AMCS/CS229: Machine Learning

Linear Regression

Xiangliang Zhang

King Abdullah University of Science and Technology

KAUST
King Abdullah University of Science and Technology

Questions from last class

- What is Machine Learning? Your definition
- What are the main types of learning?
- What is the difference between these types of learning?

Start to Learn

[Christopher M. Bishop.
Pattern Recognition and Machine Learning
2007]

Regression

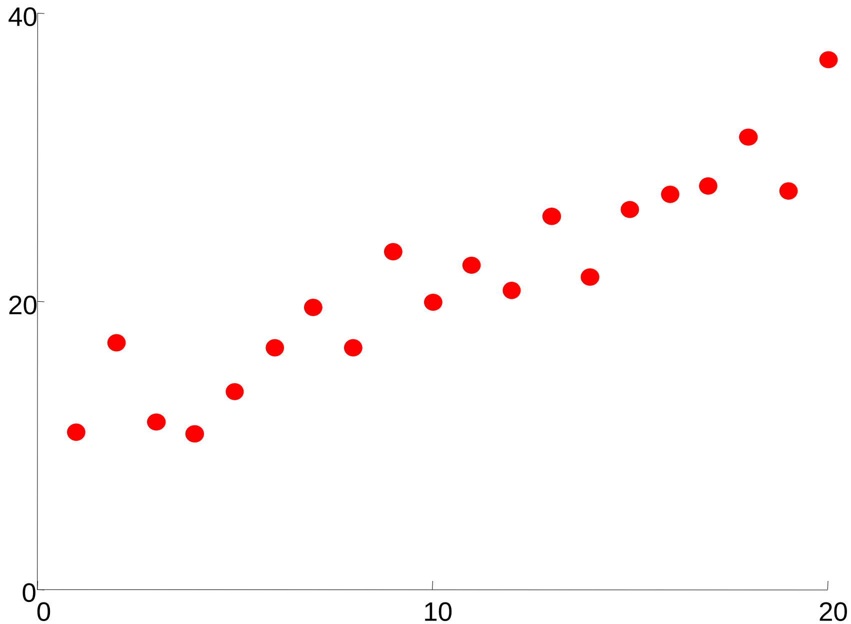
X → Y

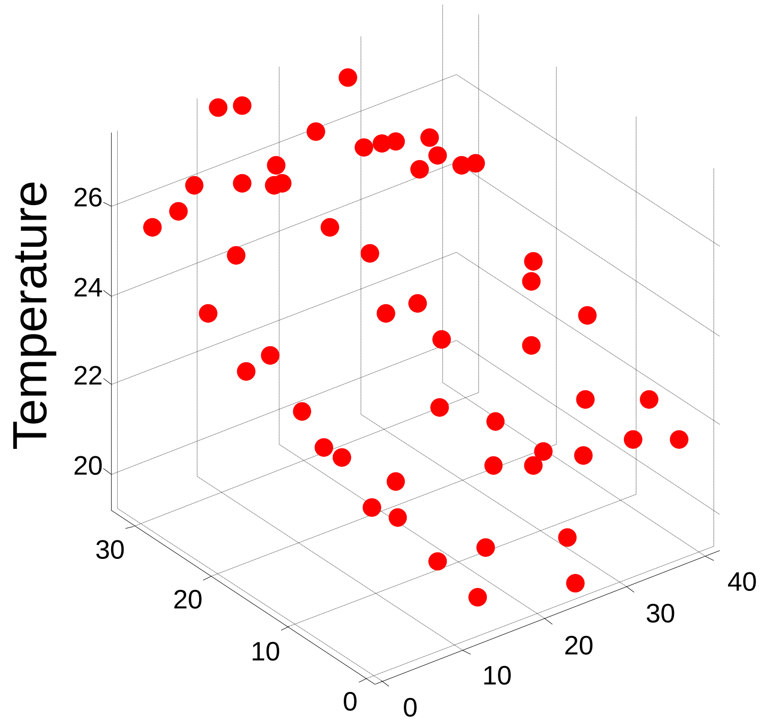
Anything:

- continuous ($\Re$, $\Re^d$, …)

- **continuous:**
  - **$\Re$, $\Re^d$**

Linear Regression

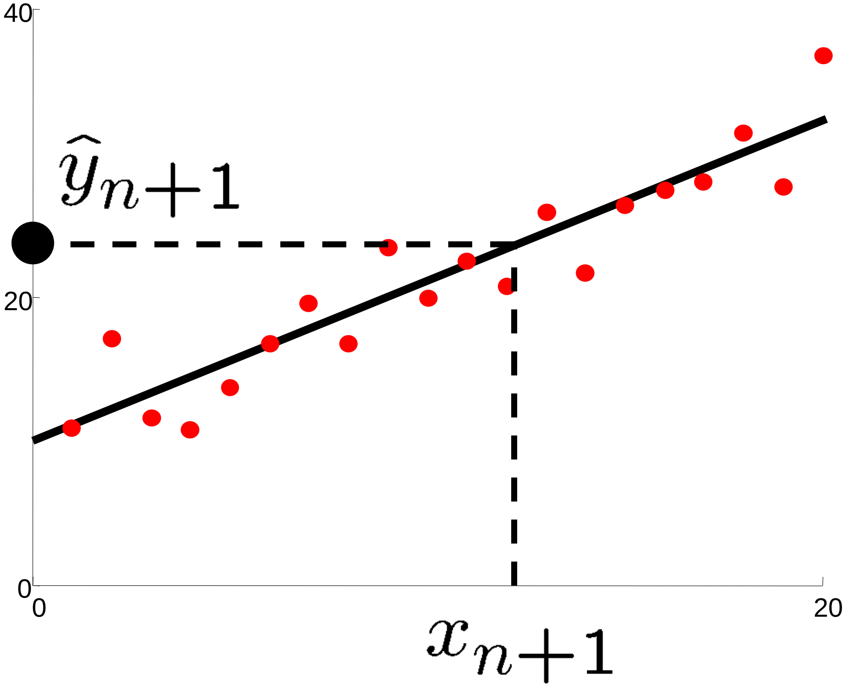


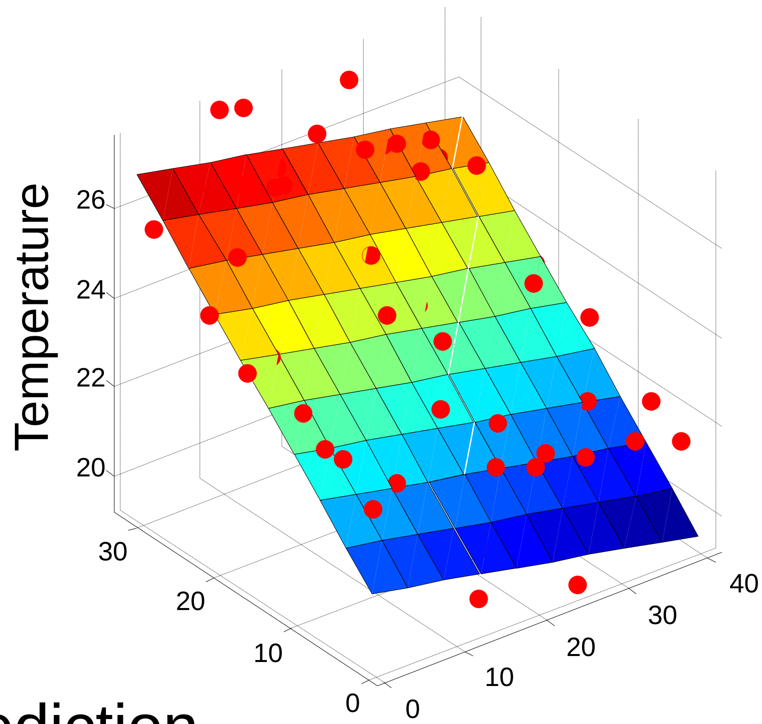


Given examples $(x_i, y_i)_{i=1...n}$

Predict y_{n+1} given a new point x_{n+1}

Linear Regression





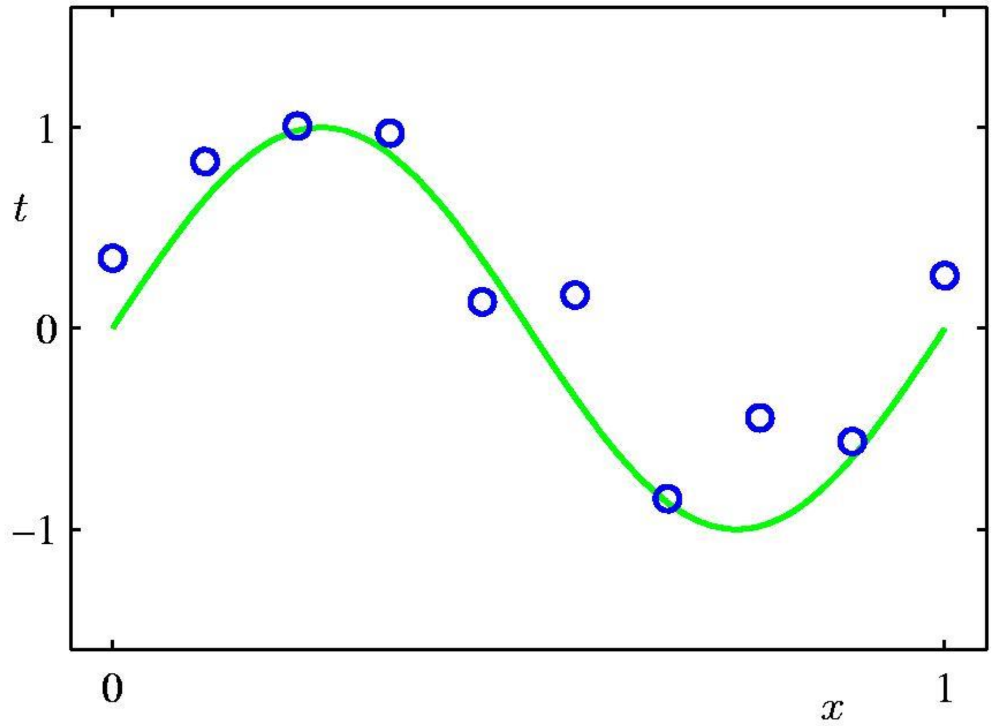
Prediction

$$\widehat{y}_i = w_0 + w_1 x_i$$

Prediction

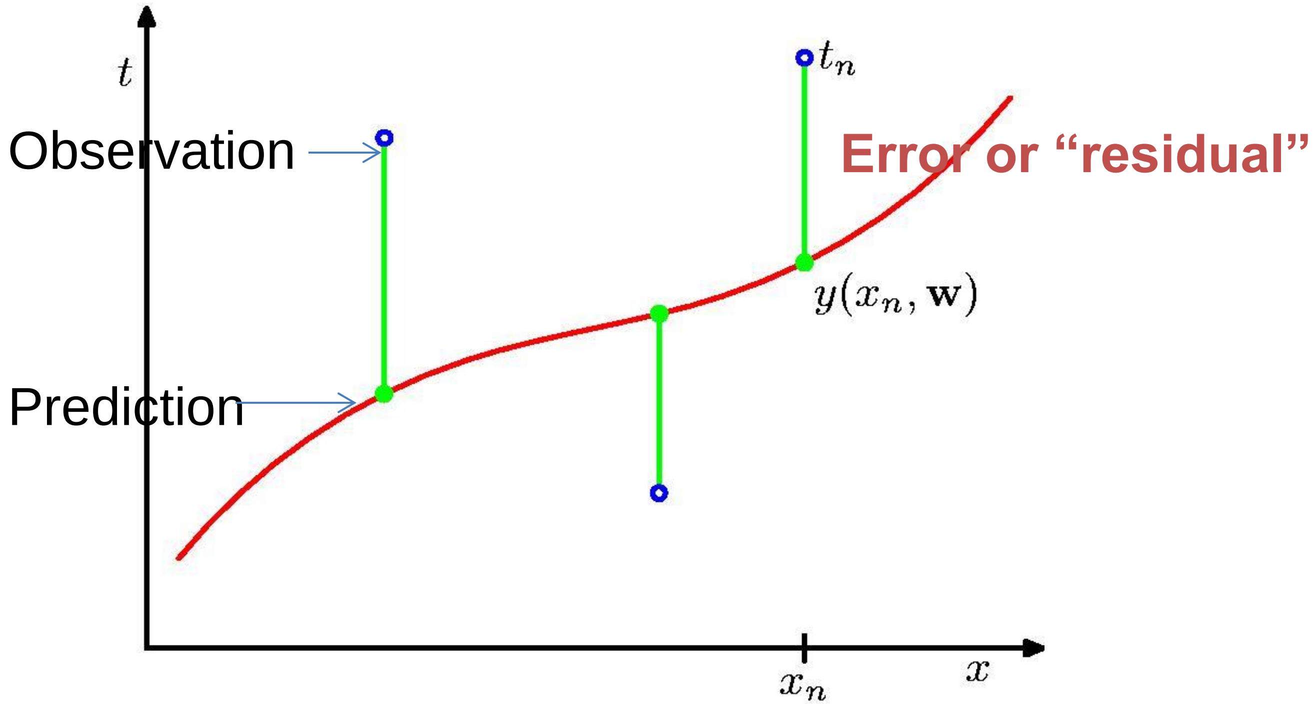
$$\begin{aligned}\widehat{y}_i &= w_0 + w_1 x_{i,1} + w_2 x_{i,2} \\ &= \begin{pmatrix} 1 & x_{i,1} & x_{i,2} \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \end{pmatrix} \\ &= X_i^\top w\end{aligned}$$

Polynomial Curve Fitting



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

Sum-of-Squares Error Function



minimizing

Sum squared error

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

Sum-of-Squares Error Function

Minimizing *E*(w)

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

to find w*

E(w) ---- quadratic function of w

Derivative of *E*(w) w.r.t. w --- linear of w

⟹ unique solution for minimizing *E*(w)

Gradient descent algorithm

1. **Batch** gradient descent

2. **Stochastic** gradient descent, or
   Incremental gradient descent

Batch gradient descent algorithm (1)

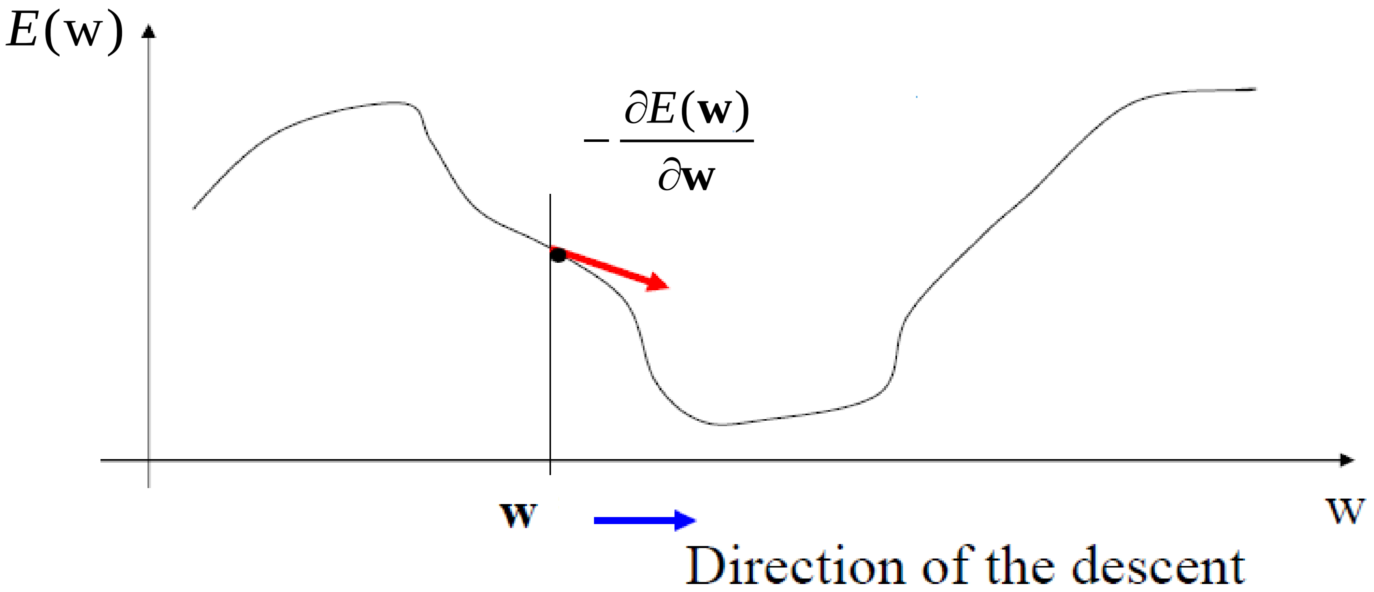
Batch gradient descent

$$w_i^{\tau+1} := w_i^{\tau} - \eta \frac{\partial E(\mathrm{w})}{\partial w_i}$$

$$:= w_i^{\tau} - \eta \sum_{n=1}^{N} (y(x_n, \mathrm{w}) - t_n) \frac{\partial y(x_n, \mathrm{w})}{\partial w_i}$$

Batch gradient descent algorithm (2)

Batch gradient descent example

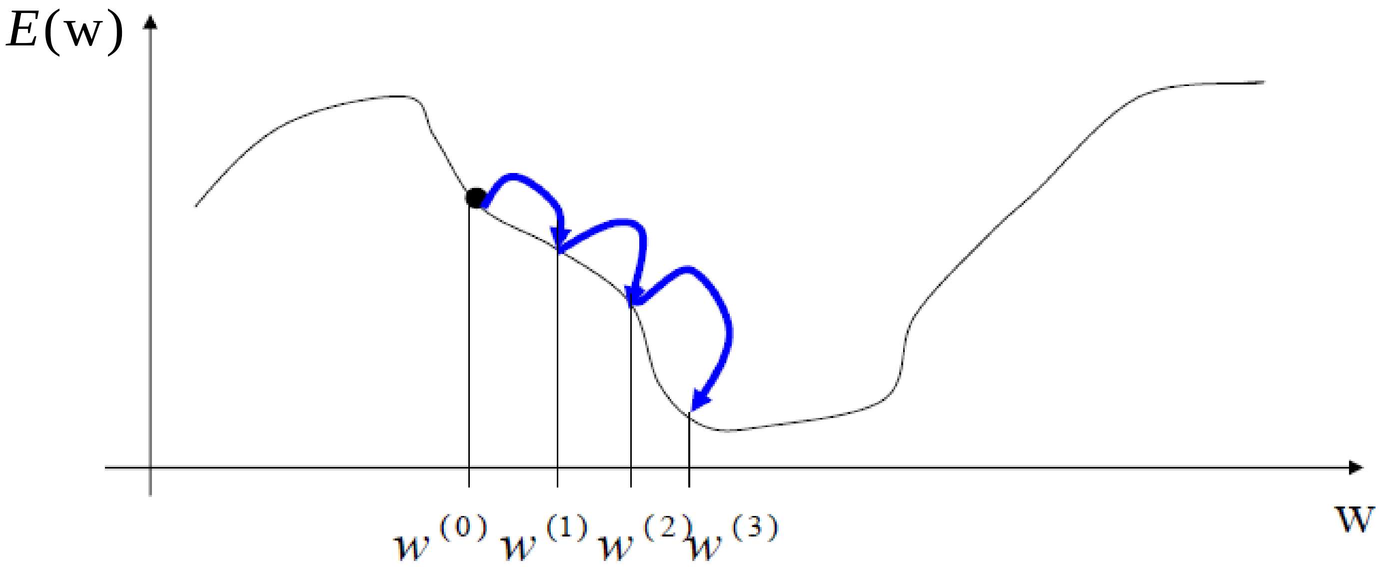


Update the value of w according to the gradient

$$\mathrm{w}^{\tau+1} := \mathrm{w}^{\tau} - \eta \frac{\partial E(\mathrm{w})}{\partial \mathrm{w}}$$

Batch gradient descent algorithm (3)

Batch gradient descent example



Iteratively approaches the optimum of the Error function

Stochastic gradient descent algorithm (1)

Stochastic gradient descent, or

Incremental gradient descent

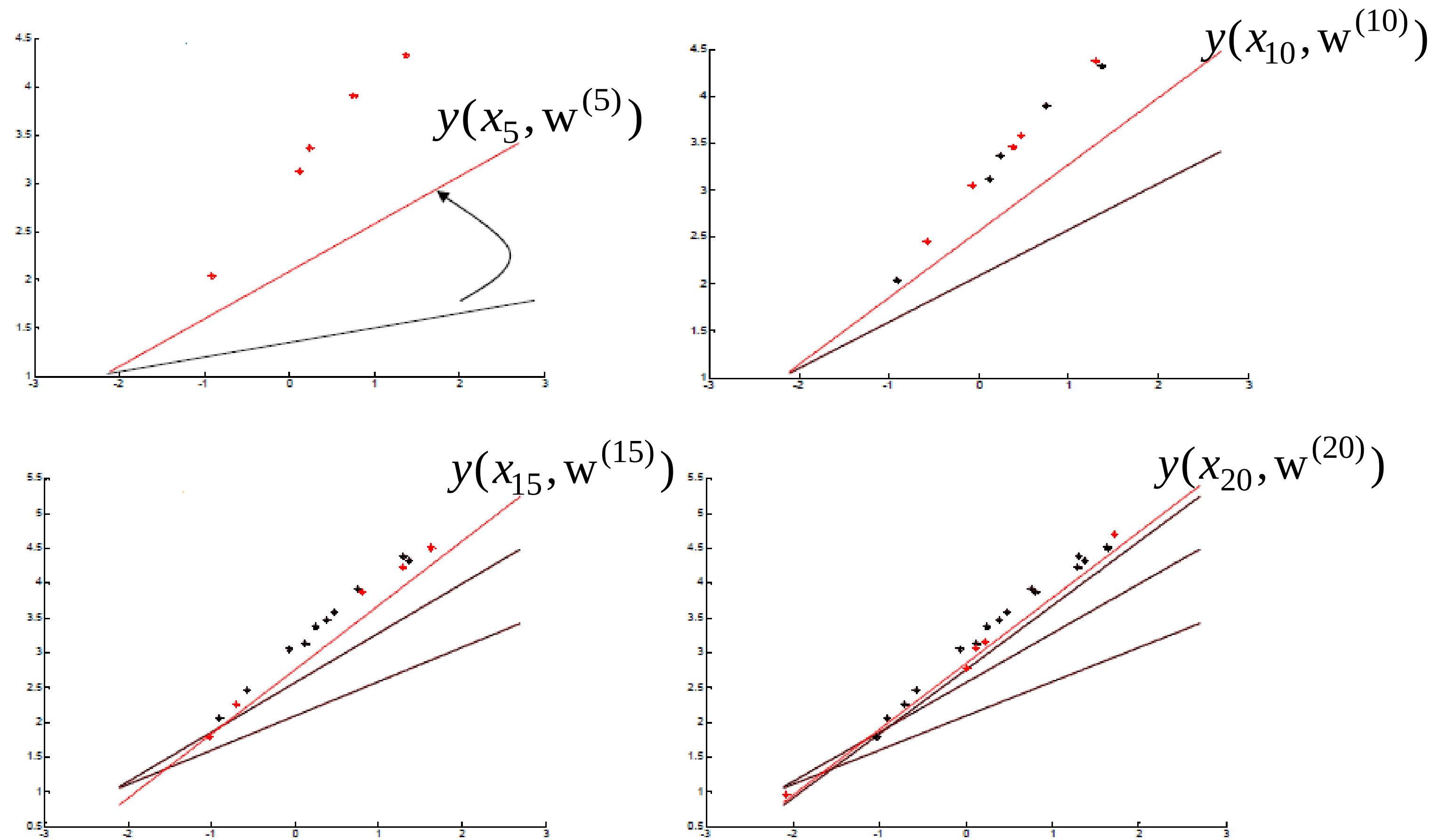
$$\text{for n} = 1 : \text{N}$$

$$w_i^n := w_i^{n-1} - \eta \frac{\partial E_n(\mathrm{w})}{\partial w_i}$$

$$:= w_i^{n-1} - \eta (y(x_n, \mathrm{w}) - t_n) \frac{\partial y(x_n, \mathrm{w})}{\partial w_i}$$

Stochastic gradient descent algorithm (2)

An example:

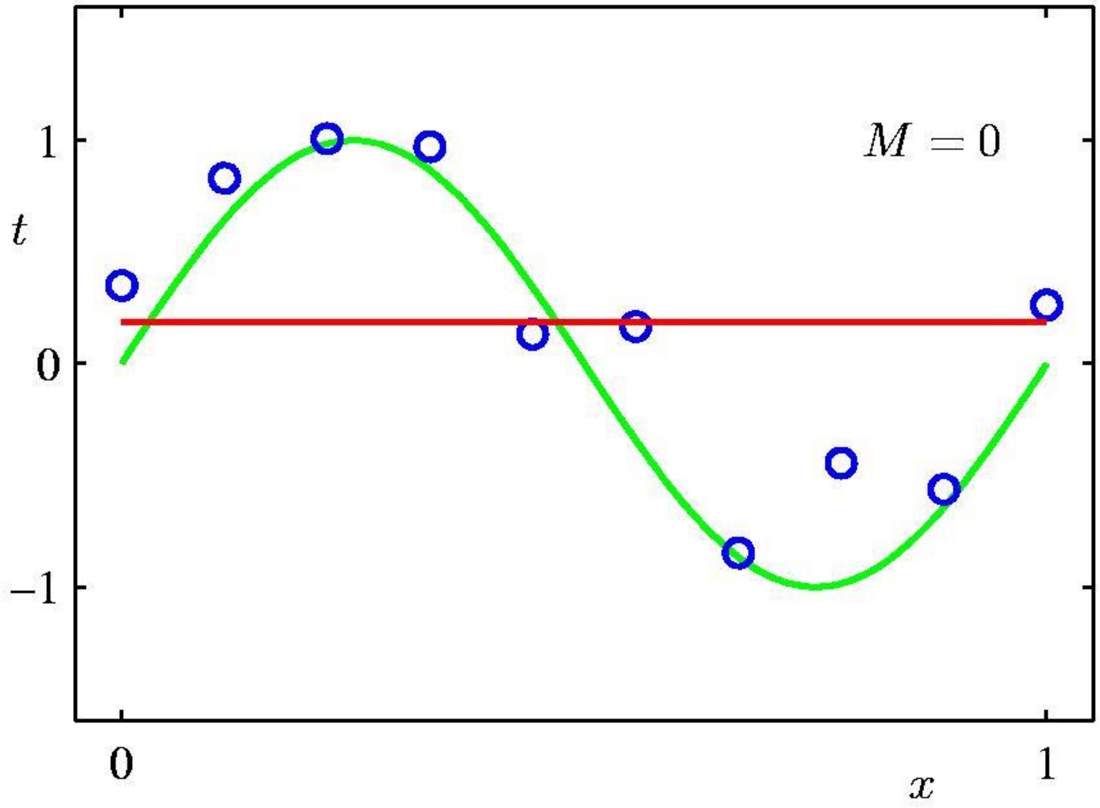


The more data, the closer to the optimum of the Error function

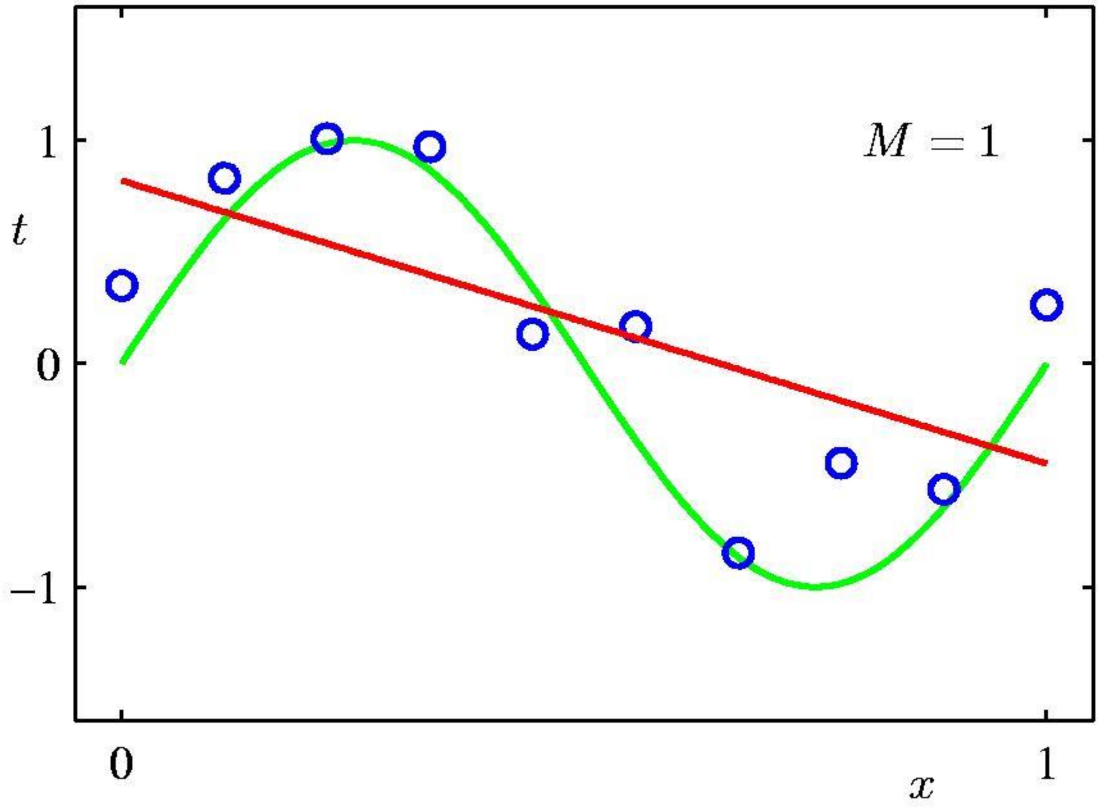
M=?

How many coefficient?

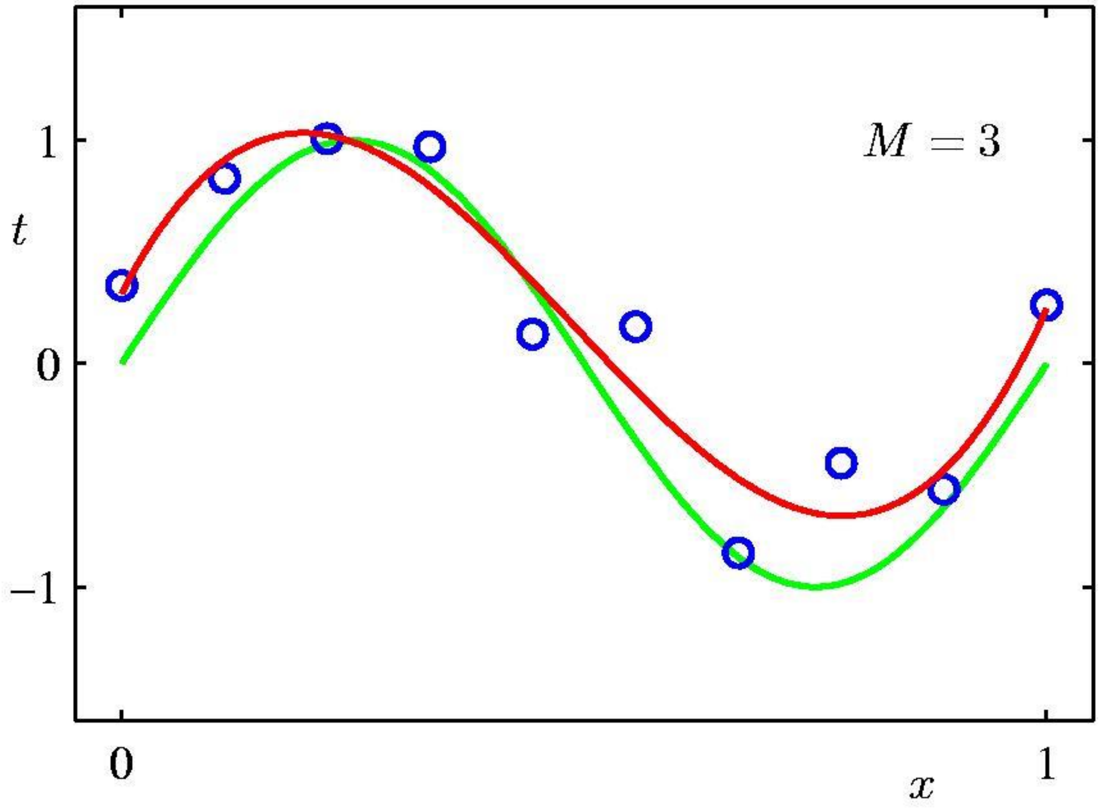
0^{th} Order Polynomial



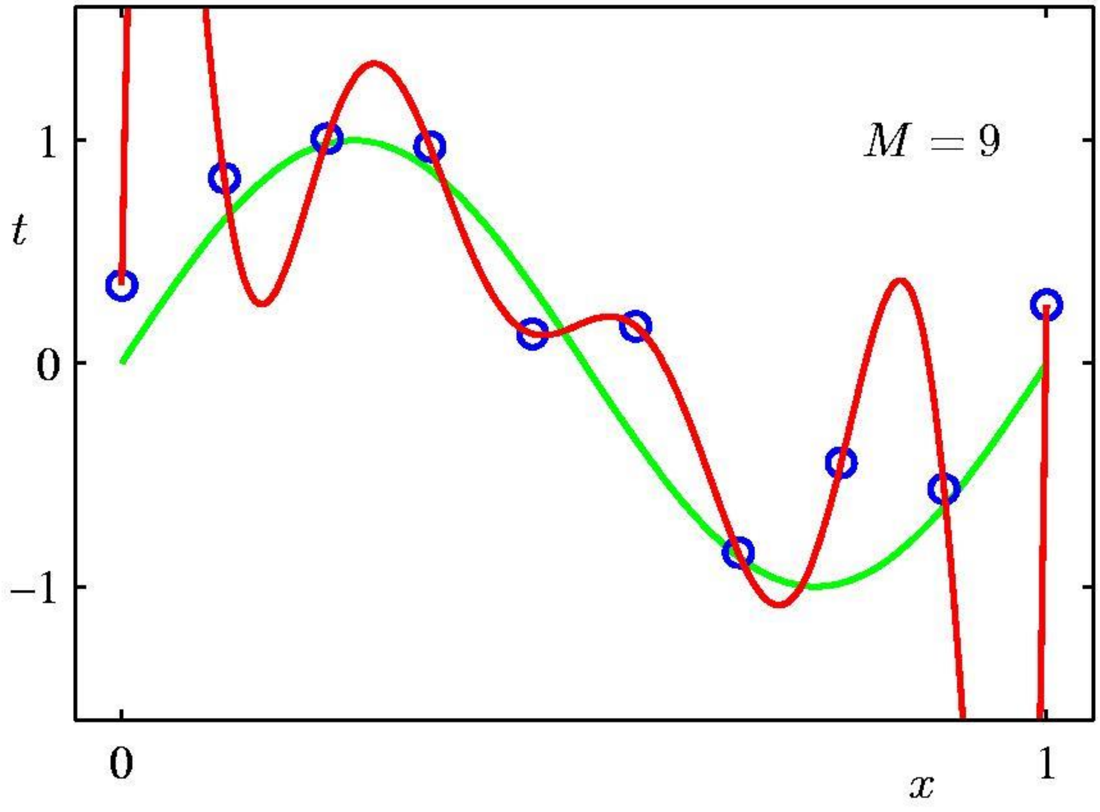
1st Order Polynomial



3rd Order Polynomial



9^{th} Order Polynomial

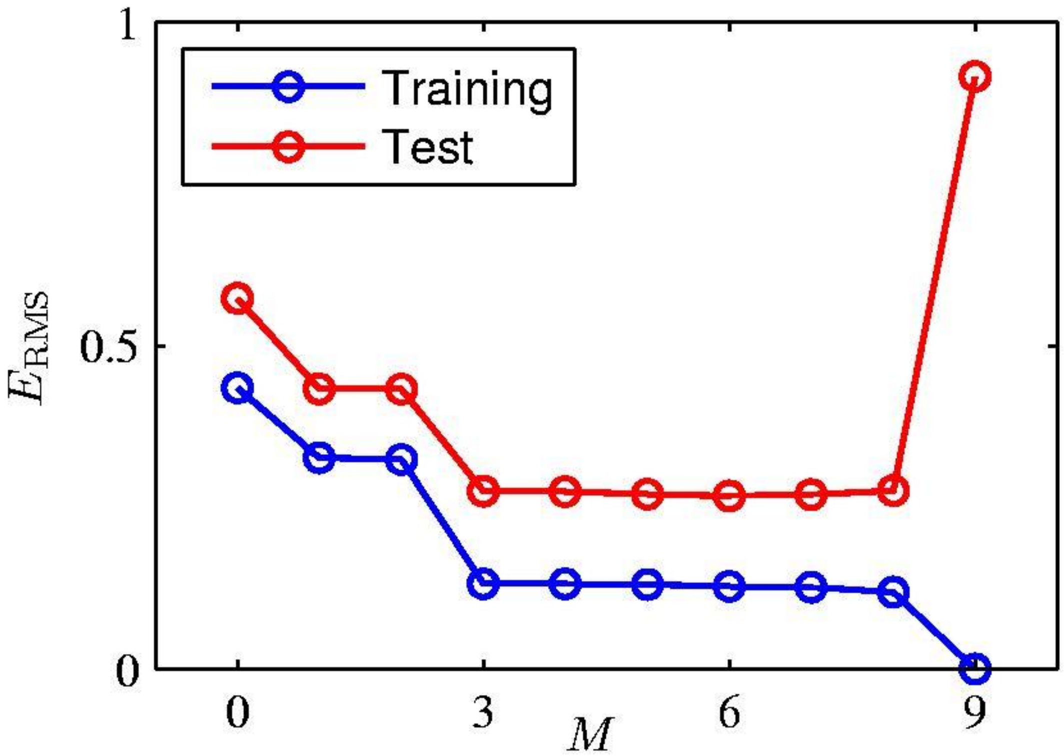


Overfitting

Over-fitting

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

- division by N: **compare different sizes of data sets on an equal footing**
- square root: RMS is **measured on the same scale** (and in the same units) as the target variable t.



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

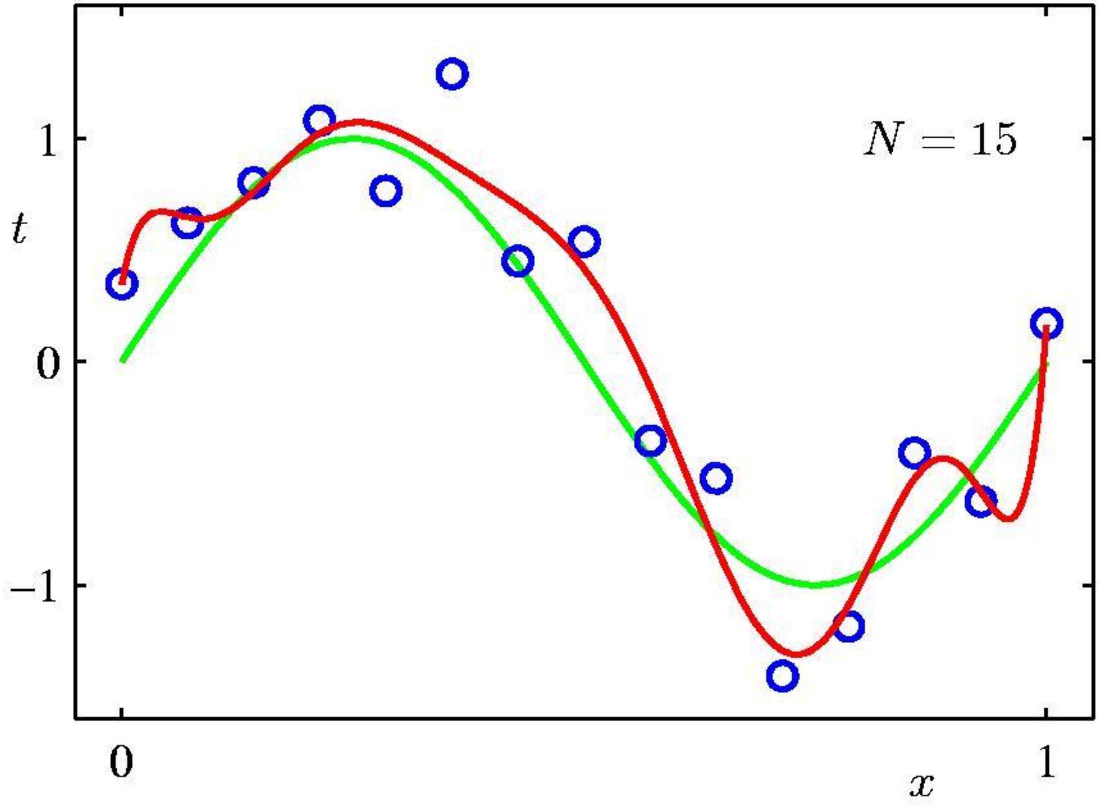
Overfitting,
complex model
magnitude of w is large

Increase N?

Larger data set?

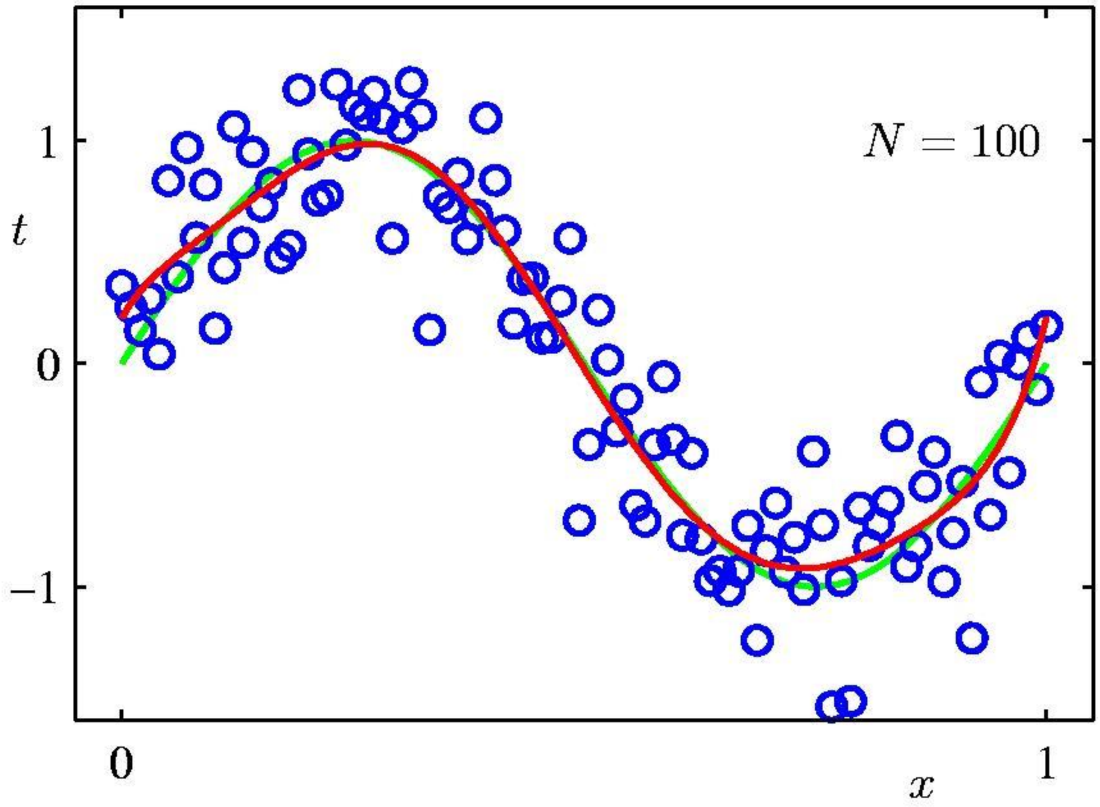
Data Set Size: $N = 15$

9th Order Polynomial



Data Set Size: $N = 100$

9th Order Polynomial



data set more complex

overfitting less severe

Rough heuristic:

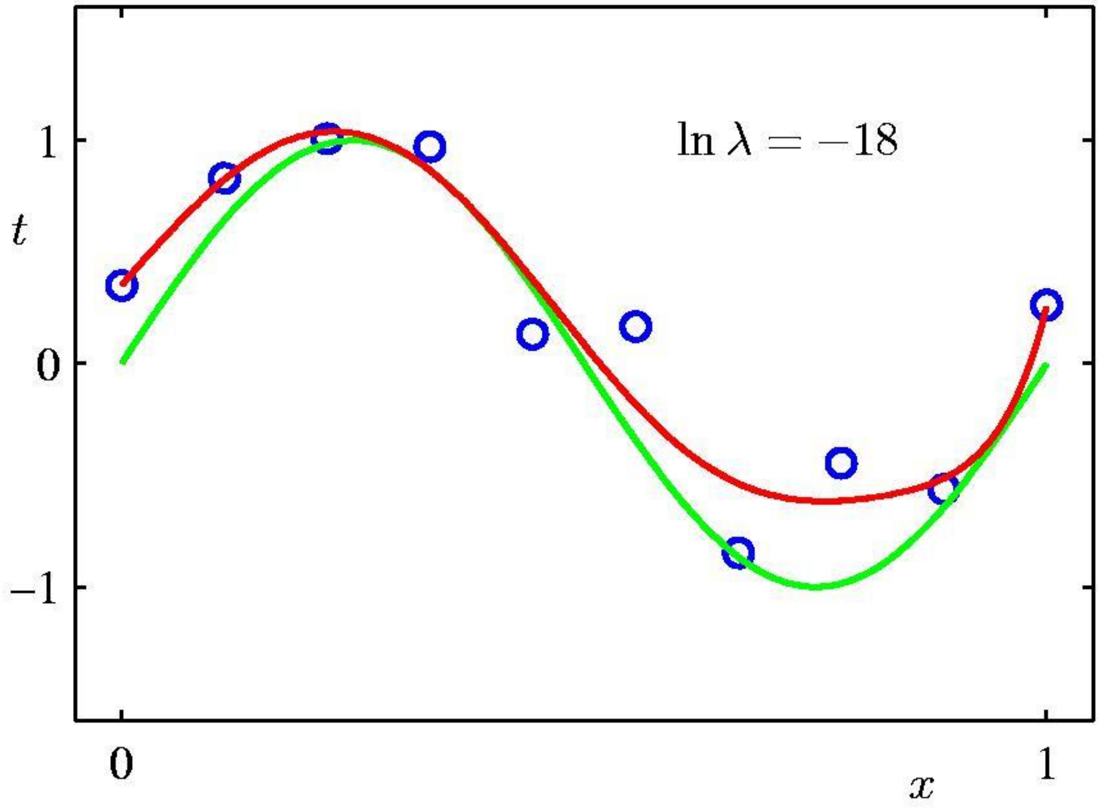
N > (5~10) * (number of parameters)

Regularization

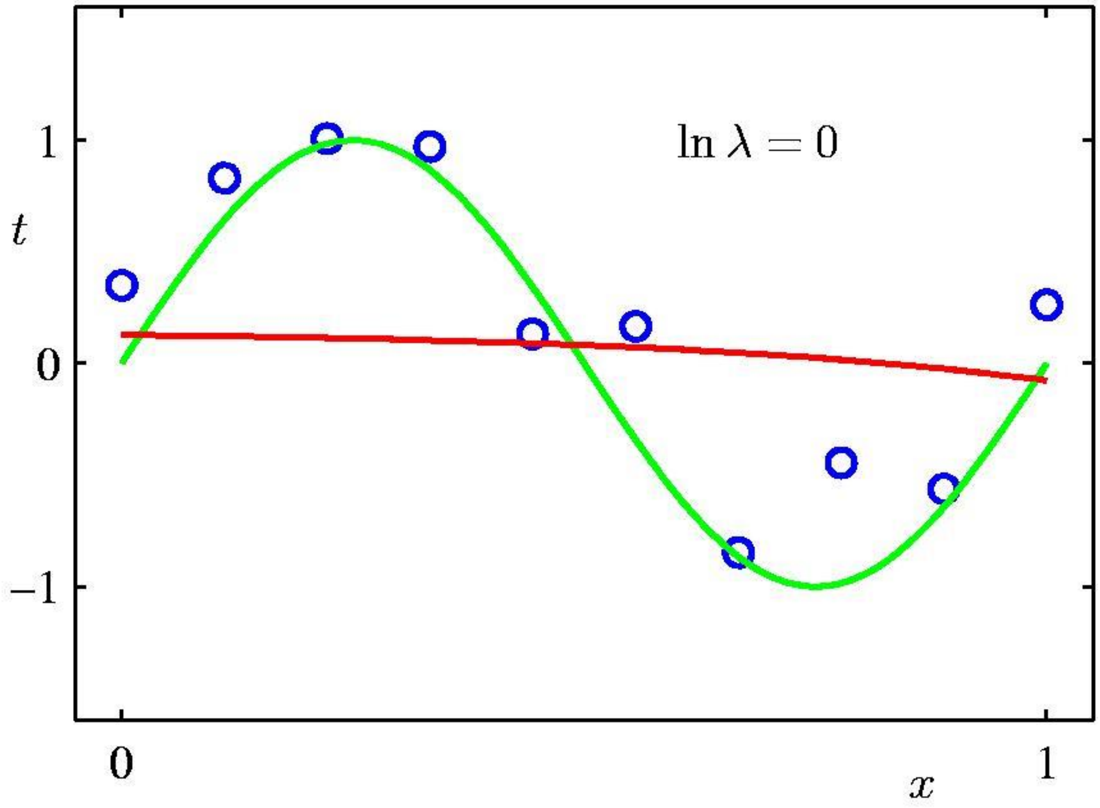
Penalize large coefficient values

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

Regularization: $\ln \lambda = -18$



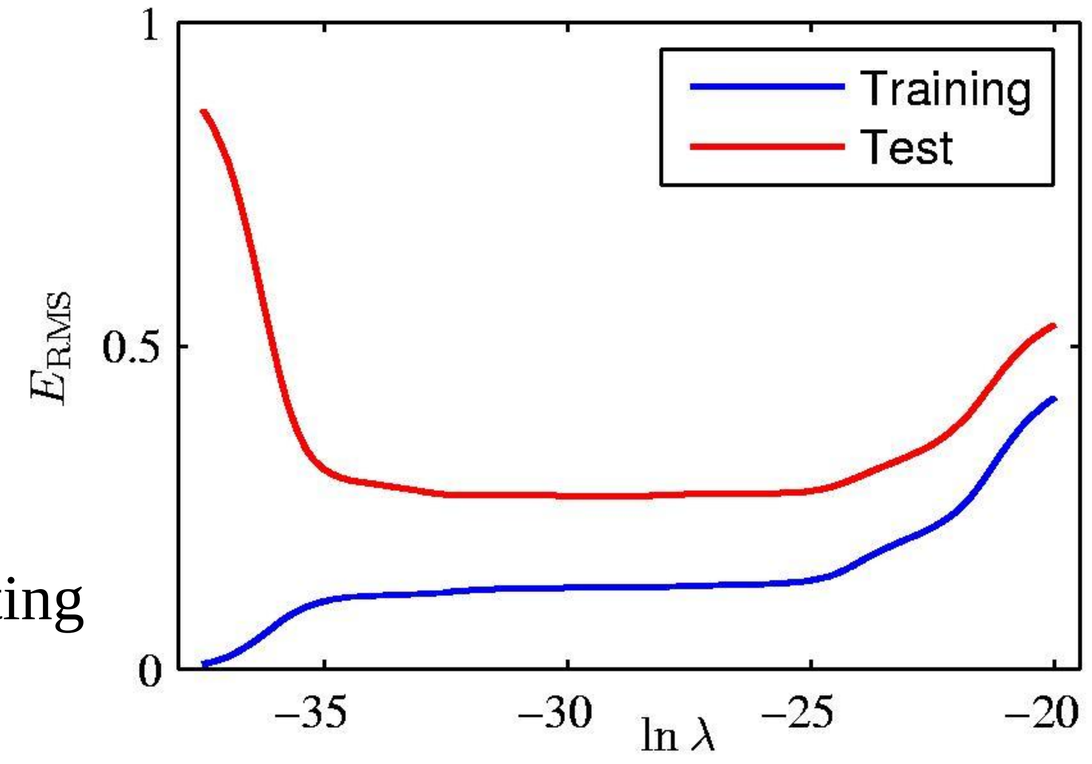
Regularization: $\ln \lambda = 0$



λ=1

Magnitude of coefficients is too much reduced

Regularization: E_{RMS} vs. $\ln \lambda$



Over-fitting

Magnitude of coefficients is too much reduced

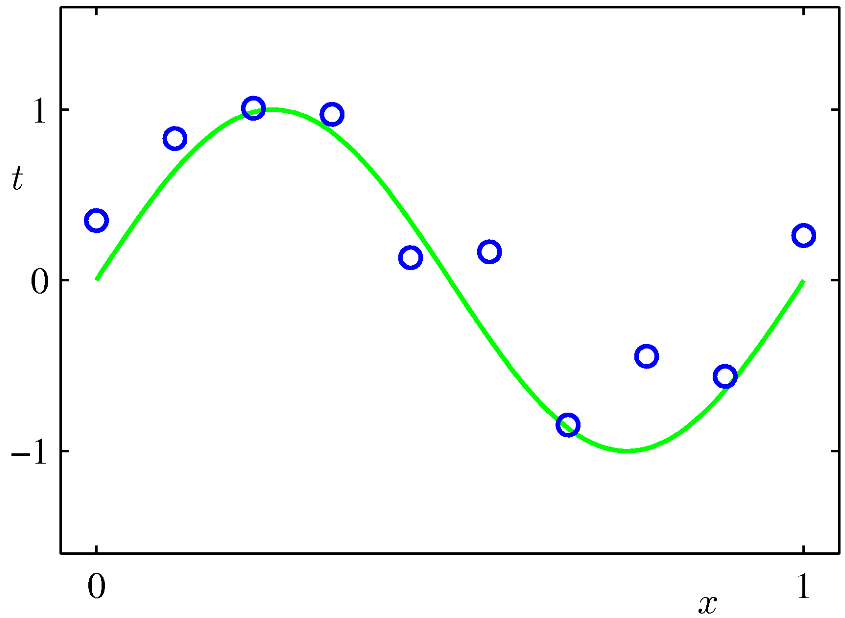
Polynomial Coefficients

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	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

Generalization

Linear Basis Function Models (1)

Example: Polynomial Curve Fitting



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

Linear Basis Function Models (2)

Generally

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

where $\Phi_j(x)$ are known as *basis functions*.

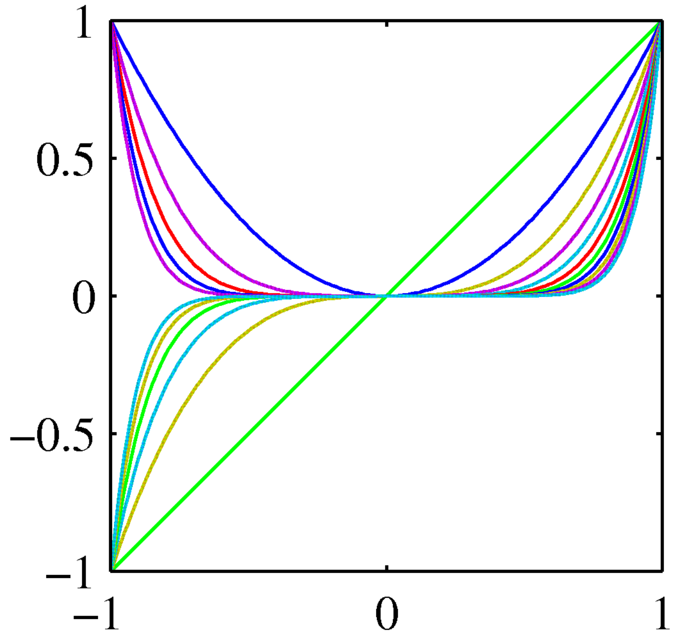
Typically, $\Phi_0(x) = 1$, so that w_0 acts as a bias.

Linear Basis Function Models (3)

Polynomial basis functions:

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

These are global; a small change in x affect all basis functions.

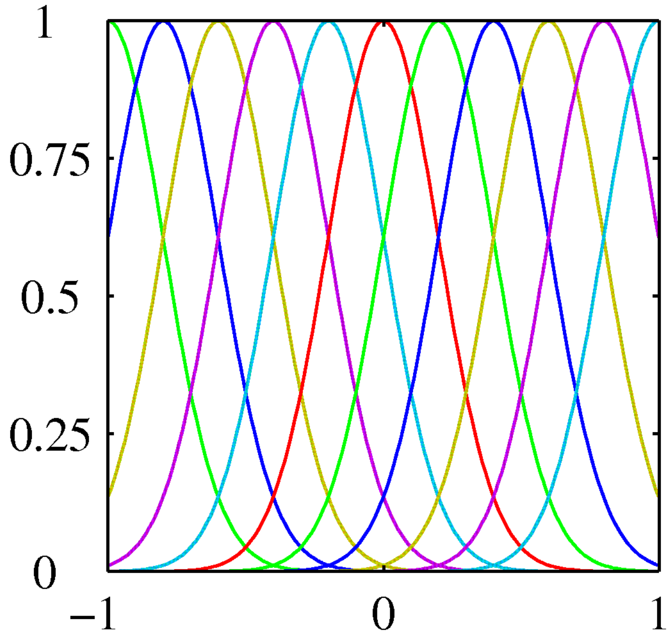


Linear Basis Function Models (4)

Gaussian basis functions:

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

These are local; a small change in x only affect nearby basis functions. μ_j and s control location and scale (width).



Linear Basis Function Models (5)

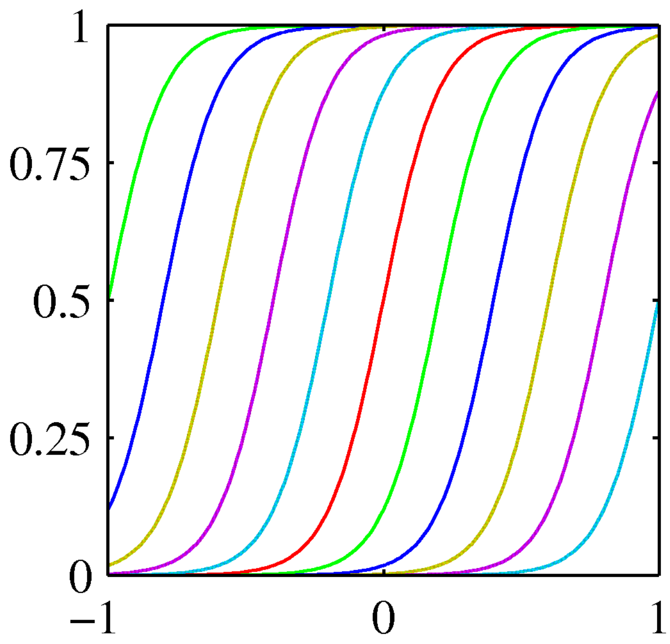
Sigmoidal basis functions:

$$\phi_j(x) = \sigma\left(\frac{x - \mu_j}{s}\right)$$

where

$$\sigma(a) = \frac{1}{1 + \exp(-a)}.$$

Also these are local; a small change in x only affect nearby basis functions. μ_j and s control location and scale (slope).

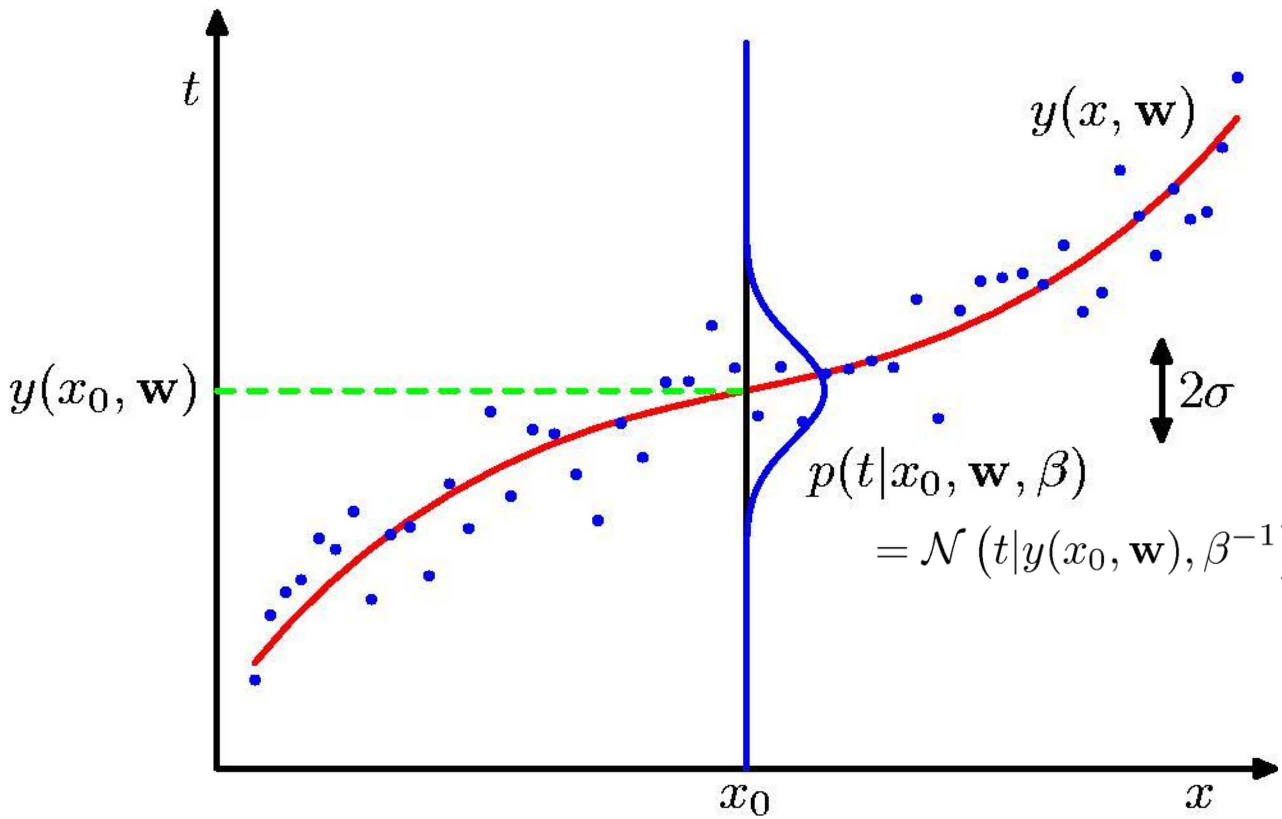


Maximize Likelihood Solution of w

Curve Fitting Re-visited

Assume observations from a deterministic function with added Gaussian noise:

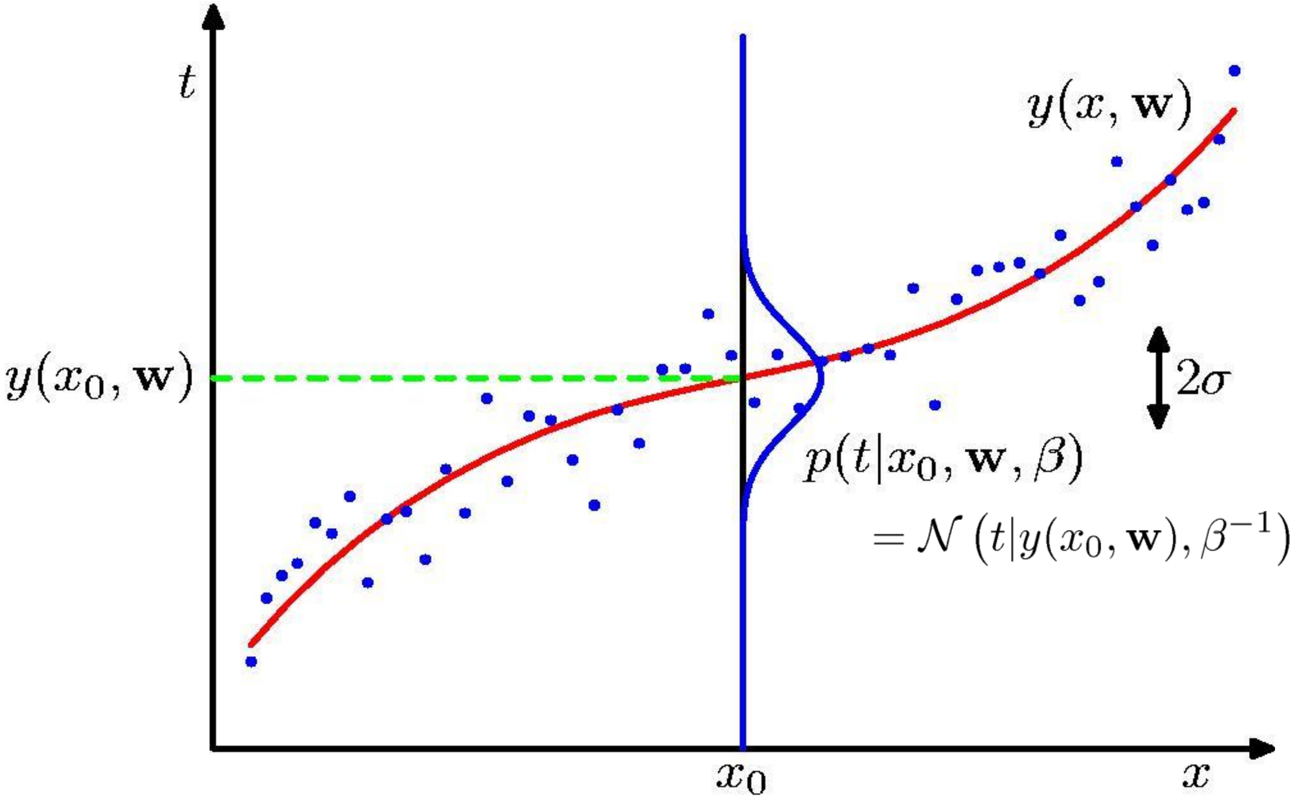
$t = y(\mathbf{x}, \mathbf{w}) + \epsilon$ where $p(\epsilon|\beta) = \mathcal{N}(\epsilon|0, \beta^{-1})$



Curve Fitting Re-visited

which is the same as saying, the conditional density of *t* given x, w, *β* is

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

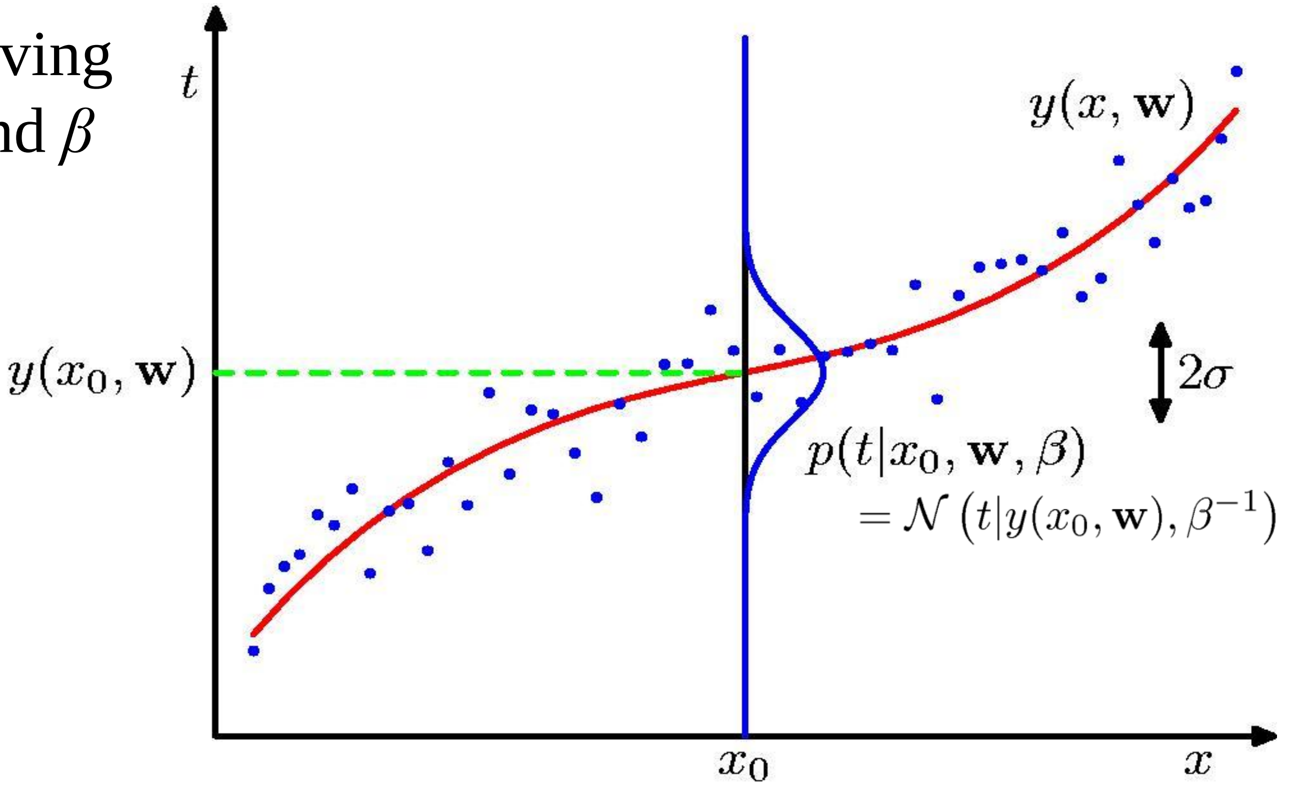


Curve Fitting Re-visited

Given observed inputs, $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and targets,
$\mathbf{t} = [t_1, \ldots, t_N]^{\mathrm{T}}$, we obtain the likelihood function

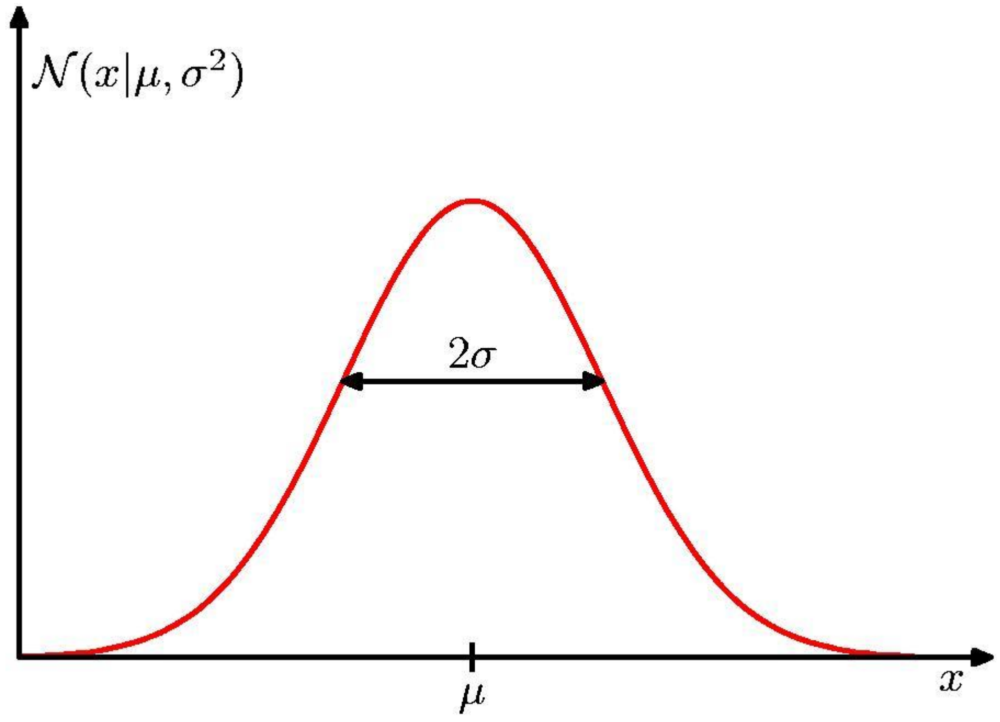
$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}).$$

the probability of observing outputs ***t*** given **X**, **w** and β



The Gaussian Distribution

$$\mathcal{N}\left(x|\mu,\sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



Maximum Likelihood and Least Squares (1)

Taking the logarithm, we get

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

where

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

is the sum-of-squares error.

$$\operatorname*{argmax}_{w} L = \operatorname*{argmin}_{w} E$$

Maximum Likelihood and Least Squares (2)

Computing the gradient and setting it to zero yields

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t}|\mathbf{w}, \beta) = \beta \sum_{n=1}^{N} \left\{ t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} = \mathbf{0}.$$

Solving for w, we get

$$\mathbf{w}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t}$$

The Moore-Penrose pseudo-inverse, $\boldsymbol{\Phi}^{\dagger}$.

where

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

See a demo

Maximum Likelihood and Least Squares (3)

Maximizing with respect to the bias, w_0, alone, we see that

$$w_0 = \bar{t} - \sum_{j=1}^{M-1} w_j \overline{\phi_j}$$

$$= \frac{1}{N}\sum_{n=1}^{N} t_n - \sum_{j=1}^{M-1} w_j \frac{1}{N}\sum_{n=1}^{N} \phi_j(\mathbf{x}_n).$$

averages (over the training set) of the target values

weighted sum of the averages of the basis function values

We can also maximize with respect to β, giving

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

residual variance of the target values around the regression function

Regularization

Regularized Least Squares (1)

Consider the error function:

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Data term + Regularization term

λ is called the regularization coefficient.

With the sum-of-squares error function and a quadratic regularizer, we get

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

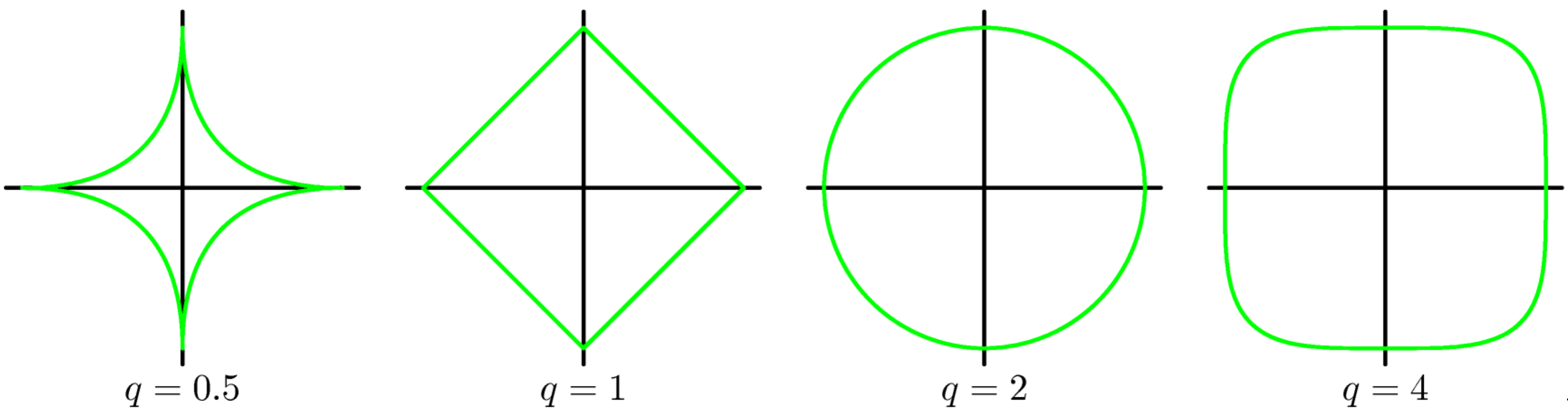
Regularized Least Squares (2)

With a more general regularizer, we have

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

Minimizing $E_D(\mathrm{w})+\lambda E_{\mathrm{w}}(\mathrm{w})$ is equivalent to minimizing $E_D(\mathrm{w})$ subject to the constraint

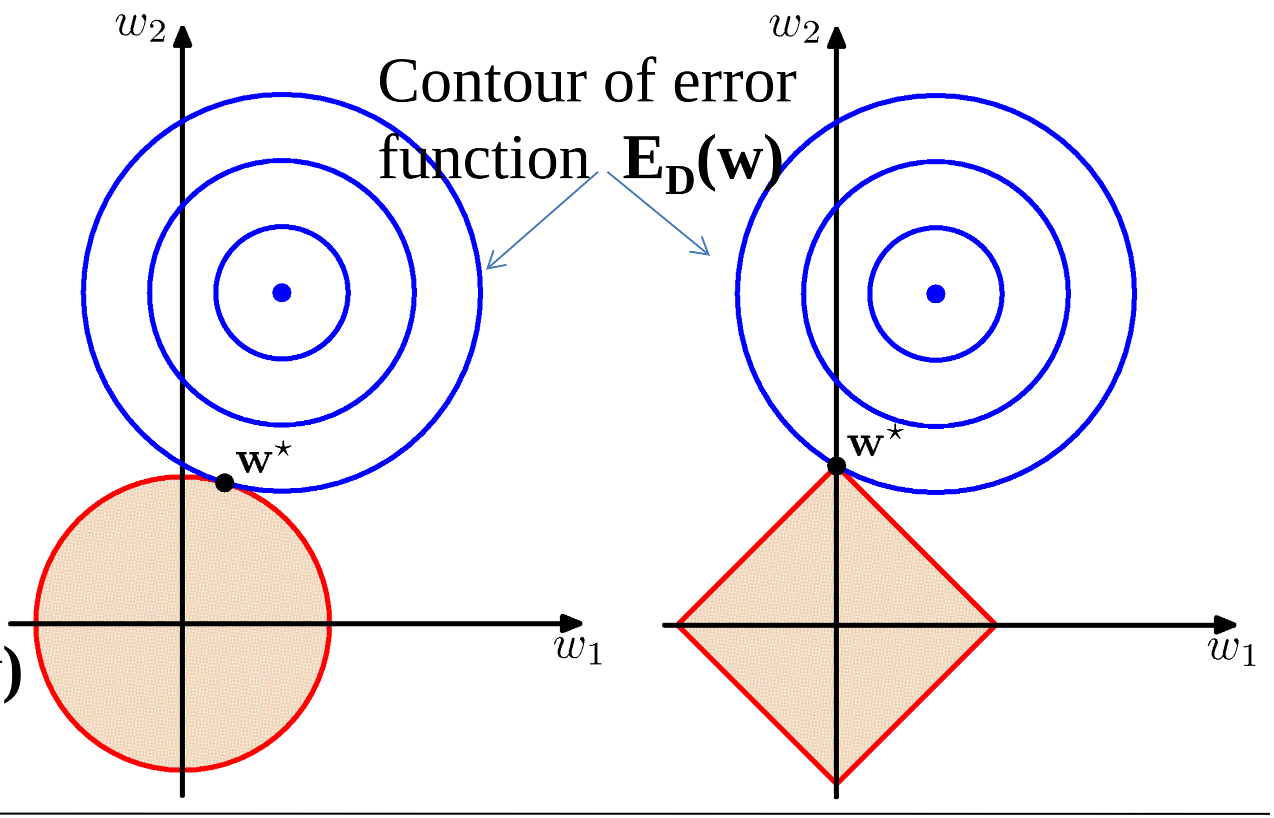
$$\sum_{j=1}^{M} \| w_j \|^q \leq \eta$$



Lasso Quadratic

Regularized Least Squares (3)

Lasso tends to generate sparser solutions than a quadratic regularizer.



See a demo at: http://www.lri.fr/~xlzhang/KAUST/CS229_slides/regularization_demo.m

Questions to ask in next class

- What is the task of regression?
- How to solve the regression problem?
- How to minimize the error function?
- The reason of overfitting?
- Maximum likelihood vs least square, same?
- How can the regularization term help the regression model?