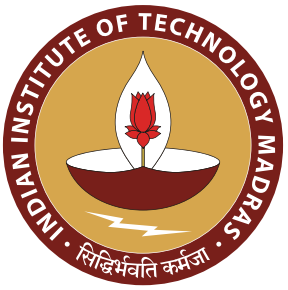


Department Of Aerospace Engineering
AS2101
Prof. Bharath Govindarajan
Prof. Ramakrishna M

LINEAR REGRESSION
TASK 2: 17-07-2021(Due-Date)

REPORT

AUTHOR
Yash Singh Jha-(AE19B016)

July 17 , 2021

Contents

Abstract

This document is like a summary report of the work we were assigned as Task 2 in the course As2101.
This week we were provided with 3 different data sets from an anonymous experiment. We then used the tweaked version of the principle of minimizing the sum of squared errors to generate the plot of the linear regression along with the scatter plot of the data set provided.
We were just required to code in Octave, but as a student of programming I was intrigued enough to code it in Python also.
The first chapter - Theory of Linear Regression, takes us through the theory of linear regression. The mathematics of which is explored in detail.
The second chapter - Analysis of Data Through Numeric Computation, takes us through the process of analysing the data set using Octave.
The third chapter - Results, presents the plots and the values of slope and intercept for every case taken up.
The fourth chapter - Conclusion, presents a concluding remark on the data set and the experimental results attained via computation. This LaTeX file itself is the part of the task for the second week of AS2101.

Chapter 1

Theory of Linear Regression

In statistics, linear regression is a linear approach to modelling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables).The case of one explanatory variable is called simple linear regression; for more than one, the process is called multiple linear regression. In this report, we will be strictly dealing with simple linear regression, mentioned as linear regression throughout the text.
In linear regression, the relationships are modeled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called linear models.[3] Most commonly, the conditional mean of the response given the values of the explanatory variables (or predictors) is assumed to be an affine function of those values; less commonly, the conditional median or some other quantile is used.
Linear regression models are often fitted using the least squares approach, but they may also be fitted in other ways, such as by minimizing the "lack of fit" in some other norm (as with least absolute deviations regression), or by minimizing a penalized version of the least squares cost function as in ridge regression (L^2-norm penalty) and lasso (L^1-norm penalty). Conversely, the least squares approach can be used to fit models that are not linear models. Thus, although the terms "least squares" and "linear model" are closely linked, they are not synonymous.

1.1 Formulation

Given a data set y_i,x_{i1},...,x_{ip} of n statistical units, a linear regression model assumes that the relationship between the dependent variable y and the p-vector of regressors, x is linear. This relationship is modeled through a disturbance term or error variable ϵ — an unobserved random variable that adds "noise" to the linear relationship between the dependent variable and regressors. Thus, the model takes the form :

$$y_i = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip} + \epsilon_i = \vec{x_i}^T \vec{\beta} + \epsilon_i \tag{1.1}$$

Sometimes the n equations are stacked together and written in matrix notation as

$$\vec{y} = \vec{X}\vec{\beta} + \vec{\epsilon} \tag{1.2}$$

1.2 Notation and Terminology

$\vec{y}$ is a vector of observed values of the variable called the regressand, endogenous variable, response variable, measured variable, criterion variable, or dependent variable.
$\vec{X}$ may be seen as a matrix of row-vectors x_i or of n-dimensional column-vectors X_j, which are known as regressors, exogenous variables, explanatory variables, covariates, input variables, predictor variables, or independent variables.
$\vec{\beta}$ is a (p+1) dimensional parameter vector, where β_0 is the intercept term.
$\vec{\epsilon}$ is a vector of values ϵ_i. This part of the model is called the error term, disturbance term, or sometimes noise (in contrast with the "signal" provided by the rest of the model). This variable captures all other factors which influence the dependent variable y other than the regressors x. The relationship between the error term and the regressors, for example their correlation, is a crucial consideration in formulating a linear regression model, as it will determine the appropriate estimation method.
Fitting a linear model to a given data set usually requires estimating the regression coefficients β such that the error term ϵ is minimised.

1.3 Example

Consider a situation where a small ball is being tossed up in the air and then we measure its heights of ascent hi at various moments in time ti. Physics tells us that, ignoring the drag, the relationship can be modeled as:

$$h_i = \beta_1 t_i + \beta_2 t_i^2 + \epsilon_i \tag{1.3}$$

where β_1 determines the initial velocity of the ball, β_2 is proportional to the standard gravity, and ϵ_i is due to measurement errors. Linear regression can be used to estimate the values of β_1 and β_2 from the measured data. This model is non-linear in the time variable, but it is linear in the parameters β_1 and β_2; if we take regressors $x_i = (x_{i1},x_{i2}) = (t_i,t_i^2)$, the model takes on the standard form

$$h_i = x_i^T \beta + \epsilon_i \tag{1.4}$$

1.4 Simple Linear Regression

The very simplest case of a single scalar predictor variable x and a single scalar response variable y is known as simple linear regression. Refer figure 1.1 as an example of the simple linear regression.

1.5 The Normal Equation

For further computations, we use the normal equation. Basically, we are minimizing the sum of the squared errors between our predicted equation and the actual y values. This is a pretty decent error measure- by far the most widely used measure. One of the most attractive features of the linear least-squares method is that it has a closed-form solution; that is, no iteration / numerical computation is needed. That closed-form solution is called the normal equation. Anyway, if you want to learn more about the derivation of the normal equation, you can read about it on wikipedia.

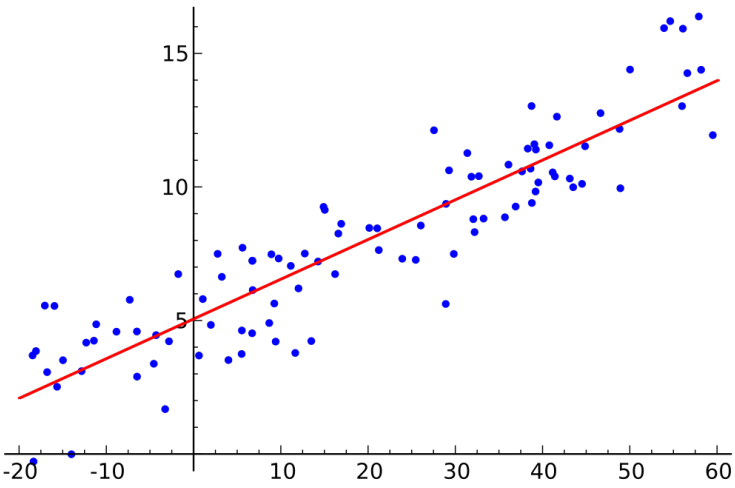


Figure 1.1: Simple Linear Regression

Chapter 2

Analysis of Data Through Numeric Computation

This chapter takes us through the code in Octave used to analyse the data set provided to us in the .txt files.

2.1 Loading Data

We are provide with the data points in a .txt file. So, we need to extract the data onto an array/matrix in Octave in order to proceed further. We do that using the dlmread() function in Octave and saving the appropriate data points as dependent and independent variables. Below $data_x$ and $data_y$ refers to the dependent and independent variables respectively.

```
data = dlmread(filename); % Example of a filename is 'data1.txt'.
% Here m is the number of data points we use in our analysis.
data_x = data(1:m,1);
data_y = data(1:m,2);
```

Listing 2.1: Loading Data

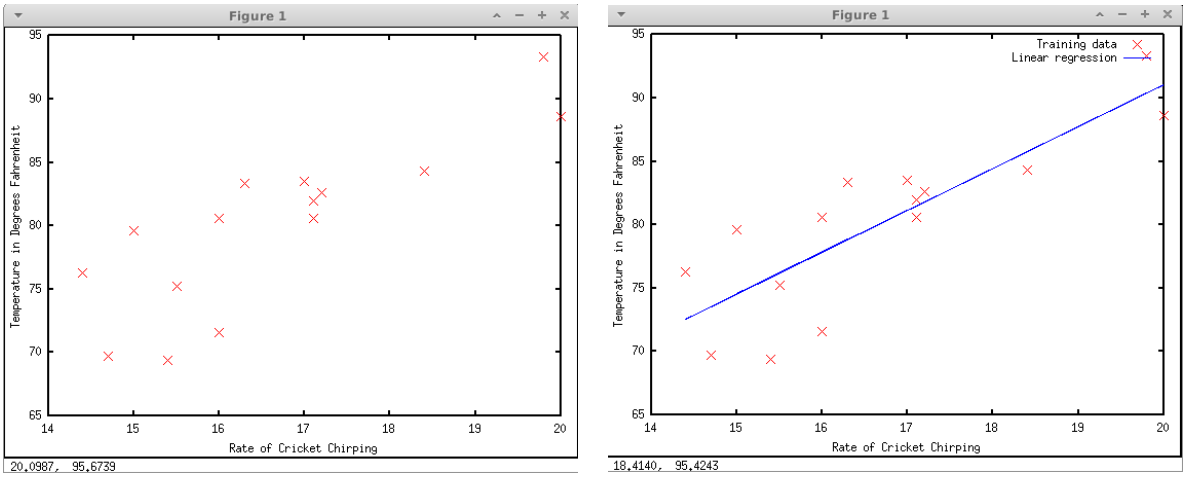
The value of m is taken to be 50,100 and 200 for every data file available.

2.2 Plotting the data

We next would like to plot the data - to see what it actually look like. We define a function plotData(x,y) that takes 2 vectors as it's input and plots the data using a desired marker.

```
function plotdata (x,y)
  % Plot the data:
  plot(x,y,'rx','MarkerSize',2);
endfunction
```

Listing 2.2: Function to plot data.



(a) Scatter plot of raw data.

(b) Linear regression model.

Figure 2.1: Plots using Octave

So, we can now plot the data as follows:

```
plotdata(data_x,data_y)
xlabel('Rate of Cricket Chirping') % Set the x-label
ylabel('Temperature in Degrees Fahrenheit')% Set the y-label
hold on
```

Listing 2.3: Plotting Data.

Figure 2.1a is an example of the scatter plot of raw data.

2.3 Using Normal Equation

Getting the theoretical gist of the Normal Equation we now present the normal equation as follows:

$$\vec{\Theta} = (X^T X)^{-1} X^T y \tag{2.1}$$

Putting that into octave as follows:

```
% We want to allow a non-zero intercept for our linear equation. So we add
    a column of all ones to our x column.
n = length(data_x);
%Add a column of all ones to data_x;
data_X = [ones(n,1) data_x];
% We use the normal equation to calculate theta.
% Calculate theta = [intercept , slope]
theta = (pinv(data_X'*data_X))*data_X'*data_y;
```

Listing 2.4: Using the normal equation

If we get theta = [24.9660; 3.3058]. This means that our fitted equation is as follows: $y = 3.3058x + 24.9660$.
Now, let's plot our fitted equation (prediction) on top of the training data, to see if our fitted equation makes sense.

Note - In the actual octave code, we define a function to calculate the inverse of the given matrix using Gauss-Jordan elimination method.

```
% Plot the fitted equation we got from the regression
 plot(data_X(:,2), data_X*theta,'-')
```

Listing 2.5: Plotting using Linear Regression

Figure 2.1b is the plot attained finally.

2.4 Inverse

This section presents the code to calculate the inverse of a matrix in octave. The code is pretty simple and follows the Gauss-Jordan Elimination method to compute the inverse of the matrix.In the original code this code is present as a function which when called upon, returns the inverted matrix of the input matrix.

```
%Finding dimensions of the matrix:
[r,c] = size(A);
n = r;
b = eye(n);
if r~=c
    disp('Only Square Matrices have inverse')
    b=[];
    exit()
end
a = [A eye(n)];
%Apply Guass Jordan Elimination:
for i = [1:n]
    if a(i,i) == 0.0
      printf("Not Invertible")
      exit()
    endif
    for j = [1:n]
      if i ~= j
        ratio = a(j,i)/a(i,i);
        for k = [1:2*n]
          a(j,k) = a(j,k) - ratio*a(i,k);
        endfor
      endif
    endfor
endfor
%Row operation to make principal diagonal element to 1:
for i = [1:n]
    divisor = a(i,i);
    for j =[1:2*n]
      a(i,j) = a(i,j)/divisor;
    endfor
endfor
for i = [1:n]
    for j =[n+1:2*n]
      b(i,j-n) = a(i,j);
    endfor
endfor
```

Listing 2.6: Inverse using Gauss-Jordan Elimination

Chapter 3

Results

In this chapter we present the results attained after running the codes for all three data sets, each for a different number of data points.

3.1 Intercept and Slope

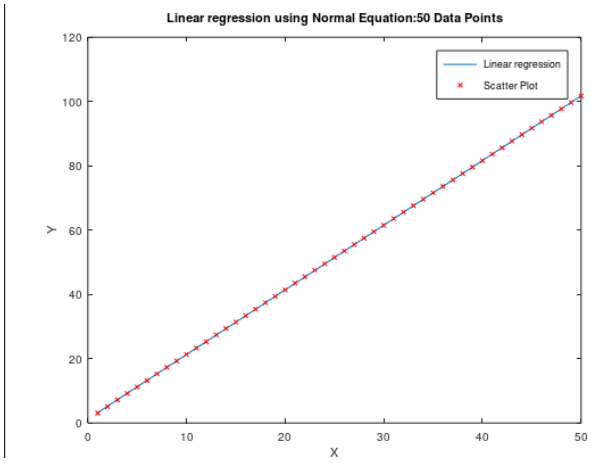
This section provides the intercept and slope values for different amount of data points considered for the three data sets provided. Table 3.1 summarises the result obtained from the Octave/Python code uploaded on github.

FILENAME	NO. OF DATA POINTS	SLOPE	INTERCEPT
'data1.txt'	50	2.011294	1.188000
	100	2.008000	1.266000
	200	2.005651	1.377050
'data2.txt'	50	2.011076	1.195634
	100	2.007912	1.271927
	200	2.005624	1.380982
'data3.txt'	50	1.999999	1.005264
	100	1.999990	1.005402
	200	2.000002	1.004887

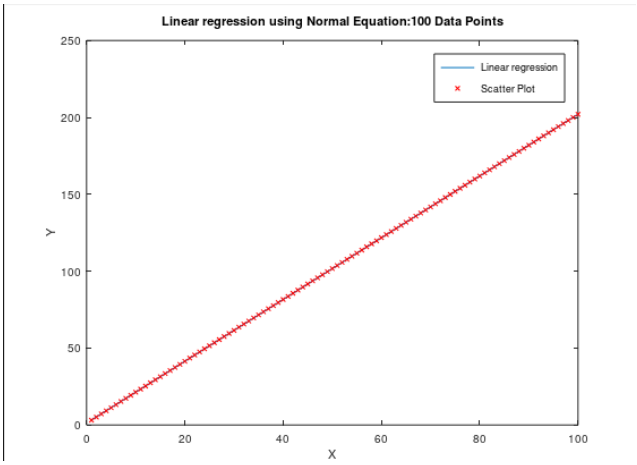
Table 3.1: Slope and Intercept Values

3.2 Plots

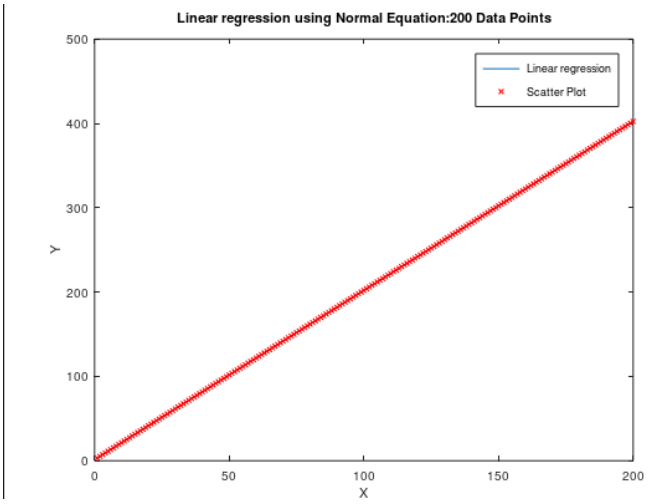
The data provided are such that all plots appear the same to the naked eye. Please visit the codes provided to see the difference yourself, or alternately visit Table 3.1 for the insights into the differences in slopes and intercepts.
Figures 3.1, 3.2and 3.3 are the plots for the data-sets provided in files-'data1.txt','data2.txt' and 'data3.txt' respectively.
Note: The author preferred the plots without mesh so he chose Figures 3.1 and 3.2 accordingly, but for the sake of presentation - he has included mesh in Figure 3.3



(a) 50 Data points.

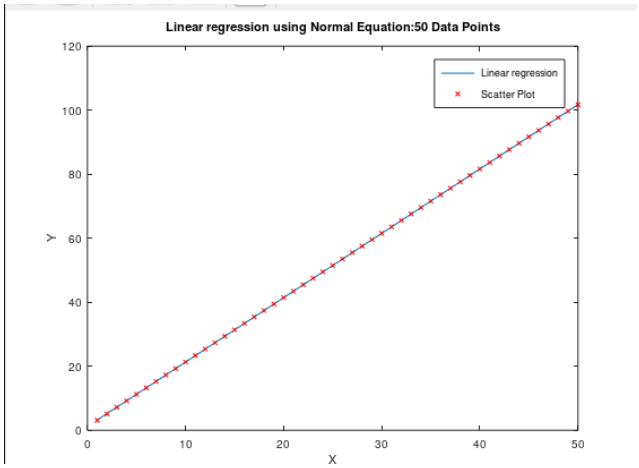


(b) 100 Data Points.

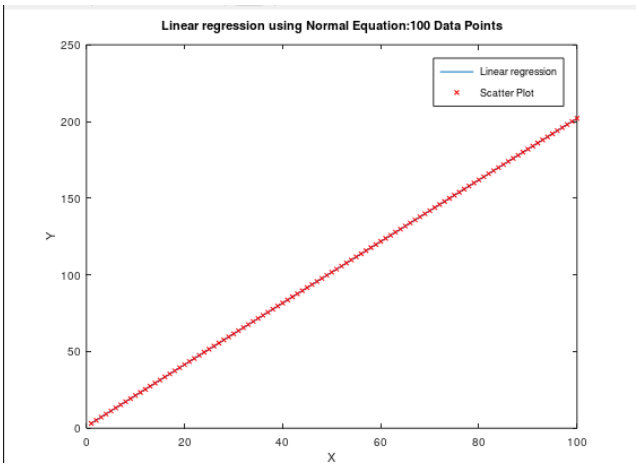


(c) 200 Data Points.

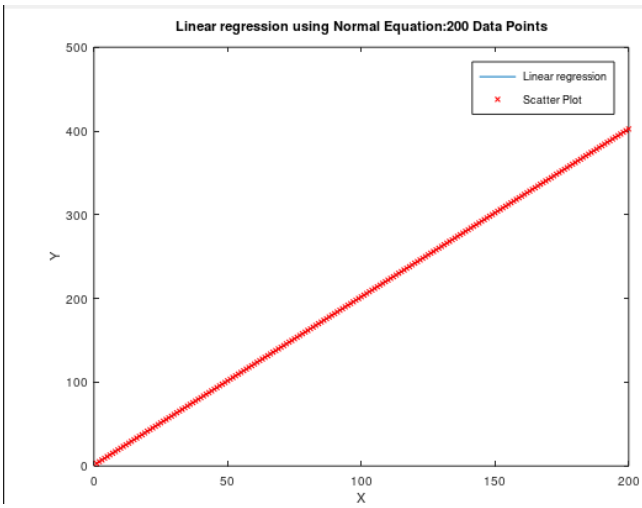
Figure 3.1: Plots for 'data1.txt'



(a) 50 Data points.

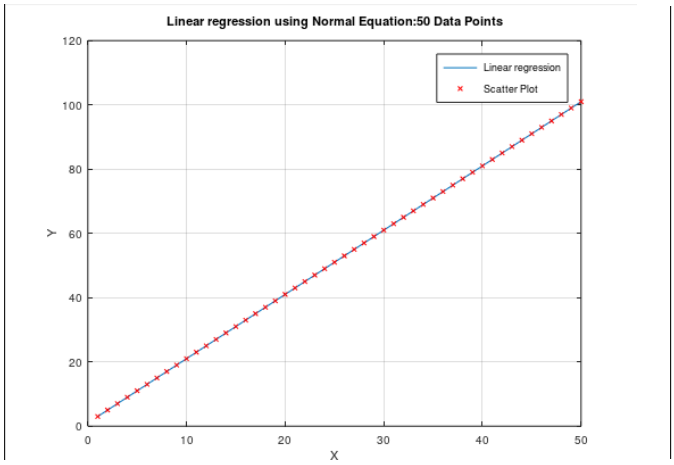


(b) 100 Data Points.

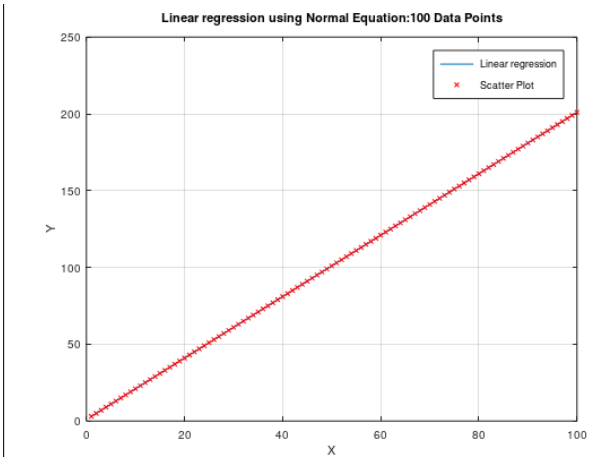


(c) 200 Data Points.

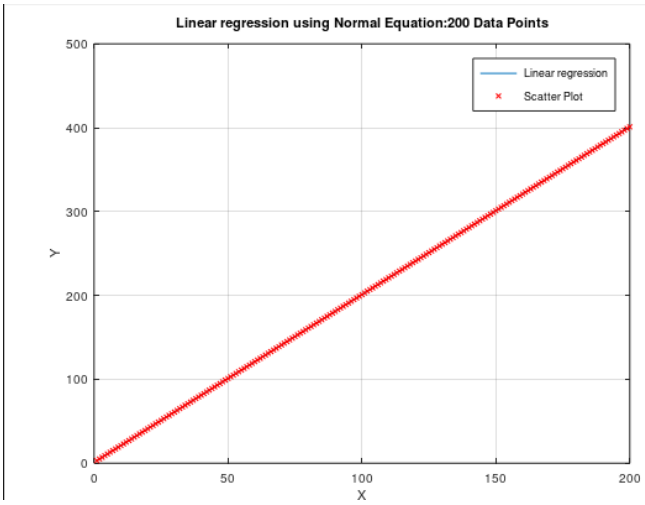
Figure 3.2: Plots for 'data2.txt'



(a) 50 Data points.



(b) 100 Data Points.



(c) 200 Data Points.

Figure 3.3: Plots for 'data3.txt'

Chapter 4

Conclusion

Looking closely into the plots presented in the previous chapter, we conclude that the method of Linear Regression works very well with the data-sets provided to us. However, for the sake of completion, the alternative methods are listed below which can be used when the data points violate the linear trend aggressively.
Alternatives are:

1. Different linear model: fitting a linear model with additional X variable(s)
2. Nonlinear model: fitting a nonlinear model when the linear model is inappropriate
3. Transformations: correcting nonnormality, nonlinearity, or unequal variances by transforming all the data values for X and/or Y.
4. Weighted least squares linear regression: dealing with unequal variances in Y by performing a weighted least squares fit
5. Alternative regression methods: dealing with problems by employing a non-least-squares method of fitting
6. Removing outliers: refitting the linear model after removing outliers or high-leverage or influential data points.
7. Mechanical methods: Finding the best selection of X variables by mechanical means
8. Methods aimed at dealing with multicollinearity

For the detailed codes in Octave and Python, please visit github.