Machine Learning - Regression

Jeremy L Wyatt

Notation Change

- In previous lectures I denoted a pattern within the training set using a superscript, here I use a subscript

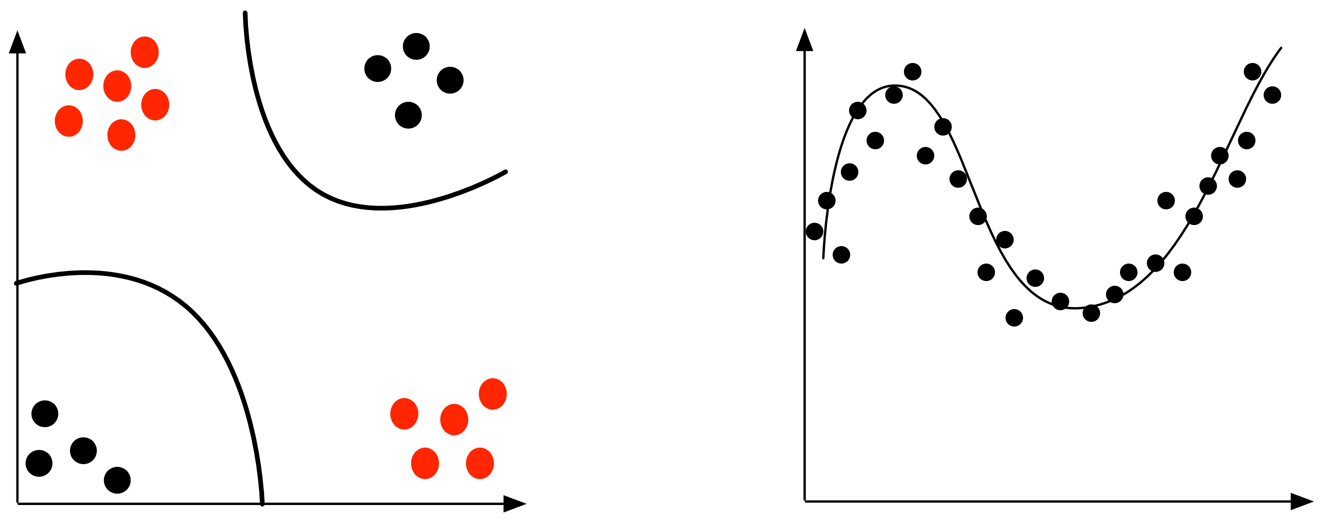
$$T = \left\{ \left(\vec{x}^1, t^1\right), \left(\vec{x}^2, t^2\right) \cdots, \left(\vec{x}^k, t^k\right) \right\} \qquad Tr = \{(x_1, t_1), (x_2, t_2) \ldots (x_k, t_k)\}$$

- In this lecture, when I use superscripts they will typically denote powers in a polynomial

- I also continue to use subscripts to denote an element of a weight vector

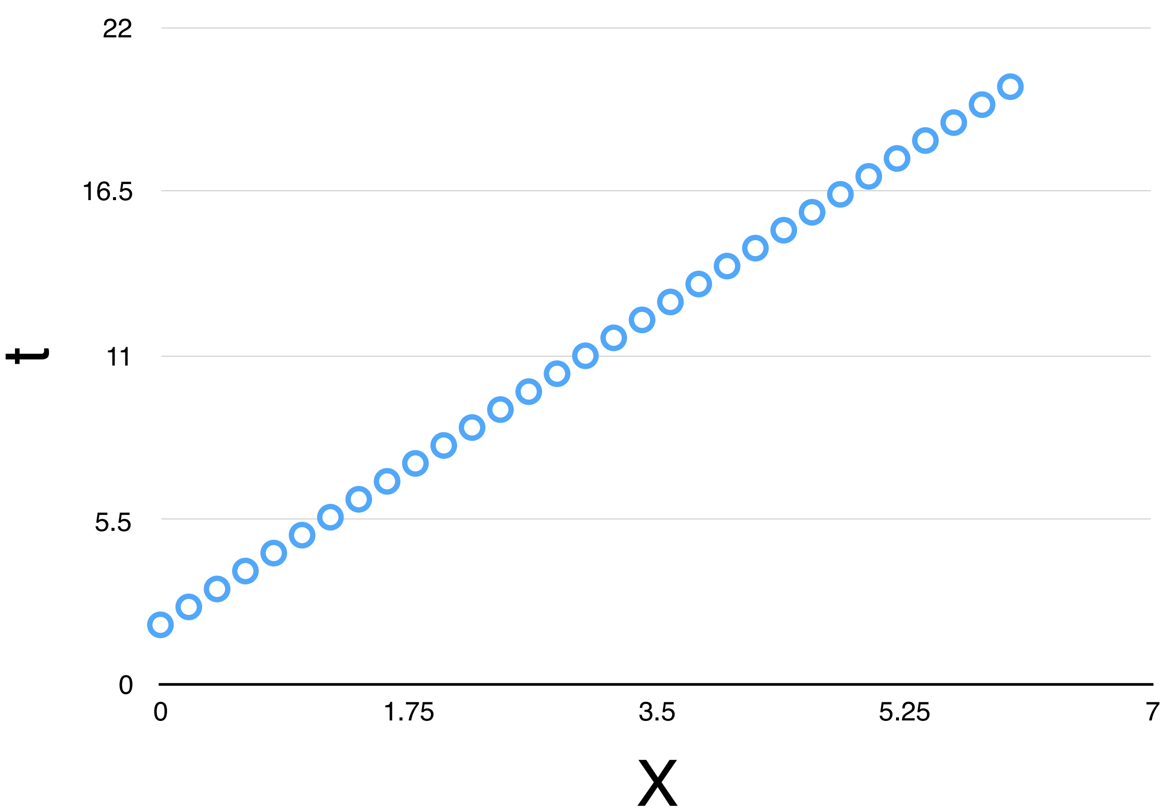
- This should not cause confusion here as most of the cases will be predicting a scalar valued output from a scalar valued input

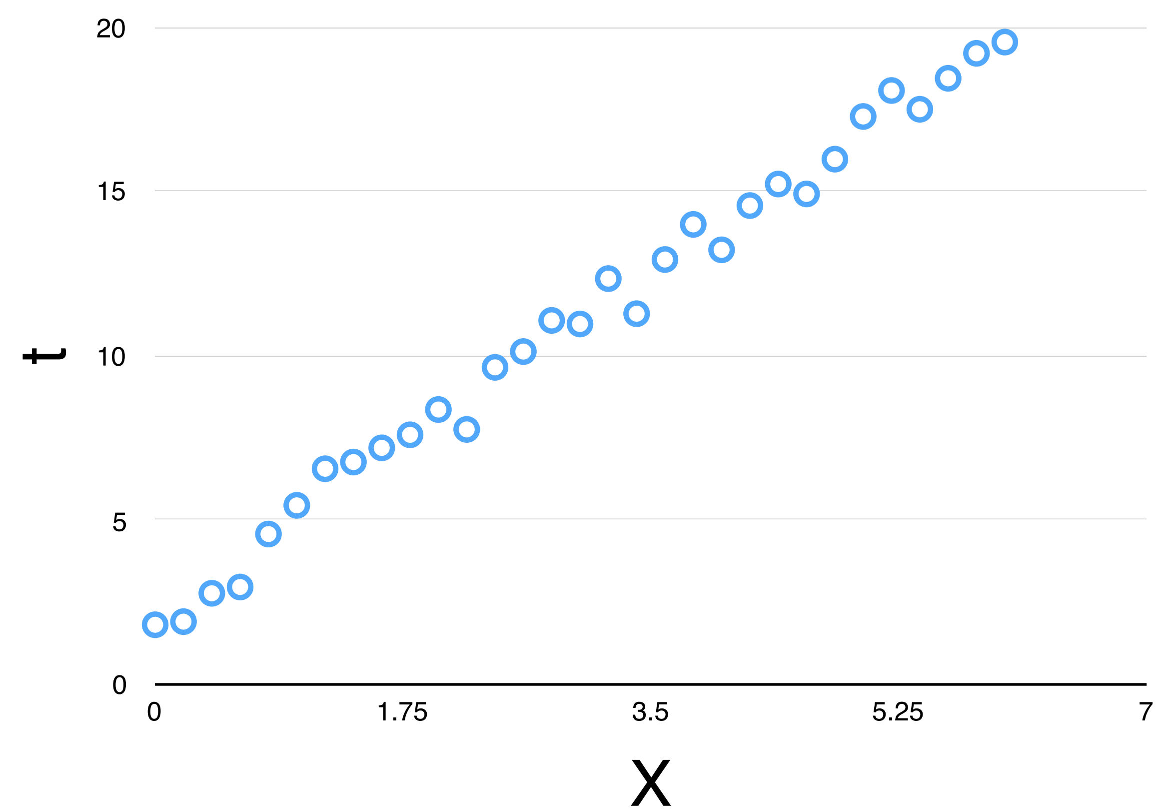
Classification vs. Regression



- RECALL: regression means there are no classes to predict
- Regression is prediction of the value of a continuous variate(s) given an input variable(s)
- Here we will study **univariate regression** (one output) with a single input variable
- i.e. predicting a scalar value **y** from a scalar value **x**

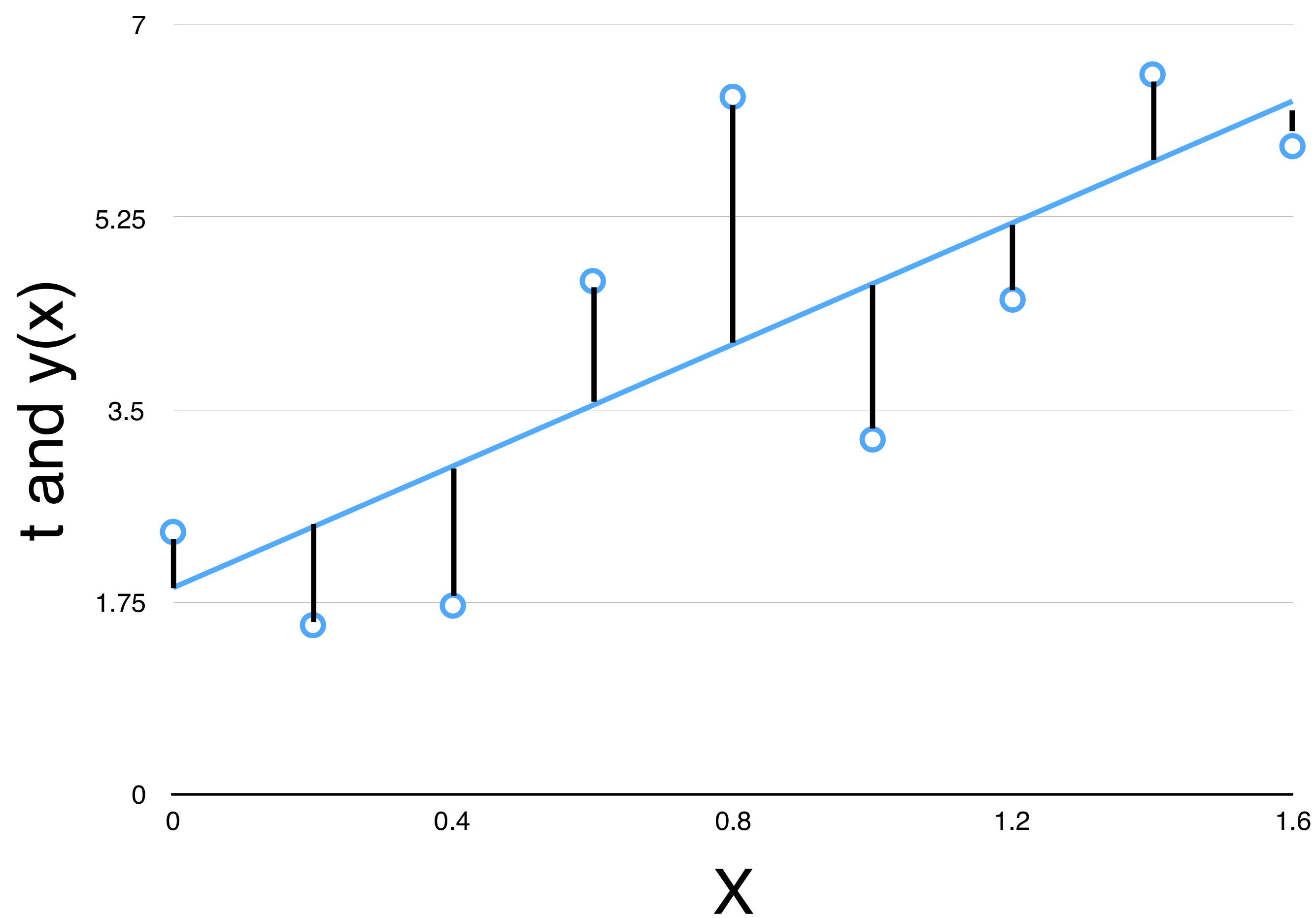
Univariate Linear Regression





- Take a simple linear relationship between x and t $t = 3x + 2$
- or add noise $t = 3x + 2 + \eta$ where $-1 < \eta < 1$
- How can we find a predictive model of the relationship from the data? Here this means fitting a straight line.

Residuals = prediction errors



y(x) is our prediction of t

$$\text{Sum of the Squared Residuals} = SSR = \sum_{i=1}^{k} (t_i - y(x_i))^2$$

Least mean squares

- Iterative fitting of the parameters of a straight line is one way. i.e. if $y(x) = ax + b$ find a and b
- Data set $Tr = \{(x_1, t_1), (x_2, t_2) \ldots (x_k, t_k)\}$
- Initialise a and b
- Loop until SSR is small where $SSR = \sum_{i=1}^{k} (t_i - y(x_i))^2$
  - For each pattern (x_i, t_i)

$$a' = a + \alpha(t_i - y(x_i))x_i$$

$$b' = b + \alpha(t_i - y(x_i))$$

Batch LMS

- Iterative fitting of the parameters of a straight line is one way.
  i.e. if $y(x) = ax + b$ find a and b
- Data set $Tr = \{(x_1, t_1), (x_2, t_2) \ldots (x_k, t_k)\}$
- Initialise a and b
- Loop until SSR is small where $SSR = \sum_{i=1}^{k} (t_i - y(x_i))^2$
  - For each pattern (x_i, t_i)

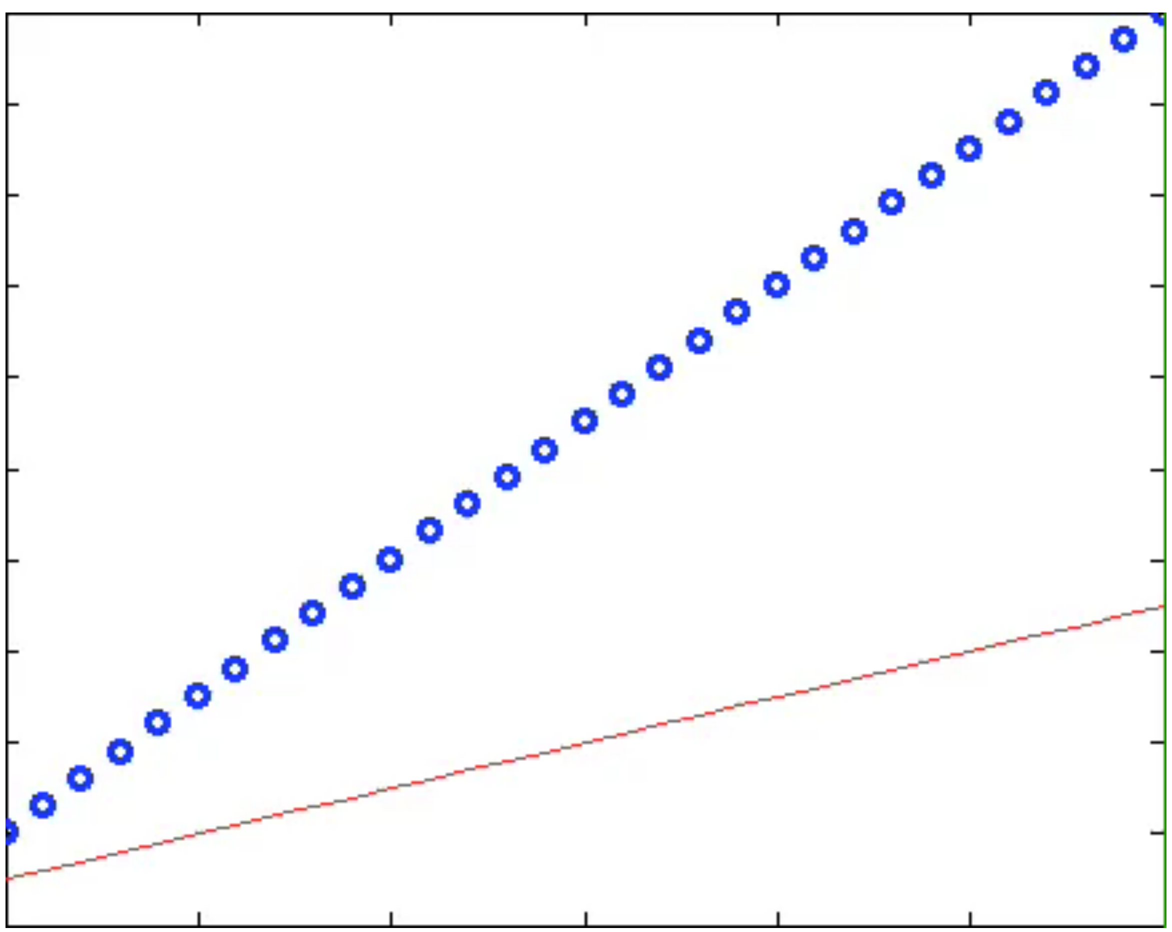
$$\Delta a = \Delta a + \alpha(t_i - y(x_i))x_i$$

$$\Delta b = \Delta b + \alpha(t_i - y(x_i))$$

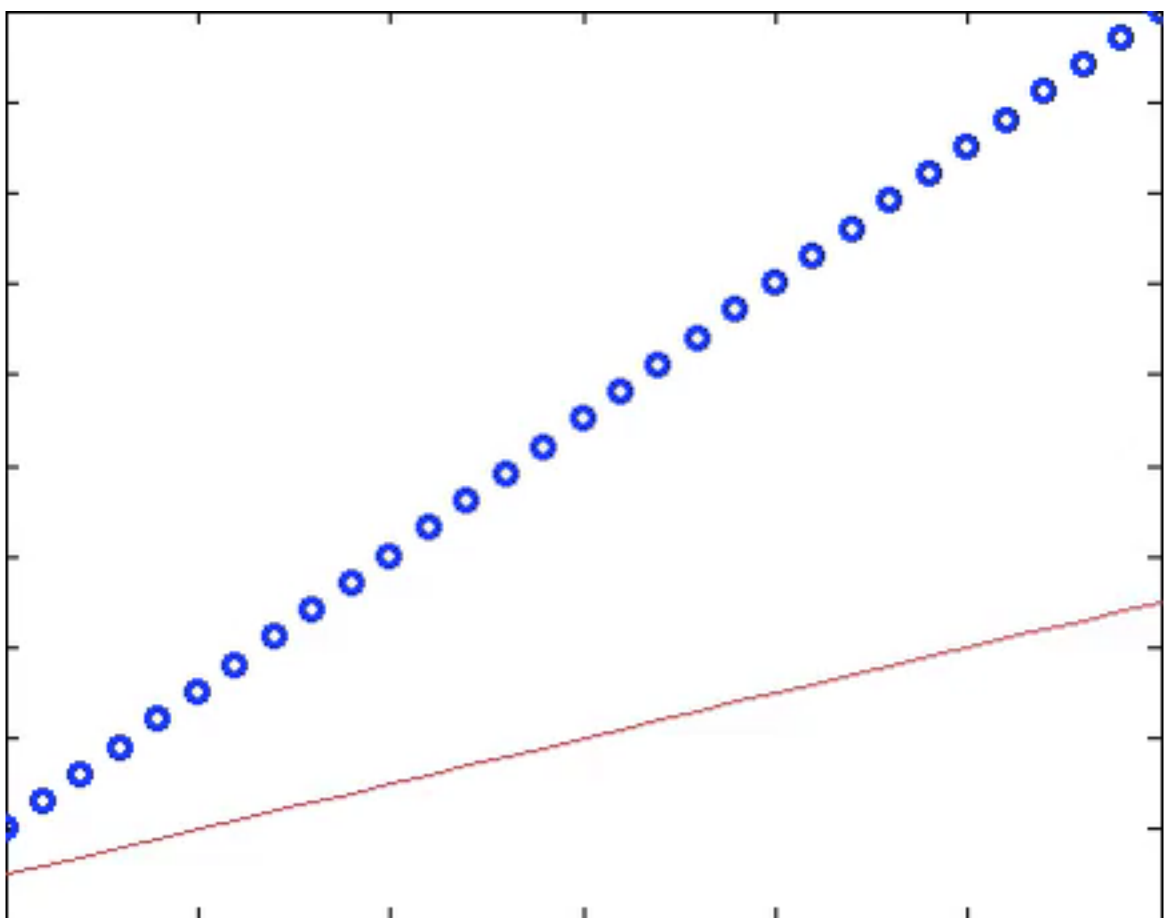
  - After all patterns

$$a' = a + \Delta a \qquad b' = b + \Delta b$$

Example runs: no noise

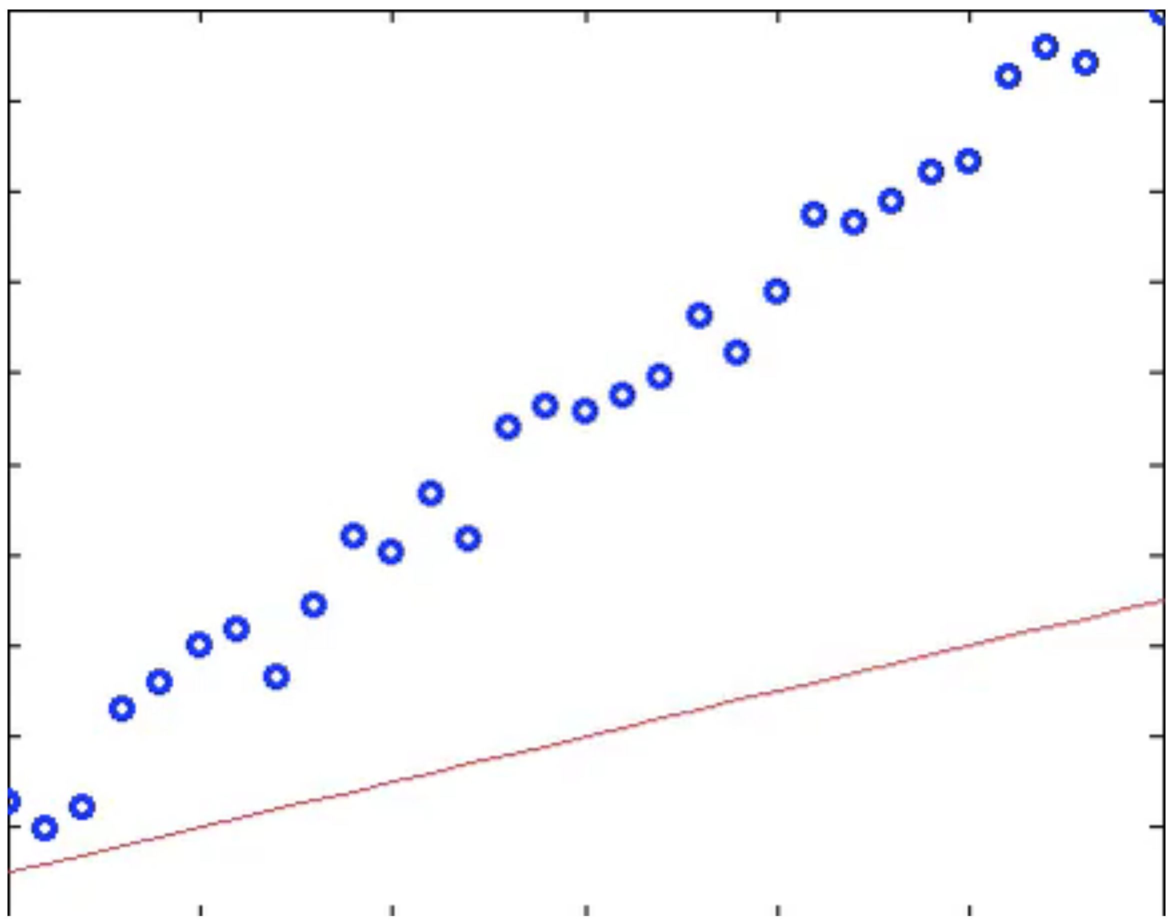


LMS 0.0001

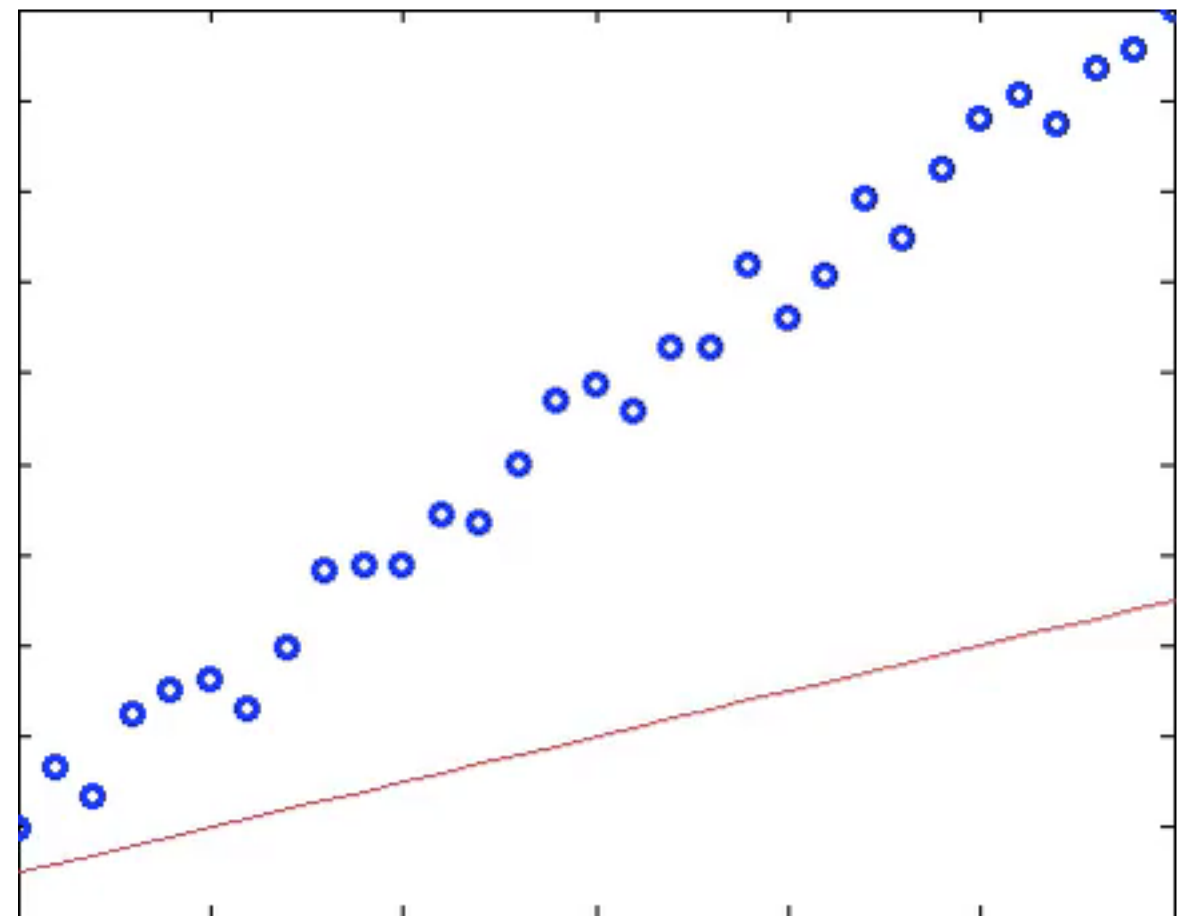


Batch LMS 0.0001

Example runs: noise

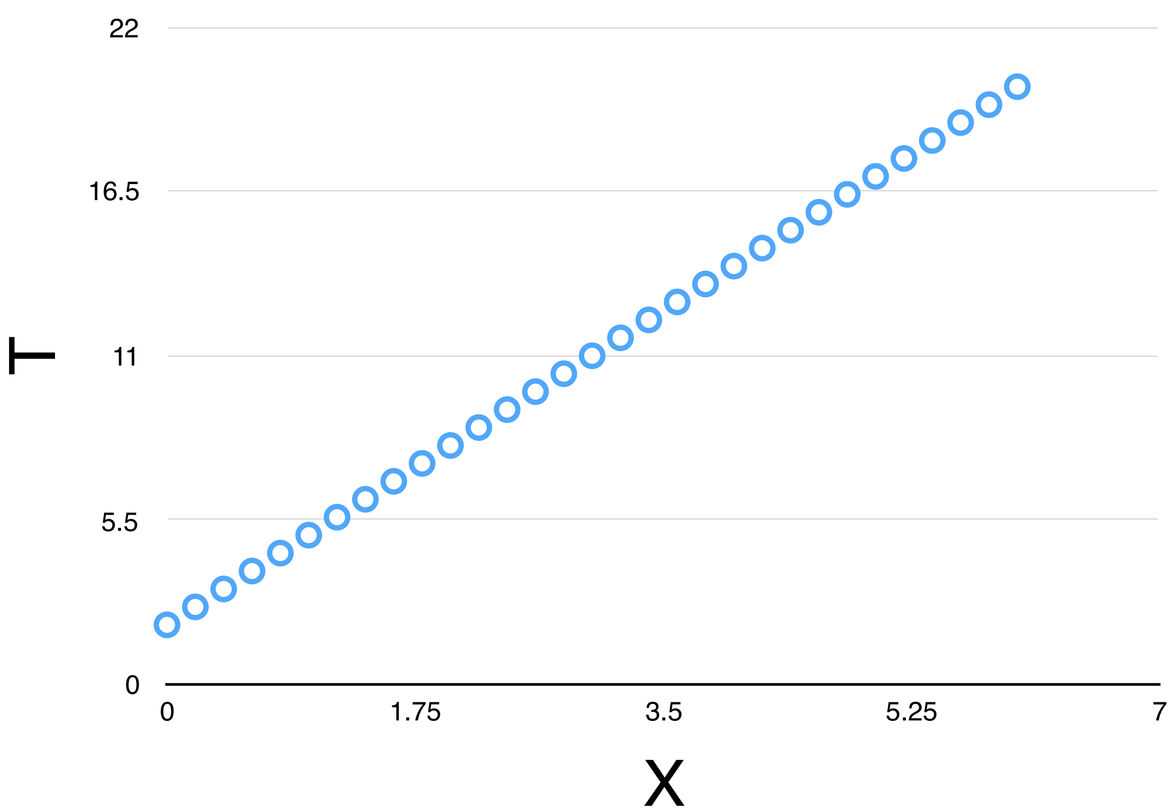


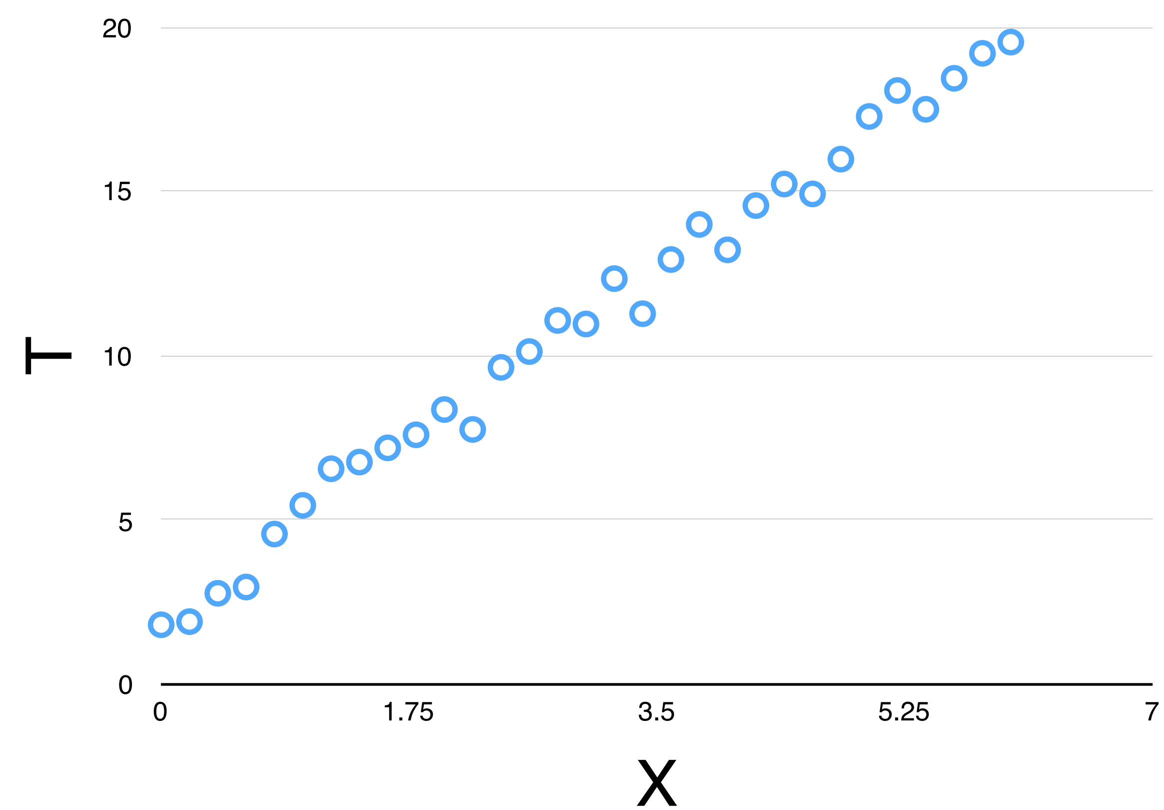
LMS 0.001



Batch LMS 0.001

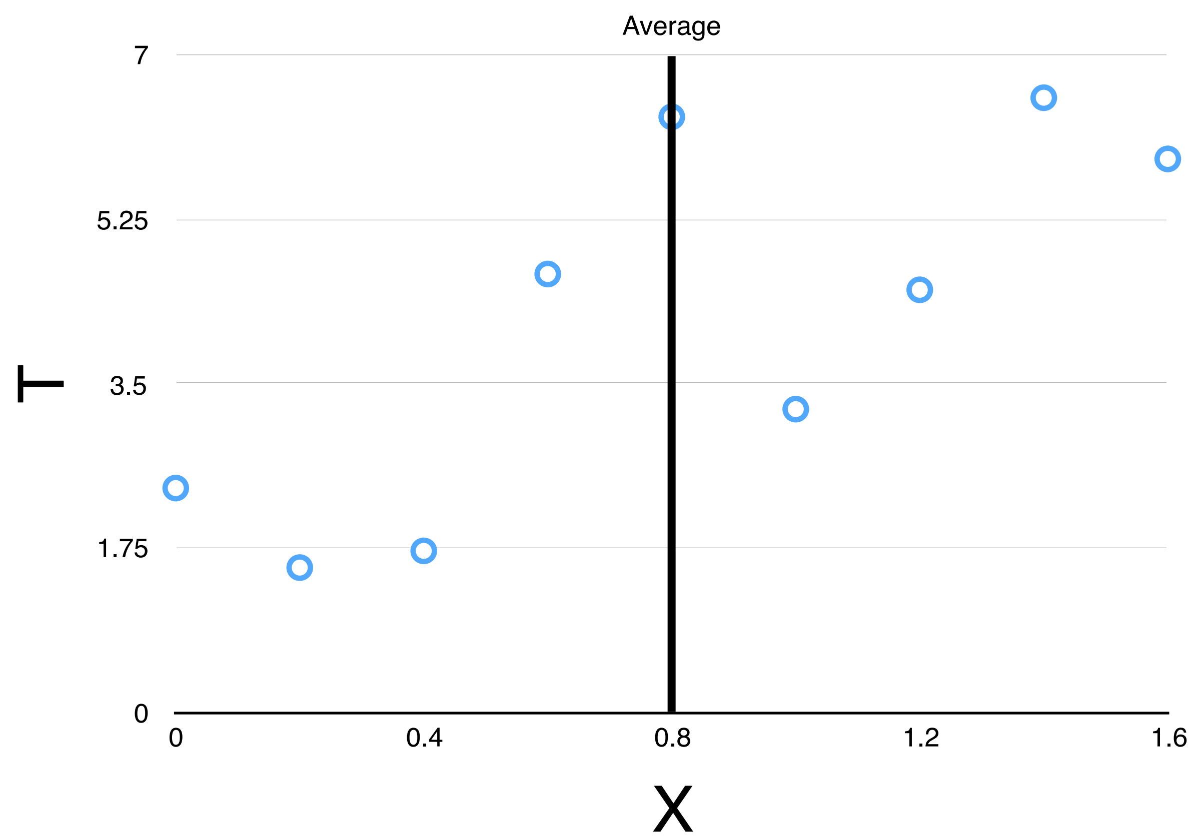
Least squares fitting





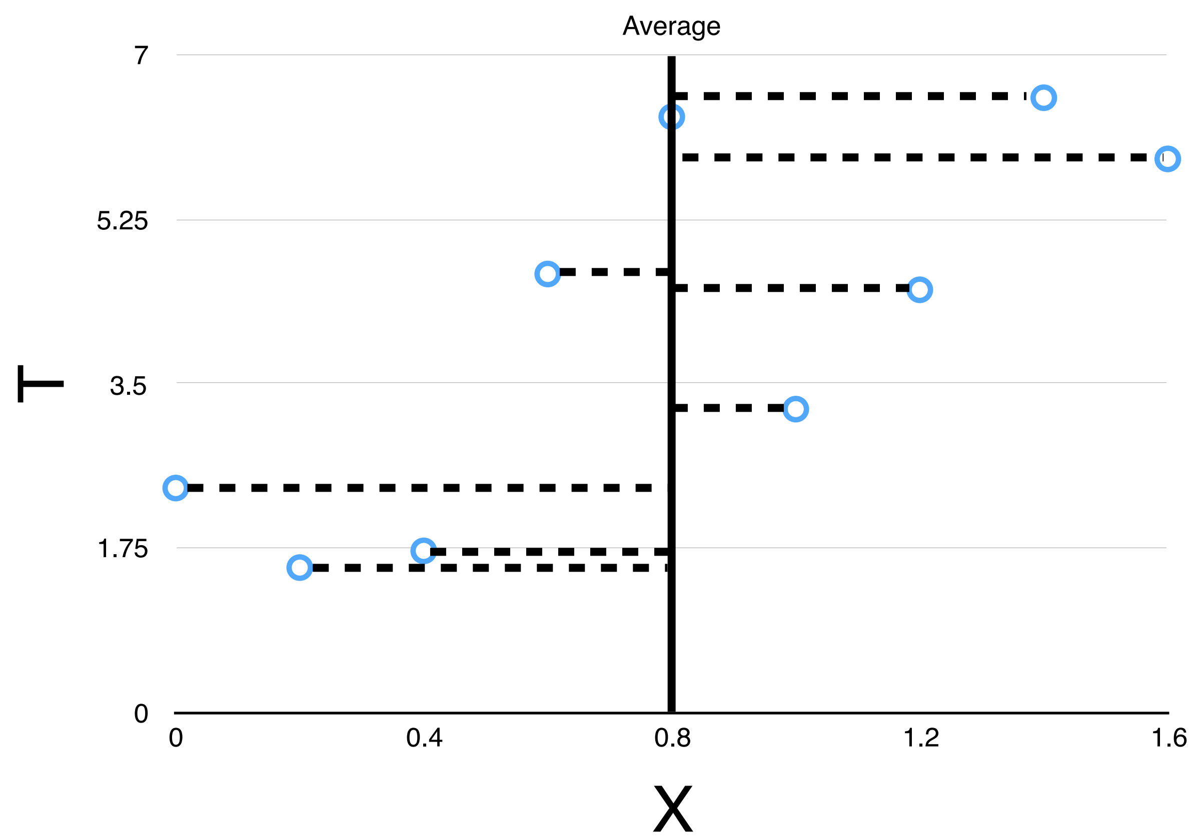
- There is also a one-shot way to calculate the solution
- Calculate how x varies
- Calculate how t varies with x
- Use these to define the solution

Average



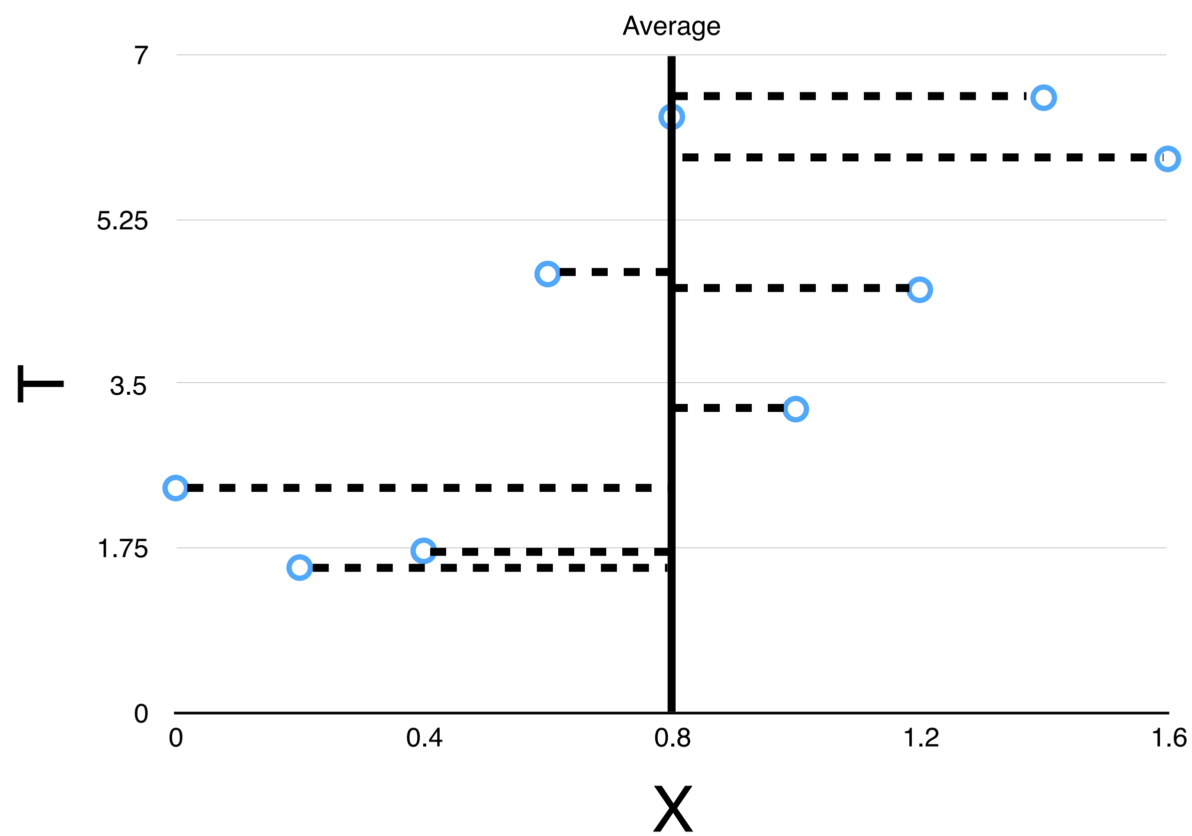
$$\bar{x} = \frac{1}{k}\sum_{i=1}^{k} x_i$$

Variance



$$\bar{x} = \frac{1}{k}\sum_{i=1}^{k} x_i$$

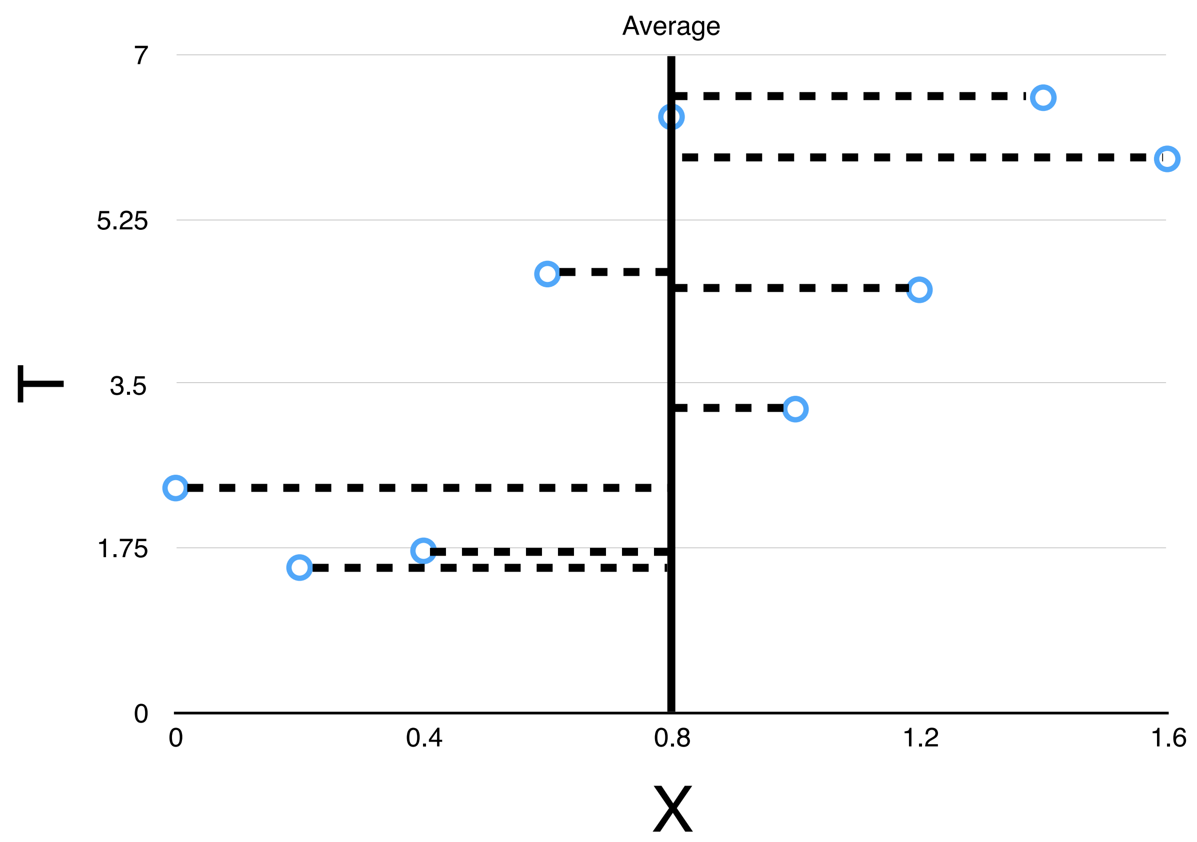
Variance



$$\mathrm{var}(X) = \frac{1}{k}\sum_{i=1}^{k}(x_i - \overline{x})^2$$

$$\overline{x} = \frac{1}{k}\sum_{i=1}^{k} x_i$$

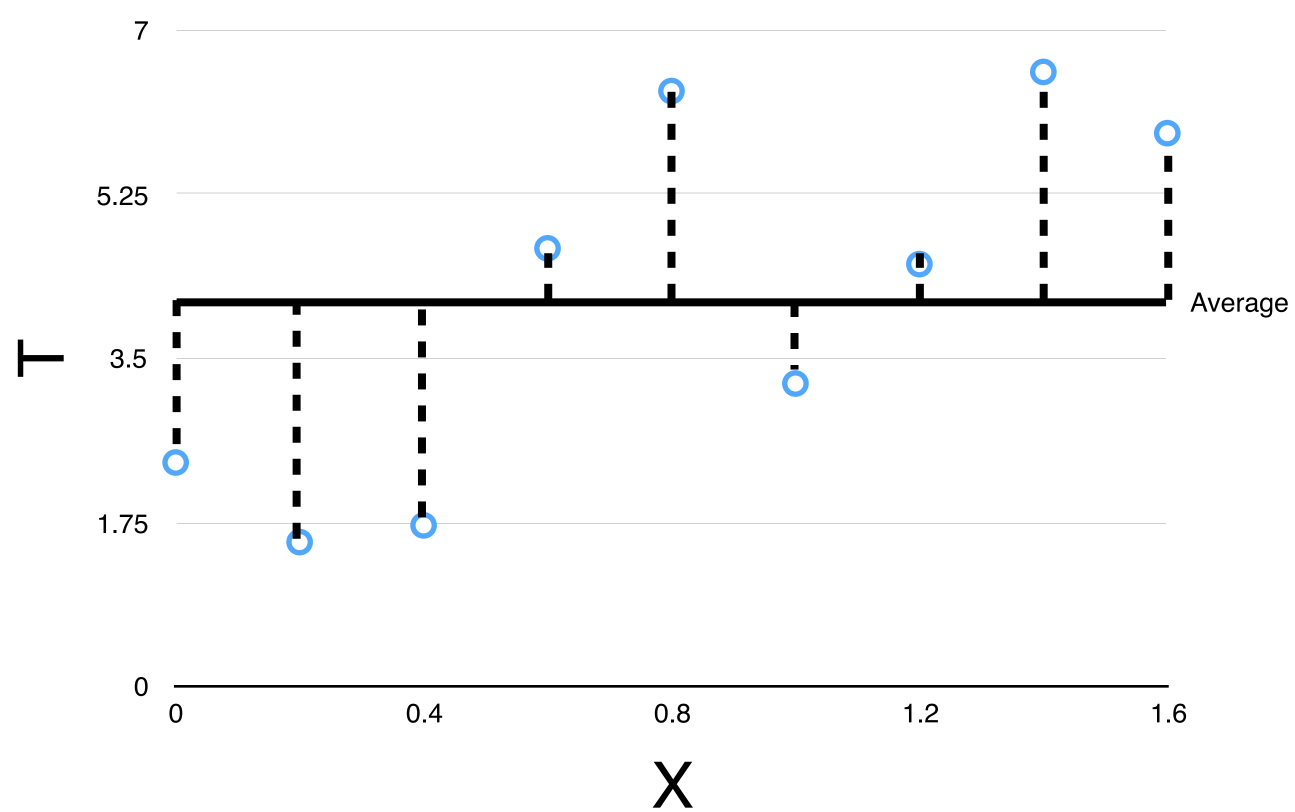
Variance



$$\mathrm{var}(X) = \frac{1}{k}\sum_{i=1}^{k}(x_i - \overline{x})^2 \qquad \overline{x} = \frac{1}{k}\sum_{i=1}^{k} x_i$$

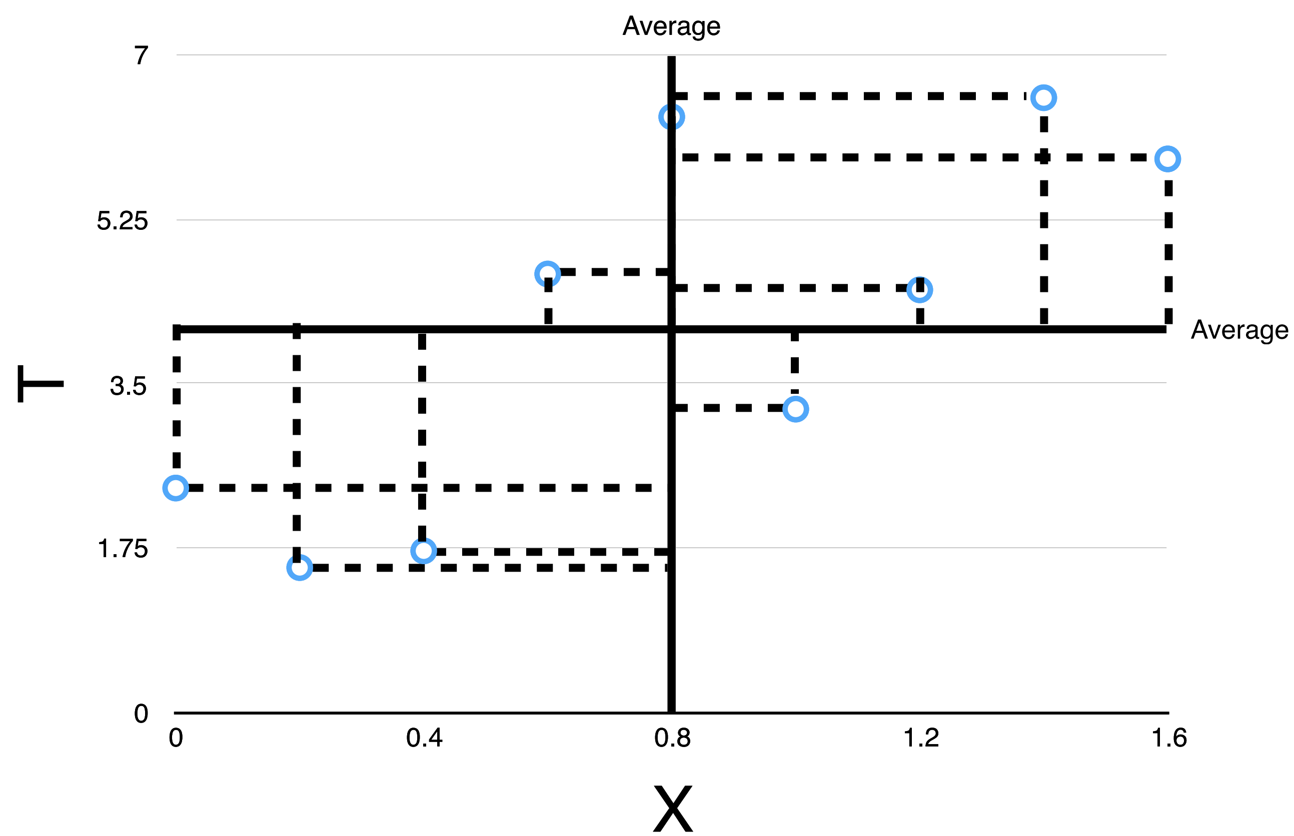
- Variance measures how much x deviates from its average
- It is the sum of the squared deviations

Variance



$$\mathrm{var}(T) = \frac{1}{k}\sum_{i=1}^{k}(t_i - \bar{t})^2 \qquad \bar{t} = \frac{1}{k}\sum_{i=1}^{k} t_i$$

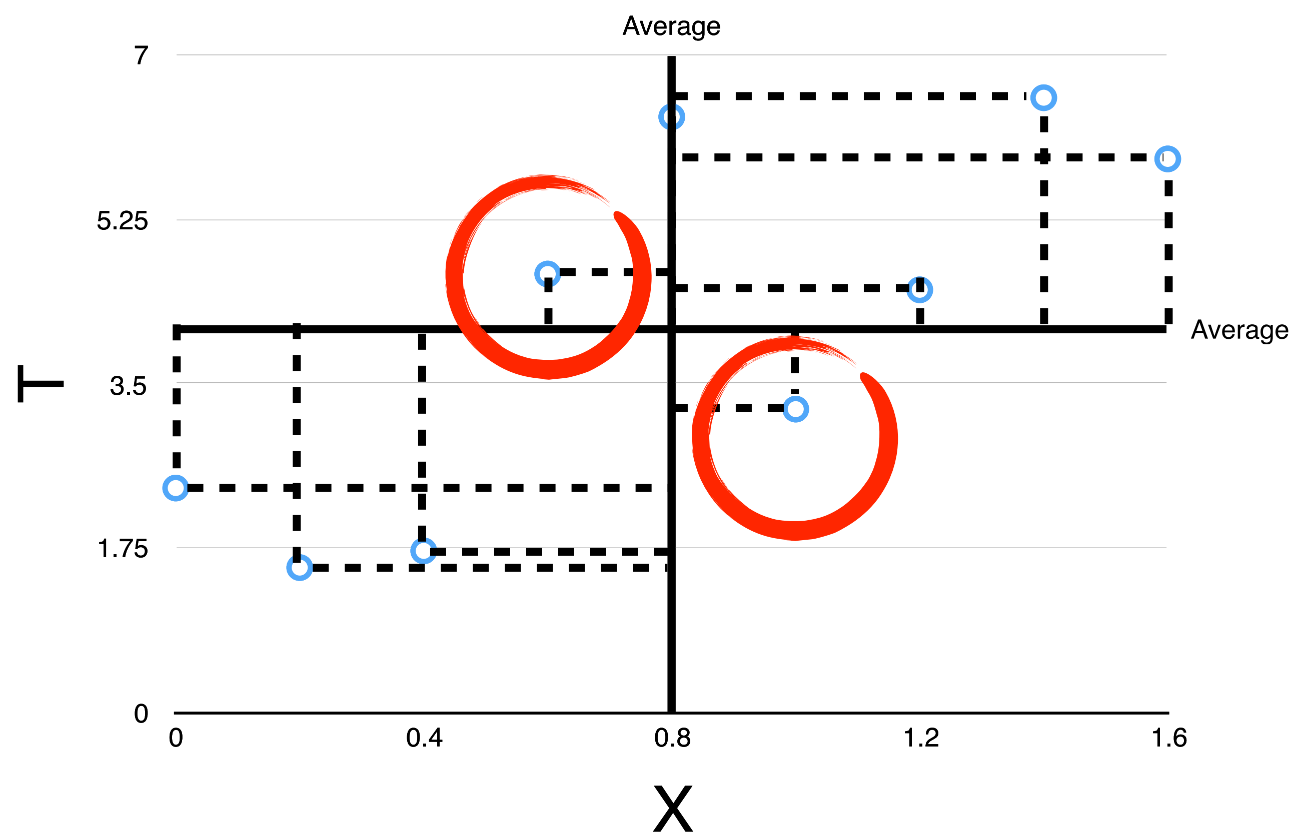
Covariance



$$\mathrm{cov}(X,T)=\frac{1}{k}\sum_{i=1}^{k}(x_i-\overline{x})(t_i-\overline{t})$$

- Covariance measures how much x and y tend to vary in the same way

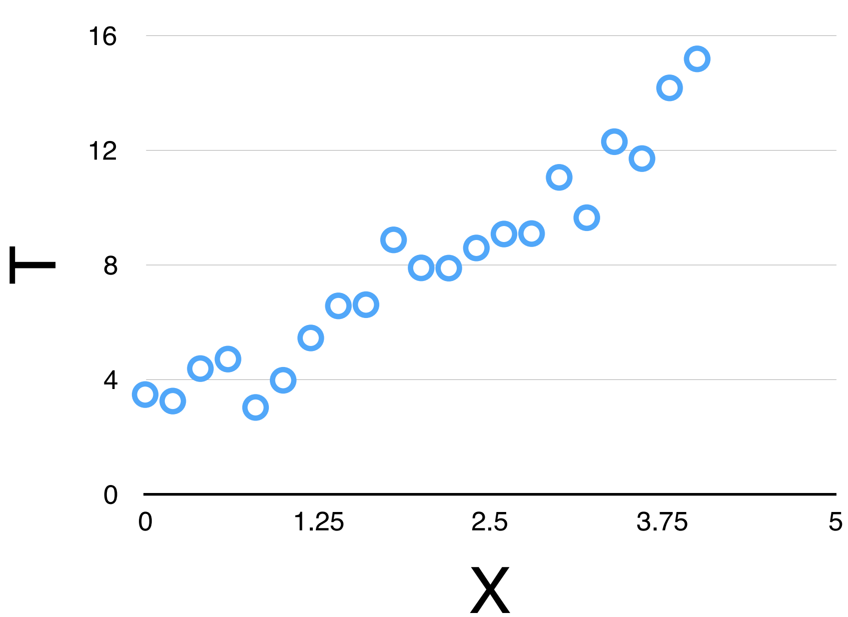
Covariance



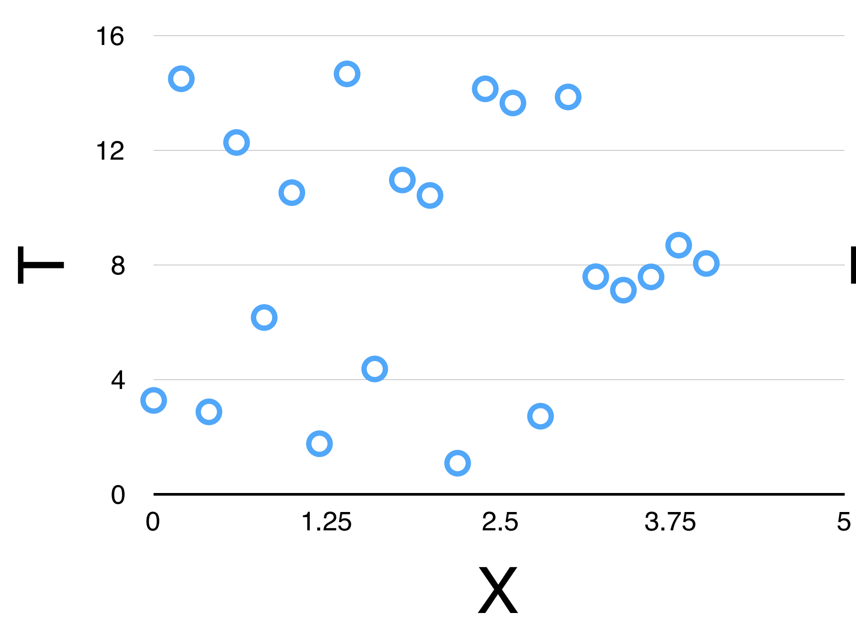
$$\mathrm{cov}(X,T) = \frac{1}{k}\sum_{i=1}^{k}(x_i - \bar{x})(t_i - \bar{t})$$

- Covariance measures how much x and y tend to vary in the same way

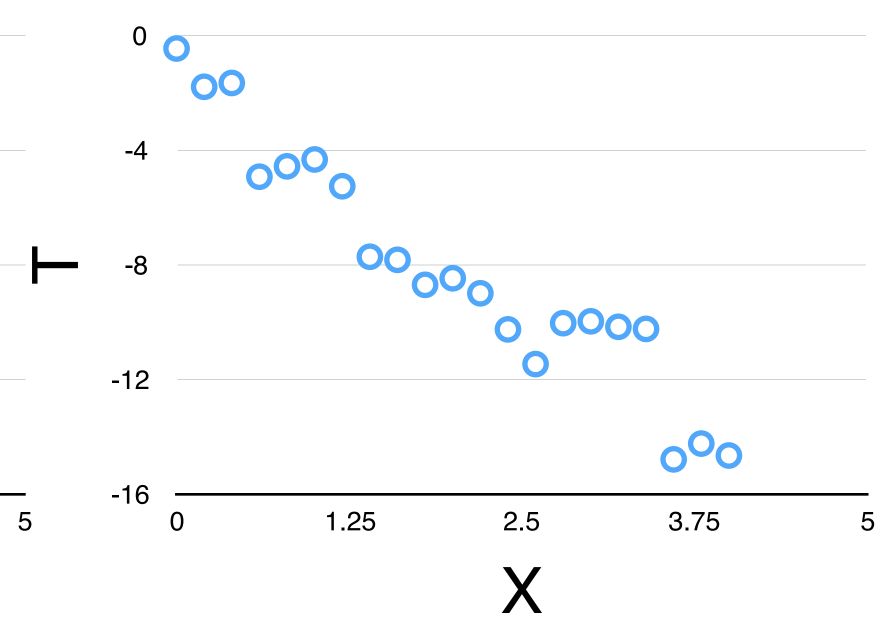
Covariance



+ve covariance

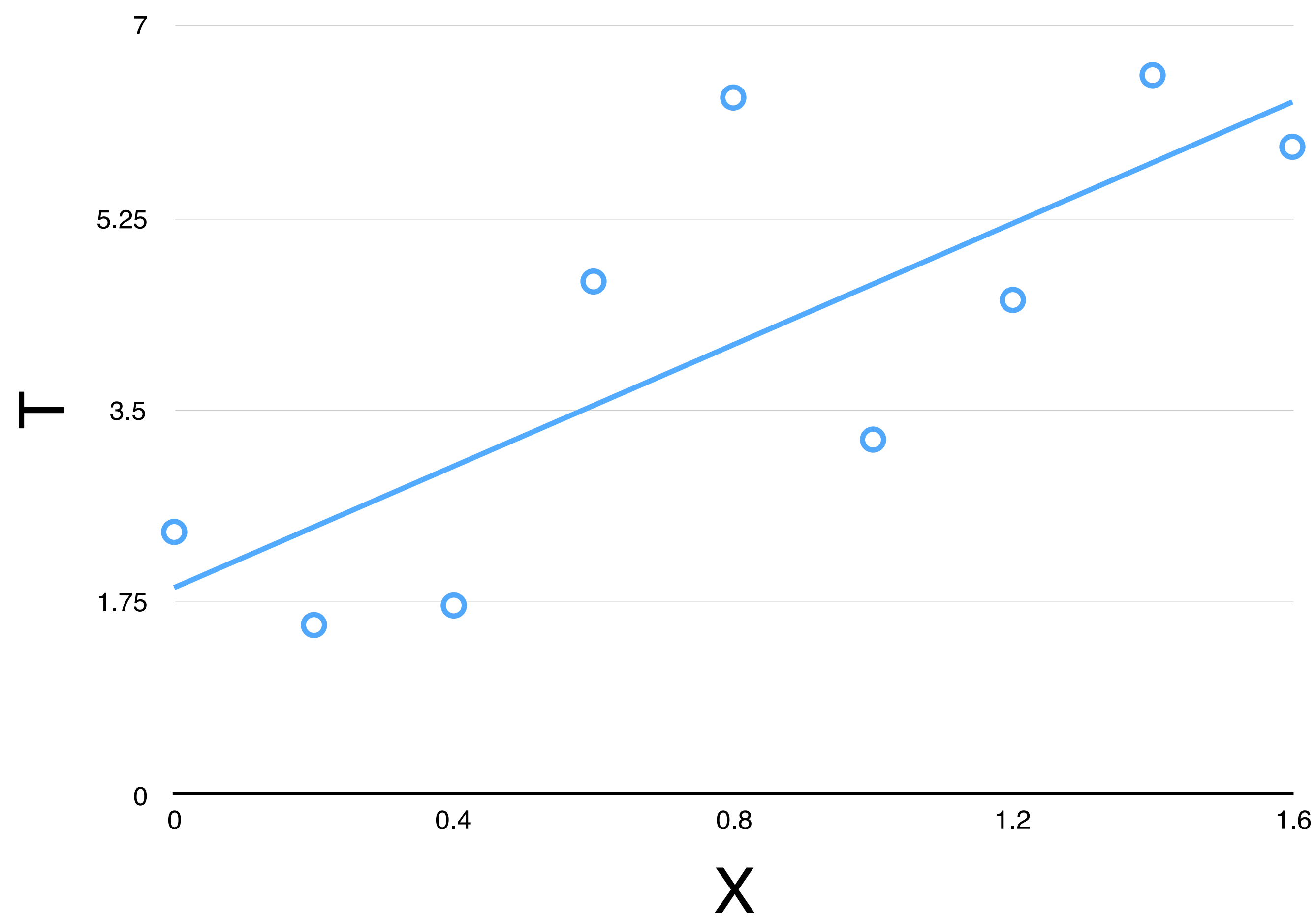


~zero covariance



-ve covariance

Variance and Covariance



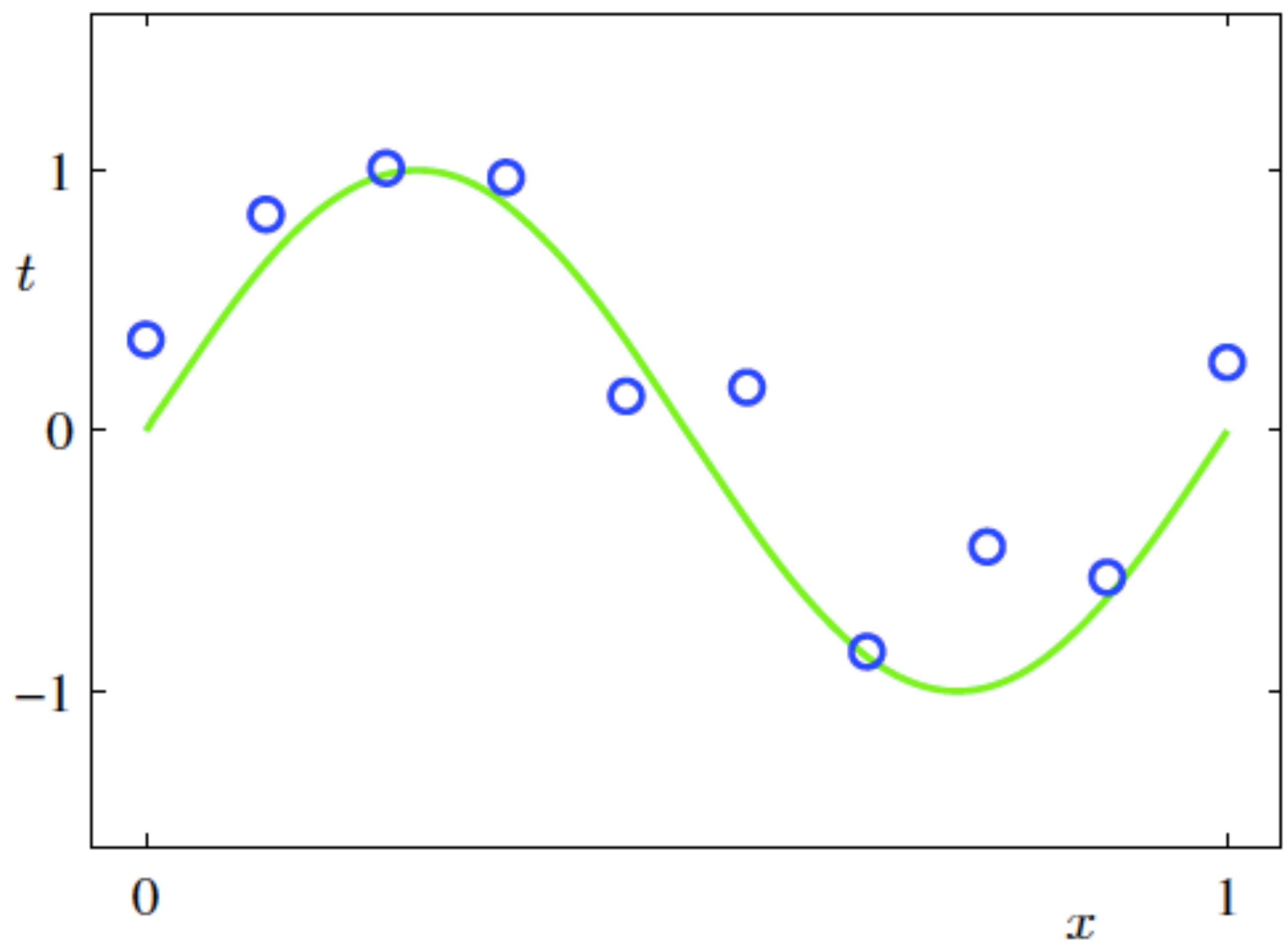
- We know that $y = ax + b$

- We can find a and b as follows $a = \dfrac{\mathrm{cov}(X,Y)}{\mathrm{var}(X)}$ $\quad b = \bar{y} - a\bar{x}$

Summary

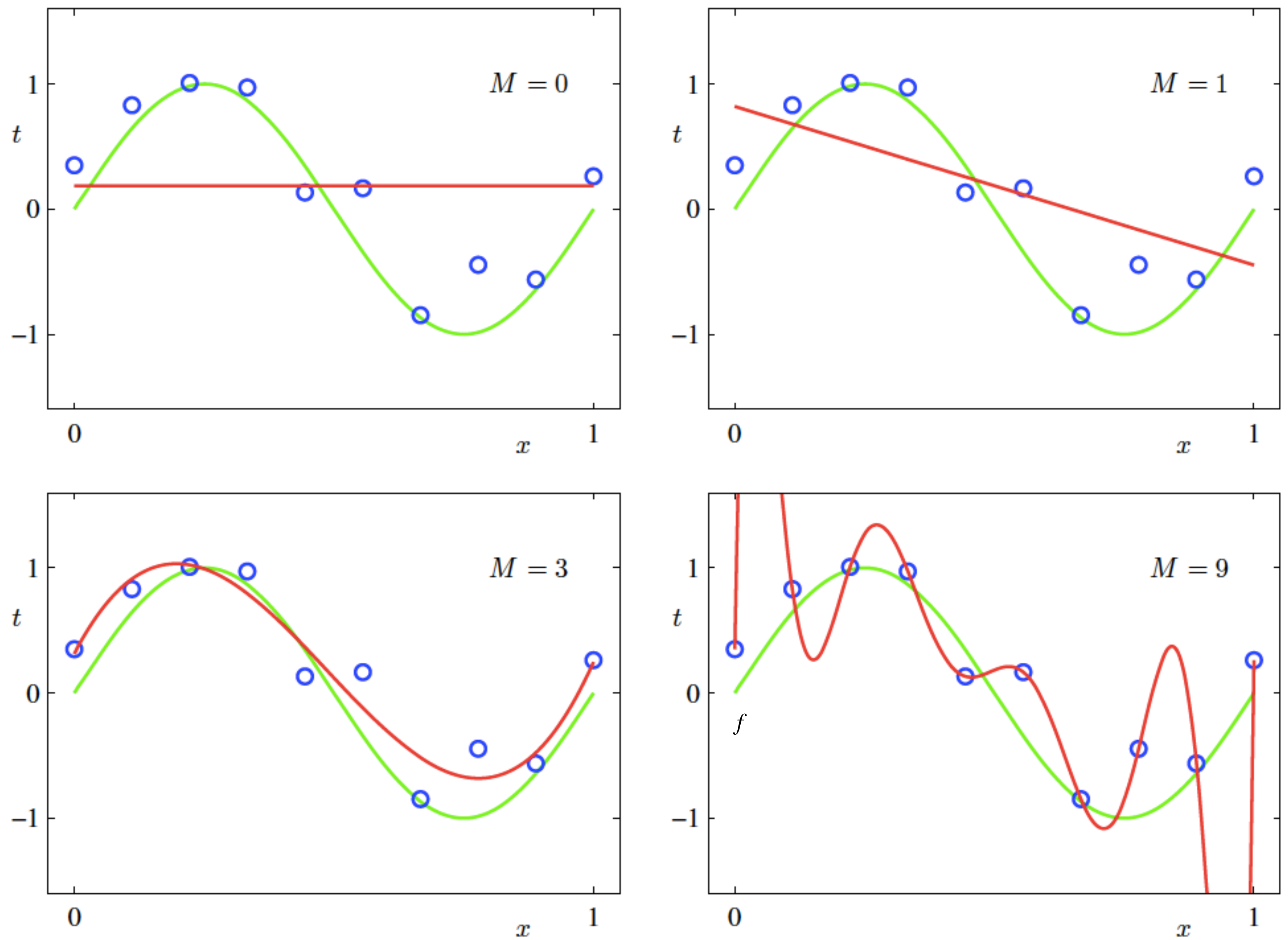
- Linear regression can be solved in several ways
- Iterative least mean squares
- Batch least mean squares
- Using covariance and variance
- Reading: Clarke and Cooke, A basic course in statistics, Chapter on Linear Regression.

Modelling Non-linear Data



- We draw noisy observations t from the green function, which is a function of x

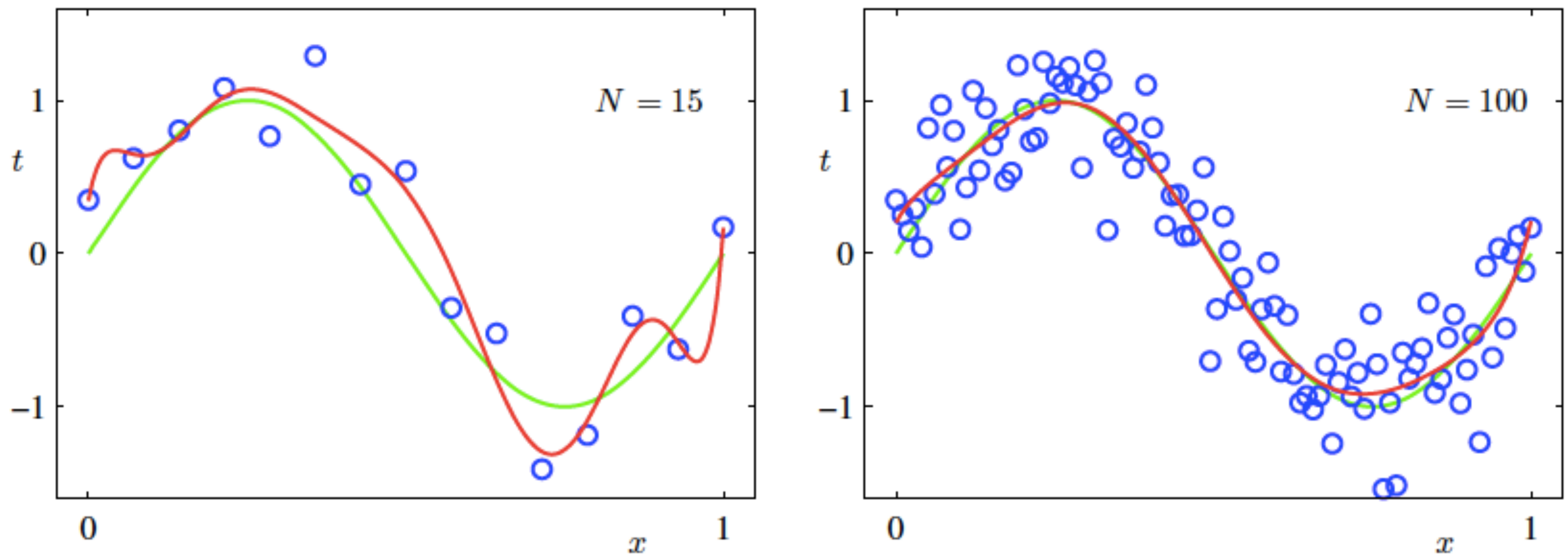
Credit: many examples in this section are taken from Bishop 2006



- We could fit polynomials of increasing order, M

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Overfitting and data



- Here, data is our friend, as it reduces overfitting as the representational power of our model rises

LMS again

- How can we decide the parameters of the polynomials?
- We can use the LMS rule

$$w'_j = w_j + \alpha(t - y)x^j$$

- where j indexes over the weights, and n over the patterns

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

- NB there is a **single** input variable x

General Linear Models for Regression

- We have seen LMS used to fit linear data by fitting a linear function of the input x

$$y(x) = ax + b \qquad y(x,\vec{w}) = \sum_{j=0}^{1} w_j x_j \qquad w'_j = w_j + \alpha(t-y)x_j$$

$$y(x) = w_1 x + w_0$$

(where $x_0 = 1$ always)

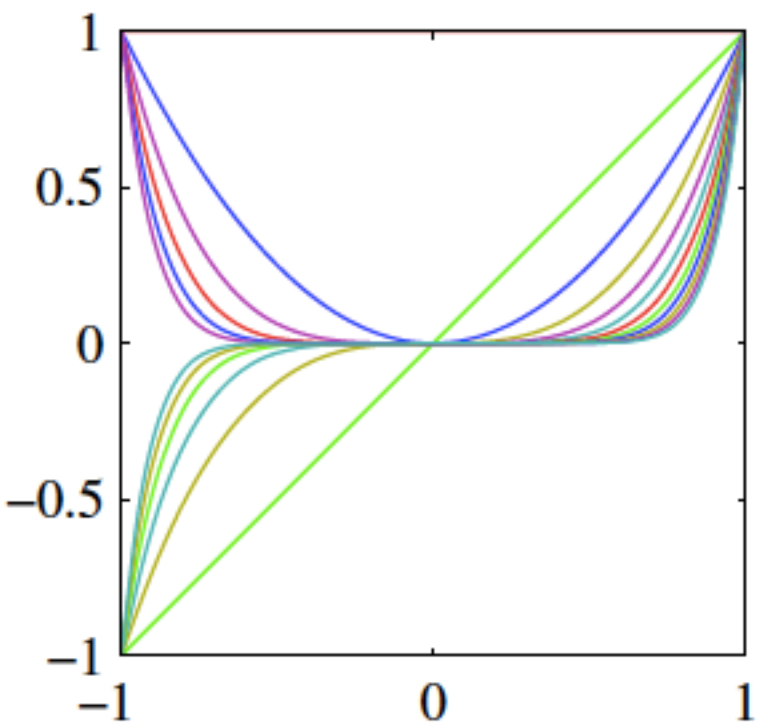
- and non-linear data by fitting a polynomial function of the input x

$$y(x,\vec{w}) = \sum_{j=1}^{M} w_j x^j \qquad w'_j = w_j + \alpha(t-y)x^j$$

- In fact we could replace x or x^j with any function of x we like

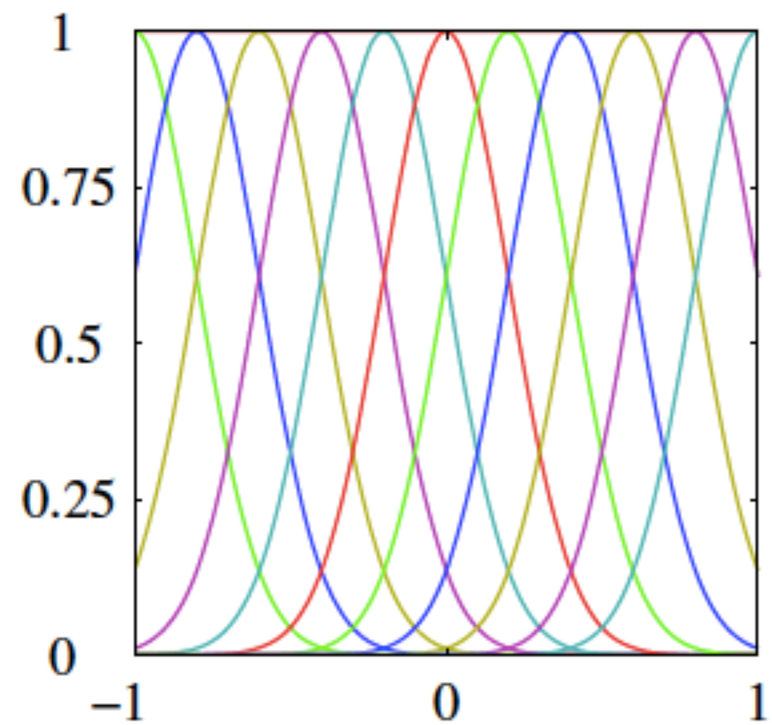
General linear models

- The general model form is $y(x,\vec{w}) = \sum_{j=0}^{M} w_j \phi_j(x)$
- $\phi_j(x)$ is called a basis function, it could be



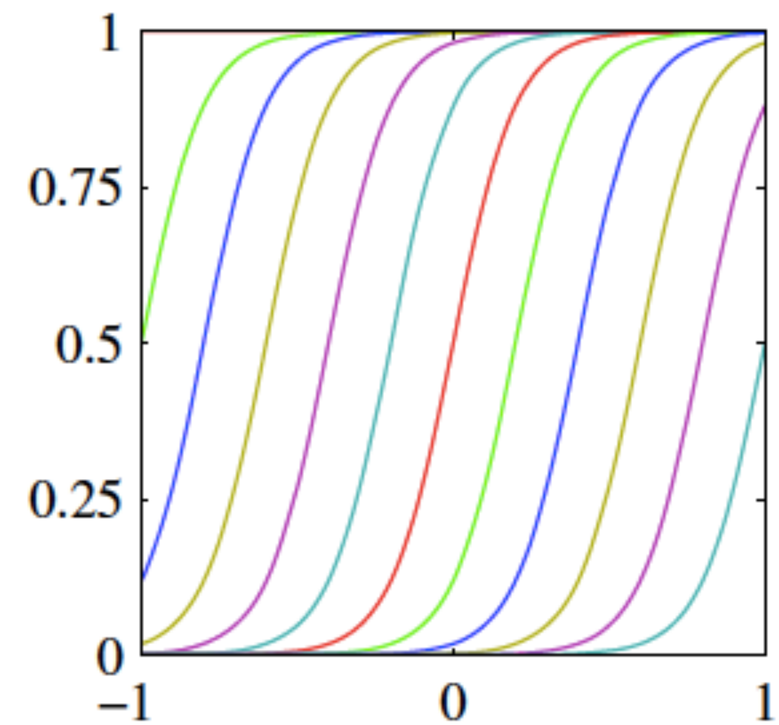
polynomial

$$\phi_j(x) = x^j$$



Gaussian

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$



sigmoidal

$$\phi_j(x) = \left(1 + \exp\left(\frac{x-\mu_j}{s}\right)\right)^{-1}$$

LMS for General Linear Models

- A general linear model is a linear combination of fixed non-linear functions of an input or inputs

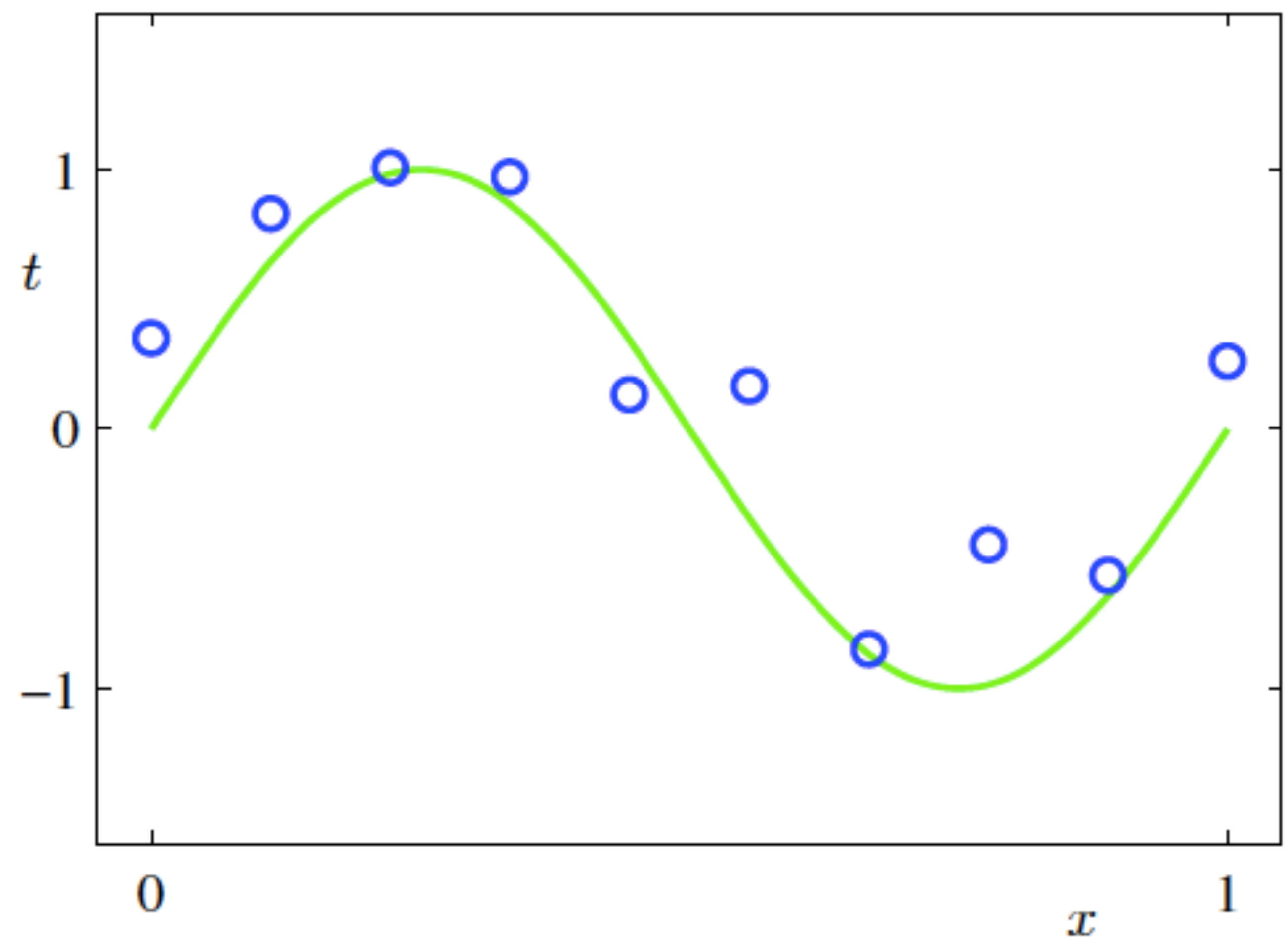
$$y(x,\vec{w}) = \sum_{j=0}^{M} w_j \phi_j(x)$$

- We can thus learn the weights of the linear combination using LMS, i.e. gradient descent

$$w'_j = w_j + \alpha(t - y)\varphi_j(x)$$

- NB Here we have just dealt with a single input scalar x and a scalar output t or y. The method here extends to multivariable inputs, i.e. $\vec{x}$. There are extensions of the method to deal with multivariate outputs, ie. $\vec{y}$

General linear models for Non-linear Data



- We can easily imagine fitting this with a small number of Gaussian functions weighted by learned coefficients

Summary

- We can turn generalise our idea of a learnable linear function of the input(s)
- We specify a learnable linear function of fixed non-linear functions of the input(s)
- We call these fixed functions basis functions
- e.g. polynomials, Gaussians
- Advanced Reading: Chris Bishop, Pattern Recognition and Machine Learning, Chapter 1, Section 1.1