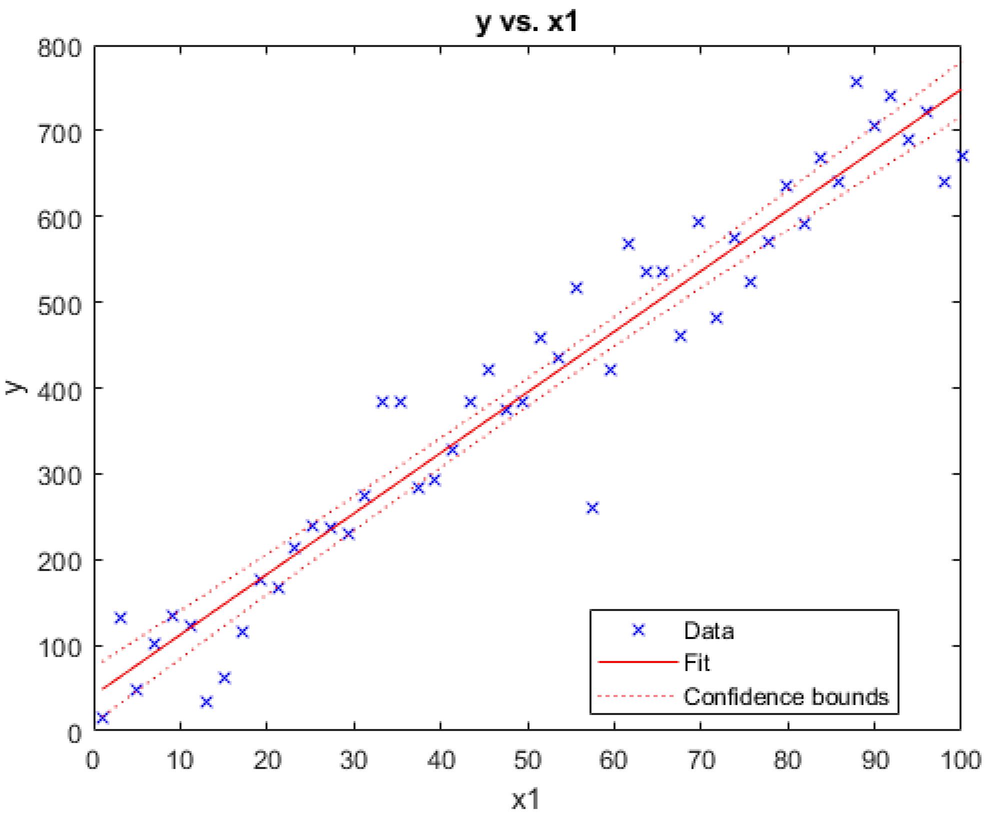
```
%    - Confidence intervals for the fit
%
% MATLAB has also automatically labelled our axes and added a legend.
```

```
myModel = 


Linear regression model:
    y ~ 1 + x1

Estimated Coefficients:
                   Estimate      SE        tStat       pValue   
                   ________    _______    _______    __________

    (Intercept)     55.482      13.781    4.0261     0.00020074
    x1              6.7941     0.23632    28.749     6.5799e-32


Number of observations: 50, Error degrees of freedom: 48
Root Mean Squared Error: 48.7
R-squared: 0.945,  Adjusted R-Squared: 0.944
F-statistic vs. constant model: 827, p-value = 6.58e-32
```



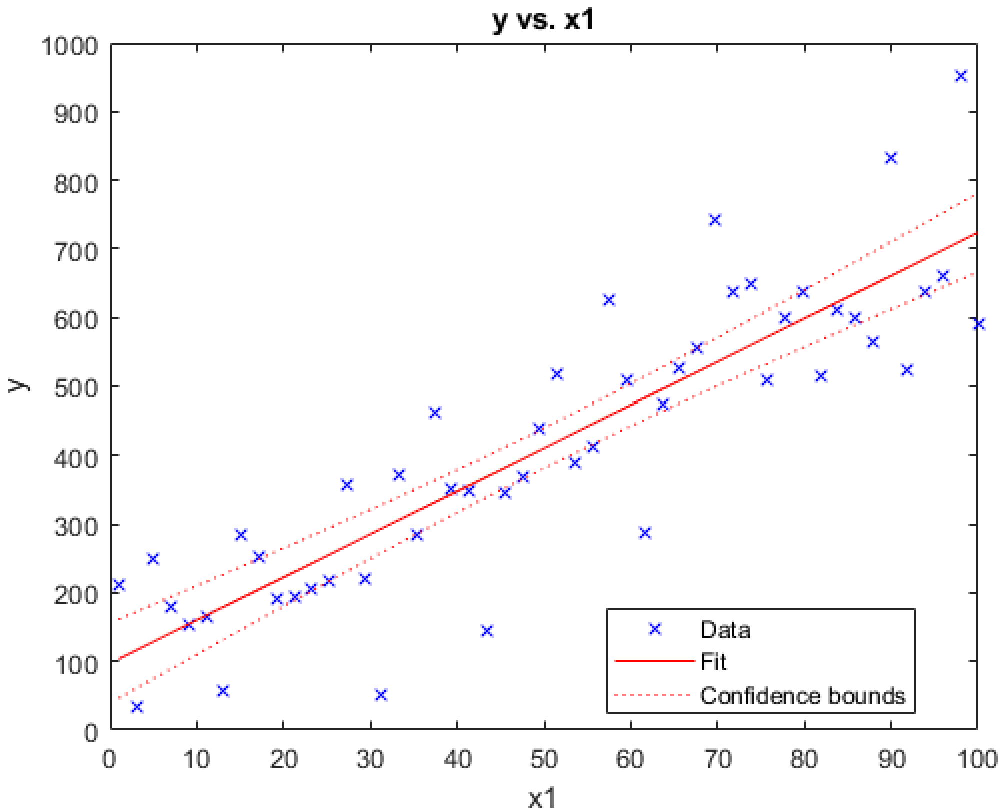

```
myModel = 
```

```
Linear regression model:
    y ~ 1 + x1

Estimated Coefficients:
                   Estimate      SE       tStat       pValue  
                   ________    _______    ______    __________

    (Intercept)     41.469      16.216    2.5573      0.013764
    x1              7.0716     0.27808     25.43    1.6633e-29


Number of observations: 50, Error degrees of freedom: 48
Root Mean Squared Error: 57.3
R-squared: 0.931,  Adjusted R-Squared: 0.929
F-statistic vs. constant model: 647, p-value = 1.66e-29
```




```
myModel = 


Linear regression model:
    y ~ 1 + x1

Estimated Coefficients:
                   Estimate      SE      tStat       pValue
```

```
                      _________    _______    _______    ___________

    (Intercept)       97.501       29.086     3.3522      0.0015707
    x1                6.2622       0.4988     12.555     8.9228e-17


Number of observations: 50, Error degrees of freedom: 48
Root Mean Squared Error: 103
R-squared: 0.767,  Adjusted R-Squared: 0.762
F-statistic vs. constant model: 158, p-value = 8.92e-17
```



h)

```
% Now fit a polynomial regression model that prdicts y using x and x^2.
% Is ther evidence that the quadratic term improves the model fit? Explain
% your answer.

myModel2 = fitlm(X,Y,'poly2')
```

```
myModel2 =


Linear regression model:
    y ~ 1 + x1 + x1^2
```

```
Estimated Coefficients:
                   Estimate       SE        tStat       pValue
                   _________    _________    ________    __________

    (Intercept)       49.108       20.859      2.3542      0.022792
    x1                7.1728      0.95413      7.5176    1.3481e-09
    x1^2          -0.0037491    0.0091472    -0.40986       0.68377


Number of observations: 50, Error degrees of freedom: 47
Root Mean Squared Error: 49.1
R-squared: 0.945,  Adjusted R-Squared: 0.943
F-statistic vs. constant model: 406, p-value = 2.19e-30


myModel2 =


Linear regression model:
    y ~ 1 + x1 + x1^2

Estimated Coefficients:
                   Estimate       SE       tStat       pValue
                   _________    ________    _______    __________

    (Intercept)       19.225      24.201    0.79437       0.43097
    x1                8.3931       1.107     7.5819    1.0784e-09
    x1^2           -0.013084    0.010613    -1.2328       0.22377


Number of observations: 50, Error degrees of freedom: 47
Root Mean Squared Error: 57
R-squared: 0.933,  Adjusted R-Squared: 0.93
F-statistic vs. constant model: 328, p-value = 2.53e-28


myModel2 =


Linear regression model:
    y ~ 1 + x1 + x1^2

Estimated Coefficients:
                   Estimate       SE       tStat      pValue
                   _________    ________    _______    _________

    (Intercept)       107.71       44.06     2.4445      0.01831
    x1                 5.656      2.0154     2.8064    0.0072676
    x1^2           0.0060022    0.019321    0.31065      0.75744


Number of observations: 50, Error degrees of freedom: 47
Root Mean Squared Error: 104
R-squared: 0.767,  Adjusted R-Squared: 0.757
```

F-statistic vs. constant model: 77.4, p-value = 1.35e-15

i)

```
% Repeat a)-g) with different noise levels by sampling from N=(0,\sigma^2)
% with \sigma={30,60,100} to generate different \eps. Describe your results.

end
```

Published with MATLAB® R2022a