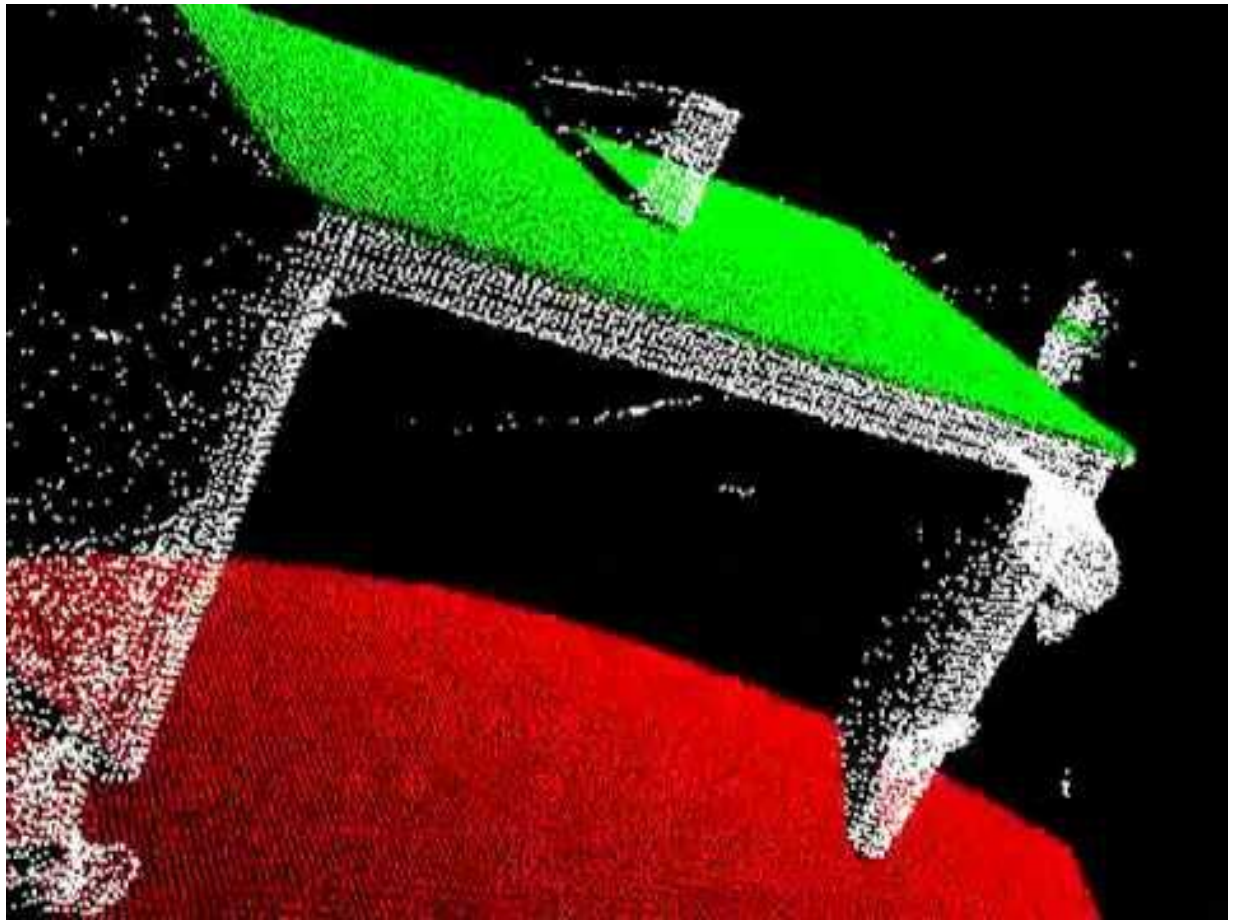
Curve fitting



Yoni Chechik

www.AlisMath.com

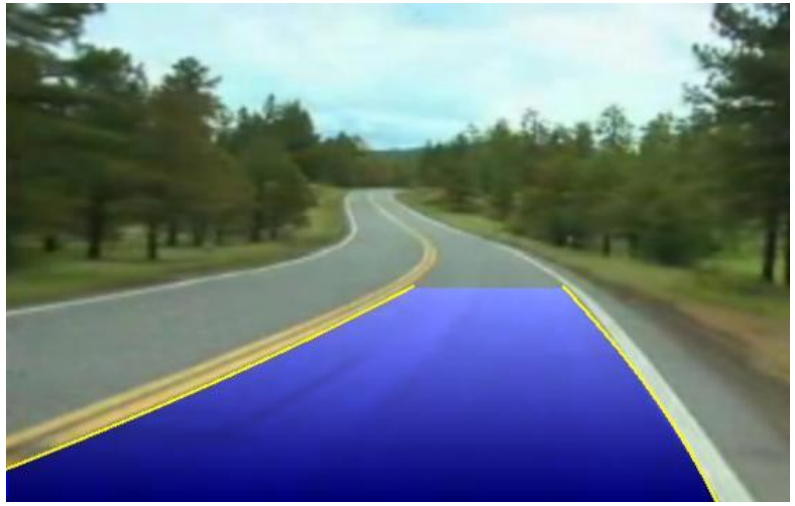
References

- http://szeliski.org/Book/
- http://www.cs.cornell.edu/courses/cs5670/2019sp/lectures/lectures.html
- http://www.cs.cmu.edu/~16385/

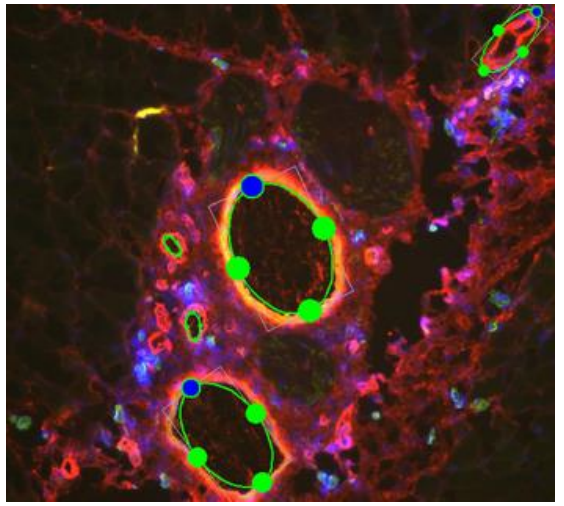
TOC

- Least squares
- Total least squares
- RANSAC

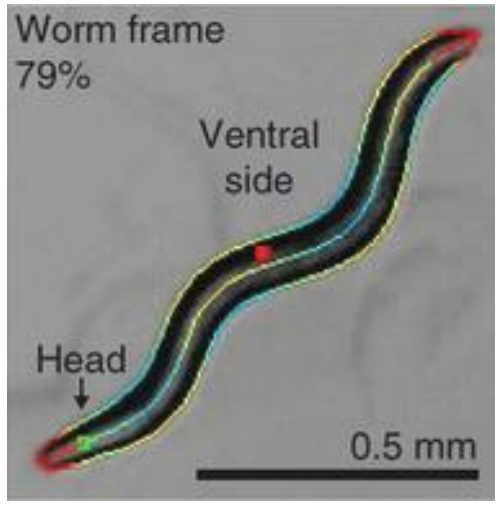
Some motivation



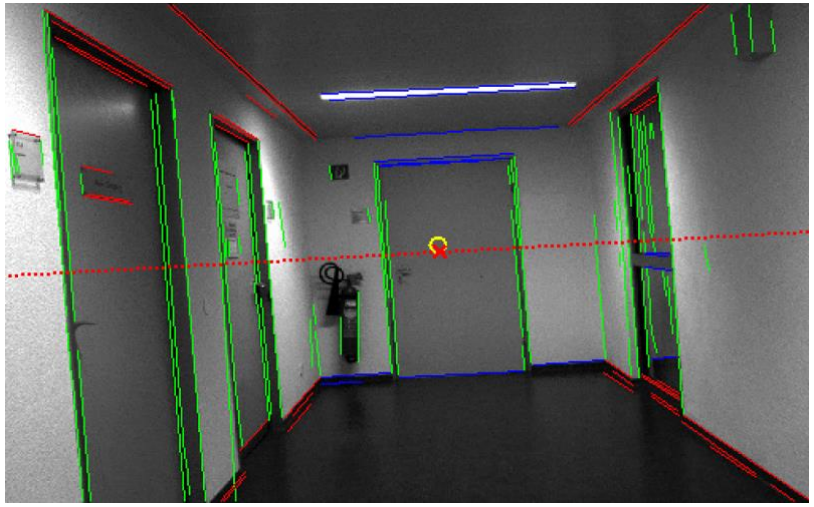
Autonomous Vehicles
(lane line detection)



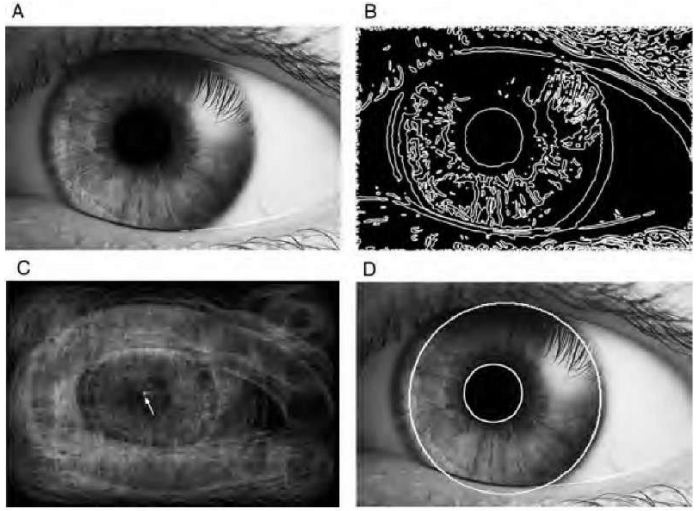
Bio-medical engineering
(blood vessel counting)



Biology
(earthworm contours)



Robotics
(scene understanding)



Psychology/ Human computer interaction
(eye tracking)

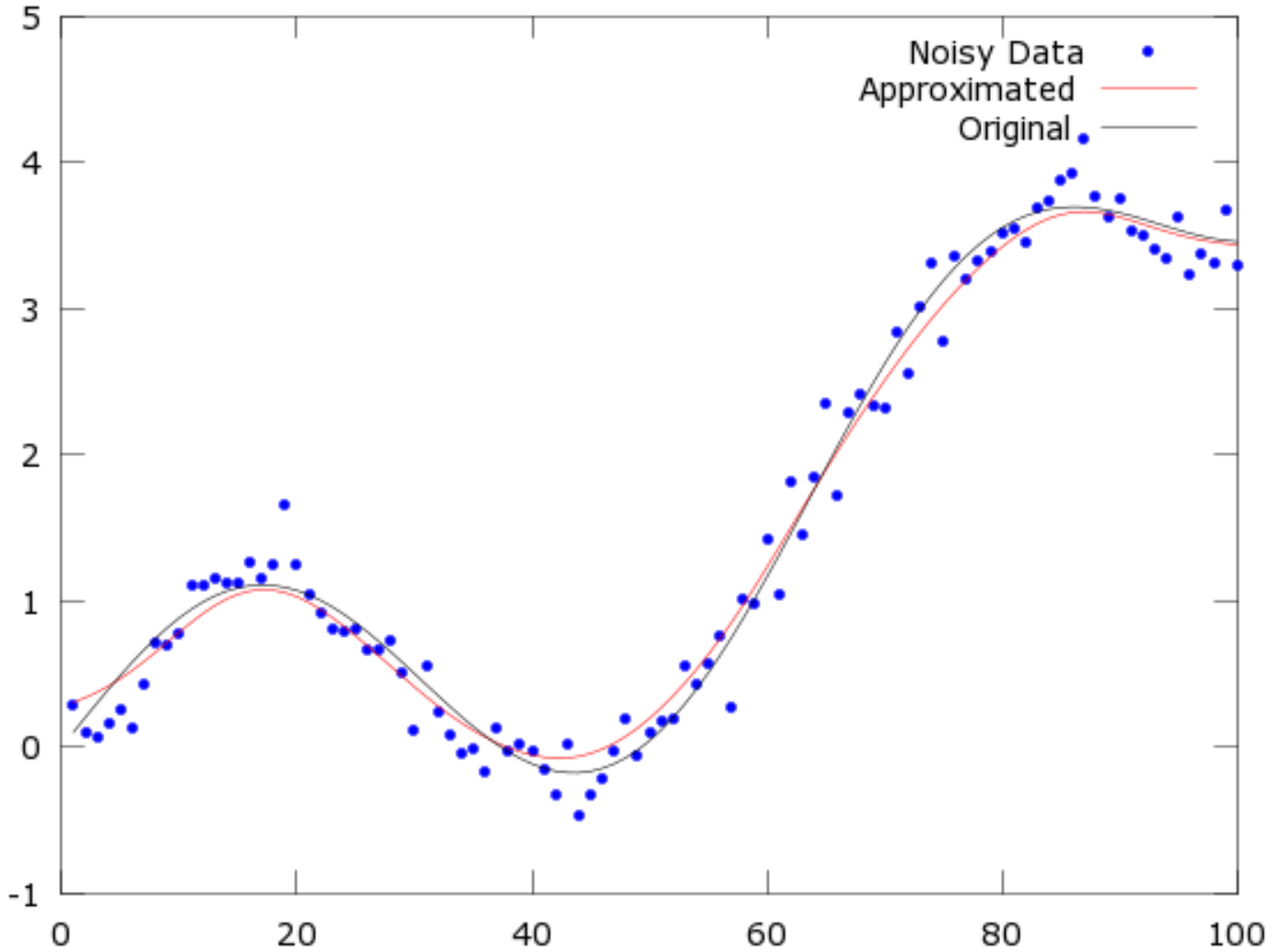
What is curve fitting

- **Curve fitting** is the process of constructing a curve, or mathematical function, that has the best fit to a series of data points. [Wikipedia]
- Input: dataset (e.g.: $\{(x_i, y_i)\}_{i=1,..,N}$ in 2D).
- Output: best representing function (e.g.: $f(x) = y$).

- This problem is also called **Regression.**

Best fit

- The best fit is the one which minimizes the error of the fitted function relative to the given data.



Error / Loss

- **Error** (also known as **loss**) is a method of evaluating how well a specific algorithm models the given data.
  - If predictions deviates too much from actual results, error would be high.
- A popular error used a lot in CV and statistics is called **MSE** (mean square error, also known as **L2 loss** or **quadratic loss**).

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2$$

- MSE: The mean of the distance from each point to the predicted regression model.
- We need to find the set of variables that minimizes the error - **MMSE** (minimum mean square error).
- Other variants of this error are **SSD (sum of square distances)** or **RMSE (root mean square error)**

Linearity assumption

- From now and until the end of the class we will assume that our fitted function $f(x)$ is linear in its unknown variables.
- linear in its unknown variables- example:
  - Fitting a line to the data $f(x) = mx + b$, (m, b) are linear in the input.
  - A parabolic fit: finding (a, b, c) for the best $f(x) = ax^2 + bx + c$ is also a linear fit! A linear combination of it's known input. One can assume $f(x) = g(x^2, x) = ax^2 + bx + c$ and see the linear combination more vividly.
- This types of problems are called **linear regression** problems.

Linearity assumption

- What is not linear in its unknown variables?

Linearity assumption

- What is not linear in its unknown variables?
- For example, an off-centered circle:

$$(x-a)^2+(y-b)^2=r^2$$
$$x^2-2ax+a^2+y^2-2by+b^2=r^2$$

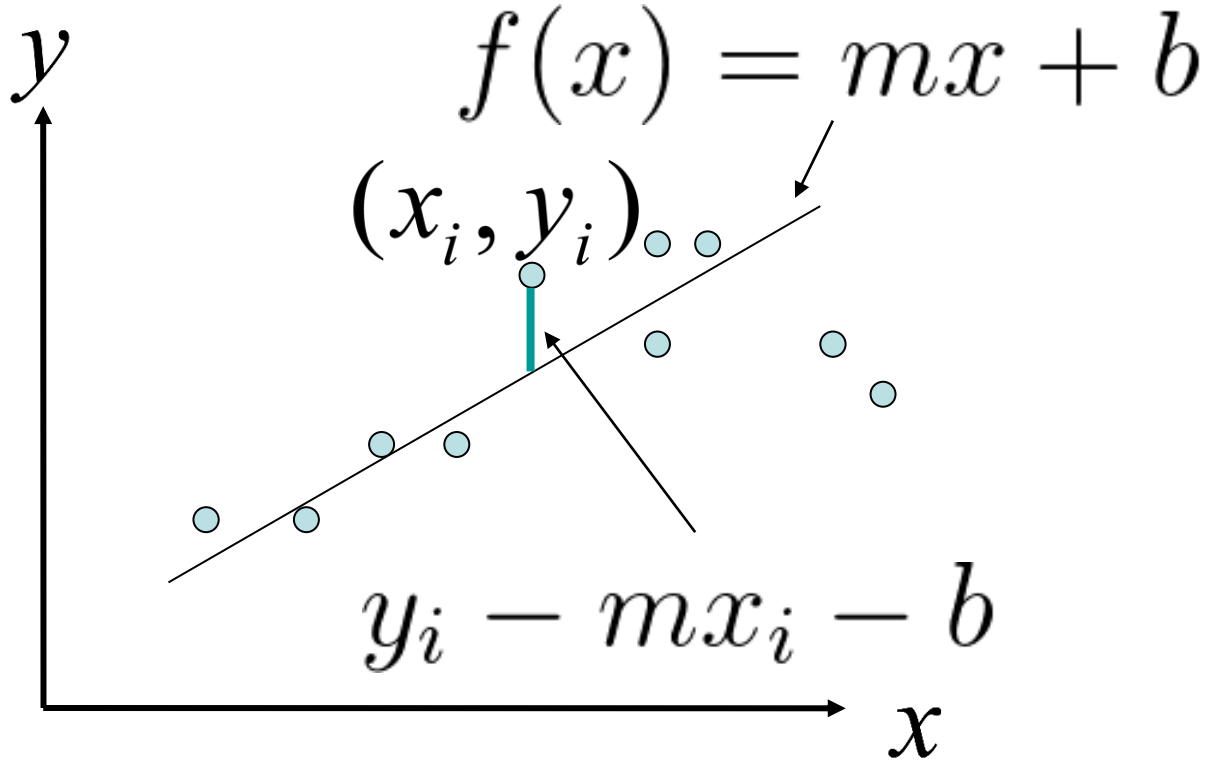
MMSE & Least squares

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2$$

- When the fitted function $f(x)$ is linear in its unknown variables (and the noisy input data is uncorrelated and have equal variance) we can use a method called **LS- least squares** (more precisely **linear least squares**) to solve the MMSE problem.
- This method is mathematically proven to be the one which minimizes the L2 error for the fitted data.

Line fitting- least squares (LS): step by step example

- Given: $\{(x_i, y_i)\}_{i=1,\ldots,N}$
  find best line representation: $f(x) = mx + b$
- The best representation (m, b) is the one that minimizes the total error e.



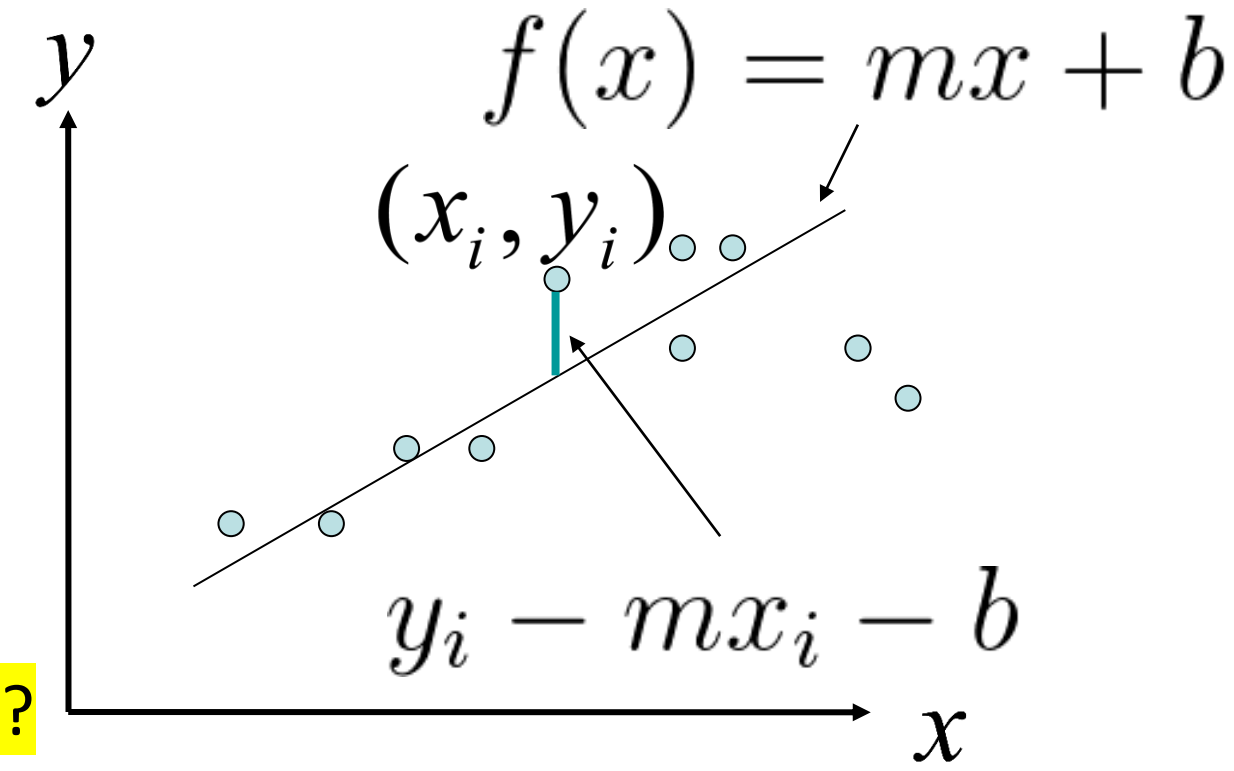
Line fitting- least squares (LS): step by step example

- Given: $\{(x_i, y_i)\}_{i=1,\ldots,N}$
  find best line representation: $f(x) = mx + b$
- The best representation (m, b) is the one that minimizes the total error e.

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2$$

$$\downarrow$$

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$



- How do we find this set of variables?

Line fitting- least squares (LS): step by step example

- Find derivatives of e with respect to both variables m, b s.t. (such that) we'll reach the minimum error (partial derivative of both variables equals zero…):

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

Line fitting- least squares (LS): step by step example

- Find derivatives of e with respect to both variables m, b s.t. (such that) we'll reach the minimum error (partial derivative of both variables equals zero…):

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i,\ \overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$

Line fitting- least squares (LS): step by step example

- Find derivatives of e with respect to both variables m, b s.t. (such that) we'll reach the minimum error (partial derivative of both variables equals zero…):

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i,\ \overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$

$$0 = \frac{\partial e}{\partial b}$$

Line fitting- least squares (LS): step by step example

- Find derivatives of e with respect to both variables m, b s.t. (such that) we'll reach the minimum error (partial derivative of both variables equals zero…):

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i,\ \overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$

$$0 = \frac{\partial e}{\partial b} = \frac{-2}{N}\sum_{i=1}^{N}(y_i - mx_i - b)$$

Line fitting- least squares (LS): step by step example

- Find derivatives of e with respect to both variables m, b s.t. (such that) we'll reach the minimum error (partial derivative of both variables equals zero…):

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i,\ \overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$

$$0 = \frac{\partial e}{\partial b} = \frac{-2}{N}\sum_{i=1}^{N}(y_i - mx_i - b) \mapsto 0 = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b) \mapsto b = \overline{y} - m\overline{x}$$

Line fitting- least squares (LS): step by step example

- Find derivatives of e with respect to both variables m, b s.t. (such that) we'll reach the minimum error (partial derivative of both variables equals zero…):

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i,\ \ \overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$

$$0 = \frac{\partial e}{\partial b} = \frac{-2}{N}\sum_{i=1}^{N}(y_i - mx_i - b) \mapsto 0 = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b) \mapsto b = \overline{y} - m\overline{x}$$

$$0 = \frac{\partial e}{\partial m} \mapsto \ldots \mapsto m = \frac{\sum_{i=1}^{N}(y_i - \overline{y})(x_i - \overline{x})}{\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

- Full derivation here.

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$y - X\beta = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} m \cdot x_1 + b \cdot 1 \\ \vdots \\ m \cdot x_N + b \cdot 1 \end{bmatrix} = \begin{bmatrix} y_1 - mx_1 - b \\ \vdots \\ y_N - mx_N - b \end{bmatrix} = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_N \end{bmatrix} = \epsilon$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$y - X\beta = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} m \cdot x_1 + b \cdot 1 \\ \vdots \\ m \cdot x_N + b \cdot 1 \end{bmatrix} = \begin{bmatrix} y_1 - mx_1 - b \\ \vdots \\ y_N - mx_N - b \end{bmatrix} = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_N \end{bmatrix} = \epsilon$$

$$\frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2 = \frac{1}{N}\sum_{i=1}^{N}\epsilon_i^2 = \frac{1}{N}\begin{bmatrix}\epsilon_1 \dots \epsilon_N\end{bmatrix}\begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_N \end{bmatrix} = \frac{1}{N}\epsilon^T\epsilon = \frac{1}{N}||\epsilon||^2 =$$

$$\frac{1}{N}||y - X\beta||^2$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$e = \frac{1}{N}||y - X\beta||^2 = \frac{1}{N}(y - X\beta)^T(y - X\beta) =$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$e = \frac{1}{N}||y - X\beta||^2 = \frac{1}{N}(y - X\beta)^T(y - X\beta) =$$

$$\frac{1}{N}(y^T y - \beta^T X^T y - y^T X\beta + \beta^T X^T X\beta) \overset{\beta^T X^T y = scalar = scalar^T = y^T X\beta}{=}$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$e = \frac{1}{N}||y - X\beta||^2 = \frac{1}{N}(y - X\beta)^T(y - X\beta) =$$

$$\frac{1}{N}(y^T y - \beta^T X^T y - y^T X\beta + \beta^T X^T X\beta) \overset{\beta^T X^T y = scalar = scalar^T = y^T X\beta}{=}$$

$$\frac{1}{N}(y^T y - 2\beta^T X^T y + \beta^T X^T X\beta)$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$e = \frac{1}{N}||y - X\beta||^2 = \frac{1}{N}(y - X\beta)^T(y - X\beta) =$$

$$\frac{1}{N}(y^T y - \beta^T X^T y - y^T X\beta + \beta^T X^T X\beta) \overset{\beta^T X^T y = scalar = scalar^T = y^T X\beta}{=}$$

$$\frac{1}{N}(y^T y - 2\beta^T X^T y + \beta^T X^T X\beta)$$

$$0 = \frac{\partial e}{\partial \beta} = 2X^T X\beta - 2X^T y$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$e = \frac{1}{N}||y - X\beta||^2 = \frac{1}{N}(y - X\beta)^T(y - X\beta) =$$

$$\frac{1}{N}(y^T y - \beta^T X^T y - y^T X\beta + \beta^T X^T X\beta) \overset{\beta^T X^T y = scalar = scalar^T = y^T X\beta}{=}$$

$$\frac{1}{N}(y^T y - 2\beta^T X^T y + \beta^T X^T X\beta)$$

$$0 = \frac{\partial e}{\partial \beta} = 2X^T X\beta - 2X^T y$$

$$X^T X\beta = X^T y$$

Line fitting - LS in matrix form

$$e = \frac{1}{N}\sum_{i=1}^{N}(y_i - mx_i - b)^2$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$e = \frac{1}{N}||y - X\beta||^2 = \frac{1}{N}(y - X\beta)^T(y - X\beta) =$$

$$\frac{1}{N}(y^T y - \beta^T X^T y - y^T X\beta + \beta^T X^T X\beta) \overset{\beta^T X^T y = scalar = scalar^T = y^T X\beta}{=}$$

$$\frac{1}{N}(y^T y - 2\beta^T X^T y + \beta^T X^T X\beta)$$

$$0 = \frac{\partial e}{\partial \beta} = 2X^T X\beta - 2X^T y$$

$$X^T X\beta = X^T y$$

$$\beta = (X^T X)^{-1} X^T y$$

Side note: pseudoinverse

$$\beta = \underbrace{(X^T X)^{-1} X^T}\, y$$

- This is a known result which is called **the pseudoinverse matrix** of X. It is also known as Moore–Penrose inverse.
- If $X^T X$ is not invertible, the solution will be: $\beta = (X^T X + \epsilon I)^{-1} X^T y$
- The solution above is **a solution for any linear LS problem that is over-determined** (==more equations than unknowns).

Curve fitting - LS

- What about curve fitting?
- Recall the matrix linear LS result: $\beta = (X^T X)^{-1} X^T y$

- Notice that in our derivation we didn't use the assumption that the data is linear…
- For example- data set of a parabola:

  $\{(x_i, y_i)\}\ s.t.\ ax_i^2 + bx_i + c = y_i$

  – How the matrices X, y, β will look like?

Curve fitting - LS

- data set of a parabola:

$$\{(x_i, y_i)\} \; s.t. \; ax_i^2 + bx_i + c = y_i$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix} \quad X = \begin{bmatrix} x_1^2 & x_1 & 1 \\ \vdots & \vdots & \vdots \\ x_N^2 & x_N & 1 \end{bmatrix} \quad \beta = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

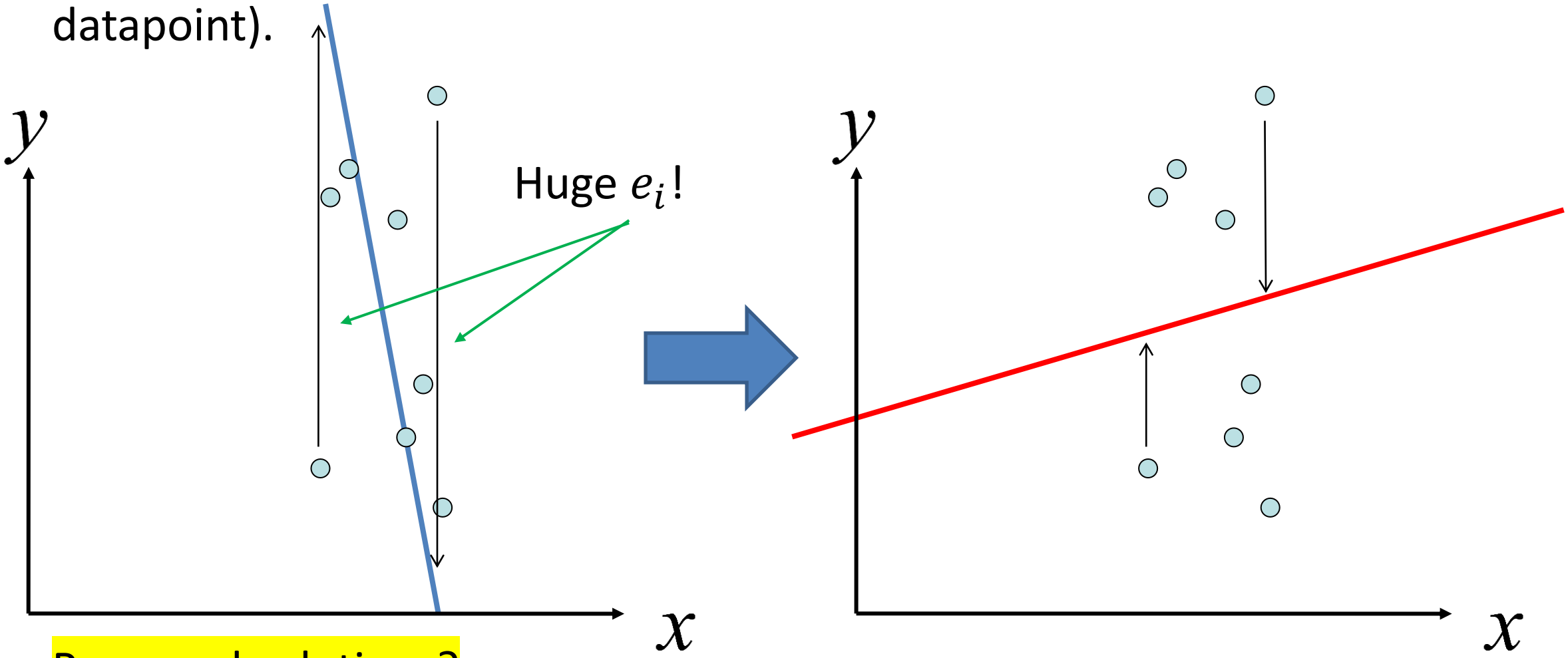
- The solution of this LS problem is the same:

$$\beta = (X^T X)^{-1} X^T y$$

- least_squares.ipynb

Problem 1: fitting vertical data

- Near vertical data is hard to fit since the error is computed perpendicular to x axis- the bigger the error, the bigger the error squared! (more weight for this datapoint).



- Proposed solutions?

Problem 1: fitting vertical data

- One possible solution is making all errors weigh the same (and not squared). This will make the far points have the same impact on error as closer points. One such error **MAE** (mean absolute error, also known as **L1 loss**) instead of MSE.

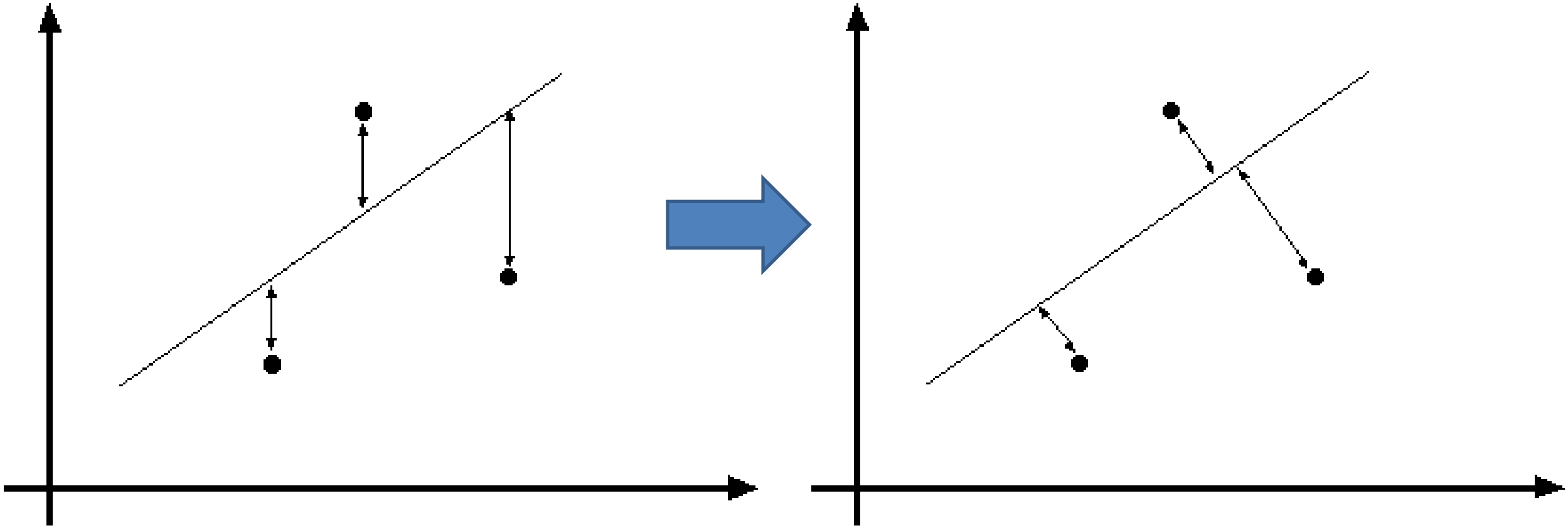
$$e_{MAE} = \frac{1}{N}\sum_{i=1}^{N} ||y_i - f(x_i)||$$

  - The derivation of MAE is out of the class scope, but some details can be found here.
- Another possible solution is computing the error distance of each point in a different way- one that considers the y data as well. One such algorithm is **total least squares.**

TOC

- Least squares
- **Total least squares**
- RANSAC

Line fitting- total least squares



Line fitting- TLS

- Another representation of a line: $ax + by + c = 0$
- Distance between a point (x_i, y_i) and line: $$d_i = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}}$$

Side note: general line equation

- $ax + by + c = 0$ is the generalized linear equation.
- If $b \neq 0$ we can divide by b and get the more known linear equation:

$$y = mx + n$$

- The (first) general equation can represent equations like a vertical line $x = const$, that the letter function can't.
- When using the general equation, we have a degree of freedom:

$$ax + by + c = 0 \leftrightarrow 2ax + 2by + 2c = 0$$

- To combat that, we can constraint the variables
  - for example, define $a = 1$ or $a^2 + b^2 = 1$

Line fitting- TLS

- Another representation of a line: $ax + by + c = 0$
- Distance between a point (x_i, y_i) and line: $$d_i = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}}$$

- Let's decide that, for later purpose, $a^2 + b^2 = 1$.

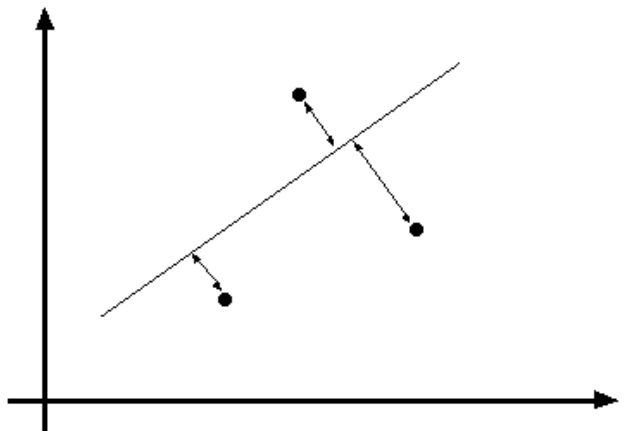
Line fitting- TLS

- Another representation of a line: $ax + by + c = 0$
- Distance between a point (x_i, y_i) and line: $$d_i = \frac{|ax_i + by_i + c|}{\sqrt{a^2 + b^2}}$$

- Let's decide that, for later purpose, $a^2 + b^2 = 1$.

- The error to be minimized:

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$



Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

$$c = -\frac{a}{N}\sum_{i=1}^{N} x_i - \frac{b}{N}\sum_{i=1}^{N} y_i = -a\overline{x} - b\overline{y}$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

$$c = -\frac{a}{N}\sum_{i=1}^{N} x_i - \frac{b}{N}\sum_{i=1}^{N} y_i = -a\overline{x} - b\overline{y}$$

$$e = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i - a\overline{x} - b\overline{y})^2$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

$$c = -\frac{a}{N}\sum_{i=1}^{N} x_i - \frac{b}{N}\sum_{i=1}^{N} y_i = -a\overline{x} - b\overline{y}$$

$$e = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i - a\overline{x} - b\overline{y})^2 = \frac{1}{N}\sum_{i=1}^{N}(a(x_i - \overline{x}) + b(y_i - \overline{y}))^2$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

$$c = -\frac{a}{N}\sum_{i=1}^{N} x_i - \frac{b}{N}\sum_{i=1}^{N} y_i = -a\overline{x} - b\overline{y}$$

$$e = \frac{1}{N}\sum_{i=1}^{N}(ax_i+by_i-a\overline{x}-b\overline{y})^2 = \frac{1}{N}\sum_{i=1}^{N}(a(x_i-\overline{x})+b(y_i-\overline{y}))^2 = \frac{1}{N}\left\| \begin{bmatrix} x_1-\overline{x} & y_1-\overline{y} \\ \vdots & \vdots \\ x_N-\overline{x} & y_N-\overline{y} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2 =$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

$$c = -\frac{a}{N}\sum_{i=1}^{N} x_i - \frac{b}{N}\sum_{i=1}^{N} y_i = -a\overline{x} - b\overline{y}$$

$$e = \frac{1}{N}\sum_{i=1}^{N}(ax_i+by_i-a\overline{x}-b\overline{y})^2 = \frac{1}{N}\sum_{i=1}^{N}(a(x_i-\overline{x})+b(y_i-\overline{y}))^2 = \frac{1}{N}\left\| \begin{bmatrix} x_1-\overline{x} & y_1-\overline{y} \\ \vdots & \vdots \\ x_N-\overline{x} & y_N-\overline{y} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2 =$$

$$\frac{1}{N}\|X\beta\|^2 = \frac{1}{N}\beta^T X^T X\beta$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

$$c = -\frac{a}{N}\sum_{i=1}^{N} x_i - \frac{b}{N}\sum_{i=1}^{N} y_i = -a\overline{x} - b\overline{y}$$

$$e = \frac{1}{N}\sum_{i=1}^{N}(ax_i+by_i-a\overline{x}-b\overline{y})^2 = \frac{1}{N}\sum_{i=1}^{N}(a(x_i-\overline{x})+b(y_i-\overline{y}))^2 = \frac{1}{N}\left\| \begin{bmatrix} x_1-\overline{x} & y_1-\overline{y} \\ \vdots & \vdots \\ x_N-\overline{x} & y_N-\overline{y} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2 =$$

$$\frac{1}{N}\|X\beta\|^2 = \frac{1}{N}\beta^T X^T X \beta$$

$$\begin{cases} minimize & \frac{1}{N}\beta^T X^T X \beta \\ s.t. & a^2 + b^2 = 1 \end{cases}$$

Line fitting- TLS- derivation

$$e_{MSE} = \frac{1}{N}\sum_{i=1}^{N} d_i^2 = \frac{1}{N}\sum_{i=1}^{N}(ax_i + by_i + c)^2$$

$$0 = \frac{\partial e}{\partial c} = \frac{1}{N}\sum_{i=1}^{N} 2(ax_i + by_i + c)$$

$$c = -\frac{a}{N}\sum_{i=1}^{N} x_i - \frac{b}{N}\sum_{i=1}^{N} y_i = -a\overline{x} - b\overline{y}$$

$$e = \frac{1}{N}\sum_{i=1}^{N}(ax_i+by_i-a\overline{x}-b\overline{y})^2 = \frac{1}{N}\sum_{i=1}^{N}(a(x_i-\overline{x})+b(y_i-\overline{y}))^2 = \frac{1}{N}\left\| \begin{bmatrix} x_1-\overline{x} & y_1-\overline{y} \\ \vdots & \vdots \\ x_N-\overline{x} & y_N-\overline{y} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right\|^2 =$$

$$\frac{1}{N}\|X\beta\|^2 = \frac{1}{N}\beta^T X^T X \beta$$

$$\begin{cases} minimize & \frac{1}{N}\beta^T X^T X\beta \\ s.t. & a^2+b^2=1 \end{cases} = \begin{cases} minimize & \beta^T X^T X\beta \\ s.t. & \beta^T\beta = 1 \end{cases}$$

Line fitting- TLS -the minimization problem

- The minimization problem is:

$$\begin{cases} minimize & \beta^T X^T X \beta \\ s.t. & \beta^T \beta = 1 \end{cases}$$

- Recall eigendecomposition: $Av = \lambda v \mapsto v^T A v = \lambda$
  - Also recall that each eigenvector v is normalized ($||v|| = v^T v = 1$).
- The solution to the minimization problem above is the eigenvector corresponding to smallest eigenvalue of $X^T X$.

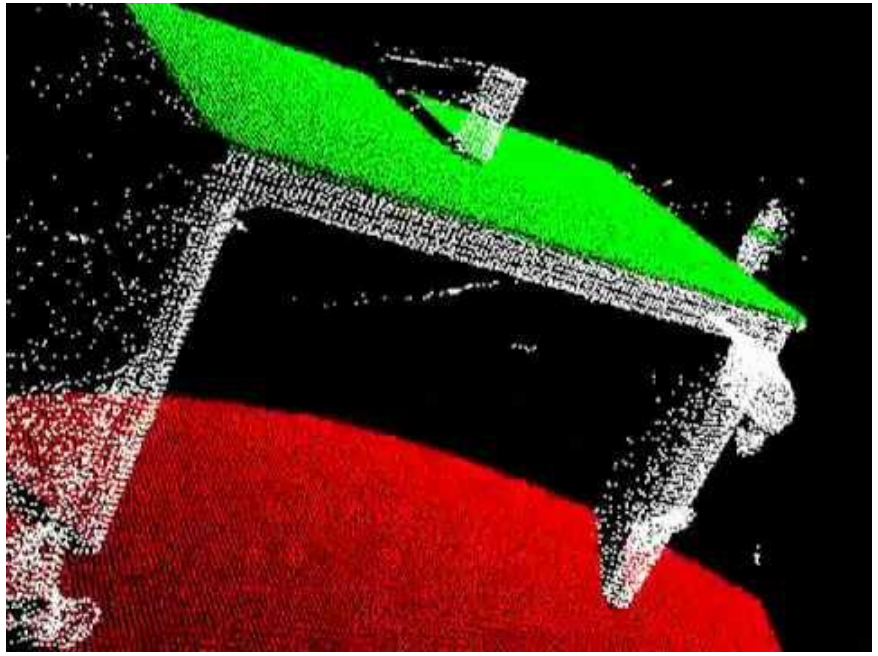
- **Watch out:** trying to minimize the problem above without the constraint $\beta^T \beta = 1$ will result with the trivial solution of $\beta = 0$.

The planarity assumption

- This method works for fitting all kind of planar data in ND space, because of the planar representation and a distance function as we've seen above.
  - For example, in 3D:

$$ax + by + cz + d = 0$$

$$\text{dist} = \frac{|ax + by + cz + d|}{\sqrt{a^2 + b^2 + c^2}}$$

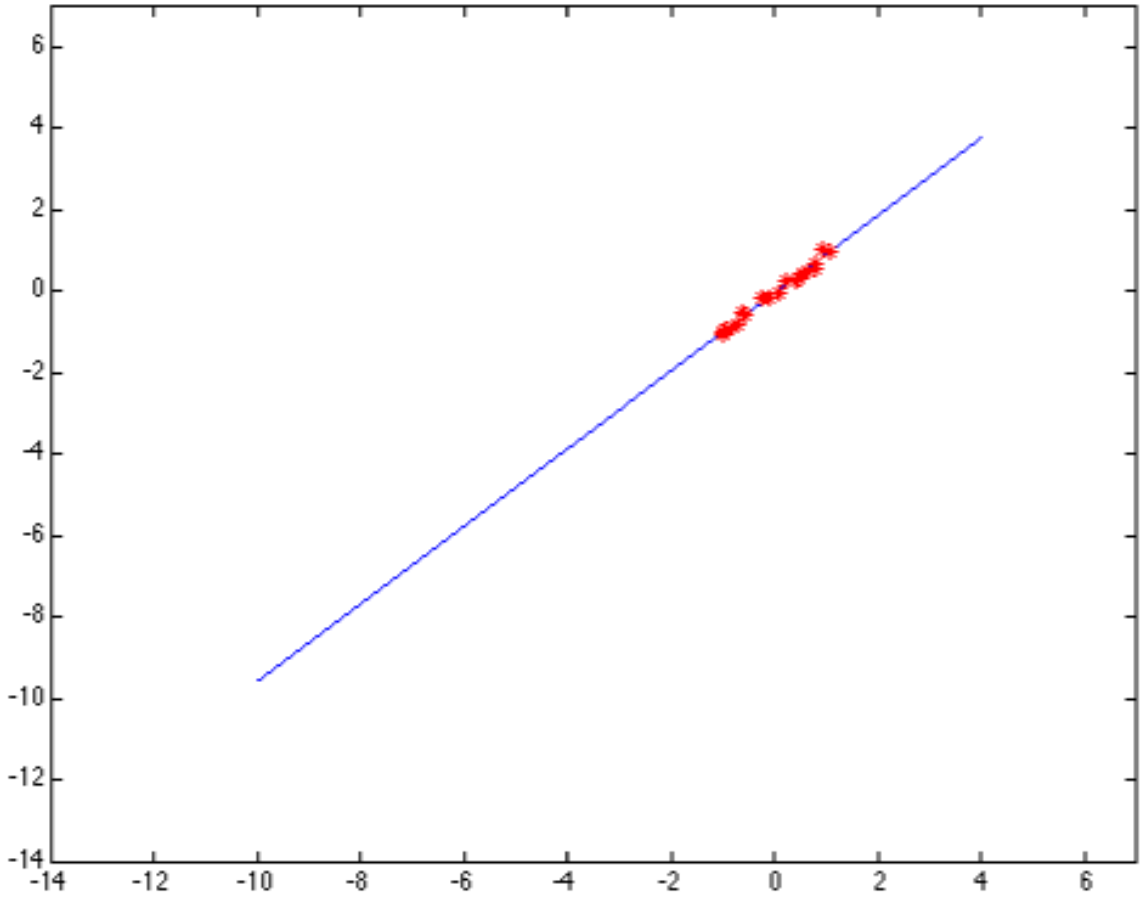


- TLS for non-planar data also exist, but this can't be solved as before (the planarity assumption was used). This topic is out of scope- proof and examples here.

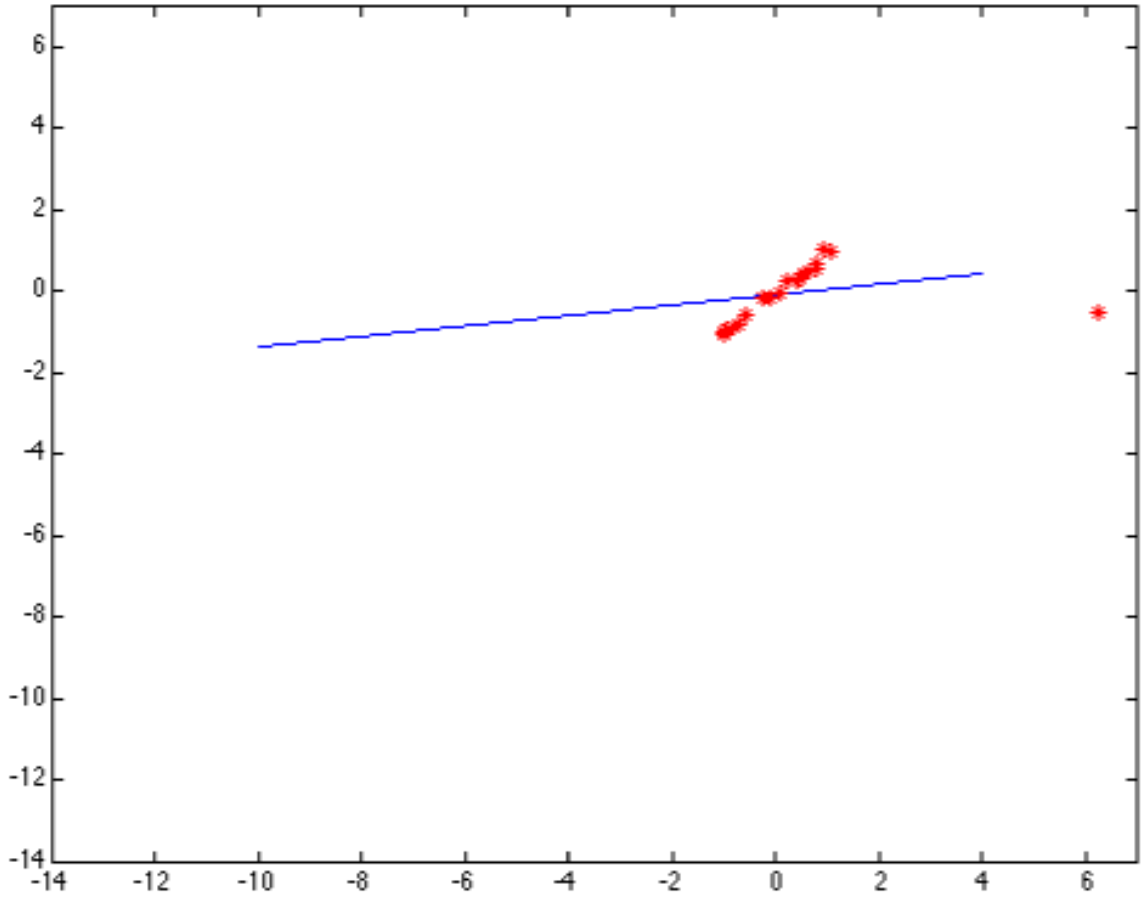
- least_squares.ipynb

Problem 2: fitting with outliers datapoints

- **Outlier**: a data point that differs significantly from other observations. [Wikipedia]



Least-squares error fit



Squared error heavily penalizes outliers

Problem 2: fitting with outliers datapoints

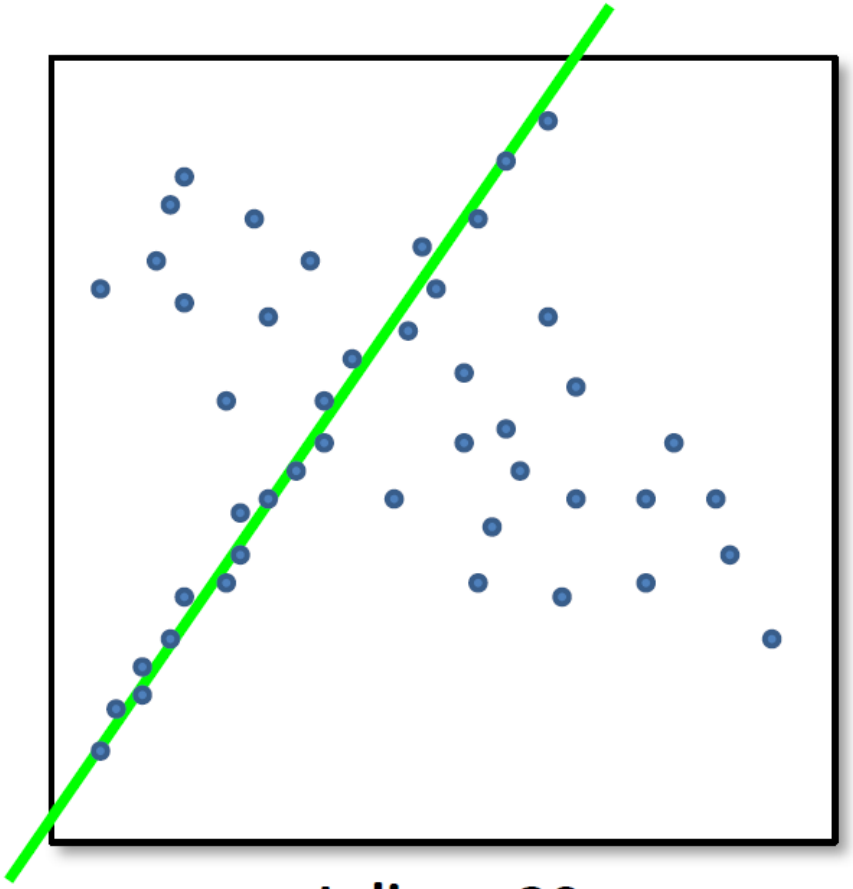
- One possible solution: MAE (again)- for less penalization on outliers.
- A better solution: removing the outliers! Possible algorithm to use: **RANSAC**

TOC

- Least squares
- Total least squares
- **RANSAC**

RANSAC

- **Random sample consensus** (**RANSAC**) is an iterative method to estimate parameters of a mathematical model from a set of observed data that contains outliers. [Wikipedia]



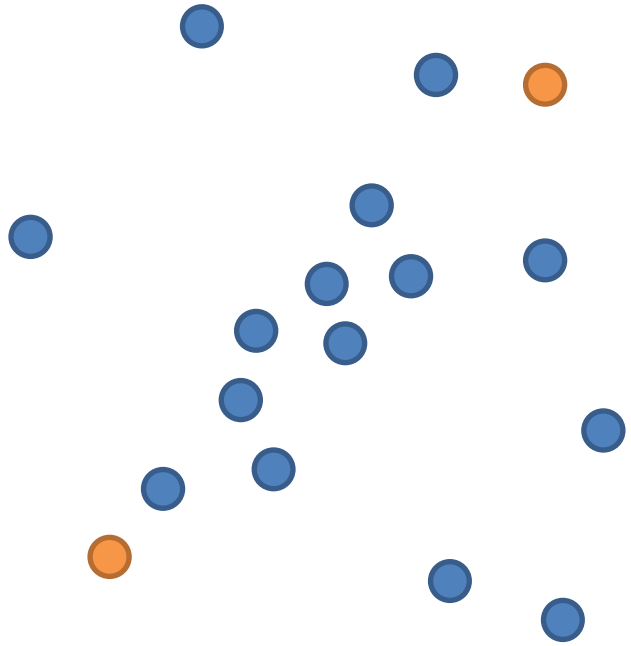
RANSAC algorithm

```
While searching for best model fit:
      Select a random subset of the original data.
      Fit a model to the data subset.
      find inliers that fit the given model.
      if number_inliers is bigger than the old best model
      number_inliers:
            Save as best model
```

RANSAC algorithm - step by step - 1

Select a random subset of the original data.

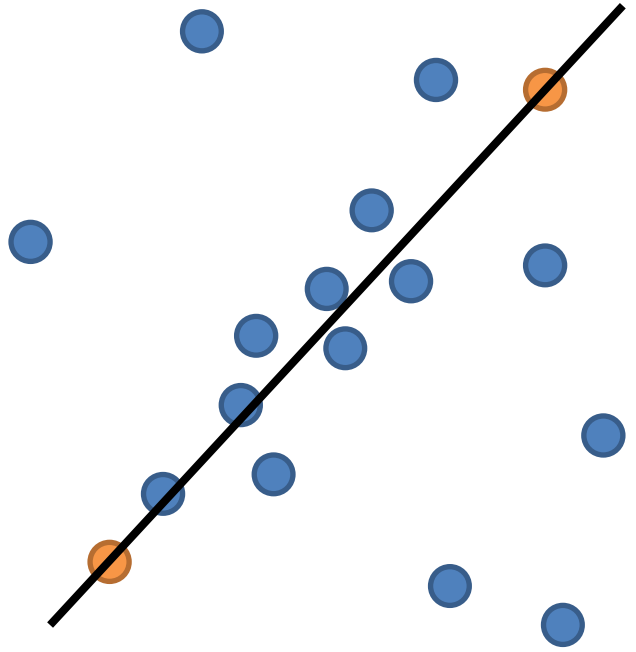
- The number of samples in the subset is the smallest needed to determine the model parameters (== number of unknow variables).
- Examples:
  - For line fit- only 2 datapoints.
  - For parabola fit- 3 datapoints.



RANSAC algorithm - step by step - 2

Fit a model to the data subset.

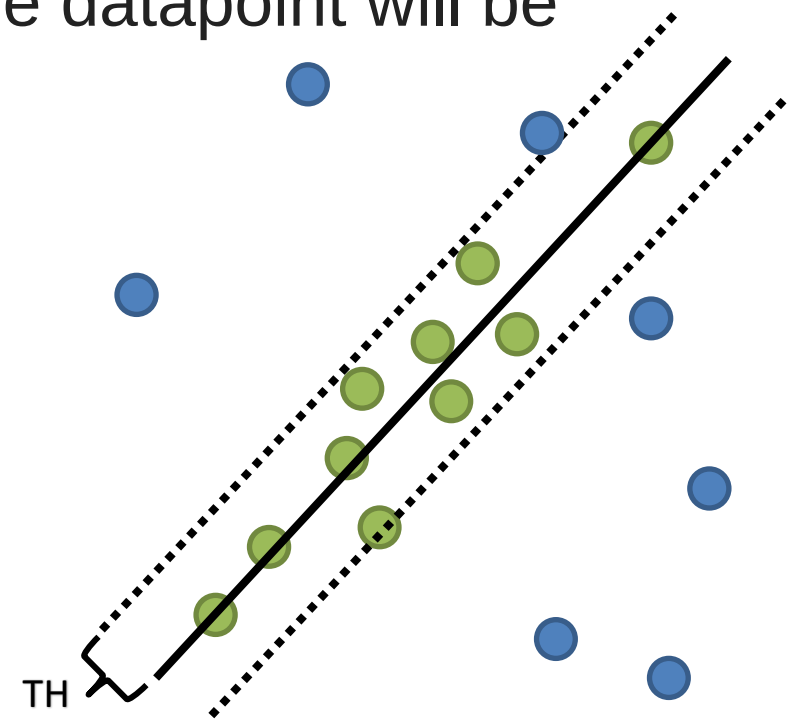
- Can be done using a chosen algorithm (for example LS).



RANSAC algorithm - step by step - 3

find inliers that fit the given model.

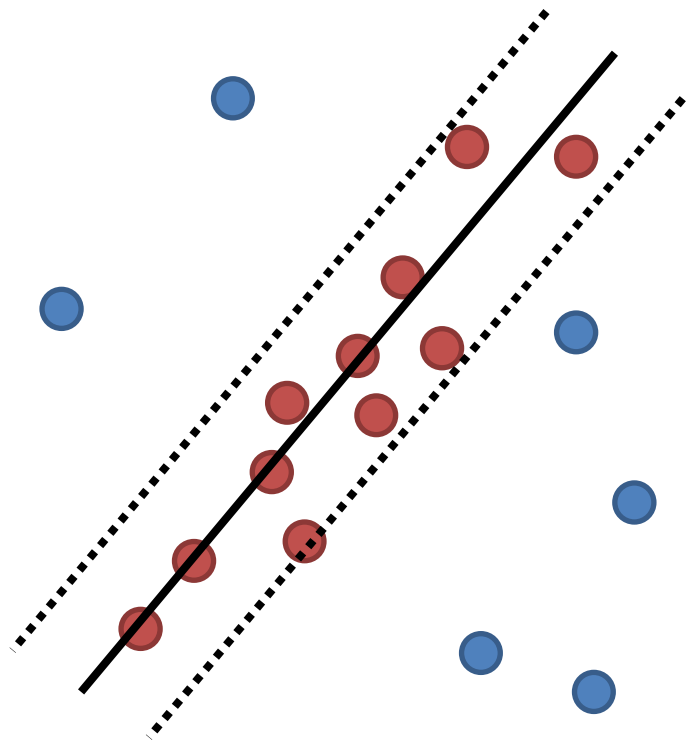
- Test all dataset against the fitted model. Points that fit the estimated model well, according to some chosen loss function, are considered as part of the consensus set. This points are called **inliers**.
- A possible loss function is MSE of the distances (same as TLS).
  - Choose a threshold for the error: below this TH the datapoint will be consider as an inlier.



RANSAC algorithm - step by step - 4

```
if number_inliers is bigger than the old best model number_inliers:
        Save as best model
```

- Trivial…
- An improvement to the final step can be iteratively re-fitting the model with all inliers to find a better describing model.



RANSAC algorithm - step by step - 0

```
While searching for best model fit:
```

- How do we know when to stop? How much we need to iterate before getting the best model?
- Let’s look at a possible statistical model for this question.

RANSAC convergence

- Denote

$\omega = \frac{\#\ inliers}{\#\ total\ datapoints}$:

  - getting ω ratio will be considered as reaching the best model.
  - This ratio is usually user specified according to dataset properties (or educated guess).

k: number of iterations (will be calculated).

p: percentage chance of achieving best model in chosen k iterations (also user specified).

n: number of initially sampled subset datapoints.

RANSAC convergence

- ω^n is the probability that all n points are inliers.
- $1 - \omega^n$ is the probability that at least one of the n points is an outlier.
- $1 - p = (1 - \omega^n)^k$ is the probability that in k iterations we will always have at least one outlier (meaning we didn't get best match).
- $k = \frac{\log(1-p)}{\log(1-\omega^n)}$

- least_squares.ipynb