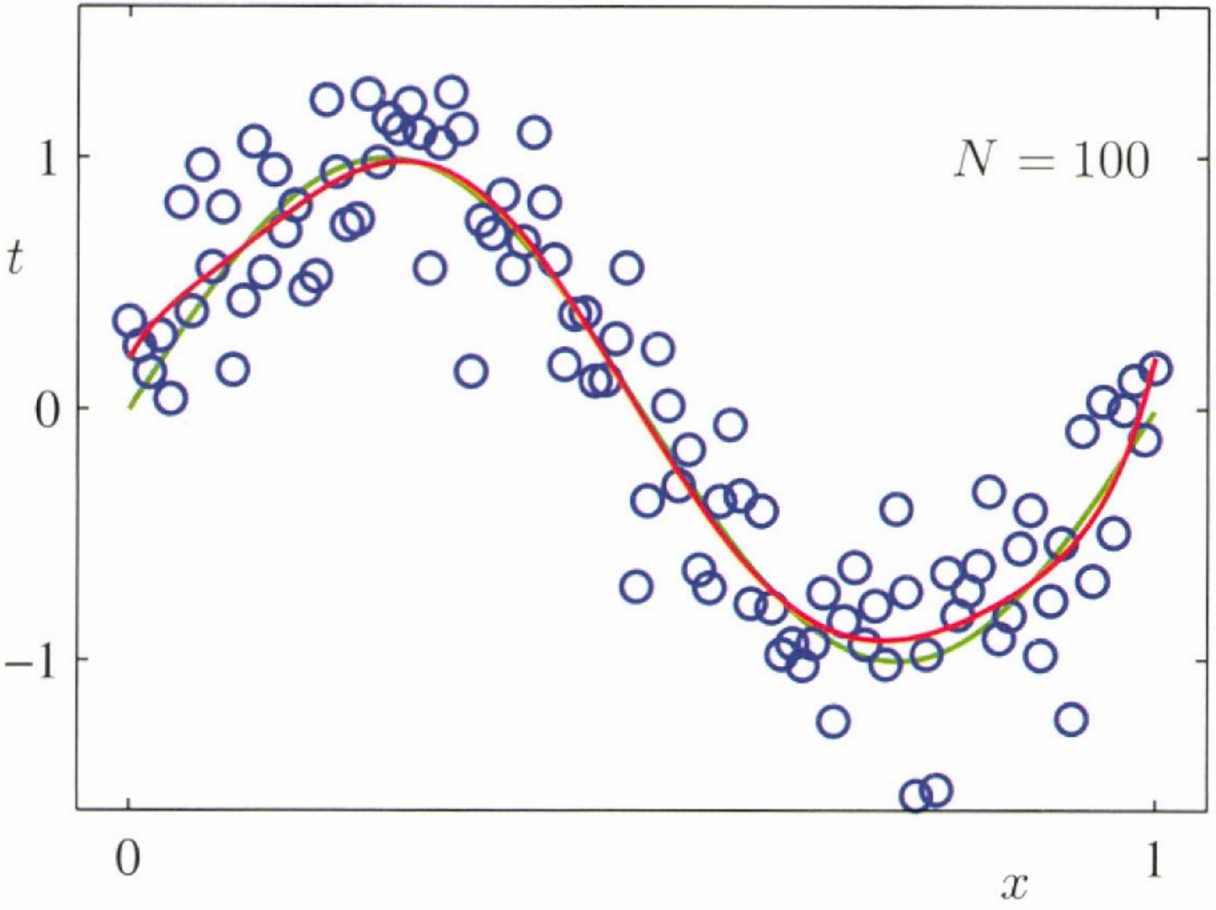
Duke 2023 ML Study

Lecture 1. Curve fitting / Regression

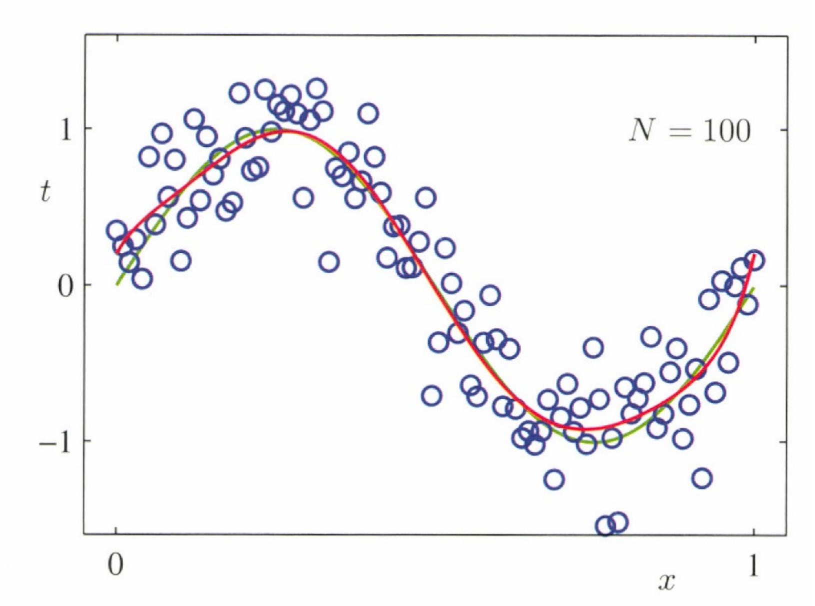
Gyeonghun Kim

1.1. Goal

Find best curve that appropriately approximate given data.



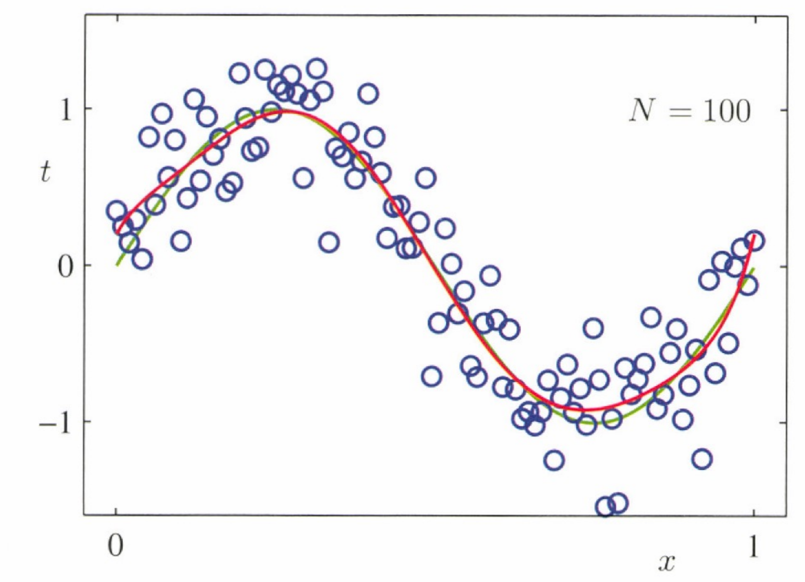
1.2. Data set, model and noise



real-world data = regularity + noise

- Intrinsic stochastic property
- Measurement noise
- Unobserved variable

1.2. Data set, model and noise

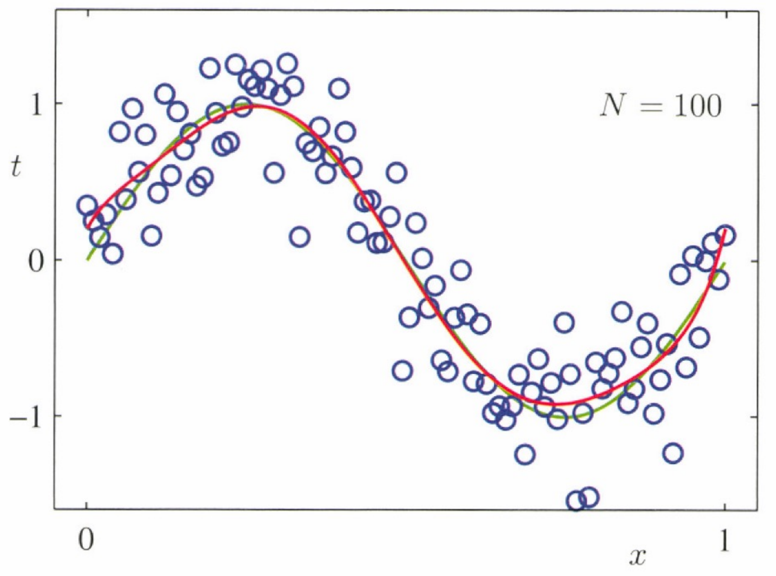


real-world data = regularity + noise

- What we want to "learn" from data
- Approximated by "model"

- Intrinsic stochastic property
- Measurement noise
- Unobserved variable

1.2. Data set, model and noise



real-world data = regularity + noise

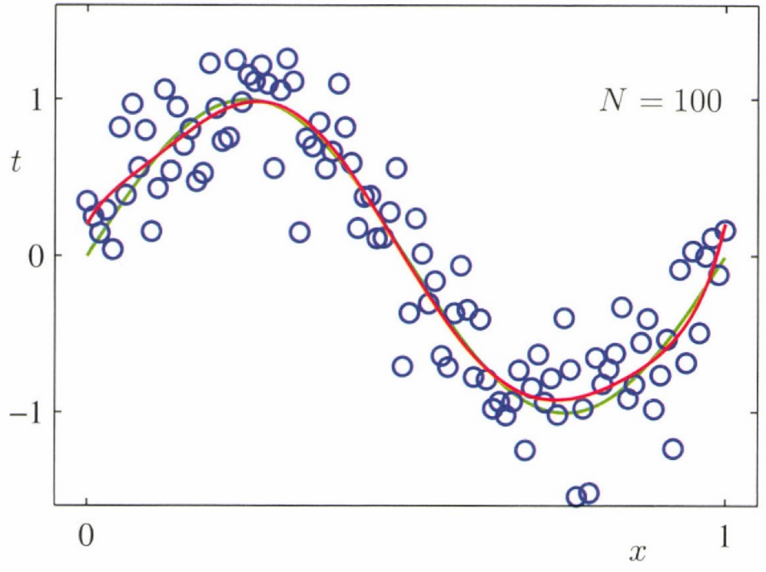
- What we want to "learn" from data
- Approximated by "model"

- Intrinsic stochastic property
- Measurement noise
- Unobserved variable

Data set = $\{(\boldsymbol{x}_n,\ t_n)\}_{n=1}^{N}$

Model = $\boldsymbol{y}(\boldsymbol{x}_n, \boldsymbol{w})$ Parameter of model

1.2. Data set, model and noise



real-world data = regularity + noise

- What we want to "learn" from data
- Approximated by "model"

- Intrinsic stochastic property
- Measurement noise
- Unobserved variable

Data set $= \{(\boldsymbol{x}_n,\ t_n)\}_{n=1}^{N}$

Model $= \boldsymbol{y}(\boldsymbol{x}_n, \boldsymbol{w})$ Parameter of model

1. Choose a best model
2. Find a best parameter for given model

2.1 Most simple method: Polynomial curve fitting

1. Choose a best model

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

2. Find a best parameter for given model

Define error function as

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

Then, find a ***w*** that minimize an error function value.

2.1 Most simple method: Polynomial curve fitting

1. Choose a best model

M+1 free parameters

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

bias

2. Find a best parameter for given model

Define error function as

This method is called "least square"

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

Then, find a ***w*** that minimize an error function value.

2.1 Most simple method: Polynomial curve fitting

1. Choose a best model

M+1 free parameters

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

bias

2. Find a best parameter for given model

Define error function as

This method is called "least square"

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

Then, find a ***w*** that minimize an error function value.

Question: How can we choose M?

2.2 Over fitting problem

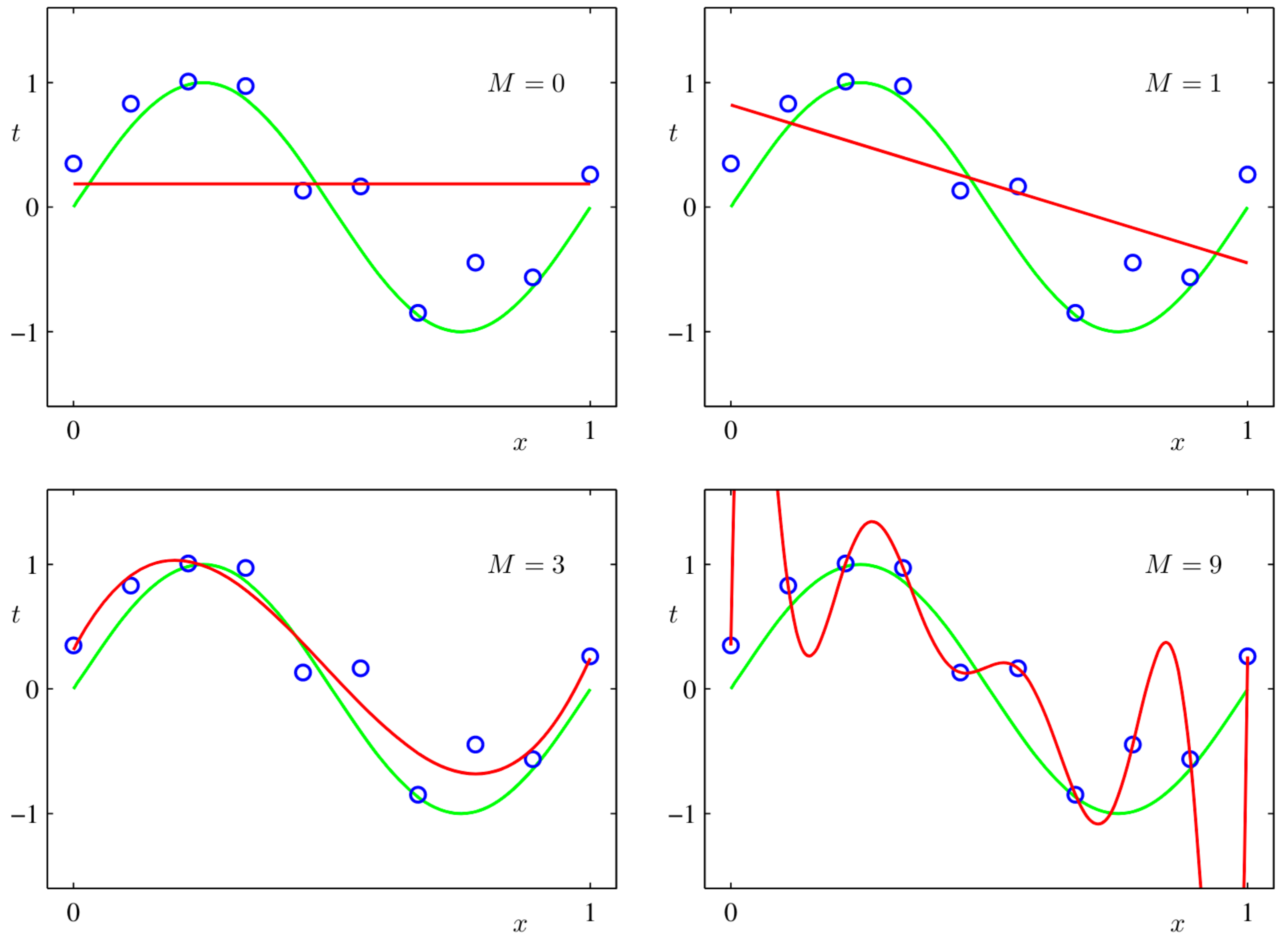


Figure 1.4 Plots of polynomials having various orders M, shown as red curves, fitted to the data set shown in Figure 1.2.

2.2 Over fitting problem

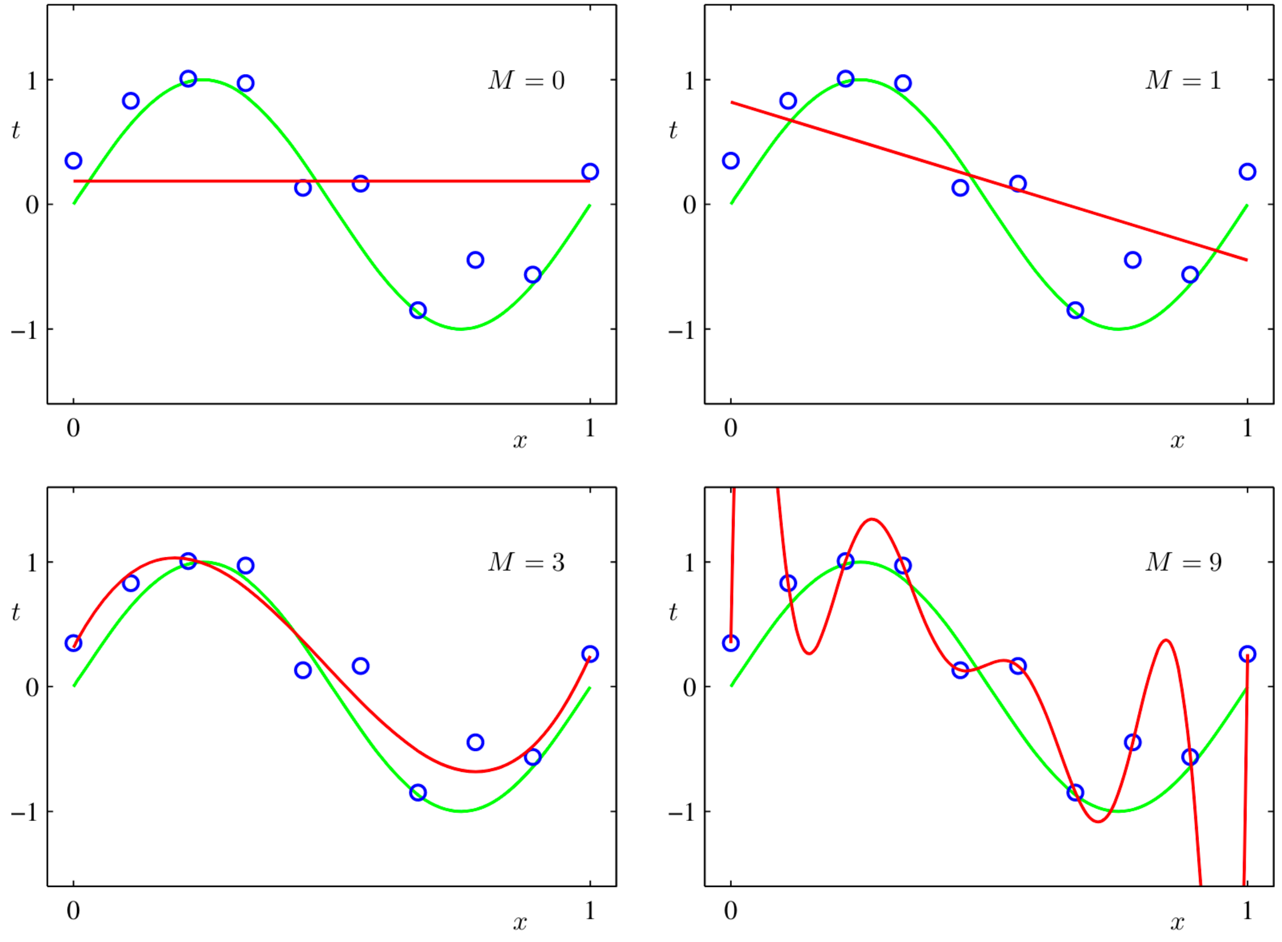
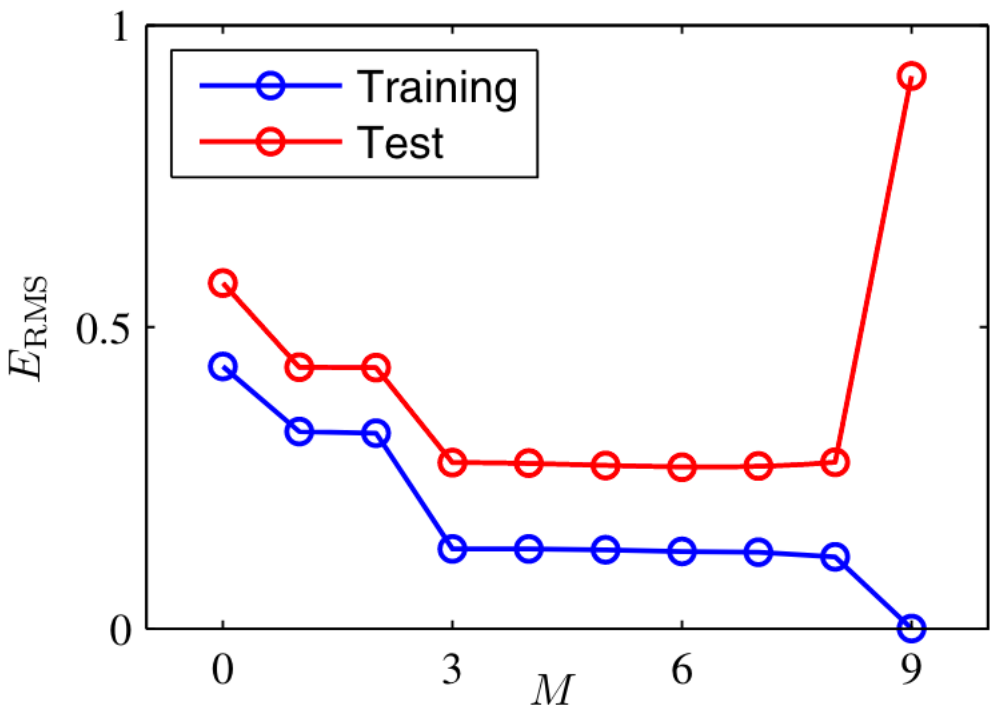


Figure 1.4 Plots of polynomials having various orders M, shown as red curves, fitted to the data set shown in Figure 1.2.

Divide Test/Training set



2.3 Regularization Method

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 \quad \longrightarrow \quad \widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \underbrace{\frac{\lambda}{2}\|\mathbf{w}\|^2}_{\text{Regularization term}}$$

2.3 Regularization Method

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 \longrightarrow \widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \underbrace{\frac{\lambda}{2}\|\mathbf{w}\|^2}_{\text{Regularization term}}$$

$\frac{\lambda}{2}\|\mathbf{w}\|^2 \longrightarrow$

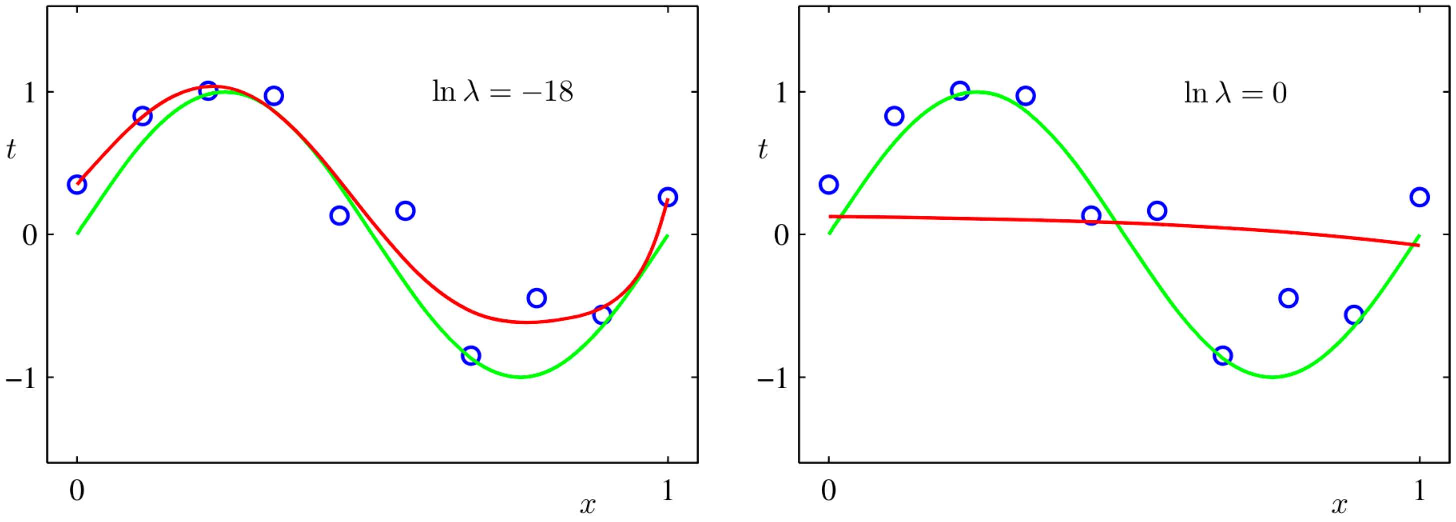
Always positive
Prefer small weight values
Have several names

- Regularization (machine learning)
- Shrinkage (statistics)
- Ridge regression
- Weight decay

2.3 Regularization Method

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 \quad \longrightarrow \quad \widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \underline{\frac{\lambda}{2}\|\mathbf{w}\|^2}$$

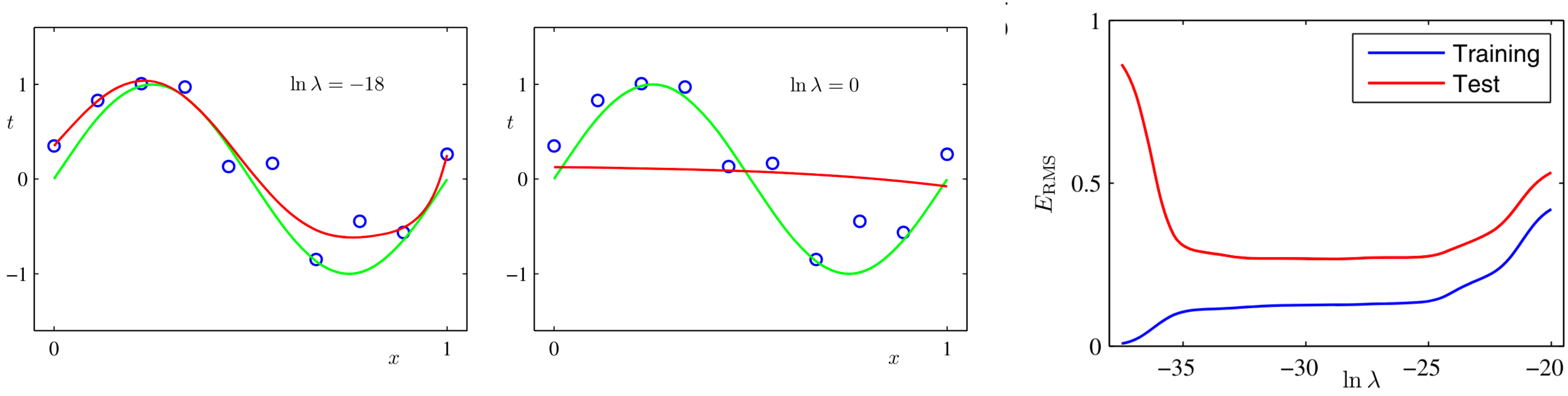
Regularization term



2.3 Regularization Method

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}) - t_n\}^2 \quad \longrightarrow \quad \widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}) - t_n\}^2 + \underline{\frac{\lambda}{2}\|\mathbf{w}\|^2}$$

Regularization term



3.1 Generalization

1. Choose a best model

Q1. Should we set our model as below polynomial?

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

2. Find a best parameter for given model

Define error function as

Q2. Should use below error function?

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

Then, find a ***w*** that minimize an error function value.

3.2 Linear Basis Function Models

Instead of using

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

We can use

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

where $\mathbf{w} = (w_0, \ldots, w_{M-1})^{\mathrm{T}}$ and $\boldsymbol{\phi} = (\phi_0, \ldots, \phi_{M-1})^{\mathrm{T}}$

3.2 Linear Basis Function Models

Instead of using

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

We can use

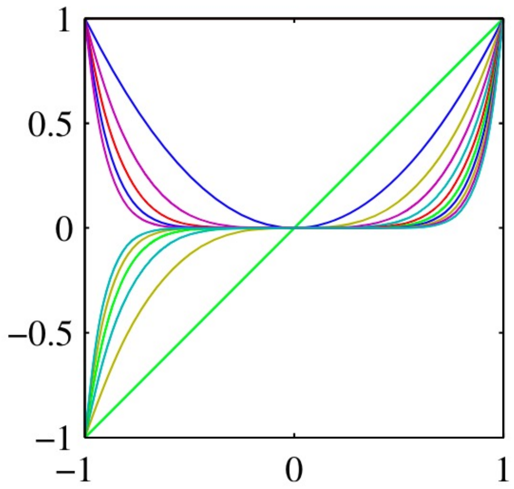
$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})$$

where $\mathbf{w} = (w_0, \ldots, w_{M-1})^{\mathrm{T}}$ and $\boldsymbol{\phi} = (\phi_0, \ldots, \phi_{M-1})^{\mathrm{T}}$

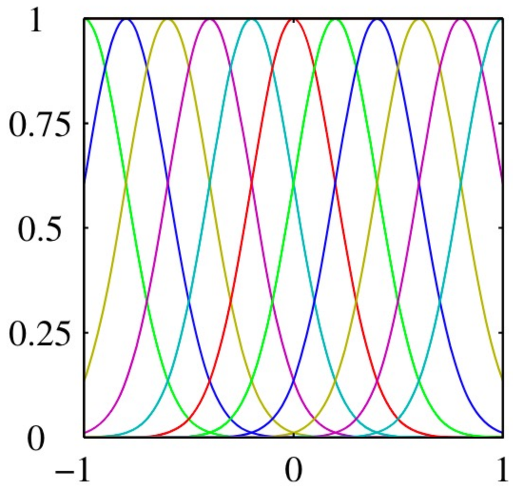
….. Why?

3.2 Linear Basis Function Models

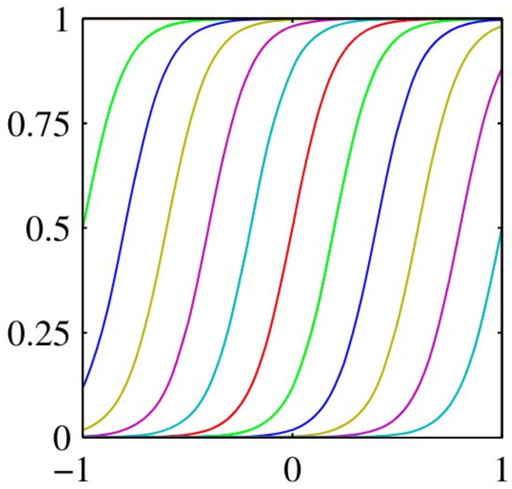
Example of possible basis sets



Polynomial



Gaussian



Sigmoidal

3.3 Finding Least Square solution

As we did for polynomial curve fitting, we can define error function as

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

3.3 Finding Least Square solution

As we did for polynomial curve fitting, we can define error function as

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

Then,

$$\nabla\, E_D(\mathbf{w}) = \sum_{n=1}^{N}\left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right\}\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} = 0$$

$$\rightarrow \quad 0 = \sum_{n=1}^{N} t_n\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} - \mathbf{w}^{\mathrm{T}}\left(\sum_{n=1}^{N}\boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}}\right)$$

3.3 Finding Least Square solution

As we did for polynomial curve fitting, we can define error function as

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

Then,

$$\nabla E_D(\mathbf{w}) = \sum_{n=1}^{N}\left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right\}\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} = 0$$

$$0 = \sum_{n=1}^{N} t_n\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} - \mathbf{w}^{\mathrm{T}}\left(\sum_{n=1}^{N}\boldsymbol{\phi}(\mathbf{x}_n)\boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}}\right)$$

$$\mathbf{w}_{\mathrm{ML}} = \underline{\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}}\mathbf{t} \quad \text{with } \boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}$$

$\boldsymbol{\Phi}^{\dagger}$: Moor-Penrose pseudo-inverse

3.4 Sequential Learning (SGD)

We can obtain a sequential learning algorithm by applying the technique of *stochastic gradient descent*, also known as *sequential gradient descent*, as follows. If the error function comprises a sum over data points $E = \sum_n E_n$, then after presentation of pattern n, the stochastic gradient descent algorithm updates the parameter vector $\mathbf{w}$ using

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n \tag{3.22}$$

where τ denotes the iteration number, and η is a learning rate parameter. We shall discuss the choice of value for η shortly. The value of $\mathbf{w}$ is initialized to some starting vector $\mathbf{w}^{(0)}$. For the case of the sum-of-squares error function (3.12), this gives

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta (t_n - \mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}_n) \boldsymbol{\phi}_n \tag{3.23}$$

where $\boldsymbol{\phi}_n = \boldsymbol{\phi}(\mathbf{x}_n)$. This is known as *least-mean-squares* or the *LMS algorithm*.

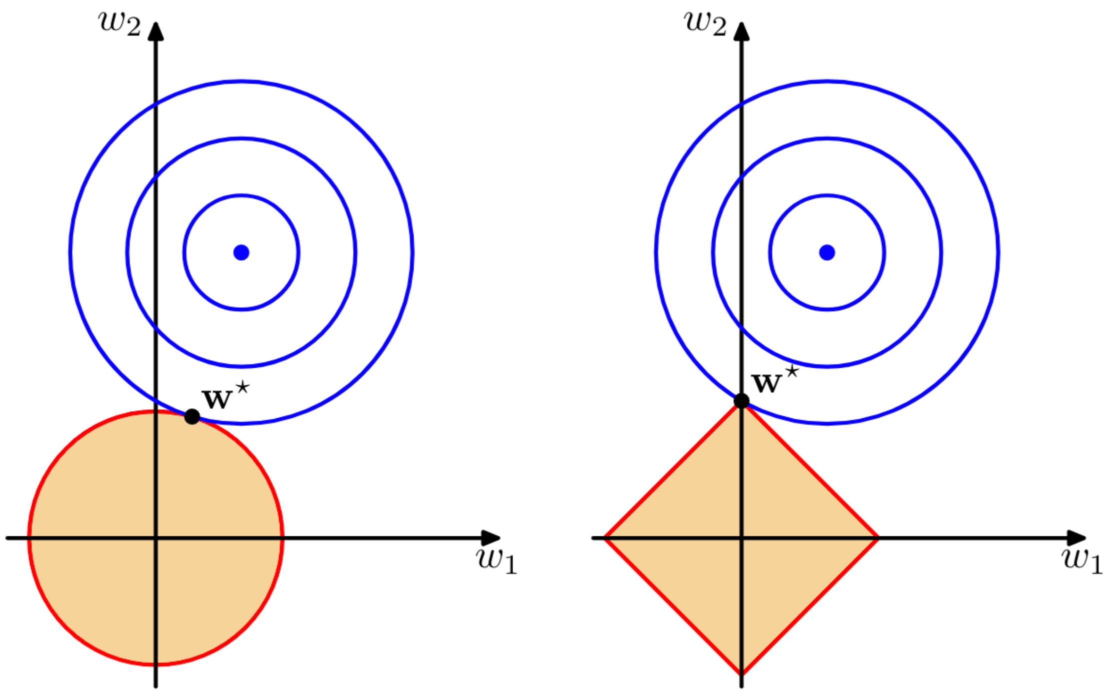
3.5 Regularization

Generalized Regularization: $$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q$$

Most famous regularization algorithms

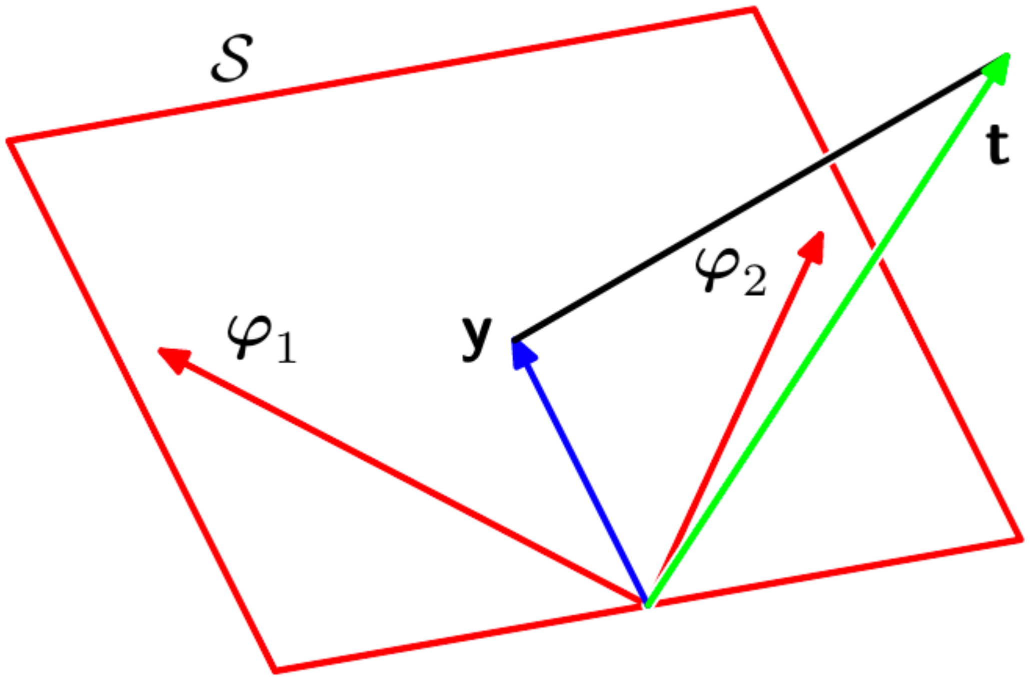
q = 1: Lasso

q = 2: Ridge



3.6 Geometrical meaning of least square

Geometrical interpretation of the least-squares solution, in an N-dimensional space whose axes are the values of $t_1, \ldots, t_N$. The least-squares regression function is obtained by finding the orthogonal projection of the data vector **t** onto the subspace spanned by the basis functions $\phi_j(\mathbf{x})$ in which each basis function is viewed as a vector $\boldsymbol{\varphi}_j$ of length N with elements $\phi_j(\mathbf{x}_n)$.



3.7 Maximum likelihood and least square

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \longrightarrow p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

3.7 Maximum likelihood and least square

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \longrightarrow p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

$$p(\mathsf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1})$$

$$\begin{aligned} \ln p(\mathsf{t}|\mathbf{w}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \\ &= \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) - \beta E_D(\mathbf{w}) \end{aligned}$$

where the sum-of-squares error function is defined by

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2.$$

4.1 Python implementation of linear regression

5. Reference

- *Christopher M. Bishop. 2006. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag, Berlin, Heidelberg.*