Regression I

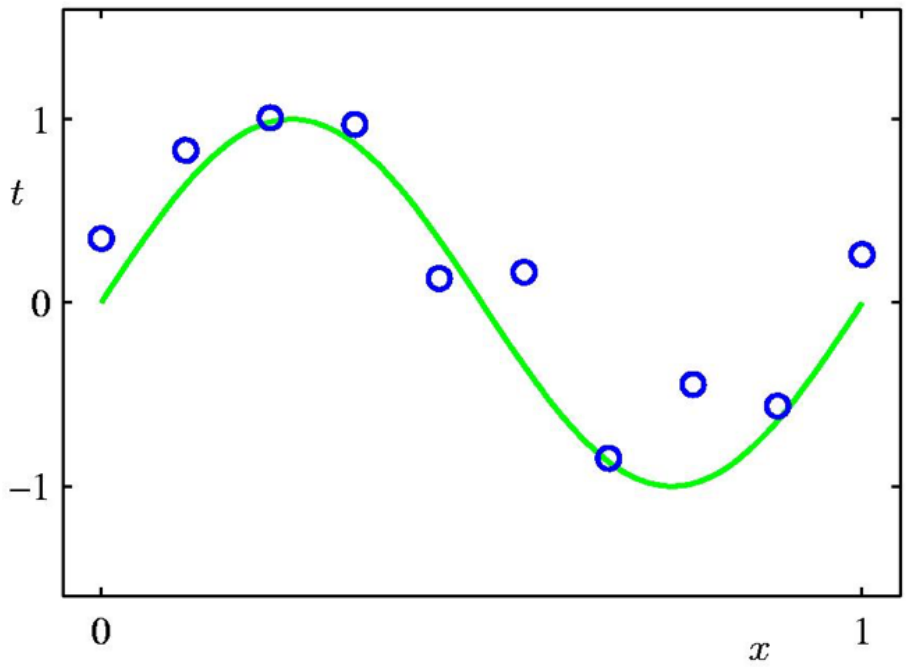
Jiayu Zhou

[1]Department of Computer Science and Engineering
Michigan State University
East Lansing, MI USA

Table of contents

Revisit the Polynomial Curve Fitting Problem

- Suppose we observe a real-valued input variable x and we wish to use this observation to predict the value of a real-valued target variable t.

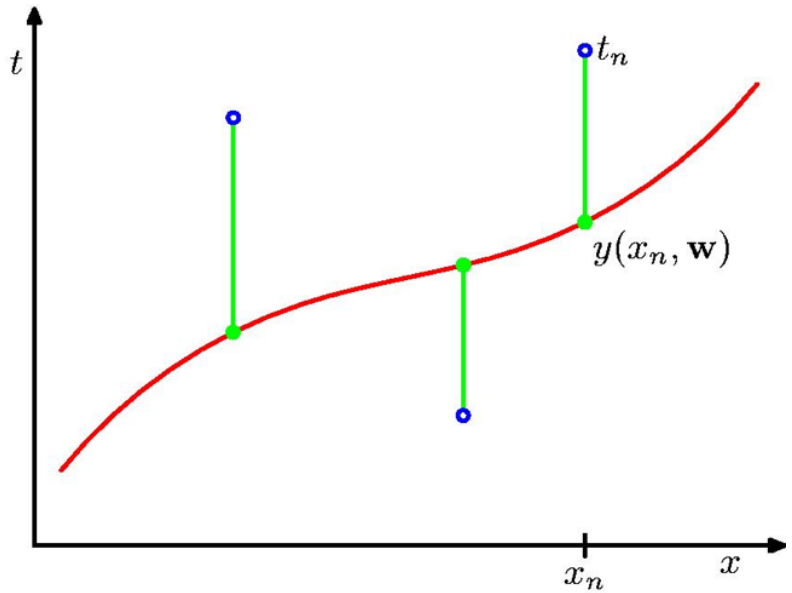


- Polynomial function

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M$$

Sum-of-Squares Error Function

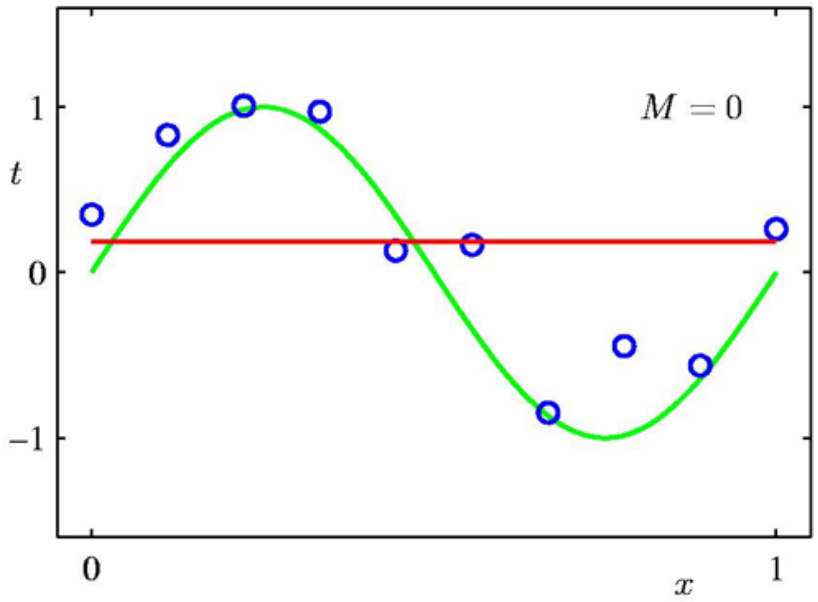
- The values of the coefficients will be determined by fitting the polynomial to the training data. This can be done by minimizing an error function that measures the misfit between the function $y(x, \mathbf{w})$, for any given value of $\mathbf{w}$, and the training set data points.



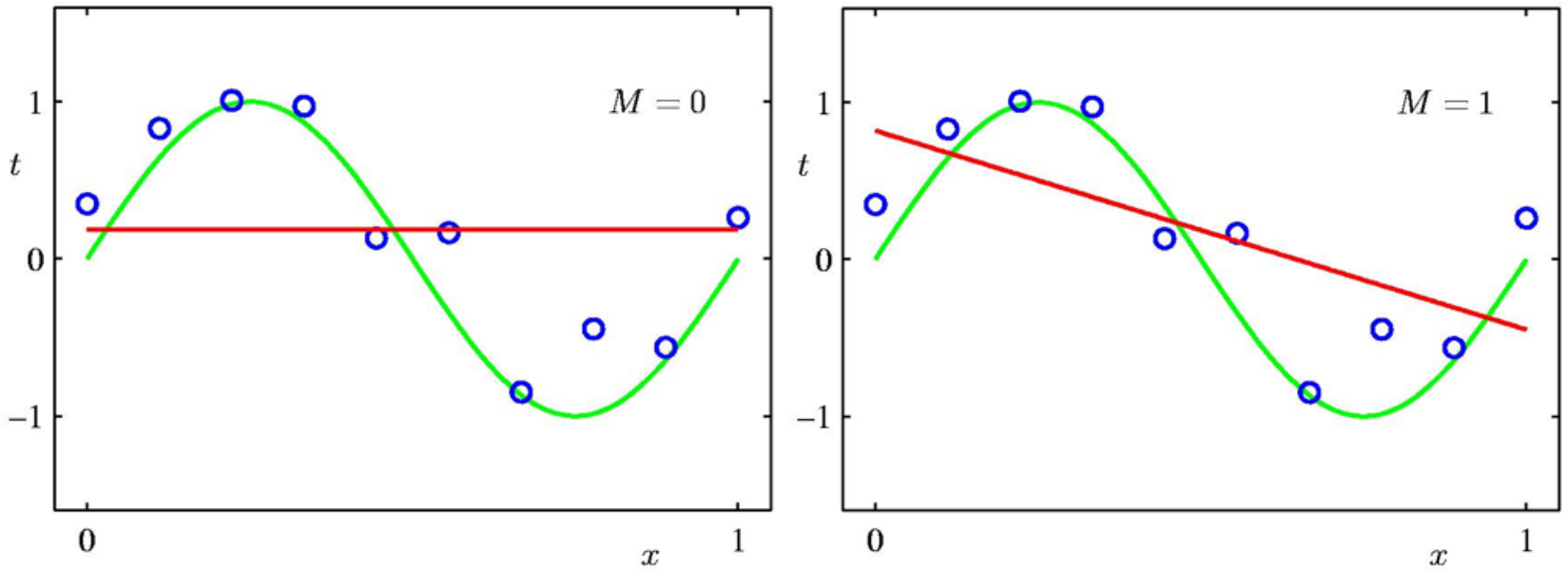
Given N observations $\{x_n, t_n\}$, the sum of the squares error function:

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

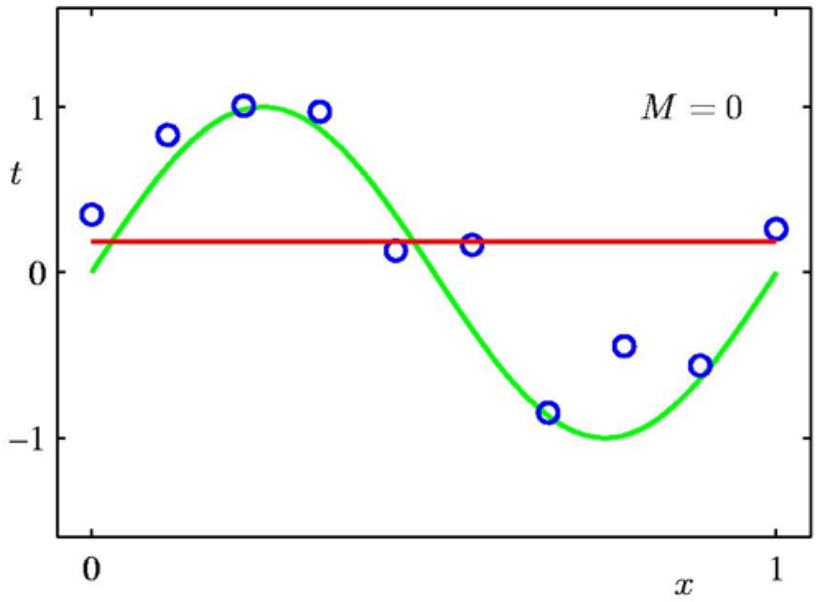
How to choose the order M

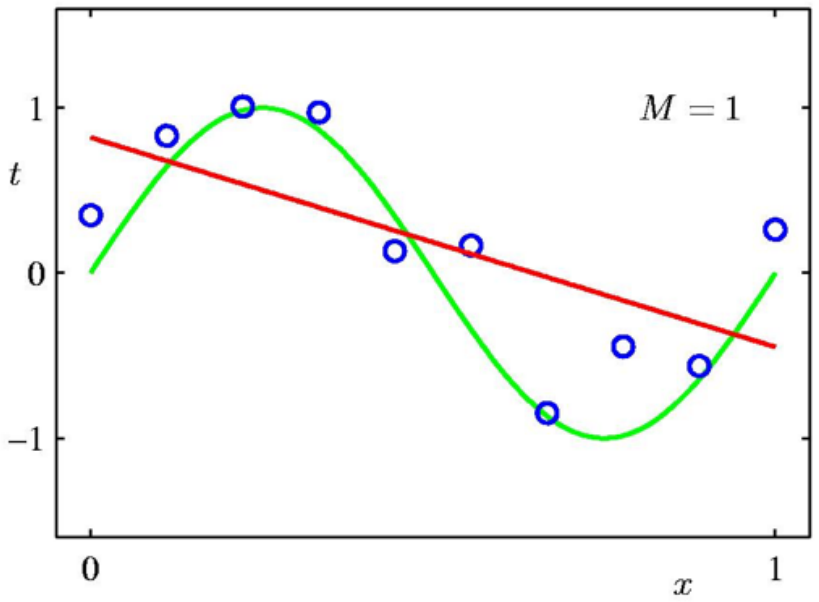


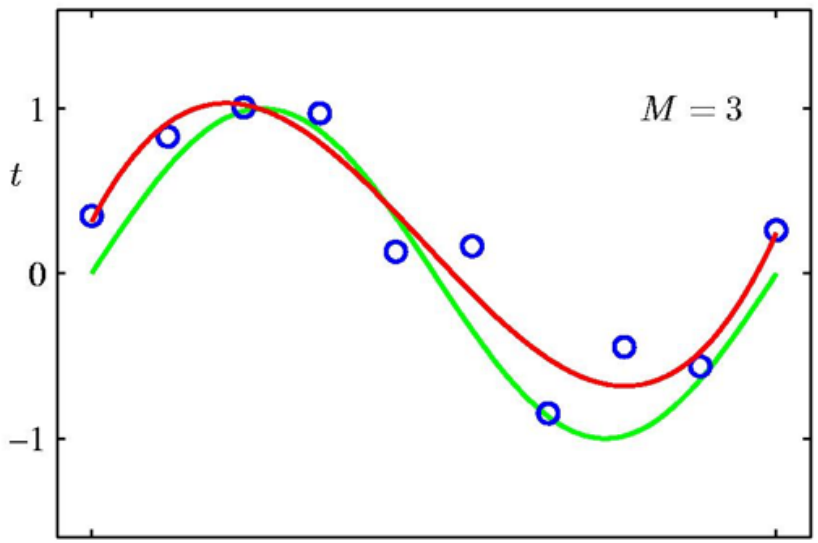
How to choose the order M



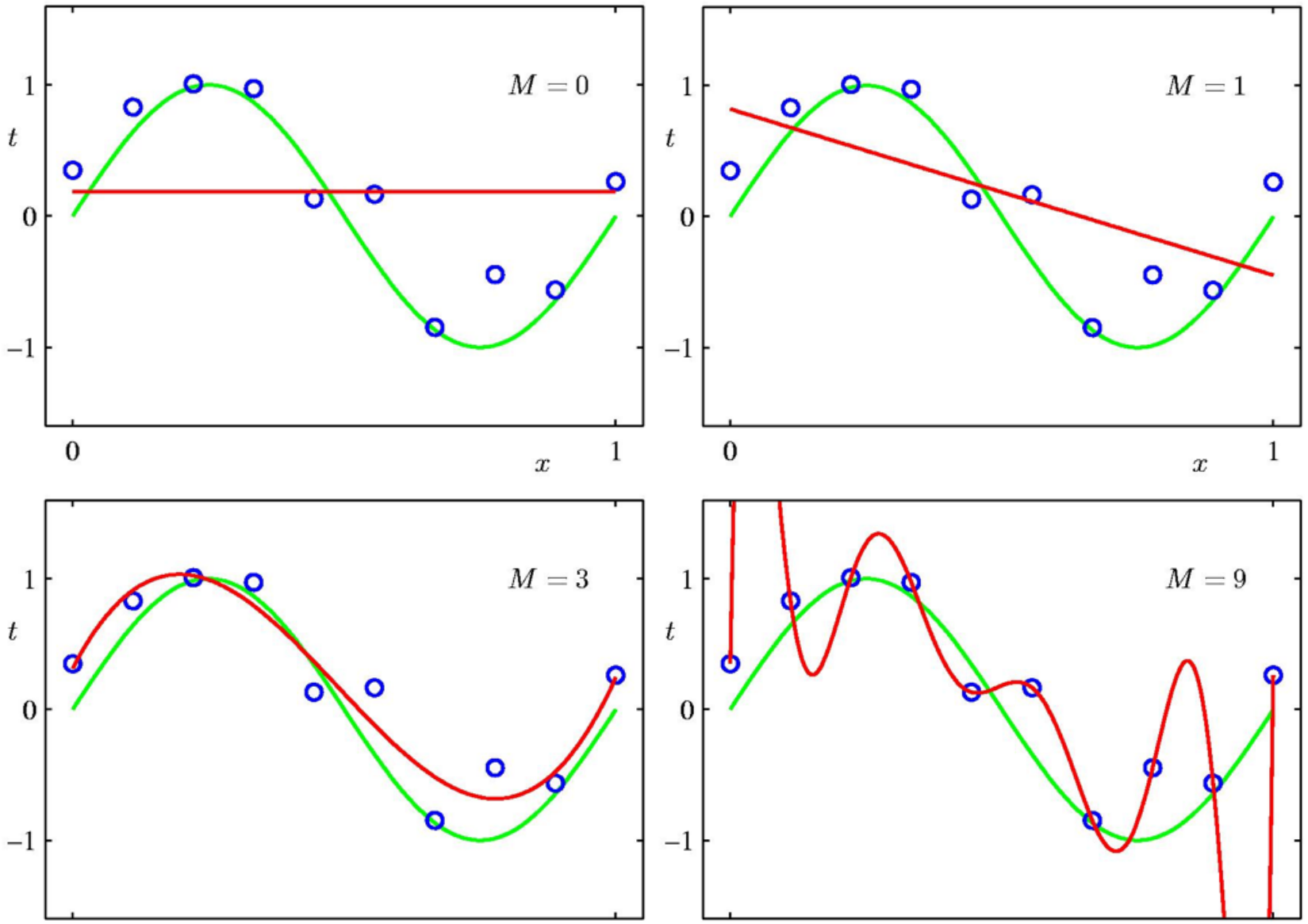
How to choose the order M







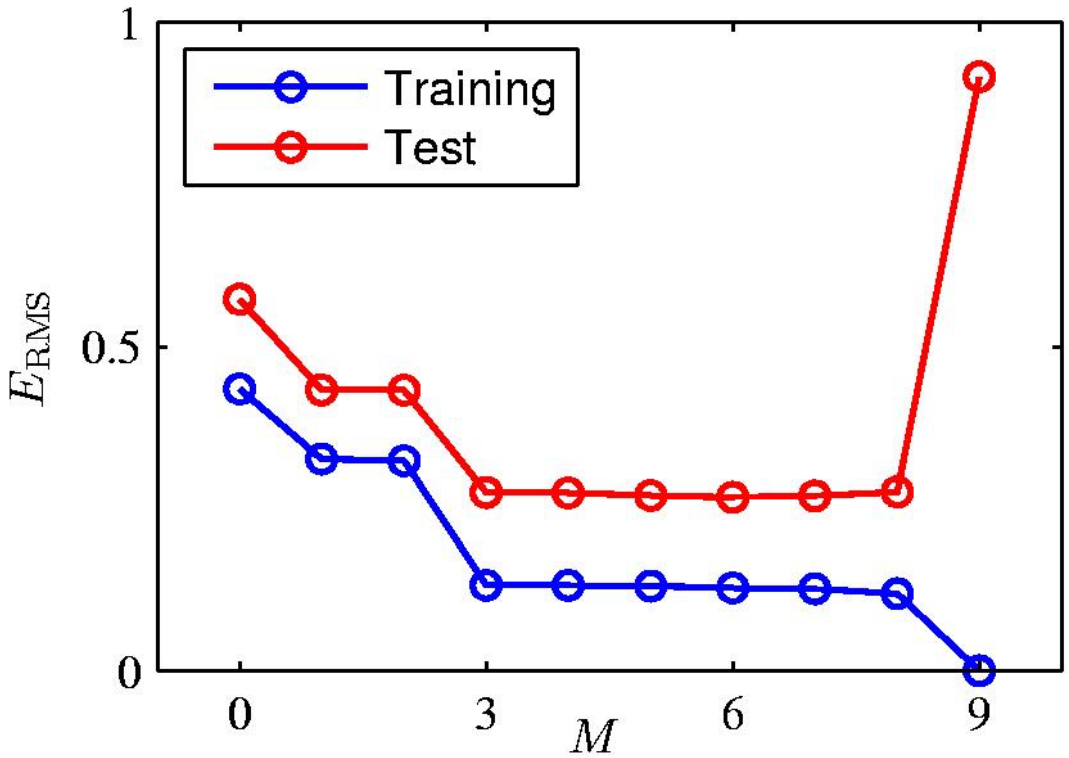
How to choose the order M



Observations

- The constant ($M = 0$) and first order ($M = 1$) polynomials give rather poor fits to the data.
- The third order ($M = 3$) polynomial seems to give the best fit to the data.
- Using a much higher order polynomial ($M = 9$), we obtain an excellent fit to the training data. However, the fitted curve oscillates wildly and gives a very poor representation. This leads to over-fitting.

Over-Fitting



Root-Mean-Square (RMS) Error: $E_{RMS} = \sqrt{2E(\mathbf{w}^*)/N}$

Polynomial Coefficients

	$M=0$	$M=1$	$M=3$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

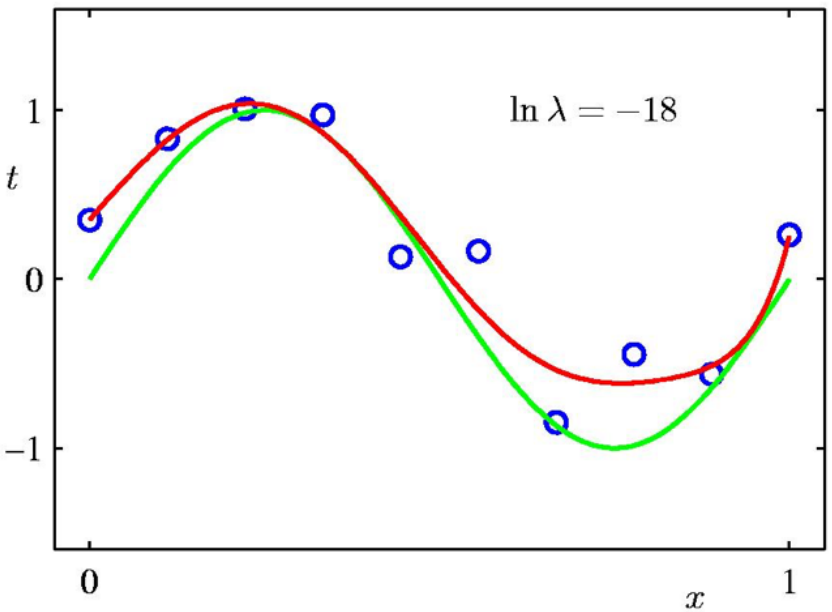
Regularization

- One technique that is often used to control the over-fitting phenomenon in such cases is that of regularization, which involves adding a penalty term to the error function in order to discourage the coefficients from reaching large values.

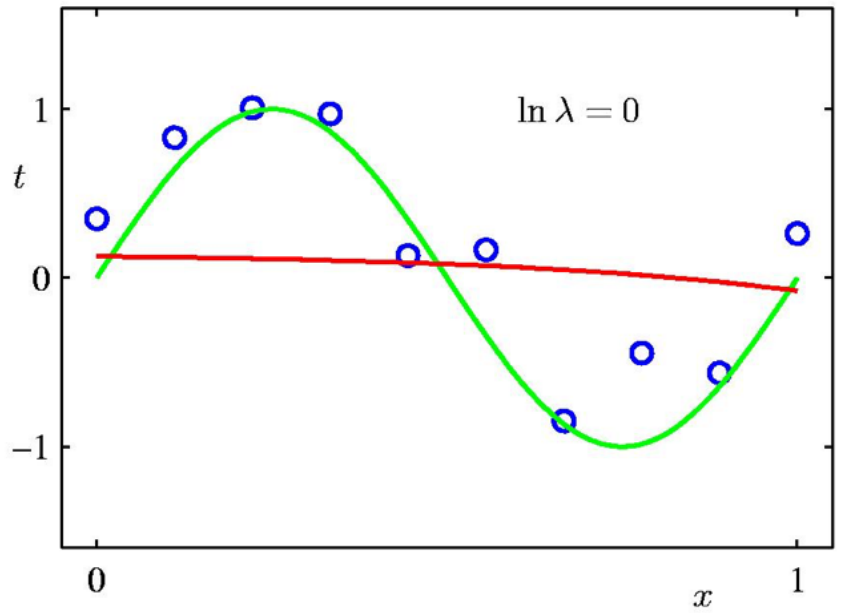
$$\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

- The coefficient λ governs the relative importance of the regularization term compared with the sum-of-squares error term.

Effect of Regularization Parameter



$\ln \lambda = -18$

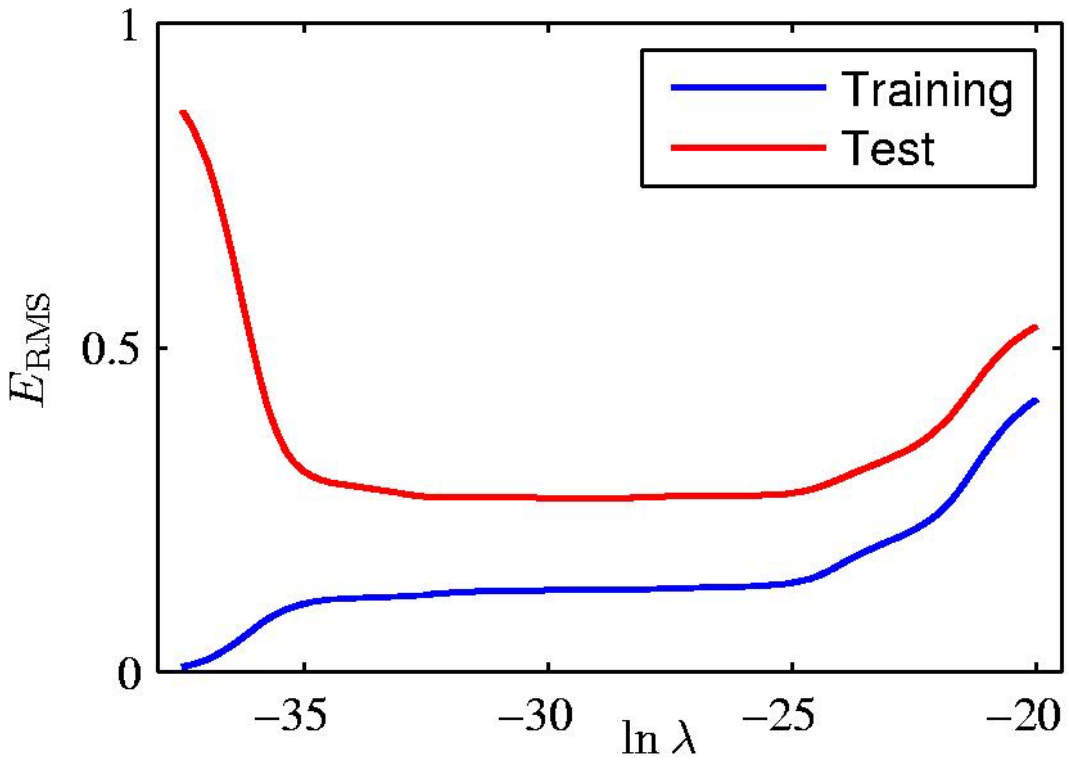


$\ln \lambda = 0$

Polynomial Coefficients

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Regularization: E_{RMS} v.s. $\ln \lambda$



Model selection: Estimation of the optimal value of the regularization parameter. In practice, cross validation is commonly applied for model selection.

Regression Introduction

- Given a training data set comprising N observations $\{\mathbf{x}_n\}$, where $n = 1, \ldots, N$, together with corresponding target values $\{t_n\}$, the goal is to predict the value of t for a new value of $\mathbf{x}$.

Regression Introduction

- Given a training data set comprising N observations $\{\mathbf{x}_n\}$, where $n = 1, \ldots, N$, together with corresponding target values $\{t_n\}$, the goal is to predict the value of t for a new value of $\mathbf{x}$.
- In the simplest approach, this can be done by directly constructing an appropriate function $y(\mathbf{x})$ whose values for new inputs $\mathbf{x}$ constitute the predictions for the corresponding values of t.

Regression Introduction

- Given a training data set comprising N observations $\{\mathbf{x}_n\}$, where $n = 1, \ldots, N$, together with corresponding target values $\{t_n\}$, the goal is to predict the value of t for a new value of $\mathbf{x}$.
- In the simplest approach, this can be done by directly constructing an appropriate function $y(\mathbf{x})$ whose values for new inputs $\mathbf{x}$ constitute the predictions for the corresponding values of t.
- More generally, from a probabilistic perspective, we aim to model the predictive distribution $p(t|\mathbf{x})$ because this expresses our uncertainty about the value of t for each value of $\mathbf{x}$.

Regression Introduction (cont.)

- From this conditional distribution $p(t|\mathbf{x})$ we can make predictions of t, for any new value of $\mathbf{x}$, in such a way as to minimize the expected value of a suitably chosen loss function.

Regression Introduction (cont.)

- From this conditional distribution $p(t|\mathbf{x})$ we can make predictions of t, for any new value of $\mathbf{x}$, in such a way as to minimize the expected value of a suitably chosen loss function.
- A common choice of loss function for real-valued variables is the squared loss, for which the optimal solution is given by the conditional expectation of t.

Regression Introduction (cont.)

- From this conditional distribution $p(t|\mathbf{x})$ we can make predictions of t, for any new value of $\mathbf{x}$, in such a way as to minimize the expected value of a suitably chosen loss function.
- A common choice of loss function for real-valued variables is the squared loss, for which the optimal solution is given by the conditional expectation of t.
- Although linear models have significant limitations as practical techniques for pattern recognition, particularly for problems involving input spaces of high dimensionality, they have nice analytical properties and form the foundation for more sophisticated models to be discussed in later chapters.

Linear Regression

- linear regression models are linear functions of the adjustable parameters, not necessarily linear in the input variables.

Linear Regression

- linear regression models are linear functions of the adjustable parameters, not necessarily linear in the input variables.
- The simplest linear model for regression is one that involves a linear combination of the input variables (features)

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D = w_0 + \sum_{i=1}^{D} w_i x_i$$

Linear Regression

- linear regression models are linear functions of the adjustable parameters, not necessarily linear in the input variables.
- The simplest linear model for regression is one that involves a linear combination of the input variables (features)

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D = w_0 + \sum_{i=1}^{D} w_i x_i$$

- There is an obvious limitation of the linear model (what is it?)

Linear Regression

- linear regression models are linear functions of the adjustable parameters, not necessarily linear in the input variables.
- The simplest linear model for regression is one that involves a linear combination of the input variables (features)

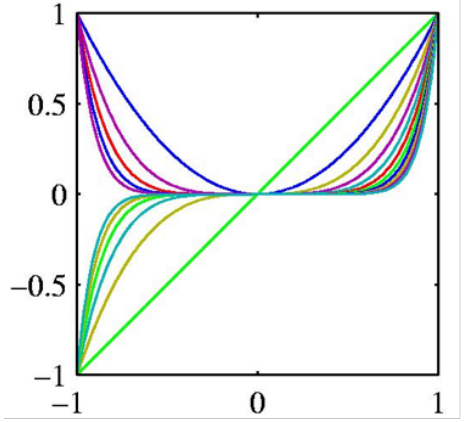
$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D = w_0 + \sum_{i=1}^{D} w_i x_i$$

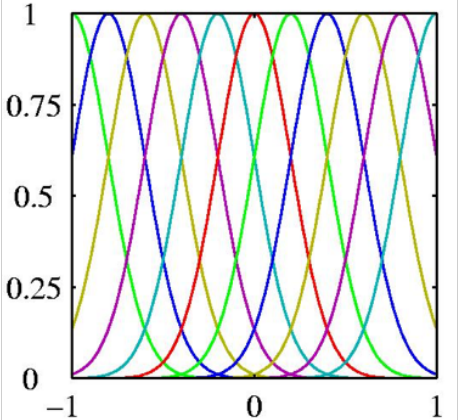
- There is an obvious limitation of the linear model (what is it?), we can overcome the limitation by introducing fixed non-linear transformations, using basis functions $\phi_j(\mathbf{x})$.

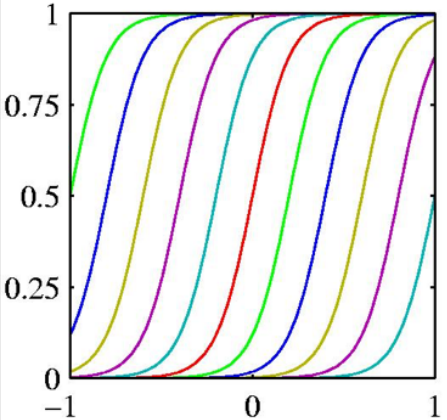
$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

Example of Basis Functions

- Polynomial regression $\phi_j(x) = x^j$
- Gaussian basis function $\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$
- Sigmoidal basis function $\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$, $\sigma(a) = \frac{1}{1+\exp(-a)}$







- Other (Fourier basis, Wavelet, etc.)

Basis Functions in Polynoimal Curve Fitting

- In the curve fitting problem we used $\phi_j(x) = x^j$, polynomial basis functions.

Basis Functions in Polynoimal Curve Fitting

- In the curve fitting problem we used $\phi_j(x) = x^j$, polynomial basis functions.
- Applying basis functions can be considered as the feature engineering process, from which we obtain new (and hopefully better) features from original feature space.

Basis Functions in Polynoimal Curve Fitting

- In the curve fitting problem we used $\phi_j(x) = x^j$, polynomial basis functions.
- Applying basis functions can be considered as the feature engineering process, from which we obtain new (and hopefully better) features from original feature space.
- If the new feature space is larger than and includes the original feature space, does it mean that the new feature space is better?

Dummy Variable

- Add additional dummy basis $\phi_0(\mathbf{x}) = 1$ to match bias x_0:

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \mathbf{\Phi}(\mathbf{x})$$

Maximum likelihood and least squares

- We assume that the target variable t is given by a deterministic function $y(\mathbf{x}, \mathbf{w})$ with additive Gaussian noise so that

$$t = y(\mathbf{x}; \mathbf{w}) + \epsilon \quad \text{where} \quad p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$$

  where ϵ is a zero mean Gaussian random variable with precision (inverse variance) β.

Maximum likelihood and least squares

- We assume that the target variable t is given by a deterministic function $y(\mathbf{x}, \mathbf{w})$ with additive Gaussian noise so that

$$t = y(\mathbf{x}; \mathbf{w}) + \epsilon \quad \text{where} \quad p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$$

  where ϵ is a zero mean Gaussian random variable with precision (inverse variance) β.
- Thus we can write:
  $p(t|\mathbf{x}; \mathbf{w}, \beta) = \mathcal{N}(t - y(\mathbf{x}; \mathbf{w})|0, \beta^{-1}) = \mathcal{N}(t|y(\mathbf{x}; \mathbf{w}), \beta^{-1})$

Maximum likelihood and least squares

- Given a dataset $\mathcal{D} = \{\mathbf{X}, \mathbf{t}\}$, where $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ are features, and $\mathbf{t} = \{t_1, \ldots, t_N\}$ are targets. The probability of data is given by:

$$p(\mathcal{D}; \mathbf{w}, \beta) = p(\mathbf{X}, \mathbf{t}; \mathbf{w}, \beta) = p(\mathbf{t}|\mathbf{X}; \mathbf{w}, \beta)p(\mathbf{X})$$

Maximum likelihood and least squares

- Given a dataset $\mathcal{D} = \{\mathbf{X}, \mathbf{t}\}$, where $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ are features, and $\mathbf{t} = \{t_1, \ldots, t_N\}$ are targets. The probability of data is given by:

$$p(\mathcal{D}; \mathbf{w}, \beta) = p(\mathbf{X}, \mathbf{t}; \mathbf{w}, \beta) = p(\mathbf{t}|\mathbf{X}; \mathbf{w}, \beta)p(\mathbf{X})$$

- When samples are drawn from *i.i.d*, with the linear prediction function $y(\mathbf{x}_n; \mathbf{w}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n)$, the likelihood function is given by:

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

Maximum likelihood and least squares

And the log-likelihood is:

$$
\begin{aligned}
&\ln p(\mathbf{t}|\mathbf{X};\mathbf{w},\beta) \\
&= \ln \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n),\beta^{-1}) = \sum_{n=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n),\beta^{-1}) \\
&= \sum_{n=1}^{N} \ln \frac{1}{(2\pi\beta^{-1})^{1/2}} \exp\left\{-\frac{1}{2\beta^{-1}}(t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n))^2\right\} \\
&= \sum_{n=1}^{N} \left(-\frac{1}{2}\ln(2\pi) + \frac{1}{2}\ln\beta - \frac{\beta}{2}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2\right) \\
&= -\frac{N}{2}\ln(2\pi) + \frac{N}{2}\ln\beta - \beta\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2
\end{aligned}
$$

Maximum likelihood and least squares

- Our goal is to maximize the log-likelihood. Removing constants, we get the following:

$$\arg\max_{\mathbf{w},\beta} \ln p(\mathbf{t}|\mathbf{X};\mathbf{w},\beta) = \arg\min_{\mathbf{w},\beta} \frac{\beta}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2 - \frac{N}{2}\ln\beta$$

Maximum likelihood and least squares

- Our goal is to maximize the log-likelihood. Removing constants, we get the following:

$$\arg\max_{\mathbf{w},\beta} \ln p(\mathbf{t}|\mathbf{X};\mathbf{w},\beta) = \arg\min_{\mathbf{w},\beta} \frac{\beta}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2 - \frac{N}{2}\ln\beta$$

- We note the problem of $\mathbf{w}$ is independent from β:

$$\mathbf{w}_{\mathsf{ML}} = \arg\min_{\mathbf{w}} \frac{1}{2}\sum\nolimits_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

Maximum likelihood and least squares

- Our goal is to maximize the log-likelihood. Removing constants, we get the following:

$$\arg\max_{\mathbf{w},\beta} \ln p(\mathbf{t}|\mathbf{X};\mathbf{w},\beta) = \arg\min_{\mathbf{w},\beta} \frac{\beta}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2 - \frac{N}{2}\ln\beta$$

- We note the problem of $\mathbf{w}$ is independent from β:

$$\mathbf{w}_{\mathsf{ML}} = \arg\min_{\mathbf{w}} \frac{1}{2}\sum\nolimits_{n=1}^{N}\{t_n - \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2$$

- Plugin in $\mathbf{w}_{\mathsf{ML}}$, the optimal solution of inverse β is given by:

$$\frac{1}{\beta_{\mathsf{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{t_n - \mathbf{w}_{\mathsf{ML}}^T\boldsymbol{\phi}(\mathbf{x}_n)\}^2,$$

which is the residual variance of the target values around the regression function.

The Matrix Form

- Introducing design matrix $\mathbf{\Phi} \in \mathbb{R}^{N \times M}$, N samples and M features (after feature engineering):

$$\mathbf{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix} = \begin{pmatrix} \boldsymbol{\phi}(\mathbf{x}_1)^T \\ \boldsymbol{\phi}(\mathbf{x}_2)^T \\ \vdots \\ \boldsymbol{\phi}(\mathbf{x}_N)^T \end{pmatrix}$$

The Matrix Form

- Introducing design matrix $\mathbf{\Phi} \in \mathbb{R}^{N \times M}$, N samples and M features (after feature engineering):

$$\mathbf{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix} = \begin{pmatrix} \boldsymbol{\phi}(\mathbf{x}_1)^T \\ \boldsymbol{\phi}(\mathbf{x}_2)^T \\ \vdots \\ \boldsymbol{\phi}(\mathbf{x}_N)^T \end{pmatrix}$$

- The least squares can then be represented as:

$$\min_{\mathbf{w}} \frac{1}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n)\}^2 = \min_{\mathbf{w}} \left\{ \frac{1}{2} \|\mathbf{\Phi}\mathbf{w} - \mathbf{t}\|_2^2 = \nabla E_D \right\}.$$

Solving the Least Squares using SVD

Solve least squares problem using SVD: $\min_{\mathbf{w}\in\mathbb{R}^n} ||\mathbf{\Phi w} - \mathbf{t}||_2^2$.

- Assume $N \geq M$, the SVD of $\mathbf{\Phi}$ is given by

$$\mathbf{\Phi} = (U_1 \ U_2) \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^T,$$

where $\Sigma \in \mathbb{R}^{N\times M}$ is diagonal, $U_1 \in \mathbb{R}^{N\times M}$ and $U_2 \in \mathbb{R}^{N\times(M-N)}$ have orthonormal columns and $V \in \mathbb{R}^{M\times M}$ is orthogonal.

Solving the Least Squares using SVD

Solve least squares problem using SVD: $\min_{\mathbf{w}\in\mathbb{R}^n} ||\mathbf{\Phi w} - \mathbf{t}||_2^2$.

- Assume $N \geq M$, the SVD of $\mathbf{\Phi}$ is given by

$$\mathbf{\Phi} = (U_1 \ U_2) \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^T,$$

  where $\Sigma \in \mathbb{R}^{N\times M}$ is diagonal, $U_1 \in \mathbb{R}^{N\times M}$ and $U_2 \in \mathbb{R}^{N\times(M-N)}$ have orthonormal columns and $V \in \mathbb{R}^{M\times M}$ is orthogonal.
- Plugin the least squares

Solving the Least Squares using SVD

Solve least squares problem using SVD: $\min_{\mathbf{w}\in\mathbb{R}^n} ||\mathbf{\Phi w} - \mathbf{t}||_2^2$.

- Assume $N \geq M$, the SVD of $\mathbf{\Phi}$ is given by

$$\mathbf{\Phi} = (U_1 \;\; U_2) \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^T,$$

where $\Sigma \in \mathbb{R}^{N\times M}$ is diagonal, $U_1 \in \mathbb{R}^{N\times M}$ and $U_2 \in \mathbb{R}^{N\times(M-N)}$ have orthonormal columns and $V \in \mathbb{R}^{M\times M}$ is orthogonal.

- Plugin the least squares and we have

$$\begin{aligned} & ||\mathbf{\Phi w} - \mathbf{t}||_2^2 \\ &= \left\| (U_1 \;\; U_2) \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^T \mathbf{w} - \mathbf{t} \right\|_2^2 = \left\| \begin{pmatrix} \Sigma \\ 0 \end{pmatrix} V^T \mathbf{w} - (U_1 \;\; U_2)^T \mathbf{t} \right\|_2^2 \\ &= \left\| \begin{pmatrix} \Sigma y \\ 0 \end{pmatrix} - \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \right\|_2^2 = \|\Sigma y - b_1\|_2^2 + \|b_2\|_2^2 , \end{aligned}$$

where $y = V^T\mathbf{w}$, $b_1 = U_1^T\mathbf{t}$, and $b_2 = U_2^T\mathbf{t}$.

Solving the Least Squares using SVD (cont.)

- Given

$$||\mathbf{\Phi w} - \mathbf{t}||_2^2 = \|\Sigma y - b_1\|_2^2 + \|b_2\|_2^2 ,$$

  where $y = V^T \mathbf{w}$, $b_1 = U_1^T \mathbf{t}$, and $b_2 = U_2^T \mathbf{t}$

- The optimal y is given by

Solving the Least Squares using SVD (cont.)

- Given

$$||\mathbf{\Phi w} - \mathbf{t}||_2^2 = \|\Sigma y - b_1\|_2^2 + \|b_2\|_2^2 ,$$

  where $y = V^T \mathbf{w}$, $b_1 = U_1^T \mathbf{t}$, and $b_2 = U_2^T \mathbf{t}$
- The optimal y is given by $y = \Sigma^{-1} b_1$.

Solving the Least Squares using SVD (cont.)

- Given

$$||\mathbf{\Phi w} - \mathbf{t}||_2^2 = \|\Sigma y - b_1\|_2^2 + \|b_2\|_2^2 ,$$

  where $y = V^T \mathbf{w}$, $b_1 = U_1^T \mathbf{t}$, and $b_2 = U_2^T \mathbf{t}$
- The optimal y is given by $y = \Sigma^{-1} b_1$.
- Thus, the least squares solution is given by

$$\mathbf{w} = Vy = V\Sigma^{-1} b_1 = V\Sigma^{-1} U_1^T \mathbf{t} = \sum_{i=1}^{M} \frac{u_i^T \mathbf{t}}{\sigma_i} v_i .$$

Solving the Least Squares using SVD (cont.)

- Given

$$||\mathbf{\Phi w} - \mathbf{t}||_2^2 = \|\Sigma y - b_1\|_2^2 + \|b_2\|_2^2 ,$$

  where $y = V^T \mathbf{w}$, $b_1 = U_1^T \mathbf{t}$, and $b_2 = U_2^T \mathbf{t}$
- The optimal y is given by $y = \Sigma^{-1} b_1$.
- Thus, the least squares solution is given by

$$\mathbf{w} = Vy = V\Sigma^{-1} b_1 = V\Sigma^{-1} U_1^T \mathbf{t} = \sum_{i=1}^{M} \frac{u_i^T \mathbf{t}}{\sigma_i} v_i.$$

  which is a weighted average of right singular vectors.
- What if $\sigma_i \to 0$?

Solving the Least Squares using Normal Equation

- Alternatively solution: set the gradient of least squares to 0
  - Some useful rules $\nabla_{\mathbf{w}}(\mathbf{w}^T A\mathbf{w}) = (A + A^T)\mathbf{w}$, $\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{b}) = \mathbf{b}$

Solving the Least Squares using Normal Equation

- Alternatively solution: set the gradient of least squares to 0
  - Some useful rules $\nabla_{\mathbf{w}}(\mathbf{w}^T A\mathbf{w}) = (A + A^T)\mathbf{w}$, $\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{b}) = \mathbf{b}$

$$\nabla_{\mathbf{w}} \frac{1}{2}\|\mathbf{\Phi w} - \mathbf{t}\|_2^2 = \mathbf{\Phi}^T(\mathbf{\Phi w} - \mathbf{t}) = \mathbf{\Phi}^T\mathbf{\Phi w} - \mathbf{\Phi}^T\mathbf{t} = 0$$

$$\Rightarrow \mathbf{\Phi}^T\mathbf{\Phi w} = \mathbf{\Phi}^T\mathbf{t} \Rightarrow \mathbf{w}^* = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$$

Solving the Least Squares using Normal Equation

- Alternatively solution: set the gradient of least squares to 0
  - Some useful rules $\nabla_{\mathbf{w}}(\mathbf{w}^T A\mathbf{w}) = (A + A^T)\mathbf{w}$, $\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{b}) = \mathbf{b}$

$$\nabla_{\mathbf{w}} \frac{1}{2}\|\mathbf{\Phi w} - \mathbf{t}\|_2^2 = \mathbf{\Phi}^T(\mathbf{\Phi w} - \mathbf{t}) = \mathbf{\Phi}^T\mathbf{\Phi w} - \mathbf{\Phi}^T\mathbf{t} = 0$$

$$\Rightarrow \mathbf{\Phi}^T\mathbf{\Phi w} = \mathbf{\Phi}^T\mathbf{t} \Rightarrow \mathbf{w}^* = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$$

  - Therefore $\mathbf{\Phi w}^* = \mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$.

Solving the Least Squares using Normal Equation

- Alternatively solution: set the gradient of least squares to 0
  - Some useful rules $\nabla_{\mathbf{w}}(\mathbf{w}^T A\mathbf{w}) = (A + A^T)\mathbf{w}$, $\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{b}) = \mathbf{b}$

$$\nabla_{\mathbf{w}} \frac{1}{2}\|\mathbf{\Phi w} - \mathbf{t}\|_2^2 = \mathbf{\Phi}^T(\mathbf{\Phi w} - \mathbf{t}) = \mathbf{\Phi}^T\mathbf{\Phi w} - \mathbf{\Phi}^T\mathbf{t} = 0$$

$$\Rightarrow \mathbf{\Phi}^T\mathbf{\Phi w} = \mathbf{\Phi}^T\mathbf{t} \Rightarrow \mathbf{w}^* = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$$

  - Therefore $\mathbf{\Phi w}^* = \mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$.
  - Geometry: This leads to a projection to the space spanned by the columns of $\mathbf{\Phi}$ (why?)

Solving the Least Squares using Normal Equation

- Alternatively solution: set the gradient of least squares to 0
  - Some useful rules $\nabla_{\mathbf{w}}(\mathbf{w}^T A\mathbf{w}) = (A + A^T)\mathbf{w}$, $\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{b}) = \mathbf{b}$

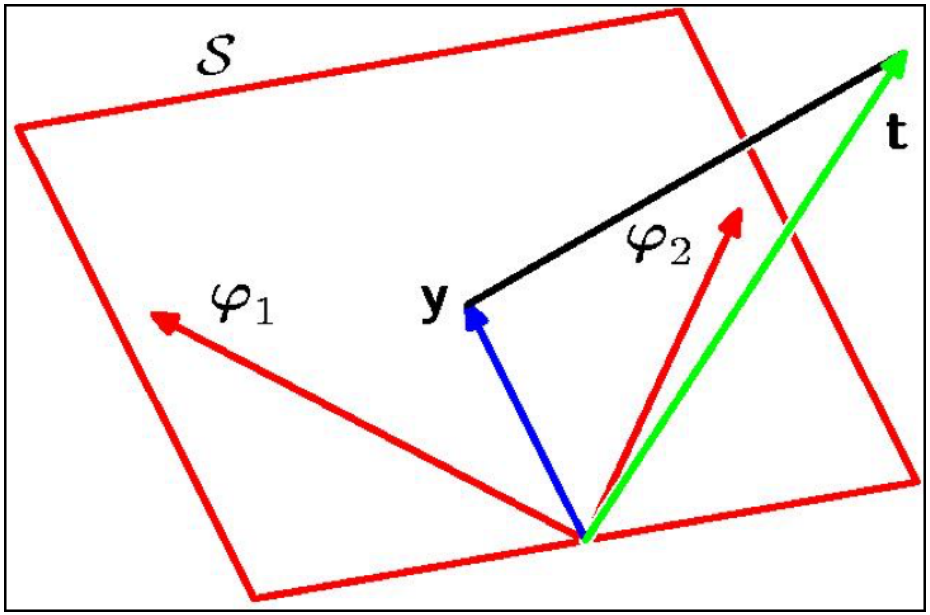
$$\nabla_{\mathbf{w}} \frac{1}{2}\|\mathbf{\Phi w} - \mathbf{t}\|_2^2 = \mathbf{\Phi}^T(\mathbf{\Phi w} - \mathbf{t}) = \mathbf{\Phi}^T\mathbf{\Phi w} - \mathbf{\Phi}^T\mathbf{t} = 0$$

$$\Rightarrow \mathbf{\Phi}^T\mathbf{\Phi w} = \mathbf{\Phi}^T\mathbf{t} \Rightarrow \mathbf{w}^* = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$$

  - Therefore $\mathbf{\Phi w}^* = \mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$.
  - Geometry: This leads to a projection to the space spanned by the columns of $\mathbf{\Phi}$ (why?)
  - When $\mathbf{\Phi}^T\mathbf{\Phi}$ is close to singular, we would have numerical problems.

Geometry of Least Squares

- The prediction $\mathbf{y} = \mathbf{\Phi}\mathbf{w}^* = \mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$
  - Orthogonal projection of $\mathbf{t}$ onto the subspace $\mathcal{S}$, which is spanned by the basis functions $\phi_j(\mathbf{x})$ (columns of $\mathbf{\Phi}$)



Solving the Least Squares using Normal Equation

- We call $\mathbf{\Phi}^\dagger \equiv (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T$ pseudo-inverse (generalization of inverse to non-square matrix). See for a square invertable matrix $\mathbf{\Phi}$, we have $\mathbf{\Phi}^\dagger = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T = \mathbf{\Phi}^{-1}(\mathbf{\Phi}^T)^{-1}\mathbf{\Phi}^T = \mathbf{\Phi}^{-1}$

Solving the Least Squares using Normal Equation

- We call $\mathbf{\Phi}^\dagger \equiv (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T$ pseudo-inverse (generalization of inverse to non-square matrix). See for a square invertable matrix $\mathbf{\Phi}$, we have $\mathbf{\Phi}^\dagger = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T = \mathbf{\Phi}^{-1}(\mathbf{\Phi}^T)^{-1}\mathbf{\Phi}^T = \mathbf{\Phi}^{-1}$
- When $\text{rank}(\mathbf{\Phi}) = r \leq \min(M, N)$, let $\mathbf{\Phi} = U_r\Sigma_r V_r^T$ be the economical SVD

$$\begin{aligned}\mathbf{w} &= (V_r\Sigma_r U_r^T U_r\Sigma_r V_r^T)^{-1} V_r S_r U_r^T \mathbf{t} \\ &= V_r\Sigma_r^{-2} V_r^T V_r\Sigma_r U_r^T \mathbf{t} \\ &= V_r\Sigma_r^{-1} U_r^T \mathbf{t} = \sum_{i=1}^{r} \frac{u_i^T\mathbf{t}}{\sigma_i} v_i\end{aligned}$$

Solving the Least Squares using Gradient Descent

- Since SVD is too expensive, typically we use gradient descent. Conceptually for each iteration:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla E_D = \mathbf{w}^t - \eta(\mathbf{\Phi}^T \mathbf{\Phi} \mathbf{w} - \mathbf{\Phi}^T \mathbf{t})$$

Solving the Least Squares using Gradient Descent

- Since SVD is too expensive, typically we use gradient descent. Conceptually for each iteration:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla E_D = \mathbf{w}^t - \eta(\mathbf{\Phi}^T \mathbf{\Phi} \mathbf{w} - \mathbf{\Phi}^T \mathbf{t})$$

- How to efficient compute the gradient $\mathbf{\Phi}^T \mathbf{\Phi} \mathbf{w} - \mathbf{\Phi}^T \mathbf{t}$?

Solving the Least Squares using Gradient Descent

- Since SVD is too expensive, typically we use gradient descent. Conceptually for each iteration:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla E_D = \mathbf{w}^t - \eta(\mathbf{\Phi}^T \mathbf{\Phi} \mathbf{w} - \mathbf{\Phi}^T \mathbf{t})$$

- How to efficient compute the gradient $\mathbf{\Phi}^T \mathbf{\Phi} \mathbf{w} - \mathbf{\Phi}^T \mathbf{t}$?
  - Recall one classifical problem in dynamic programming – Matrix Chain Multiplication.
    - $A \in \mathbb{R}^{10 \times 30}, B \in \mathbb{R}^{30 \times 5}, C \in \mathbb{R}^{5 \times 60}$

Solving the Least Squares using Gradient Descent

- Since SVD is too expensive, typically we use gradient descent. Conceptually for each iteration:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla E_D = \mathbf{w}^t - \eta(\mathbf{\Phi}^T \mathbf{\Phi} \mathbf{w} - \mathbf{\Phi}^T \mathbf{t})$$

- How to efficient compute the gradient $\mathbf{\Phi}^T \mathbf{\Phi} \mathbf{w} - \mathbf{\Phi}^T \mathbf{t}$?
  - Recall one classifical problem in dynamic programming – Matrix Chain Multiplication.
    - $A \in \mathbb{R}^{10 \times 30}, B \in \mathbb{R}^{30 \times 5}, C \in \mathbb{R}^{5 \times 60}$
    - $(AB)C = (10 \times 30 \times 5) + (10 \times 5 \times 60) = 1500 + 3000 = 4500$ ops
    - $A(BC) = (30 \times 5 \times 60) + (10 \times 30 \times 60) = 9000 + 18000 = 27000$ ops

Advanced Gradient Techniques

- What if data is very large, and compute the gradient is nearly impossible?

Advanced Gradient Techniques

- What if data is very large, and compute the gradient is nearly impossible? We can use stochastic gradient. Each time we see one data point n, we have loss function $E_n = (t_n - \mathbf{w}^{(t)T}\phi_n)^2$, then we perform gradient descent.

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla E_n = \mathbf{w}^t - \eta \boldsymbol{\phi}_n (\mathbf{w}^{(t)T} \boldsymbol{\phi}_n - t_n)$$

where $\phi_n = \boldsymbol{\phi}(\mathbf{x}_n)$

Advanced Gradient Techniques

- What if data is very large, and compute the gradient is nearly impossible? We can use stochastic gradient. Each time we see one data point n, we have loss function $E_n = (t_n - \mathbf{w}^{(t)T}\phi_n)^2$, then we perform gradient descent.

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla E_n = \mathbf{w}^t - \eta \boldsymbol{\phi}_n (\mathbf{w}^{(t)T} \boldsymbol{\phi}_n - t_n)$$

  where $\phi_n = \boldsymbol{\phi}(\mathbf{x}_n)$

- How about data is located in different servers ($\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_s\}$)?

Advanced Gradient Techniques

- What if data is very large, and compute the gradient is nearly impossible? We can use stochastic gradient. Each time we see one data point n, we have loss function $E_n = (t_n - \mathbf{w}^{(t)T}\phi_n)^2$, then we perform gradient descent.

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \nabla E_n = \mathbf{w}^t - \eta \boldsymbol{\phi}_n (\mathbf{w}^{(t)T} \boldsymbol{\phi}_n - t_n)$$

  where $\phi_n = \boldsymbol{\phi}(\mathbf{x}_n)$

- How about data is located in different servers ($\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_s\}$)?

$$\begin{aligned}\nabla E_D &= \boldsymbol{\Phi}^T(\boldsymbol{\Phi}\mathbf{w} - \mathbf{t}) = \sum_{i=1}^{N} \boldsymbol{\phi}_i(\boldsymbol{\phi}_i^T \mathbf{w} - t_i) \\ &= \sum_{i \in \mathcal{S}_1} \boldsymbol{\phi}_i(\boldsymbol{\phi}_i^T \mathbf{w} - t_i) + \cdots + \sum_{i \in \mathcal{S}_s} \boldsymbol{\phi}_i(\boldsymbol{\phi}_i^T \mathbf{w} - t_i)\end{aligned}$$

Multiple Outputs

- Predict K targets simultaneously $\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x})$

Multiple Outputs

- Predict K targets simultaneously $\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x})$
  - Assume the observation is from a multi-response deterministic function with added Isotropic Gaussian distribution $\mathbf{t} = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}) + \boldsymbol{\epsilon}$, then $p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I})$.

Multiple Outputs

- Predict K targets simultaneously $\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x})$
  - Assume the observation is from a multi-response deterministic function with added Isotropic Gaussian distribution $\mathbf{t} = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}) + \boldsymbol{\epsilon}$, then $p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I})$.
  - Given a set of observations $t_1, \ldots, t_N$, we can combine these into a matrix $\mathbf{T} \in \mathbb{R}^{N \times K}$, and the likelihood function is given by

$$\ln p(\mathbf{T}|\mathbf{X}, \mathbf{W}, \beta) = \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n|\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}\mathbf{I})$$

$$= \frac{NK}{2} \ln \left(\frac{\beta}{2\pi} \right) - \frac{\beta}{2} \sum_{n=1}^{N} \|\mathbf{t}_n - \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n)\|^2$$

Multiple Outputs

- Predict K targets simultaneously $\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x})$
  - Assume the observation is from a multi-response deterministic function with added Isotropic Gaussian distribution $\mathbf{t} = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}) + \boldsymbol{\epsilon}$, then $p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I})$.
  - Given a set of observations $t_1, \ldots, t_N$, we can combine these into a matrix $\mathbf{T} \in \mathbb{R}^{N \times K}$, and the likelihood function is given by

$$\ln p(\mathbf{T}|\mathbf{X}, \mathbf{W}, \beta) = \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n|\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}\mathbf{I})$$

$$= \frac{NK}{2} \ln \left(\frac{\beta}{2\pi} \right) - \frac{\beta}{2} \sum_{n=1}^{N} \|\mathbf{t}_n - \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n)\|^2$$

  - Maximize the log-likelihood gives $\mathbf{W}_{\text{ML}} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{T}$.

Multiple Outputs

- Predict K targets simultaneously $\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x})$
  - Assume the observation is from a multi-response deterministic function with added Isotropic Gaussian distribution $\mathbf{t} = \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}) + \boldsymbol{\epsilon}$, then $p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I})$.
  - Given a set of observations $t_1, \ldots, t_N$, we can combine these into a matrix $\mathbf{T} \in \mathbb{R}^{N \times K}$, and the likelihood function is given by

$$\ln p(\mathbf{T}|\mathbf{X}, \mathbf{W}, \beta) = \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n|\mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}\mathbf{I})$$

$$= \frac{NK}{2} \ln \left(\frac{\beta}{2\pi} \right) - \frac{\beta}{2} \sum_{n=1}^{N} \|\mathbf{t}_n - \mathbf{W}^T \boldsymbol{\phi}(\mathbf{x}_n)\|^2$$

  - Maximize the log-likelihood gives $\mathbf{W}_{\text{ML}} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{T}$.
- Compare the solution from solving a multi-response least sqaures, and the solution from solving a set of least squares independently. Are they the same?