

Week 8: Curve Fitting

Acknowledgement: Un-copyrighted photos and pictures were taken from the Internet

Prediction

- We often need to predict quantities.
- **Examples:** position of a vehicle.
- **Examples:** the price of a house.
- **Examples:** the population of a country.
- **Examples:** oil price.
- **Example:** the electricity price.

Prediction

- It is **easy** to **predict** certain **quantities**.
- It is **easy** to **predict** the **position of a vehicle**.



Prediction

- It is easy to predict certain quantities.
- The position of a vehicle can be **predicted** using **Newton's laws**.

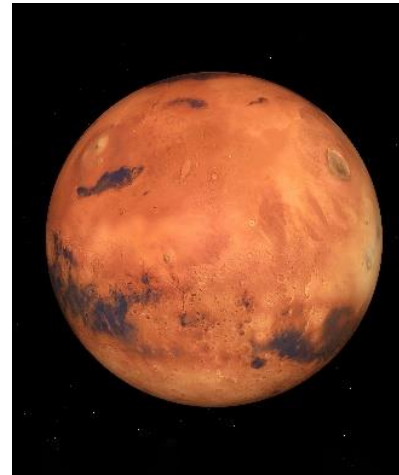


Newton's Laws
of Motion

position

Prediction

- It is **easy** to **predict** certain **quantities**.
- **When** does a **spacecraft** arrive at **Mars**?



Prediction

- It is **easy** to **predict** certain **quantities**.
- **Physical laws** can be used to **predict** the **travel time** to **Mars**.



Physical Laws



Travel time

Prediction

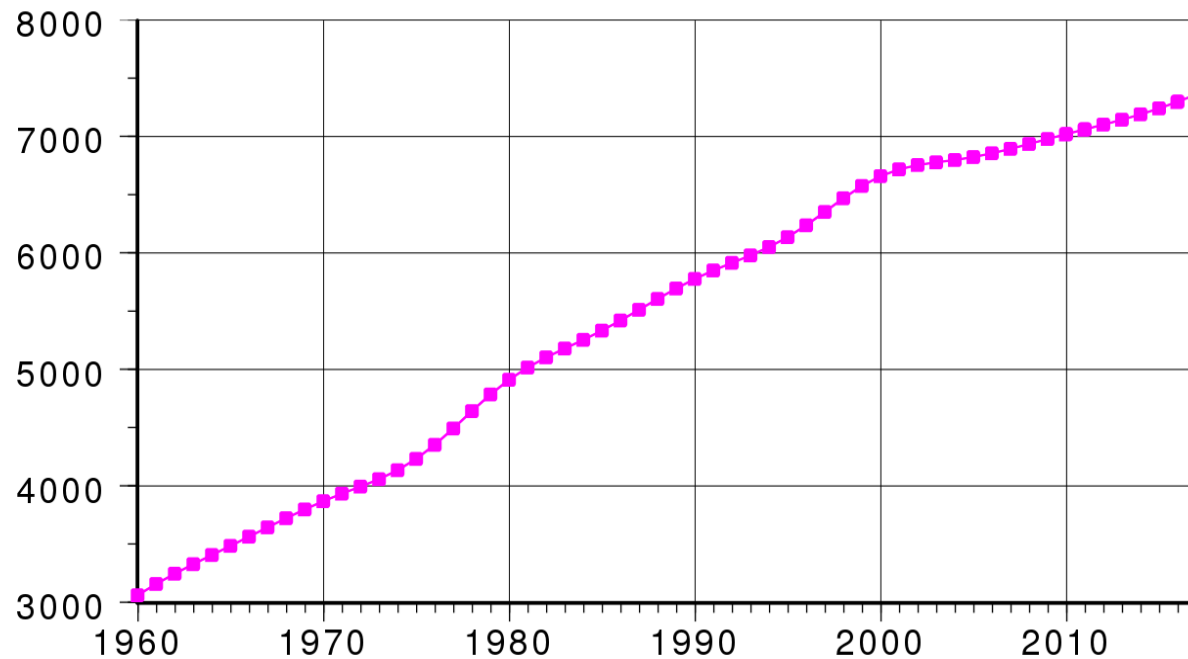
- It is **easy** to **predict** certain quantities.
- They **obey** certain **mathematical or physical rules**.
- We can use these **rules** to **predict** them.

Prediction

- It is difficult to predict certain quantities.
- Example: the population of Hong Kong in 2030.
- Example: the market price of a house.
- Example: the electricity price.
- There is no clear rule to predict them.
- Sometimes, the rules (laws) are too complex.
- So, they cannot be used for prediction.

Prediction

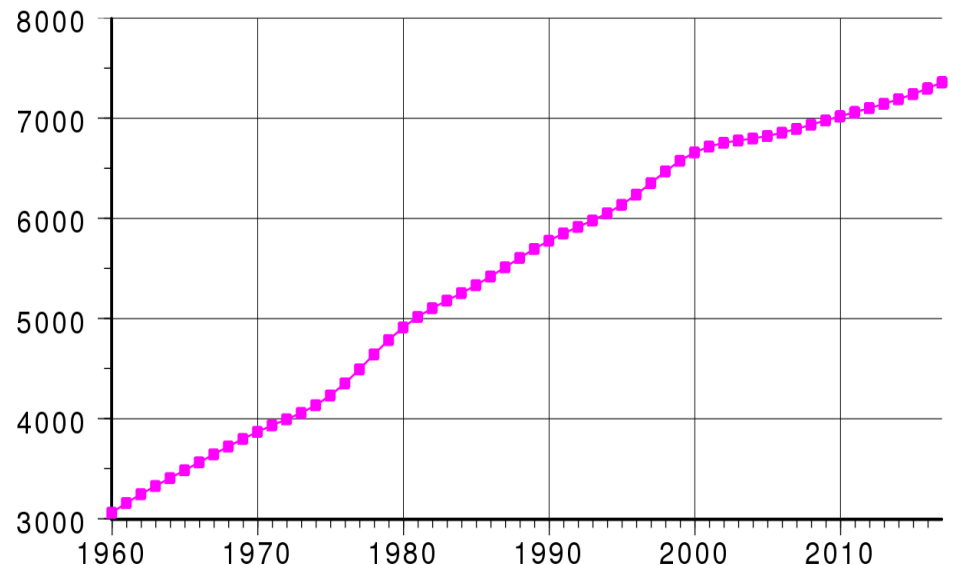
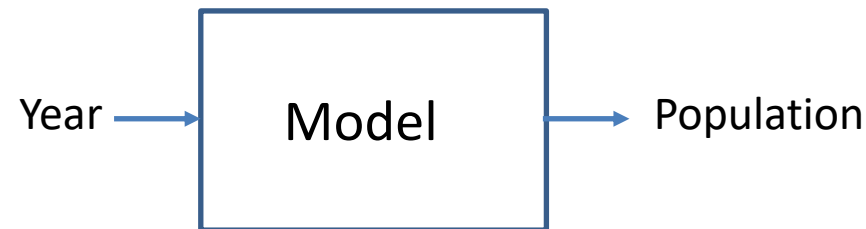
- Example: the **population of Hong Kong** in **2030**.
- There are no explicit **rules**. But, there are **many historical observations** (data).



HK population (in thousand) in different years

Prediction

- Using the **historical data**, we can build a **model** to **predict** the **population**.
- The **input** to the **model** is the **year**. The **output** is the **population**.



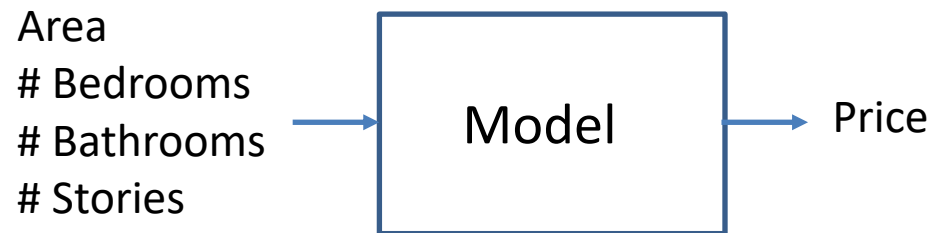
Prediction

- Example: the market price of a house.
- The price of a house depends on its area, neighborhood, etc.
- There are no explicit rules. But, there are many data available.

area	bedrooms	bathrooms	stories	price
7420	4	2	3	13300000
8960	4	4	4	12250000
9960	3	2	2	12250000
7500	4	2	2	12215000
7420	4	1	2	11410000

Prediction

- Using **data**, we can **build** a **model** to **predict** the **price** of a house.
- The **input** to the model will be the **area, number of bedrooms, bathrooms, stories**. The **price** will be the **output**.

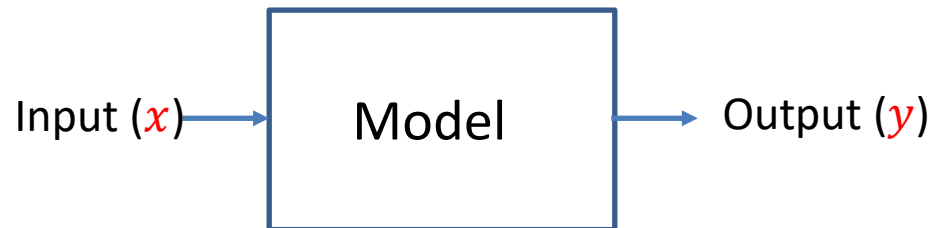


area	bedrooms	bathrooms	stories	price
7420	4	2	3	13300000
8960	4	4	4	12250000
9960	3	2	2	12250000
7500	4	2	2	12215000
7420	4	1	2	11410000

Models

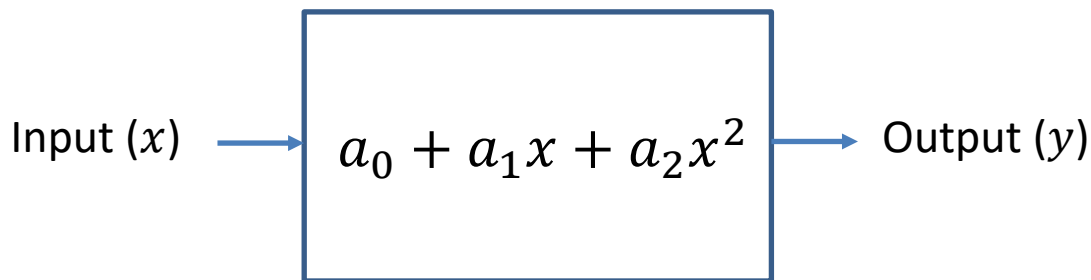
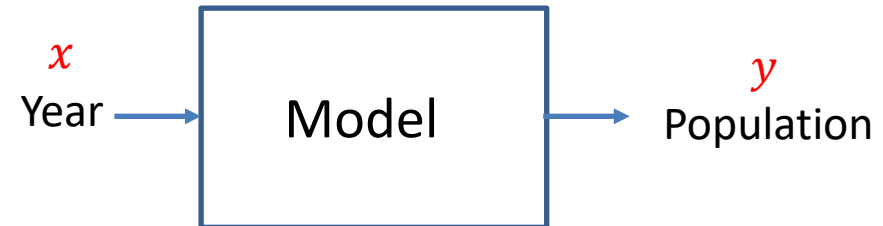
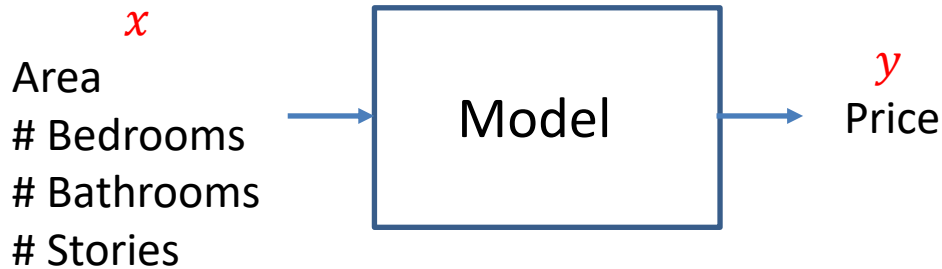
What is a model?

- Suppose that we would like to **predict** a **quantity**. Let's call it y .
- The **value of y** is **predicted** based on the **values of other quantities**. Let's call them x .
- A **model** is a **mathematical function** of x . The model will predict the value of y based on x .
- x will be the **input** of the **model**, and y will be the **output** of the **model**.



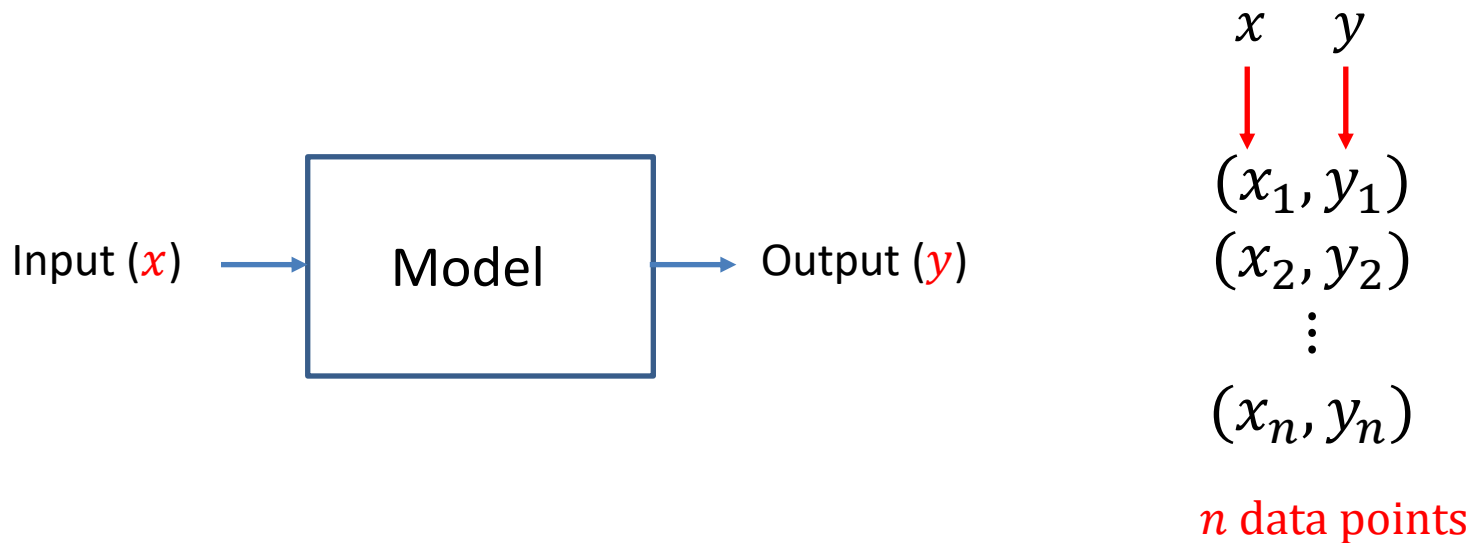
What is a model?

- Examples of models.



What is a model?

- Suppose that we would like to build a model to predict y based on x .
- We have n data points. Each data point has two components: x and y .
- We will build mathematical models based on the available data to predict the values of y .



What is a model?

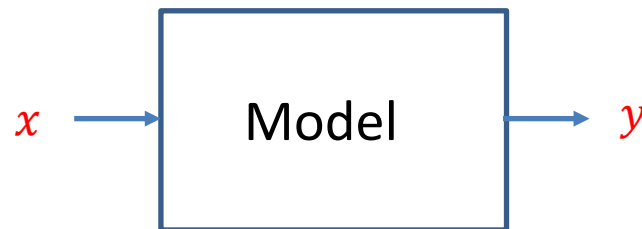
- Think of x as the **year** and y as the **population of Hong Kong in that year**.

x (year)	y (population)
1960	3000000
$\vdots$	$\vdots$
1980	5000000
$\vdots$	$\vdots$
2010	7000000

x	y
x_1	y_1
x_2	y_2
$\vdots$	$\vdots$
x_n	y_n

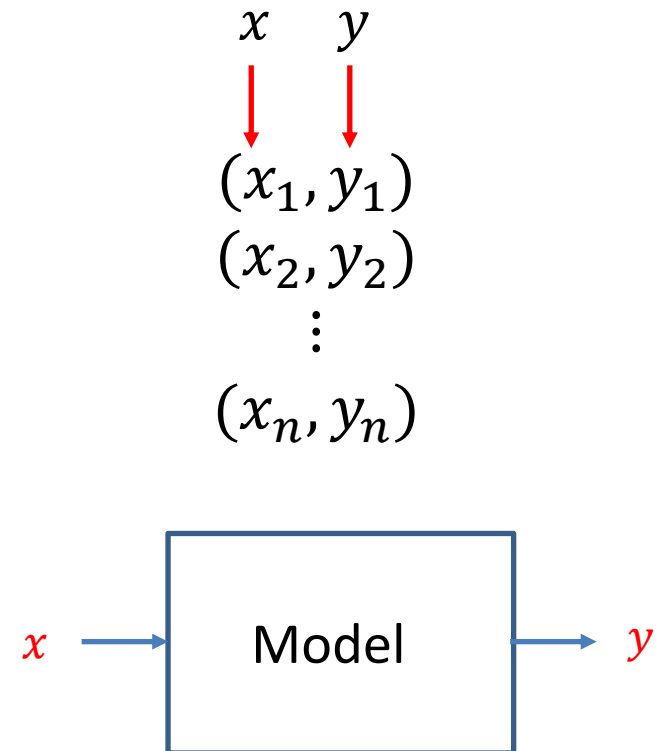
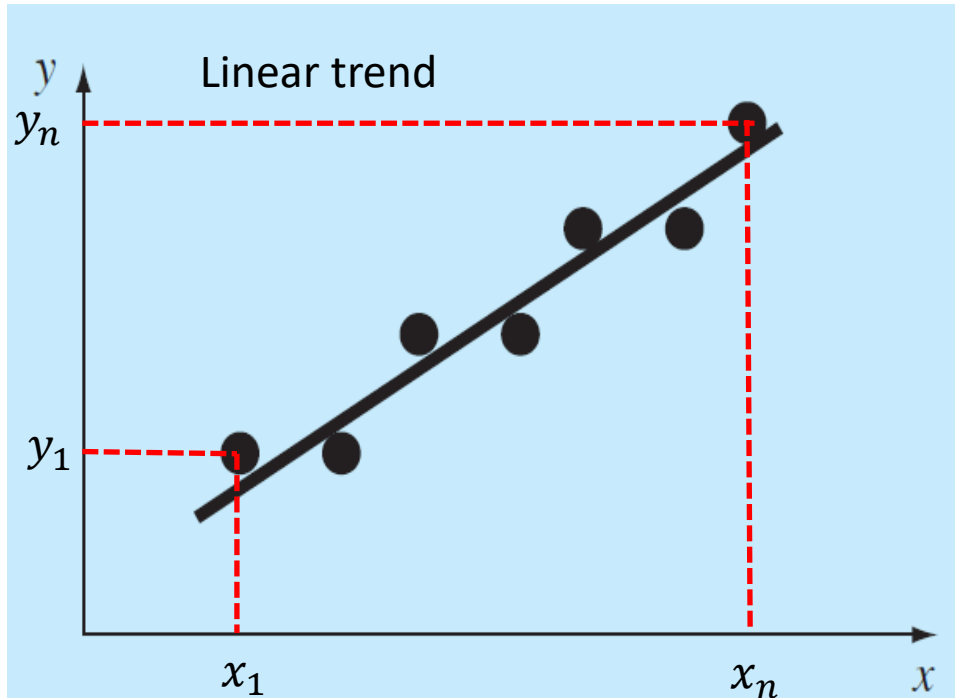
- We would like to **predict the value of y (population) for other values of x (year)**.
- Example: **population in 2030**. That is, the **value of y for $x = 2030$** .

Linear Least-Squares Regression



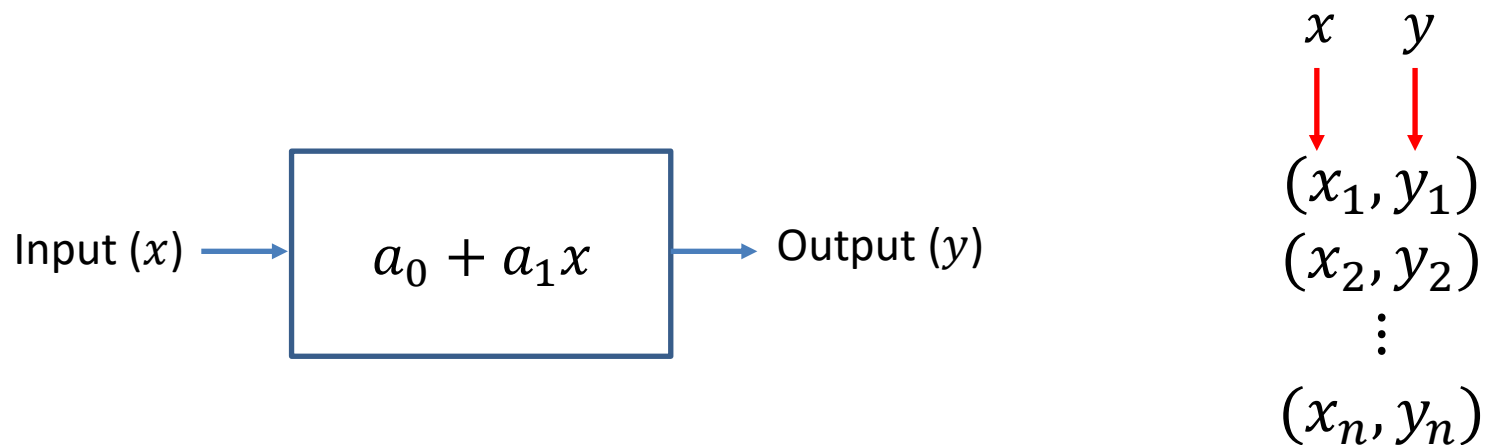
Linear Least-Squares Regression

- Suppose that we have n data points.
- For certain data, the relationship between x and y can be described by a line.



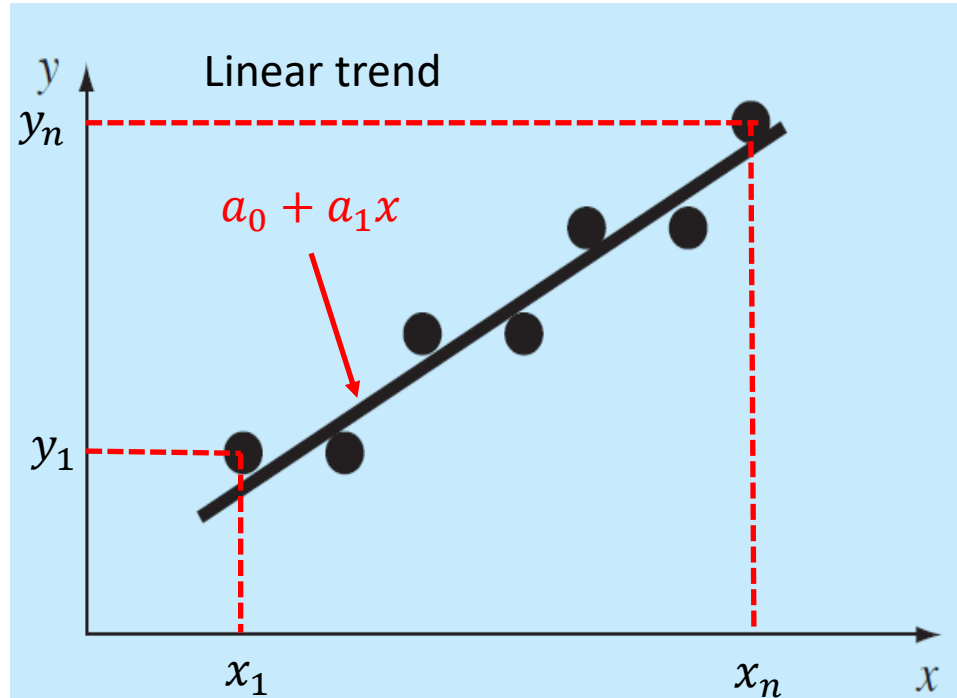
Linear Least-Squares Regression

- In the **linear least-squares method**, we build a **linear model** to **predict** the **value of y** .
- A **linear** model will be in the form of $a_0 + a_1x$. Here, a_0 and a_1 are the **parameters** of the **model**.
- If we know a_0 and a_1 we can predict the value of y for different **values of x** .

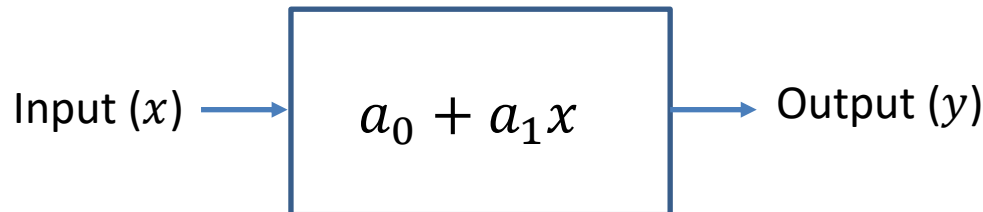


Linear Least-Squares Regression

- The linear model can be used to predict the value of Y for points that we do not have measurements.

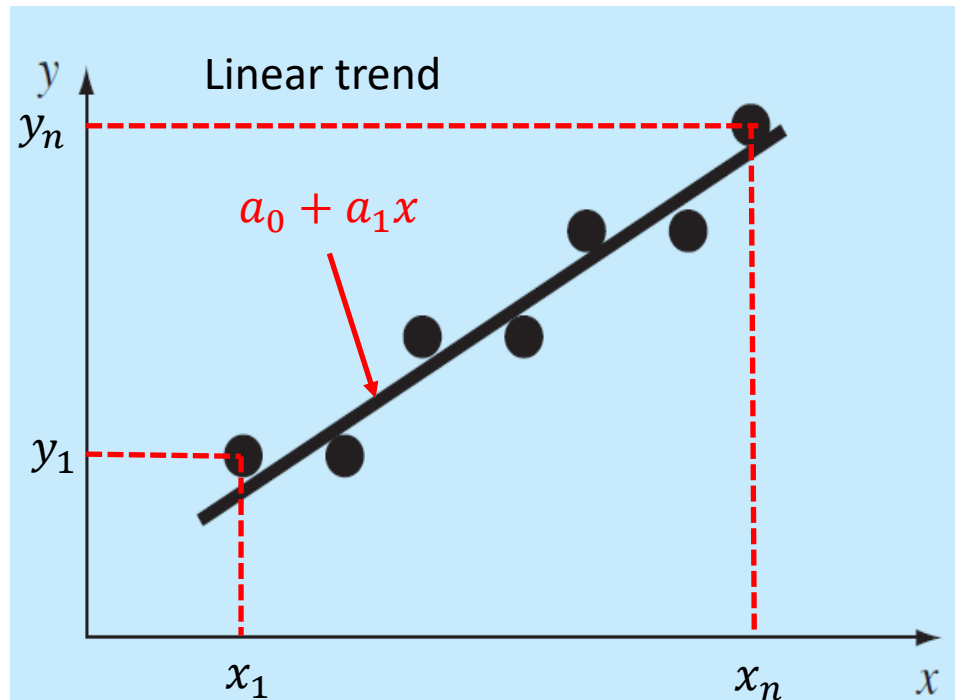


x y
↓ ↓
 (x_1, y_1)
 (x_2, y_2)
⋮
 (x_n, y_n)

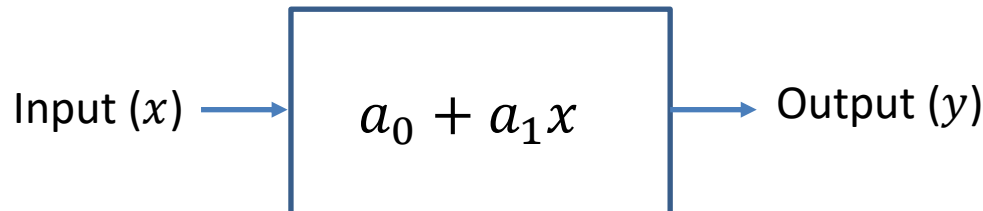


Linear Least-Squares Regression

- We say, a **linear model** is **fitted** to the **data**.



x y
↓ ↓
 (x_1, y_1)
 (x_2, y_2)
⋮
 (x_n, y_n)



Linear Least-Squares Regression

- The values of a_0 and a_1 are computed as follows.

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

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

- $$\sum_{i=1}^n x_i = x_1 + x_2 + \cdots + x_n$$

- $$\sum_{i=1}^n y_i = y_1 + y_2 + \cdots + y_n$$

- $$\sum_{i=1}^n x_i^2 = x_1^2 + x_2^2 + \cdots + x_n^2$$

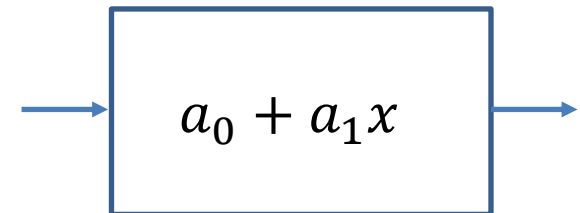
- $$\sum_{i=1}^n x_i y_i = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

(x_1, y_1)

(x_2, y_2)

$\vdots$

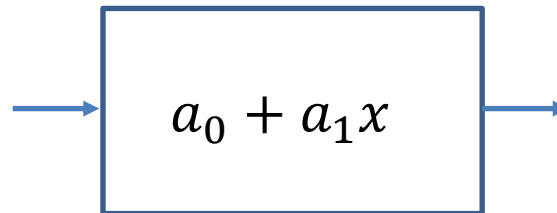
(x_n, y_n)



Example

- Find the **linear least-squares** model for the following data.

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



Example

- Solution

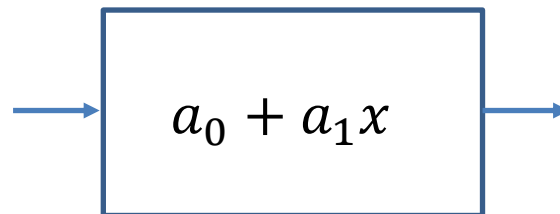
$$n = 7 \quad \sum x_i y_i = 119.5 \quad \sum x_i^2 = 140$$

$$\sum x_i = 28$$

$$\sum y_i = 24$$

$$a_1 = \frac{7(119.5) - 28(24)}{7(140) - (28)^2} = 0.8392857$$

$$a_0 = 3.428571 - 0.8392857(4) = 0.07142857$$



How does it work?

- Suppose that a_0 and a_1 are known.

- We can compute the output of this model for each x_i

(x_1, y_1)

(x_2, y_2)

$\vdots$

(x_n, y_n)

$x_1,$

$x_2,$

$\dots$

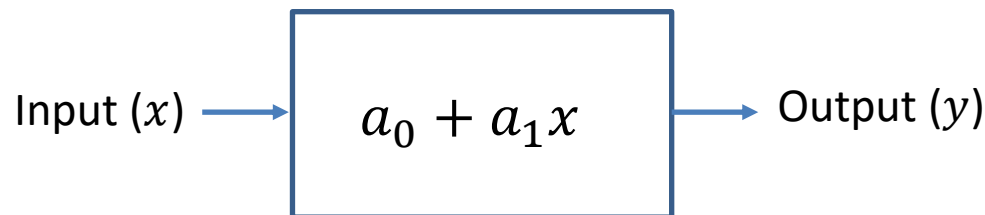
x_n

$a_0 + a_1 x_1,$

$a_0 + a_1 x_2, \dots, a_0 + a_1 x_n$

Output of
the model

- The output of the model for point x_i may be different from y_i (data).



How does it work?

- The error of the model for point x_i is $y_i - (a_0 + a_1 x_i)$.
- Data Model output

$x_1,$

$x_2,$

...

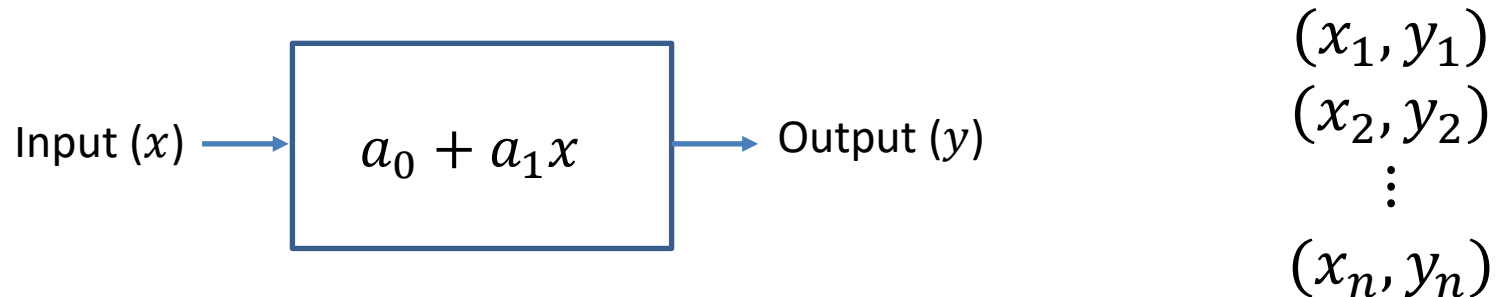
x_n

$a_0 + a_1 x_1,$

$a_0 + a_1 x_2, \quad \dots,$

$a_0 + a_1 x_n$ ← Output of the model

$y_1 - (a_0 + a_1 x_1), \quad y_2 - (a_0 + a_1 x_2), \quad \dots, \quad y_n - (a_0 + a_1 x_n)$



How does it work?

- The **error** of the **model** for point x_i is $y_i - (a_0 + a_1 x_i)$.

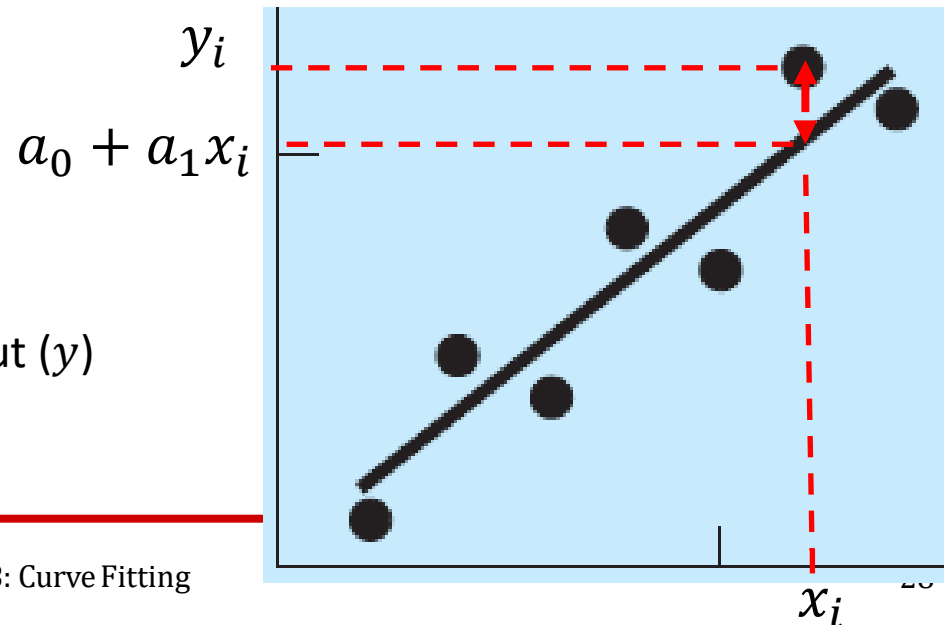
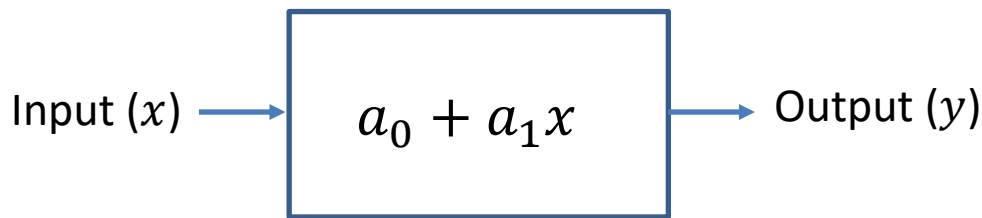
(x_1, y_1)

(x_2, y_2)

$\vdots$

(x_n, y_n)

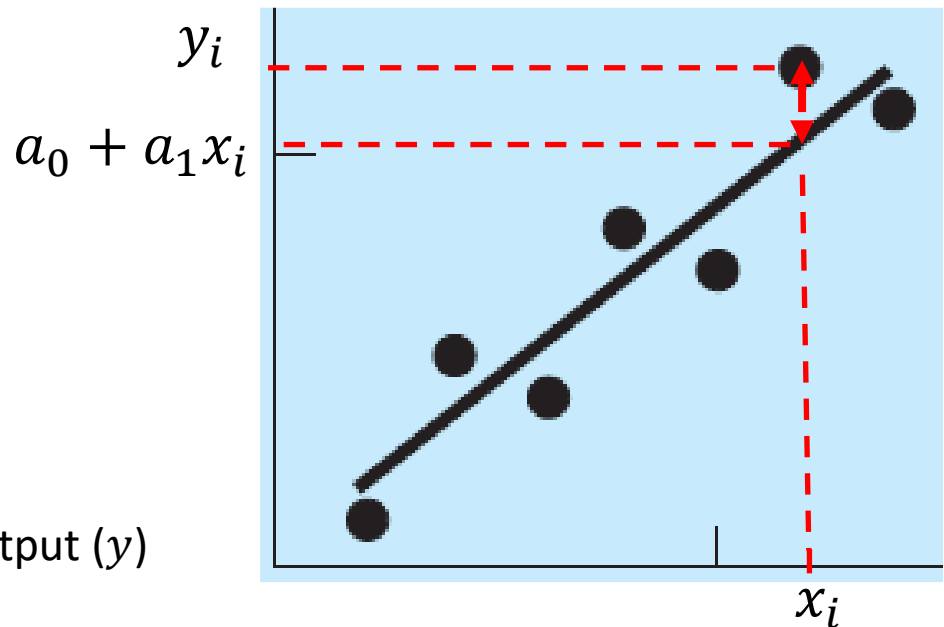
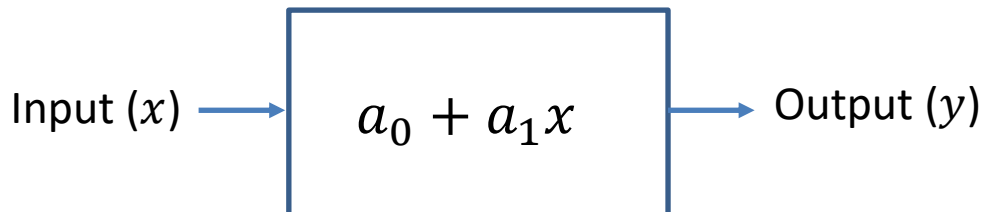
$$\begin{aligned} S_r &= \sum_{i=1}^n (y_i - a_0 - a_1 x_i)^2 \\ &= (y_1 - a_0 - a_1 x_1)^2 + \cdots + (y_n - a_0 - a_1 x_n)^2 \end{aligned}$$



How does it work?

- Every **model** (a_0 and a_1) results in an **error** (S_r). If we change a_0 and a_1 , the error of the model will change.
- In the **linear least-squares regressions**, we find a_0 and a_1 that result in the **smallest error**.

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

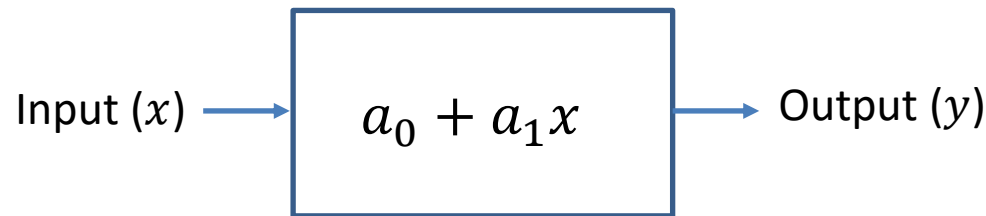


How does it work?

- This is called the **least-squares method** since we find the model that has the **smallest error** (S_r).
- It is called **linear** because the model is in the form of a **line**.

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

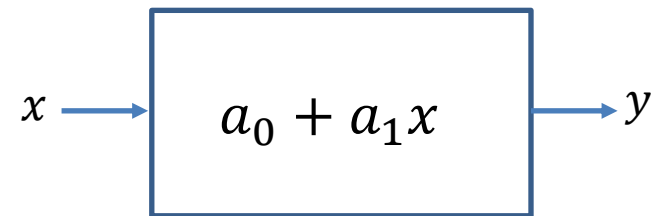
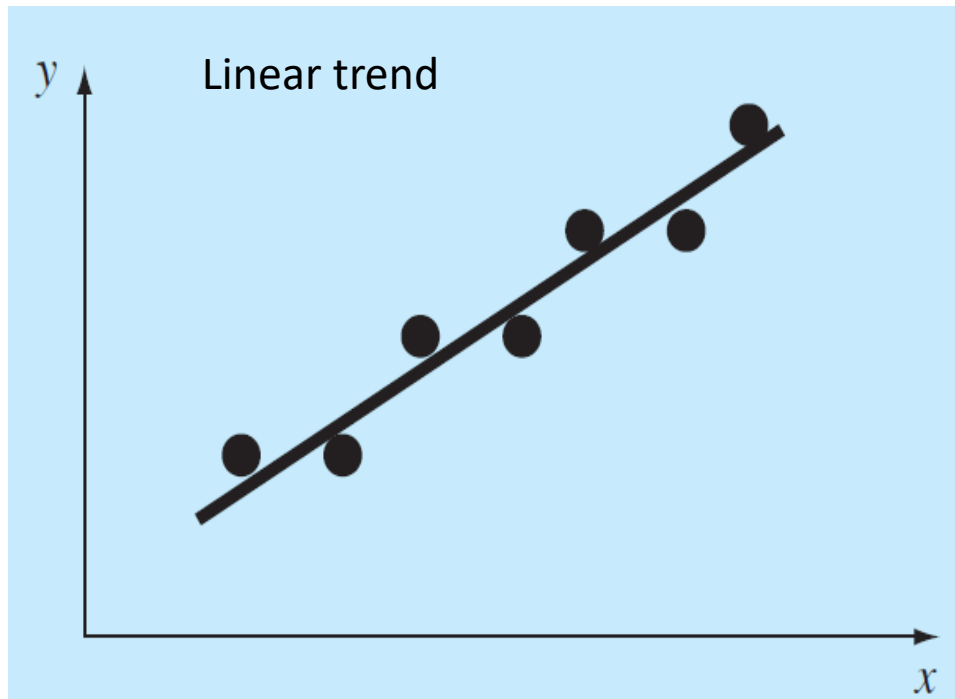
$$\begin{array}{c} (x_1, y_1) \\ (x_2, y_2) \\ \vdots \\ (x_n, y_n) \end{array}$$



Linearization of Nonlinear Relationships

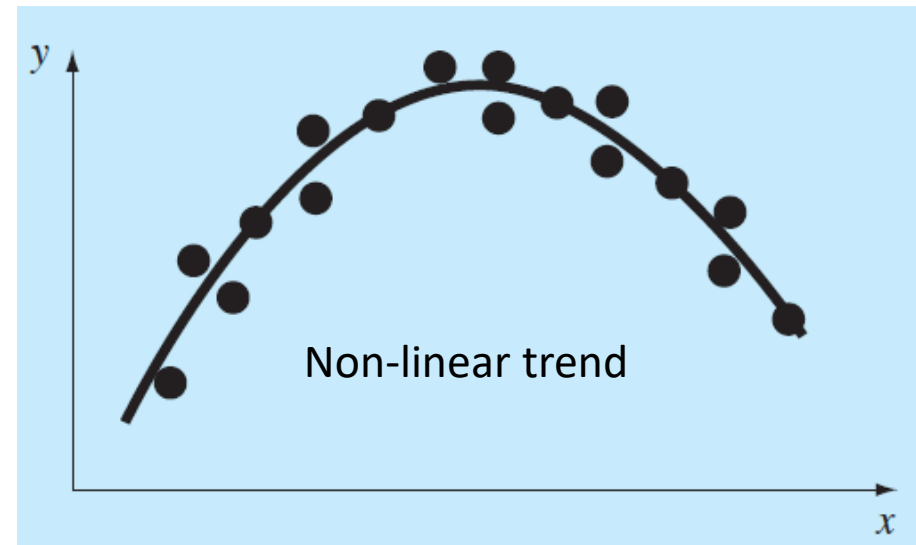
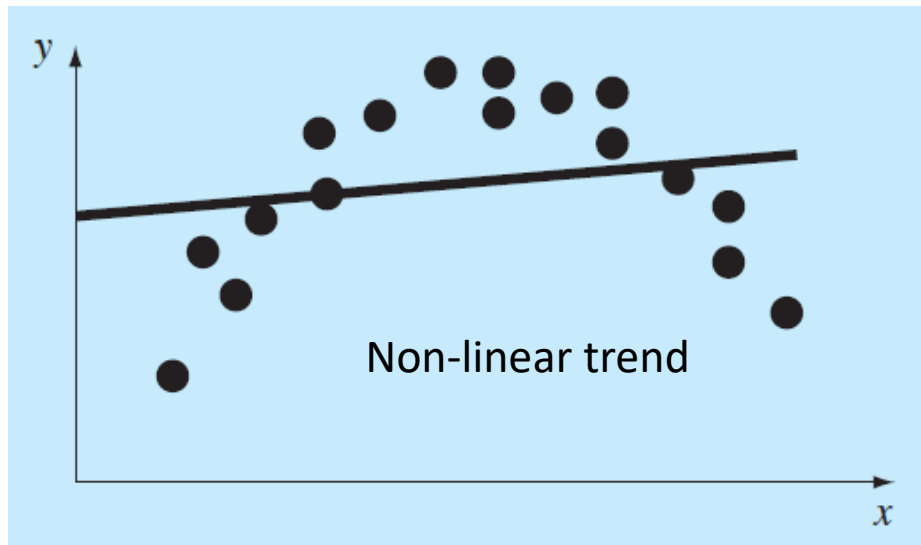
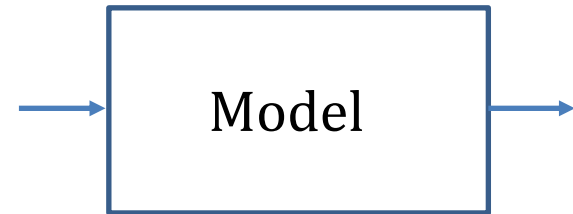
Linear Relationship

- **Linear** regression is **efficient** when the data points follow a **linear trend**.
- That is, the relationship between **x** and **y** can be described by **a line**.



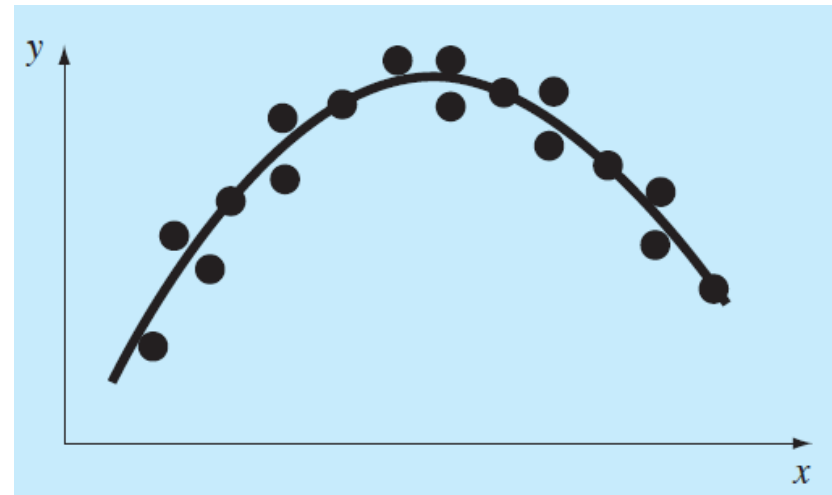
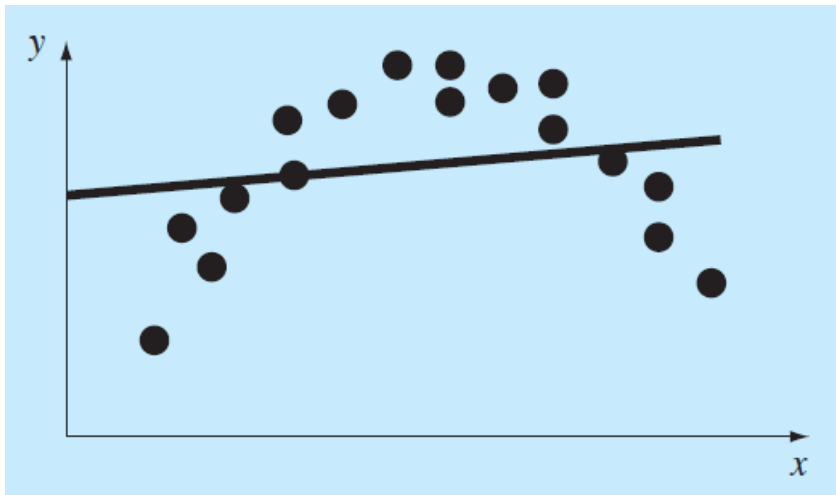
Non-linear Relationships

- Certain data **may not obey** the linear relationship.
- How to build a model in this case?



Linearization

- In some cases, **transformations** can be applied to **data** to obtain **linear relationships**.
- **After transformation**, the relationship becomes **linear**.
- So, **linear regression** can be applied to **data** after **transformation**.



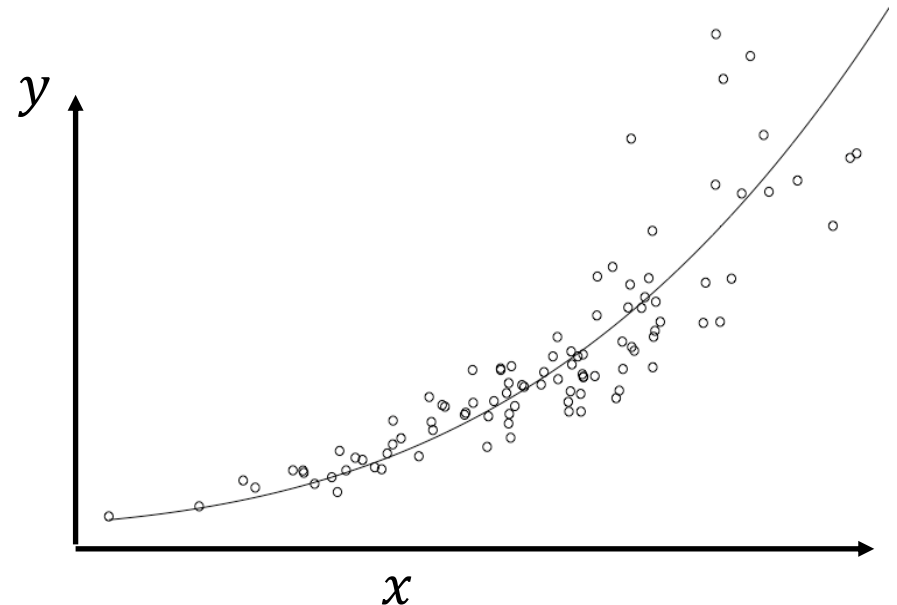
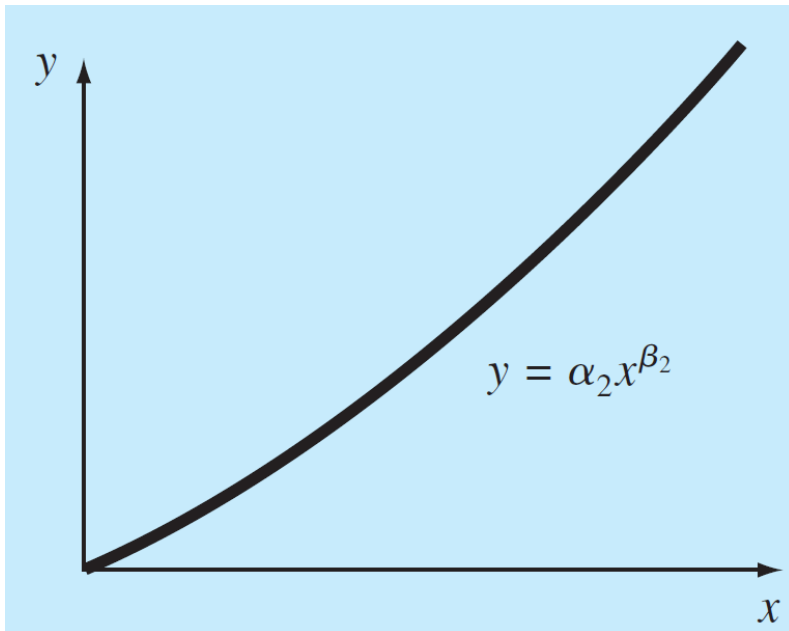
Linearization

- We will study these transformations for three different models:
 1. Power model
 2. Exponential models
 3. Saturation-Growth-Rate Model

Power Form

Power Form

- Sometimes the data follows a **power** trend.
- A **power model** is in the form of $y = \alpha_2 x^{\beta_2}$ where α_2 and β_2 are constant parameters.
- We need to find α_2 and β_2 .

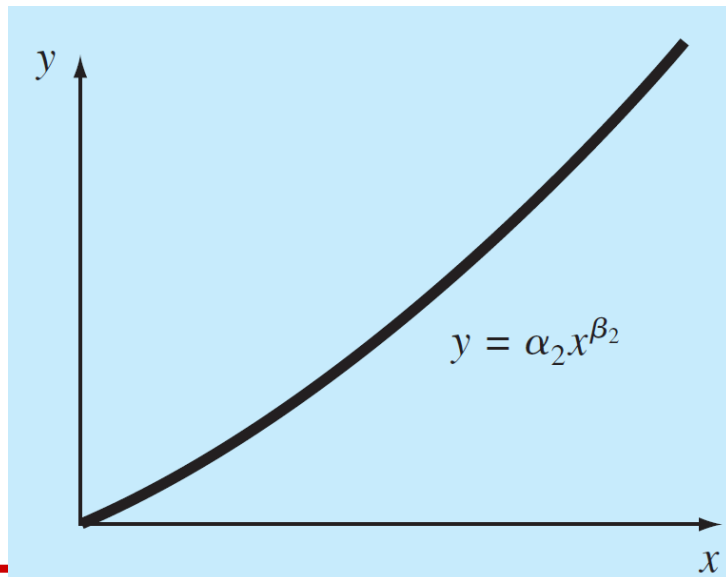


Power Form

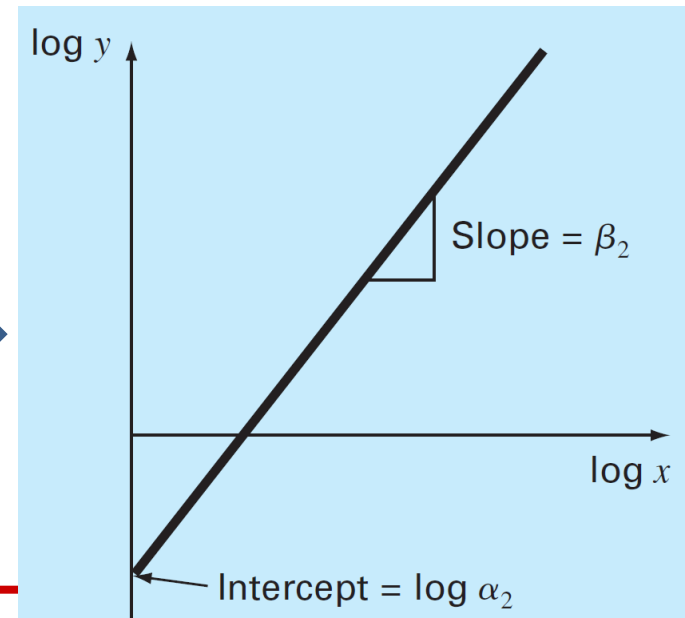
- data points with a power model can be linearized by taking the logarithm to the base 10 of x and y .

- $y = \alpha_2 x^{\beta_2} \Rightarrow \log_{10}(y) = \log_{10}(\alpha_2) + \beta_2 \log_{10}(x) \Rightarrow$

- $y' = \log_{10}(\alpha_2) + \beta_2 x'$



Linearization



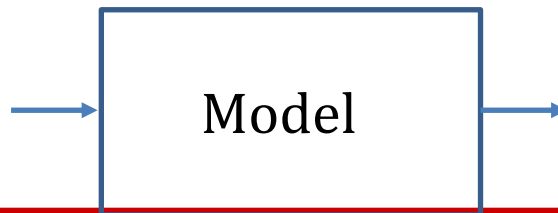
Power Form

- Power data can be **linearized** by taking the **logarithm to the base 10** of x and y in the data points.

Power model data	Linearized data	
	x'	y'
(x_1, y_1)	$(x'_1 = \log_{10}(x_1),$	$y'_1 = \log_{10}(y_1))$
(x_2, y_2)	$(x'_2 = \log_{10}(x_2),$	$y'_2 = \log_{10}(y_2))$
$\vdots$	$\vdots$	
(x_n, y_n)	$(x'_n = \log_{10}(x_n),$	$y'_n = \log_{10}(y_n))$

$$y = \alpha_2 x^{\beta_2} \quad \log_{10}(y) = \log_{10}(\alpha_2) + \beta_2 \log_{10}(x) \quad y' = \log_{10}(\alpha_2) + \beta_2 x'$$

y'
 x'



Power Form

- We then apply the **linear least-squares regression** to the **linearized data** and find **a_0** , and **a_1** .

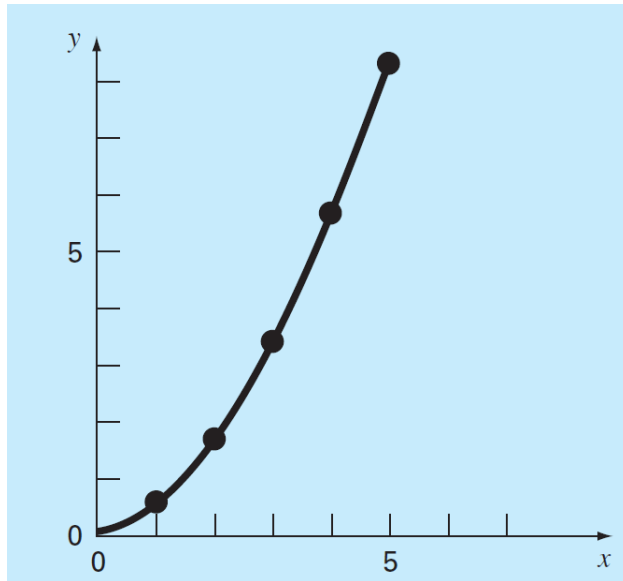
Linearized data

$$\begin{array}{l} (x'_1 = \log_{10}(x_1), y'_1 = \log_{10}(y_1)) \\ (x'_2 = \log_{10}(x_2), y'_2 = \log_{10}(y_1)) \\ \vdots \\ (x'_n = \log_{10}(x_n), y'_n = \log_{10}(y_n)) \end{array} \xrightarrow{\text{Linear regression}} y' = a_0 + a_1 x' \xrightarrow{\quad} \begin{array}{l} \beta_2 = a_1 \\ \alpha_2 = 10^{a_0} \end{array}$$

$$y = \alpha_2 x^{\beta_2} \xrightarrow{\quad} y' = \log_{10}(\alpha_2) + \beta_2 x'$$

Example

- Fit the power model of $y = \alpha_2 x^{\beta_2}$ to the data in the table below.



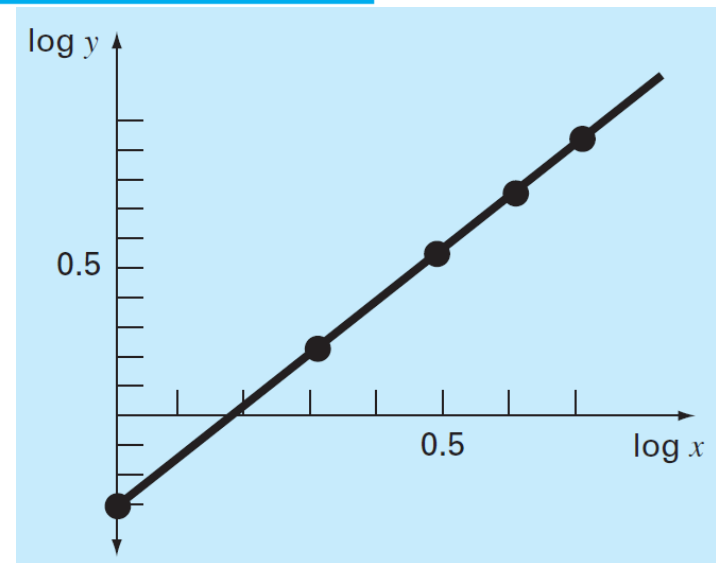
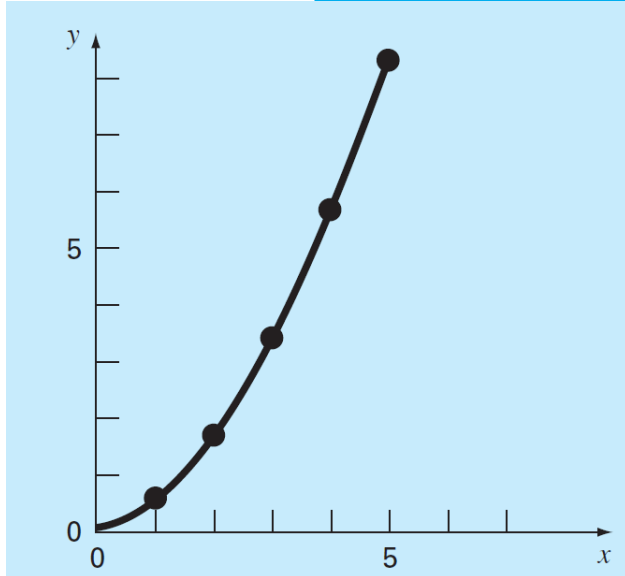
x	y
1	0.5
2	1.7
3	3.4
4	5.7
5	8.4

Example

- We first take the logarithm of x and y .

Linearized data

x	y	$\log x$	$\log y$
1	0.5	0	-0.301
2	1.7	0.301	0.226
3	3.4	0.477	0.534
4	5.7	0.602	0.753
5	8.4	0.699	0.922



Example

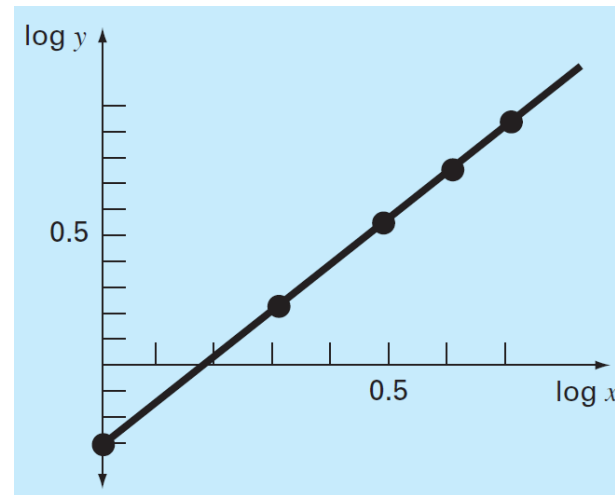
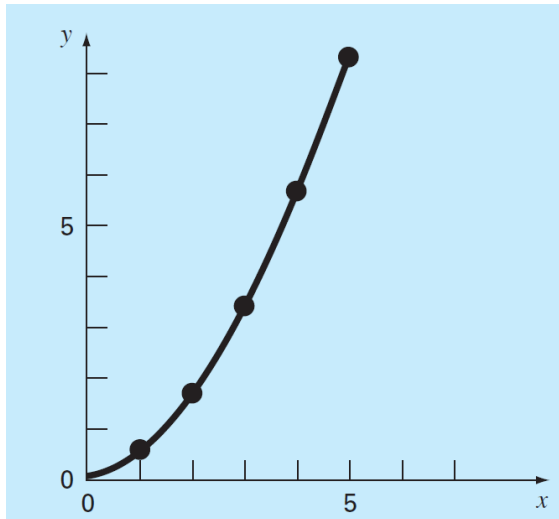
- The linear regression model of the linearized data is $y' = 1.75x' - 0.3$.
- $a_0 = -0.3, a_1 = 1.75$.

$$\beta_2 = 1.75$$



$$y = 0.5x^{1.75}$$

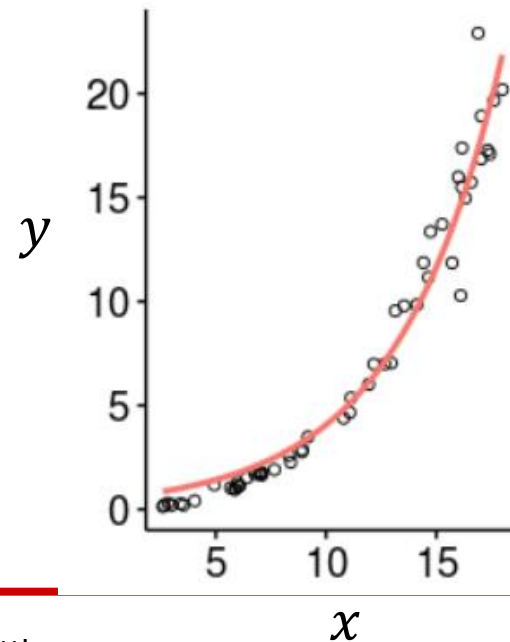
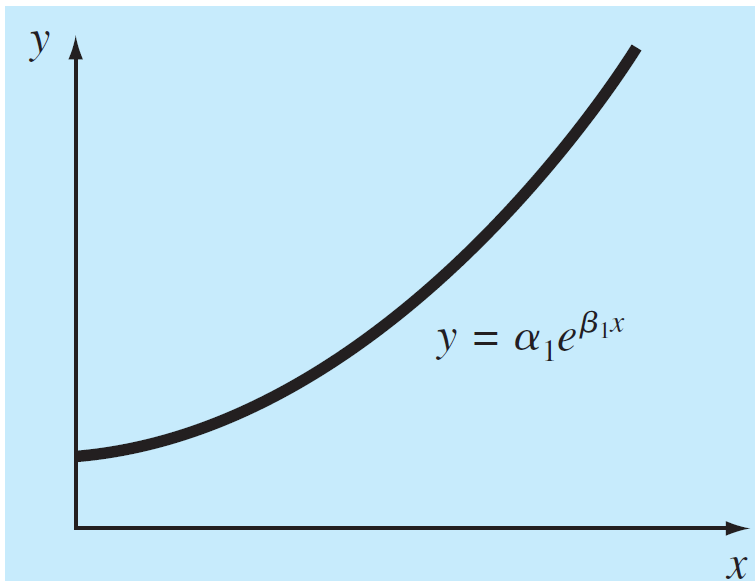
$$-0.3 = \log_{10}(\alpha_2) \Rightarrow \alpha_2 = 10^{-0.3} = 0.5$$



Exponential Model

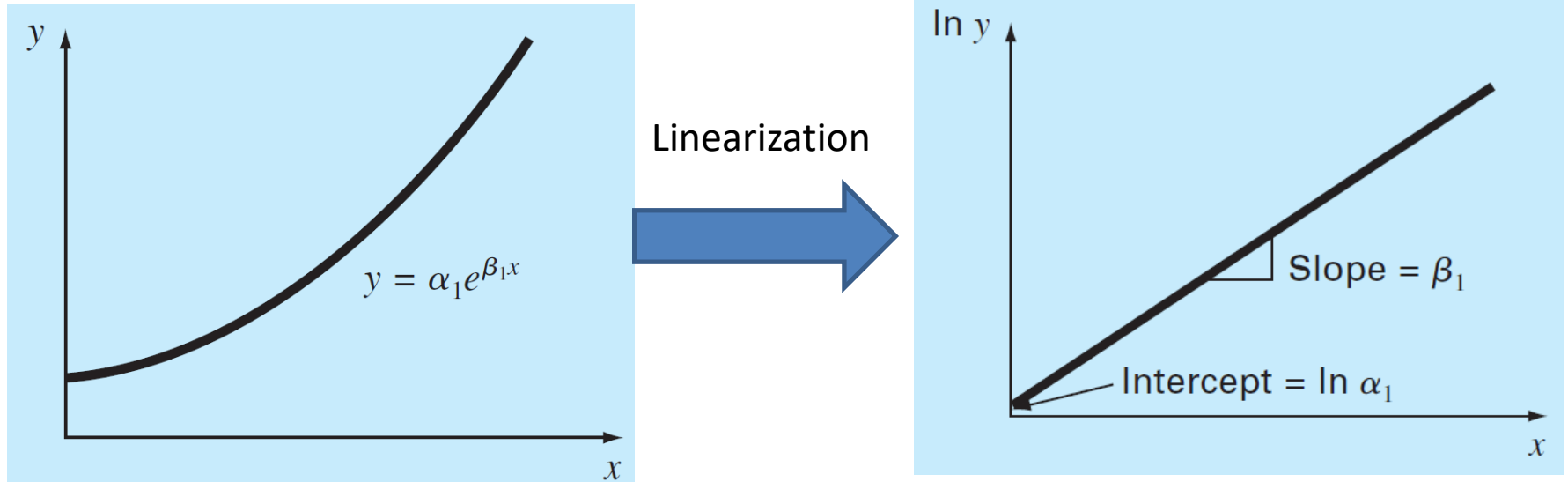
Exponential Models

- Sometimes the data follow an **exponential trend**.
- An **exponential model** is in the form of $y = \alpha_1 e^{\beta_1 x}$ where α_1 and β_1 are **constant parameters**.
- We need to find α_1 and β_1 .



Exponential Models

- Exponential data can be linearized by taking the natural logarithm of y .
- $y = \alpha_1 e^{\beta_1 x} \Rightarrow \ln(y) = \ln(\alpha_1) + \beta_1 x \Rightarrow y' = \ln(\alpha_1) + \beta_1 x$



Exponential Models

- **Exponential** data can be linearized by taking the natural logarithm of y in the **data points**.

Exponential
data

$$(x_1, y_1)$$

$$(x_2, y_2)$$

$$\vdots$$

$$(x_n, y_n)$$

Linearized data

$$(x_1, y'_1 = \ln(y_1))$$

$$(x_2, y'_2 = \ln(y_2))$$

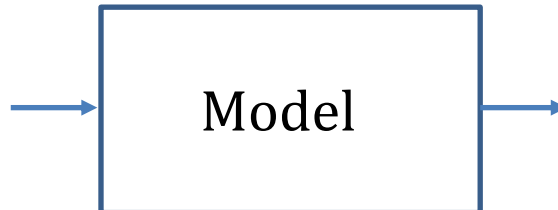
$$\vdots$$

$$(x_n, y'_n = \ln(y_n))$$

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

$$\boxed{\ln(y)} = \ln(\alpha_1) + \beta_1 x$$

y'



Exponential Models

- We then apply the **linear regression** to the **linearized data** to find a_0 and a_1 .

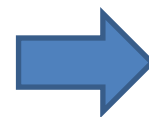
Exponential
data

(x_1, y_1)
 (x_2, y_2)
 $\vdots$
 (x_n, y_n)

Linearized data

$(x_1, y'_1 = \ln(y_1))$
 $(x_2, y'_2 = \ln(y_2))$
 $\vdots$
 $(x_n, y'_n = \ln(y_n))$

Linear
regression



$$y' = a_0 + a_1 x$$



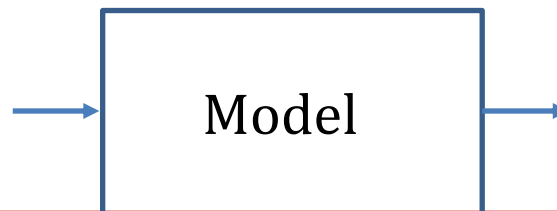
$$\begin{aligned}\beta_1 &= a_1 \\ \alpha_1 &= e^{a_0}\end{aligned}$$

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

$$\boxed{\ln(y)} = \ln(\alpha_1) + \beta_1 x$$

y'

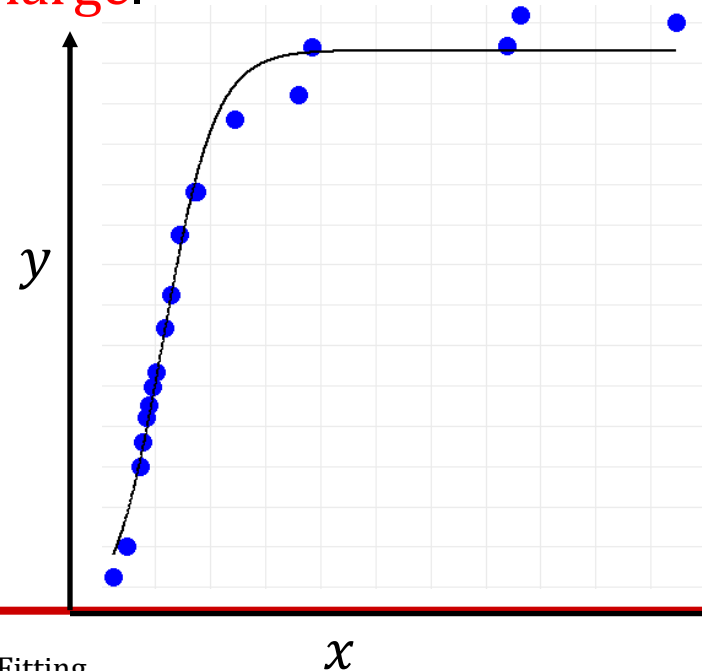
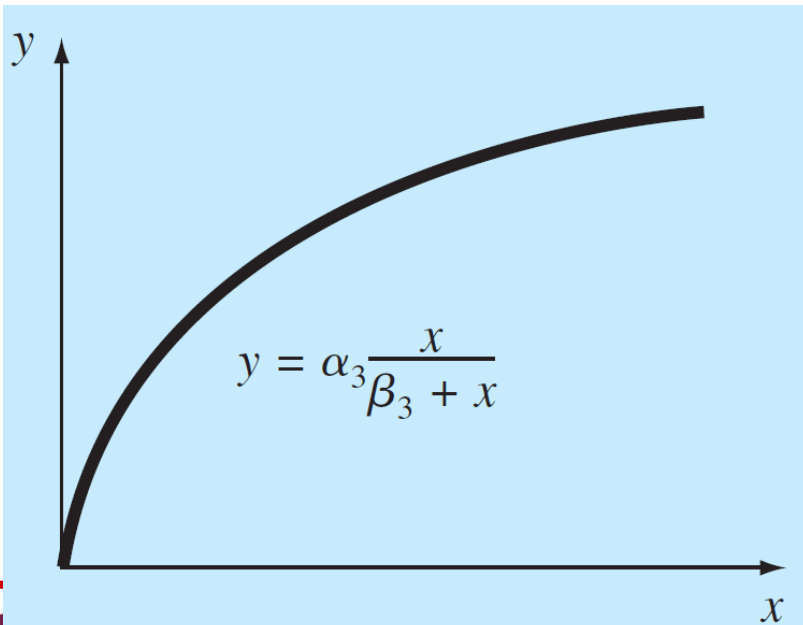
$$y' = \ln(\alpha_1) + \beta_1 x$$



Saturation-Growth-Rate Model

Saturation-Growth-Rate Models

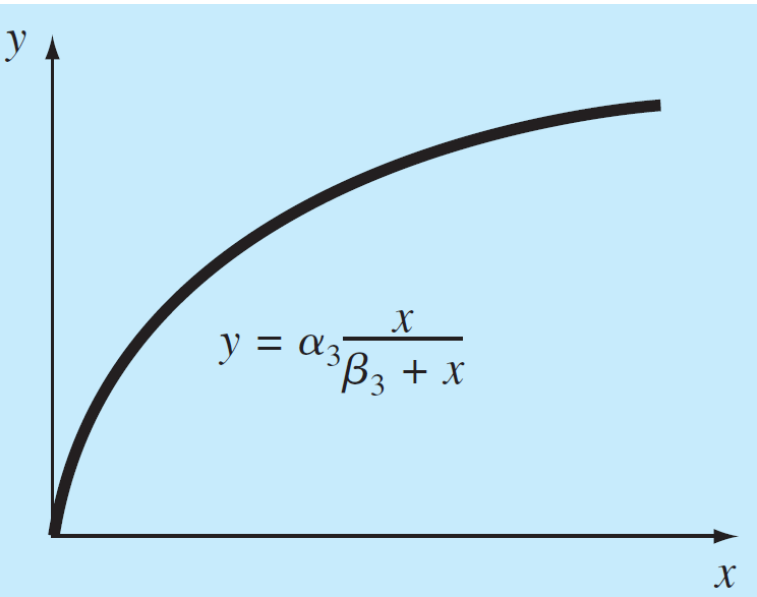
- Sometimes the data follows a **saturation-growth-rate** model.
- This model is in the form of $y = \alpha_3 \frac{x}{\beta_3 + x}$ where α_3 and β_3 are constants.
- We need to find **α_3 and β_3** .
- The value of y **saturates** as x becomes **large**.



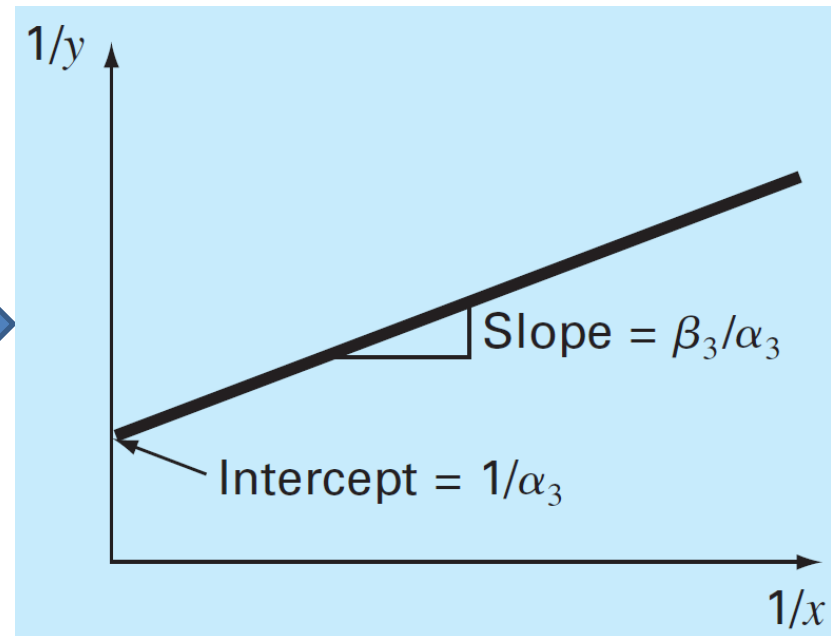
Saturation-Growth-Rate Models

- This model can be linearized by inverting x and y .

- $$y = \alpha_3 \frac{x}{\beta_3 + x} \quad \longrightarrow \quad \frac{1}{y} = \frac{\beta_3}{\alpha_3} \frac{1}{x} + \frac{1}{\alpha_3} \quad \longrightarrow \quad y' = \frac{\beta_3}{\alpha_3} x' + \frac{1}{\alpha_3}$$



Linearization



Saturation-Growth-Rate Models

- This model can be linearized by **inverting** x and y .

Data	Linearized data	
(x_1, y_1)	$(x'_1 = 1/x_1,$	$y'_1 = 1/y_1)$
(x_2, y_2)	$(x'_2 = 1/x_2,$	$y'_2 = 1/y_2)$
$\vdots$	$\vdots$	
(x_n, y_n)	$(x'_n = 1/x_n,$	$y'_n = 1/y_n)$

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

$$\boxed{\frac{1}{y}}_{y'} = \frac{\beta_3}{\alpha_3} \boxed{\frac{1}{x}}_{x'} + \frac{1}{\alpha_3}$$

$$y' = \frac{\beta_3}{\alpha_3} x' + \frac{1}{\alpha_3}$$

Saturation-Growth-Rate Models

- We then apply the linear regression to the linearized data.

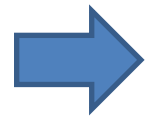
Linearized data

$$(x'_1 = 1/x_1, \quad y'_1 = 1/y_1)$$

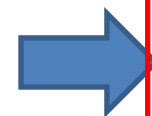
$$(x'_2 = 1/x_2, \quad y'_2 = 1/y_2)$$

$$(x'_n = 1/x_n, \quad y'_n = 1/y_n)$$

Linear
regression



$$y' = a_0 + a_1 x'$$

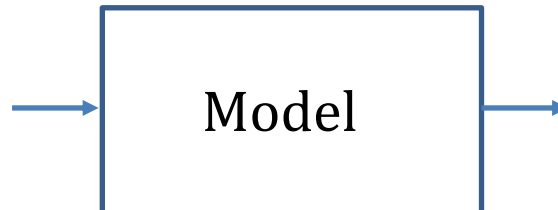


$$\alpha_3 = \frac{1}{a_0}$$

$$\beta_3 = a_1 \alpha_3$$

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

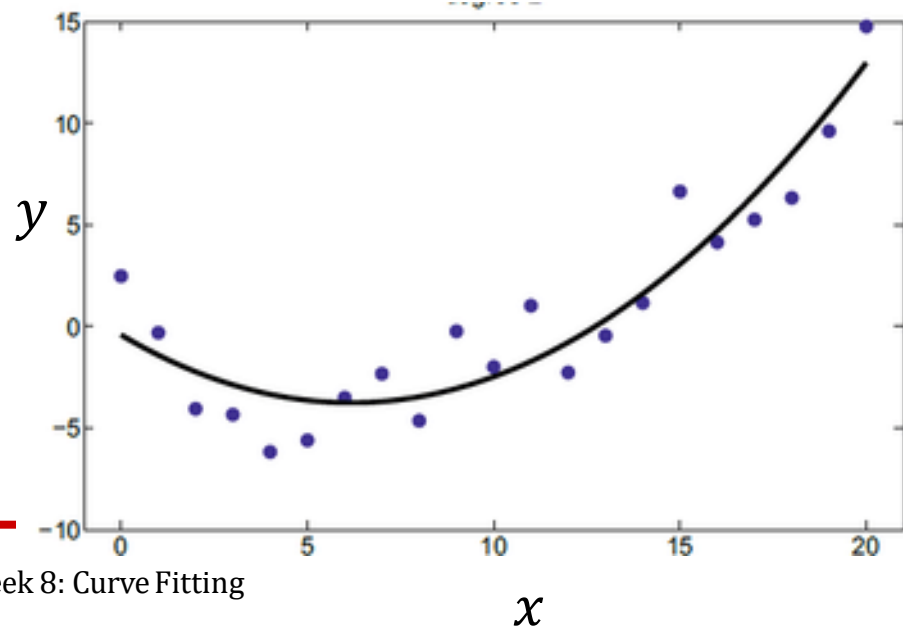
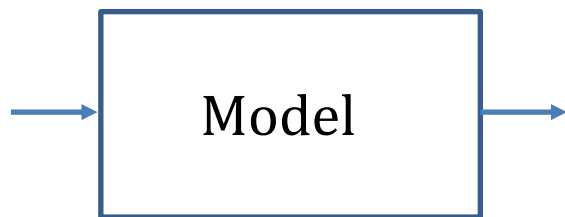
$$y' = \frac{\beta_3}{\alpha_3} x' + \frac{1}{\alpha_3}$$



Polynomial Least-Squares Regression

Polynomial Regression

- So far, we have learned how to **fit a linear model** to **data**. We also learned how **linearize** certain **non-linear** data.
- Certain data **cannot** be **linearized** using **transformations**.
- **Polynomial regression** is an alternative solution for building a model in this case. Here, we **fit a polynomial** model to **our data**.



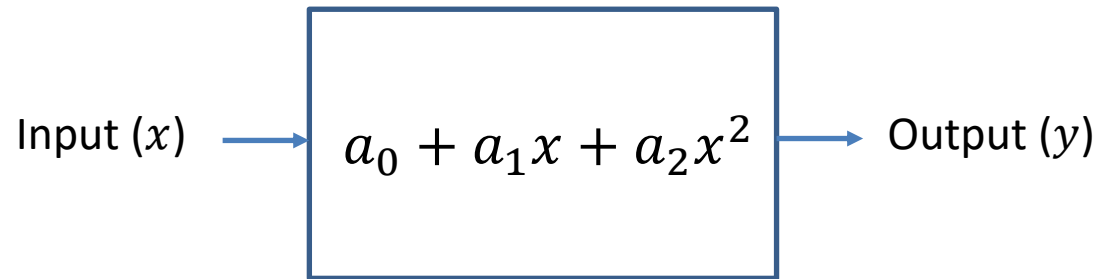
Polynomial Regression

- Suppose that we would like to fit a **quadratic polynomial** to our data.
- A **quadratic polynomial** model is of the form $y = a_0 + a_1x + a_2x^2$ where a_0, a_1, a_2 are constant parameters.
- We need to find a_0, a_1, a_2 .

x y

↓ ↓

(x_1, y_1)
 (x_2, y_2)
 $\vdots$
 (x_n, y_n)



Polynomial Regression

- The parameters a_0, a_1, a_2 are obtained by solving the following system of linear equations.

$$(n)a_0 + \left(\sum x_i\right)a_1 + \left(\sum x_i^2\right)a_2 = \sum y_i$$

$$\left(\sum x_i\right)a_0 + \left(\sum x_i^2\right)a_1 + \left(\sum x_i^3\right)a_2 = \sum x_i y_i$$

$$\left(\sum x_i^2\right)a_0 + \left(\sum x_i^3\right)a_1 + \left(\sum x_i^4\right)a_2 = \sum x_i^2 y_i$$

$$(x_1, y_1)$$

$$(x_2, y_2)$$

$$\vdots$$

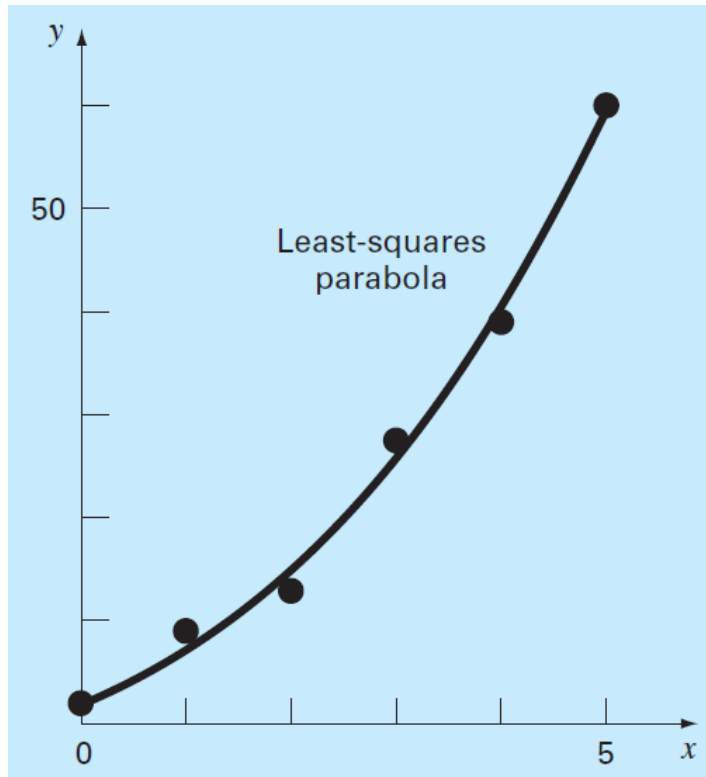
$$(x_n, y_n)$$

Polynomial Regression

- $\sum_{i=1}^n x_i^2 = x_1^2 + x_2^2 + \cdots + x_n^2$
 - $\sum_{i=1}^n x_i^3 = x_1^3 + x_2^3 + \cdots + x_n^3$
 - $\sum_{i=1}^n x_i^4 = x_1^4 + x_2^4 + \cdots + x_n^4$
 - $\sum_{i=1}^n x_i = x_1 + x_2 + \cdots + x_n$
 - $\sum_{i=1}^n y_i = y_1 + y_2 + \cdots + y_n$
 - $\sum_{i=1}^n x_i y_i = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$
 - $\sum_{i=1}^n x_i^2 y_i = x_1^2 y_1 + x_2^2 y_2 + \cdots + x_n^2 y_n$
- (x_1, y_1)
 (x_2, y_2)
 $\vdots$
 (x_n, y_n)

Example

- Fit a **quadratic polynomial** to the following data.



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

Example

- We first form the system of linear equations.

$$\sum x_i = 15 \quad \sum x_i^4 = 979$$

$$\sum y_i = 152.6 \quad \sum x_i y_i = 585.6$$

$$\sum x_i^2 = 55 \quad \sum x_i^2 y_i = 2488.8$$

$$\sum x_i^3 = 225$$

$$(n)a_0 + \left(\sum x_i\right)a_1 + \left(\sum x_i^2\right)a_2 = \sum y_i$$

$$\left(\sum x_i\right)a_0 + \left(\sum x_i^2\right)a_1 + \left(\sum x_i^3\right)a_2 = \sum x_i y_i$$

$$\left(\sum x_i^2\right)a_0 + \left(\sum x_i^3\right)a_1 + \left(\sum x_i^4\right)a_2 = \sum x_i^2 y_i$$

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

Example

The solution of the system of linear equations is $a_0 = 2.47857$, $a_1 = 2.35929$, and $a_3 = 1.86071$.

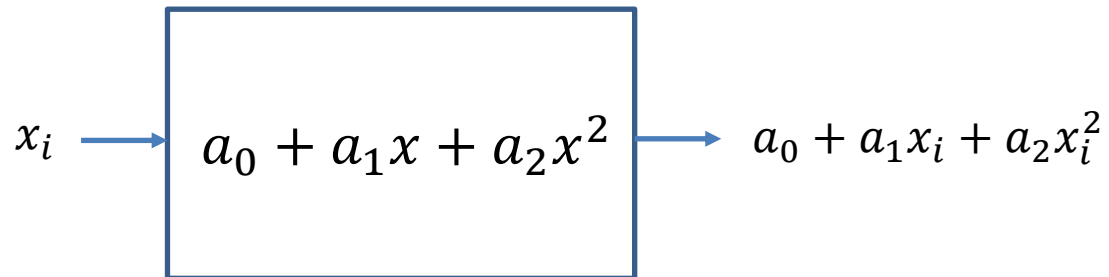
- The optimal model is

$$y = 2.47857 + 2.35929x + 1.86071x^2$$

How does it work?

- Consider the polynomial model $y = a_0 + a_1x + a_2x^2$.
- When the input to the model is x_i , its output is $a_0 + a_1x_i + a_2x_i^2$.
- The error for x_i is $y_i - a_0 - a_1x_i - a_2x_i^2$
- The **sum of squares of error** is $\sum_{i=1}^n (y_i - a_0 - a_1x_i - a_2x_i^2)^2$

(x_1, y_1)
 (x_2, y_2)
 $\vdots$
 (x_n, y_n)



How does it work?

- We will adjust a_0, a_1, a_2 such that the error is **minimized**.

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

- To find the optimal parameters of the model, we take the derivative of the total error with respect to a_0, a_1, a_2 .

$$S_r = \sum_{i=1}^n (y_i - a_0 - a_1 x_i - a_2 x_i^2)^2 \quad \frac{\partial S_r}{\partial a_1} = -2 \sum x_i (y_i - a_0 - a_1 x_i - a_2 x_i^2)$$

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

How does it work?

- We then set these equations to zero.

$$\frac{\partial S_r}{\partial a_0} = 0 \quad \frac{\partial S_r}{\partial a_1} = 0 \quad \frac{\partial S_r}{\partial a_2} = 0$$

- We obtain a system of linear equations in a_0, a_1, a_2 .

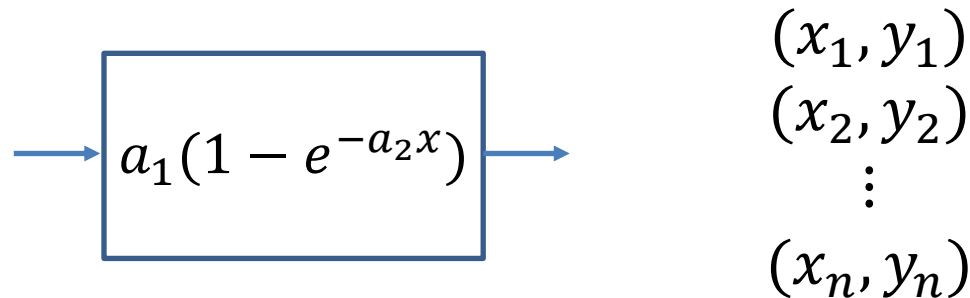
$$\begin{aligned}(n)a_0 + \left(\sum x_i\right)a_1 + \left(\sum x_i^2\right)a_2 &= \sum y_i \\ \left(\sum x_i\right)a_0 + \left(\sum x_i^2\right)a_1 + \left(\sum x_i^3\right)a_2 &= \sum x_i y_i \\ \left(\sum x_i^2\right)a_0 + \left(\sum x_i^3\right)a_1 + \left(\sum x_i^4\right)a_2 &= \sum x_i^2 y_i\end{aligned}$$

- The **solution** of this system of linear equation is the optimal parameters a_0, a_1, a_2 .

Non-linear Least-Squares Regression

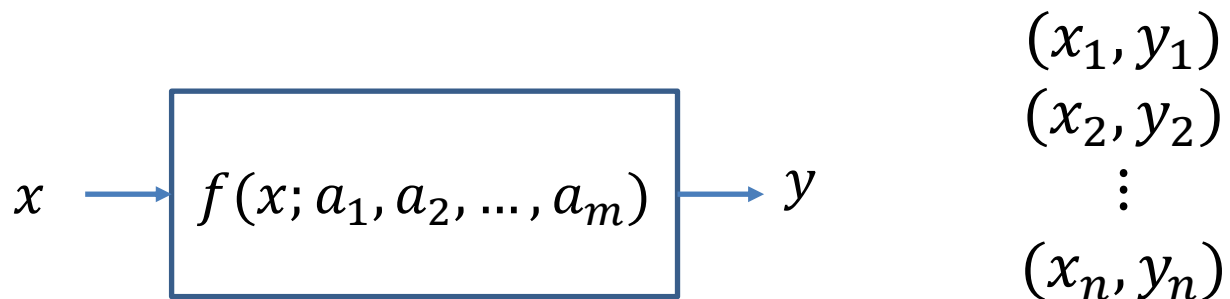
Non-linear Least-Squares Regression

- Sometimes we may need to fit a **non-linear model** to our data.
- For example, we may need to fit $a_1(1 - e^{-a_2x})$ to our data where a_1 and a_2 are constant parameters.
- The model is **not** a **polynomial**.
- We **cannot linearize** it using **transformations**.



Non-linear Least-Squares Regression

- Suppose that we need to fit the nonlinear model $f(x; a_1, a_2, \dots, a_m)$ to n data points.
- The model has m parameters $a_1, a_2, \dots, a_m$.
- We will discuss the Gauss-Newton algorithm for finding the optimal values of $a_1, a_2, \dots, a_m$.



Matrix Transpose

- Suppose that you have a matrix J .
- The **transpose of J** is **another matrix**. We show it using J^T .
- The **first row of J** is the **first column of J^T** . The **second row of J** is the **second column of J^T** , etc.

$$J = \begin{bmatrix} 24 & 24 & -6 & -6 & 36 \\ 0 & 30 & 0 & 0 & -30 \\ 30 & -30 & 0 & 30 & -30 \end{bmatrix}$$

$$J^T = \begin{bmatrix} 24 & 0 & 30 \\ 24 & 30 & -30 \\ -6 & 0 & 0 \\ -6 & 0 & 30 \\ 36 & -30 & -30 \end{bmatrix}$$

Gauss-Newton Method

- Gauss-Newton method is an **iterative algorithm**.
- k is the **iteration counter** of the algorithm.
- Use a vector to represent the parameter, $A = [a_1, a_2, \dots, a_m]$.

$$f(x; A) = f(x; a_1, a_2, \dots, a_m)$$

- The values of parameters are **updated** in each **iteration**.
- $A_k = [a_1^k, a_2^k, \dots, a_m^k]$ stores the values of parameters at **iteration k** .
- $A_0 = [a_1^0, a_2^0, \dots, a_m^0]$ is the **initial values** of the **parameters**.
- a_1^0 is the initial value of a_1 . a_i^0 is the initial value of a_i .

Gauss-Newton Method

1. $k = 1$ (iteration counter)
2. $A_0 = [a_1^0, a_2^0, \dots, a_m^0]$ (initial values of parameters are input to the algorithm)
3. Compute the vector ΔY using A_{k-1} .

$$f(x; A) = f(x; a_1, a_2, \dots, a_m)$$

$$\Delta Y = \begin{bmatrix} y_1 - f(x_1; A_{k-1}) \\ y_2 - f(x_2; A_{k-1}) \\ \vdots \\ y_n - f(x_n; A_{k-1}) \end{bmatrix} \quad \begin{matrix} (x_1, y_1) \\ (x_2, y_2) \\ \vdots \\ (x_n, y_n) \end{matrix}$$

Gauss-Newton Method

4. Compute the matrix J using A_{k-1} .

$$f(x; A) = f(x; a_1, a_2, \dots, a_m)$$

$$J = \begin{bmatrix} \frac{\partial}{\partial a_1} f(x_1; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_1; A_{k-1}) & \dots & \frac{\partial}{\partial a_m} f(x_1; A_{k-1}) \\ \frac{\partial}{\partial a_1} f(x_2; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_2; A_{k-1}) & \dots & \frac{\partial}{\partial a_m} f(x_2; A_{k-1}) \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial}{\partial a_1} f(x_n; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_n; A_{k-1}) & \dots & \frac{\partial}{\partial a_m} f(x_n; A_{k-1}) \end{bmatrix}$$

- In this step, we need to partial derivatives

$$\frac{\partial}{\partial a_1} f(x; A) = \frac{\partial}{\partial a_1} f(x; a_1, a_2, \dots, a_m)$$

$$\vdots$$

$$\frac{\partial}{\partial a_m} f(x; A) = \frac{\partial}{\partial a_m} f(x; a_1, a_2, \dots, a_m)$$

Gauss-Newton Method

5. Update the parameter according to $A_k = A_{k-1} + (J^T J)^{-1} J^T \Delta Y$.
6. Compute d_k according to $d_k = |A_k - A_{k-1}|$.
7. If all the elements of d_k are less than ϵ , stop.
8. Otherwise, $k \leftarrow k+1$ and go to 3.
9. The last generated A_k is the optimal value of the parameters.

$$J = \begin{bmatrix} \frac{\partial}{\partial a_1} f(x_1; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_1; A_{k-1}) & \cdots & \frac{\partial}{\partial a_m} f(x_1; A_{k-1}) \\ \frac{\partial}{\partial a_1} f(x_2; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_2; A_{k-1}) & \cdots & \frac{\partial}{\partial a_m} f(x_2; A_{k-1}) \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial}{\partial a_1} f(x_n; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_n; A_{k-1}) & \cdots & \frac{\partial}{\partial a_m} f(x_n; A_{k-1}) \end{bmatrix} \quad \Delta Y = \begin{bmatrix} y_1 - f(x_1; A_{k-1}) \\ y_2 - f(x_2; A_{k-1}) \\ \vdots \\ y_n - f(x_n; A_{k-1}) \end{bmatrix}$$

Gauss-Newton Method

1. $k = 1$ (iteration counter)
2. $A^0 = [a_1^0, a_2^0, \dots, a_m^0]$ (initial values of parameters)
3. Compute the vector ΔY using A_{k-1} .
4. Compute the matrix J using A_{k-1} .
5. Update the parameter according to $A_k = A_{k-1} + (J^T J)^{-1} J^T \Delta Y$.
6. Compute d_k according to $d_k = |A_k - A_{k-1}|$.
7. If all the elements of d_k are less than ϵ , stop.
8. Otherwise, $k \leftarrow k+1$ and go to 3.

Gauss–Newton Method

To compute $(J^T J)^{-1} J^T \Delta Y$:

1. Compute the matrix J .
2. Compute the matrix J^T using J .
3. To compute the matrix $J^T J$, multiply J^T by J .
4. Compute the inverse of the matrix $J^T J$.
5. To compute $(J^T J)^{-1} J^T$, multiply the inverse of $J^T J$, i.e., $(J^T J)^{-1}$, by J^T .
6. Multiply ΔY by the matrix $(J^T J)^{-1} J^T$.

$$\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix} = (X^T X)^{-1} X^T Y \quad X = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{nk} \end{bmatrix} \quad X^T = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ x_{11} & x_{21} & x_{31} & \dots & x_{n1} \\ x_{12} & x_{22} & x_{32} & \dots & x_{n2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{1k} & x_{2k} & x_{3k} & \dots & x_{nk} \end{bmatrix}$$

Optimal parameters

Example

- Fit the model $a_1(1 - e^{-a_2x})$ to the data in the table below.
- Use $a_1^0 = 1$ and $a_2^0 = 1$ as the initial parameters. $A_0 = [1,1]$.
- Assume $\epsilon = 0.0001$.

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

Example

- $x_1 = 0.25, x_2 = 0.75, x_3 = 1.25, x_4 = 1.75, x_5 = 2.25$.

- $A = [a_1, a_2]$

$$f(x; A) = a_1(1 - e^{-a_2 x})$$

- $A_k = [a_1^k, a_2^k]$.

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

Example

- We first form the vector ΔY using $A_{k-1} = [a_1^{k-1}, a_2^{k-1}]$.

$$\Delta Y = \begin{bmatrix} y_1 - f(x_1; A_{k-1}) \\ y_2 - f(x_2; A_{k-1}) \\ \vdots \\ y_n - f(x_n; A_{k-1}) \end{bmatrix}$$

$$f(x; A) = a_1(1 - e^{-a_2 x})$$

Example

- We then form the matrix J using $A_{k-1} = [a_1^{k-1}, a_2^{k-1}]$.

$$f(x; A) = a_1(1 - e^{-a_2 x})$$

$$J = \begin{bmatrix} \frac{\partial}{\partial a_1} f(x_1; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_1; A_{k-1}) \\ \frac{\partial}{\partial a_1} f(x_2; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_2; A_{k-1}) \\ \vdots & \vdots \\ \frac{\partial}{\partial a_1} f(x_5; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_5; A_{k-1}) \end{bmatrix}$$

$$\frac{\partial f}{\partial a_1} = 1 - e^{(-a_2 x)}$$

$$\frac{\partial f}{\partial a_2} = a_1 x e^{(-a_2 x)}$$

Example

- After 5 iterations, all the elements of the difference vector are less than 0.0001.

Compute the vector ΔY using A_{k-1} .

- So, we stop.

Compute the matrix J using A_{k-1} .

Update the parameter according to $A_k = A_{k-1} + (J^T J)^{-1} J^T \Delta Y$.

Compute d_k according to $d_k = |A_k - A_{k-1}|$.

If all the elements of d_k are less than ϵ , stop.

	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
a_1^k	1	0.7285	0.7910	0.7919	0.7919	0.7919
a_2^k	1	1.5019	1.6777	1.6753	1.6751	1.6751
d_1^k		0.2715	0.0625	8.0867e-04	1.5129e-05	7.4659e-07
d_2^k		0.5019	0.1758	0.0024	1.2254e-04	6.3282e-06

Example

- In the first iteration, we have $A_0 = [1,1]$ ($k = 1$).

$$f(x; A) = a_1(1 - e^{-a_2 x})$$

$$\Delta Y = \begin{bmatrix} 0.28 - (1 - e^{-0.25}) \\ 0.57 - (1 - e^{-0.75}) \\ 0.68 - (1 - e^{-1.25}) \\ 0.74 - (1 - e^{-1.75}) \\ 0.79 - (1 - e^{-2.25}) \end{bmatrix} = \begin{bmatrix} 0.0588 \\ 0.0424 \\ -0.0335 \\ -0.0862 \\ -0.1046 \end{bmatrix}$$

$$\Delta Y = \begin{bmatrix} y_1 - f(x_1; A_{k-1}) \\ y_2 - f(x_2; A_{k-1}) \\ \vdots \\ y_n - f(x_n; A_{k-1}) \end{bmatrix}$$

Example

- In the first iteration, we have $A_0 = [1,1]$ ($k = 1$).

$$f(x; A) = a_1(1 - e^{-a_2 x})$$

$$J = \begin{bmatrix} 1 - e^{-0.25} & 0.25e^{-0.25} \\ 1 - e^{-0.75} & 0.75e^{-0.75} \\ 1 - e^{-1.25} & 1.25e^{-1.25} \\ 1 - e^{-1.75} & 1.75e^{-1.75} \\ 1 - e^{-2.25} & 2.25e^{-2.25} \end{bmatrix} = \begin{bmatrix} 0.2212 & 0.1947 \\ 0.5276 & 0.3543 \\ 0.7135 & 0.3581 \\ 0.8262 & 0.3041 \\ 0.8946 & 0.2371 \end{bmatrix}$$

$$\frac{\partial f}{\partial a_1} = 1 - e^{(-a_2 x)}$$

$$\frac{\partial f}{\partial a_2} = a_1 x e^{(-a_2 x)}$$

$$J = \begin{bmatrix} \frac{\partial}{\partial a_1} f(x_1; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_1; A_{k-1}) \\ \frac{\partial}{\partial a_1} f(x_2; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_2; A_{k-1}) \\ \vdots & \vdots \\ \frac{\partial}{\partial a_1} f(x_5; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_5; A_{k-1}) \end{bmatrix}$$

Example

- In the first iteration, we have $A_0 = [1,1]$ ($k = 1$).

$$J^T = \begin{bmatrix} 0.2212 & 0.5276 & 0.7135 & 0.8262 & 0.8946 \\ 0.1947 & 0.3543 & 0.3581 & 0.3041 & 0.2371 \end{bmatrix}$$

$$J^T J = \begin{bmatrix} 2.3194 & 0.9489 \\ 0.9489 & 0.4404 \end{bmatrix}$$

$$(J^T J)^{-1} = \begin{bmatrix} 3.6407 & -7.8447 \\ -7.8447 & 19.1738 \end{bmatrix}$$

$$(J^T J)^{-1} J^T = \begin{bmatrix} -0.7220 & -0.8582 & -0.2118 & 0.6224 & 1.3966 \\ 1.9979 & 2.6537 & 1.2696 & -0.6506 & -2.4708 \end{bmatrix}$$

Example

- In the second iteration, we have $A_1 = [0.7285, 1.5019]$ ($k = 2$).

$$f(x; A) = a_1(1 - e^{-a_2 x})$$

$$\Delta Y = \begin{bmatrix} 0.28 - 0.7285(1 - e^{-1.5019 \times 0.25}) \\ 0.57 - 0.7285(1 - e^{-1.5019 \times 0.75}) \\ 0.68 - 0.7285(1 - e^{-1.5019 \times 1.25}) \\ 0.74 - 0.7285(1 - e^{-1.5019 \times 1.75}) \\ 0.79 - 0.7285(1 - e^{-1.5019 \times 2.25}) \end{bmatrix} = \begin{bmatrix} 0.0519 \\ 0.0777 \\ 0.0629 \\ 0.0641 \\ 0.0863 \end{bmatrix}$$

$$\Delta Y = \begin{bmatrix} y_1 - f(x_1; A_{k-1}) \\ y_2 - f(x_2; A_{k-1}) \\ \vdots \\ y_n - f(x_n; A_{k-1}) \end{bmatrix}$$

Example

- In the first iteration, we have $A_1 = [0.7285, 1.5019]$ ($k = 2$).

$$f(x; A) = a_1(1 - e^{-a_2 x})$$

$$J = \begin{bmatrix} 1 - e^{-1.5019 \times 0.25} & 0.7285 \times 0.25 e^{-1.5019 \times 0.25} \\ 1 - e^{-1.5019 \times 0.75} & 0.7285 \times 0.75 e^{-1.5019 \times 0.75} \\ 1 - e^{-1.5019 \times 1.25} & 0.7285 \times 1.25 e^{-1.5019 \times 1.25} \\ 1 - e^{-1.5019 \times 1.75} & 0.7285 \times 1.75 e^{-1.5019 \times 1.75} \\ 1 - e^{-1.5019 \times 2.25} & 0.7285 \times 2.25 e^{-1.5019 \times 2.25} \end{bmatrix} = \begin{bmatrix} 0.3130 & 0.1251 \\ 0.6758 & 0.1771 \\ 0.8470 & 0.1393 \\ 0.9278 & 0.0920 \\ 0.9659 & 0.0558 \end{bmatrix}$$

$$J = \begin{bmatrix} \frac{\partial}{\partial a_1} f(x_1; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_1; A_{k-1}) \\ \frac{\partial}{\partial a_1} f(x_2; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_2; A_{k-1}) \\ \vdots & \vdots \\ \frac{\partial}{\partial a_1} f(x_5; A_{k-1}) & \frac{\partial}{\partial a_2} f(x_5; A_{k-1}) \end{bmatrix}$$

$$\frac{\partial f}{\partial a_1} = 1 - e^{(-a_2 x)}$$

$$\frac{\partial f}{\partial a_2} = a_1 x e^{(-a_2 x)}$$

Example

- In the second iteration, we have $A_1 = [0.7285, 1.5019]$ ($k = 2$).

$$J^T = \begin{bmatrix} 0.3130 & 0.6758 & 0.8470 & 0.9278 & 0.9659 \\ 0.1251 & 0.1771 & 0.1393 & 0.0920 & 0.0558 \end{bmatrix}$$

$$J^T J = \begin{bmatrix} 3.0660 & 0.4162 \\ 0.4162 & 0.0780 \end{bmatrix}$$

$$(J^T J)^{-1} = \begin{bmatrix} 1.1823 & -6.3065 \\ -6.3065 & 46.4553 \end{bmatrix}$$

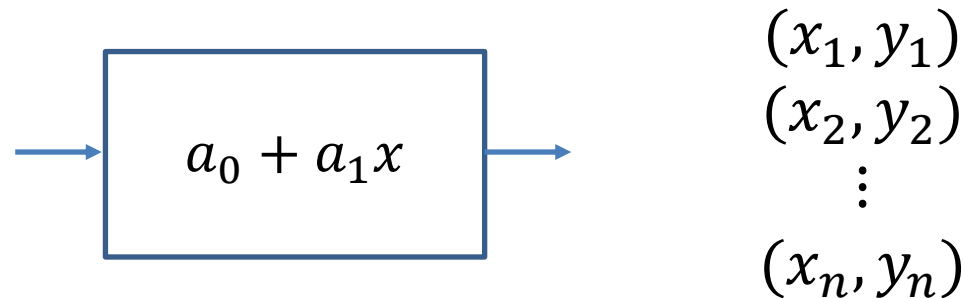
$$(J^T J)^{-1} J^T = \begin{bmatrix} -0.4189 & -0.3181 & 0.1228 & 0.5165 & 0.7898 \\ 3.8381 & 3.9667 & 1.1303 & -1.5753 & -3.4972 \end{bmatrix}$$

How does it work?

- When we use the **linear** model $a_0 + a_1x$, the **error** becomes a **quadratic** function of a_0, a_1 .

$$S_r = \sum_{i=1}^n (y_i - a_0 - a_1x_i)^2$$

- We can **easily** find the **optimal values** of a_0, a_1 when the **error is quadratic**.



How does it work?

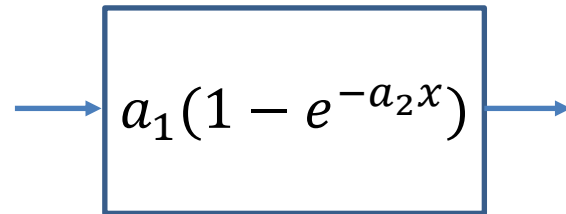
- When we use the model $a_1(1 - e^{-a_2x})$, the error is not a quadratic function of a_1, a_2 .

$$S_r = \sum_{i=1}^n (y_i - a_1 + a_1 e^{-a_2 x_i})^2$$

(x_1, y_1)
 (x_2, y_2)
 $\vdots$
 (x_n, y_n)

- Therefore, we cannot find a closed-form expression for the optimal values of a_1, a_2 .

- How can we solve this issue?



How does it work?

- The basic idea is to **approximate** the **non-linear** model with a **linear model** using Taylor's approximation.
1. Start with an **initial value** for the **parameters**, e.g., a_1, a_2 .
 2. **Approximate** the **non-linear model** around the **values of parameters** (e.g., a_1, a_2) with a **linear model**.
 3. Solve the **linear least-squares problem** for the **approximate model**.
 4. Update the **values of parameters** based on the **solution of the linear model**.
 5. **Repeat** steps 2-4 until the values of parameters **do not change**.

END

See you next week 😊!