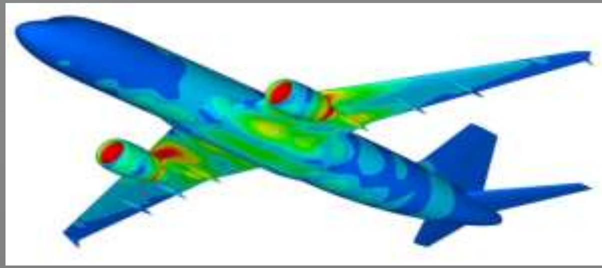
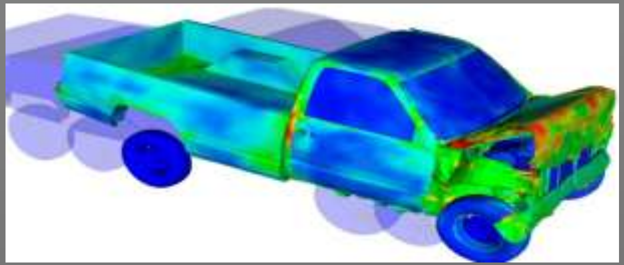
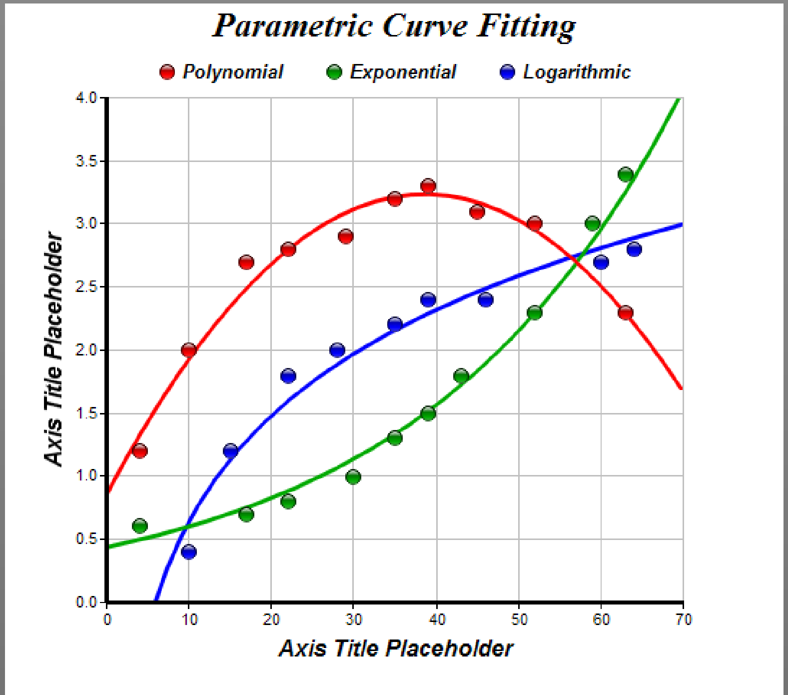
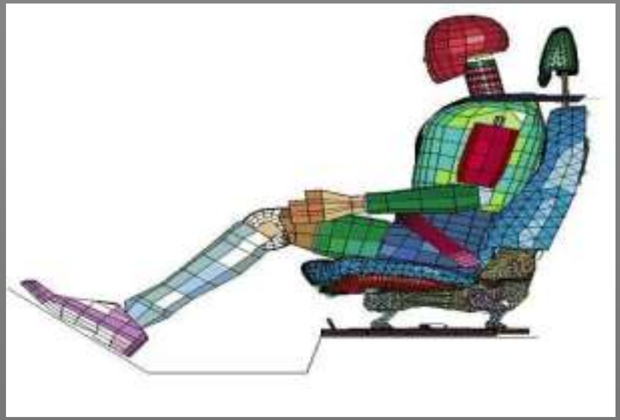
Curve Fitting









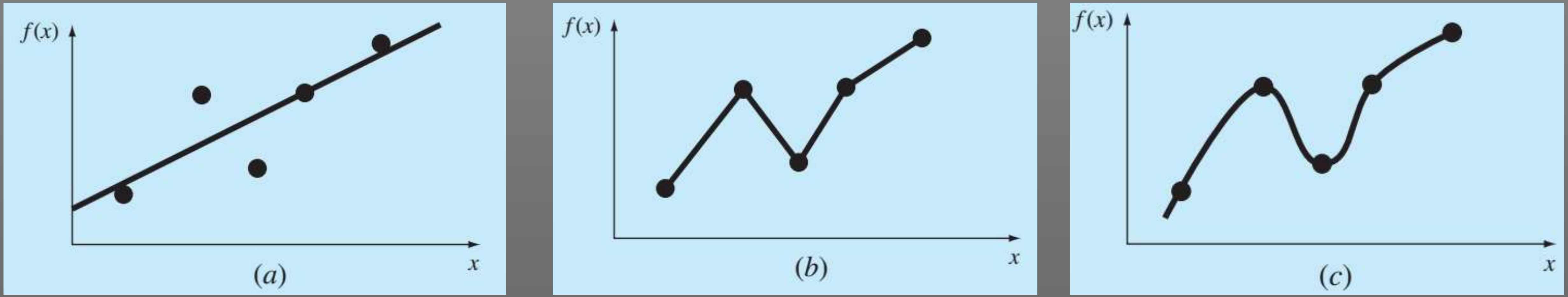
Dr. Himanshu Pathak
himanshu@iitmandi.ac.in

Curve Fitting

- **It is the process of constructing a curve, or mathematical function, that has the best fit to a series of data points, possibly subject to constraints.**
- **It is a statistical technique use to drive coefficient values for equations that express the value of one(dependent) variable as a function of another (independent variable)**

Regression: Data exhibit a significant degree of scatter. The strategy is to derive a single curve that represents the general trend of the data.

Interpolation: Data is very precise. The strategy is to pass a curve or a series of curves through each of the points.



Least square Regression

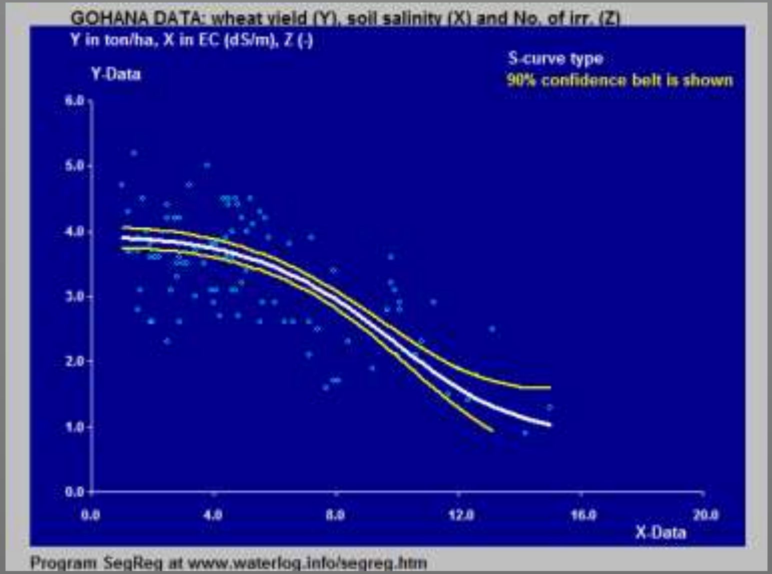
Linear interpolation

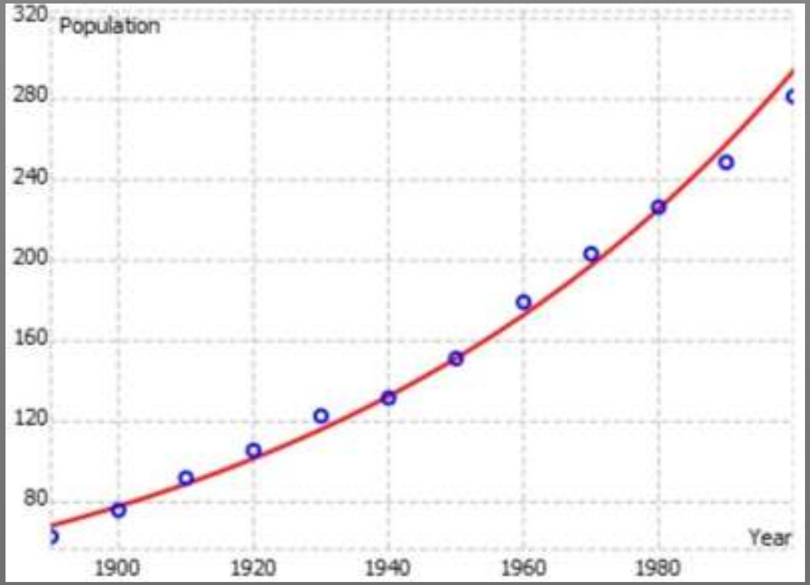
Curvilinear interpolation

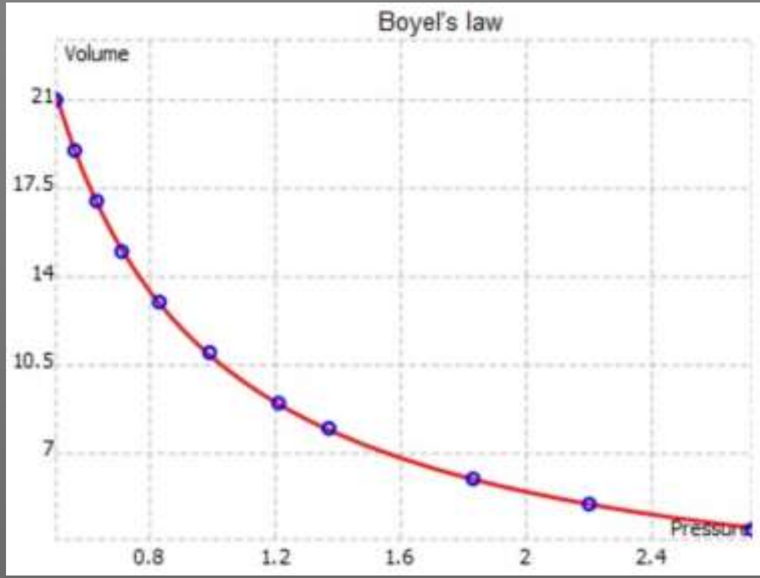
Curve Fitting

In engineering, two types of applications are encountered:

1. Trend analysis: extrapolating data model to predict future behaviour.
2. Hypothesis testing: comparing/testing measured data with existing mathematical model.



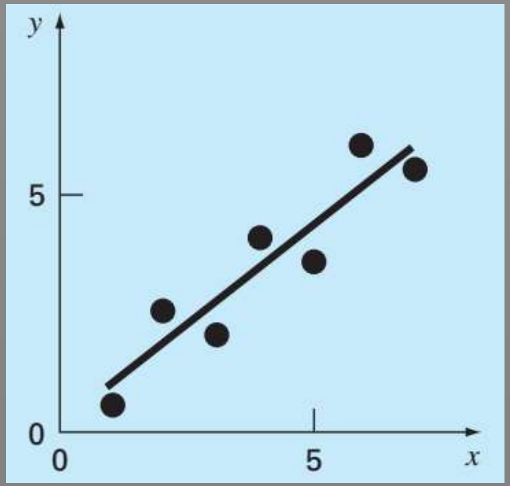




Linear Regression

- A simplest approach to fitting a straight line to a set of data using least-square approximation.

$$y = a_0 + a_1 x + e$$



Objective is to find coefficient/equation with minimum residual

$$e = y - a_0 - a_1 x$$

For bets fit curve:

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

Using least-square:

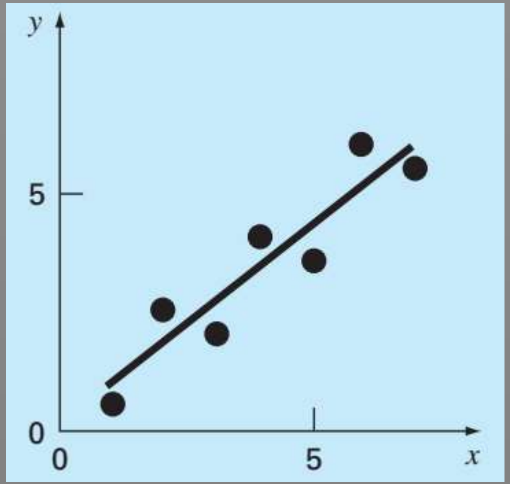
$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_{i,measured} - y_{i,model} \right)^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i \right)^2$$

Linear Regression

To determine coefficients:

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

$$\frac{\partial S_r}{\partial a_1} = -2\sum_{i=1}^{n}\left[\left(y_i - a_0 - a_1 x_i\right)x_i\right]$$



$$\sum_{i=1}^{n} y_i - \sum_{i=1}^{n} a_0 - \sum_{i=1}^{n} a_1 x_i = 0$$

$$\sum_{i=1}^{n} y_i x_i - \sum_{i=1}^{n} a_0 x_i - \sum_{i=1}^{n} a_1 x_i^2 = 0$$

$$\boxed{\begin{aligned} a_1 &= \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2} \\ a_0 &= \bar{y} - a_1 \bar{x} \end{aligned}}$$

Linear Regression

Example: Fit a straight line for given data.

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

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

x_i	y_i
1	0.5
2	2.5
3	2.0
4	4.0
5	3.5
6	6.0
7	5.5
Σ	24.0

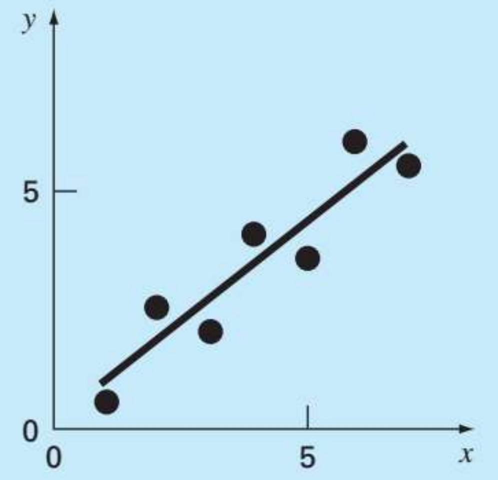
$n = 7$

$\sum x_i y_i = 119.5; \quad \sum x_i^2 = 140; \quad \sum x_i = 28; \quad \bar{x} = 7; \quad \sum y_i = 24; \quad \bar{y} = 3.42$

$a_1 = 0.8392$

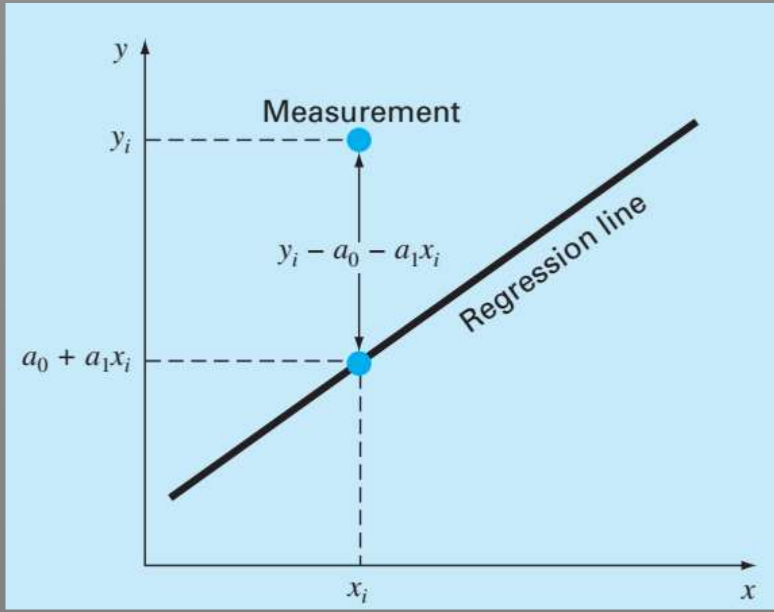
$a_0 = 0.0714$

Best fit line: $y = 0.0714 + 0.8392x$



Error in Linear Regression

Standard Deviation of data $$S_y = \sqrt{\frac{\sum\left(y_i - \bar{y}\right)}{n-1}}$$



Standard Error $$S_{y/x} = \sqrt{\frac{\sum\left(y_i - a_0 - a_1x_i\right)^2}{n-2}}$$

If $S_{y/x} < S_y$ **fit line is acceptable.**

Correlation coefficient $$r = \frac{n\sum x_i y_i - \left(\sum x_i\right)\left(\sum y_i\right)}{\left(\sqrt{n\sum x_i^2 - \left(\sum x_i\right)^2}\right)\left(\sqrt{n\sum y_i^2 - \left(\sum y_i\right)^2}\right)}$$

If $r = 1$ **fit line will have zero percent error (ideal case).**

If $r = 0$ **fit line need to improve.**

Linear Regression

Step 1: Input of data set (x_i, y_i).

Step 2: Estimate straight line coefficients

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

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

Step 3: Estimate standard deviation of data set S_y, standard error of curve fit $S_{y/x}$:

(a) If $S_{y/x} < S_y$, the curve fit is acceptable.

Else, straight line fit is not compatible.

(b) Find correlation coefficient r to show confidence of curve.

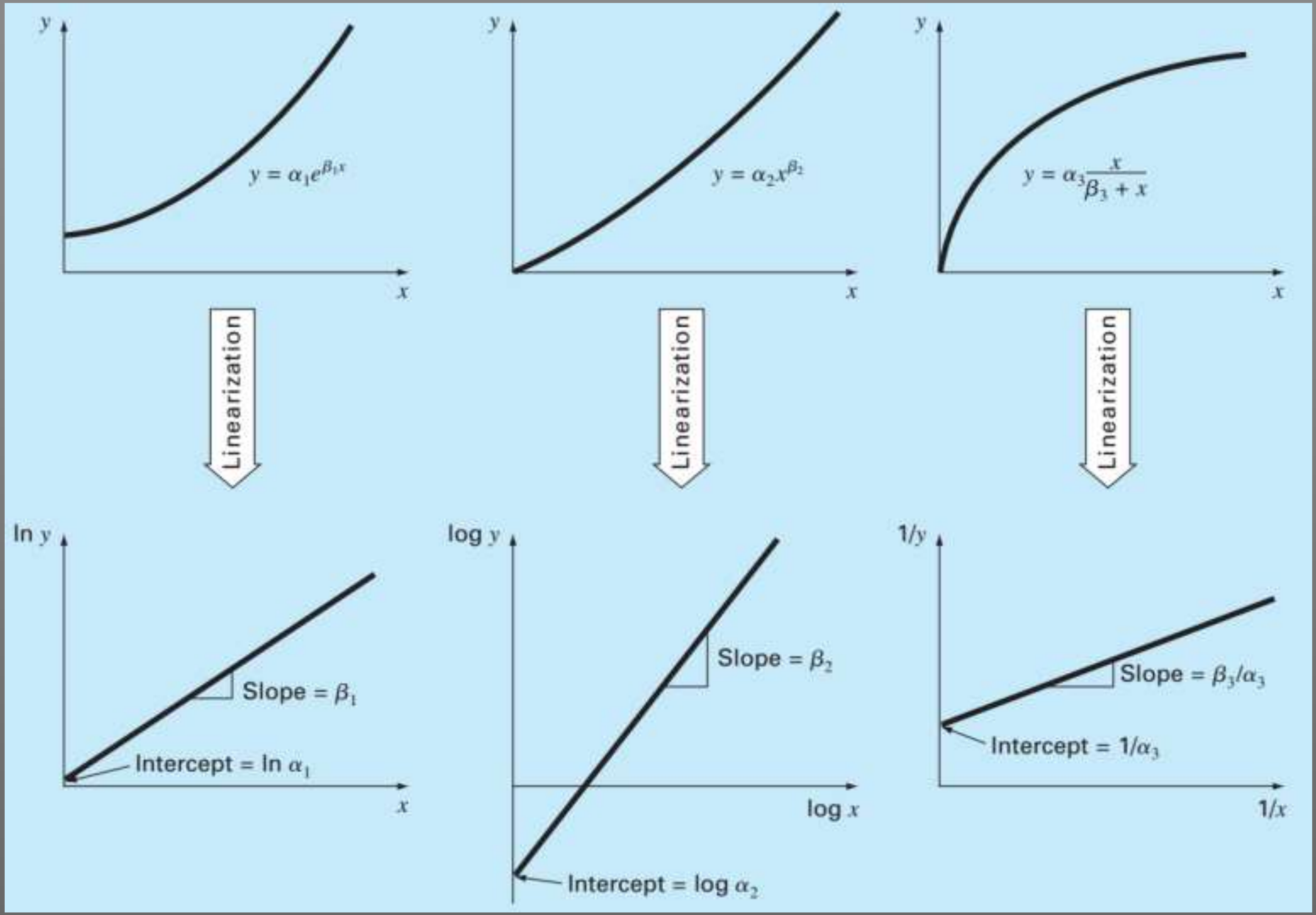
MATLAB© Script

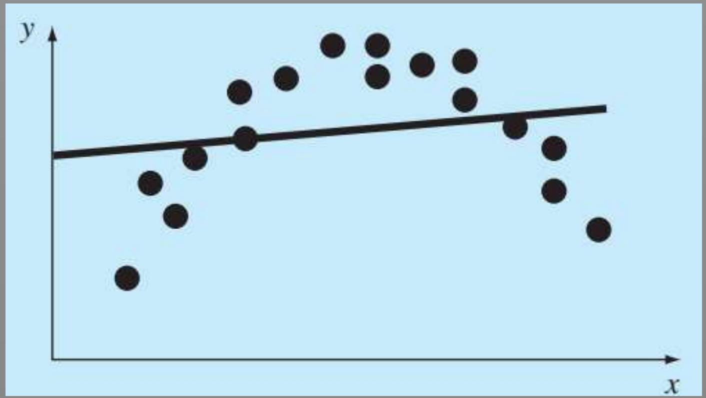
MATLAB Program for Linear Regression Method

```
clear all
clc
x = 1:7; % independent variable data set
y = [0.5 2.5 2.0 4.0 3.5 6.0 5.5]; % dependent variable data
n=length(x);
x_sum = sum(x);
y_sum = sum(y);
x_mean = mean(x);
y_mean = mean(y);
xy_sum = 0;
xi_sqsum = 0;
for i=1:n
    xy_sum = xy_sum + x(i)*y(i);
    xi_sqsum = xi_sqsum + (x(i))^2;
end
a1 = ((n*xy_sum) - x_sum*y_sum) / (n*xi_sqsum - x_sum^2)
a0 = y_mean - a1*x_mean
% Check with inbuit function
coef = polyfit(x,y,1)
% Plot data with curve
figure
hold on
plot(x,y,'d','LineWidth',1,'MarkerEdgeColor','k','MarkerFaceColor','g','MarkerSize',10)
yc = polyval([a1 a0],x);
plot(x,yc,'-b','LineWidth',2)
hleg=legend('Data set','Fitted Curve','Location','North');
set(hleg,'FontAngle','italic')
grid on; box on; axis on
hold off
```

Linearization

- For non-linear data set, linear regression can be applied after linearization of data set.

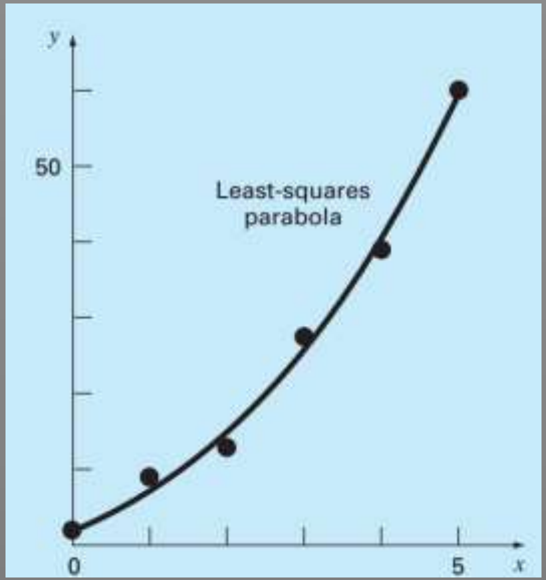




Polynomial Regression

- A polynomial based curve is used to fit for a set of data using least-square approximation.

$$y = a_0 + a_1 x + a_2 x^2 + e$$



Objective is to find coefficient/equation with minimum residual

$$S_r = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i - a_2 x_i^2 \right)^2 \approx 0$$

Using least-square:

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

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

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

Polynomial Regression

Objective is to find coefficient/equation with minimum residual

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

Standard Error

$$S_{y/x} = \sqrt{\frac{\sum\left(y_i - a_0 - a_1 x_i - a_2 x_i^2\right)^2}{n-(m+1)}} = \sqrt{\frac{S_r}{n-(m+1)}}$$

Determination coefficient

$$r^2 = \frac{\left(\sum\left(y_i - \bar{y}\right)^2\right) - S_r}{\left(\sum\left(y_i - \bar{y}\right)^2\right)}$$

Polynomial Regression

Example: Fit a second order polynomial for given data.

x_i	y_i
0	2.1
1	7.7
2	13.6
3	27.2
4	40.9
5	61.1
Σ	152.6

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

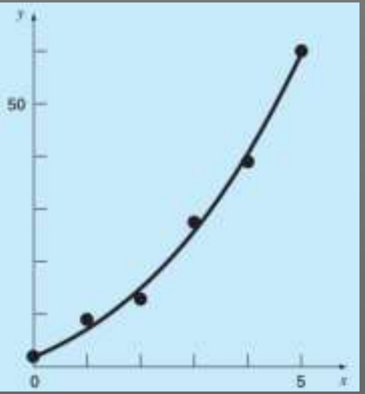
$$m = 2; \quad n = 6$$

$$\sum x_i = 15; \quad \sum x_i^2 = 55; \quad \sum x_i^3 = 225; \quad \sum x_i^4 = 979;$$
$$\sum y_i = 152.6; \quad \sum x_i y_i = 585.6; \quad \sum x_i^2 y_i = 2488.8;$$
$$\bar{x} = 2.5; \quad \bar{y} = 25.433$$

$$\Rightarrow \begin{bmatrix} 6 & 15 & 55 \\ 15 & 55 & 225 \\ 55 & 225 & 979 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} 152.6 \\ 585.6 \\ 2488.8 \end{Bmatrix}$$

$$a_0 = 2.4785; \quad a_1 = 2.3592; \quad a_2 = 1.8607$$

Best fit line: $\boxed{y = 2.4785 + 2.3592x + 1.8607x^2}$ $\Rightarrow$



Determination coefficient $r^2 = 0.9985$

Polynomial Regression

Step 1: Input order of polynomial to be fit m.

Step 2: Input n data sets (x_i, y_i).

Step 3: If $n < m+1$, print data set is insufficient, otherwise continue

Step 4: Compute the elements of least square approximation equations in the form of augmented matrix.

Step 5: Solve the augmented matrix for coefficients $a_0, a_1, a_2, a_3, a_4, \ldots a_n$.

MATLAB© Script

MATLAB Program for Polynomial Regression (Quadratic)

```
clear all
clc
x = 0:5; % data set
y = [2.1 7.7 13.6 27.2 40.9 61.1];
% data set
n=length(x);
x_sum = sum(x);
y_sum = sum(y);
x_mean = mean(x);
y_mean = mean(y);
x2sum = 0;
x3sum = 0;
x4sum = 0;
xy_sum = 0;
x2y_sum = 0;
for i=1:n
    x2sum = x2sum + (x(i))^2;
    x3sum = x3sum + (x(i))^3;
    x4sum = x4sum + (x(i))^4;
    xy_sum = xy_sum + x(i)*y(i);
    x2y_sum = x2y_sum + (x(i)^2)*y(i);
end
C=[n  x_sum  x2sum; x_sum  x2sum  x3sum;...
    x2sum  x3sum  x4sum];
b= [y_sum; xy_sum; x2y_sum];
A = inv(C)*b;
a0 = A(1); a1 = A(2); a2 = A(3);
% Check with inbuilt function
coef = polyfit(x,y,2)
```

Nonlinear Regression

- **Determine the values of parameters of non-linear fitting, to minimize the sum of the squares of the residuals.**
- **Gauss-Newton Method: A algorithm for minimizing the sum of the squares of the residuals between data and nonlinear equation.**

Objective is to find coefficient/equation with minimum residual

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

$$y_i = f(x_i) + e_i$$

Using Taylor series:

$$f(x_i)_{j+1} = f(x_i)_j + \frac{\partial f(x_i)_j}{\partial a_0}\Delta a_0 + \frac{\partial f(x_i)_j}{\partial a_1}\Delta a_1$$

$$y_i - f(x_i)_j = \frac{\partial f(x_i)_j}{\partial a_0}\Delta a_0 + \frac{\partial f(x_i)_j}{\partial a_1}\Delta a_1 + e_i \Rightarrow \{D\} = [Z_j]\{\Delta A_j\} + \{E\}$$

$$[Z_j] = \begin{bmatrix} \partial f_1/\partial a_0 & \partial f_1/\partial a_1 \\ \partial f_2/\partial a_0 & \partial f_2/\partial a_1 \\ . & . \\ \partial f_n/\partial a_0 & \partial f_n/\partial a_1 \end{bmatrix}$$

$$\{D\} = \begin{Bmatrix} y_1 - f(x_1) \\ y_2 - f(x_2) \\ . \\ y_n - f(x_n) \end{Bmatrix}$$

$$\{\Delta A\} = \begin{Bmatrix} \Delta a_0 \\ \Delta a_1 \\ . \\ \Delta a_n \end{Bmatrix}$$

Nonlinear Regression

From Taylor series implementation: $\{D\} = [Z_j]\{\Delta A_j\} + \{E\}$

Applying linear least square: $\left[[Z_j]^T [Z_j]\right]\{\Delta A\} = \left\{[Z_j]^T \{D\}\right\}$

$$a_{0,j+1} = a_{0,j} + \Delta a_0$$

$$a_{1,j+1} = a_{1,j} + \Delta a_1$$

Limitations of Gauss-Newton Method:

- Calculation of partial derivative of function.
- Converge slowly.
- Oscillate widely.
- May not converge, some times.

Nonlinear Regression (Gauss-Newton Method)

Example: Fit given nonlinear function for given data.

x	0.25	0.75	1.25	1.75	2.25
y	0.28	0.57	0.68	0.74	0.79

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

Use initial guesses $a_0 = 1; \quad a_1 = 1$

Partial derivatives of function:

$$\frac{\partial f}{\partial a_0} = \left(1 - e^{-a_1 x}\right) \qquad \frac{\partial f}{\partial a_1} = a_0 x e^{-a_1 x}$$

First Iteration:

$$[Z_0] = \begin{bmatrix} 0.2212 & 0.1947 \\ 0.5276 & 0.3543 \\ 0.7135 & 0.3581 \\ 0.8262 & 0.3041 \\ 0.8946 & 0.2371 \end{bmatrix}$$

$$\{D\} = \begin{Bmatrix} 0.28 - 0.2212 \\ 0.57 - 0.5276 \\ 0.68 - 0.7135 \\ 0.74 - 0.8262 \\ 0.79 - 0.8946 \end{Bmatrix} = \begin{Bmatrix} 0.0588 \\ 0.0424 \\ -0.0335 \\ -0.0862 \\ -0.1046 \end{Bmatrix}$$

$$[Z_0]^T [Z_0] = \begin{bmatrix} 2.3193 & 0.9489 \\ 0.9489 & 0.4404 \end{bmatrix}$$

Nonlinear Regression (Gauss-Newton Method)

x	0.25	0.75	1.25	1.75	2.25
y	0.28	0.57	0.68	0.74	0.79

$$[Z_0]^T\{D\} = \begin{Bmatrix} -0.1533 \\ -0.0365 \end{Bmatrix}$$

$$\left[[Z_j]^T[Z_j]\right]\{\Delta A\} = \left\{[Z_j]^T\{D\}\right\} \Rightarrow \{\Delta A\} = \begin{Bmatrix} -0.2714 \\ 0.5019 \end{Bmatrix}$$

$$\begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} + \begin{Bmatrix} -0.2714 \\ 0.5019 \end{Bmatrix} = \begin{Bmatrix} 0.7286 \\ 1.5019 \end{Bmatrix}$$

Second Iteration as so on till acceptable tolerance!!!

$$\begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \begin{Bmatrix} 0.7918 \\ 1.6751 \end{Bmatrix} \Rightarrow y = 0.7981\left(1 - e^{-1.6751x}\right)$$

MATLAB© Script

MATLAB Program for Gauss-Newton Regression (Nonlinear)

```
clear all
clc
X = 0.25:0.5:2.25; % data set
Y = [0.28 0.57 0.68 0.74 0.79]; % data set
A0 = 1;     A1 = 1;% Initial guess
n=length(X);
syms a0 a1 x
fun=a0*(1-exp(-a1*x));
fun_diff0=diff(fun,a0);
fun_diff1=diff(fun,a1);
dfa0 = subs(fun_diff0,a1,A1);
dfa0 = vpa(dfa0);
dfa1 = subs(fun_diff1,a0,A0);     % Initial guess a1=1
dfa1 = subs(dfa1,a1,A1);
dfa1 = vpa(dfa1);
for i = 1:n
    dFa0 = subs(dfa0,x,X(i));
    Z_a0(i) = double(dFa0);
    dFa1 = subs(dfa1,x,X(i));
    Z_a1(i) = double(dFa1);
    D(i,1) = Y(i) - gaussnewfun(A0,A1,X(i));
end
z = [Z_a0'   Z_a1' ];
Z = z'*z;
DA = (inv(Z))*(z'*D);
A = [A0; A1]+DA  % First iteration coefficient
```

```
function y =gaussnewfun(a0,a1,x)
y=a0*(1-exp(-a1*x));
```

THANK YOU

Questions??