EECS 545: Machine Learning

Lecture 10. Regularization and model selection

Honglak Lee

2/9/2011

Outline

- ML, MAP, and Bayesian
  - Maximum Likelihood
  - MAP
  - Bayesian
- Bias-Variance Tradeoff
- Model selection
  - Cross validation
- Advice on applying Machine Learning

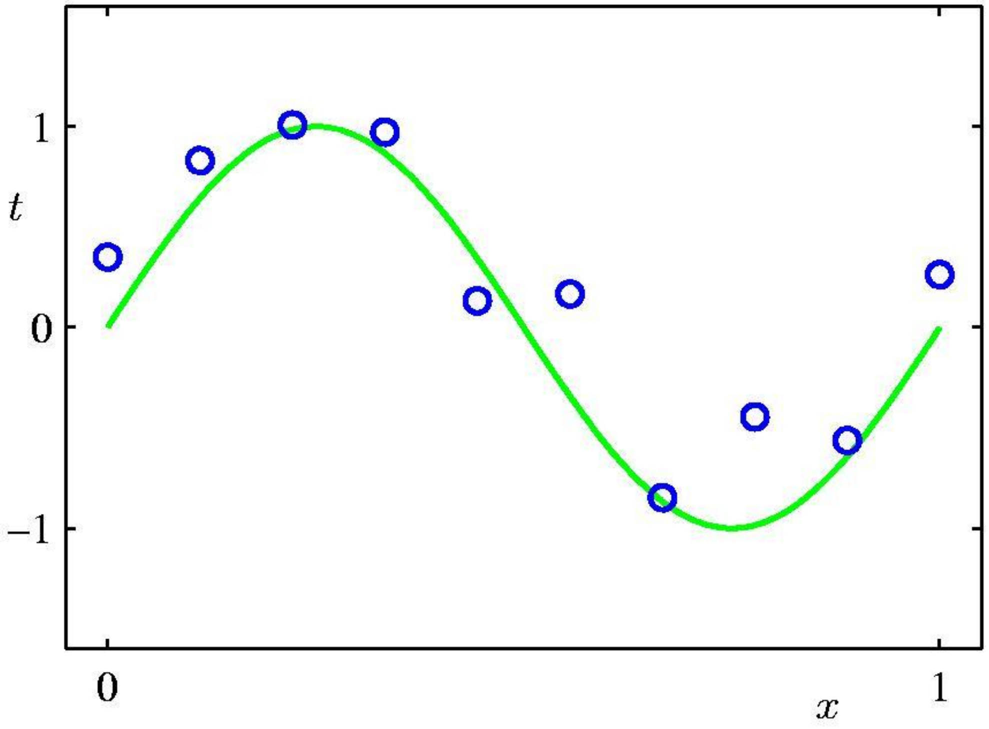
ML vs. MAP vs. Bayesian (summary)

- Maximum Likelihood
  - Objective: Log-likelihood

    $\log P(D|w)$
  - Example: linear regression (w/o regularization)
- MAP (Maximum a Posteriori)
  - Objective: Log-likelihood + Log-Prior

    $\log P(D|w) + \log P(w)$
  - Example: Regularized linear regression
- Bayesian
  - Objective: Estimate $P(w|D) \propto P(D|w)P(w)$
    - Goal is not a point estimate!! (Unlike ML or MAP)
  - Example: Bayesian linear regression

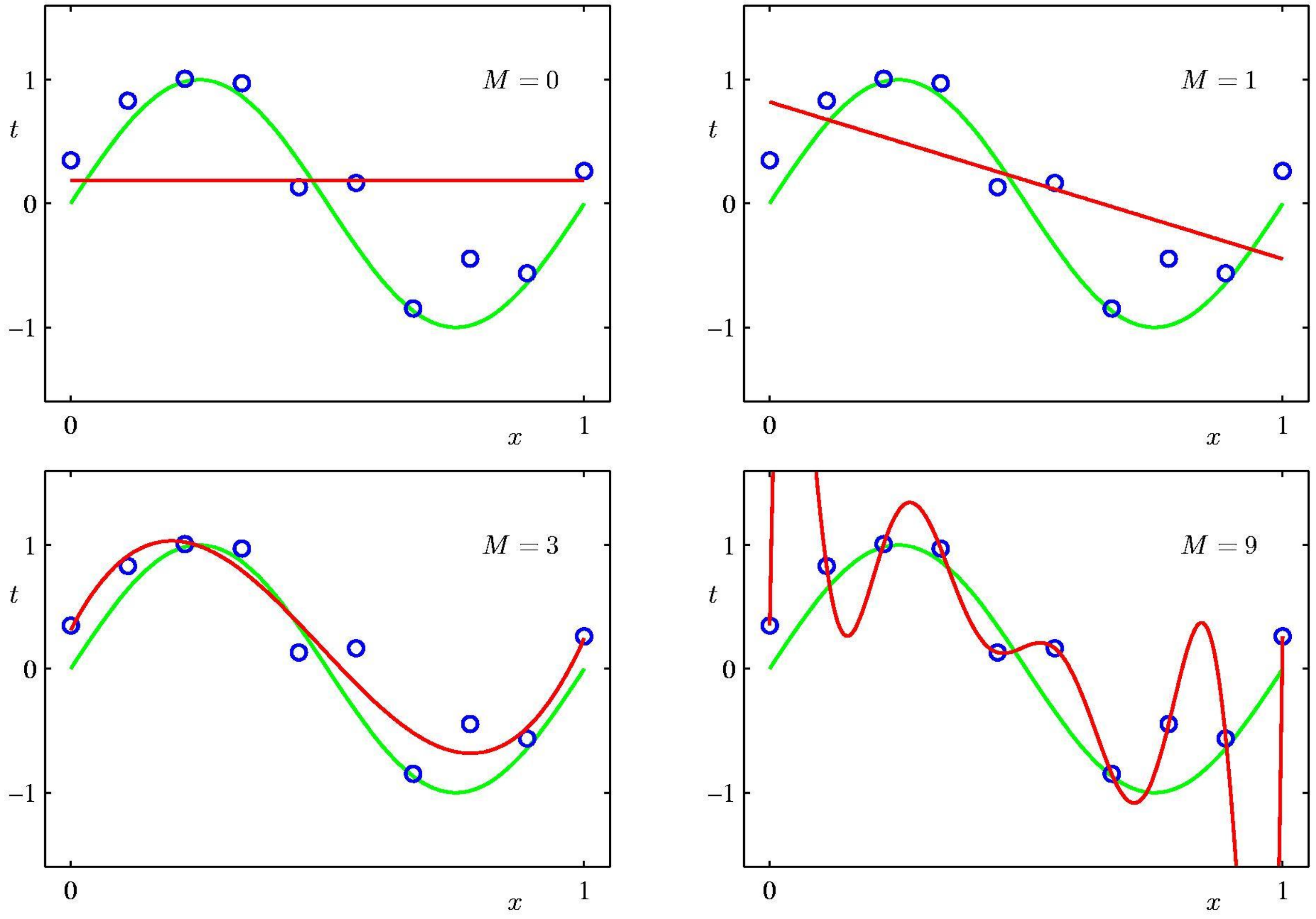
Polynomial Curve Fitting



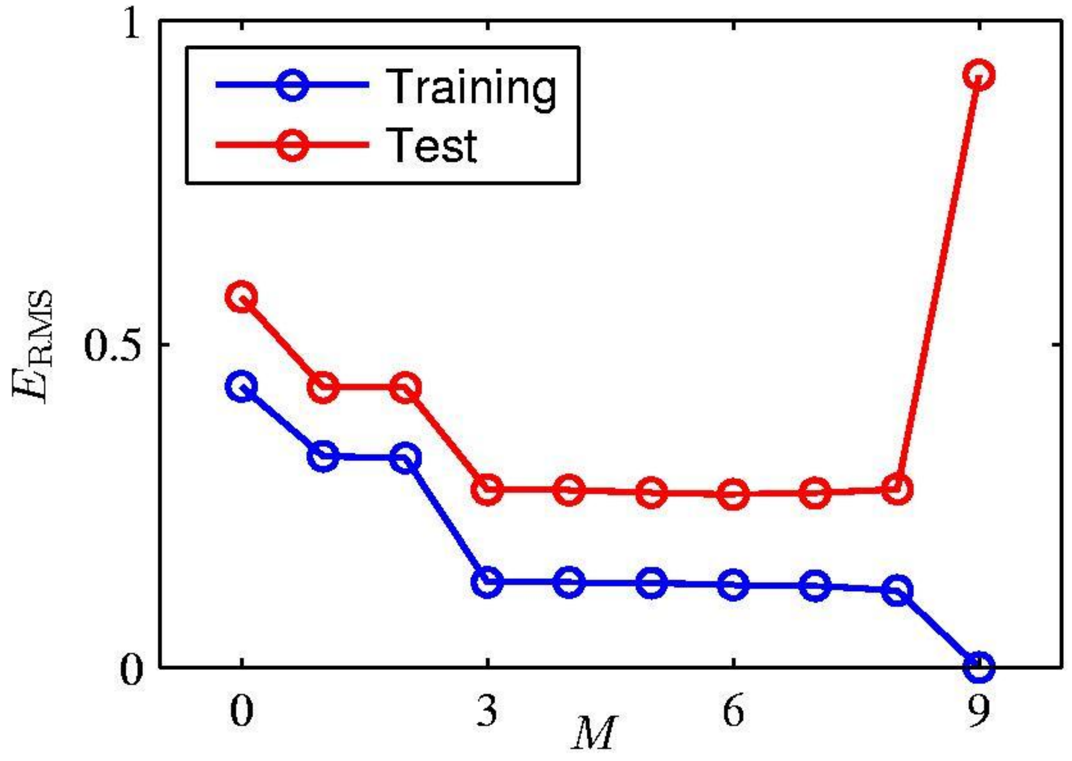
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Maximum Likelihood (in Linear Regression)

- Choosing the right complexity is important
  - Watch out for underfitting/overfitting



Underfitting vs. overfitting



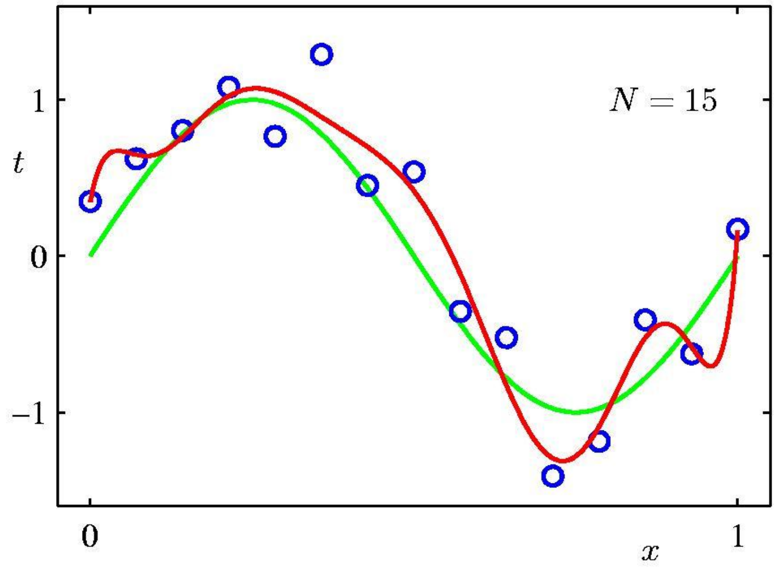
Root-Mean-Square (RMS) Error: $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^{\star})/N}$

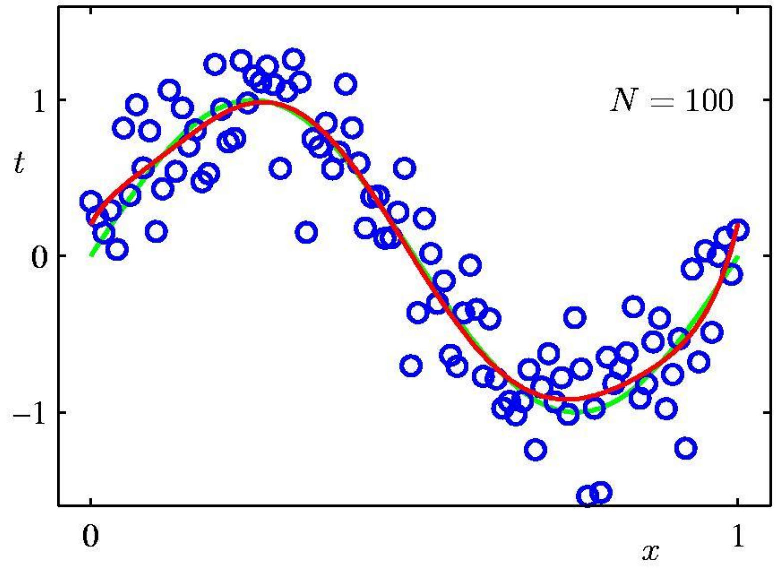
How can we avoid overfitting?

- More training data
  - Always helps
- Regularizaiton
- Bayesian

More training data

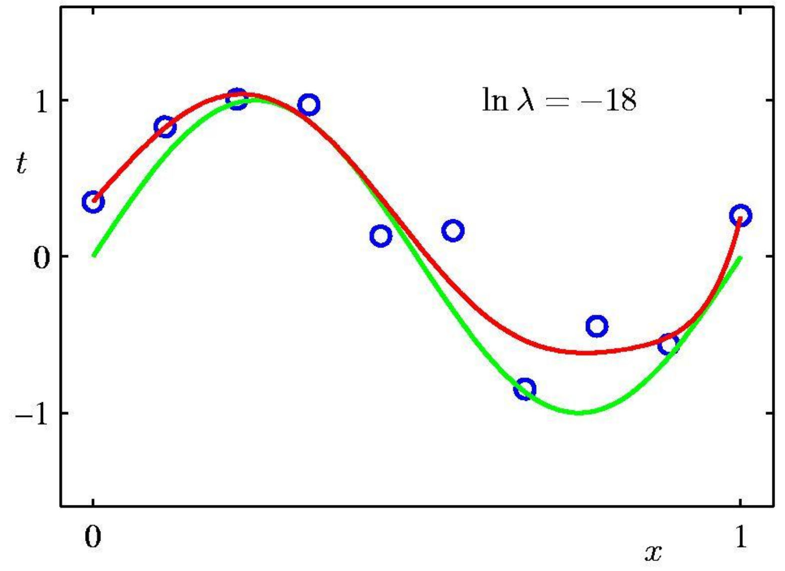
- Even complicated models can benefit (avoid overfitting) by having large amount of data
  - Example: 9th order polynomial (M=9)

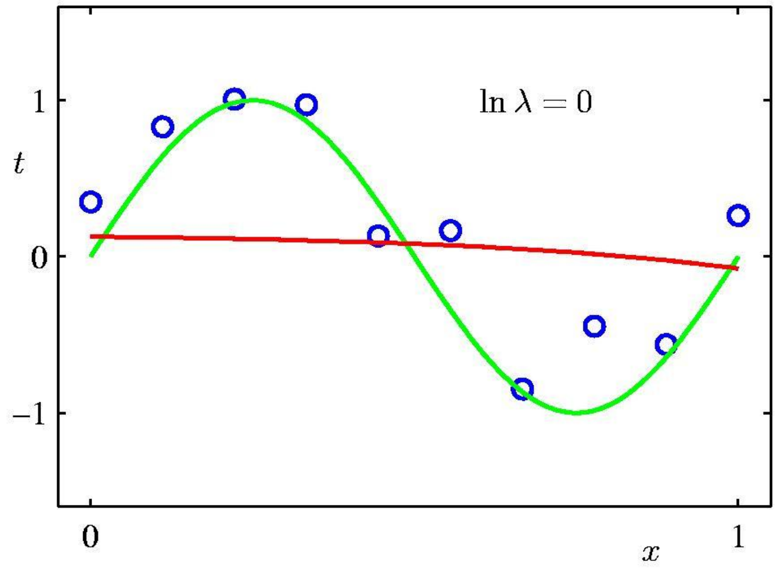




Regularization (for Linear Regression)

- Regularization can implicitly control the complexity of models
  - Example: 9th order polynomial (M=9)
  - Choosing right level of regularization is important



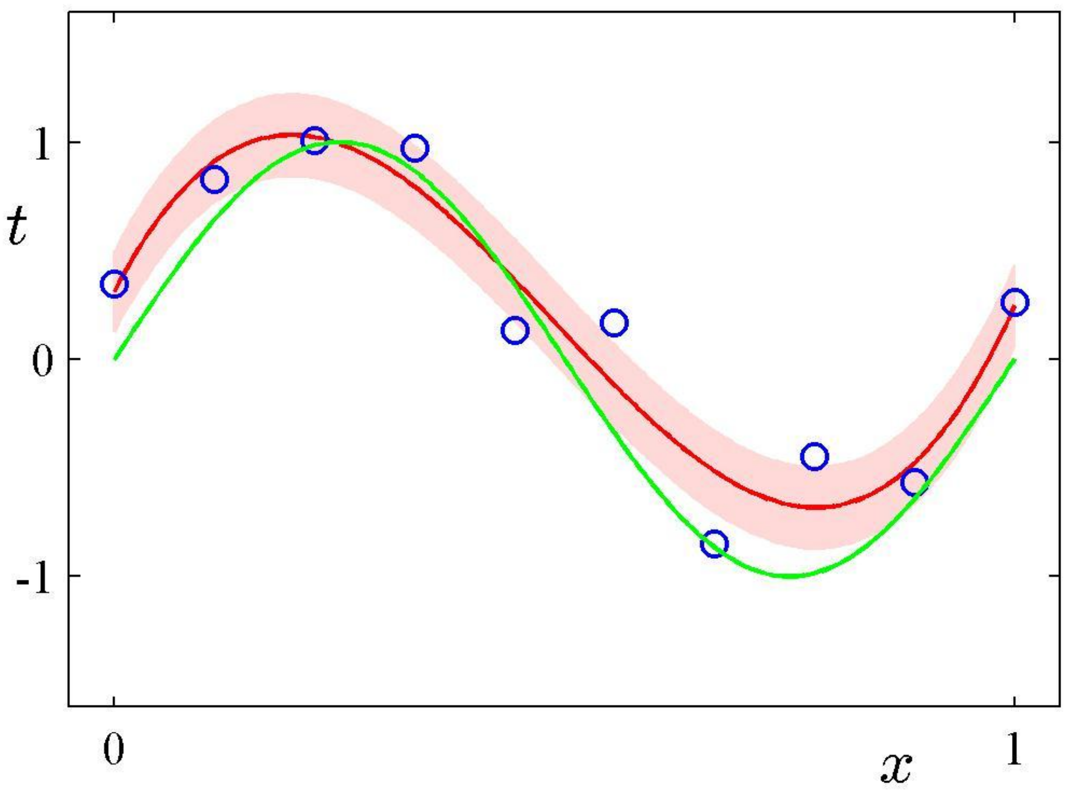


$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w})$$

Data term + Regularization term

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

Bayesian (Linear Regression)



Bayesian still assumes some hyperparameters, but overfitting is generally much less than maximum-likelihood

ML vs. MAP vs. Bayesian (summary)

- Maximum Likelihood
  - Objective: Log-likelihood

    log P(D|w)
  - Example: linear regression (w/o regularization)
- MAP (Maximum a Posteriori)
  - Objective: Log-likelihood + Log-Prior

    log P(D|w) + log P(w)
  - Example: Regularized linear regression
- Bayesian
  - Objective: Estimate P(w|D) ∝ P(D|w)P(w)
    - Goal is not a point estimate!! (Unlike ML or MAP)
  - Example: Bayesian linear regression

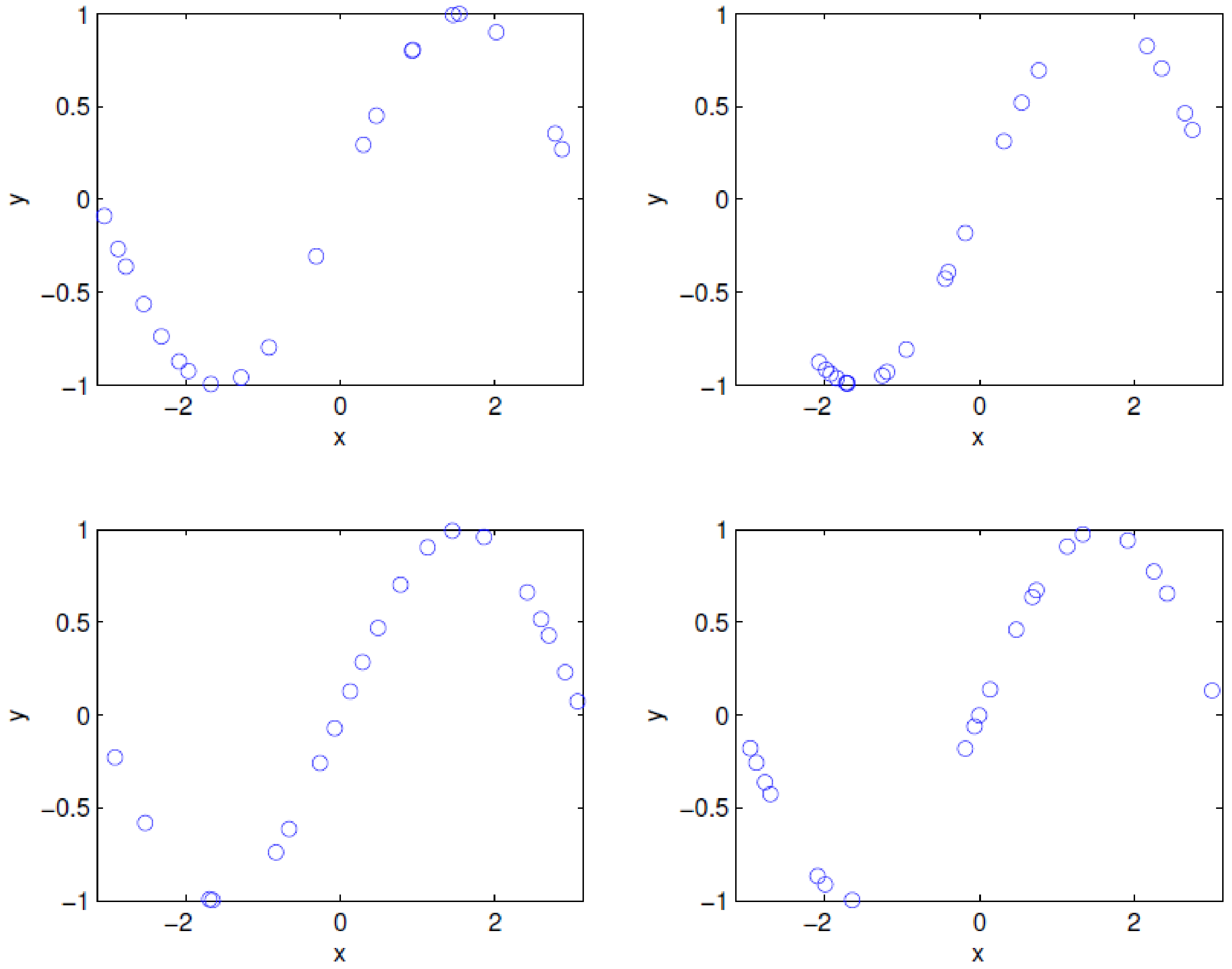
Variance-Bias Tradeoff

The Bias-Variance Decomposition

- Setting:
  - Given a distribution of P(x,y)
  - Sample training data
    $$D_{train} = \{(x_n, y_n): n = 1, \dots, N\} \sim P(x, y)$$
  - Train a learning algorithm on D
- Depending on samples, learning algorithm can still give different results (ML, MAP, etc.)
- Goal: We want to learn a model with
  - Small bias (i.e., how well a model fits the data?)
  - Small variance (i.e., how stable a model is wrt data samples?)

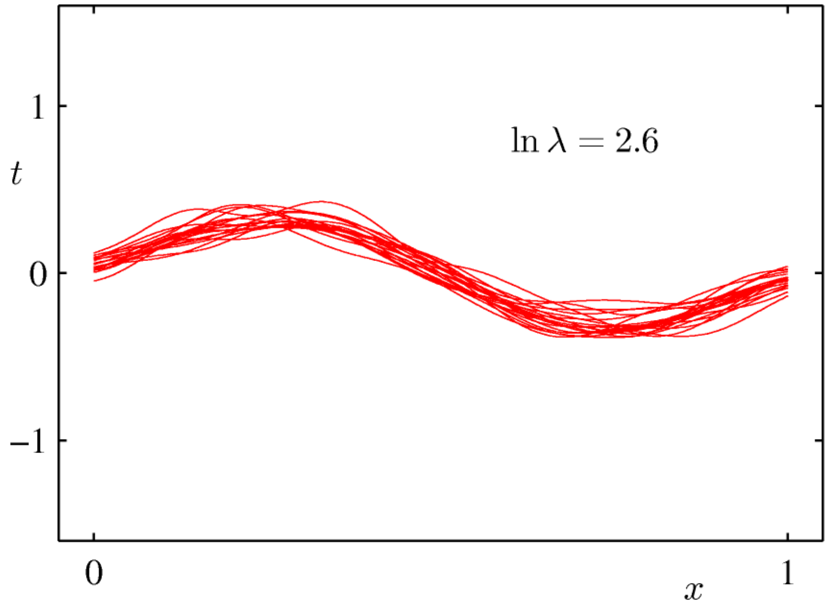
The Bias-Variance Decomposition

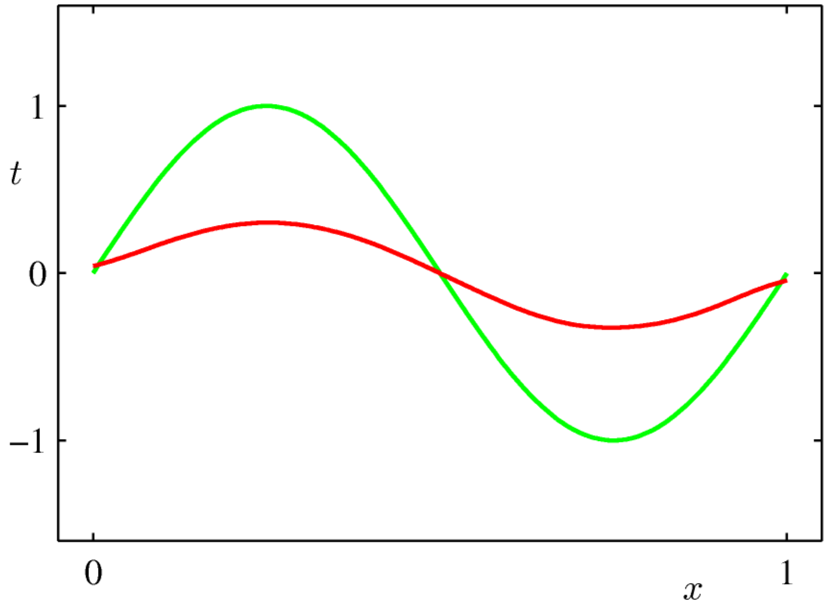
- Example: samples from the sinusoidal function y=sin(x)



The Bias-Variance Decomposition

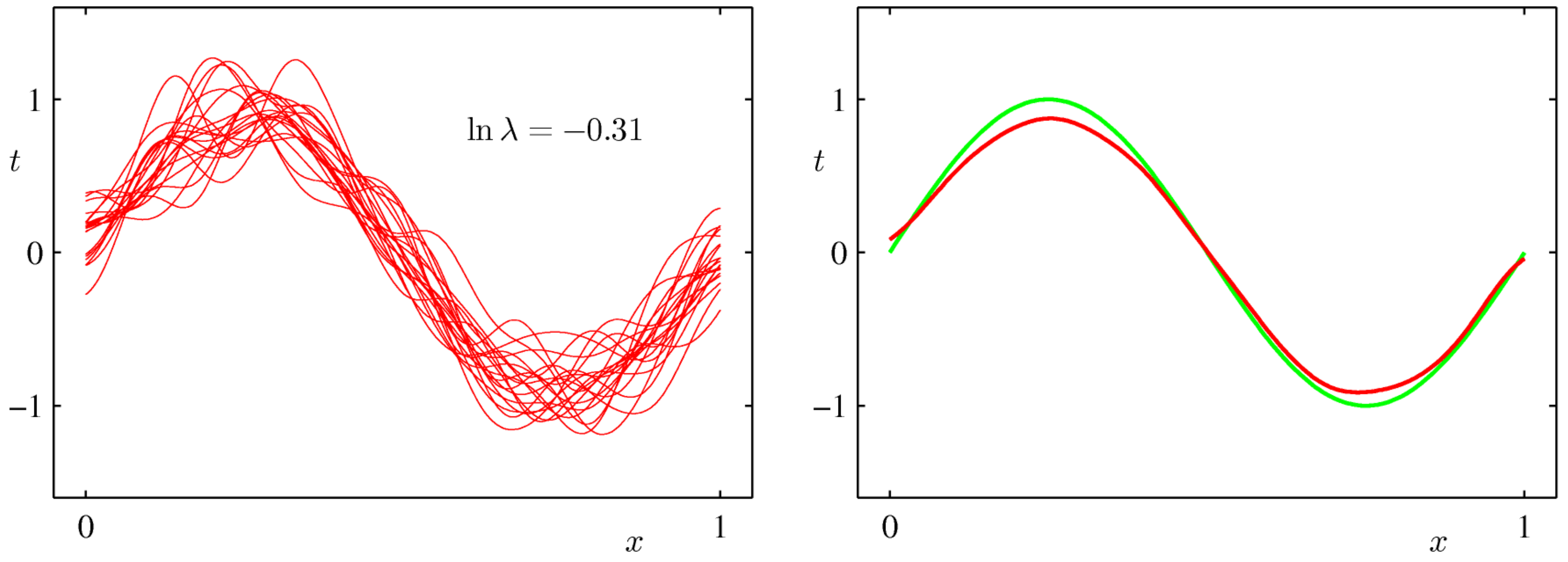
- Example: 25 data sets from the sinusoidal, varying the degree of regularization λ.





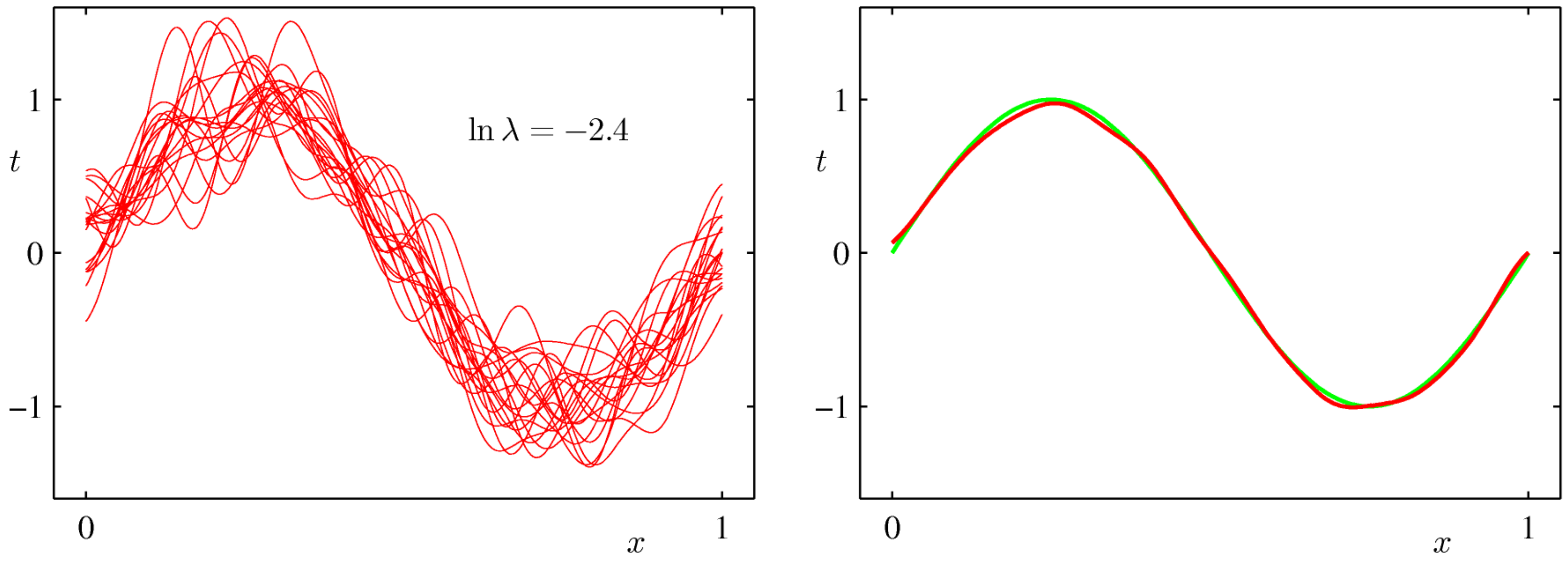
The Bias-Variance Decomposition

- Example: 25 data sets from the sinusoidal, varying the degree of regularization λ.



The Bias-Variance Decomposition

- Example: 25 data sets from the sinusoidal, varying the degree of regularization λ.



The Squared Loss Function

Let's assume that we sample (x,t) according to a probability distribution p(x,t). Then expectation of squared loss can be written as:

$$\mathbb{E}[L] = \iint \{y(\mathbf{x}) - t\}^2 p(\mathbf{x}, t)\, \mathrm{d}\mathbf{x}\, \mathrm{d}t$$

Decompose:

$$\begin{aligned}\{y(\mathbf{x}) - t\}^2 &= \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}] + \mathbb{E}[t|\mathbf{x}] - t\}^2 \\ &= \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}^2 + 2\{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}\{\mathbb{E}[t|\mathbf{x}] - t\} + \{\mathbb{E}[t|\mathbf{x}] - t\}^2\end{aligned}$$

We finally get:

$$\mathbb{E}[L] = \int \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}^2 p(\mathbf{x})\, \mathrm{d}\mathbf{x} + \int \operatorname{var}[t|\mathbf{x}]\, p(\mathbf{x})\, \mathrm{d}\mathbf{x}$$

The Bias-Variance Decomposition (1)

- Recall the *expected squared loss,*

$$\mathbb{E}[L] = \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x} + \underbrace{\iint \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t}$$

- where

$$h(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}] = \int t p(t|\mathbf{x})\,\mathrm{d}t.$$

- The second term of E[L] corresponds to the noise inherent in the random variable t.

- What about the first term?

The Bias-Variance Decomposition (2)

- Suppose we were given multiple data sets, each of size N. Any particular data set, D, will give a particular function y(x;D). We then have

$$
\begin{aligned}
&\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2 \\
&= \{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] + \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 \\
&= \{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2 + \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 \\
&\quad +2\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}.
\end{aligned}
$$

The Bias-Variance Decomposition (3)

- Taking the expectation over D yields

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2\right] \\
&\quad = \underbrace{\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2}_{(\text{bias})^2} + \underbrace{\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right]}_{\text{variance}}.
\end{aligned}
$$

The Bias-Variance Decomposition (4)

- Thus we can write

$$\text{expected loss} = (\text{bias})^2 + \text{variance} + \text{noise}$$

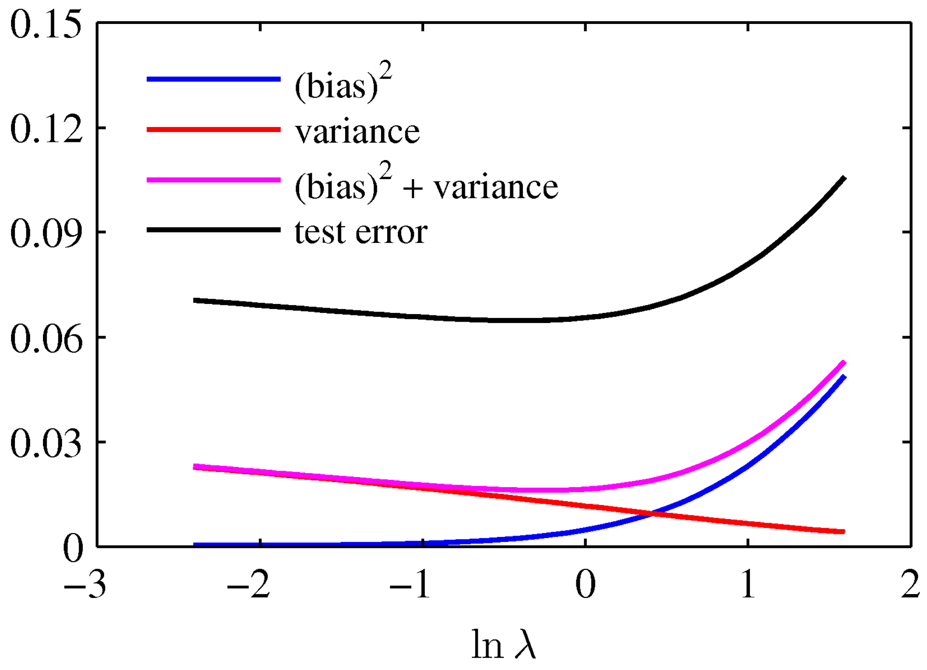
- where

$$\begin{aligned}
(\text{bias})^2 &= \int \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x} \\
\text{variance} &= \int \mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right] p(\mathbf{x})\,\mathrm{d}\mathbf{x} \\
\text{noise} &= \iint \{h(\mathbf{x}) - t\}^2 p(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t
\end{aligned}$$

The Bias-Variance Trade-off

From these plots, we note that

- An over-regularized model (large λ) will have a high bias,
- While an under-regularized model (small λ) will have a high variance.



Model Selection

Choosing right models

- For polynomial curve fitting, which value of M should we choose?
- For SVM, which hyperparameter values should we choose?
  - Linear SVM: C (penalty for margin violation)
  - RBF kernel SVM: C, kernel width
  - Polynomial kernel SVM: C, degree of polynomial
- Generally, given a set of models, $M = \{M_1, M_2, \dots, M_d\}$, how can we choose optimal M_i?
  - Model:
    - Class (or set) of hypothesis: learning algorithm, hyperparameters, etc.
    - Fixed during training
  - Parameter:
    - Aka, hypothesis: (w values for logistic regression/SVM/linear regression)
    - Can be trained based on data.

Simple Idea (that doesn’t work)

- Given Data D
- Train each model M_i on D, to get some hypothesis (model) h_i
- Pick the hypothesis with the smallest training error

Cross validation

- Hold-out cross validation (aka simple cross validation)
  - 1. Randomly split D into D_{train} (say, 70% of the data) and D_{CV} (the remaining 30%).
    - Here, D_{CV} is called the hold-out cross validation set.
  - 2. Train each model M_i on D_{train} only, to get some hypothesis h_i.
  - 3. Select and output the hypothesis h_i that had the smallest error on the hold out cross validation set.
- Disadvantage:
  - Waste 30% of the data!

K fold Cross validation

- 1. Randomly split D into k disjoint subsets of N/k training examples each.
  - Lets call these subsets $D_1, \dots, D_k$.
- 2. For each model M_i, we evaluate it as follows:
  - For j = 1, . . . , k
    - Train the model M_i on $D_1 \cup \cdots \cup D_{j-1} \cup D_{j+1} \cup \cdots \cup D_k$ (i.e., train on all the data except D_j) to get some hypothesis h_{ij}.
    - Test the hypothesis h_{ij} on D_j , to get $\epsilon_{D_j}(h_{ij})$.
  - The estimated generalization error of model M_i is then calculated as the average of the $\epsilon_{D_j}(h_{ij})$'s (averaged over j).
- 3. Pick the model M_i with the lowest estimated generalization error, and retrain that model on the entire training set S. The resulting hypothesis is then output as our final answer.

K fold Cross validation

- Popular choice of k = 10, 5
- Special case: K=1
  - Called, Leave-one-out cross validation (LOO CV)
  - Expensive, but wastes least amount of training data for cross validation.