

Why do 'Curve Fitting' ?

- The main purpose of 'curve fitting' is to establish a mathematical relation between x (input) and y (output).
- This relation usually is determined by performing a calibration in which y 's are measured for known x 's. The resulting calibration relation has an uncertainty.
- Often the calibration relation subsequently is used to determine an unknown x from a measured y .
- This type of use introduces an additional uncertainty.

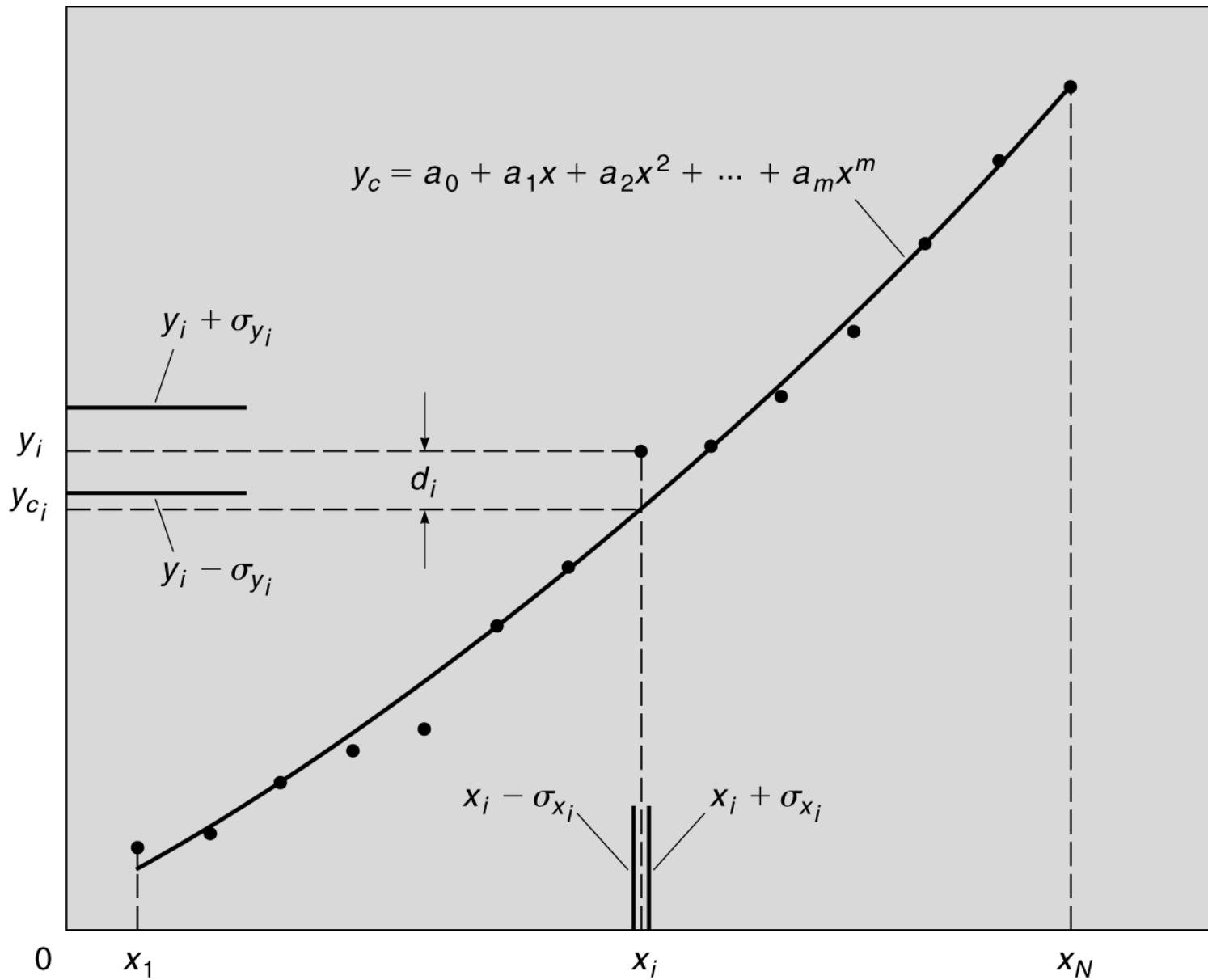
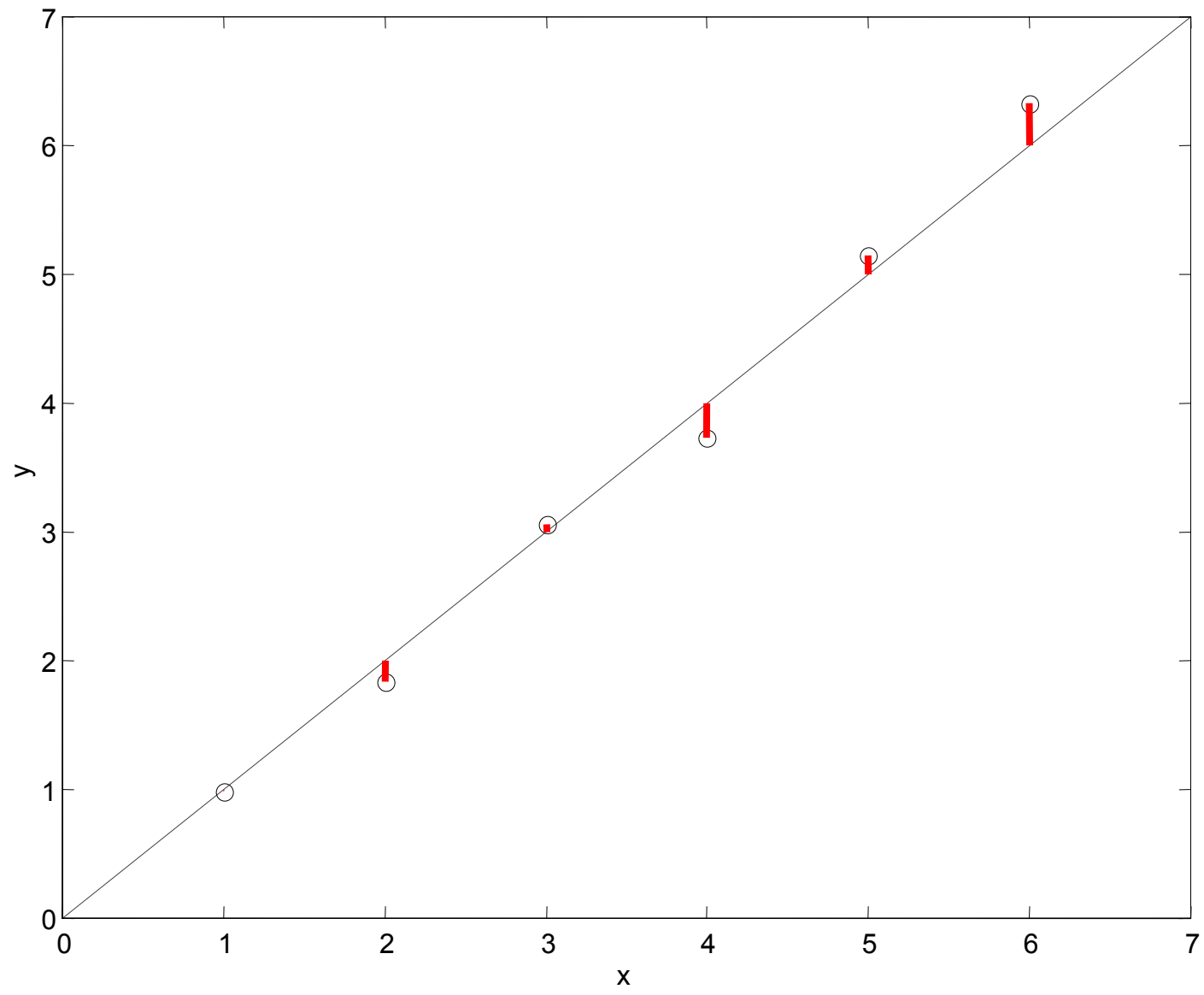


Figure 10.1



Least-Squares Regression Analysis (LSRA)

$$y_c = a_0 + a_1x + a_2x^2 + \dots + a_mx^m$$

$$D = \sum_{i=1}^N d_i^2 = \sum_{i=1}^N (y_i - y_{c_i})^2 = \sum_{i=1}^N (y_i - \{a_0 + a_1x_i + \dots + a_mx_i^m\})^2$$

$$dD = 0 = \frac{\partial D}{\partial a_0} da_0 + \frac{\partial D}{\partial a_1} da_1 + \dots + \frac{\partial D}{\partial a_m} da_m.$$

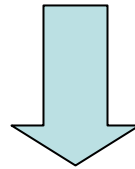
Linear Least-Squares Regression Analysis

$$y_c = a_0 + a_1x$$

Linear Least-Squares Regression Analysis

$$\bar{y} = a_0 + a_1 \bar{x}$$

$$\overline{xy} = a_0 \bar{x} + a_1 \overline{x^2}$$



$$a_0 = \left(\sum_{i=1}^N x_i^2 \sum_{i=1}^N y_i - \sum_{i=1}^N x_i \sum_{i=1}^N x_i y_i \right) / \Delta$$

$$a_1 = \left(N \sum_{i=1}^N x_i y_i - \sum_{i=1}^N x_i \sum_{i=1}^N y_i \right) / \Delta$$

$$\Delta = N \sum_{i=1}^N x_i^2 - \left[\sum_{i=1}^N x_i \right]^2$$

Standard Error of the Fit

- The quality of the curve fit is characterized by the standard error of the fit, S_{yx}

- The estimated value of y_i for a specified x is given by

plotfit.m

**** only some parts are shown ****

```
% this m-file reads a three-column data file of x, y, y-error,  
% plots the data with y-error bars, performs an m-th order regression analysis,  
% then plots the regression fit with plus, minus %P fits
```

```
clear all;
```

```
% the input data file is read
```

```
filename = input('enter the filename w/o its extension: ','s');
```

```
ext = input('enter the files extension, e.g., dat: ','s');
```

```
eval(['load ',filename,','.ext]);
```

```
x = eval([filename,':,1']);
```

```
y = eval([filename,':,2']);
```

```
ey = eval([filename,':,3']);
```

```
% the order of the regression fit is set
```

```
m = input('enter the order of the regression fit, e.g., m=1 linear, m = ');
```

```
% the x,y data is fit and evaluated using an m-th order regression
```

```
p = polyfit(x,y,m);
```

```
f = polyval(p,x);
```

```

% the standard error of the fit is calculated
ydiff = y-f;
xmax = max(x);
xx = (0:xmax/100:xmax); %100 is arbitrary factor to give smooth plot of the fit
g = polyval(p,xx);
[r,c] = size(x);
nu = r-m-1; % degrees of freedom
%warning message when nu < 1
    if nu<1
        disp(' ')
        disp('WARNING: degrees of freedom less than one >> invalid results !!!')
    end
syx = sqrt(1/nu).*norm(ydiff,2);

```

```

% the percent confidence is set
P = input('enter the percent confidence of the fit: P(%) = ');

% compute the student-t factor
tnup = tinv((1+.01*P)/2,nu);

% compute the precision interval of the fit
preint = tnup*syx;

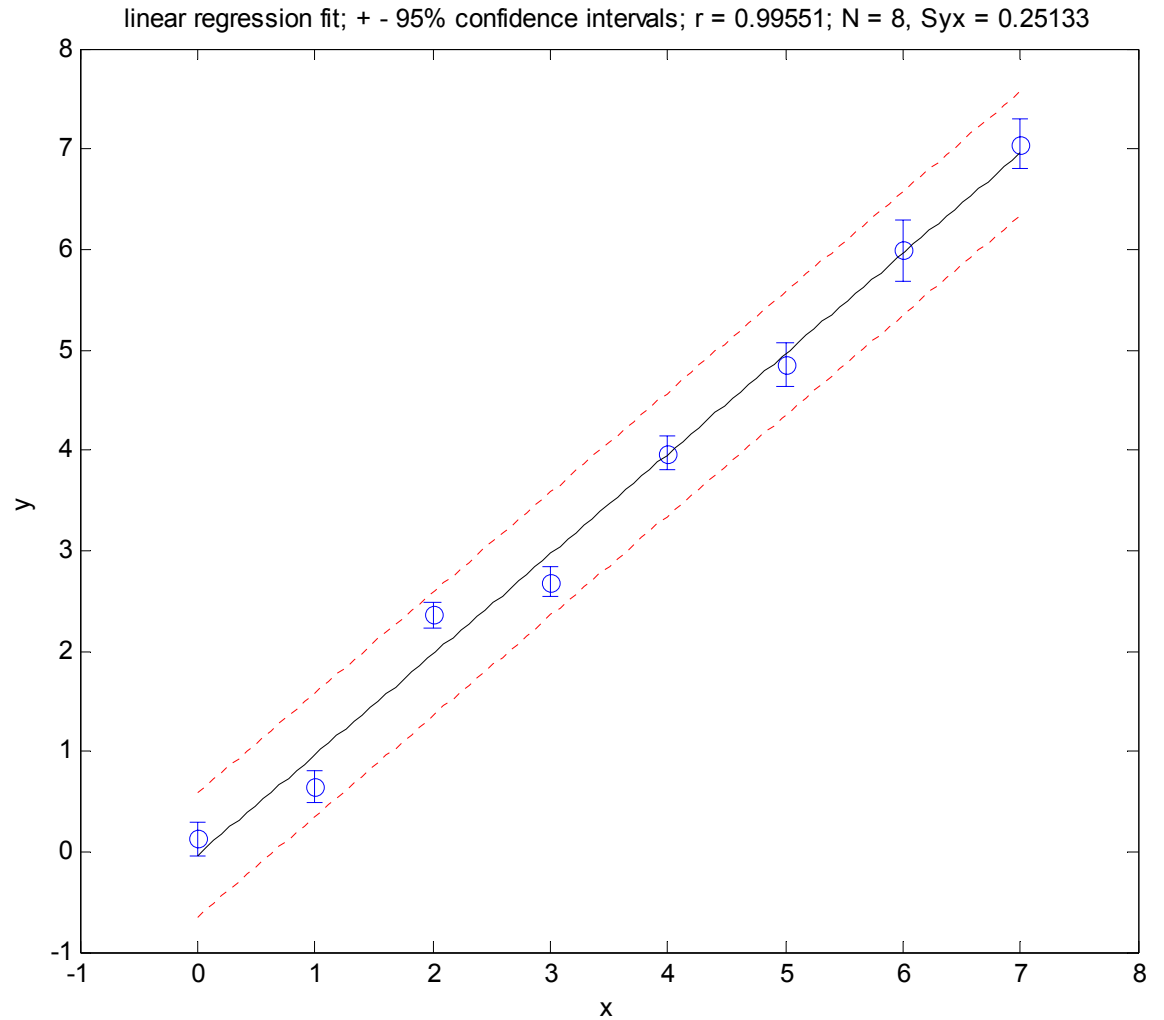
% construct fits at plus, minus %P confidence
gu = g+preint;
gl = g-preint;

% the plot is constructed
errorbar(x,y,ey,'o');
xlabel('x'); ylabel('y');

```

Using plotfit.m

x	y	ey
0	0.13	0.17
1	0.65	0.15
2	2.36	0.13
3	2.69	0.14
4	3.97	0.17
5	4.85	0.21
6	5.99	0.31
7	7.05	0.25



x-from-y Estimate Uncertainty

- When x is estimated from y , additional uncertainty arises because of the finite amount of data. This is because the intercept and slope are slightly different for each finite set of data.

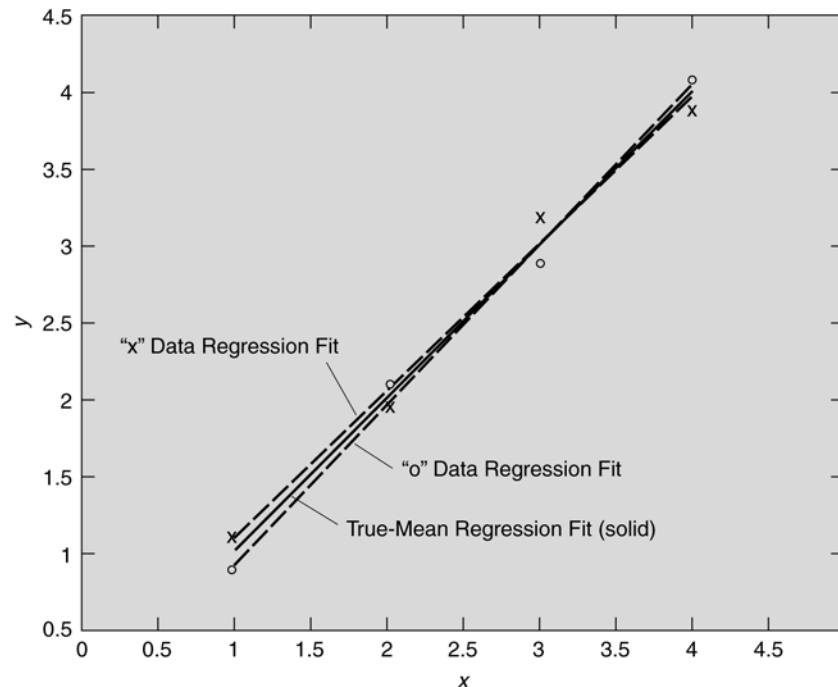
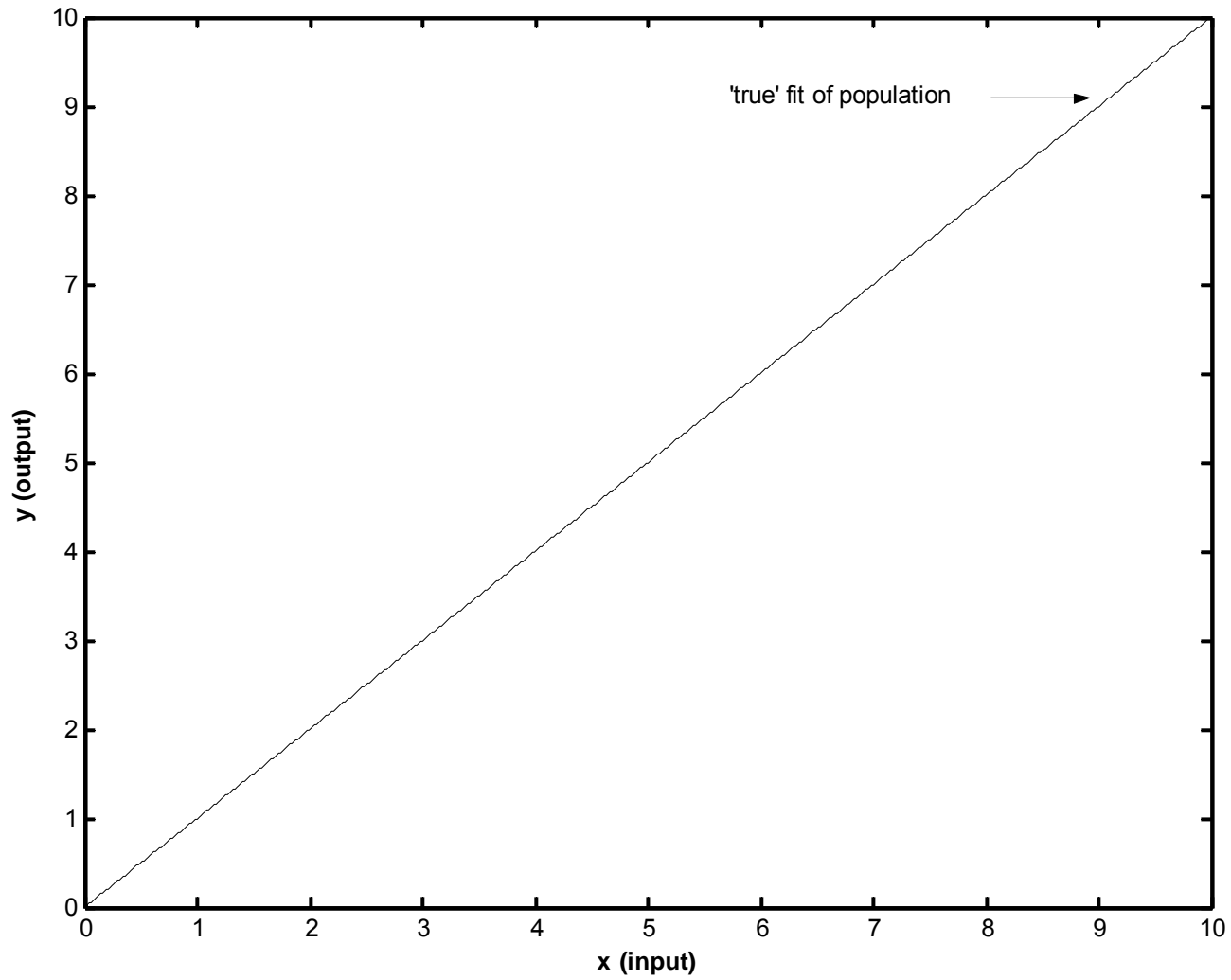
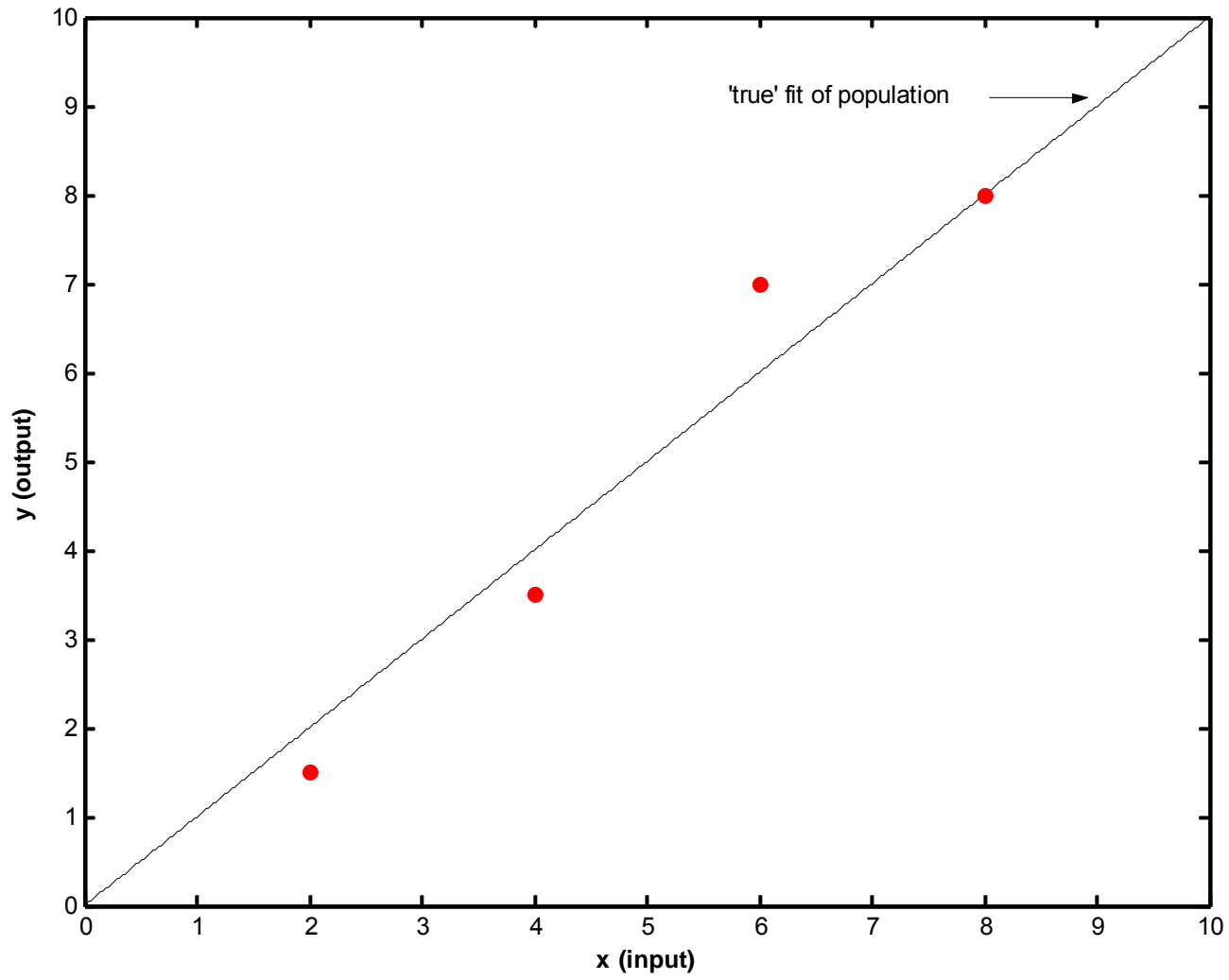
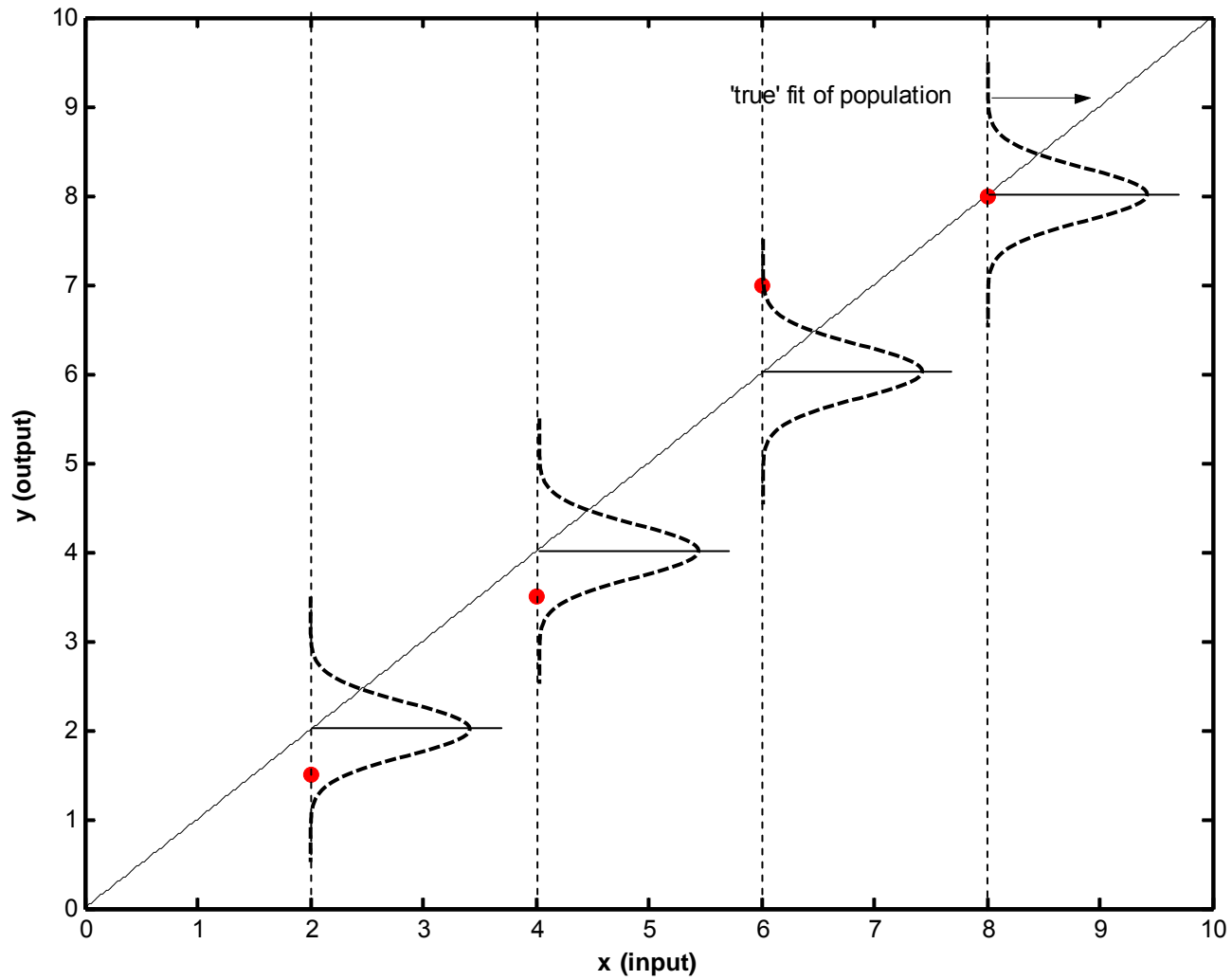
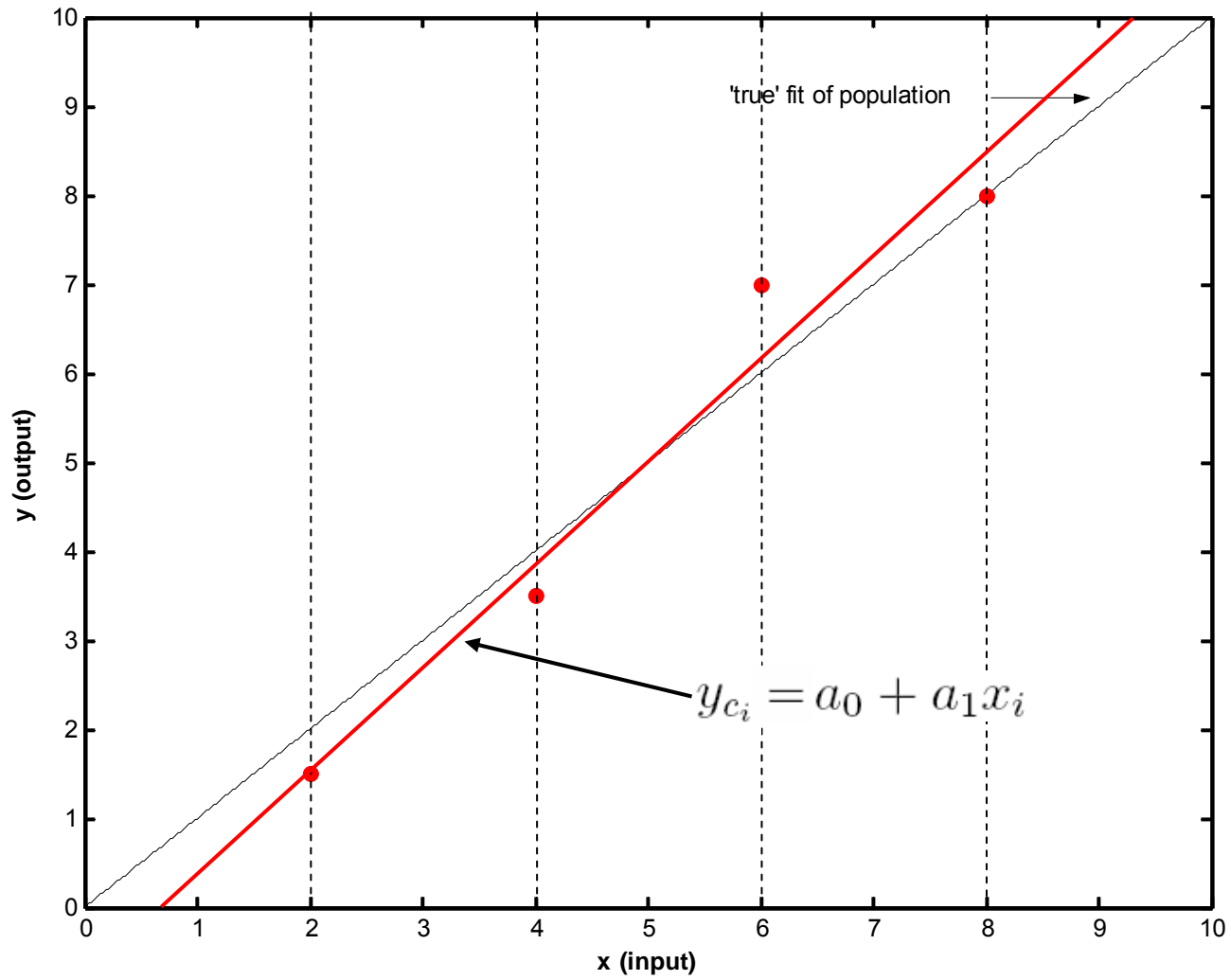


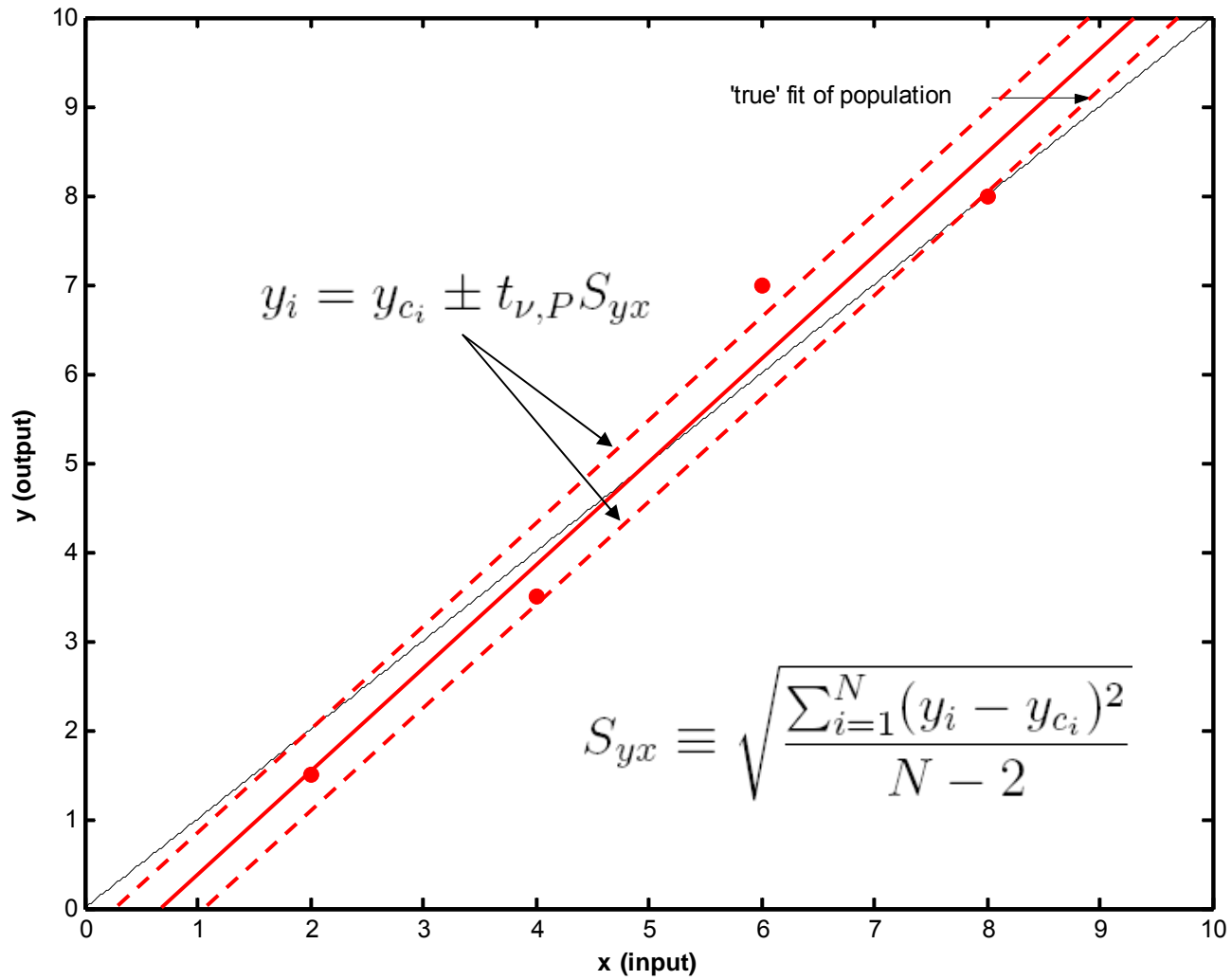
Figure 10.7

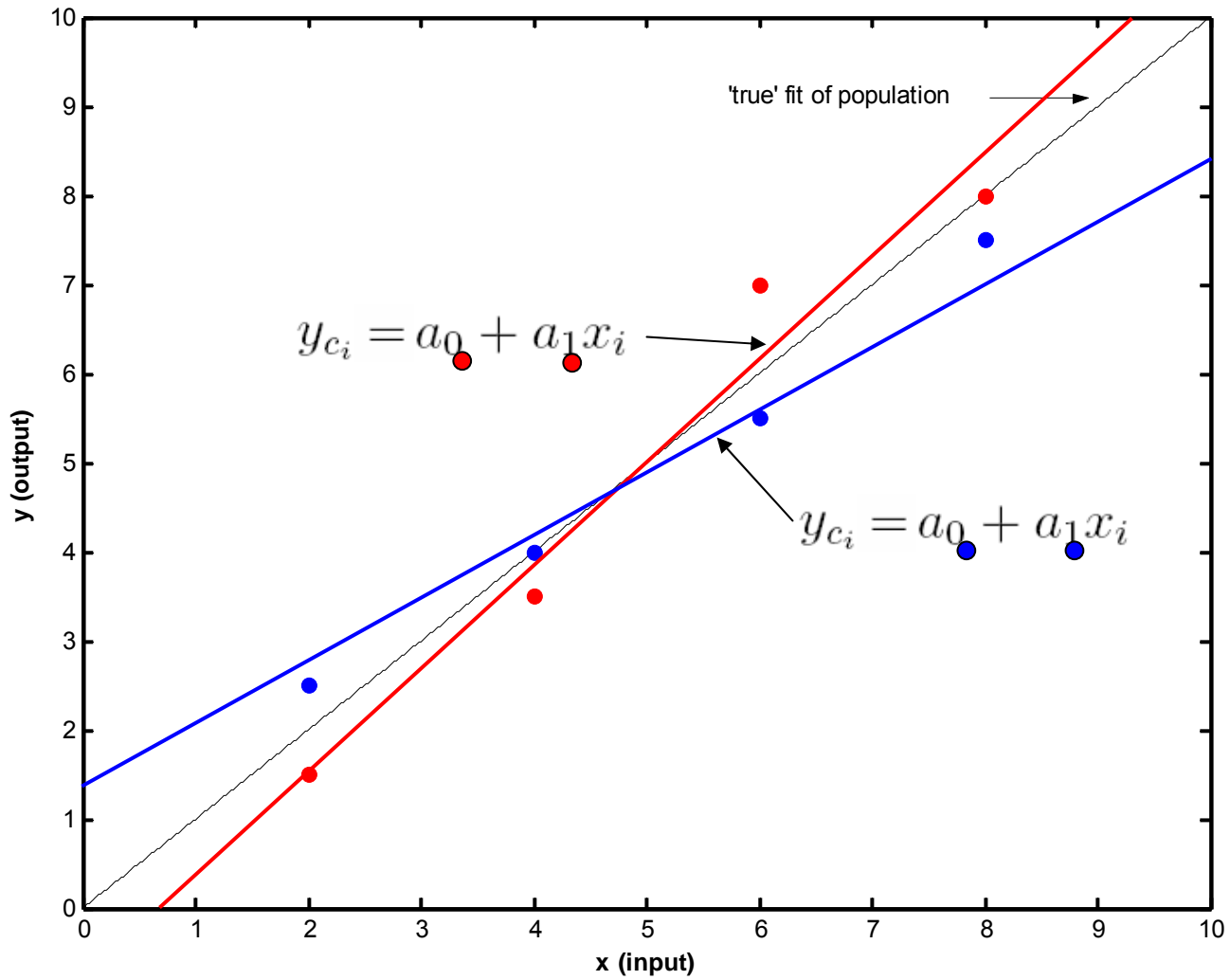


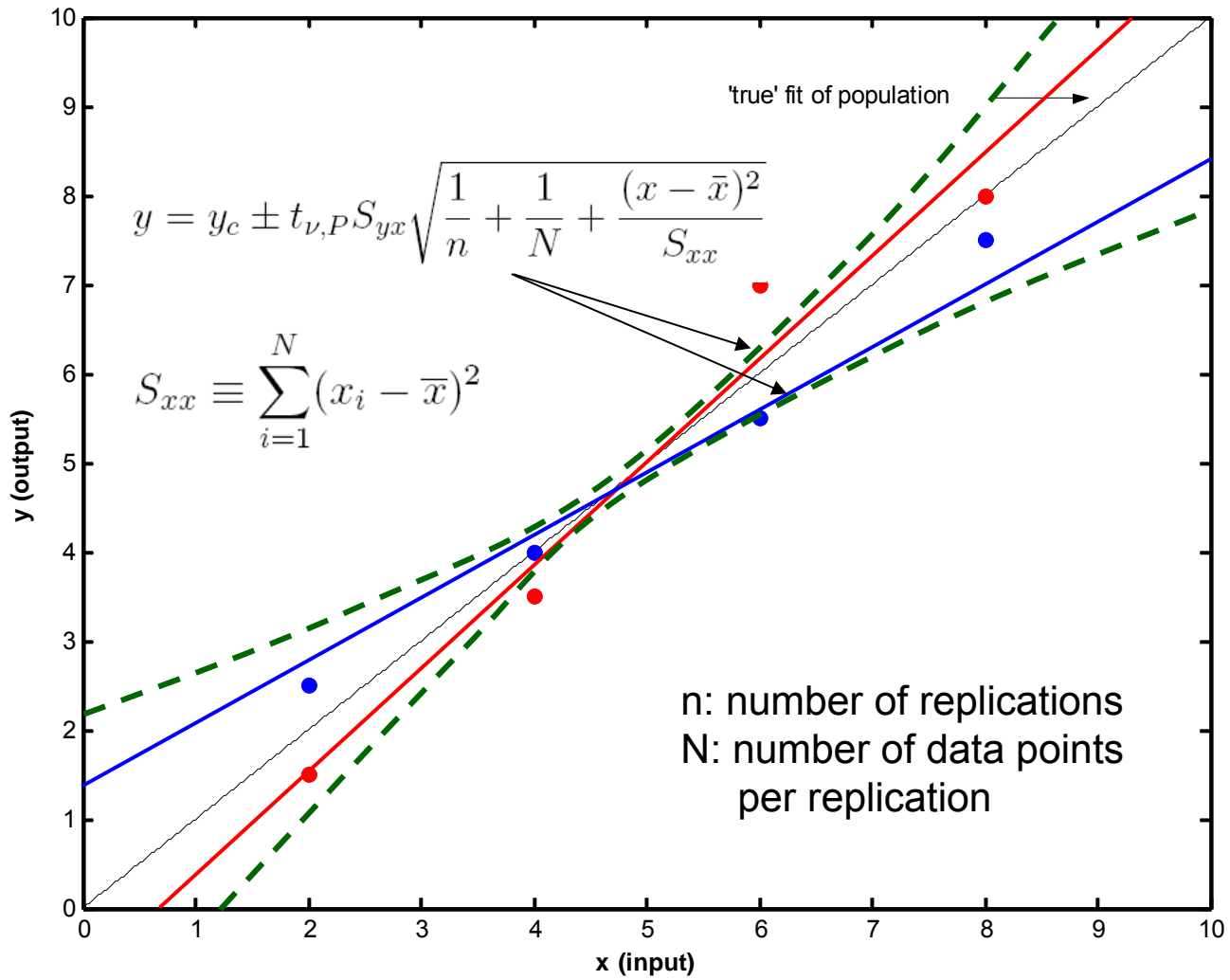












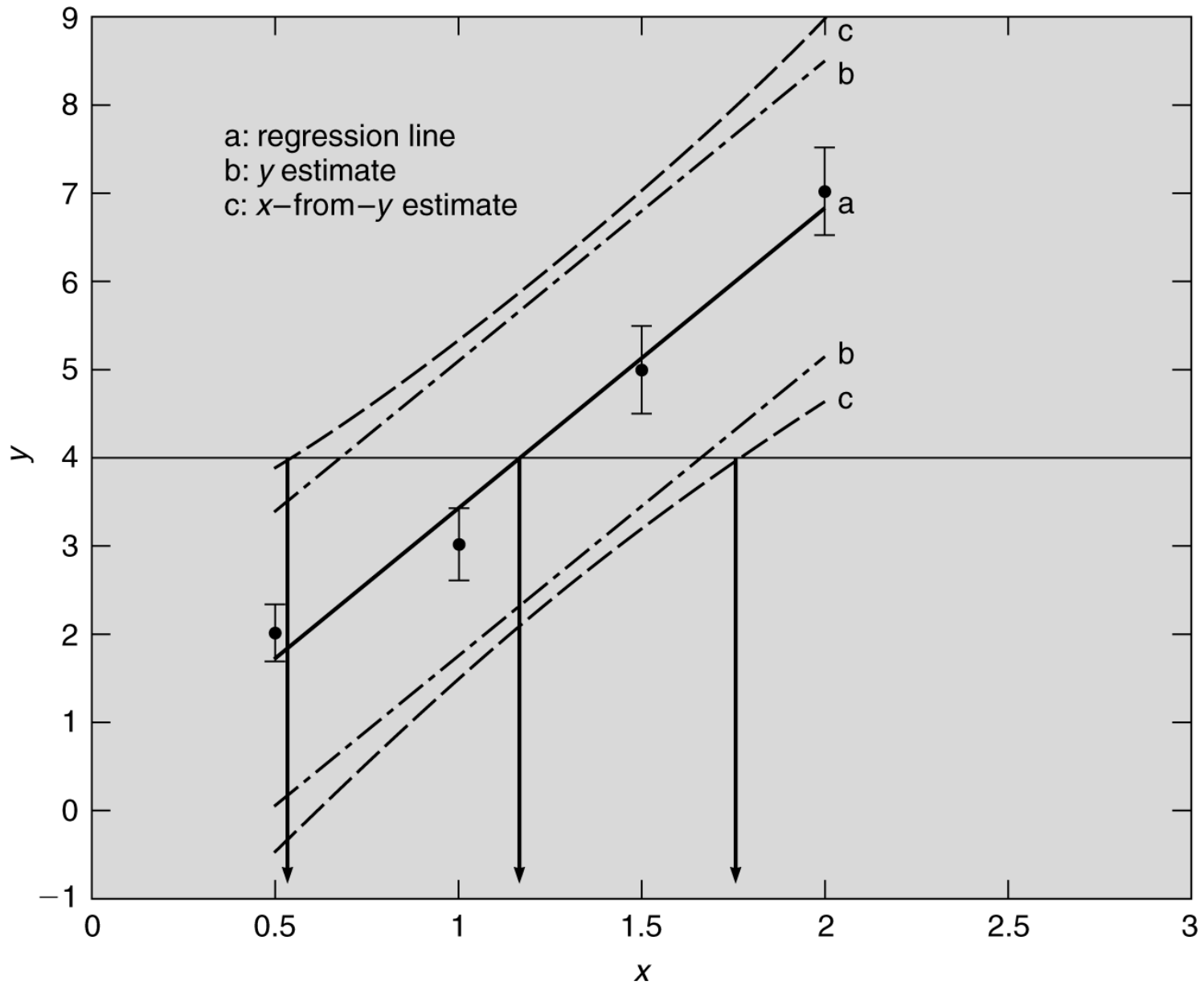
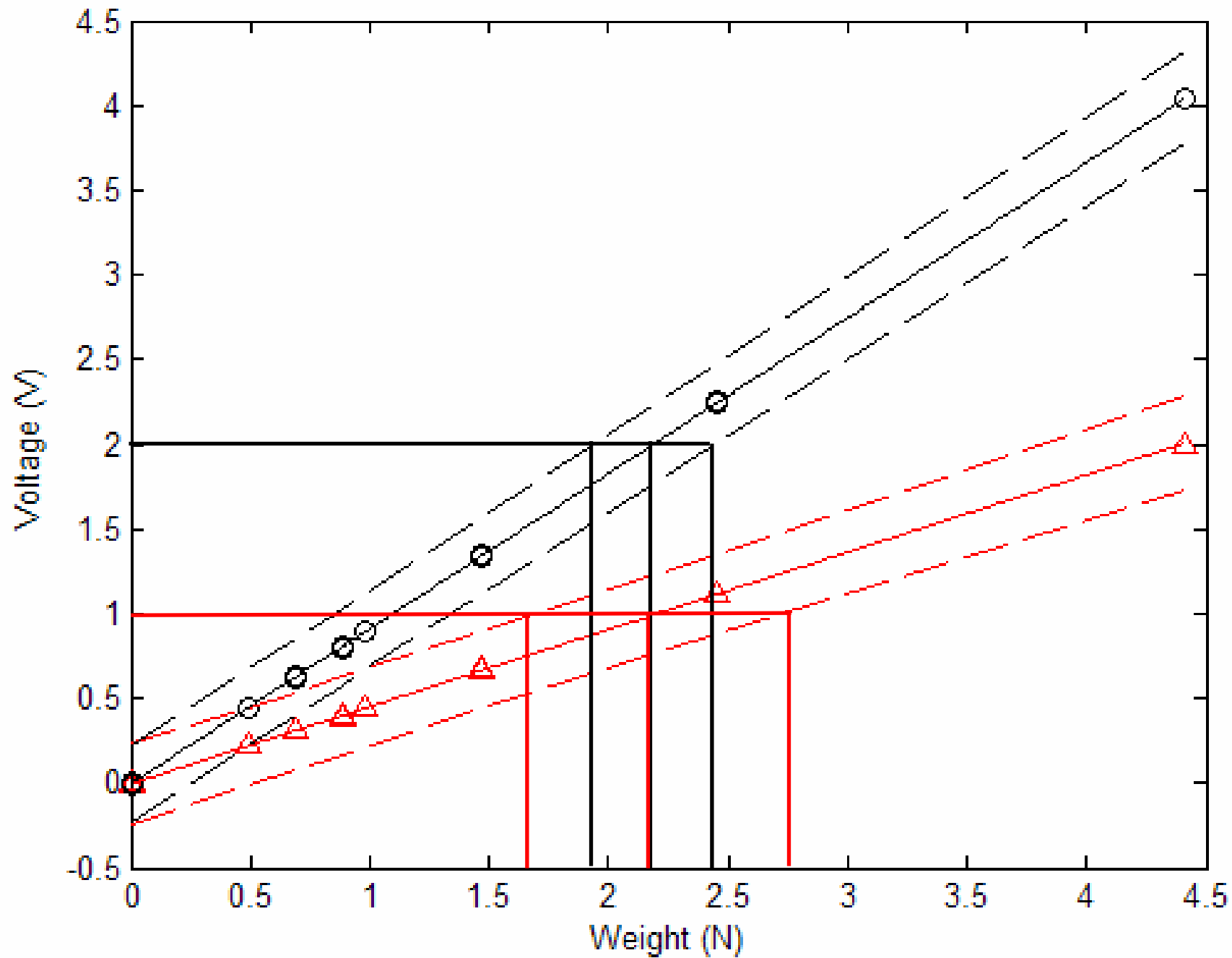


Figure 10.9

Calibration Plots



Putting it all in one m-file

- The m-file *jcaleyIII.m* (found under Text Information on course web site) was written for this purpose.
- The m-file uses a 3-column text file of x, y, y-error.

$P = 95 \%$; $N = 8$; $n = 1$; $S_{yx} = 0.25133$; $u_{int} = \pm 0.39698$; $u_{slo} = \pm 0.094896$

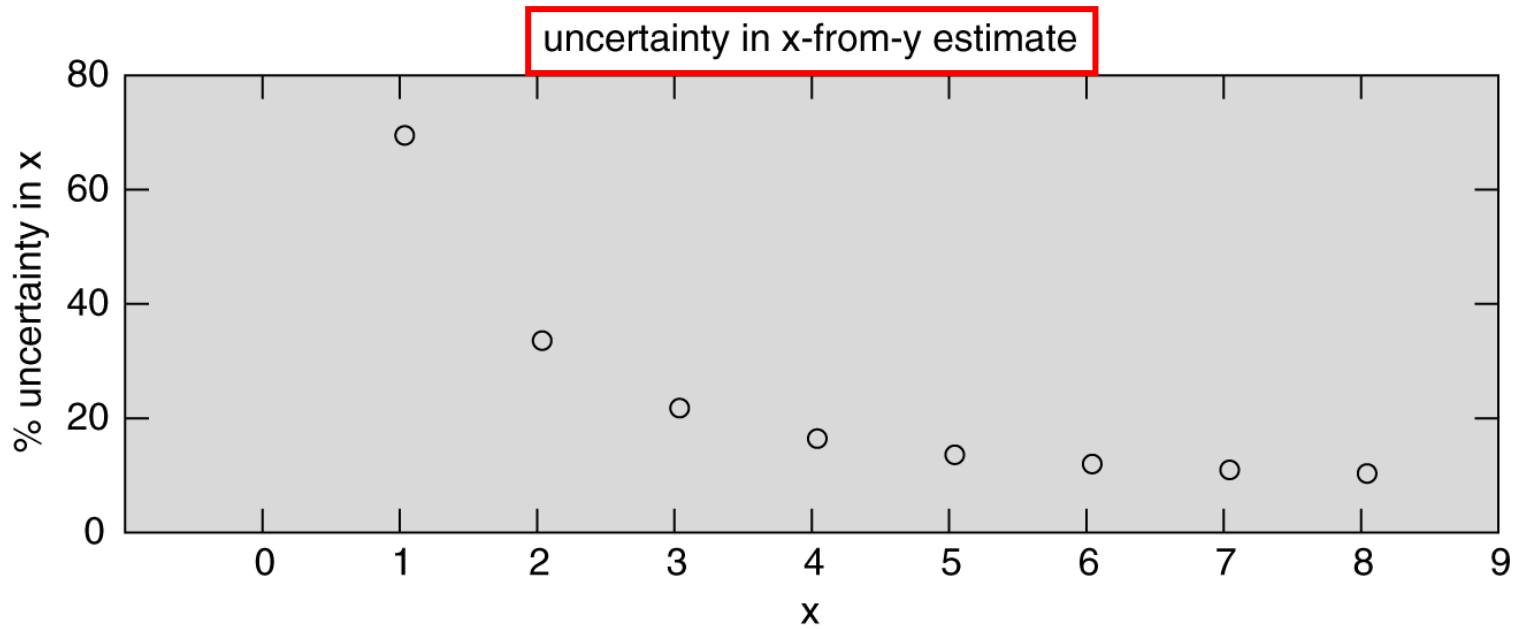
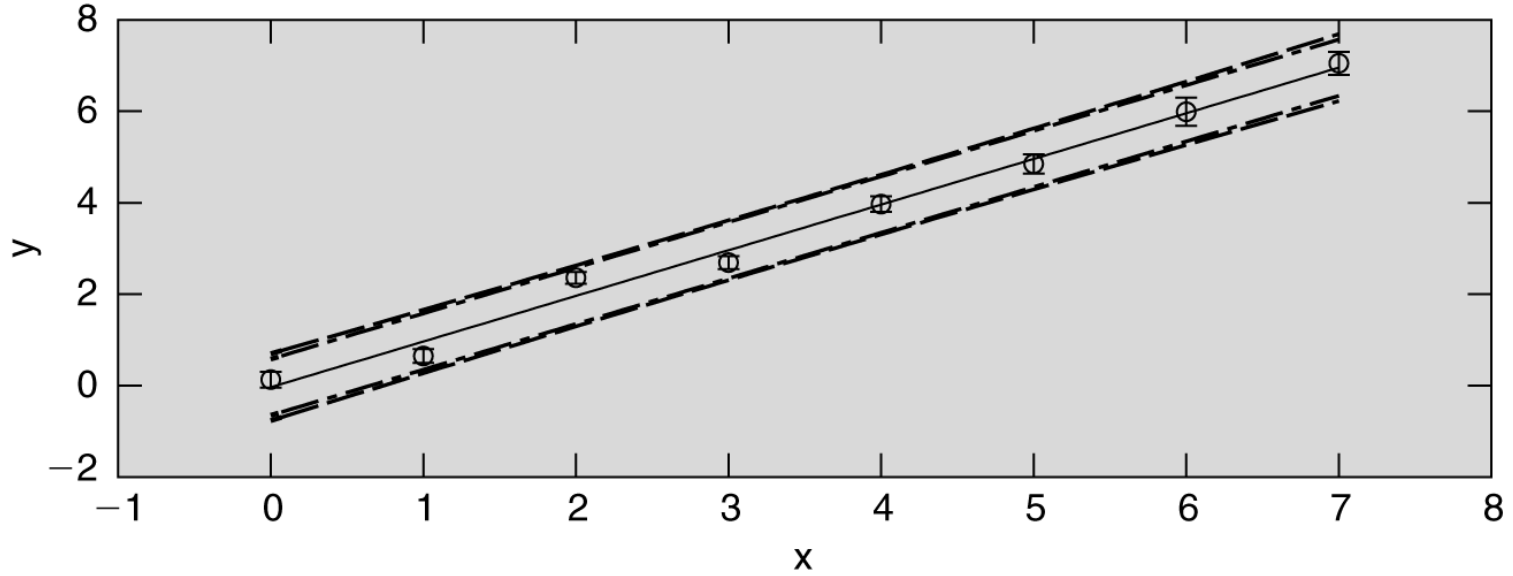


Figure 10.12

What happens if the relationship is not linear?

- You can use higher-order regression analysis. This involves inverting larger matrices – see section 10.6.
- For some situations, the non-linear expression can be converted into a linear one, if the variables are *intrinsically linear*.

- Example:

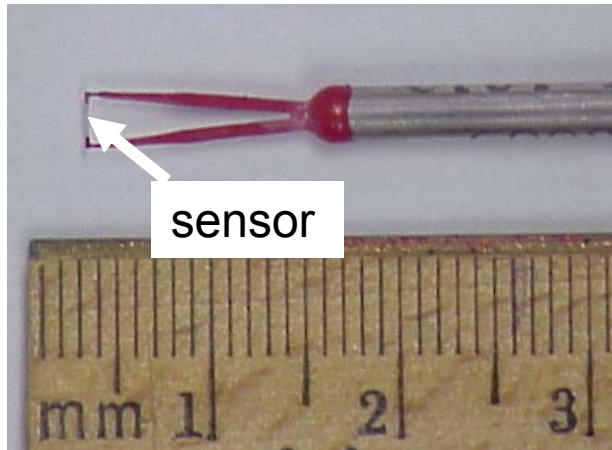
$$y = a x^b$$



Example: Fitting Hot-Wire Data

- Hot-wire anemometry can be used to measure local flow velocity. The relationship between the flow velocity, U , and the anemometer circuit's voltage, E , is given by King's Law (a form of the conservation of energy).

- King's Law: $E^2 = A + BU^n$.



- For the following data, determine A , B and n using LSLRA.

Velocity (m/s)	Voltage (V)
0.00	3.19
3.05	3.99
6.10	4.30
9.14	4.48
12.20	4.65

Example: Fitting Hot-Wire Data

- King's Law can be transformed into linear intrinsic variables, where

$$\log(E^2 - A) = \log(B) + n \log(U)$$

$$E^* = b + nU^*$$

$$\text{where } E^* = \log(E^2 - A)$$

$$b = \log(B)$$

$$U^* = \log(U)$$

- Using the MATLAB command:

```
[slope,intercept]=polyfit(ustar,estar,1)
```

gives slope (n) = 0.494 and intercept (b) = 0.523 ($\rightarrow B = 10^b = 3.33$).

Example: Fitting Hot-Wire Data

So the best fit equation is: $E^2 = 3.19^2 + 3.33U^{0.494}$

