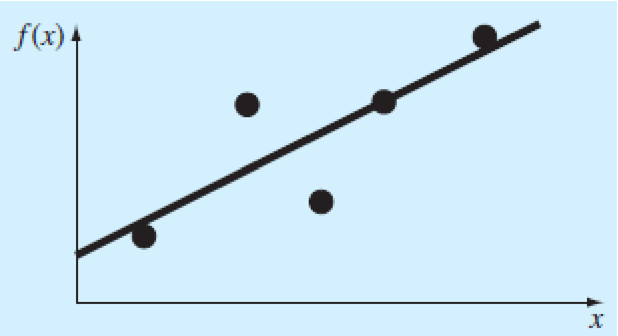
Linear Regression

Hsiao-Lung Chan
Dept Electrical Engineering
Chang Gung University, Taiwan
chanhl@mail.cgu.edu.tw

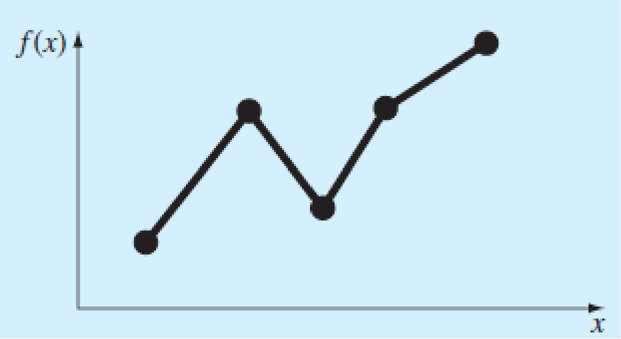
Curve fitting

- Least-squares regression
  - Data exhibit a significant degree of error or "scatter"
  - A curve for the trend of the data

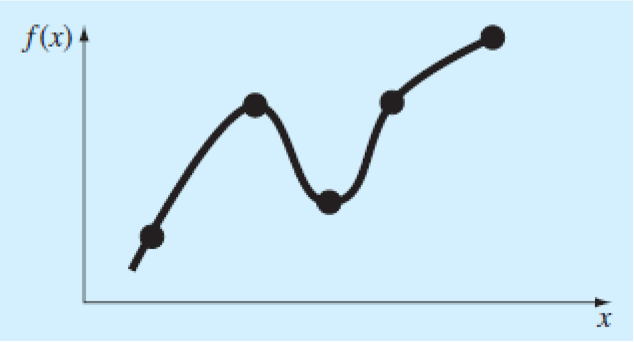


- Interpolation
  - Data are very precise
  - Fit a curve passing directly through each point

Linear interpolation

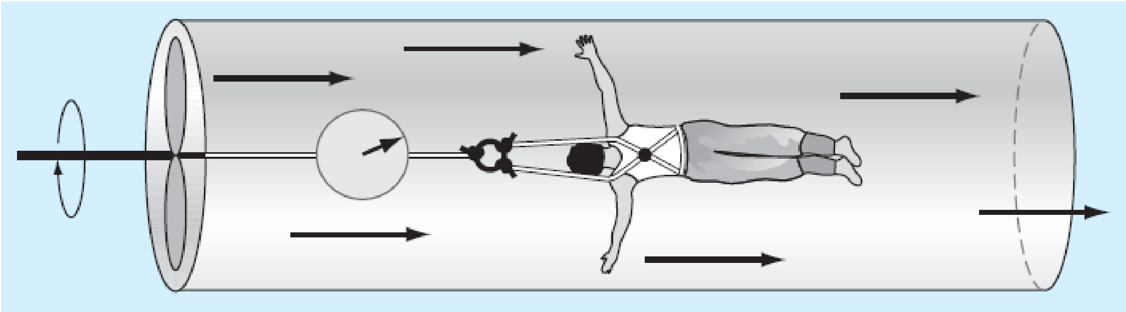


Curvilinear interpolation

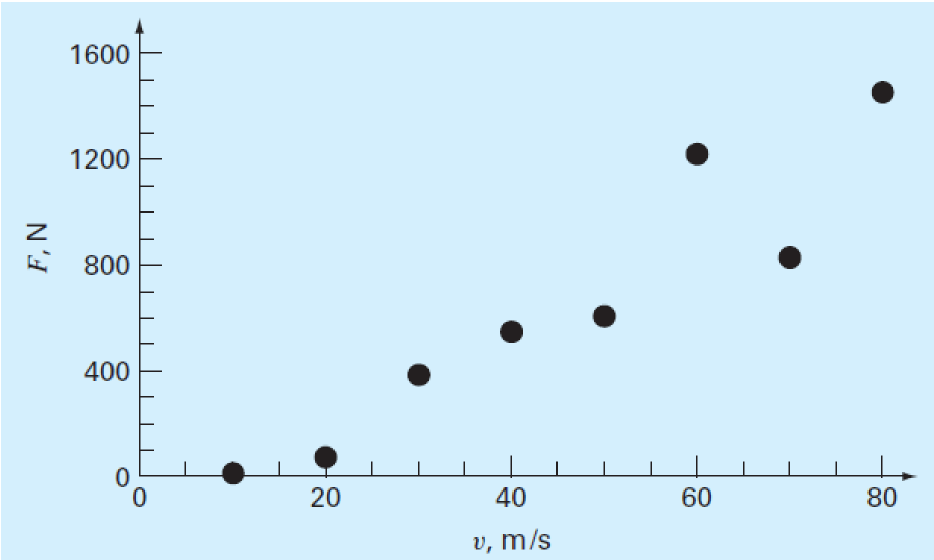


Wind tunnel experiment

- Measure how the force of air resistance depends on velocity



- Force versus wind velocity for an object suspended in a wind tunnel



Basic statistics

- Arithmetic mean $$\bar{y} = \frac{\sum y_i}{n}$$ **mean(y)**

- Variance

$$s_y^2 = \frac{\sum (y_i - \bar{y})^2}{n-1} = \frac{\sum y_i^2 - \left(\sum y_i\right)^2 / n}{n-1}$$

var(y) or std(y)^2

degrees of freedom:
ȳ is known and n-1 of the values are specified

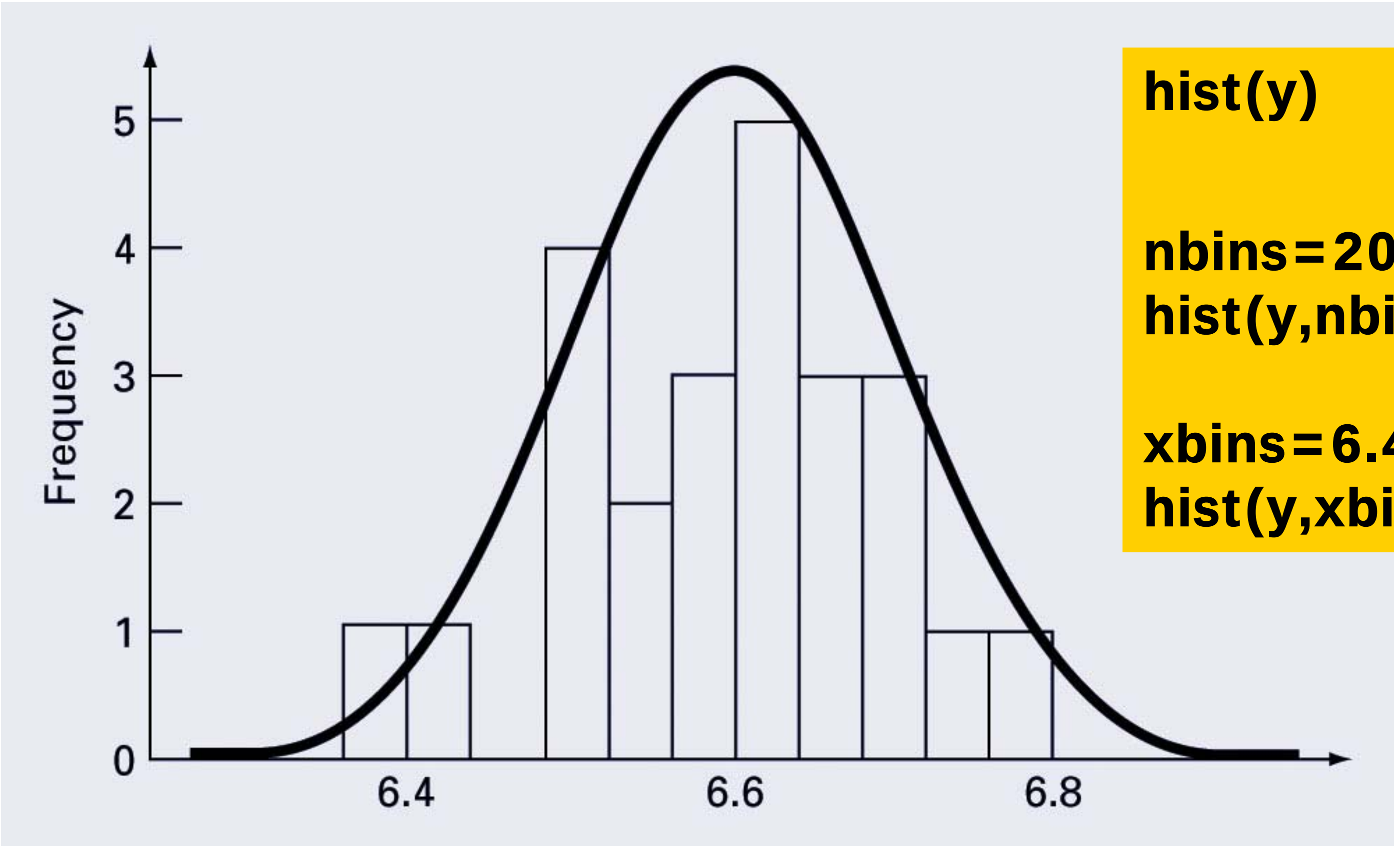
- Coefficient of variation **cv=std(y)/mean(y)*100;**

$$\text{c.v.} = \frac{s_y}{\bar{y}} \times 100\%$$

quantifying the spread of data

- Median, the midpoint of a group of data **median(y)**

Normal distribution (Gaussian distribution)




```
hist(y)

nbins=20;
hist(y,nbins)

xbins=6.4:0.2:6.8;
hist(y,xbins)
```

Random number based on uniform distribution

- Generates a sequence of numbers that are uniformly distributed between 0 and 1

```
r = rand(m, n);   % m-by-n matrix of random numbers
```

- Generate a uniform distribution on another interval

```
runiform = low + (up – low) * rand(m, n);
% low = the lower bound,  up = the upper bound
```

Simulate downward velocity based on uniform random values of drag in free-falling bungee jumper

$$v = \sqrt{\frac{gm}{c_d}} \tanh\left(\sqrt{\frac{gc_d}{m}}\, t\right)$$

```
t=4; m=68.1; g=9.81;  % parameters
cd=0.25;  % drag coefficients
cdmin=cd-0.025; cdmax=cd+0.025;

r=rand(1000,1);   % generate random values of drag
cdrand=cdmin+(cdmax-cdmin)*r;

subplot(2,2,1)
plot(cdrand), ylabel('drag coefficient')

Subplot(2,2,2)
hist(cdrand), title('Distribution of drag'), xlabel('cd (kg/m)')
```

Simulate downward velocity (cont.)

$$v = \sqrt{\frac{gm}{c_d}} \tanh\left(\sqrt{\frac{gc_d}{m}}\, t\right)$$

```
vrand=sqrt(g*m./cdrand).*tanh(sqrt(g*cdrand/m)*t);

subplot(2,2,3)
plot(vrand), ylabel(‘velocity')

subplot(2,2,4)
hist(vrand), title('Distribution of velocity'), xlabel('v (m/s)')
```

Random number based on normal distribution

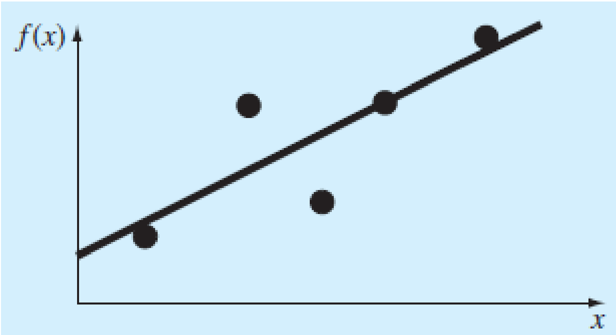
- Generates a sequence of numbers that have a normal distribution with zero mean and standard deviation of 1

```
r = randn(m, n);   % m-by-n matrix of random numbers
```

- Generate a normal distribution with a different mean (mn) and standard deviation (s)

rnormal = mn + s * **randn**(m, n);

Linear least-squares regression



$$f(x) = a_0 + a_1 x$$

$$y = a_0 + a_1 x$$

"Best" for least-squares regression means minimizing the sum of the squares of the estimate residuals.

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

$$\frac{\partial S_r}{\partial a_1} = 0$$

$$\frac{\partial S_r}{\partial a_2} = 0$$

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

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

Curve fitting by linear regression

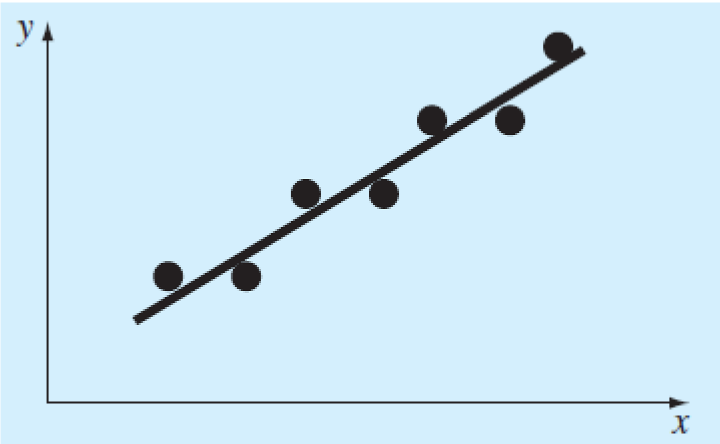
```
function [a, r2] = linregr(x,y)
  % x = independent variable, y = dependent variable
  % output: a(1) = slope, a(2)=intercept
  % r2 = coefficient of determination
  n = length(x);
  x = x(:); y = y(:); % convert to column vectors
  sx = sum(x); sy = sum(y);
  sx2 = sum(x.*x);
  sxy = sum(x.*y);
  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;
```

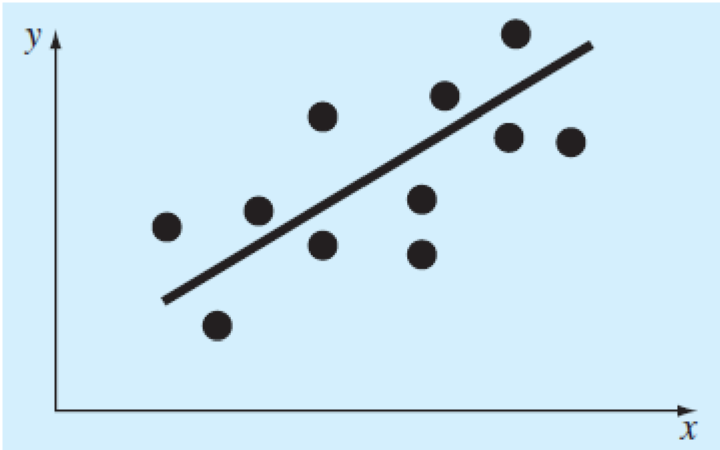
Curve fitting by linear regression (main program)

```
% get a x y pairs
x=[10 20 30 40 50 60 70 80];
y=[25 70 380 550 610 1220 830 1450];

% compute the coefficients of regression line
[a, r2] = linregr(x,y);

% 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
```

Quantification of error of linear regression

- Linear regression with small and large residual errors





How to quantify the "goodness" of regression fit?

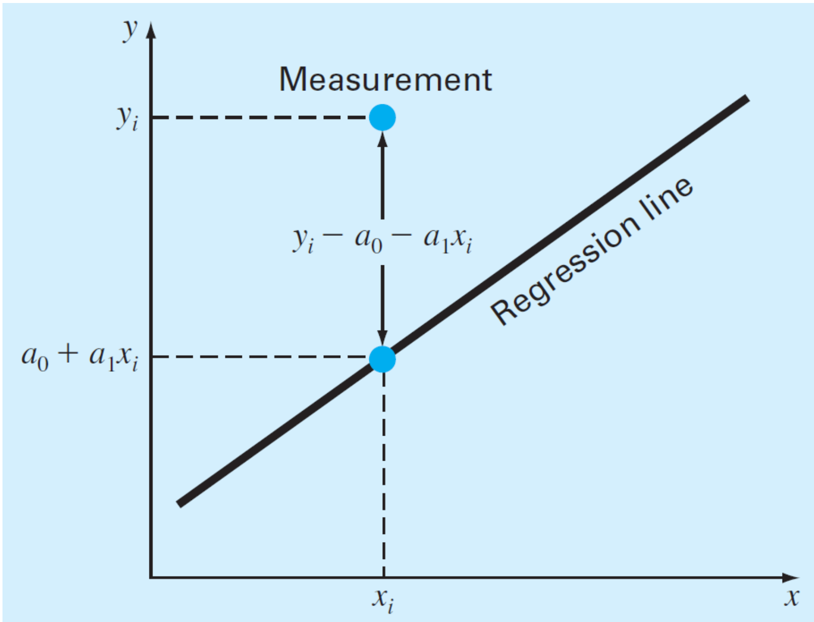
Quantification of error of linear regression (cont.)

- Sum of the squares of the residuals between data points and regression line

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

- Sum of the squares of the residuals between data points and mean

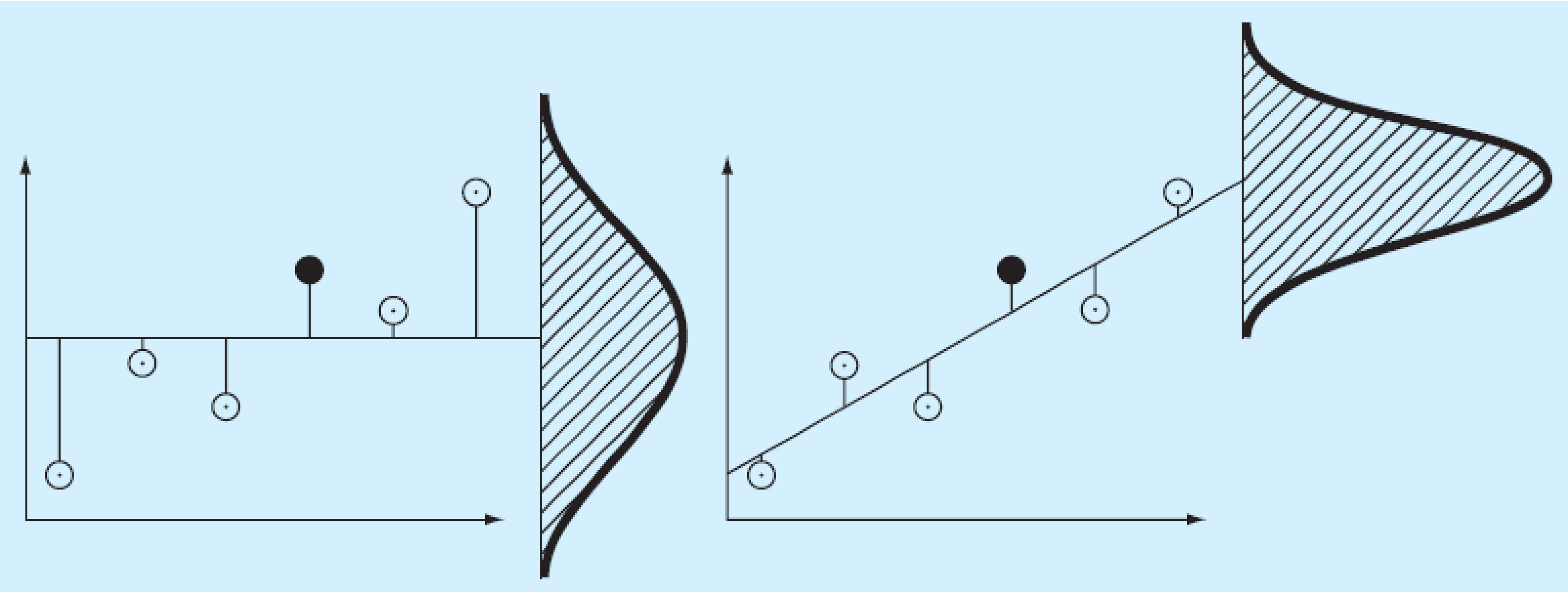
$$S_t = \sum (y_i - \bar{y})^2$$



Quantification of error of linear regression (cont.)

Spread of the data around the mean

Spread of the data around the best-fit line



Quantification of error of linear regression (cont.)

- Coefficient of determination

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

MATLAB built-in function **polyfit**

- Fits a least-squares n^{th}-order polynomial to data

  p = polyfit(x, y, n);

$$f(x) = p_1 x^n + p_2 x^{n-1} + \cdots + \boxed{p_n x + p_{n+1}}$$

- Example

  x = [10 20 30 40 50 60 70 80];

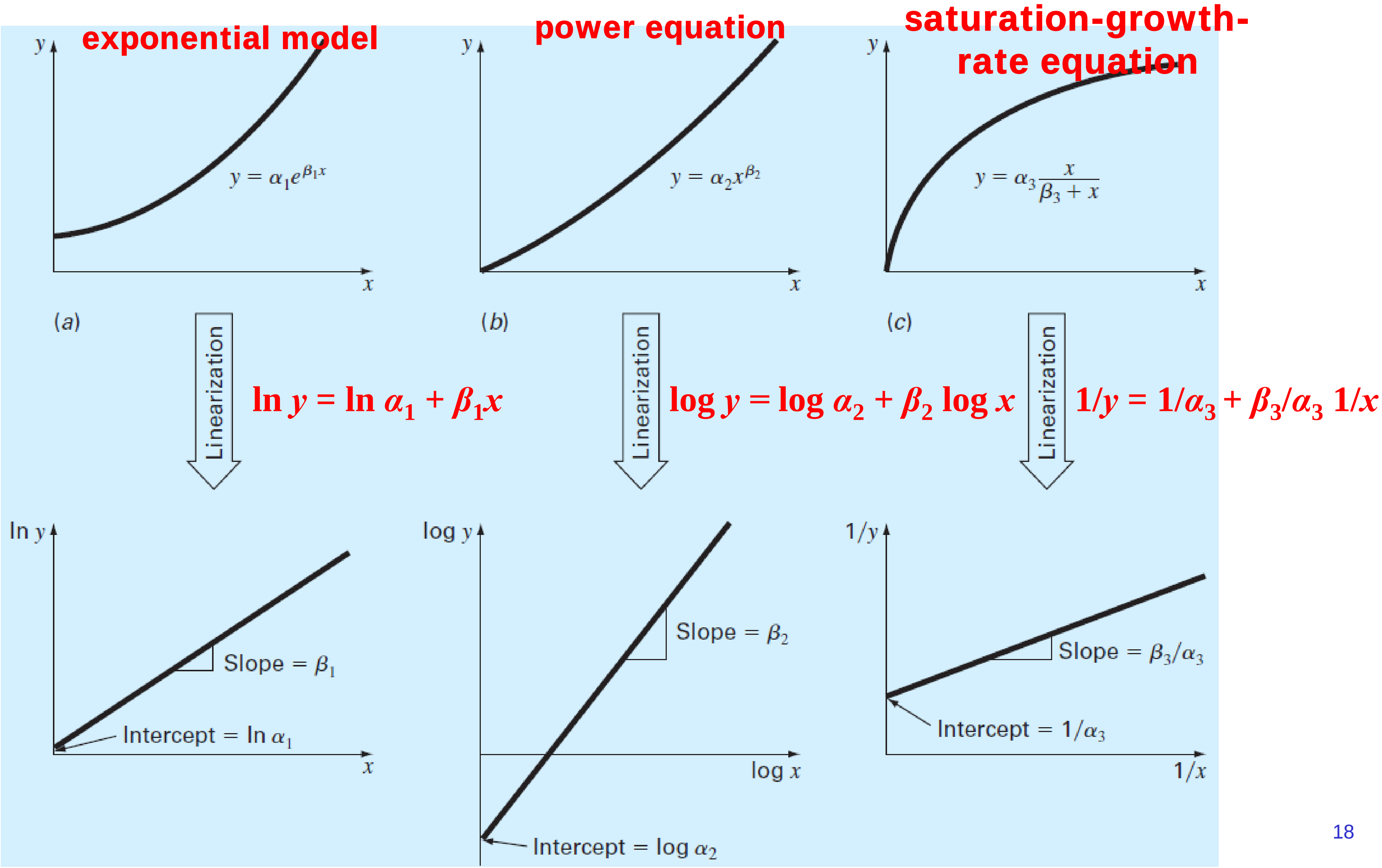
  y = [25 70 380 550 610 1220 830 1450];

  a = polyfit(x,y,**1**);

- polyval, computing a value using the coefficients

  y = polyval(p, x);

Linearization of nonlinear relationships



Example: Fitting data with the power equation

i	x_i	y_i	$\log x_i$	$\log y_i$	$(\log x_i)^2$	$\log x_i \log y_i$
1	10	25	1.000	1.398	1.000	1.398
2	20	70	1.301	1.845	1.693	2.401
3	30	380	1.477	2.580	2.182	3.811
4	40	550	1.602	2.740	2.567	4.390
5	50	610	1.699	2.785	2.886	4.732
6	60	1220	1.778	3.086	3.162	5.488
7	70	830	1.845	2.919	3.404	5.386
8	80	1450	1.903	3.161	3.622	6.016
Σ			12.606	20.515	20.516	33.622

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

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

$$\bar{x} = \frac{12.606}{8} = 1.5757 \qquad \bar{y} = \frac{20.515}{8} = 2.5644$$

$$a_1 = \frac{8(33.622) - 12.606(20.515)}{8(20.516) - (12.606)^2} = 1.9842$$

$$a_0 = 2.5644 - 1.9842(1.5757) = -0.5620$$

The least-squares fit $\log y = -0.5620 + 1.9842 \log x$

Reference

- Steven C. Chapra "Applied Numerical Methods with MATLAB", 3rd ed., McGraw Hill, 2012.