7. Linear Regression 2
STA3142 Statistical Machine Learning

Kibok Lee

Assistant Professor of

Applied Statistics / Statistics and Data Science

Mar {19, 21}, 2024

Assignment 1

- Due **Friday 3/29, 11:59pm**
- Topics
  - (Programming) NumPy basics
  - (Programming) Linear regression on a polynomial
  - (Math) Derivation and proof for linear regression

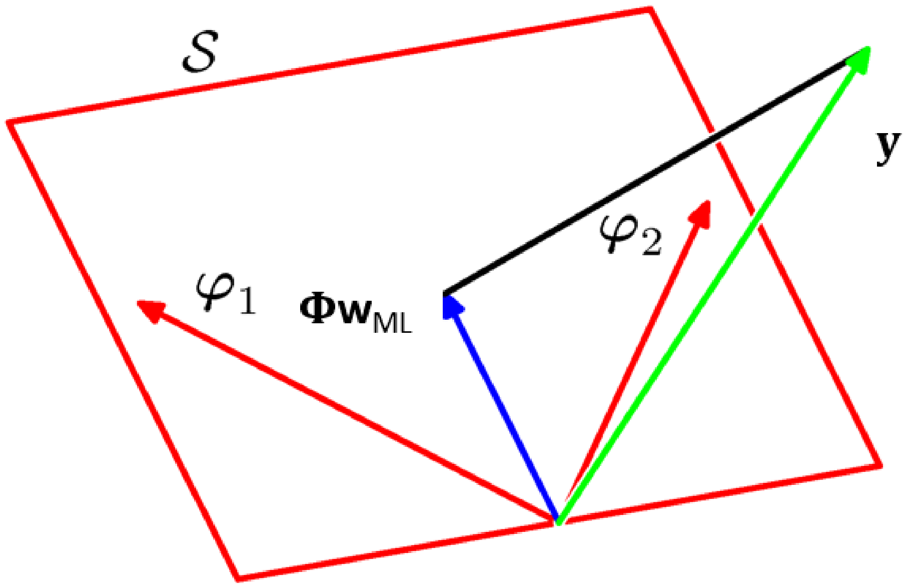
- Please read the instruction carefully!
  - Submit one <u>pdf</u> and one <u>zip</u> file separately
  - Write your code only in the designated spaces
  - Do not import additional libraries
  - …
- If you feel difficult, consider to take **option 2**.

Outline

- Uniqueness of Least-Squares Solution
  - Geometrical Interpretation
- Overfitting
- Regularized Linear Regression
- Maximum Likelihood Interpretation
  - Review on Probability
- Locally-Weighted Linear Regression

Geometrical Interpretation

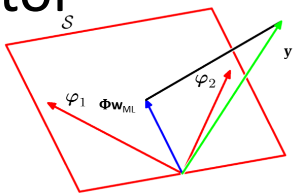
- Assuming many more observations (N) than the M basis functions $\phi_j(x)$ (j=0,…,M-1)
- View the observed target values $\mathbf{y} = \{y^{(1)}, \dots, y^{(N)}\}$ as a vector in an N-dim. space.
- The M basis functions $\phi_j(x)$ span the N-dimensional subspace.
  - Where the N-dim vector for ϕ_j is $\{\phi_j(\mathbf{x}^{(1)}), \dots, \phi_j(\mathbf{x}^{(N)})\}$
- $\mathbf{\Phi w}_{\mathrm{ML}}$ is the point in the subspace with minimal squared error from $\mathbf{y}$.
- It's the projection of $\mathbf{y}$ onto that subspace.



Slide credit: Ben Kuipers

Uniqueness of Least-Squares Solution

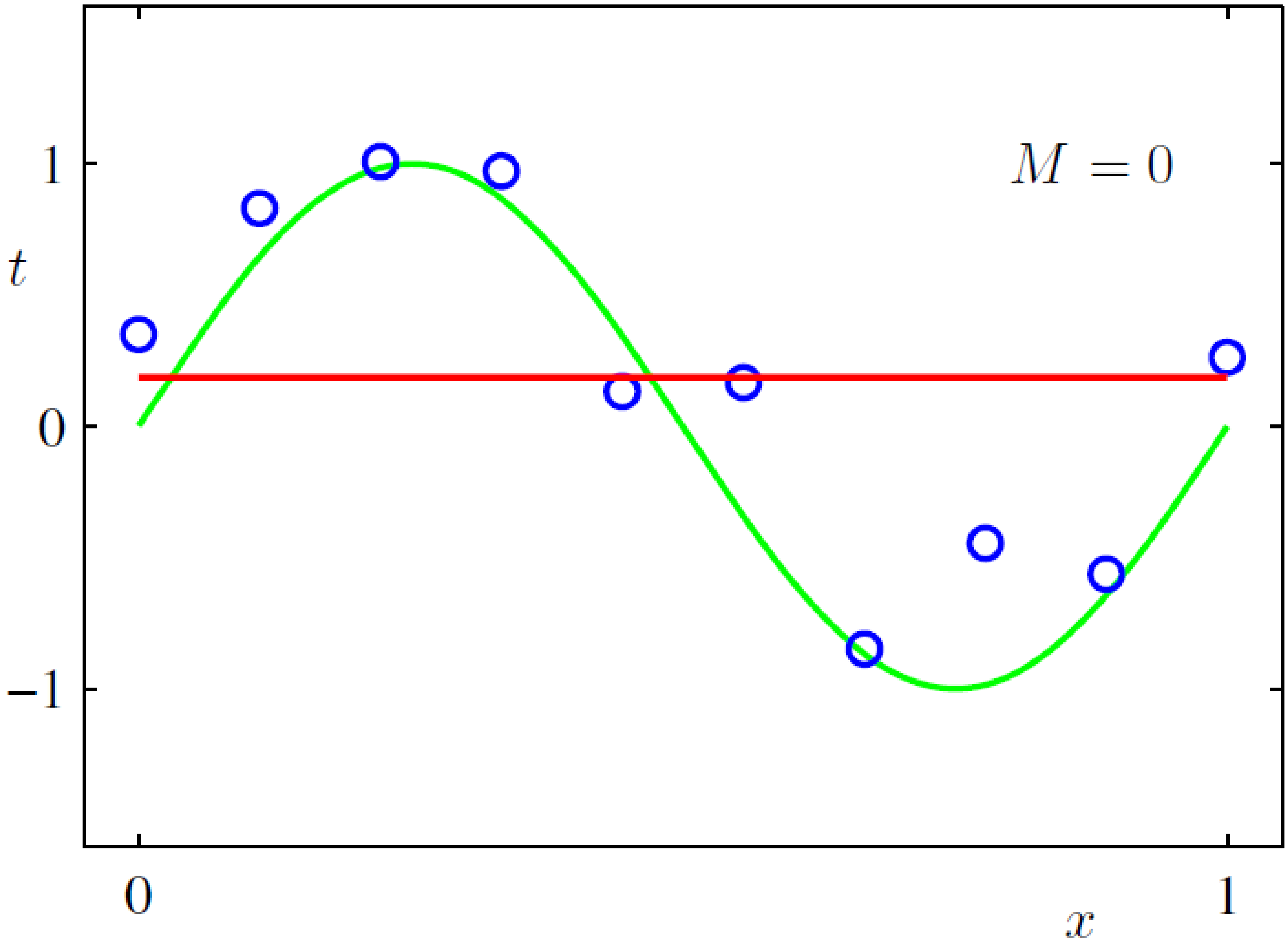
- For $\Phi \in \mathbb{R}^{N \times M}$, least squares finds $\mathbf{w}$ satisfying
$$\Phi\mathbf{w} \simeq \mathbf{y}$$
- When $N \geq M$ (overdetermined system) and $\operatorname{rank}(\Phi) = M$, least-squares solution is unique.
  - The orthogonal projection of the ground-truth vector onto the subspace spanned by the basis functions.
$$\mathbf{w} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y}$$



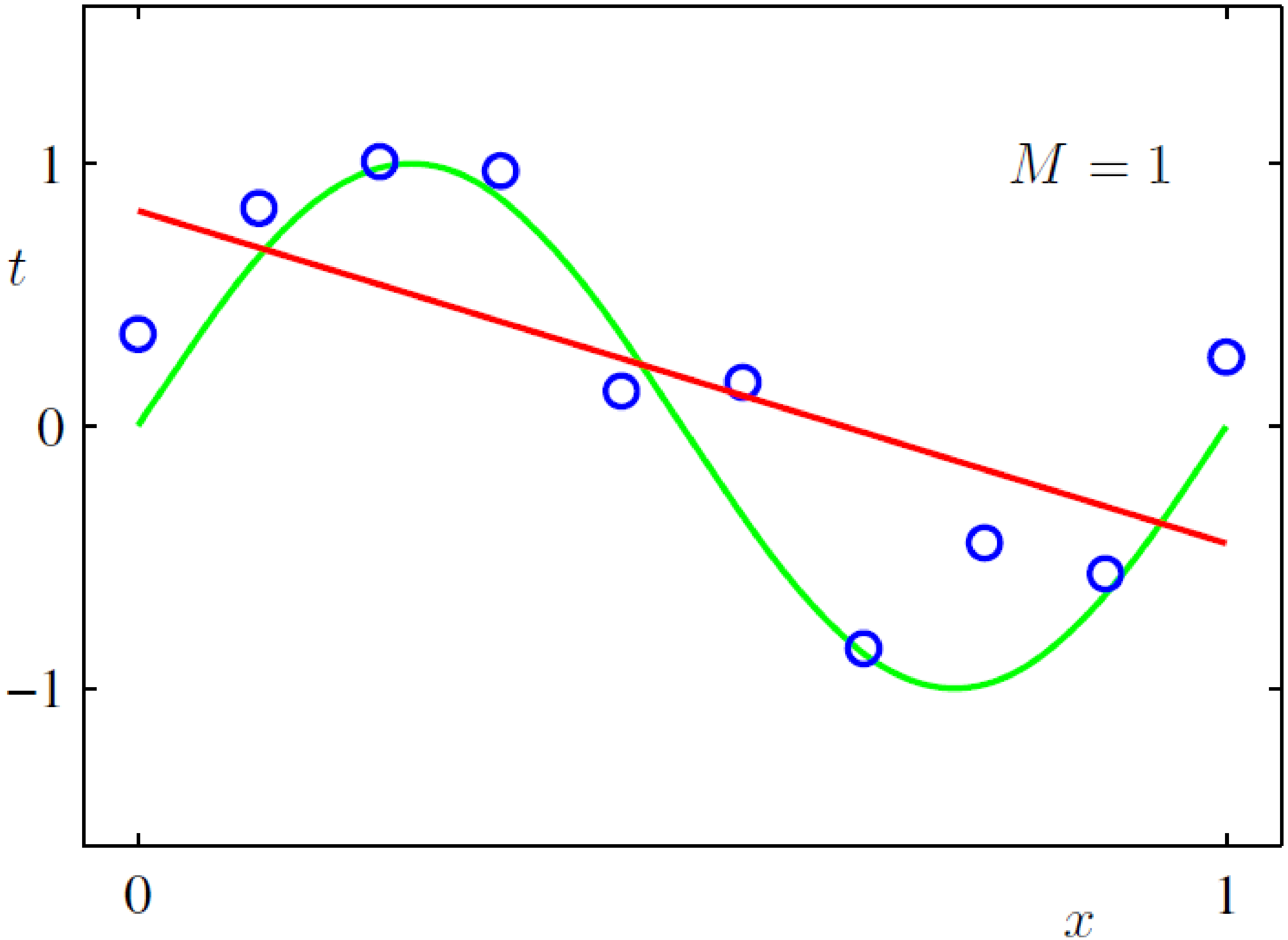
- Cf. When $N < M$ (underdetermined system), least-squares solution is not unique, i.e., there are infinite number of solutions:
$$\mathbf{w} = \Phi^T(\Phi\Phi^T)^{-1}\mathbf{y} + \xi \ \text{ where } \ \xi \in \operatorname{null}(\Phi)$$
(when $\operatorname{rank}(\Phi) = N$)

Overfitting

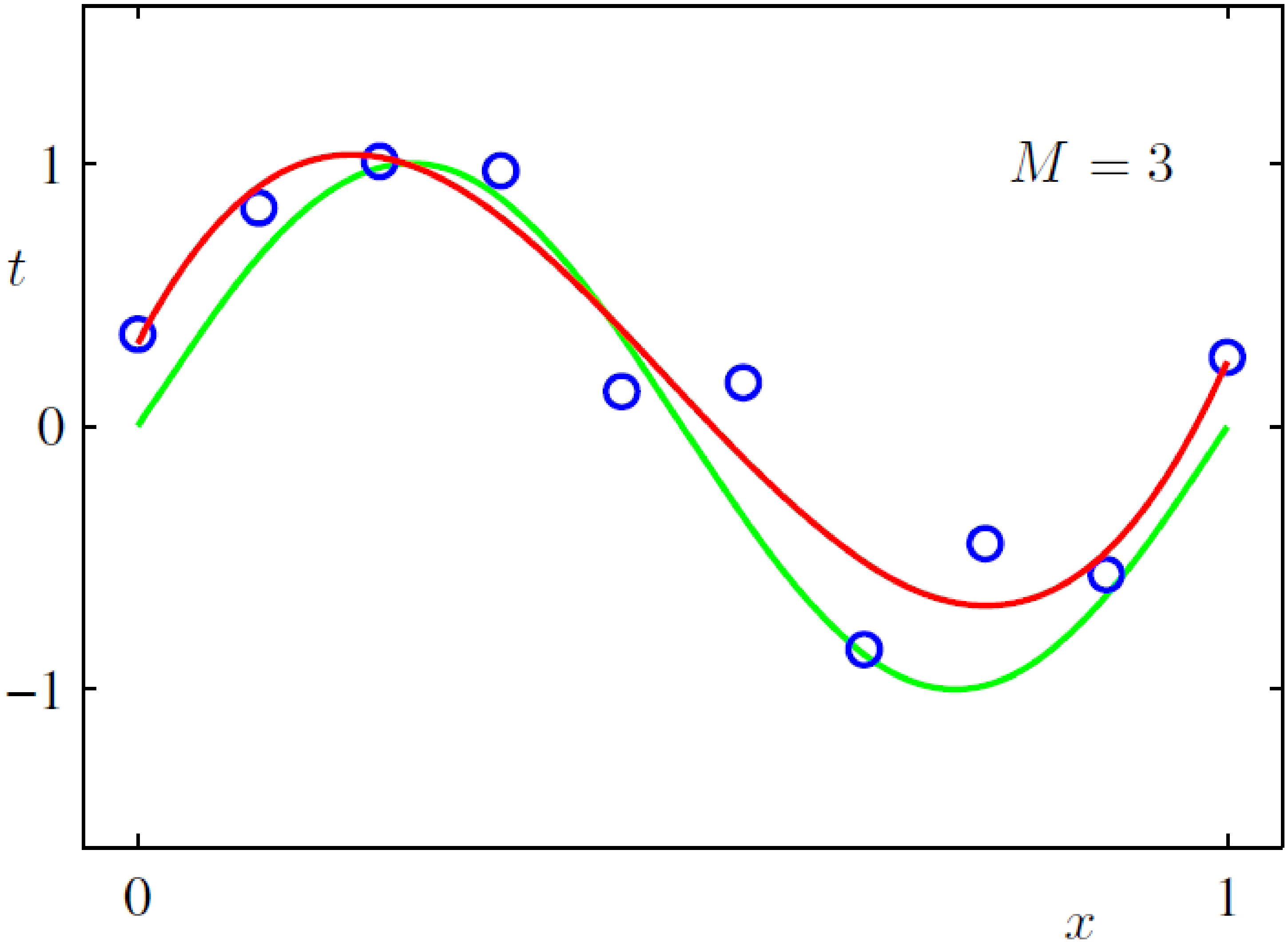
0^{th} Order Polynomial



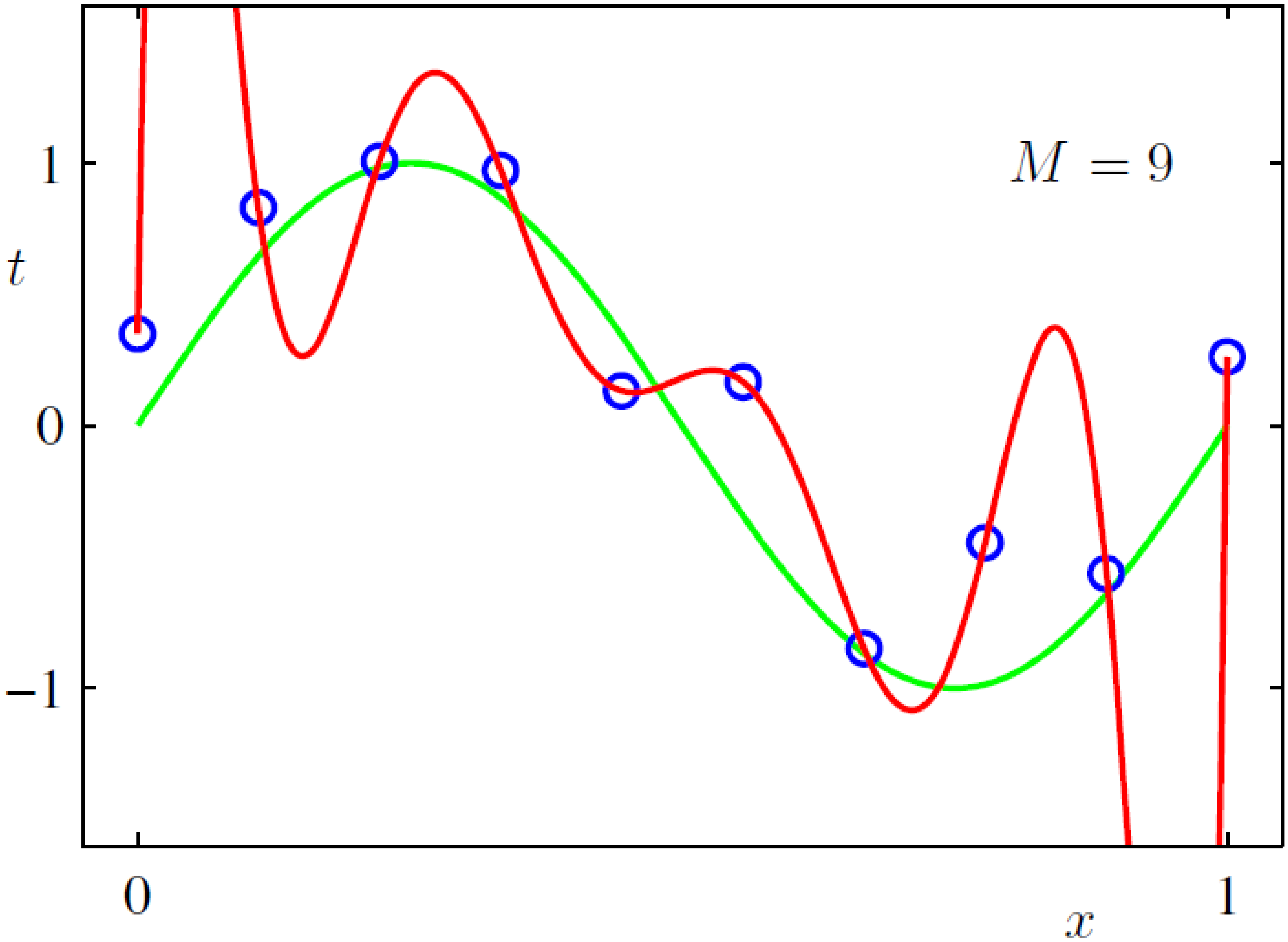
1st Order Polynomial



3^{rd} Order Polynomial

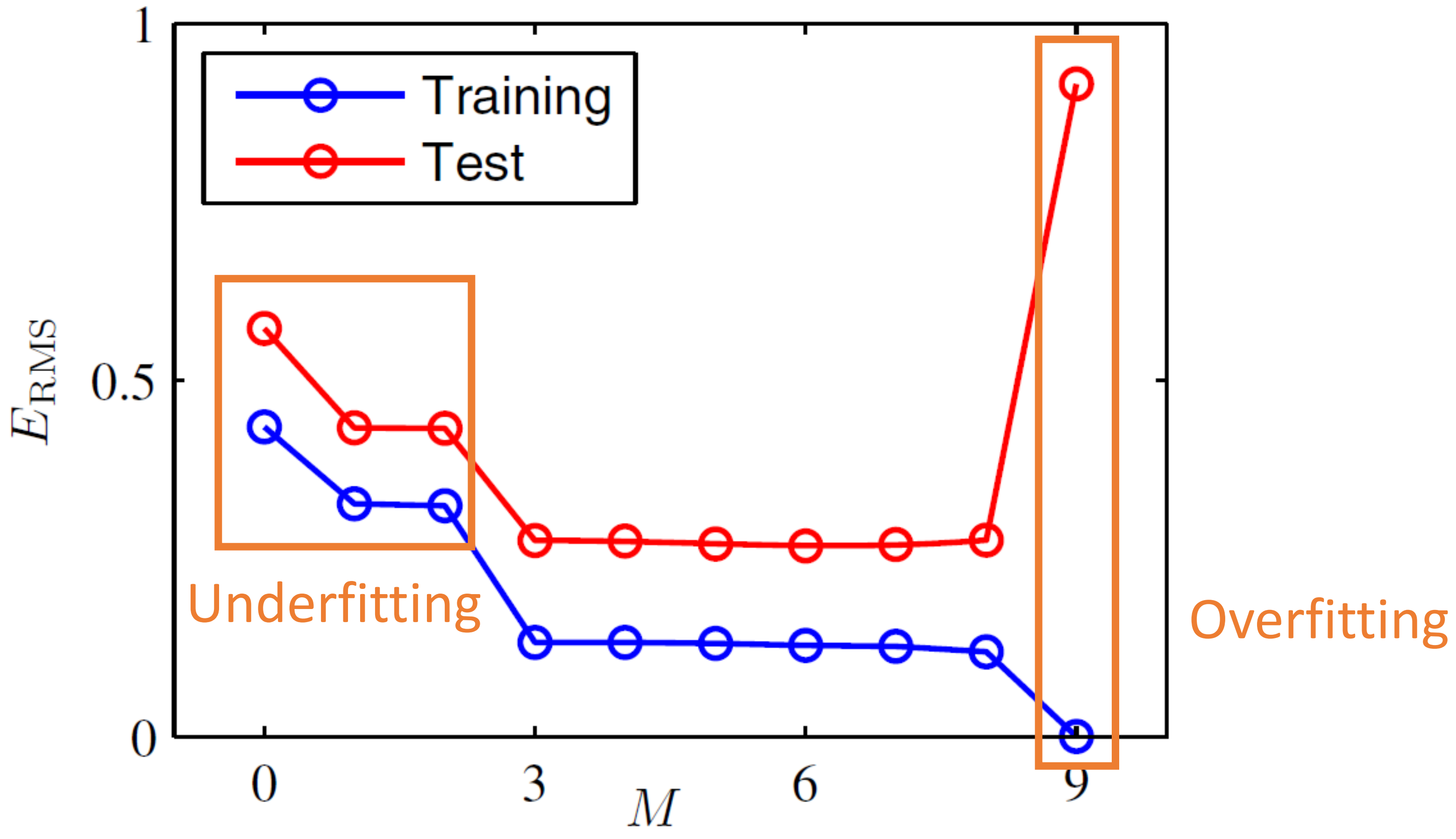


9th Order Polynomial



Overfitting

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

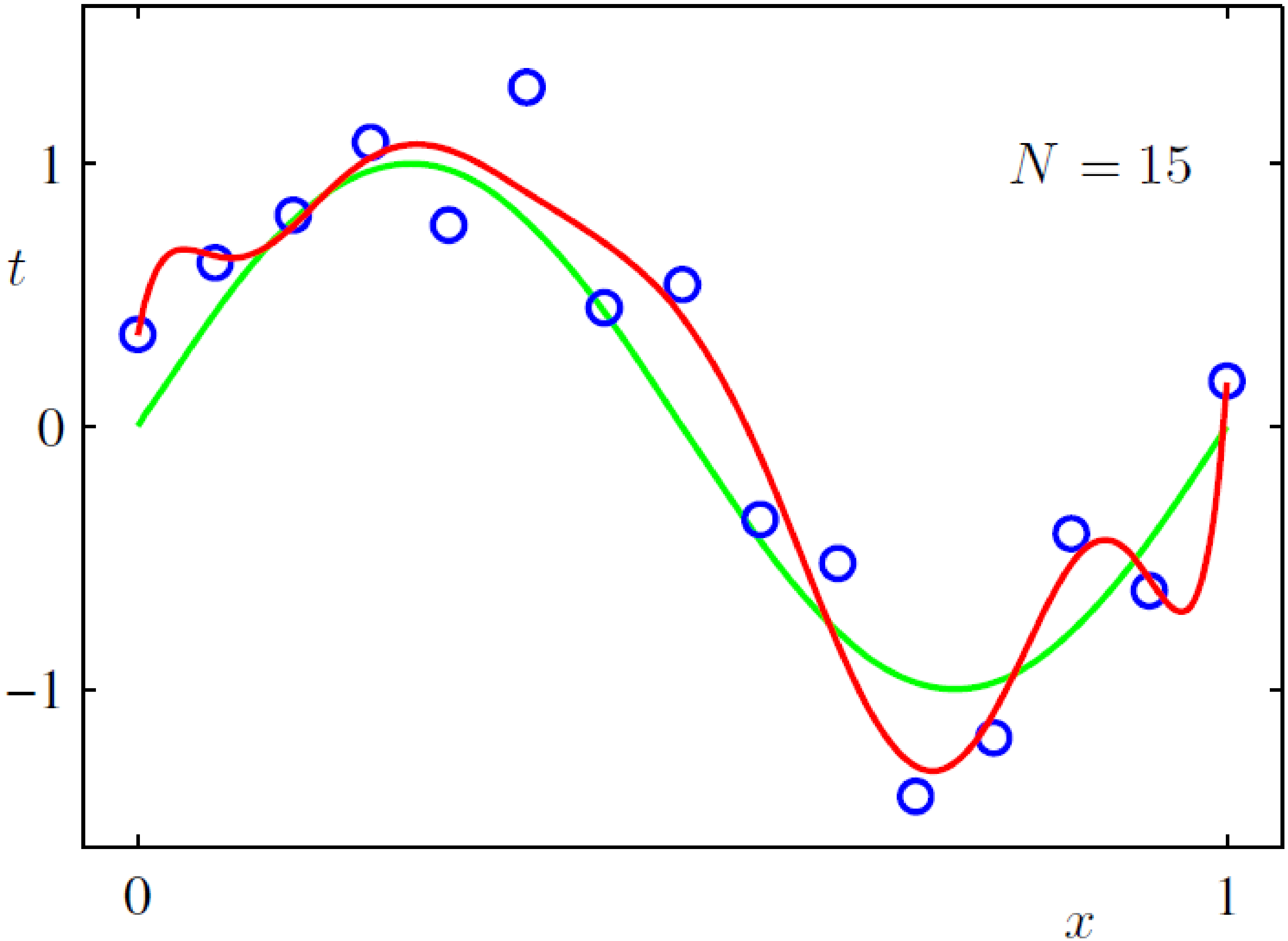


Polynomial Coefficients

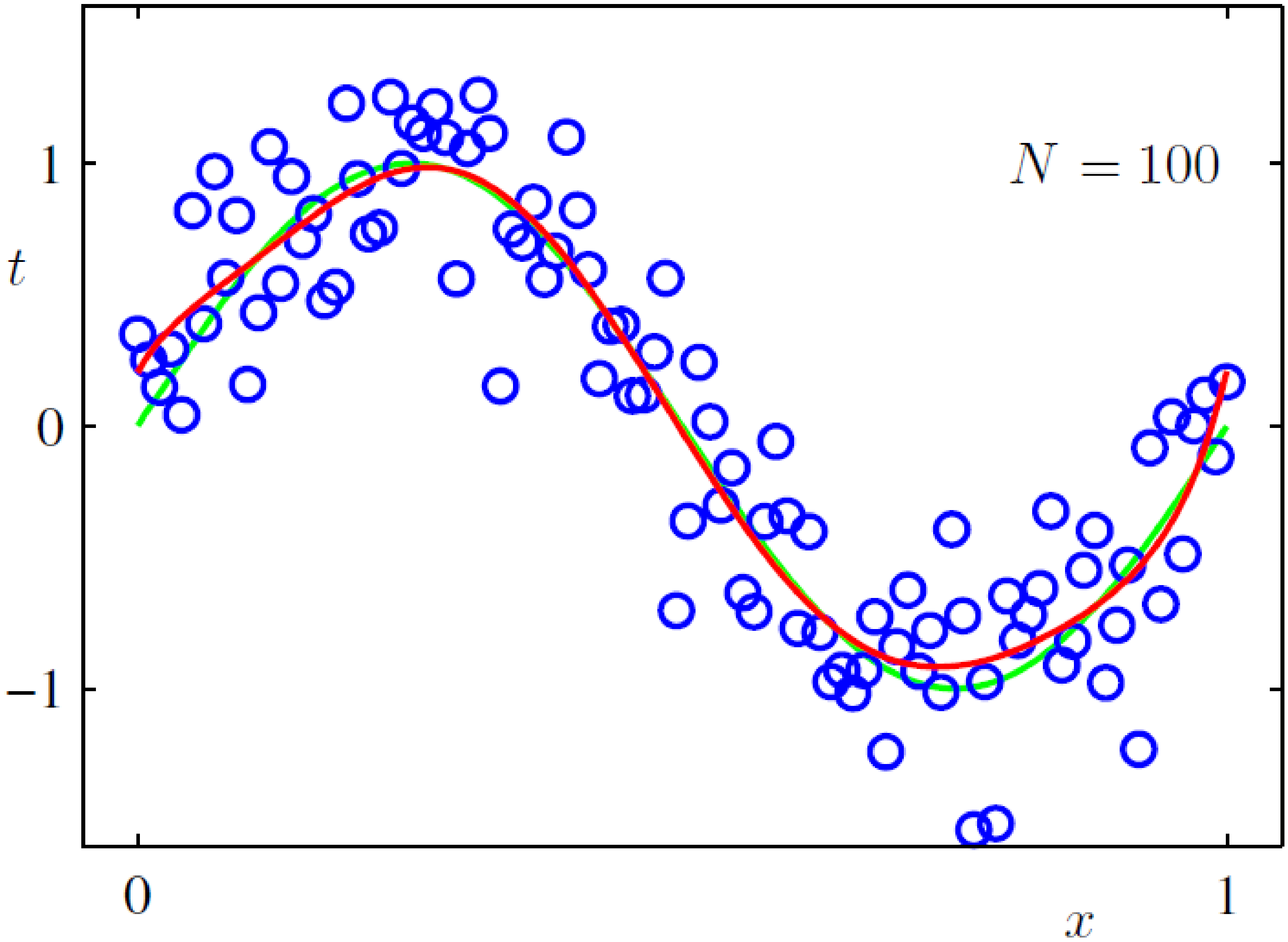
- When M is large, the scale of $\mathbf{w}$ tends to be large
  - Even a small change of $\mathbf{x}$ results in a large change on the output; leading overfitting

	$M = 0$	$M = 1$	$M = 3$	$M = 9$
$w_0^\star$	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

9th Order Polynomial, 15 data



9^{th} Order Polynomial, 100 data



How to Avoid Overfitting

- Increasing dataset size N
  - Collecting a large training dataset is expensive
  - Optimization takes a long time

- Finding an appropriate degree M
  - How?

How to Choose the Degree of Polynomial

- If you have a small number of data, then use low order polynomial.
  - Small number of features
  - Otherwise, your model will overfit.
- As you obtain more data, you can gradually increase the order of the polynomial.
  - Large number of features
  - Still limited by the finite amount of the data available (i.e., the optimal model for finite data cannot be infinite dimensional polynomial)
- Controlling model complexity by **regularization**

Regularized Linear Regression

Regularized Least Squares

- 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 (a.k.a. ridge or L2) regularizer, we get

Penalize large $\mathbf{w}$ values

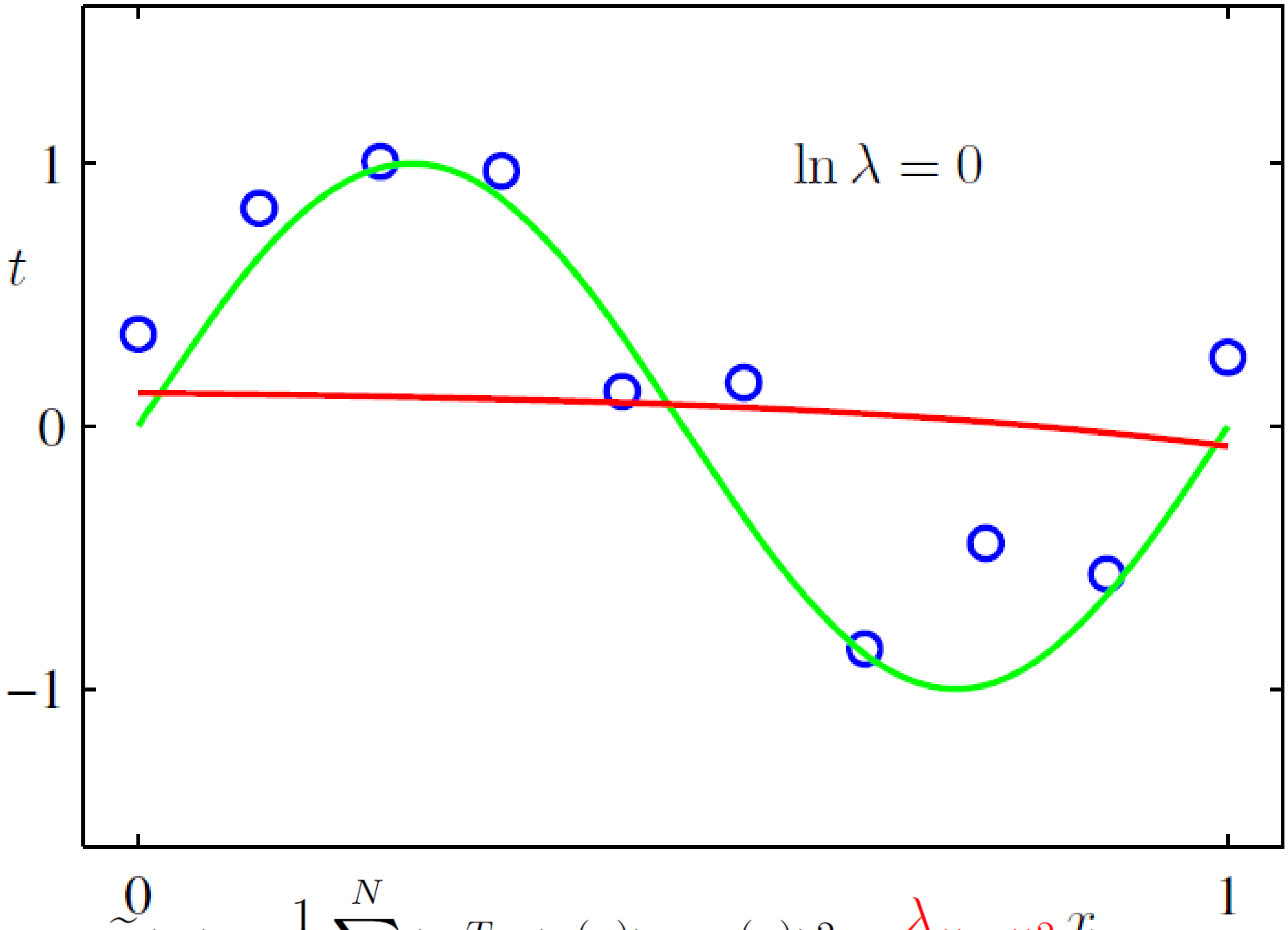
$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\phi(\mathbf{x}^{(n)}) - y^{(n)})^2 + \frac{\lambda}{2}||\mathbf{w}||^2$$

New objective function

Definition (L2): $||\mathbf{w}||_2^2 = \sum_{j=0}^{M-1} w_j^2$

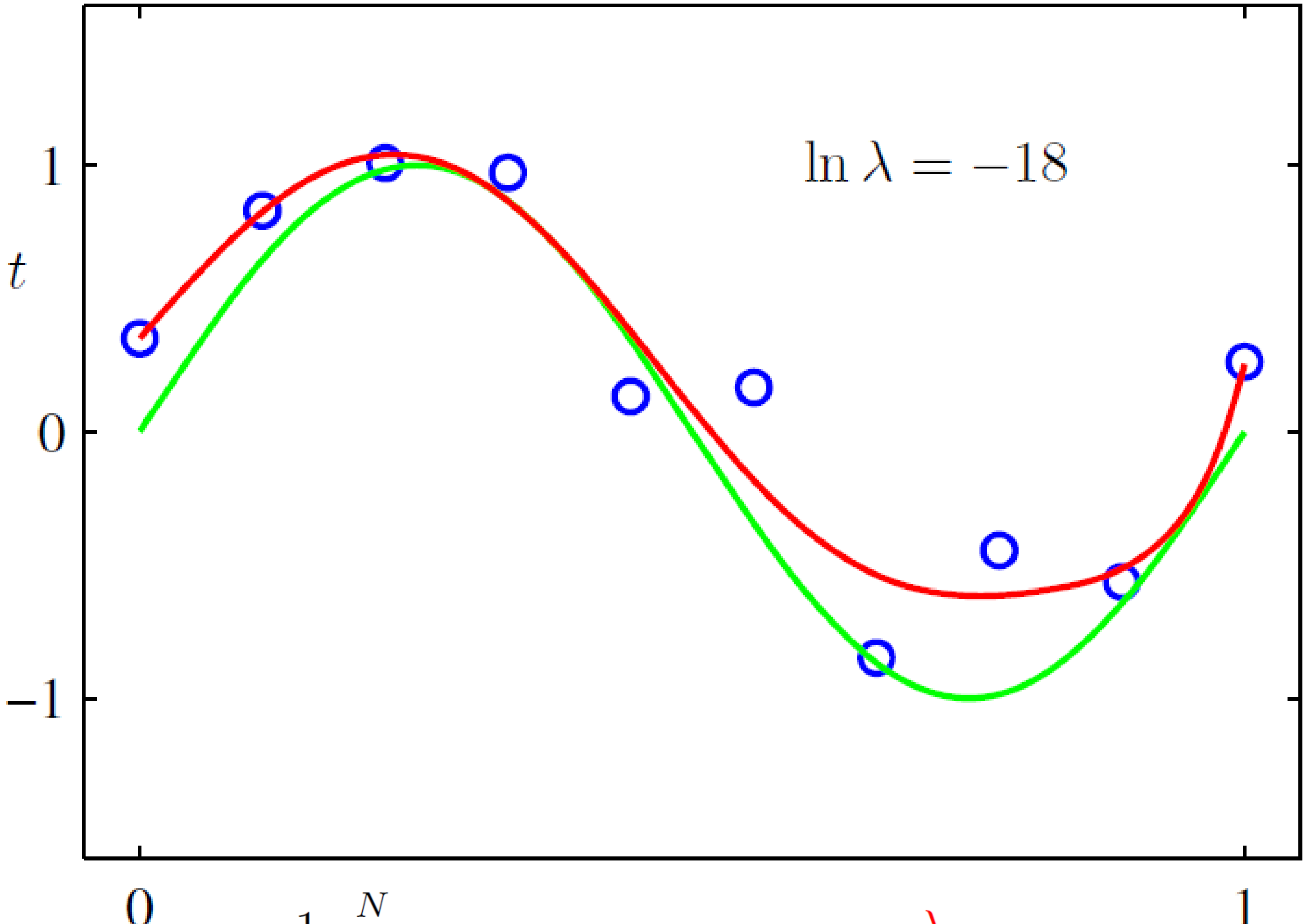
- Effect of λ?

L2 Regularization when $\log \lambda = 0$



$M = 9$ $\quad \widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\phi(\mathbf{x}^{(n)}) - y^{(n)})^2 + \frac{\lambda}{2}||\mathbf{w}||^2$

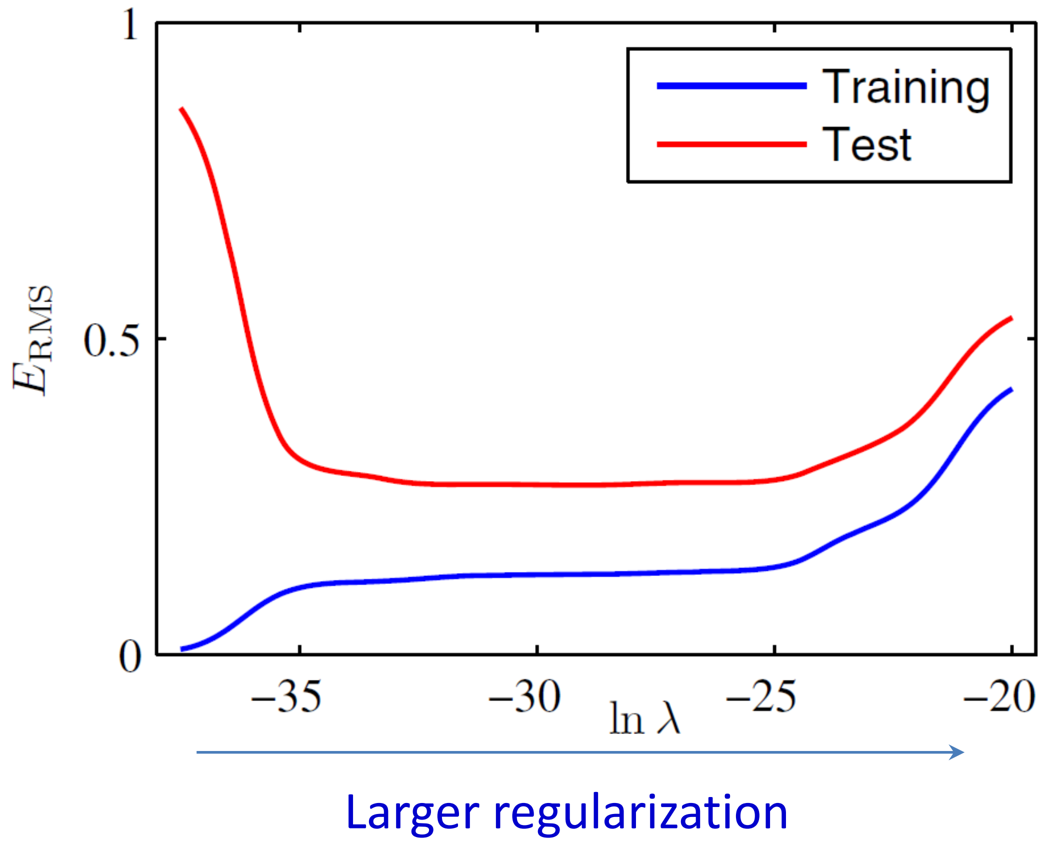
L2 Regularization when $\log \lambda = -18$



$M = 9$ $\quad \widetilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} (\mathbf{w}^T \phi(\mathbf{x}^{(n)}) - y^{(n)})^2 + \frac{\lambda}{2} ||\mathbf{w}||^2$

L2 Regularization: E_{RMS} vs. λ

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



NOTE: For simplicity of presentation, we divided the data into training set and test set. However, it's **<u>not</u>** legitimate to find the optimal hyperparameter based on the test set. We will talk about legitimate ways of doing this when we cover model selection and validation.

Polynomial Coefficients

- With an appropriate λ, we can avoid overfitting

	Overfitting	Sweet spot	Underfitting
	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	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

Regularized Least Squares

- 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 (a.k.a. ridge or L2) regularizer, we get

Penalize large $\mathbf{w}$ values

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\phi(\mathbf{x}^{(n)}) - y^{(n)})^2 + \frac{\lambda}{2}||\mathbf{w}||^2$$

- Closed-form solution:

$$\mathbf{w}_{ML} = (\lambda\mathbf{I} + \Phi^T\Phi)^{-1}\Phi^T\mathbf{y}$$

Derivation

Objective function

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\phi(\mathbf{x}^{(n)}) - y^{(n)})^2 + \frac{\lambda}{2}||\mathbf{w}||^2$$

$$= \frac{1}{2}\mathbf{w}^T\Phi^T\Phi\mathbf{w} - \mathbf{w}^T\Phi^T\mathbf{y} + \frac{1}{2}\mathbf{y}^T\mathbf{y} + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

Compute gradient and set it zero:

$$\begin{aligned}\nabla_{\mathbf{w}}E(\mathbf{w}) &= \nabla_{\mathbf{w}}[\frac{1}{2}\mathbf{w}^T\Phi^T\Phi\mathbf{w} - \mathbf{w}^T\Phi^T\mathbf{y} + \frac{1}{2}\mathbf{y}^T\mathbf{y} + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}] \\ &= \Phi^T\Phi\mathbf{w} - \Phi^T\mathbf{y} + \lambda\mathbf{w} \\ &= (\lambda\mathbf{I} + \Phi^T\Phi)\mathbf{w} - \Phi^T\mathbf{y} \\ &= 0\end{aligned}$$

$$\mathbf{w}_{ML} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{y}$$

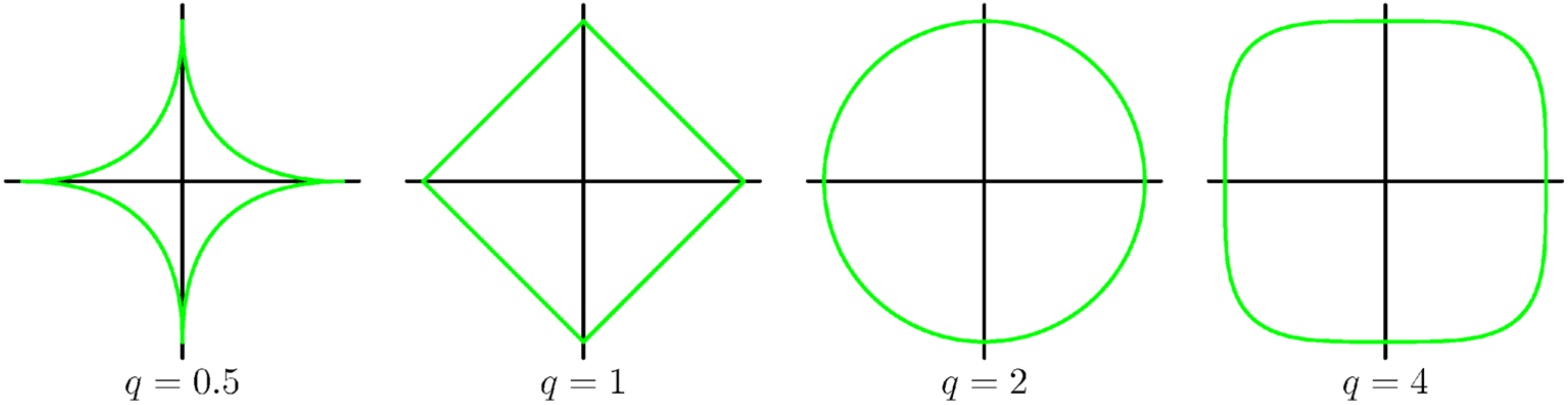
Cf. Ordinary Least Squares

Therefore, we get: $\mathbf{w}_{ML} = (\lambda\mathbf{I} + \Phi^T\Phi)^{-1}\Phi^T\mathbf{y}$

Regularized Least Squares

- With a more general regularizer, we have

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



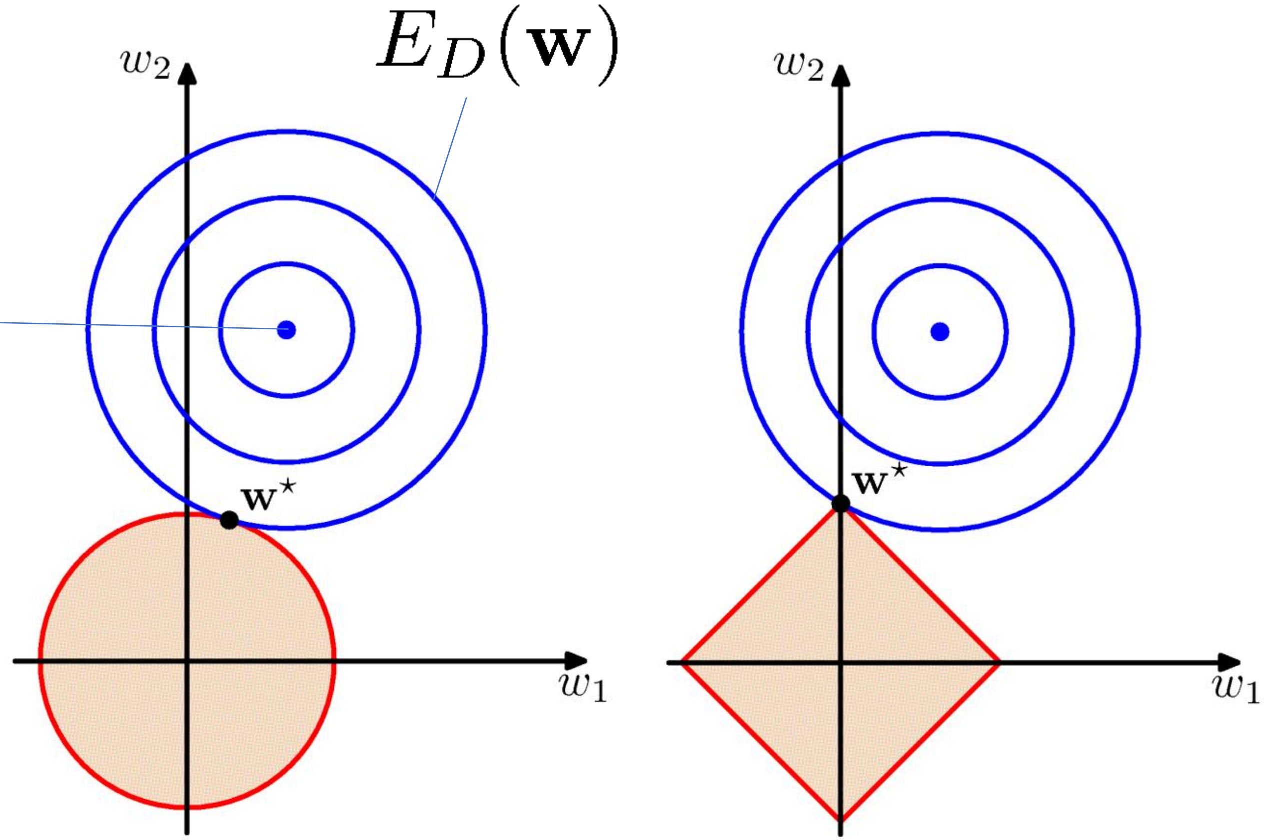
Lasso/L1 regularization

Quadratic/Ridge/L2 regularization

Regularized Least Squares

- Lasso tends to generate sparser solutions than ridge regularization.

Ordinary least square solution



Regularization as constraints:

$$\sum_j w_j^2 \leq C \qquad \sum_j |w_j| \leq C$$

Summary: Regularized Linear Regression

- Simple modification of linear regression
- Regularization controls the tradeoff between "fitting error" and "complexity."
  - Small regularization results in complex models (with risk of overfitting)
  - Large regularization results in simple models (with risk of underfitting)

- It is important to find an optimal regularization that balances between the two.

Review on Probability

Probability: Terminology

- Experiment: Procedure that yields an outcome
  - E.g., Tossing a coin three times:
    - Outcome: HHH in one trial, HTH in another trial, etc.
- Sample space: Set of all possible outcomes in the experiment, denoted as Ω (or S)
  - E.g., for the above example:
    - Ω = {HHH, HHT, HTH, HTT, THH, THT, TTH, TTT}
- Event: subset of the sample space Ω (i.e., an event is a set consisting of individual outcomes)
  - Event space: Collection of all events, called $\mathcal{F}$ (aka σ-algebra)
  - E.g., Event that # of heads is an even number.
    - E = {HHT, HTH, THH, TTT}
- Probability measure: function (mapping) from events to probability levels. I.e., $P: \mathcal{F} \rightarrow [0,1]$ (see next slide)
  - Probability that # of heads is an even number: 4/8 = 1/2.
- Probability space: $(\Omega, \mathcal{F}, P)$

Law of Total Probability

- $P(A) \geq 0, \forall A \in \mathcal{F}$
- $P(\Omega) = 1$

- Law of total probability

$$P(A) = P(A \cap B) + P(A \cap B^C)$$

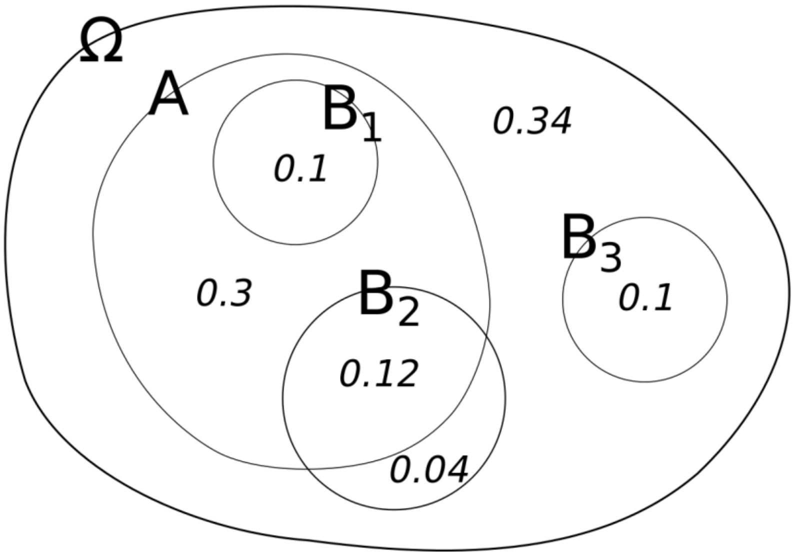
$$P(A) = \sum_i P(A \cap B_i)$$ Discrete B_i

$$P(A) = \int P(A \cap B_i) dB_i$$ Continuous B_i

Conditional Probability

For events $A, B \in \mathcal{F}$ with $P(B) > 0$, we may write the **conditional probability of A given B**:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$



From Wikipedia

$P(A|B_1) = 1$

$P(A|B_2) = 0.12 \div (0.12 + 0.04) = 0.75$

$\mathrm{P}(A|B_3) = 0$ (disjoint)

P(A) (The unconditional probability)

$= 0.30 + 0.10 + 0.12 = 0.52$

Bayes' Rule

Using the chain rule we may see:

$$P(A|B)P(B) = P(A \cap B) = P(B|A)P(A)$$

Rearranging this yields **Bayes' rule**:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Often this is written as:

$$P(B_i|A) = \frac{P(A|B_i)P(B_i)}{\sum_i P(A|B_i)P(B_i)}$$

Where B_i are a partition of Ω (note the bottom is just the law of total probability).

Likelihood Functions

- Why is Bayes' so useful in learning? Allows us to compute the posterior of w given data D:

$$\underbrace{p(w|D)}_{\text{Posterior}} = \frac{\overbrace{p(D|w)}^{\text{Likelihood}}\,\overbrace{p(w)}^{\text{Prior}}}{\underbrace{p(D)}_{\text{Evidence}}}$$

- Bayes' rule in words: posterior ∝ likelihood × prior

$$p(\mathbf{w}|D) \propto p(D|\mathbf{w})p(\mathbf{w})$$

- The likelihood function, $p(D|w)$, is evaluated for observed data D as a function of w. It expresses how probable the observed data set is for various parameter settings w.

Maximum Likelihood Estimation

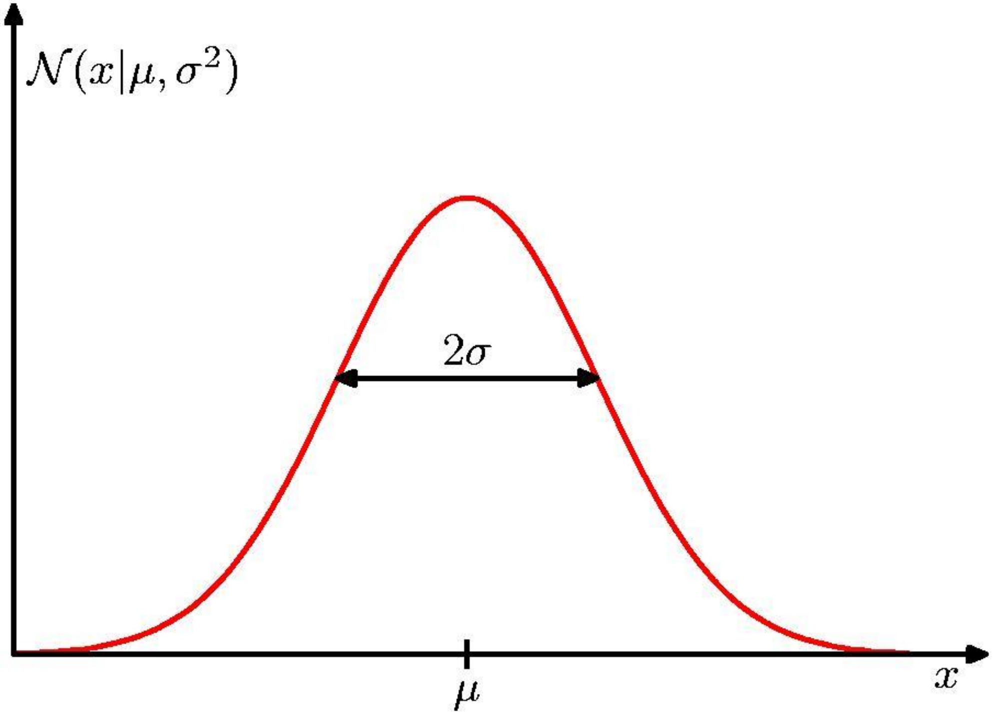
- Maximum Likelihood Estimation (MLE):
  - Choose parameters w that maximizes likelihood function $p(D|w)$.
  - Choose the value of w that maximizes the probability of observed data.

- Cf. Maximum A Posteriori (MAP) Estimation
  - Equivalent to maximizing $p(w|D) \propto p(D|w)p(w)$
  - Can compute this using Bayes' rule!
  - (Will be covered later)

The Gaussian Distribution

- Gaussian (Posterior)
  = Gaussian (Likelihood) x Gaussian (Prior)

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



$$\mathcal{N}(x|\mu,\sigma^2) > 0$$

$$\int_{-\infty}^{\infty}\mathcal{N}\left(x|\mu,\sigma^2\right)\,\mathrm{d}x = 1$$

Conjugate Priors

- When the posterior is in the same probability distribution family as the prior, the prior is called a **conjugate prior**.

Likelihood	Conjugate Prior Distribution
Bernoulli Binomial w/ known # trials Geometric	Beta
Poisson Exponential	Gamma
Categorical Multinomial	Dirichlet
Uniform	Pareto
Normal w/ known variance	Normal
Normal w/ known mean	Inverse gamma

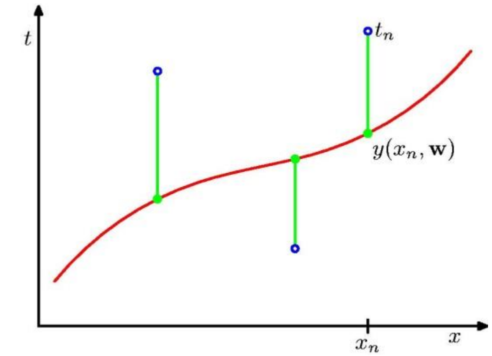
Recall: Probability Distributions

Distribution	PDF or PMF	Mean	Variance
Bernoulli(p)	$\begin{cases} p & \text{if } x = 1, \\ 1-p & \text{if } x = 0. \end{cases}$	p	$p(1-p)$
Binomial(n, p)	$\binom{n}{k} p^k (1-p)^{n-k}$ for $k = 0, 1, \ldots, n$	np	$np(1-p)$
Geometric(p)	$p(1-p)^{k-1}$ for $k = 1, 2, \ldots$	$\frac{1}{p}$	$\frac{1-p}{p^2}$
Poisson(λ)	$\frac{e^{-\lambda}\lambda^k}{k!}$ for $k = 0, 1, \ldots$	λ	λ
Uniform(a, b)	$\frac{1}{b-a}$ for all $x \in (a, b)$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$
Gaussian(μ, σ^2)	$\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ for all $x \in (-\infty, \infty)$	μ	σ^2
Exponential(λ)	$\lambda e^{-\lambda x}$ for all $x \geq 0, \lambda \geq 0$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$

Maximum Likelihood Interpretation

MLE for Linear Regression

- Assume a stochastic model:



$$y^{(n)} = \mathbf{w}^T \phi(\mathbf{x}^{(n)}) + \epsilon \text{ where } \epsilon \sim \mathcal{N}(0, \beta^{-1})$$

- This gives a likelihood function:

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

- With input matrix $\mathbf{\Phi}$ and output matrix $\mathbf{y}$, the data likelihood is:

$$p(\mathbf{y} | \mathbf{\Phi}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(y^{(n)} | \mathbf{w}^T \phi(\mathbf{x}^{(n)}), \beta^{-1})$$

Log-likelihood

- Data likelihood:

$$p(\mathbf{y}|\mathbf{\Phi}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(y^{(n)}|\mathbf{w}^T \phi(\mathbf{x}^{(n)}), \beta^{-1})$$

- Log-likelihood:

$$\log p(\mathbf{y}|\mathbf{\Phi}, \mathbf{w}, \beta) = \frac{N}{2} \log \beta - \frac{N}{2} \log 2\pi - \beta E_D(\mathbf{w})$$

where $E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} (\mathbf{w}^T \phi(\mathbf{x}^{(n)}) - y^{(n)})^2$

- Derivation?

Derivation of Log-likelihood of p

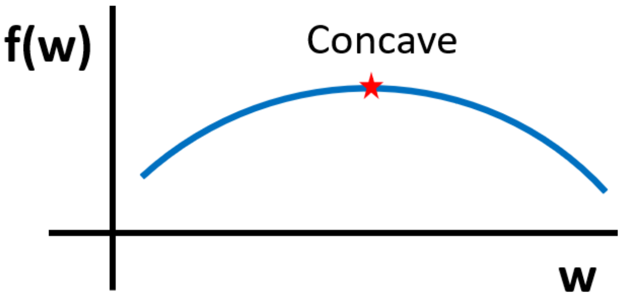
From $$p(y^{(n)}|\phi(\mathbf{x}^{(n)}),\mathbf{w},\beta) = \mathcal{N}(y^{(n)}|\mathbf{w}^T\phi(\mathbf{x}^{(n)}),\beta^{-1})$$
$$= \sqrt{\frac{\beta}{2\pi}}\exp(-\frac{\beta}{2}||y^{(n)}-\mathbf{w}^T\phi(\mathbf{x}^{(n)})||^2)$$

Derive: $$\log p(y^{(1)},y^{(2)},\dots,y^{(N)}|\mathbf{\Phi},\mathbf{w},\beta)$$
$$= \log\prod_{n=1}^{N}\mathcal{N}(y^{(n)}|\mathbf{w}^T\phi(\mathbf{x}^{(n)}),\beta^{-1})$$
$$= \sum_{n=1}^{N}\log\left(\sqrt{\frac{\beta}{2\pi}}\exp(-\frac{\beta}{2}||y^{(n)}-\mathbf{w}^T\phi(\mathbf{x}^{(n)})||^2)\right)$$
$$= \sum_{n=1}^{N}\left(\frac{1}{2}\log\beta-\frac{1}{2}\log 2\pi-\frac{\beta}{2}||y^{(n)}-\mathbf{w}^T\phi(\mathbf{x}^{(n)})||^2\right)$$
$$= \frac{N}{2}\log\beta-\frac{N}{2}\log 2\pi-\sum_{n=1}^{N}\frac{\beta}{2}||y^{(n)}-\mathbf{w}^T\phi(\mathbf{x}^{(n)})||^2$$

Maximum Likelihood Estimation

- Let's maximize the log-likelihood!
- Set the gradient of log-likelihood = 0 (Why?)

$$\nabla_{\mathbf{w}} \log p(\mathbf{y}|\mathbf{\Phi}, \mathbf{w}, \beta) = \nabla_{\mathbf{w}} \left(\underbrace{\frac{N}{2} \log \beta - \frac{N}{2} \log 2\pi}_{\text{Constant}} - \sum_{n=1}^{N} \frac{\beta}{2} ||y^{(n)} - \mathbf{w}^T \phi(\mathbf{x}^{(n)})||^2 \right)$$



$$= \beta \sum_{n=1}^{N} (y^{(n)} - \underbrace{\mathbf{w}^T \phi(\mathbf{x}^{(n)})}_{\text{Scalar}}) \phi(\mathbf{x}^{(n)})$$

$$= \beta \left(\sum_{n=1}^{N} y^{(n)} \phi(\mathbf{x}^{(n)}) - \phi(\mathbf{x}^{(n)}) \phi(\mathbf{x}^{(n)})^T \mathbf{w} \right) = 0$$

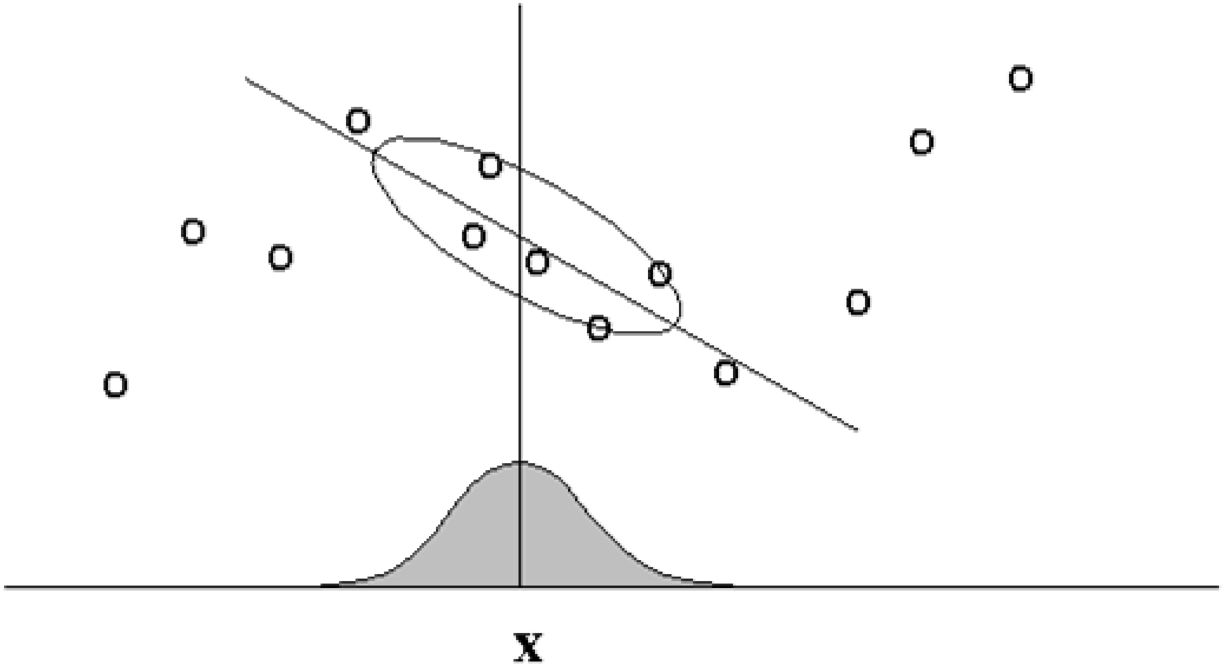
- In matrix form, $\beta(\Phi^T \mathbf{y} - \Phi^T \Phi \mathbf{w}) = 0$
- MLE solution is equivalent to OLS solution!

$$\mathbf{w}_{ML} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{y}$$

Locally-Weighted Linear Regression

Locally-Weighted Linear Regression

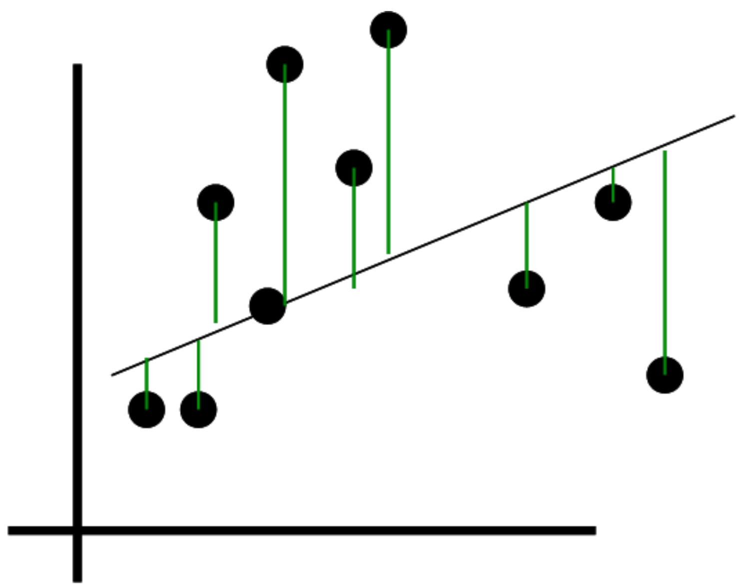
- Main idea: When predicting $h(\mathbf{x})$, give high weights for "neighbors" of $\mathbf{x}$.



In locally-weighted regression, points are weighted by proximity to the current $\mathbf{x}$ in question using a kernel. A regression is then computed using the weighted points.

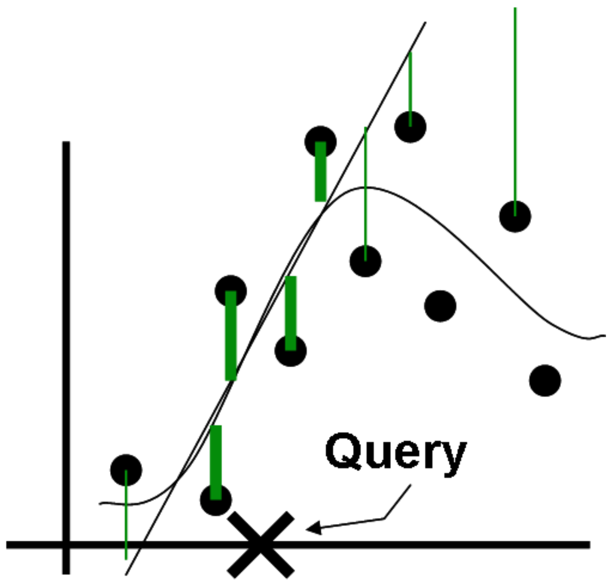
Slide credit: William Cohen

Linear Regression vs. Locally-Weighted Linear Regression



Linear regression

$$\sum_{n=1}^{N} (\mathbf{w}^T \phi(\mathbf{x}^{(n)}) - y^{(n)})^2$$



Locally-weighted linear regression

$$\sum_{n=1}^{N} r^{(n)} (\mathbf{w}^T \phi(\mathbf{x}^{(n)}) - y^{(n)})^2$$

Linear Regression vs. Locally-Weighted Linear Regression

- A new observation $\mathbf{x}$, training set $\{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$
- Linear regression
  - 1. Fit **w** to minimize $\sum_{n=1}^{N}(\mathbf{w}^T\phi(\mathbf{x}^{(n)}) - y^{(n)})^2$
  - 2. Predict: $\mathbf{w}^T\phi(\mathbf{x})$
- Locally-weighted linear regression
  - 1. Fit **w** to minimize $\sum_{n=1}^{N} r^{(n)}(\mathbf{w}^T\phi(\mathbf{x}^{(n)}) - y^{(n)})^2$
  - 2. Predict: $\mathbf{w}^T\phi(\mathbf{x})$

Linear Regression vs. Locally-Weighted Linear Regression

- Locally-weighted linear regression
  - 1. Fit **w** to minimize $\sum_{n=1}^{N} r^{(n)}(\mathbf{w}^T\phi(\mathbf{x}^{(n)}) - y^{(n)})^2$
  - 2. Predict: $\mathbf{w}^T\phi(\mathbf{x})$

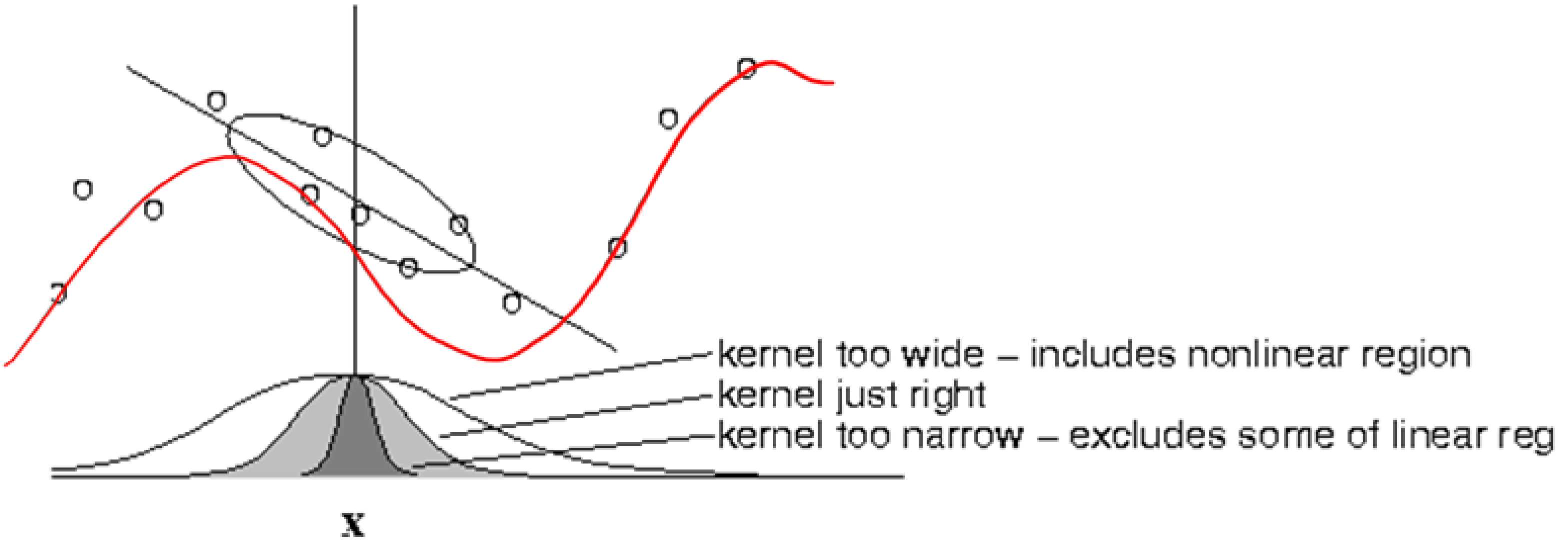
- Remarks:

  "Gaussian Kernel" τ: "kernel width"

  - Standard choice: $r^{(n)} = \exp\left(-\frac{\|\phi(\mathbf{x}^{(n)})-\phi(\mathbf{x})\|^2}{2\tau^2}\right)$
  - Note that $r^{(n)}$ depends on x (query point), and you solve linear regression for each query point x.

Locally-Weighted Linear Regression

- Choice of kernel width τ matters
  - Requires hyper-parameter tuning



The estimator is minimized when kernel includes as many training points as can be accommodated by the model. Too large a kernel includes points that degrade the fit; too small a kernel neglects points that increase confidence in the fit.

Slide credit: William Cohen

Next: Logistic Regression