Engineering Analysis ENG 3420 Fall 2009

Dan C. Marinescu
Office: HEC 439 B
Office hours: Tu-Th 11:00-12:00
Lecture 21

- Last time:
  - Relaxation
  - Non-linear systems
  - Random variables, probability distributions, Matlab support for random variables

- Today
  - Histograms
  - Linear regression
  - Linear least squares regression
  - Non-linear data models

- Next Time
  - Multiple linear regression
  - General linear squares
Statistics built-in functions

- Built-in statistics functions for a column vector $s$:
  - mean($s$), median($s$), mode($s$)
    - Calculate the mean, median, and mode of $s$. mode is a part of the statistics toolbox.
  - min($s$), max($s$)
    - Calculate the minimum and maximum value in $s$.
  - var($s$), std($s$)
    - Calculate the variance and standard deviation of $s$

- If a matrix is given, the statistics will be returned for each column.
Histories

- \([n, x] = \text{hist}(s, x)\)
  - Determine the number of elements in each bin of data in \(s\).
  - \(x\) is a vector containing the center values of the bins.
- \([n, x] = \text{hist}(s, m)\)
  - Determine the number of elements in each bin of data in \(s\) using \(m\) bins.
  - \(x\) will contain the centers of the bins.
  - The default case is \(m=10\)
- \(\text{hist}(s, x)\) or \(\text{hist}(s, m)\) or \(\text{hist}(s)\)
  - With no output arguments, \(\text{hist}\) will actually produce a histogram.
Histogram Example
Linear Least-Squares Regression

- Linear least-squares regression is a method to determine the “best” coefficients in a linear model for given data set.
- “Best” for least-squares regression means minimizing the sum of the squares of the estimate residuals. For a straight line model, this gives:
  \[ S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i)^2 \]
- This method will yield a unique line for a given set of data.
Least-Squares Fit of a Straight Line

- Using the model:
  \[ y = a_0 + a_1 x \]

  the slope and intercept producing the best fit can be found using:

  \[
  a_1 = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{n \sum x_i^2 - (\sum x_i)^2}
  \]

  \[
  a_0 = \bar{y} - a_1 \bar{x}
  \]
Example

<table>
<thead>
<tr>
<th>i</th>
<th>x&lt;sub&gt;i&lt;/sub&gt;</th>
<th>y&lt;sub&gt;i&lt;/sub&gt;</th>
<th>(x&lt;sub&gt;i&lt;/sub&gt;)&lt;sup&gt;2&lt;/sup&gt;</th>
<th>x&lt;sub&gt;i&lt;/sub&gt;y&lt;sub&gt;i&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>25</td>
<td>100</td>
<td>250</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>70</td>
<td>400</td>
<td>1400</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>380</td>
<td>900</td>
<td>11400</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>550</td>
<td>1600</td>
<td>22000</td>
</tr>
<tr>
<td>5</td>
<td>50</td>
<td>610</td>
<td>2500</td>
<td>30500</td>
</tr>
<tr>
<td>6</td>
<td>60</td>
<td>1220</td>
<td>3600</td>
<td>73200</td>
</tr>
<tr>
<td>7</td>
<td>70</td>
<td>830</td>
<td>4900</td>
<td>58100</td>
</tr>
<tr>
<td>8</td>
<td>80</td>
<td>1450</td>
<td>6400</td>
<td>116000</td>
</tr>
<tr>
<td>Σ</td>
<td>360</td>
<td>5135</td>
<td>20400</td>
<td>312850</td>
</tr>
</tbody>
</table>

\[ a_1 = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{n \sum x_i^2 - (\sum x_i)^2} = \frac{8(312850) - (360)(5135)}{8(20400) - (360)^2} = 19.47024 \]

\[ a_0 = \bar{y} - a_1 \bar{x} = 641.875 - 19.47024(45) = -234.2857 \]

\[ F_{est} = -234.2857 + 19.47024v \]
Nonlinear models

- Linear regression is predicated on the fact that the relationship between the dependent and independent variables is linear - this is not always the case.

- Three common examples are:

  exponential: \[ y = \alpha_1 e^{\beta_1 x} \]

  power: \[ y = \alpha_2 x^{\beta_2} \]

  saturation - growth - rate: \[ y = \alpha_3 \frac{x}{\beta_3 + x} \]
# Linearization of nonlinear models

<table>
<thead>
<tr>
<th>Model</th>
<th>Nonlinear</th>
<th>Linearized</th>
</tr>
</thead>
<tbody>
<tr>
<td>exponential:</td>
<td>$y = \alpha_1 e^{\beta_1 x}$</td>
<td>$\ln y = \ln \alpha_1 + \beta_1 x$</td>
</tr>
<tr>
<td>power:</td>
<td>$y = \alpha_2 x^{\beta_2}$</td>
<td>$\log y = \log \alpha_2 + \beta_2 \log x$</td>
</tr>
<tr>
<td>saturation-growth-rate:</td>
<td>$y = \frac{x}{\beta_3 + x}$</td>
<td>$\frac{1}{y} = \frac{1}{\alpha_3} + \frac{\beta_3}{\alpha_3} \frac{1}{x}$</td>
</tr>
</tbody>
</table>
Transformation Examples

\( y = \alpha_1 e^{\beta_1 x} \)  
\( y = \alpha_2 x^{\beta_2} \)  
\( y = \frac{\alpha_3 x}{\beta_3 + x} \)

\( \ln y \)  
\( \log y \)  
\( 1/y \)

Slope = \( \beta_1 \)  
Slope = \( \beta_2 \)  
Slope = \( \beta_3 / \alpha_3 \)

Intercept = \( \ln \alpha_1 \)  
Intercept = \( \log \alpha_2 \)  
Intercept = \( 1/\alpha_3 \)
function [a, r2] = linregr(x,y)

% linregr: linear regression curve fitting
% [a, r2] = linregr(x,y); Least squares fit of straight
% line to data by solving the normal equations

% input:
% x = independent variable
% y = dependent variable
% output:
% a = vector of slope, a(1), and intercept, a(2)
% r2 = coefficient of determination

n = length(x);
if length(y)~=n, error('x and y must be same length'); end
x = x(:); y = y(:); % convert to column vectors
sx = sum(x); sy = sum(y);
sx2 = sum(x.*x); sxy = sum(x.*y); sy2 = sum(y.*y);
a(1) = (n*sxy-sx*sy)/(n*sx2-sx^2);
a(2) = sy/n-a(1)*sx/n;
r2 = ((n*sxy-sx*sy)/sqrt(n*sx2-sx^2)/sqrt(n*sy2-sy^2))^2;
% create plot of data and best fit line
xp = linspace(min(x),max(x),2);  
yp = a(1)*xp+a(2);  
plot(x,y,'o',xp,yp)
grid on
Polynomial least-fit squares

- MATLAB has a built-in function `polyfit` that fits a least-squares n-th order polynomial to data:
  - \( p = \text{polyfit}(x, y, n) \)
    - \( x \): independent data
    - \( y \): dependent data
    - \( n \): order of polynomial to fit
    - \( p \): coefficients of polynomial
      \[ f(x) = p_1 x^n + p_2 x^{n-1} + \ldots + p_n x + p_{n+1} \]

- MATLAB’s `polyval` command can be used to compute a value using the coefficients.
  - \( y = \text{polyval}(p, x) \)
Polynomial Regression

- The least-squares procedure from can be extended to fit data to a higher-order polynomial. 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:

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

- In general, this would mean minimizing:

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

- 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:

\[ r^2 = \frac{S_t - S_r}{S_t} \]