IVC on Astrostatistics and Machine Learning (2-13 August 2021)

Introduction to Regression

Norhasliza Yusof
Department of Physics
University of Malaya

https://github.com/lizayusof/IVC_Astrostat_ML

The notes is adapted from course content SIF2007 (Norhasliza Yusof) and SQA7006 (Hasan Abu Kassim)

Regression

Linear Regression Algorithm

- Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data.
- The most common method for fitting a regression line is the method of least-squares.
- Least-squares method calculates the best-fitting line for the observed data by minimizing the sum of the squares of the vertical deviations from each data point to the line (if a point lies on the fitted line exactly, then its vertical deviation is 0).
- Because the deviations are first squared, then summed, there are no cancellations between positive and negative values.

Least Squares

- Also called **Linear Regression**.
- Physics experiments measuring several values of two different physical variables to investigate mathematical relationship.
- Most important is expected relation is linear.
- For example: a body falling under constant g , the expected relation is

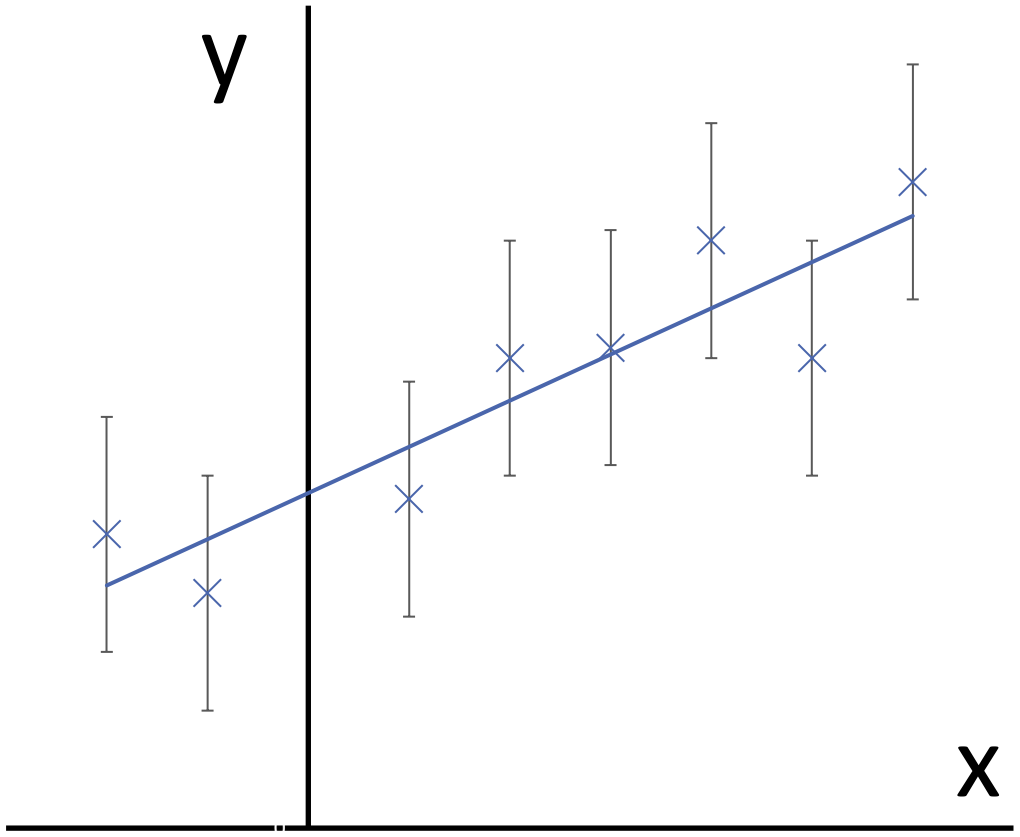
$$v = u + gt.$$

- We consider linear relation between two physical variables x and y:

$$y = mx + b \ \cdots (1)$$

where m and b are gradient and y-axis intercept, respectively.

Least Squares



- If measurements no uncertainties or errors then points are on line.
- In practice, there are uncertainties or errors from experiments.
- Points are some distance from line comparable with error bars.

Least Squares

- ***Question***: if y and x are linearly related how to find the straight line $y = mx + b$ that **best fits** the experimental data?
- ***Answer***: find the **best estimate** for m and b.
- This method is called **linear regression** or **least-squares** fit for a straight line.

Least Squares

- Least-squares estimates for m and b for n number of measurements:

$$m = \frac{n\sum_n xy - \sum_n x \sum_n y}{n\sum_n x^2 - (\sum_n x)^2} \cdots (2)$$

$$\mathrm{b} = \frac{n\sum_n x^2 \sum_n y - \sum_n x \sum_n xy}{n\sum_n x^2 - (\sum_n x)^2} \cdots (3)$$

- Note: the sum of all values of a variable, for example $\sum_n x = \sum_{i=1}^{i=n} x_i$.

Least Squares

- Uncertainties in measurements y_i:

$$\Delta y = \sqrt{\frac{\sum_{i=1}^{i=n}(y_i - mx_i - b)^2}{n-2}} \cdots (4)$$

- Note: the values of m and c are from least-squares formulae.

Least Squares

- Uncertainties or errors for m and b:

$$\Delta m = \Delta y \sqrt{\frac{n}{n\sum_n x^2 - (\sum_n x)^2}} \cdots (5)$$

$$\Delta b = \Delta y \sqrt{\frac{\sum_n x^2}{n\sum_n x^2 - (\sum_n x)^2}} \cdots (6)$$

- Note: the sum of all values of a variable, for example $\sum_n x = \sum_{i=1}^{i=n} x_i$.

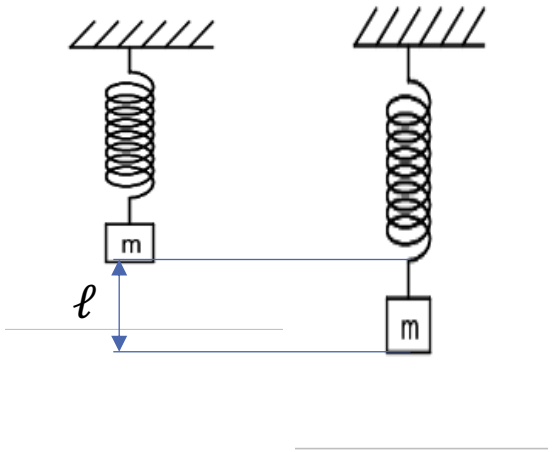
Least Squares

- ***Question***: how do we know the best straight line is accurate or how good is the fitting?
- ***Answer***: calculate the R^2 (R-squared) to the straight line.
- The range of R^2:

$$0 \leq R^2 \leq 1.$$

- The best fit (100% fit): $R^2 = 1$.
- The worst fit (0% fit): $R^2 = 0$.

Least Squares



- Example: Hooke's law. A student measures the elongation, ℓ of a spring when the mass, m is varied.

i	m (kg)	l (cm)
1	2	42.0
2	4	48.4
3	6	51.3
4	8	56.3
5	10	58.6

Least Squares

i	m (kg)	l (cm)
1	2	42.0
2	4	48.4
3	6	51.3
4	8	56.3
5	10	58.6

- To analyse the data:
  - Plot a graph
  - Calculations:
    1. Use a table to calculate least squares formula by hand with the help of a hand calculator. OR
    2. Use a python program to calculate least squares formula.

Least Squares

- Python implementation:

scipy.stats.linregress(*x*, *y*)

- scipy.stats.linregress(*x*, *y*) calculates m and b in Equation (2) and Equation (3) where x and y are the x-axis and y-axis data.
- Reference:

https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.linregress.html

Least Squares

- Python example programs: you are given two versions of the least squares programs which are

1. leastsquares1.py → the data is in an external file (CSV file).

2. leastsquares2.py → the data is typed in the program itself.

Polynomial Regression

- A polynomial of n^{th} order:

$$y = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \cdots + a_{n-2} x^2 + a_{n-1} x + a_n$$

- A polynomial of the second order (quadratic), n=2:

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

- A polynomial of the first order (linear), n=1:

$$y = a_0 x + a_1.$$

- Can we do a polynomial regression on the data to fit a polynomial equation to it? Yes!
- Can we do a linear regression on the data to fit a polynomial equation to it? Yes, but a simple linear regression would not fit very well!

Polynomial Regression

- Python implementation: use numpy library
- Least squares polynomial fit:

numpy.polyfit(*x, y, deg, rcond=None, full=False, w=None, cov=False*)

- Fit a polynomial p(x) = p[0] * x**deg + ... + p[deg] of degree *deg* to points *(x, y)*. Returns a vector of coefficients *p* that minimises the squared error in the order *deg*, *deg-1*, … *0*.
- Reference:

https://numpy.org/doc/stable/reference/generated/numpy.polyfit.html

- Note: deg = n and coefficients p[0], …, p[deg] = $a_0, \cdots, a_n$ in our notes.

Polynomial Regression

- Use **poly1d** in Numpy to get the polynomial from the coefficients generated by **polyfit**.
- numpy.poly1d(*c_or_r*, *r=False*, *variable=None*)
  - Reference:

https://numpy.org/doc/stable/reference/generated/numpy.poly1d.html#numpy.poly1d

Introduction to Regression

THE END

THANK YOU.