General Linear Least-Squares and Nonlinear Regression

Berlin Chen
Department of Computer Science & Information Engineering
National Taiwan Normal University

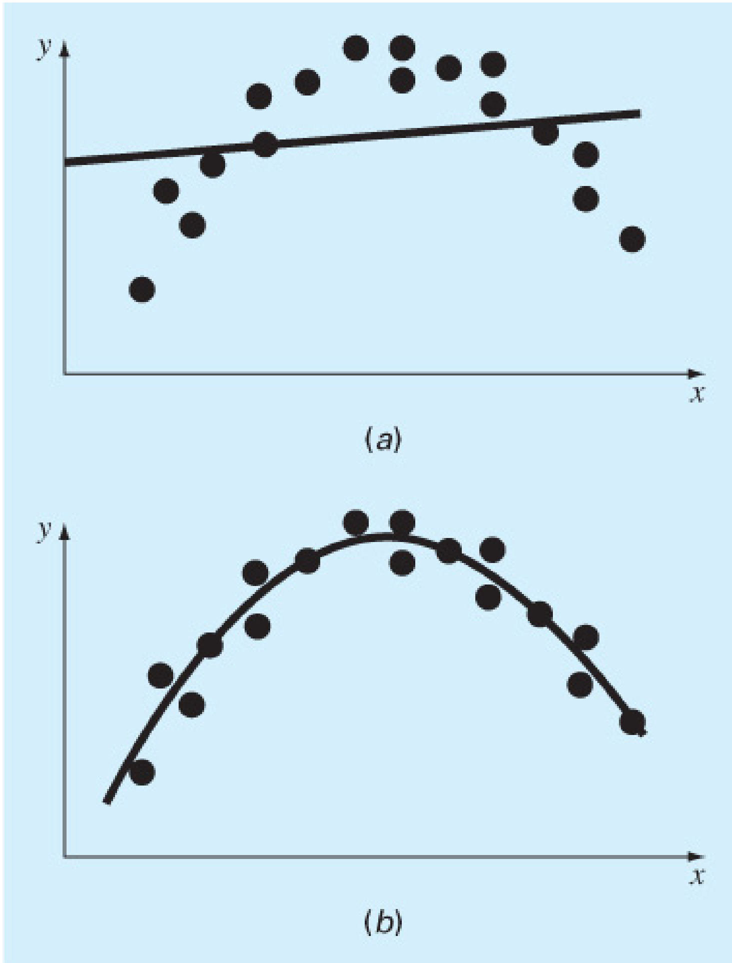
Reference:
1. *Applied Numerical Methods with MATLAB for Engineers*, Chapter 15 & Teaching material

Chapter Objectives

- Knowing how to implement polynomial regression
- Knowing how to implement multiple linear regression
- Understanding the formulation of the general linear least-squares model
- Understanding how the general linear least-squares model can be solved with MATLAB using either the normal equations or left division
- Understanding how to implement nonlinear regression with optimization techniques

Polynomial Regression

- The least-squares procedure from Chapter 14 can be readily extended to fit data to a higher-order polynomial. Again, the idea is to minimize the sum of the squares of the estimate residuals
- The figure shows the same data fit with:
  a) A first order polynomial
  b) A second order polynomial



Process and Measures of Fit

- For **a second order polynomial**, the best fit would mean minimizing:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i - a_2 x_i^2\right)^2$$

- In general, for **an m^{th} order polynomial**, this would mean minimizing :

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i - a_2 x_i^2 - \cdots - a_m x_i^m\right)^2$$

- The standard error for fitting an m^{th} order polynomial to n data points is:

$$s_{y/x} = \sqrt{\frac{S_r}{n-(m+1)}}$$

because the m^{th} order polynomial has $(m+1)$ coefficients

- The coefficient of determination r^2 is still found using:

$$r^2 = \frac{S_t - S_r}{S_t}$$

Polynomial Regression: An Example

- Second Order Polynomial

For this case the sum of the squares of the residuals is

$$S_r = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i - a_2 x_i^2\right)^2 \tag{15.2}$$

To generate the least-squares fit, we take the derivative of Eq. (15.2) with respect to each of the unknown coefficients of the polynomial, as in

$$\frac{\partial S_r}{\partial a_0} = -2 \sum \left(y_i - a_0 - a_1 x_i - a_2 x_i^2\right)$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum x_i \left(y_i - a_0 - a_1 x_i - a_2 x_i^2\right)$$

$$\frac{\partial S_r}{\partial a_2} = -2 \sum x_i^2 \left(y_i - a_0 - a_1 x_i - a_2 x_i^2\right)$$

These equations can be set equal to zero and rearranged to develop the following set of normal equations:

$$\begin{aligned} (n)a_0 + \left(\sum x_i\right) a_1 + \left(\sum x_i^2\right) a_2 &= \sum y_i \\ \left(\sum x_i\right) a_0 + \left(\sum x_i^2\right) a_1 + \left(\sum x_i^3\right) a_2 &= \sum x_i y_i \\ \left(\sum x_i^2\right) a_0 + \left(\sum x_i^3\right) a_1 + \left(\sum x_i^4\right) a_2 &= \sum x_i^2 y_i \end{aligned} \Rightarrow \begin{bmatrix} n & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} \sum y_i \\ \sum x_i y_i \\ \sum x_i^2 y_i \end{bmatrix}$$

Multiple Linear Regression (1/2)

- Another useful extension of linear regression is the case where *y* is a linear function of two or more independent variables:

$$y = a_0 + a_1 x_1 + a_2 x_2 + \cdots a_m x_m$$

- Again, the best fit is obtained by minimizing the sum of the squares of the estimate residuals:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_{1,i} - a_2 x_{2,i} - \cdots a_m x_{m,i} \right)^2$$

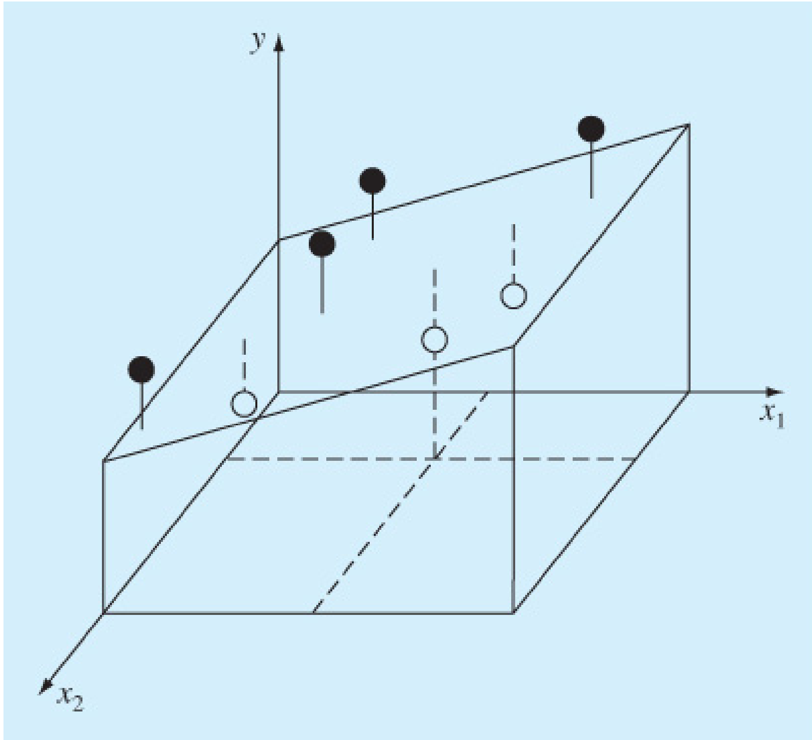


FIGURE 15.3
Graphical depiction of multiple linear regression where y is a linear function of x_1 and x_2.

For two-dimensional case, the regression "line" becomes a "plane"

Multiple Linear Regression (2/2)

As with the previous cases, the "best" values of the coefficients are determined by formulating the sum of the squares of the residuals:

$$S_r = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_{1,i} - a_2 x_{2,i})^2 \tag{15.4}$$

and differentiating with respect to each of the unknown coefficients:

$$\frac{\partial S_r}{\partial a_0} = -2 \sum (y_i - a_0 - a_1 x_{1,i} - a_2 x_{2,i})$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum x_{1,i}(y_i - a_0 - a_1 x_{1,i} - a_2 x_{2,i})$$

$$\frac{\partial S_r}{\partial a_2} = -2 \sum x_{2,i}(y_i - a_0 - a_1 x_{1,i} - a_2 x_{2,i})$$

The coefficients yielding the minimum sum of the squares of the residuals are obtained by setting the partial derivatives equal to zero and expressing the result in matrix form as

$$\begin{bmatrix} n & \sum x_{1,i} & \sum x_{2,i} \\ \sum x_{1,i} & \sum x_{1,i}^2 & \sum x_{1,i} x_{2,i} \\ \sum x_{2,i} & \sum x_{1,i} x_{2,i} & \sum x_{2,i}^2 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \sum y_i \\ \sum x_{1,i} y_i \\ \sum x_{2,i} y_i \end{Bmatrix} \tag{15.5}$$

Multiple Linear Regression: An Example

Multiple Linear Regression

Problem Statement. The following data were created from the equation $y = 5 + 4x_1 - 3x_2$:

x_1	x_2	y
0	0	5
2	1	10
2.5	2	9
1	3	0
4	6	3
7	2	27

Example 15.2

Use multiple linear regression to fit this data.

Solution. The summations required to develop Eq. (15.5) are computed in Table 15.2. Substituting them into Eq. (15.5) gives

$$\begin{bmatrix} 6 & 16.5 & 14 \\ 16.5 & 76.25 & 48 \\ 14 & 48 & 54 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} 54 \\ 243.5 \\ 100 \end{Bmatrix} \tag{15.6}$$

which can be solved for

$$a_0 = 5 \qquad a_1 = 4 \qquad a_2 = -3$$

which is consistent with the original equation from which the data were derived.

TABLE 15.2 Computations required to develop the normal equations for Example 15.2.

y	x_1	x_2	x_1^2	x_2^2	x_1x_2	x_1y	x_2y
5	0	0	0	0	0	0	0
10	2	1	4	1	2	20	10
9	2.5	2	6.25	4	5	22.5	18
0	1	3	1	9	3	0	0
3	4	6	16	36	24	12	18
27	7	2	49	4	14	189	54
54	16.5	14	76.25	54	48	243.5	100

General Linear Least Squares

- Linear, polynomial, and multiple linear regression all belong to the ***general linear least-squares model***:

$$y = a_0 z_0 + a_1 z_1 + a_2 z_2 + \cdots a_m z_m + e$$

  - where z_0, z_1, …, z_m are a set of m+1 ***basis functions*** and e is the error of the fit

- The basis functions can be any function data but *cannot* contain any of the coefficients a_0, a_1, etc.
  - E.g.,

$$y = a_0 + a_1 \cos(\omega x) + a_2 \sin(\omega x)$$

  - However, the following simple-looking model is truly "***nonlinear***"

$$y = a_0\left(1 - e^{-a1x}\right)$$

Solving General Linear Least Squares Coefficients (1/2)

- The equation:

$$y = a_0 z_0 + a_1 z_1 + a_2 z_2 + \cdots a_m z_m + e$$

  can be re-written for each data point as a matrix equation:

$$\{y\} = [Z]\{a\} + \{e\}$$

  where {*y*} contains the dependent data, {*a*} contains the coefficients of the equation, {*e*} contains the error at each point, and [*Z*] is:

$$[Z] = \begin{bmatrix} z_{01} & z_{11} & \cdots & z_{m1} \\ z_{02} & z_{12} & \cdots & z_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ z_{0n} & z_{1n} & \cdots & z_{mn} \end{bmatrix}$$

-

  with z_{ji} representing the the value of the j^{th} basis function calculated at the I^{th} point

Solving General Linear Least Squares Coefficients (2/2)

- Generally, [*Z*] is not a square matrix, so simple inversion cannot be used to solve for {*a*}. Instead the sum of the squares of the estimate residuals is minimized:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{m} a_j z_{ji} \right)^2$$

- The outcome of this minimization process is the ***normal equations*** that can expressed concisely in a matrix form as:

$$\left[[Z]^T [Z] \right] \{a\} = \left\{ [Z]^T \{y\} \right\}$$

MATLAB Example

- Given *x* and *y* data in columns, solve for the coefficients of the best fit line for $y=a_0+a_1x+a_2x^2$

```
Z = [ones(size(x) x x.^2]
a = (Z'*Z)\(Z'*y)
```

  – Note also that MATLAB's left-divide will automatically include the $[Z]^T$ terms if the matrix is not square, so

```
a = Z\y
```

  would work as well

- To calculate measures of fit:

```
St = sum((y-mean(y)).^2)
Sr = sum((y-Z*a).^2)
r2 = 1-Sr/St
syx = sqrt(Sr/(length(x)-length(a)))
```

coefficient of determination

standard error

Nonlinear Regression

- As seen in the previous chapter, not all fits are linear equations of coefficients and basis functions, e.g.,

$$y = a_0\left(1 - e^{-a_1 x}\right) + e$$

- One method to handle this is to transform the variables and solve for the best fit of the transformed variables. There are two problems with this method
  - Not all equations can be transformed easily or at all
  - The best fit line represents the best fit for the transformed variables, not the original variables

- Another method is to perform nonlinear regression to directly determine the least-squares fit, e.g.,

$$f\left(a_0, a_1\right)y = \sum_{i=1}^{n}[y_i - a_0(1 - e^{-a_1 x_1})]^2$$

  – Using the MATLAB ***fminsearch*** function

Nonlinear Regression in MATLAB

- To perform nonlinear regression in MATLAB, write a function that returns the sum of the squares of the estimate residuals for a fit and then use MATLAB's `fminsearch` function to find the values of the coefficients where a minimum occurs
- The arguments to the function to compute S_r should be the coefficients, the independent variables, and the dependent variables

Nonlinear Regression in MATLAB Example

- Given dependent force data F for independent velocity data v, determine the coefficients for the fit:

$$F = a_0 v^{a_1}$$

- First - write a function called `fSSR.m` containing the following:

```
function f = fSSR(a, xm, ym)
yp = a(1)*xm.^a(2);
f = sum((ym-yp).^2);
```

- Then, use fminsearch in the command window to obtain the values of a that minimize `fSSR`:

```
a = fminsearch(@fSSR, [1, 1], [], v, F)
```

where [1, 1] is an initial guess for the [a0, a1] vector, [] is a placeholder for the options