Mahidol University
Faculty of Information and Communication Technology

Linear Regression

Rawesak Tanawongsuwan, Ph.D.
rawesak.tan@mahidol.ac.th

This slide is part of teaching materials for ITCS122 Numerical Methods
Semester 2/2023, Calendar year 2024

Linear least-squares regression

- A curve-fitting strategy by approximating the shape of the data without necessarily matching or passing through the individual points.
- One simple way is to fit a straight line to a set of data : $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- We should minimize the discrepancy between data points and the line.

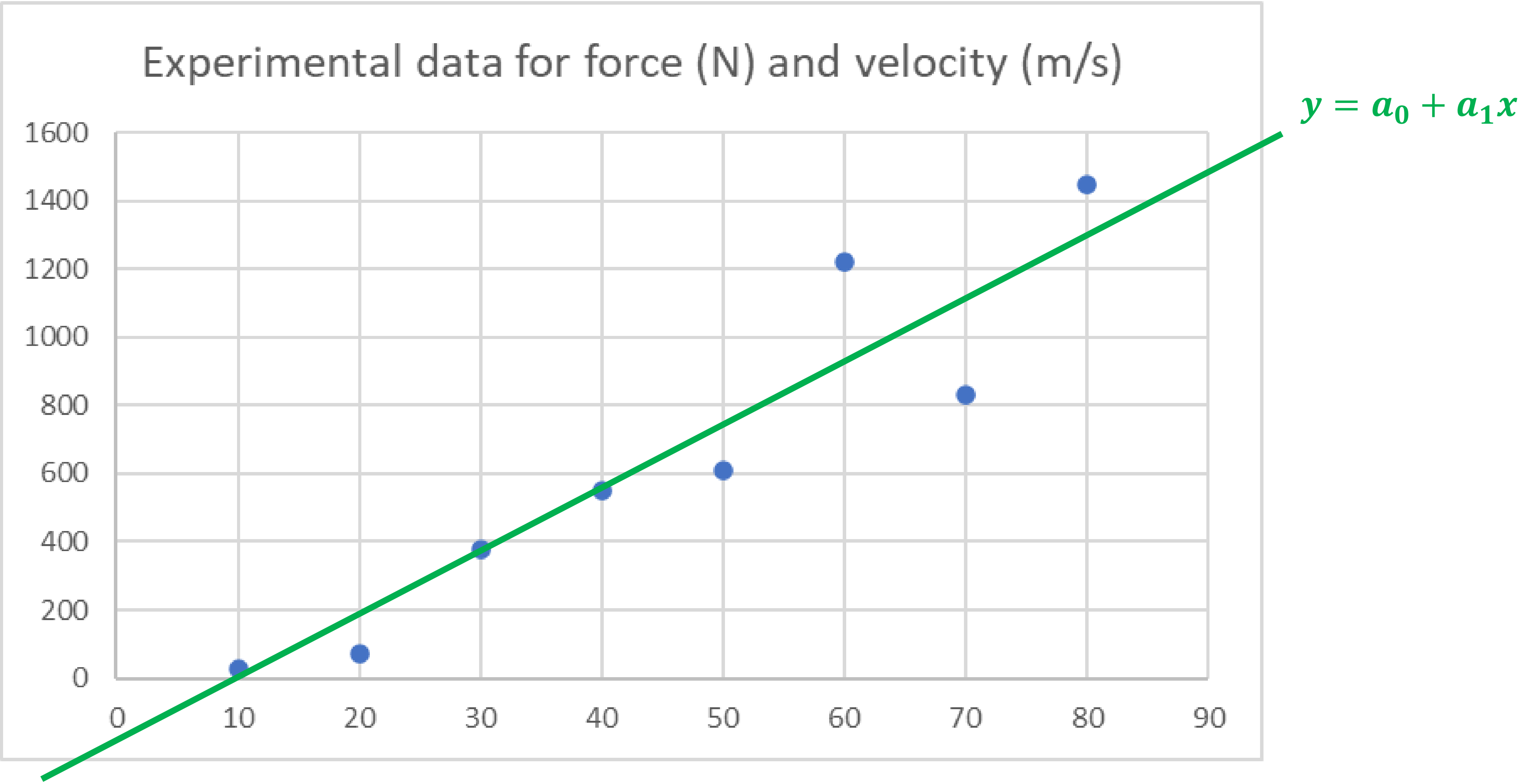
$$y = a_0 + a_1 x + e$$

  - a_0 and a_1 are coefficient representing the intercept and the slope.
  - e is the error or residual between the model and the data

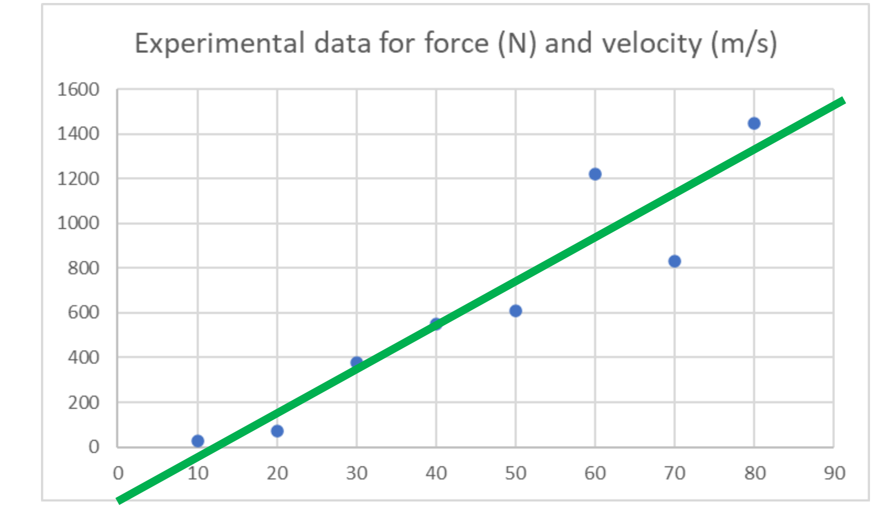
$$e = y - a_0 - a_1 x$$

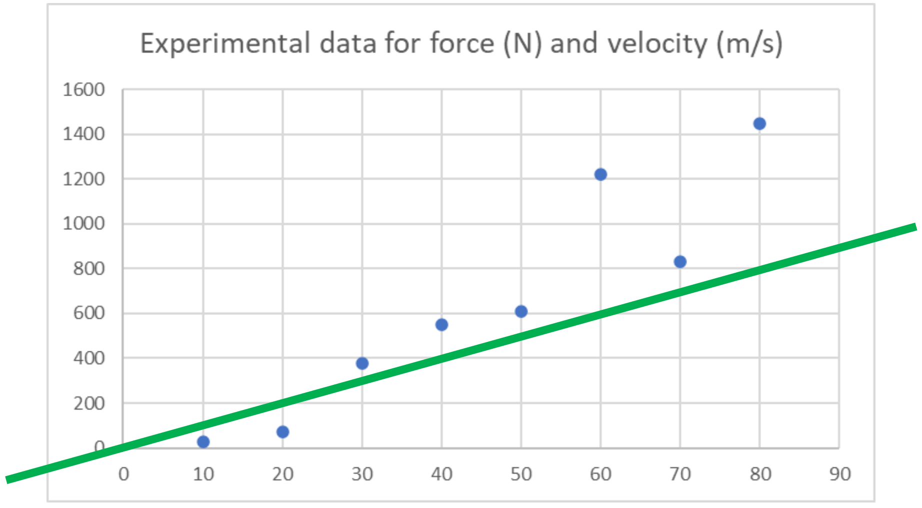
  - the residual e is the discrepancy between true value of y and the approximate value $a_0 + a_1 x$, predicted by the linear equation.

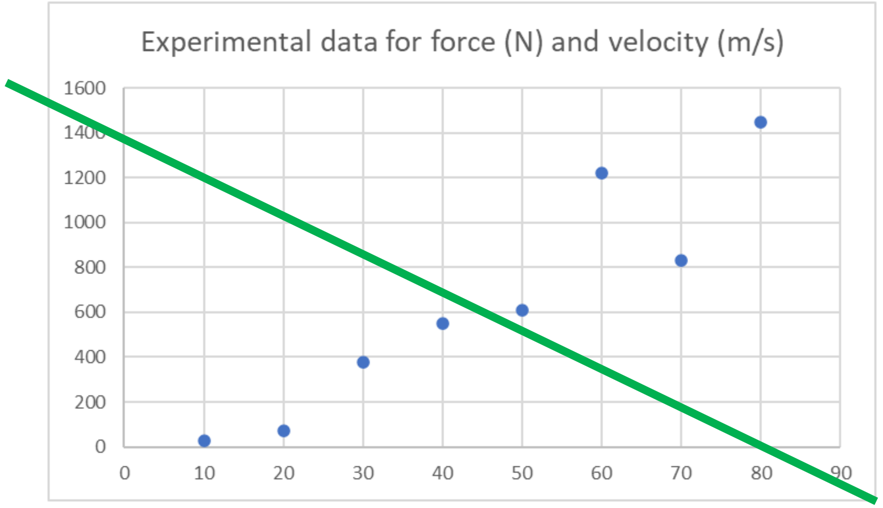
The residual e

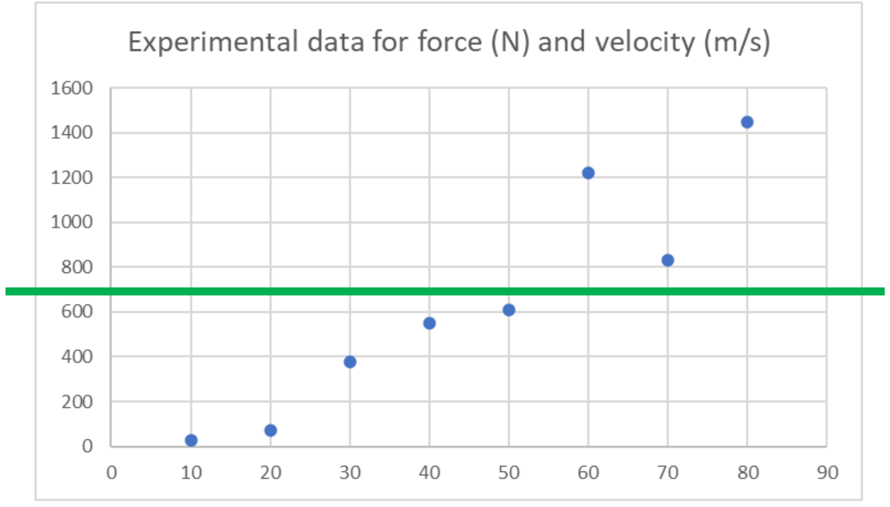


Many different ways to do line-fitting





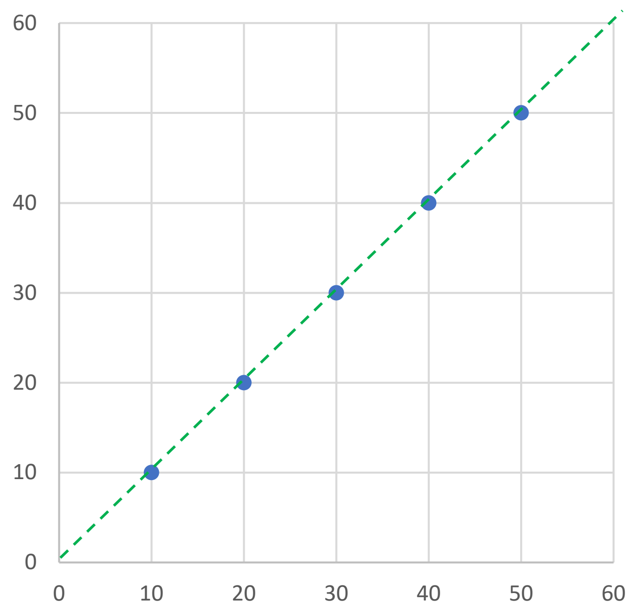


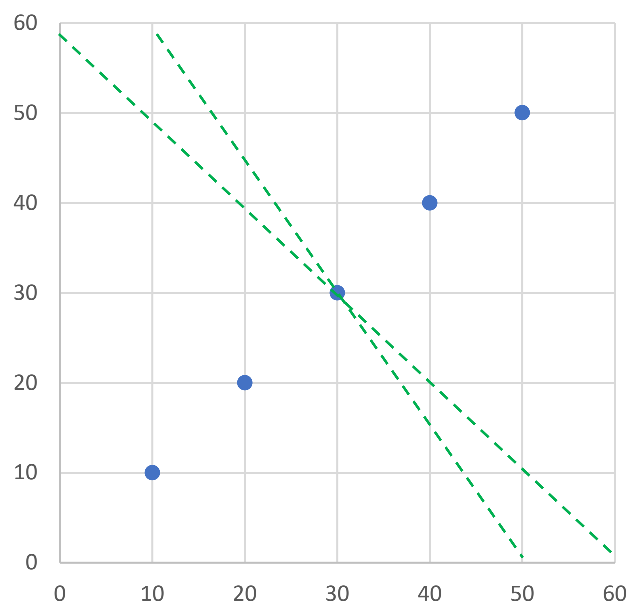


Criteria for a “Best” Fit

- Criteria 1 : minimize the sum of the residual errors for all the available data, where $n =$ total number of points

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

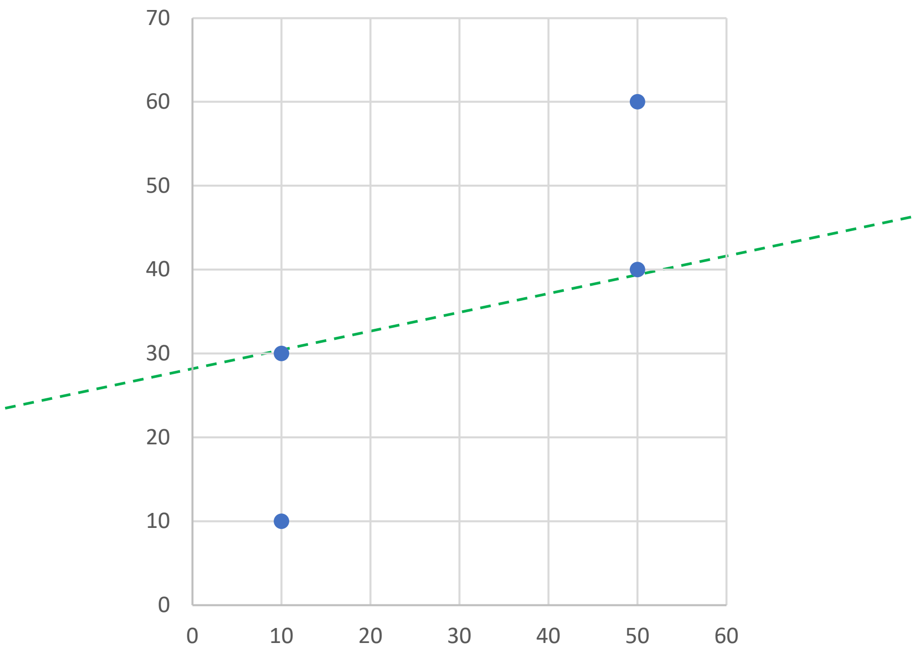


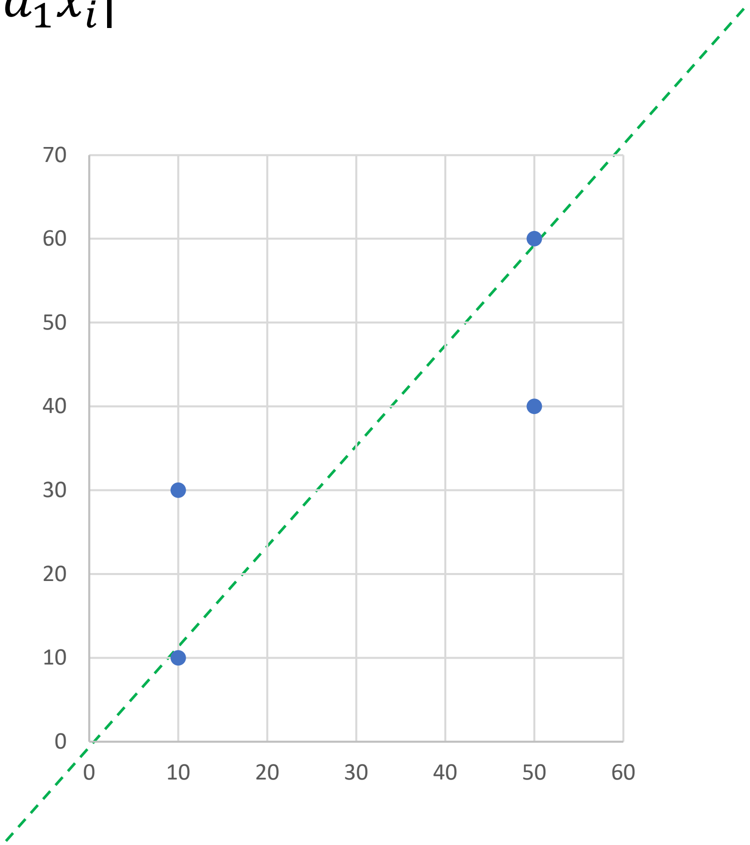


Criteria for a "Best" Fit

- Criteria 2 : minimize the sum of the absolute values of residual errors for all the available data, where $n =$ total number of points

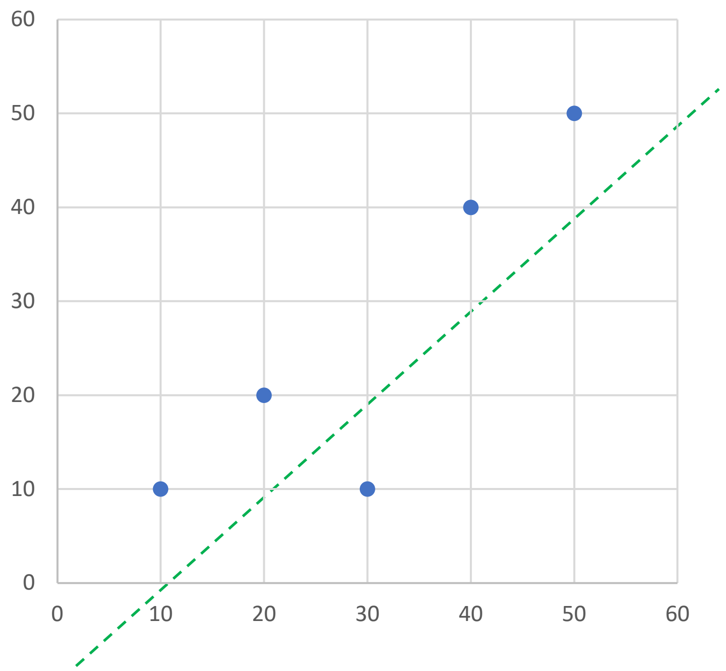
$$\sum_{i=1}^{n} |e_i| = \sum_{i=1}^{n} |y_i - a_0 - a_1 x_i|$$





Criteria for a “Best” Fit

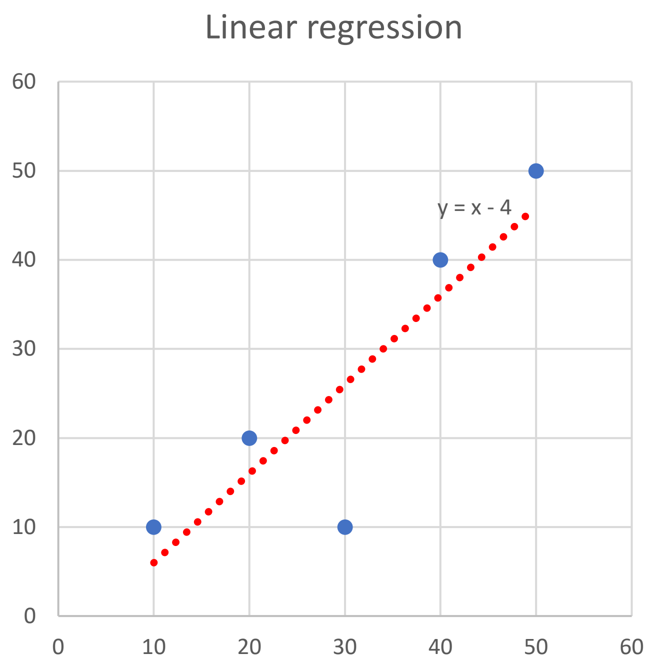
- Criteria 3 : minimize the maximum distance that an individual point falls from the line (*minimax* criterion)



Criteria for a "Best" Fit

- Criteria 4 : minimize the sum of the squares of the residuals (*least squares* criterion)

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



Least-squares fit of a straight line

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

$$\frac{\partial S_r}{\partial a_0} = -2\sum(y_i - a_0 - a_1 x_i) \quad \Rightarrow \quad 0 = \sum y_i - \sum a_0 - \sum a_1 x_i$$

$$\frac{\partial S_r}{\partial a_1} = -2\sum[(y_i - a_0 - a_1 x_i)x_i] \quad \Rightarrow \quad 0 = \sum x_i y_i - \sum a_0 x_i - \sum a_1 x_i^2$$

Note that $\sum a_0 = n a_0$

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

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

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

Example : A free-falling object problem setup

- A free-falling object which is subject to the upward force of air resistance. A simple approximation is that the force was proportional to the square of velocity

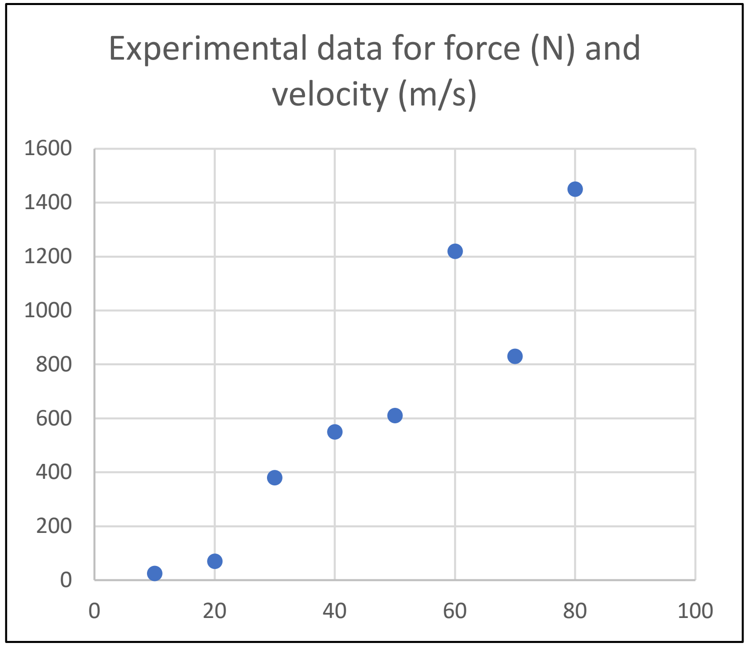
$$F_U = c_d v^2$$

F_U = the upward force of air resistance $[N = kg\, m/s^2]$

c_d = a drag coefficient $[kg/m]$

v = velocity $[m/s]$

	Experimental data for force (N) and velocity (m/s)							
v	10	20	30	40	50	60	70	80
F	25	70	380	550	610	1220	830	1450



Linear regression

i	x_i	y_i	x_i^2	$x_i y_i$
1	10	25	100	250
2	20	70	400	1400
3	30	380	900	11400
4	40	550	1600	22000
5	50	610	2500	30500
6	60	1220	3600	73200
7	70	830	4900	58100
8	80	1450	6400	116000
$\sum$	360	5135	20400	312850

- Means

$\bar{x} =$ $\bar{y} =$

- Slope and intercept :

$a_1 =$

$a_0 =$

- The least-squares fit :

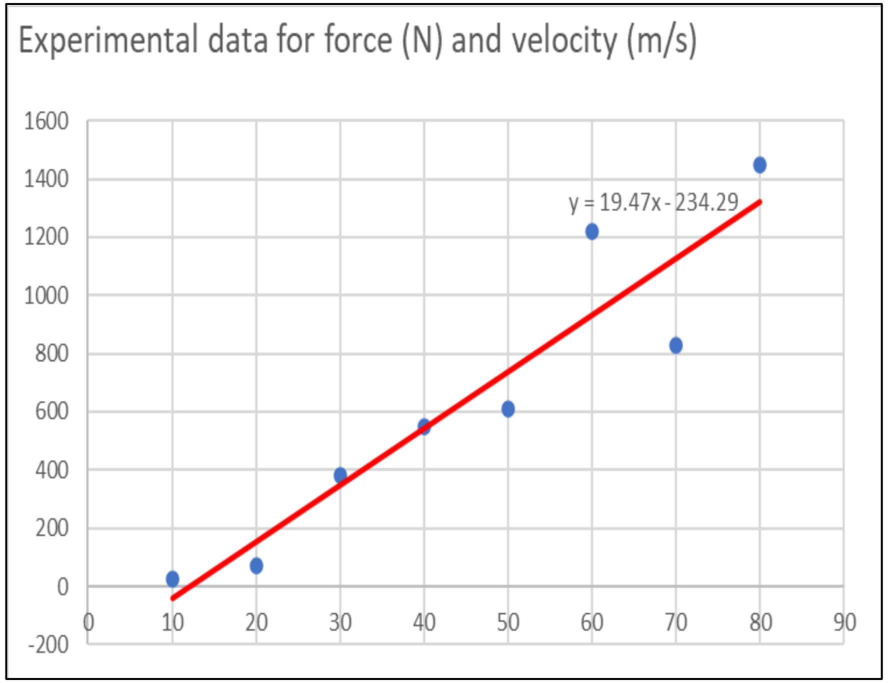
$F =$

Physically unrealistic negative forces

- This example shows that the line that fits the data, however, the zero intercept means that the equation predicts physically unrealistic negative forces at low velocities.

- The linear model might not be a good choice in this fitting problem. However, it is still a simple technique to use in many cases.

- Later on, transformation techniques can be used to come up with alternative best-fit line that is more physically realistic.



Quantification of error of linear regression

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

- The square of the discrepancy between the data and a single estimate of the measure of central tendency – the mean

$$S_r = \sum (y_i - a_0 - a_1 x_i)^2$$

- The square of the vertical distance between the data and another measure of central tendency – the straight line

$$s_y = \sqrt{\frac{S_t}{n-1}}$$

- Standard deviation quantifies the spread of the data around the mean

$$s_{y/x} = \sqrt{\frac{S_r}{n-2}}$$

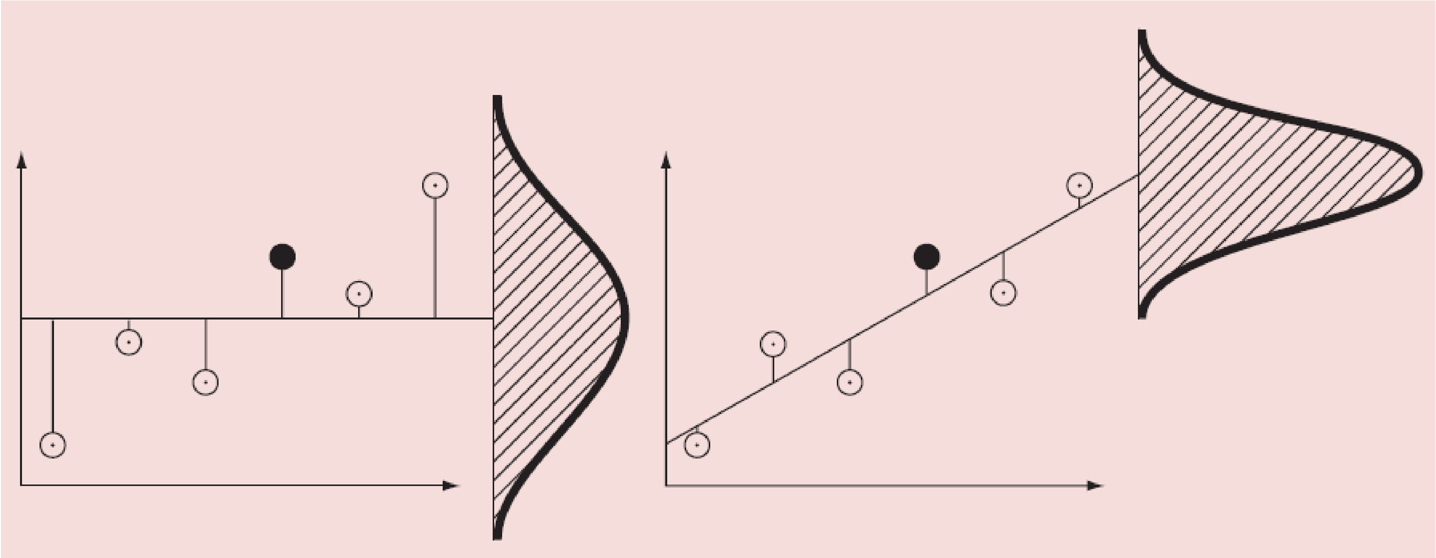
- Standard error of the estimate quantifies the spread of the data around the regression line

The standard error of the estimate $s_{y/x}$

- The subscript notation "y/x“ represents the error for a predicted value of y corresponding to a particular value of x.
- Dividing by $n-2$ comes from two data-derived estimates a_0 and a_1 were used to compute S_r, therefore we have lost two degrees of freedom.
- Another justification for dividing by $n-2$ is that there is no such thing as the “spread of data” around a straight line connecting two points. Therefore, for the case where $n=2$, $s_{y/x}$ yields a meaningless result of infinity.

Around the mean vs around the regression line

- Regression data showing the spread of the data around the mean vs the spread of the data around the regression line
- The reduction in the spread from the left plot to the right plot represents the improvement due to linear regression



Goodness of fit

- The difference $S_t - S_r$ quantifies the improvement or error reduction due to describing the data in terms of a straight line rather than an average value (mean).
- Since the magnitude of this quantify is scale-dependent, we normalize to S_t to be

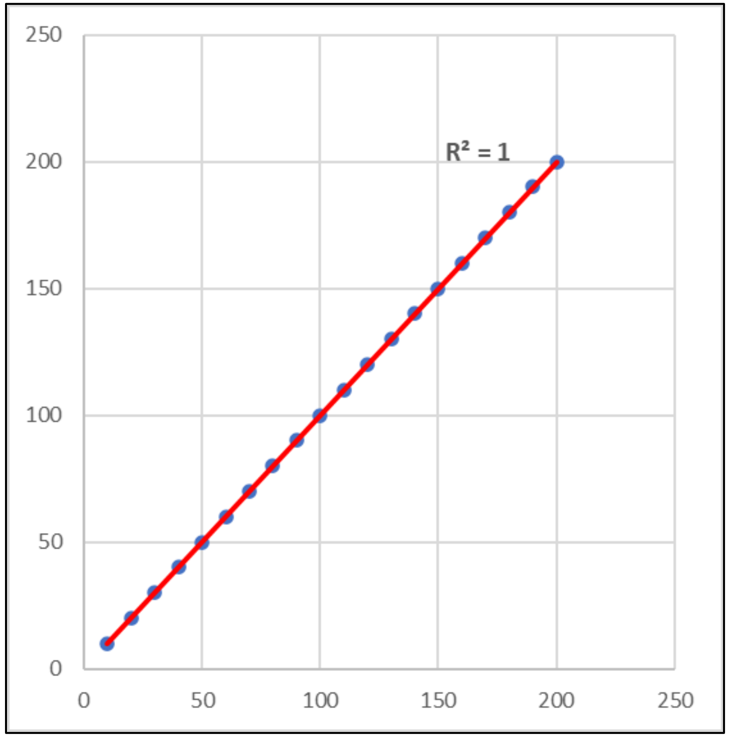
$$r^2 = \frac{S_t - S_r}{S_t}$$

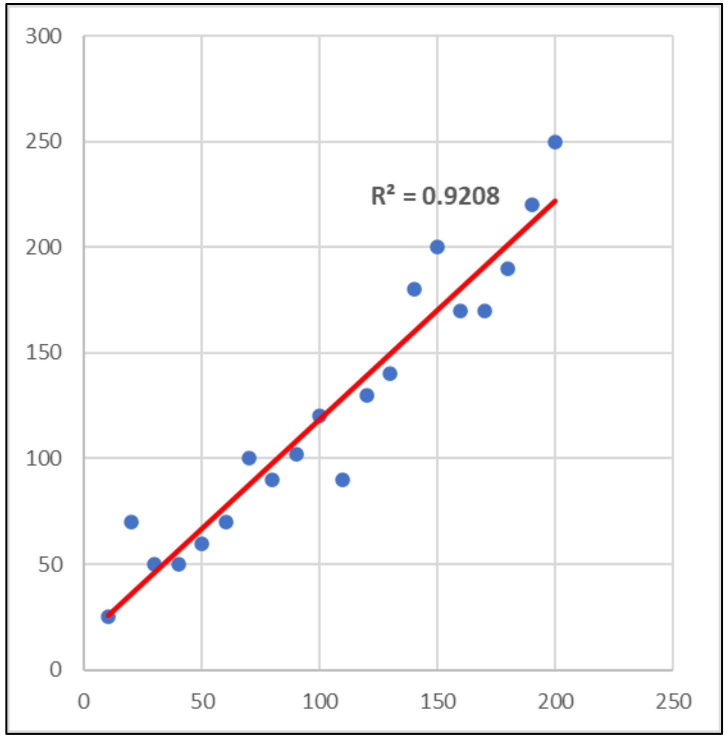
- r^2 is called *the coefficient of determination*
- r is called *the correlation coefficient* → $r = \sqrt{r^2}$
- A more convenient formula for r is

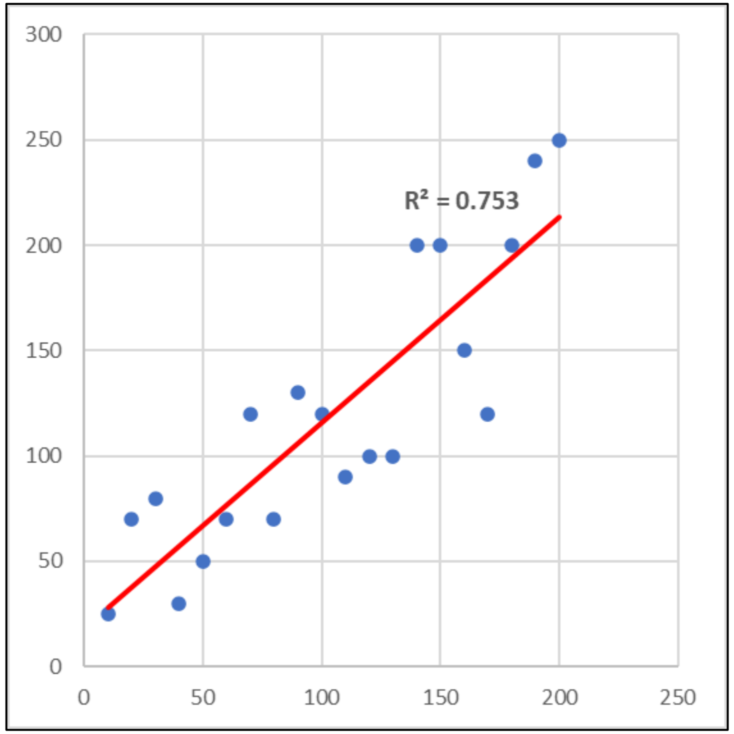
$$r = \frac{n\sum(x_i y_i) - (\sum x_i)(\sum y_i)}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}}$$

Goodness of fit

- For a perfect fit, $S_r = 0$ and $r^2 = 1$ → the line explains 100% of the variability of the data.
- For $r^2 = 0$ and $S_r = S_t$ → the line represents no improvement.







Example : A free-falling object problem setup

- A free-falling object which is subject to the upward force of air resistance. A simple approximation is that the force was proportional to the square of velocity

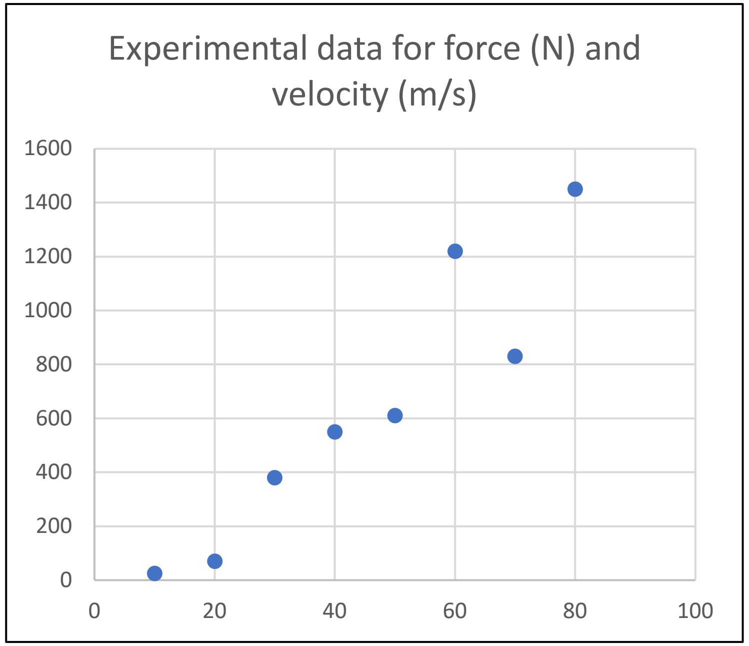
$$F_U = c_d v^2$$

F_U = the upward force of air resistance $[N = kg\, m/s^2]$

c_d = a drag coefficient $[kg/m]$

v = velocity $[m/s]$

	Experimental data for force (N) and velocity (m/s)							
v	10	20	30	40	50	60	70	80
F	25	70	380	550	610	1220	830	1450



Goodness of fit (free-falling object example)

- The standard deviation

  $s_y =$

- The standard error of the estimate

  $s_{y/x} =$

- The coefficient of determination

  $r^2 =$

- The correlation coefficient

  $r =$

- ________% of the original uncertainty has been explained by the linear model.

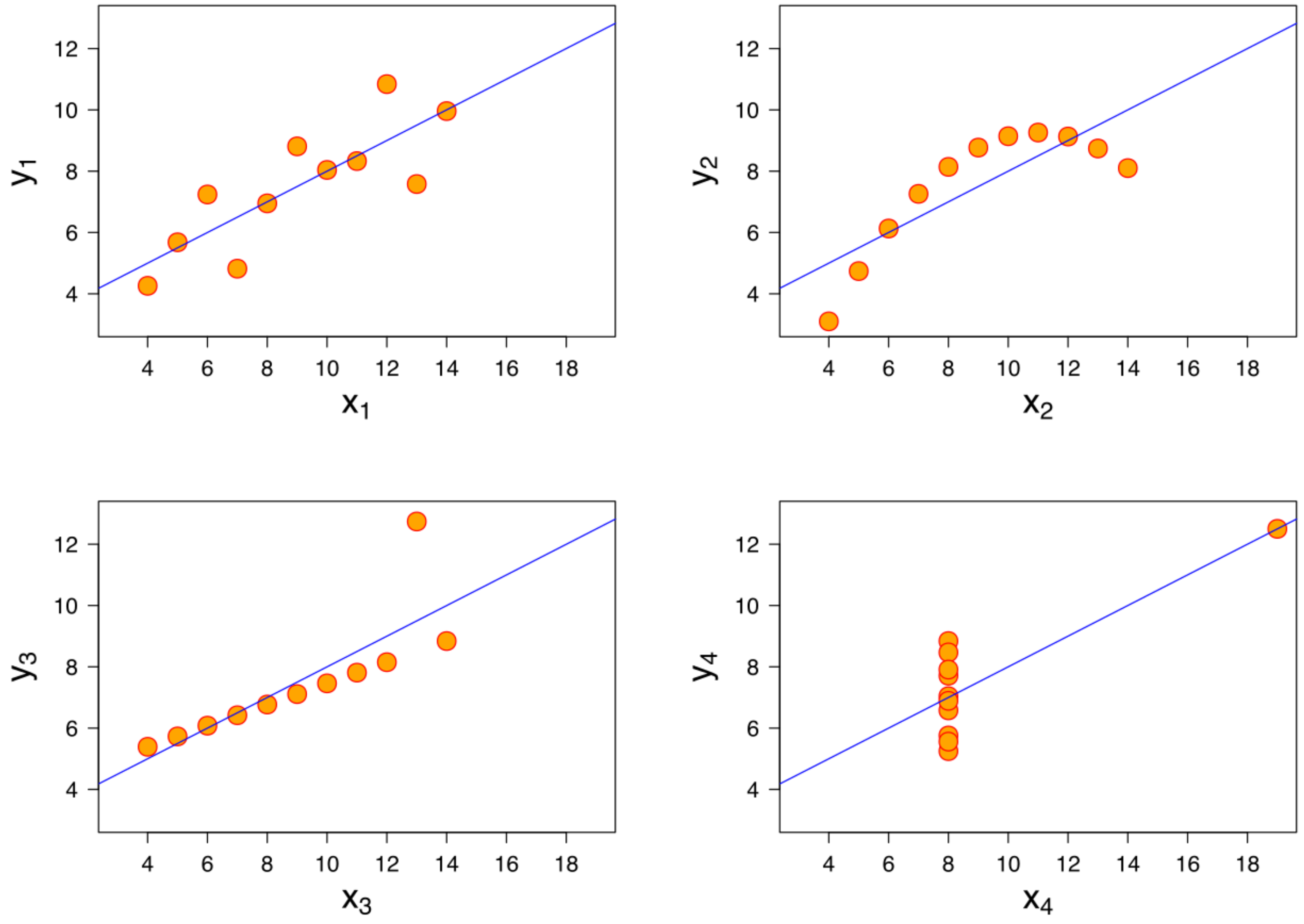
i	x_i	y_i	$a_0 + a_1 x_i$	$(y_i - \bar{y})^2$	$(y_i - a_0 - a_1 x_i)^2$
1	10	25	-39.58	380535	4171
2	20	70	155.12	327041	7245
3	30	380	349.82	68579	911
4	40	550	544.52	8441	30
5	50	610	739.23	1016	16699
6	60	1220	933.93	334229	81837
7	70	830	1128.63	35391	89181
8	80	1450	1323.33	653066	16044
$\sum$	360	5135		1808297	216118

Goodness of fit -- A word of caution

- r^2 is close to 1 does not mean that the fit is always "good"
  - Especially when the underlying relationship between y and x is not even linear
  - We should always inspect a plot of the data along with the regression curve.

- Anscombe's quartet
  - 4 datasets consisting of 11 data points each
  - The graphs are very different, but all have the same best-fit equation and the same coefficient of determination
  - The quartet is used to illustrate the importance of looking at a set of data graphically before starting to analyze

Anscombe's quartet



https://en.wikipedia.org/wiki/Anscombe%27s_quartet

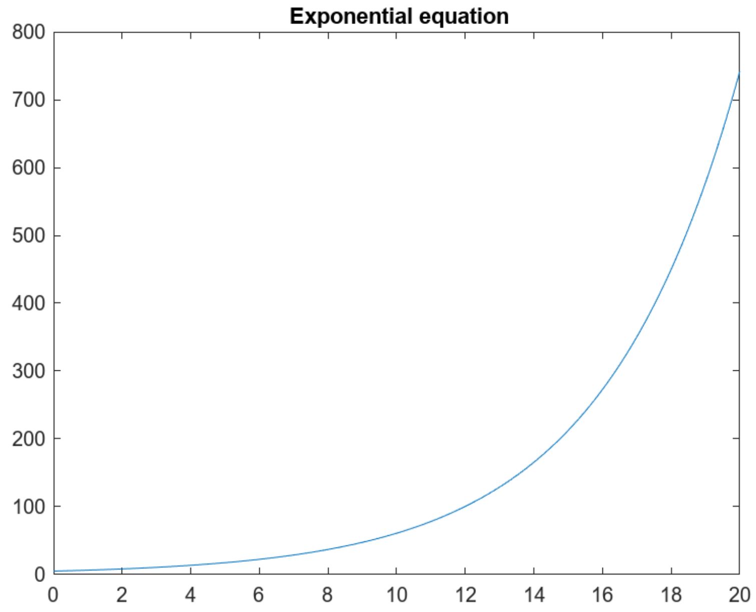
Linearization of nonlinear relationships

- Linear regression for fitting a best line to data is suitable if the relationship between the dependent and independent variables is linear.

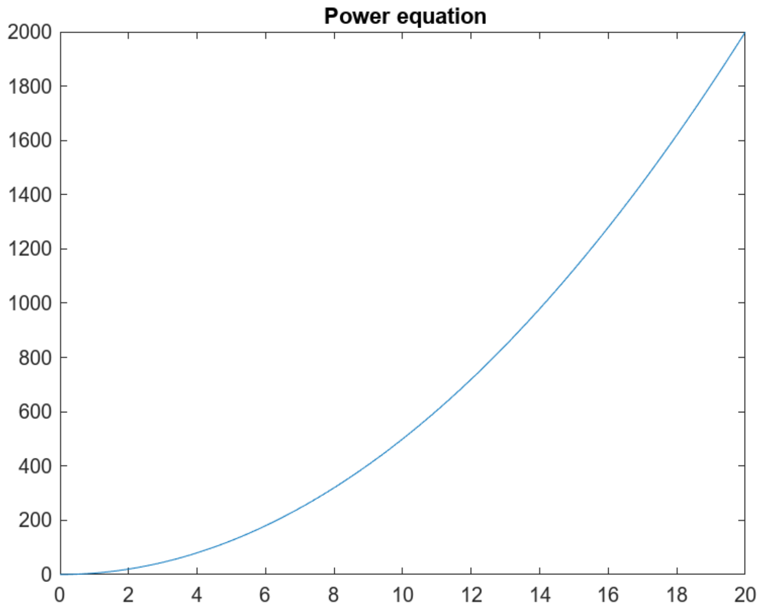
- For nonlinear relationships, techniques such as polynomial regression should be explored.

- However, for some nonlinear relationships, transformation techniques can be used to express the data in a form that is compatible with linear regression.

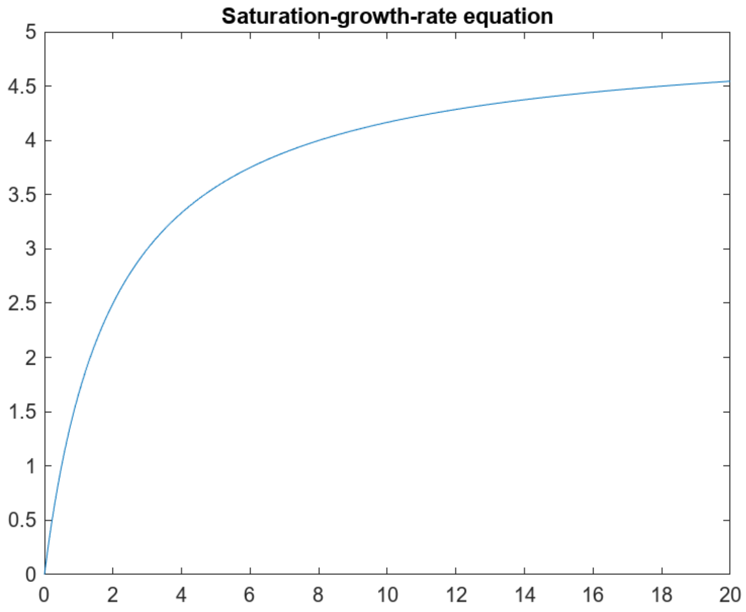
Examples of nonlinear relationships that can be linearized



$$y = \alpha_1 e^{\beta_1 x}$$



$$y = \alpha_2 x^{\beta_2}$$



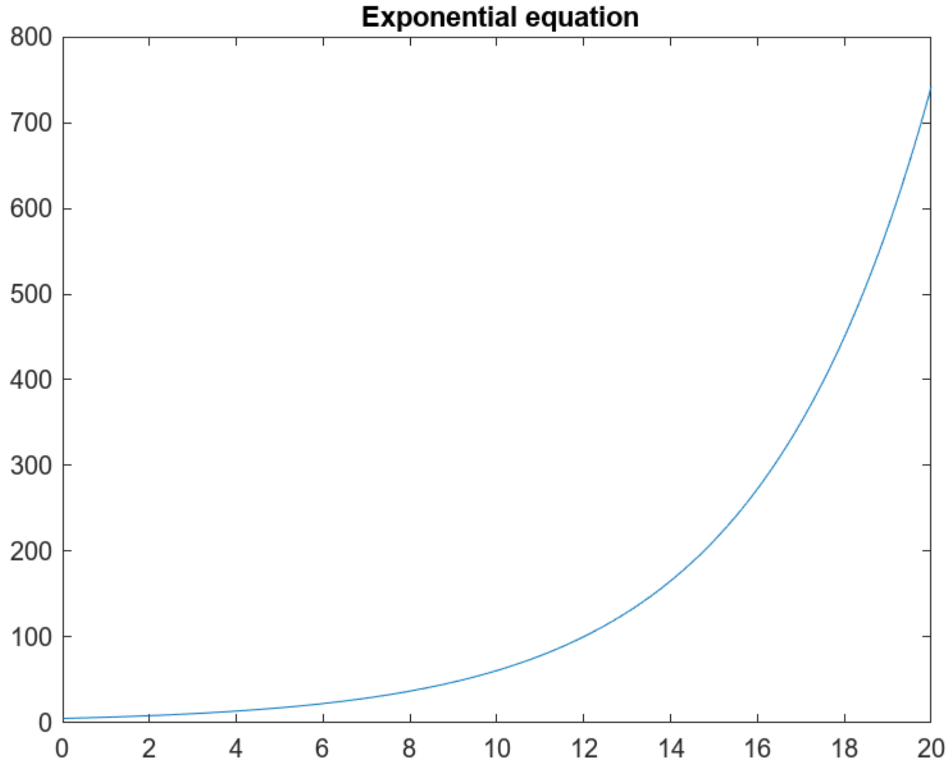
$$y = \alpha_3 \frac{x}{\beta_3 + x}$$

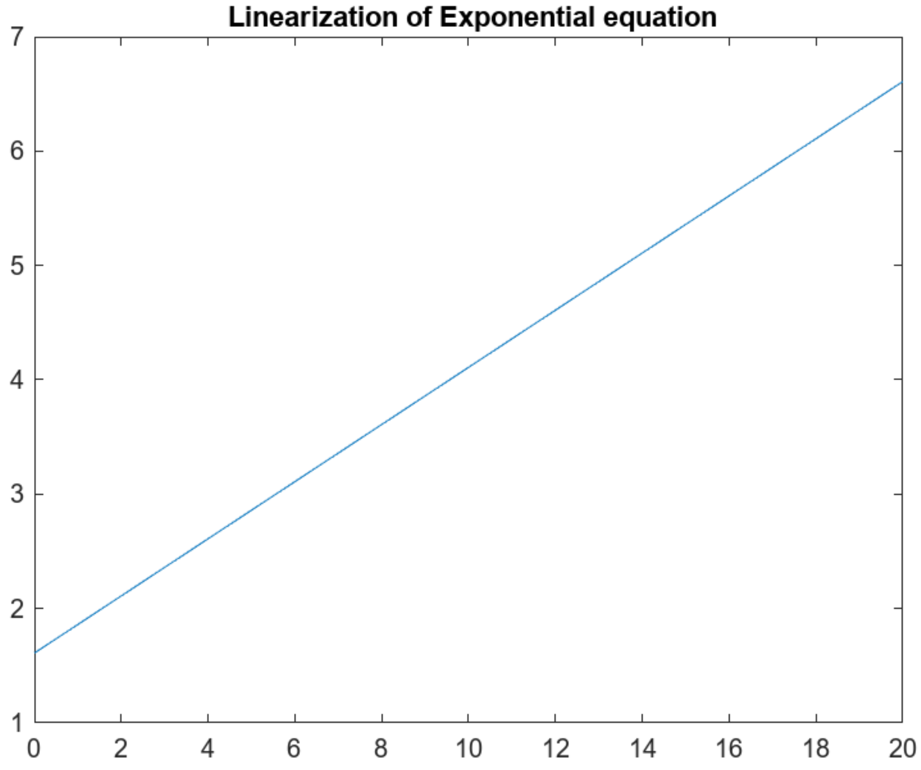
- There are nonlinear regression techniques that can be used to fit these equations to data directly.
- However, a simpler alternative is to use mathematical manipulations to transform the equations into a linear form.
- Then linear regression can be employed to fit the equations to data.

Linearization of exponential equation

Linearized by taking its natural logarithm

$$y = \alpha_1 e^{\beta_1 x} \qquad \rightarrow$$

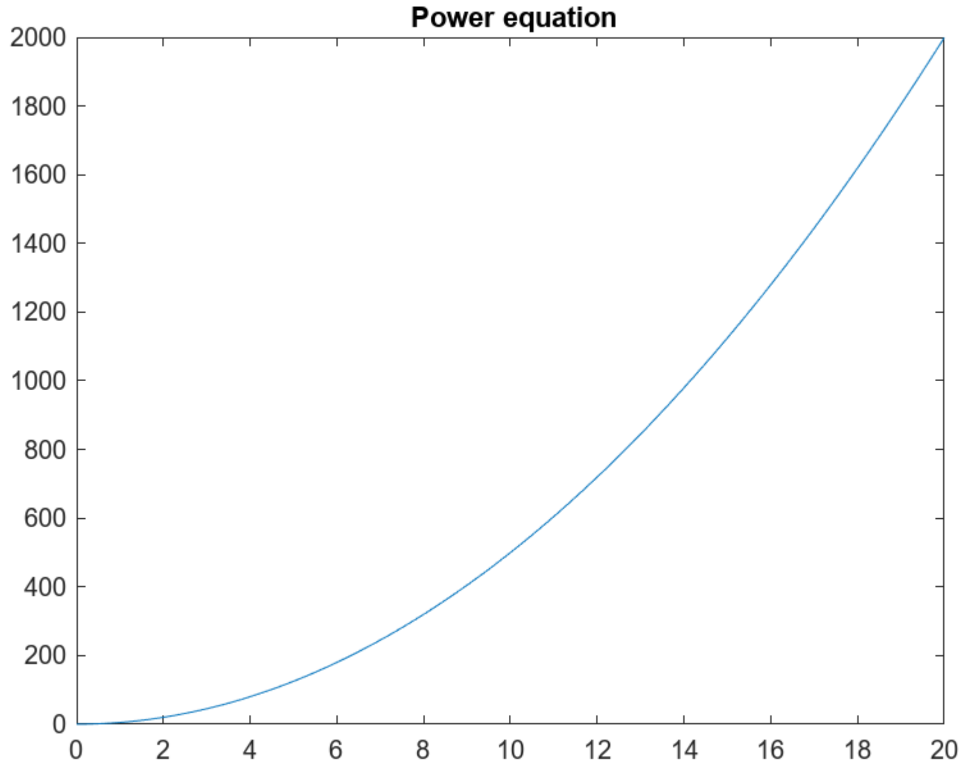


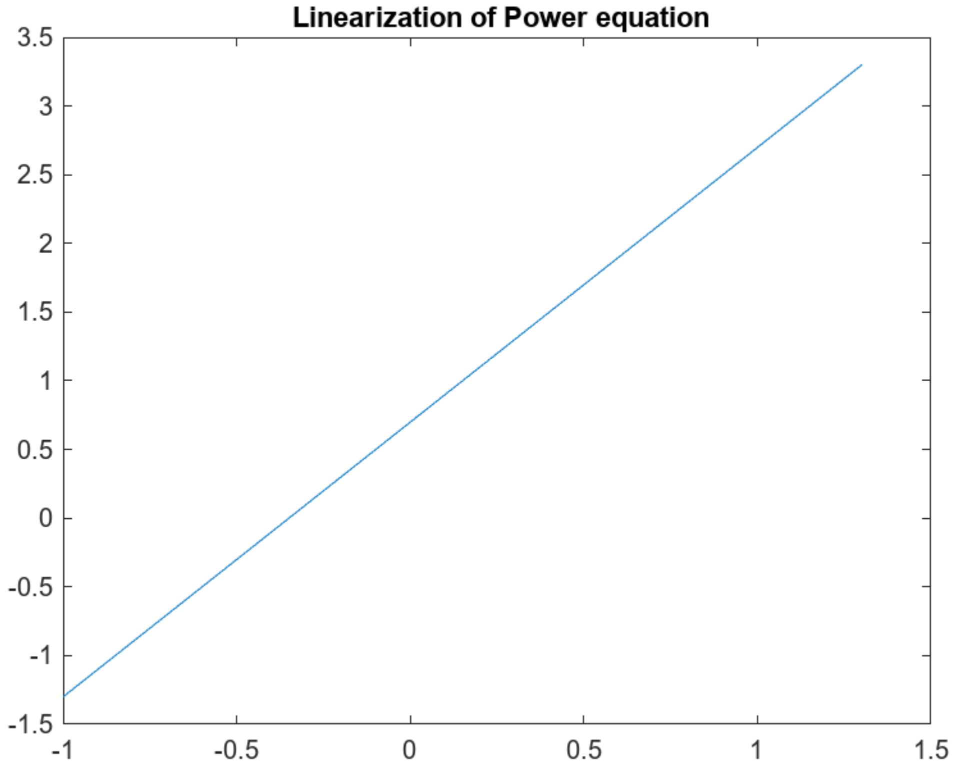


Linearization of power equation

Linearized by taking its base-10 logarithm

$$y = \alpha_2 x^{\beta_2} \qquad \rightarrow$$

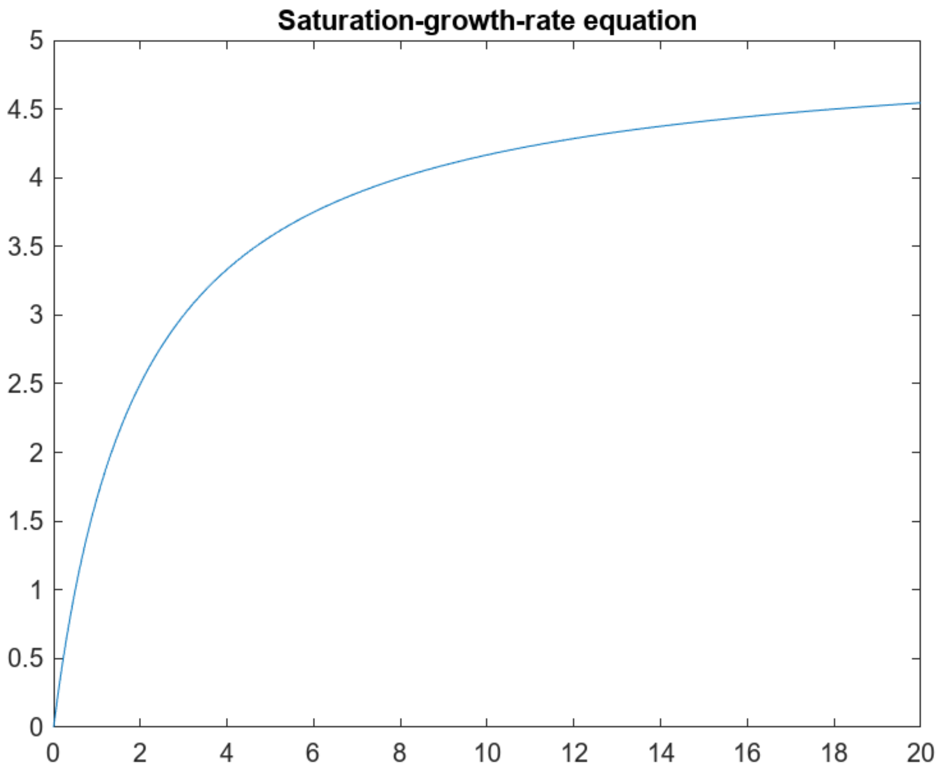


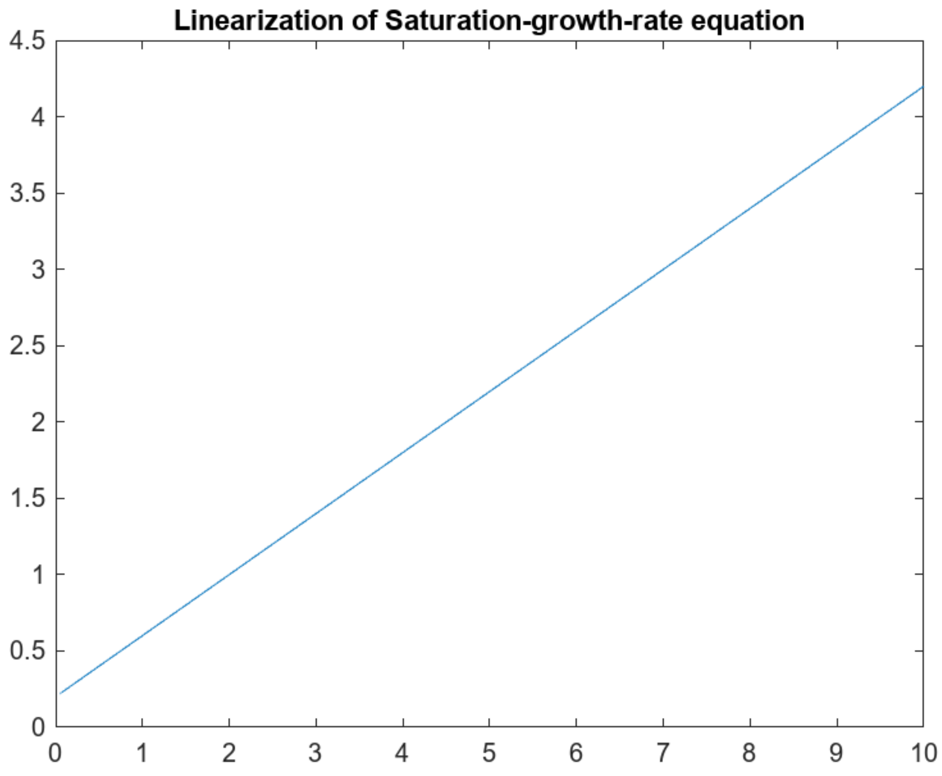


Linearization of saturation-growth-rate equation

Linearized by inverting

$$y = \alpha_3 \frac{x}{\beta_3 + x} \quad \rightarrow$$





Linearization of nonlinear relationships

$$y = a_0 + a_1 x$$

$$y = \alpha_1 e^{\beta_1 x} \rightarrow$$

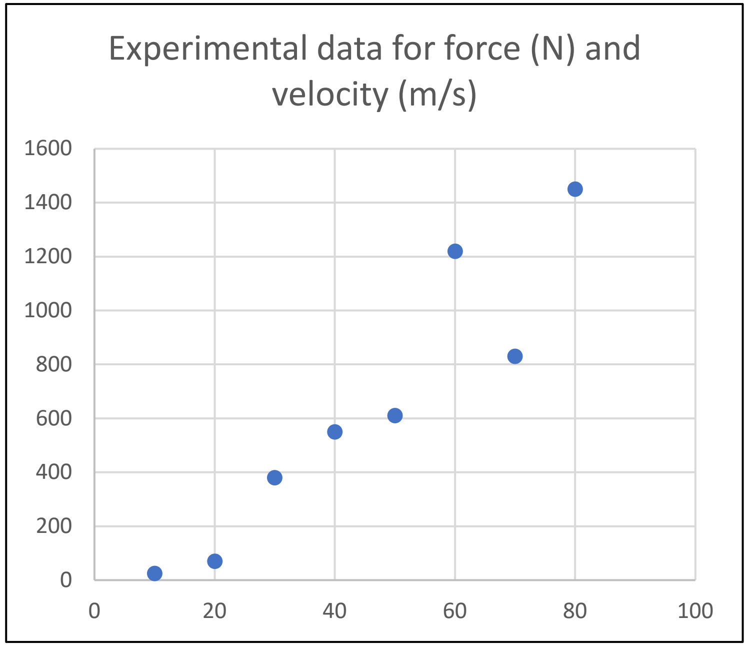
$$y = \alpha_2 x^{\beta_2} \rightarrow$$

$$y = \alpha_3 \frac{x}{\beta_3 + x} \rightarrow$$

Example : A free-falling object problem setup

- A free-falling object which is subject to the upward force of air resistance. A simple approximation is that the force was proportional to the square of velocity

$$F_U = c_d v^2$$



F_U = the upward force of air resistance $[N = kg\, m/s^2]$

c_d = a drag coefficient $[kg/m]$

v = velocity $[m/s]$

	Experimental data for force (N) and velocity (m/s)							
v	10	20	30	40	50	60	70	80
F	25	70	380	550	610	1220	830	1450

Example : Fitting free-falling object 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
$\sum$			12.606	20.515	20.516	33.622

Example : Fitting free-falling object data with the power equation

- Means

$$\bar{x} =$$

$$\bar{y} =$$

- Slope (a_1) and intercept (a_0)

$$a_1 =$$

$$a_0 =$$

- Least-squares fit

$$\log y =$$

- Coefficients of the power equation

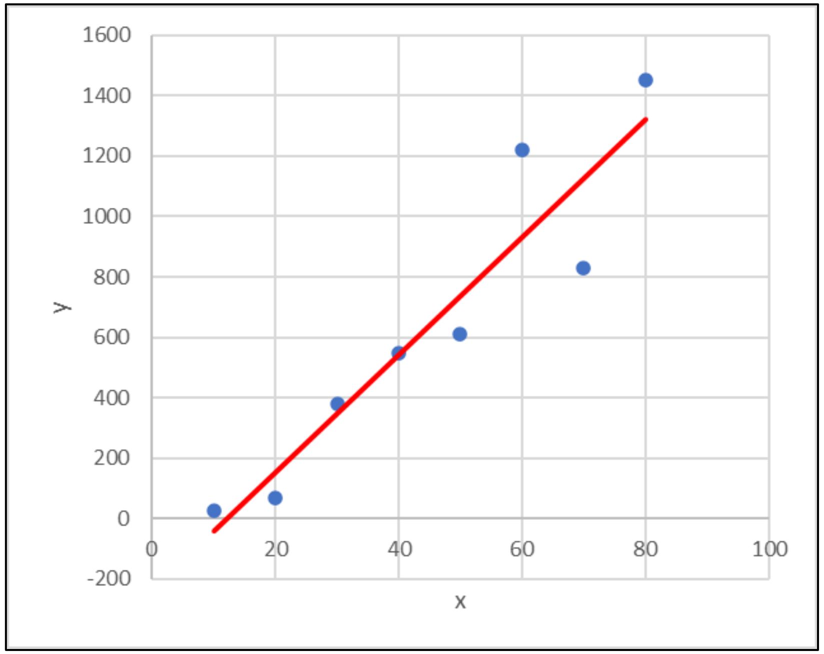
$$\alpha_2 =$$

$$\beta_2 =$$

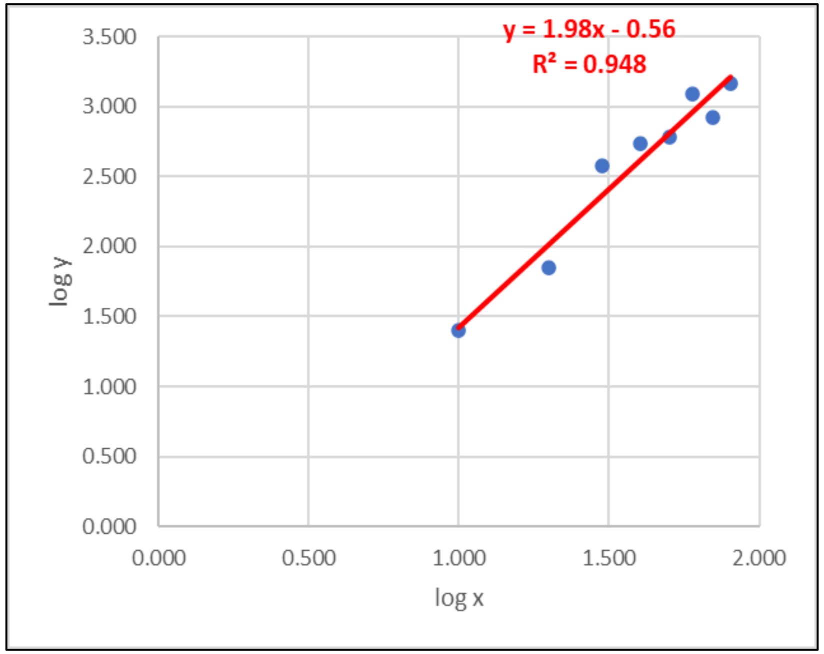
- Force equation

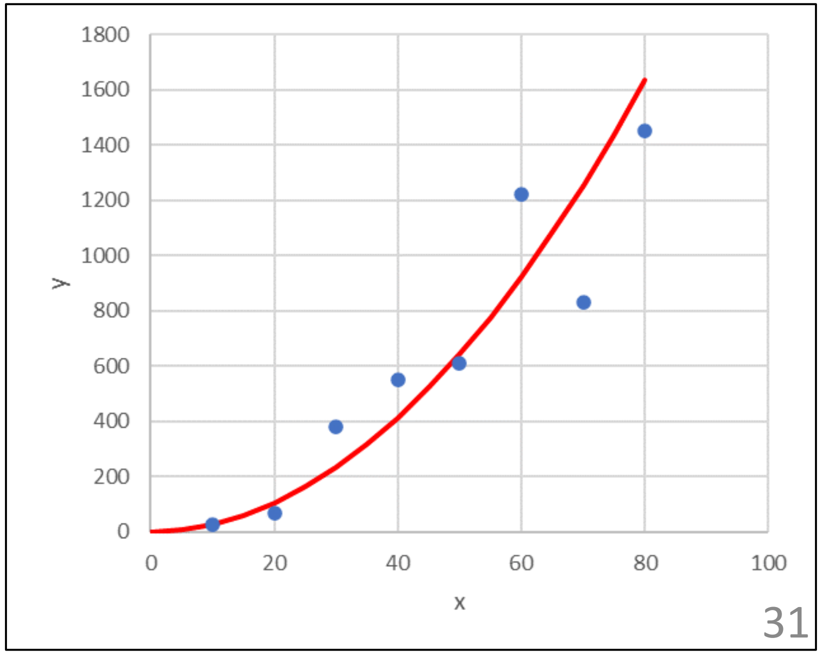
$$F =$$

Least-squares fit of the original data



Least-squares fit of the transformed data

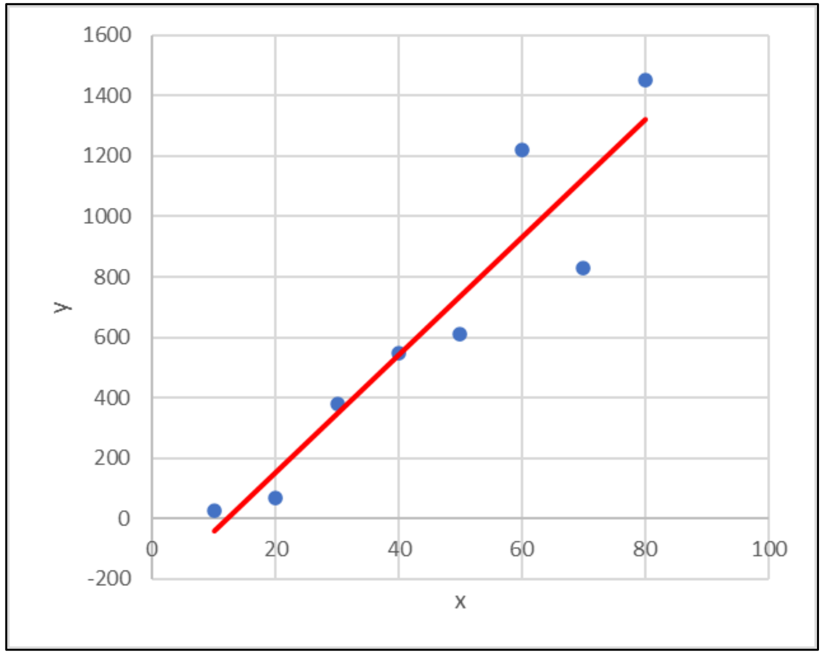


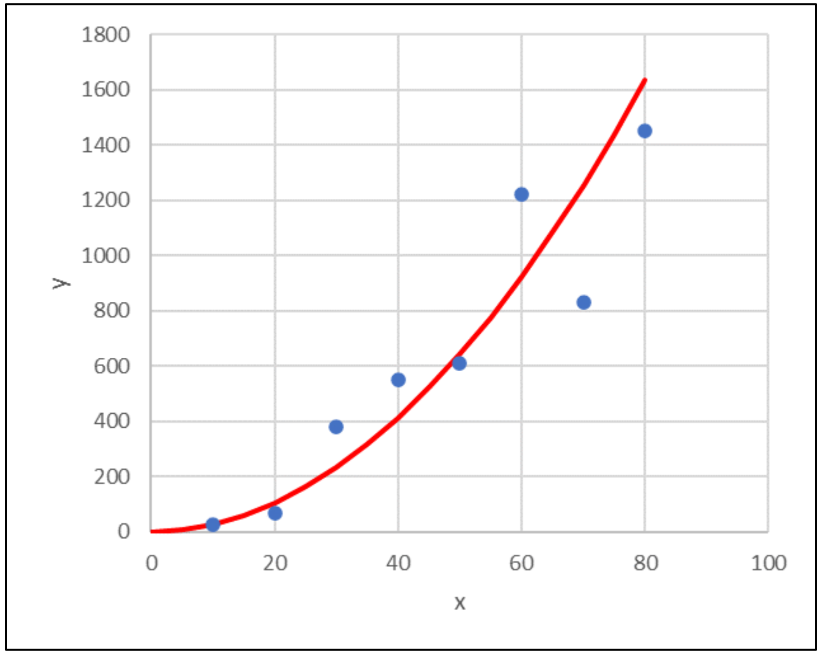


Lessons learned

- The transformed result has the advantage that it does not yield negative force predictions at low velocities.

- If we have knowledge from the field of fluid mechanics that suggests that the drag force of an object moving through a fluid is described by a model with velocity squared, then it can help us select the more appropriate model for curve fitting.





An M-file to implement linear regression

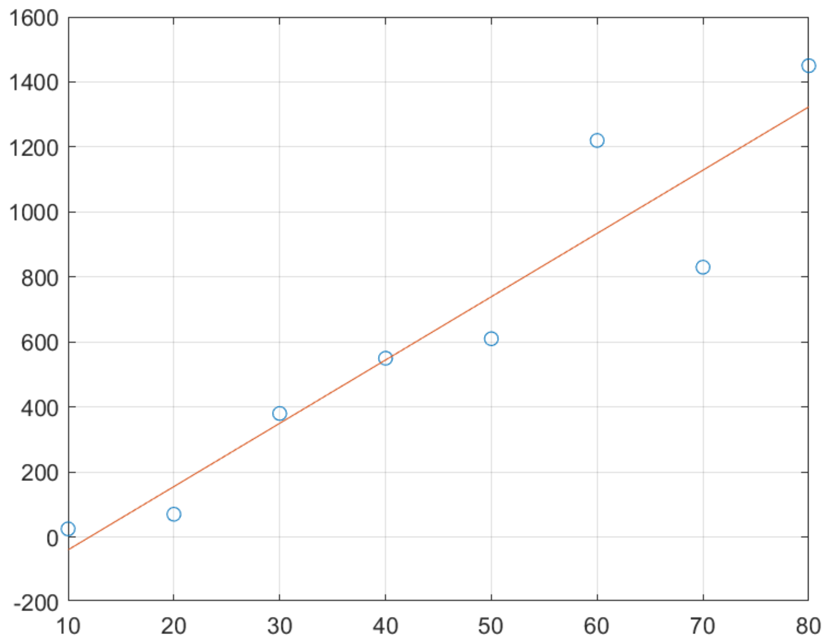
```
function [a, r2] = linregr(x,y)
% linregr: linear regression curve fitting
%   [a, r2] = linregr(x,y): Least squares fit of straight
%           line to data by solving the normal equations

% 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
```

MATLAB M-file : linregr

```
x = [10 20 30 40 50 60 70 80];
y = [25 70 380 550 610 1220 830 1450];
[a, r2] = linregr(x,y)
```



MATLAB Functions

MATLAB has a built-in function `polyfit` that fits a least-squares nth order polynomial to data:

`p = polyfit(x, y, n).`

- `x`: independent data.
- `y`: dependent data.
- `n`: order of polynomial to fit.
- `p`: coefficients of polynomial

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

MATLAB's `polyval` command can be used to compute a value using the coefficients.

`y = polyval(p, x).`