第三章 线性回归

吴洪
电子科技大学计算机学院

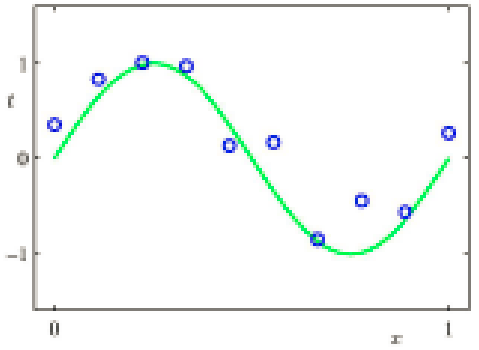
*With thanks to Bishop, Sontag, Xing, Hauskrecht, Urtasun and Zemel for use of their figures and slides.

2020年春季

Outline

- Linear regression and least squares
- Regularized Least Squares
- The Bias-Variance Decomposition
- Bayesian Linear Regression

Polynomial Curve Fitting with a Scalar



Training data set
N=10, Input x, target t

Simplest form of regression:

– With a single input variable x

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

M is the order of the polynomial,

x^j denotes x raised to the power j,

Coefficients $w_0,\ldots,w_M$ are collectively denoted by vector $\mathbf{w}$

Polynomial Curve Fitting with a Scalar

Task: Learn $\mathbf{w}$ from training data $D = \{(x_i, t_i)\},\ i = 1, \ldots, N$

- Can be done by minimizing an error function that minimizes the misfit between $y(x, \mathbf{w})$ for any given $\mathbf{w}$ and training data
- One simple choice of error function is sum of squares of error between predictions $y(x_n, \mathbf{w})$ for each data point x_n and corresponding target values t_n so that we minimize

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

- It is zero when function $y(x, \mathbf{w})$ passes exactly through each training data point

Regression with multiple inputs

- Generalization
  - Predict value of continuous target variable t given value of d input variables $\mathbf{x}=[x_1,..x_D]$
  - t can also be a set of variables (multiple regression)
  - Linear functions of adjustable parameters
    - Specifically linear combinations of nonlinear functions of input variable
- Polynomial curve fitting is good only for:
  - Single input variable scalar x
  - It cannot be easily generalized to several variables, as we will see

Simplest Linear Model with D inputs

- Regression with D input variables

$$y(\mathbf{x},\mathbf{w}) = w_0 + w_1 x_1 + .. + w_d x_d = \mathbf{w}^T\mathbf{x}$$

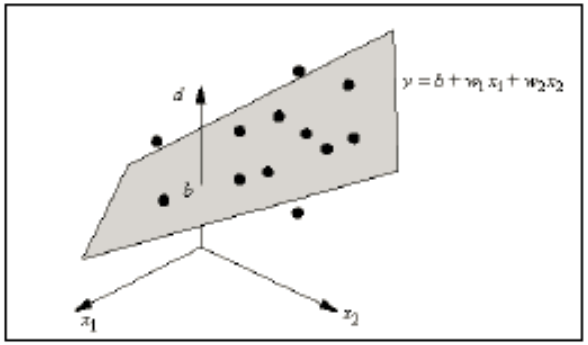
This differs from
Linear Regression with one variable
and Polynomial Reg with one variable

where $\mathbf{x}=(x_1,..,x_D)^T$ are the input variables

- Called Linear Regression since it is a linear function of
  - parameters $w_0,..,w_D$
  - input variables $x_1,..,x_D$
- Significant limitation since it is a linear function of input variables
  - In the one-dimensional case this amounts a straight-line fit (degree-one polynomial)
  - $y(x,\mathbf{w}) = w_0 + w_1 x$

Fitting a Regression Plane

- Assume t is a function of inputs $x_1, x_2, \ldots x_D$
  Goal: find best linear regressor of t on all inputs
  - Fitting a hyperplane through N input samples
  - For $D = 2$:



x_1	x_2	t
1	2	2
2	5	1
2	3	2
2	2	2
3	4	1
3	5	3
4	6	2
5	5	3
5	6	4
5	7	3
6	8	4
7	6	2
8	4	4
8	9	3
9	8	4

- Being a linear function of input variables imposes limitations on the model
  - Can extend class of models by considering fixed nonlinear functions of input variables

Not all functions can be approximated using the input values directly

$$y=10+3x_1^2-2x_2^2+\varepsilon$$

In some cases we would like to use polynomial or other terms based on the input data, are these still linear regression problems?

Yes. As long as the coefficients are linear the equation is still a linear regression problem!

Basis Functions

- In many applications, we apply some form of fixed-preprocessing, or feature extraction, to the original data variables
- If the original variables comprise the vector $\mathbf{x}$, then the features can be expressed in terms of basis functions $\{\phi_j(\mathbf{x})\}$
  - By using nonlinear basis functions we allow the function $y(\mathbf{x},\mathbf{w})$ to be a nonlinear function of the input vector $\mathbf{x}$
    - They are linear functions of parameters (gives them simple analytical properties), yet are nonlinear wrt input variables

Linear Regression with M Basis Functions

Extended by considering nonlinear functions of input variables

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

- where $\phi_j(\mathrm{x})$ are called Basis functions
- We now need M weights for basis functions instead of D weights for features
- With a dummy basis function $\phi_0(\mathrm{x})=1$ corresponding to the bias parameter w_0, we can write

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

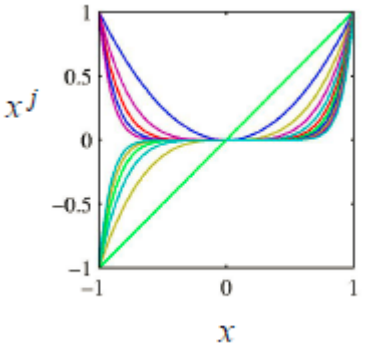
- where $\mathrm{w}=(w_0, w_1, .., w_{M-1})$ and $\Phi=(\phi_0, \phi_1, .., \phi_{M-1})^{\mathrm{T}}$

Choice of Basis Functions

- Many possible choices for basis function:
  1. Polynomial regression
     - Good only if there is only one input variable
  2. Gaussian basis functions
  3. Sigmoidal basis functions
  4. Fourier basis functions
  5. Wavelets

1. Polynomial Basis for one variable

- Polynomial Basis (for single variable x)
  $\phi_j(x) = x^j$ with degree M-1 polynomial



- Disadvantage
  - Global:
    - changes in one region of input space affects others
  - Difficult to formulate
    - Number of polynomials increases exponentially with M
  - Can divide input space into regions
    - use different polynomials in each region:
    - equivalent to spline functions

Can we use Polynomial with D variables?

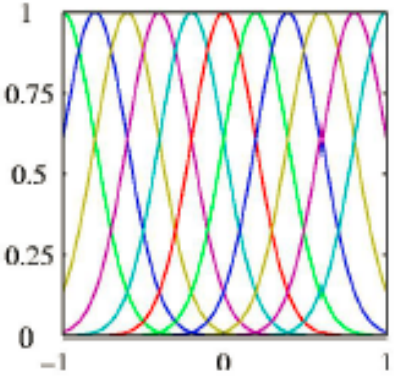
- Consider (for a vector x) the basis: $\phi_j(\mathrm{x}) = \|\mathrm{x}\|^j = \left[\sqrt{x_1^2 + x_2^2 + ..+ x_d^2}\right]^j$
  - $\mathrm{x}=(2,1)$ and $\mathrm{x}=(1,2)$ have the same squared sum, so it is unsatisfactory
  - Vector is being converted into a scalar value thereby losing information
- Better polynomial approach:
  - Polynomial of degree M-1 has terms with variables taken none, one, two… M-1 at a time.
  - Use multi-index $j=(j_1,j_2,..j_D)$ such that $j_1+j_2+..j_D \leq M$-1
  - For a quadratic $(M=3)$ with three variables $(D=3)$

$$y(\mathrm{x},\mathrm{w}) = \sum_{(j_1,j_2,j_3)} w_j \phi_j(\mathrm{x}) = w_0 + w_{1,0,0}x_1 + w_{0,1,0}x_2 + w_{0,0,1}x_3 + w_{1,1,0}x_1x_2 + w_{1,0,1}x_1x_3 +$$
$$w_{0,1,1}x_2x_3 + w_{2,0,0}x_1^2 + w_{0,2,0}x_2^2 + w_{0,0,2}x_3^2$$

  - Number of quadratic terms is $1+D+D(D\text{-}1)/2+D$
  - For $D=46$, it is 1128
- Better to use Gaussian kernel, discussed next

2. Gaussian Radial Basis Functions

- Gaussian $$\phi_j(x) = \exp\left(\frac{(x-\mu_j)^2}{2\sigma^2}\right)$$
  - Does not necessarily have a probabilistic interpretation
  - Usual normalization term is unimportant
    - since basis function is multiplied by weight w_j



- Choice of parameters
  - μ_j govern the locations of the basis functions
    - Can be an arbitrary set of points within the range of the data
      - Can choose some representative data points
  - σ governs the spatial scale
    - Could be chosen from the data set e.g., average variance
- Several variables
  - A Gaussian kernel would be chosen for each dimension
  - For each j a different set of means would be needed– perhaps chosen from the data

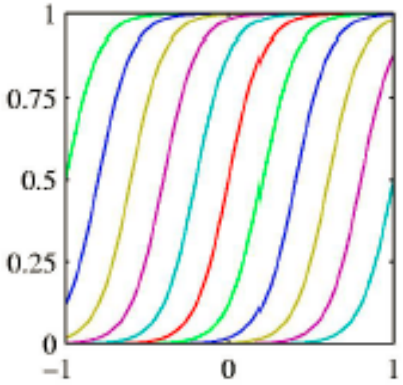
$$\phi_j(\mathrm{x}) = \exp\left(-\frac{1}{2}(\mathrm{x}-\mu_j)^t\Sigma^{-1}(\mathrm{x}-\mu_j)\right)$$

3. Sigmoidal Basis Function

- Sigmoid $$\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right) \quad \text{where} \quad \sigma(a) = \frac{1}{1+\exp(-a)}$$
- Equivalently, $\tanh$ because it is related to logistic sigmoid by

$$\tanh(a) = 2\sigma(a) - 1$$

Logistic Sigmoid
For different μ_j



4. Other Basis Functions

- Fourier
  - Expansion in sinusoidal functions
  - Infinite spatial extent
- Signal Processing
  - Functions localized in time and frequency
  - Called *wavelets*
    - Useful for lattices such as images and time series
- Further discussion independent of choice of basis including $\phi(\mathrm{x}) = \mathrm{x}$

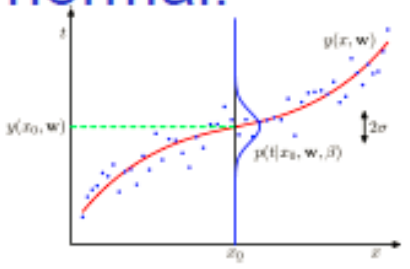
Least Squares - A Maximum Likelihood Explanation

- Will show that Minimizing sum-of-squared errors is the same as maximum likelihood solution under a Gaussian noise model
- Target variable is a scalar t given by deterministic function $y(\mathrm{x},\mathrm{w})$ with additive Gaussian noise ε

  $t = y(\mathrm{x},\mathrm{w}) + \varepsilon$

  – which is a *zero-mean* Gaussian with *precision* β
- Thus distribution of t is univariate normal:

$$p(t|\mathrm{x},\mathrm{w},\beta) = N(t \mid \underbrace{y(\mathrm{x},\mathrm{w})}_{\text{mean}}, \underbrace{\beta^{-1}}_{\text{variance}})$$

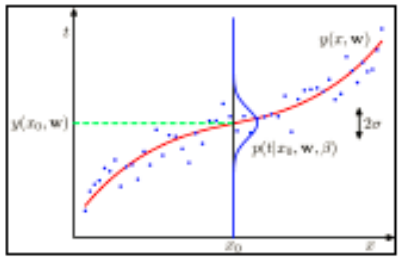


Likelihood Function

- Data set:
  - Input $X=\{x_1,..x_N\}$ with target $t = \{t_1,..t_N\}$
  - Target variables t_n are scalars forming a vector of size N

- Likelihood of the target data
  - It is the probability of observing the data assuming they are independent
  - since $p(t|x,w,\beta)=N(t \mid y(x,w),\beta^{-1})$
  - and $y(x,w) = \sum_{j=0}^{M-1} w_j \phi_j(x) = w^T \phi(x)$

$$p(t \mid X, w, \beta) = \prod_{n=1}^{N} N\left(t_n \mid w^T \phi(x_n), \beta^{-1}\right)$$



Maximum Likelihood and Least Squares

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.

Maximum Likelihood and Least Squares

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 $\mathbf{w}$, 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

Design Matrix $\boldsymbol{\Phi} =$

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

What is the role of Bias parameter w_0?

- Sum-of squares error function is:

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

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{ t_n - w_0 - \sum_{j=1}^{M-1} w_j\phi_j(\mathbf{x}_n) \right\}^2$$

  - Setting derivatives wrt w_0 equal to zero and solving for w_0

$$w_0 = \bar{t} - \sum_{j=1}^{M-1} w_j \bar{\phi}_j$$

where $\bar{t} = \frac{1}{N}\sum_{n=1}^{N} t_n$ and $\bar{\phi}_j = \frac{1}{N}\sum_{n=1}^{N}\phi_j(\mathbf{x}_n)$

- Thus bias w_0 compensates for difference between average target values and weighted sum of averages of basis function values

Maximum Likelihood for precision

- We have determined m.l.e. solution for $\mathbf{w}$ using a probabilistic formulation
  - $p(t|\mathbf{x},\mathbf{w},\beta)=N(t|y(\mathbf{x},\mathbf{w}),\beta^{-1})$
  - With log-likelihood

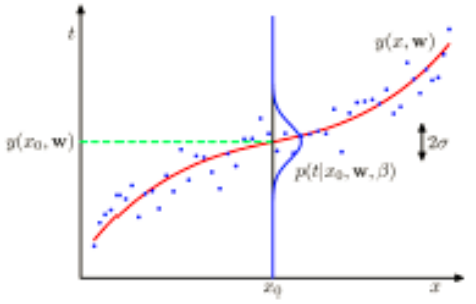
$$\ln p(\mathbf{t}\,|\,\mathbf{w},\beta)=\frac{N}{2}\ln\beta-\frac{N}{2}\ln 2\pi-\beta E_D(\mathbf{w})$$

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

$$\mathbf{w}_{ML}=\Phi^{+}\mathbf{t}$$

- Taking gradient wrt β gives

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



Difficulty of Direct solution

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

- This direct solution can lead to numerical difficulties
  - When $\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}$ is close to singular ($\mathrm{determinant}=0$)
  - When two basis functions are collinear parameters can have large magnitudes
- Not uncommon with real data sets
- Can be addressed using
  - Singular Value Decomposition
  - Addition of regularization term ensures matrix is non-singular

Sequential Learning

- Data items considered one at a time (a.k.a. online learning); use stochastic (sequential) gradient descent:

$$\begin{aligned}\mathbf{w}^{(\tau+1)} &= \mathbf{w}^{(\tau)} - \eta \nabla E_n \\ &= \mathbf{w}^{(\tau)} + \eta(t_n - \mathbf{w}^{(\tau)\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n))\boldsymbol{\phi}(\mathbf{x}_n).\end{aligned}$$

- This is known as the *least-mean-squares (LMS) algorithm*. Issue: how to choose η?

Outline

- Linear regression and least squares
- Regularized Least Squares
- The Bias-Variance Decomposition
- Bayesian Linear Regression

Regularized Least Squares

- As model complexity increases, e.g., degree of polynomial or no.of basis functions, then it is likely that we overfit
- One way to control overfitting is not to limit complexity but to add a regularization term to the error function
- Error function to minimize takes the form

$$E(\mathrm{w}) = E_D(\mathrm{w}) + \lambda E_W(\mathrm{w})$$

  - where λ is the *regularization coefficient* that controls relative importance of data-dependent error $E_D(\mathrm{w})$ and regularization term $E_W(\mathrm{w})$

Simplest Regularizer is weight decay

- Simple form of regularization term is

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

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

Ridge Regression

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

- This regularizer is called *weight decay*
  - because in sequential learning weight values decay towards zero unless supported by data

Closed-form Solution with Regularizer

- Error function with *quadratic regularizer is*,

$$E(\mathrm{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{ t_n - \mathrm{w}^T\phi(x_n) \right\}^2 + \frac{\lambda}{2}\mathrm{w}^T\mathrm{w}$$

- Its exact minimizer can be found in closed form
  - By setting gradient wrt w to zero and solving for w

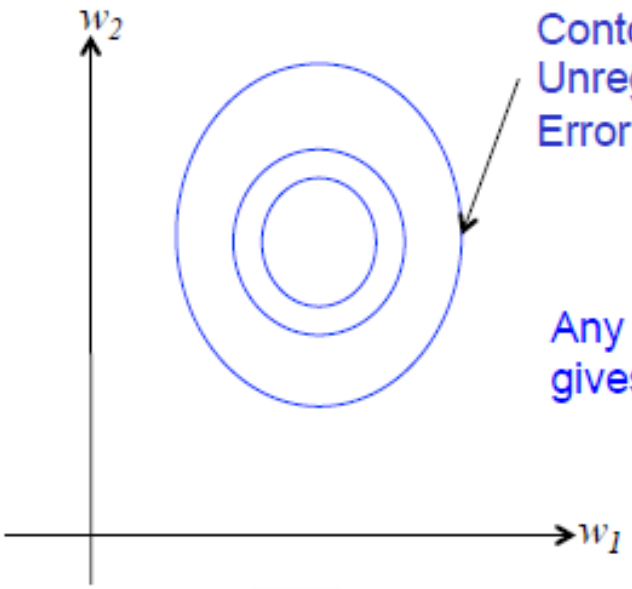
$$\mathrm{w} = (\lambda I + \Phi^T\Phi)^{-1}\Phi^T\mathrm{t}$$

- This is a simple extension of the least squared solution

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

Note that the problem of inverting the potentially singular matrix $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is averted as $\lambda\mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is full rank even if $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is not.

Geometric Interpretation of Regularizer



Contours of Unregularized Error function

In *unregularized* case:
we are trying to find $\mathbf{w}$ that minimizes

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

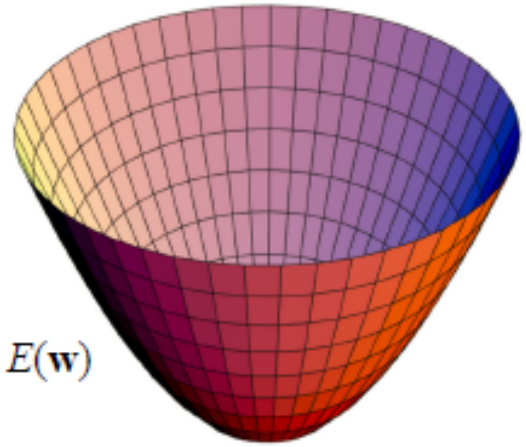
Any value of $\mathbf{w}$ on contour gives same error

In *regularized* case:
choose that value of $\mathbf{w}$ subject to the constraint

$$\sum_{j=1}^{M} |\, w_j \,|^2 \le \eta$$

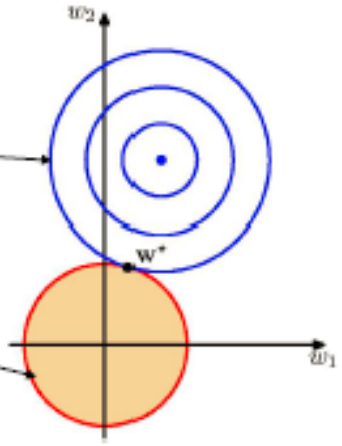
We don't want the weights to become too large
The two approaches related by Lagrange multipliers

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left\{ t_n - \mathbf{w}^T\phi(x_n) \right\}^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$



Minimization of Regularized Least Squares

- Blue: Contours of unregularized error function
- Constraint region
- w^* is optimum value



Minimization With quadratic Regularizer, $q=2$

A more general regularizer

- Regularized Error

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

- Where $q=2$ corresponds to the *quadratic* regularizer

  $q=1$ is known as *lasso*

- Lasso has the property that if λ is sufficiently large some of the coefficients w_j are driven to zero leading to a sparse model in which the corresponding basis functions play no role

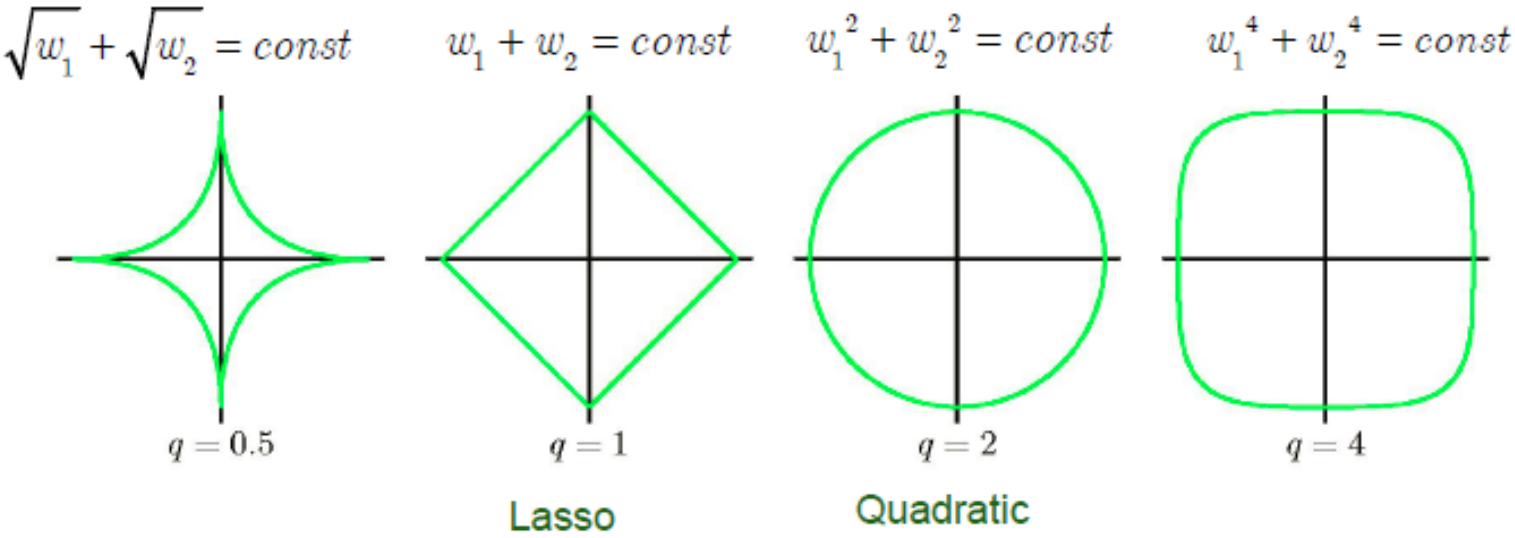
Contours of regularization term

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

- Contours of regularization term $|w_j|^q$ for values of q

Space of w_1, w_2

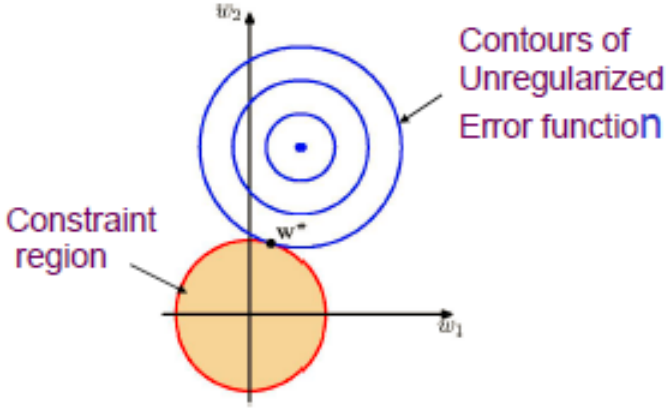
Any choice along the contour has the same value of $\sum_{j=1}^{M}| w_j |^q$



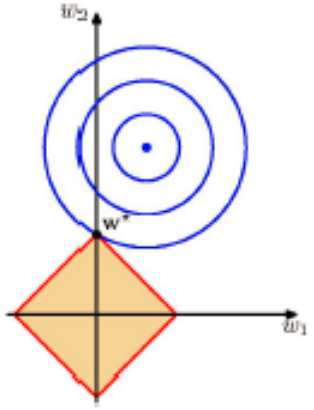
Sparsity with Lasso constraint

- With $q=1$ and λ is sufficiently large, some of the coefficients w_j are driven to zero
- Leads to a sparse model
  - where corresponding basis functions play no role
- Origin of sparsity is illustrated here:

Quadratic solution where w_1^* and w_0^* are nonzero



Minimization with Lasso Regularizer
A sparse solution with $w_1^*=0$



Regularization: Conclusion

- Regularization allows
  - complex models to be trained on small data sets
  - without severe over-fitting
- It limits model complexity
  - i.e., how many basis functions to use?
- Problem of limiting complexity is shifted to
  - one of determining suitable value of regularization coefficient

Outline

- Linear regression and least squares
- Regularized Least Squares
- The Bias-Variance Decomposition
- Bayesian Linear Regression

Recall the Decision Theory for Regression

the Squared Loss Function

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

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

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

$$y(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}]$$

The Bias-Variance Decomposition (1)

Recall the *expected squared loss*,

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

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)

a frequentist perspective

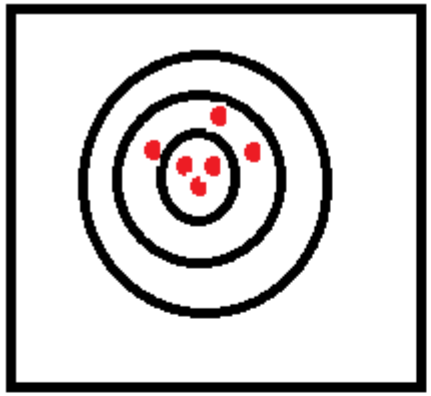
Suppose we were given multiple data sets, each of size N. Any particular data set, D, will give a particular function y(x;D). We then have

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

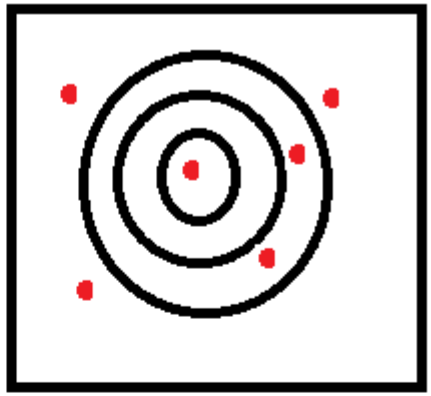
The Bias-Variance Decomposition (3)

Taking the expectation over D yields

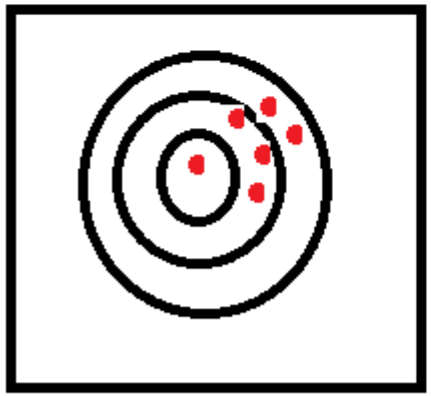
$$\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2\right]$$
$$= \underbrace{\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2}_{(\text{bias})^2} + \underbrace{\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right]}_{\text{variance}}.$$



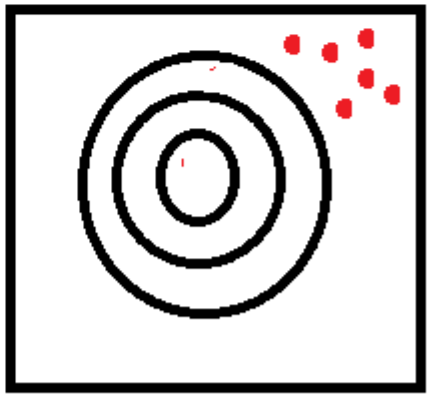
Small Variance -
Small Bias



Large Variance -
Small Bias



Small Variance -
Large Bias



Small Variance -
Huge Bias

The Bias-Variance Decomposition (4)

Thus we can write

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

where

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

Bias-variance decomposition

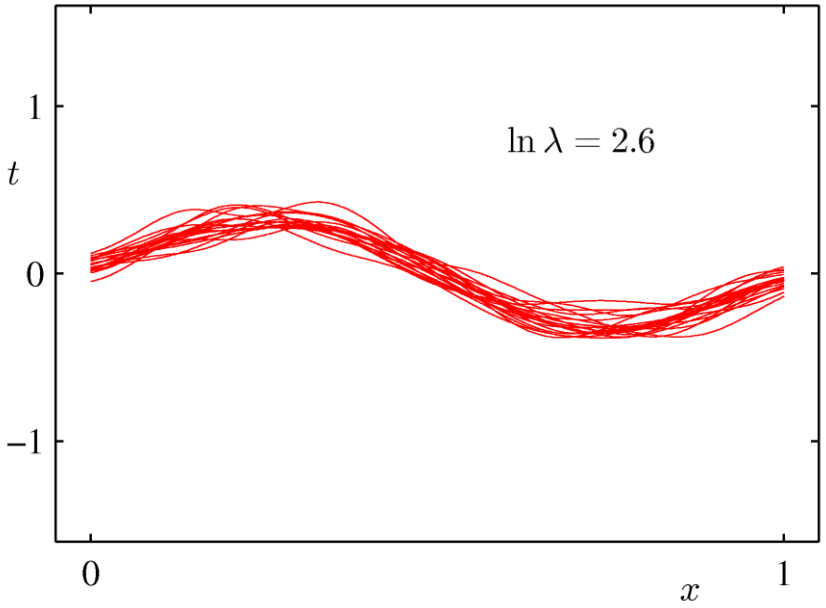
$Noise^2$ = lower bound on performance

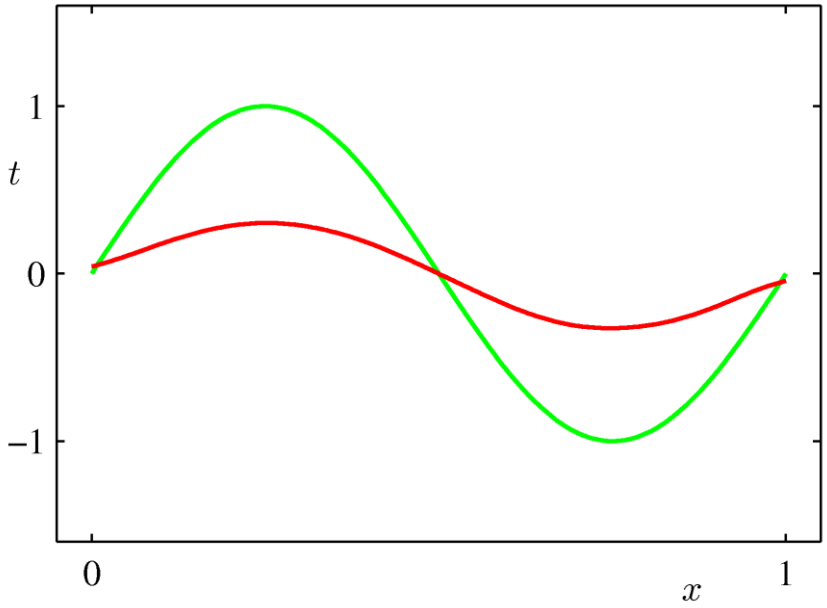
$Bias^2$ = (expected error due to model mismatch)^2

$Variance$ = variation due to train sample and randomization

The Bias-Variance Decomposition (5)

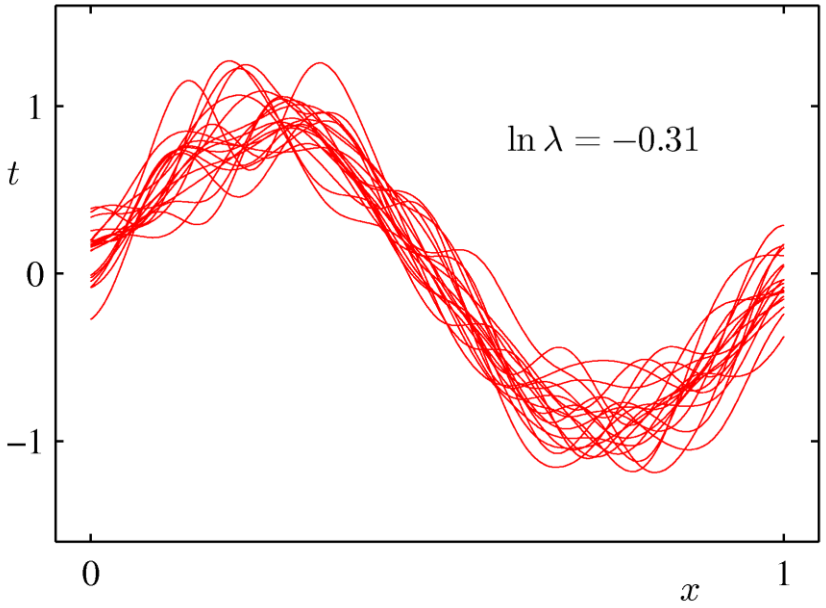
Example: 25 data sets from the sinusoidal, varying the degree of regularization, λ.

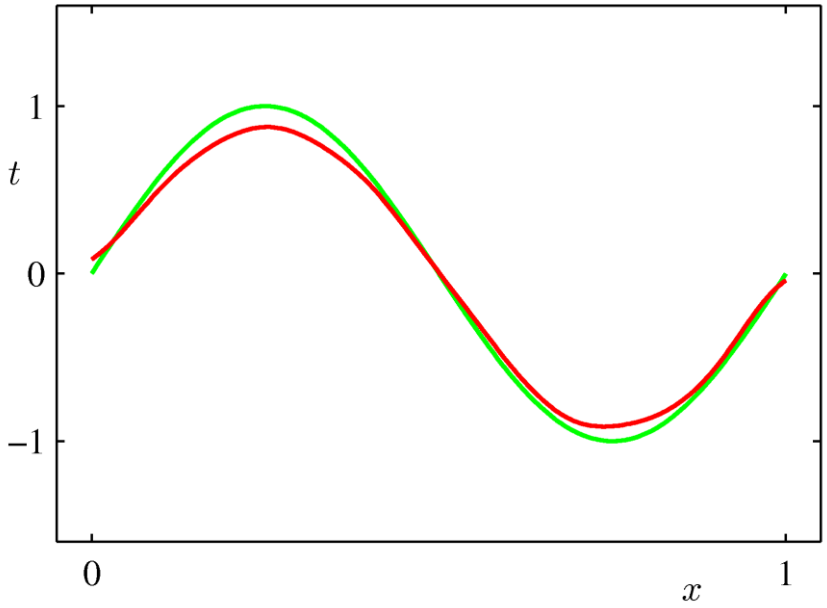




The Bias-Variance Decomposition (6)

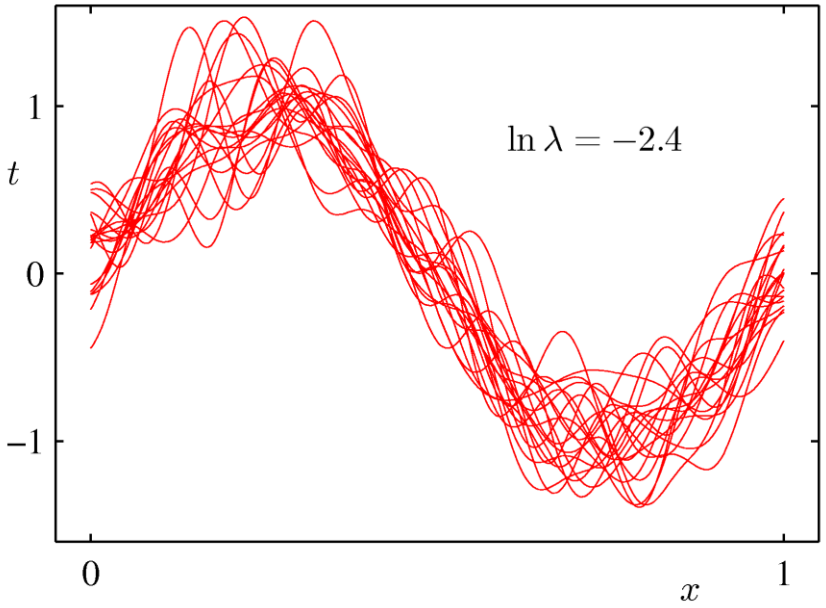
Example: 25 data sets from the sinusoidal, varying the degree of regularization, λ.

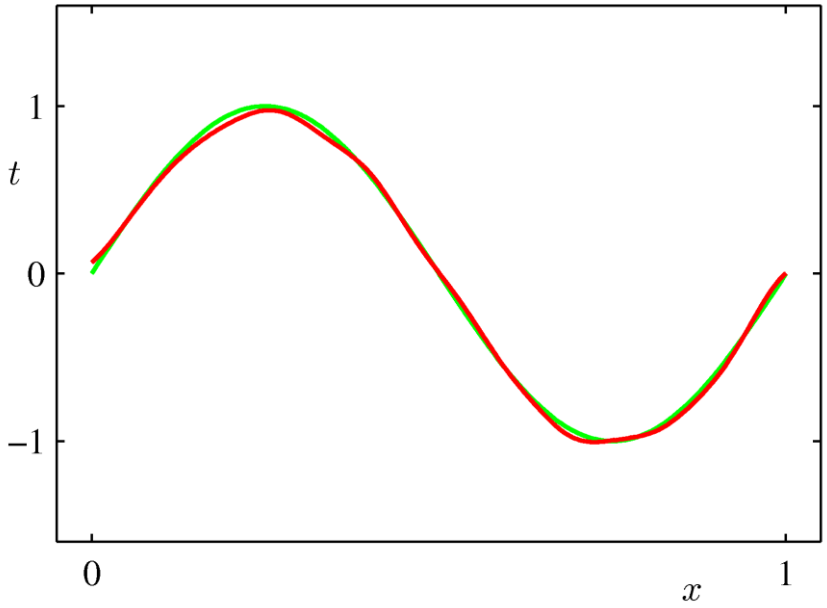




The Bias-Variance Decomposition (7)

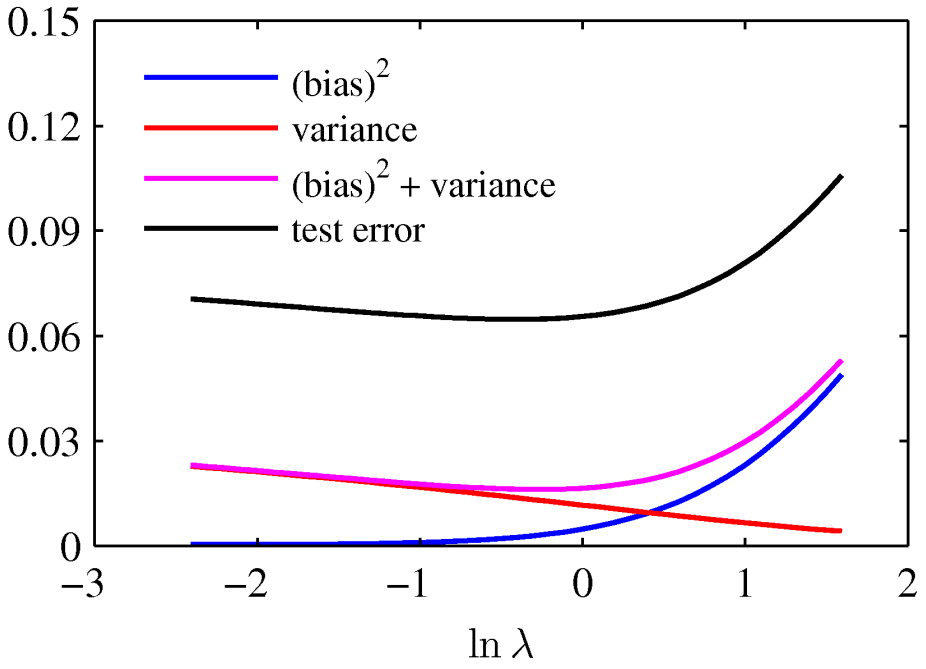
Example: 25 data sets from the sinusoidal, varying the degree of regularization, ¸.





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.



Bias

- Low bias
  - linear regression applied to linear data
  - 2nd degree polynomial applied to quadratic data
  - ANN with many hidden units trained to completion
- High bias
  - constant function
  - linear regression applied to non-linear data – ANN with few hidden units applied to non-linear data

Variance

- Low variance
  - constant function
  - model independent of training data
  - model depends on stable measures of data
    - mean
    - median
- High variance
  - high degree polynomial
  - ANN with many hidden units trained to completion

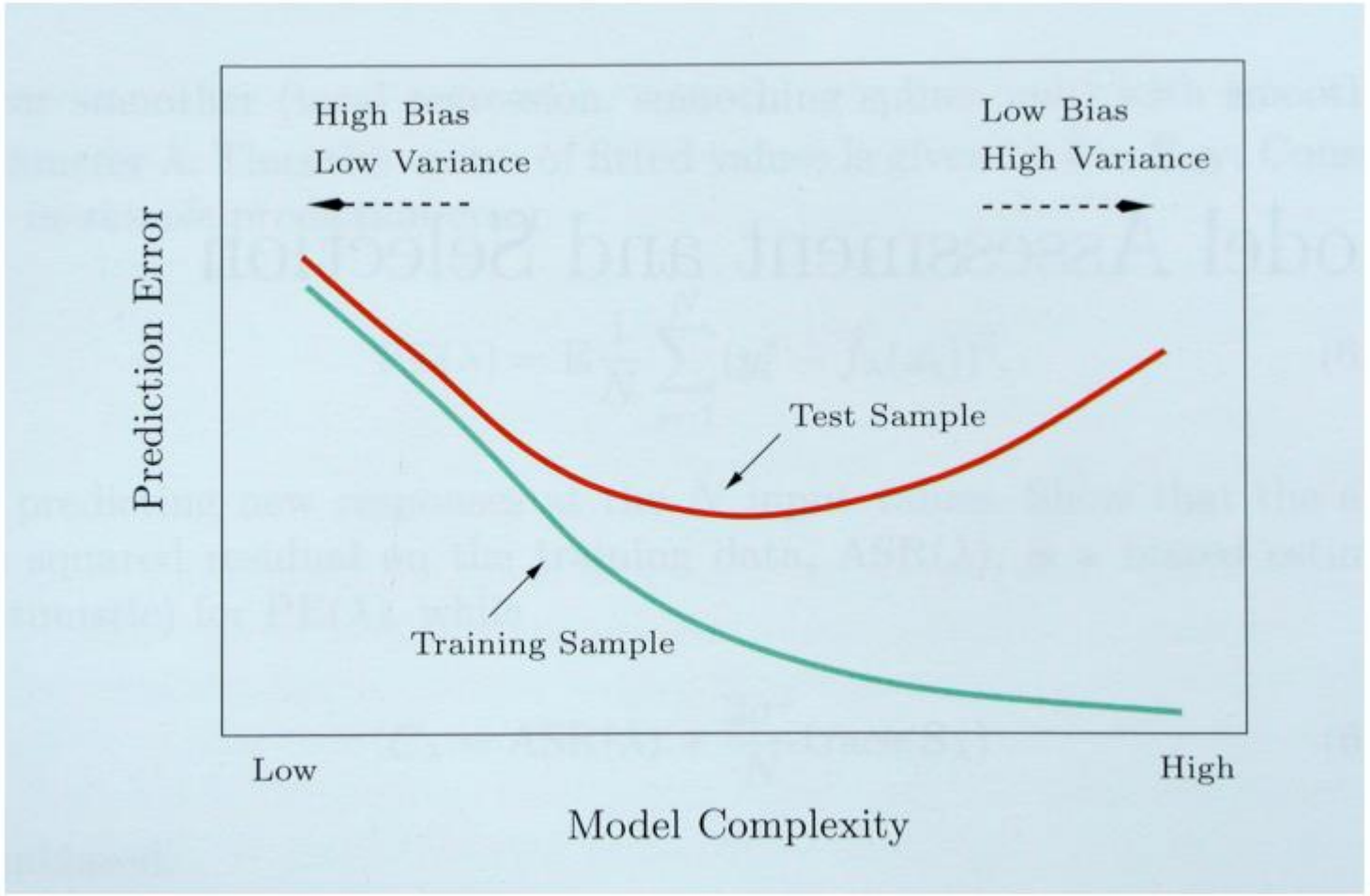
recall: Feature selection

- Feature selection: select a subset of "useful" features
  - Less features → less complex models
  - Lower "variance" in the risk of the learning method ☺
  - But might lead to higher "bias" ☹

Sources of Variance in Supervised Learning

- noise in targets or input attributes
- training sample
- randomness in learning algorithm
  - neural net weight initialization
- randomized subsetting of train set:
  - cross validation, train and early stopping set

Bias/Variance Tradeoff



Hastie, Tibshirani, Friedman "Elements of Statistical Learning" 2001

Bias/Variance Tradeoff

We need a good tradeoff between bias and variance

- Class of models are not too simple (so that we can ***approximate the true function well)***
- But not too complex to overfit the training samples (so that the ***estimation is stable)***

Outline

- Linear regression and least squares
- Regularized Least Squares
- The Bias-Variance Decomposition
- Bayesian Linear Regression

Bayesian Linear Regression

$$p(t_n \mid w, x_n) = \mathcal{N}(t_n \mid \phi(x_n)^T w, \beta^{-1}) = \sqrt{\frac{\beta}{2\pi}} \exp\left\{-\frac{\beta}{2}(t_n - \phi(x_n)^T w)^2\right\}$$

- Define a conjugate, Gaussian prior over w

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0).$$

- Combine this with Gaussian likelihood & using results for marginal and conditional Gaussian, the posterior equals

$$p(\mathbf{w}|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N)$$

$$\begin{aligned} \mathbf{m}_N &= \mathbf{S}_N\left(\mathbf{S}_0^{-1}\mathbf{m}_0 + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}\right) \\ \mathbf{S}_N^{-1} &= \mathbf{S}_0^{-1} + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}. \end{aligned}$$

- A common choice for the prior is:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

$$\begin{aligned} \mathbf{m}_N &= \beta\mathbf{S}_N\mathbf{\Phi}^{\mathrm{T}}\mathbf{t} \\ \mathbf{S}_N^{-1} &= \alpha\mathbf{I} + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}. \end{aligned}$$

Bayesian Regression & Regularization

- Because the posterior is Gaussian, the posterior mean is equal to the posterior mode (MAP estimate):

$$\hat{w} = \arg\min_w -\log p(w \mid t) = \arg\min_w -\log p(w) - \log p(t \mid w)$$

$$-\sum_{n=1}^{N} \log p(t_n \mid w, x_n) = -\frac{N}{2}\log\frac{\beta}{2\pi} + \frac{\beta}{2}\sum_{n=1}^{N}(t_n - \phi(x_n)^T w)^2$$

$$-\log p(w) = -\frac{M}{2}\log\frac{\alpha}{2\pi} + \frac{\alpha}{2}\sum_{m=1}^{M} w_m^2$$

$$\hat{w} = \arg\min_w ||t - \Phi w||^2 + \frac{\alpha}{\beta}||w||^2$$

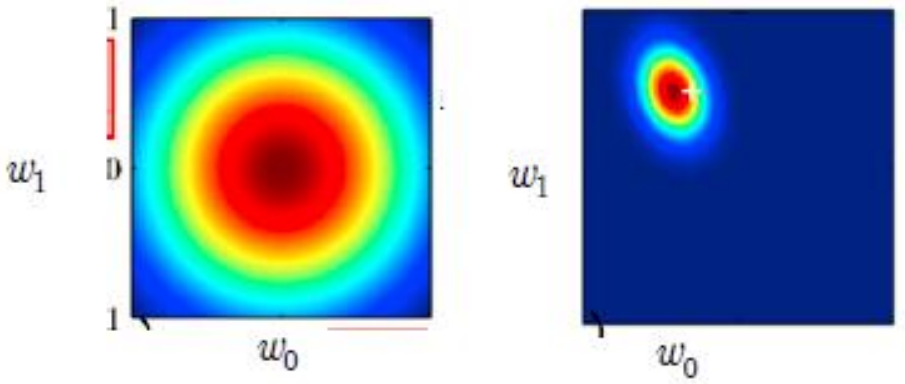
Bayesian Linear Regression

- A common choice for the prior is:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$$

$$\begin{aligned} \mathbf{m}_N &= \beta \mathbf{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} \\ \mathbf{S}_N^{-1} &= \alpha \mathbf{I} + \beta \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}. \end{aligned}$$

Prior $p(\boldsymbol{w} \mid \alpha) = N(\boldsymbol{w} \mid \mathbf{0}, \alpha^{-1} I)$
and Posterior in weight space
for scalar input x and
$y(x, \boldsymbol{w}) = w_0 + w_1 x$



Properties of Posterior

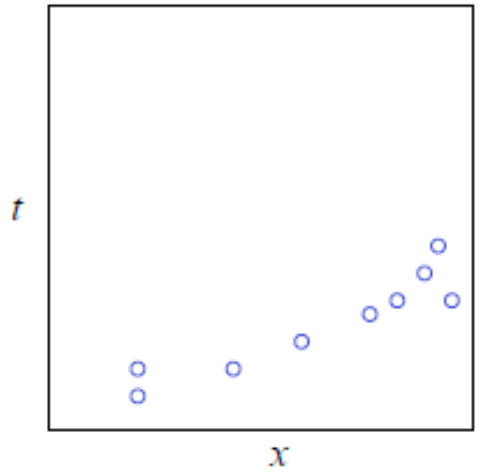
1. Since posterior $p(\boldsymbol{w}|\boldsymbol{t}) = N(\boldsymbol{w}|\boldsymbol{m}_N, \mathrm{S}_N)$ is Gaussian its mode coincides with its mean
   - Thus maximum posterior weight is $\boldsymbol{w}_{\mathrm{MAP}} = \boldsymbol{m}_N$
2. Infinitely broad prior $\mathrm{S}_0 = \alpha^{-1}\mathrm{I}$, i.e., precision $\alpha \to 0$
   - Then mean $\boldsymbol{m}_N$ reduces to the maximum likelihood value, i.e., mean is the solution vector

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

3. If $N = 0$, posterior reverts to the prior
4. If data points arrive sequentially, then posterior to any stage acts as prior distribution for subsequent data points

Example (Straight Line Fit)

- Single input variable x
- Single target variable t
- Goal is to fit
  - Linear model $y(x, \boldsymbol{w}) = w_0 + w_1 x$

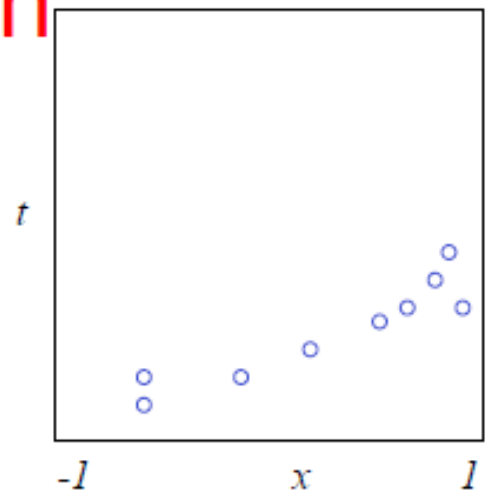


- Goal of Linear Regression is to recover

  $\boldsymbol{w} = [w_0, w_1]$ given the samples

Data Generation

- Synthetic data generated from $f(x,\boldsymbol{w})=w_0+w_1x$ with parameter values $w_0=$ -0.3 and $w_1=0.5$

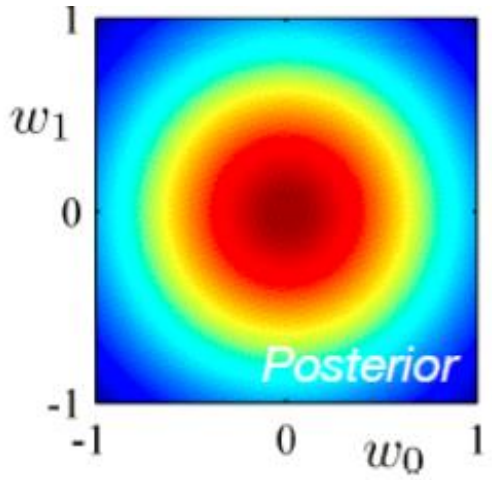


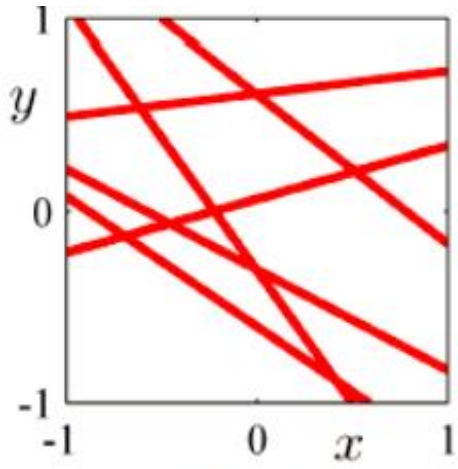
  - First choose x_n from $\mathrm{U}(x|\text{-}1,1)$, then evaluate $f(x_n,\boldsymbol{w})$
  - Add Gaussian noise with st dev 0.2 to get target t_n
    - Precision parameter $\beta = (1/0.2)^2 = 25$

- For prior over $\boldsymbol{w}$ we choose $\alpha = 2$

$$p(\boldsymbol{w} \mid \alpha) = N(\boldsymbol{w} \mid \mathbf{0}, \alpha^{-1} I)$$

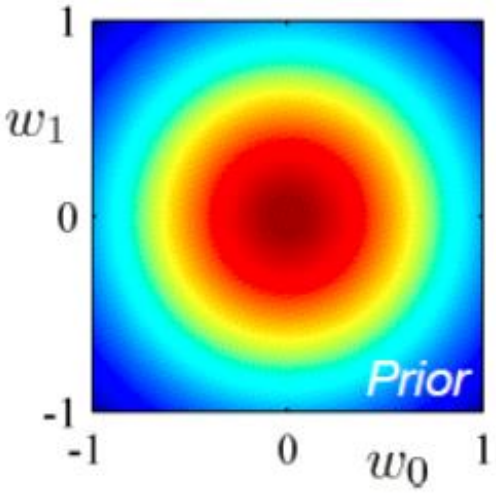
Bayesian Regression: 0 Observations



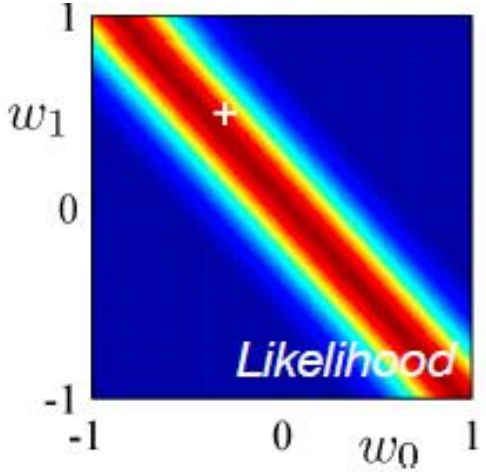


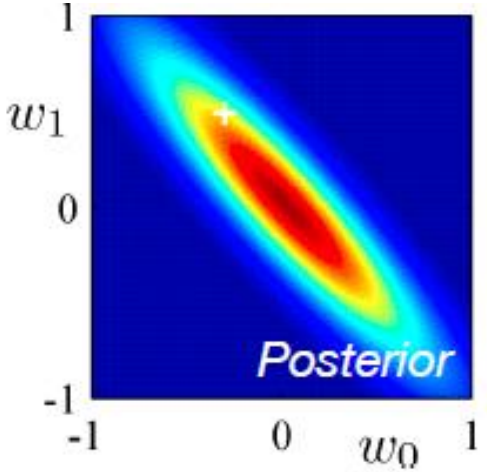
Data Space

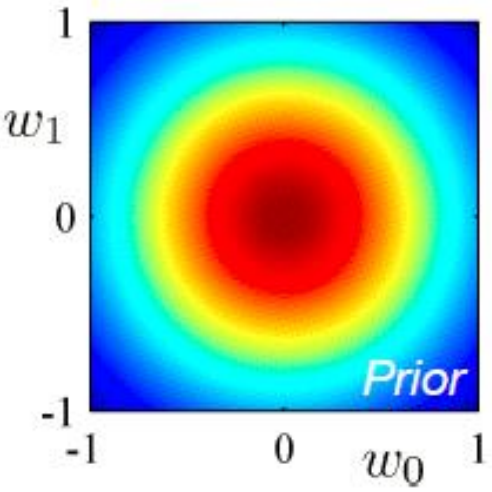
$$y = w_0 + w_1 x$$

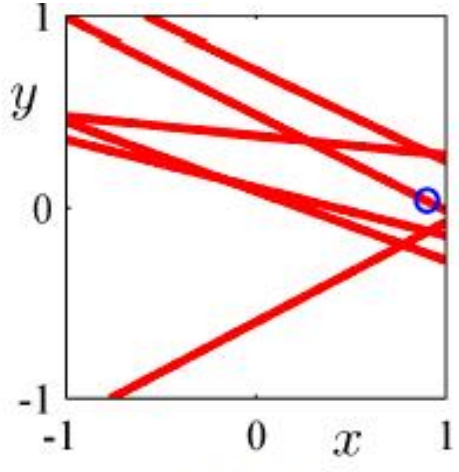


Bayesian Regression: 1 Observations





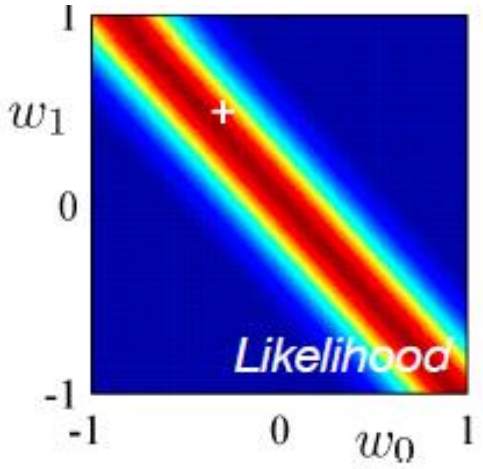


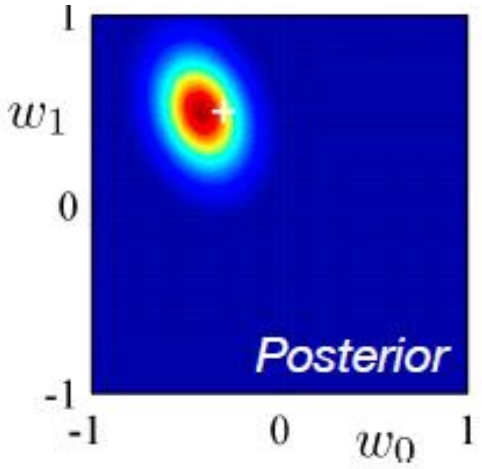


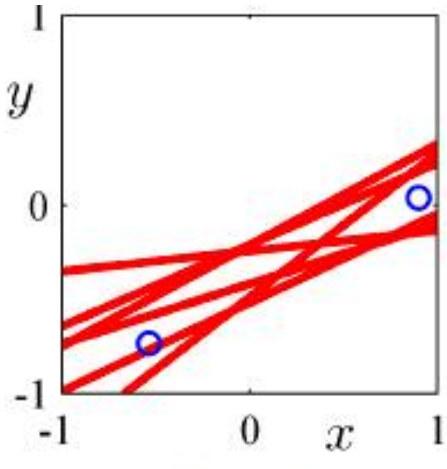
Data Space

$$y = w_0 + w_1 x$$

Bayesian Regression: 2 Observations

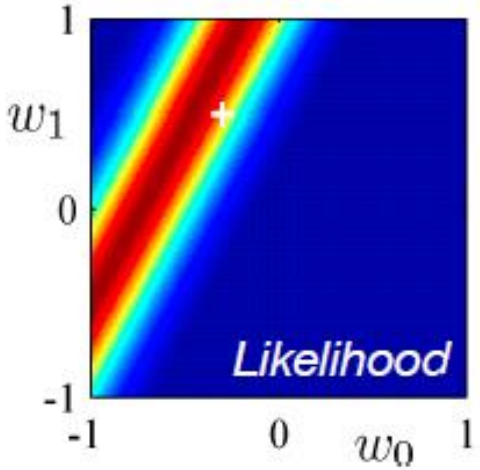


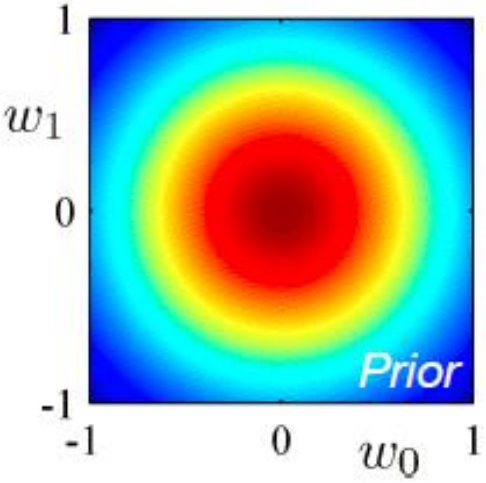




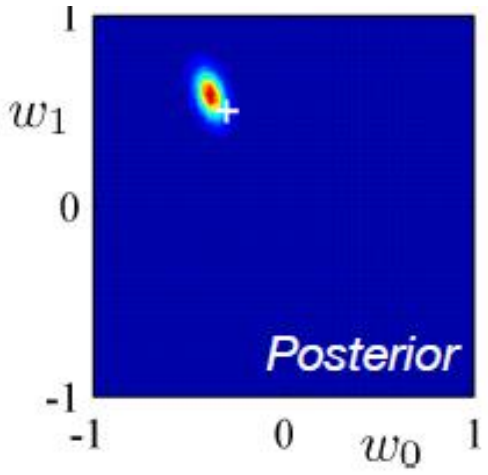
Data Space

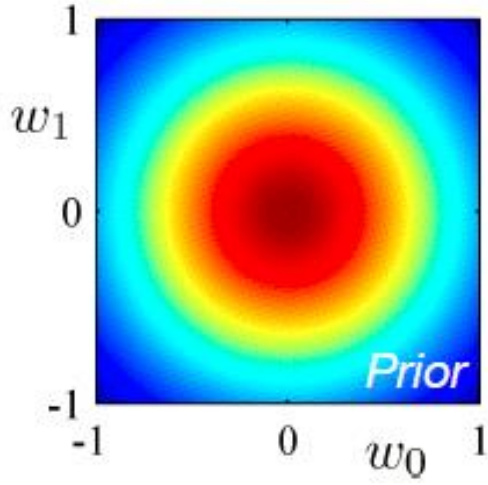
$$y = w_0 + w_1 x$$

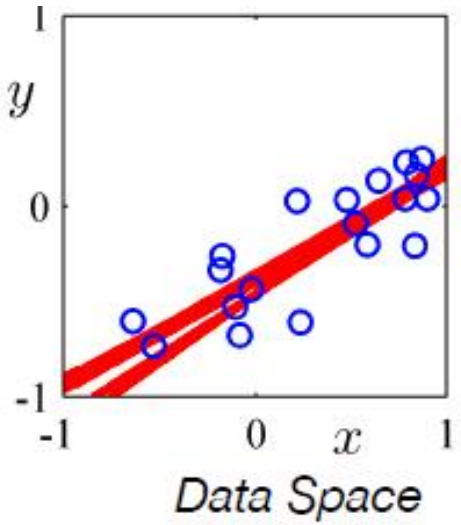




Bayesian Regression: 20 Observations







$$y = w_0 + w_1 x$$

Generalization of Gaussian prior

- Other prior yields Lasso and variations:

$$p(\mathbf{w} \mid \alpha) = \left[\frac{q}{2} \left(\frac{\alpha}{2} \right)^{1/q} \frac{1}{\Gamma(1/q)} \right]^M \exp\left(-\frac{\alpha}{2} \sum_{j=1}^{M} | w_j |^q \right)$$

  - $q=2$ corresponds to Gaussian
  - Corresponds to minimization of regularized error function

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

q=1, Lasso Regression

Predictive Distribution

- Usually not interested in the value of $\boldsymbol{w}$ itself
- But predicting t for a new value of x

$$p(t|\mathbf{t},\mathrm{X},\boldsymbol{x}) \text{ or}$$

$$p(t|\mathbf{t})$$

  - Leaving out conditioning variables X and $\boldsymbol{x}$ for convenience
- Marginalizing over parameter variable $\boldsymbol{w}$, is the standard Bayesian approach
  - Sum rule of probability $\boxed{p(t)=\int p(t,\boldsymbol{w})\,d\boldsymbol{w}=\int p(t|\boldsymbol{w})p(\boldsymbol{w})\,d\boldsymbol{w}}$
  - We can now write $\boxed{p(t\mid \boldsymbol{t})=\int p(t|\boldsymbol{w})p(\boldsymbol{w}|\boldsymbol{t})\,d\boldsymbol{w}}$

Predictive Distribution

- We can predict t for a new value of $\boldsymbol{x}$ using

$$p(t \mid \boldsymbol{t}) = \int p(t \mid \boldsymbol{w}) p(\boldsymbol{w} \mid \boldsymbol{t})\, d\boldsymbol{w}$$

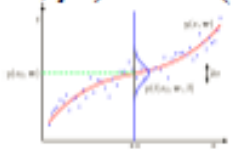
We have left out conditioning variables $\mathbf{X}$ and $\boldsymbol{w}$ for convenience. Also we have applied sum rule of probability $p(t) = \Sigma_{\boldsymbol{w}} p(t \mid \boldsymbol{w}) p(\boldsymbol{w})$

  - With explicit dependence on prior parameter α, noise parameter β, & targets in training set $\boldsymbol{t}$

$$p(t \mid \mathbf{t}, \alpha, \beta) = \int \underbrace{p(t \mid \mathrm{w}, \beta)} \cdot \underbrace{p(\mathrm{w} \mid \mathbf{t}, \alpha, \beta)}\, d\mathrm{w}$$

Conditional of target t given weight $\boldsymbol{w}$

$$p(t \mid \boldsymbol{x}, \boldsymbol{w}, \beta) = N(t \mid y(\boldsymbol{x}, \boldsymbol{w}), \beta^{-1})$$



posterior of weight $\boldsymbol{w}$

$$p(\boldsymbol{w} \mid \boldsymbol{t}) = N(\boldsymbol{w} \mid \boldsymbol{m}_N, \mathrm{S}_N)$$

where

$$\boldsymbol{m}_N = \beta \mathrm{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{t}$$

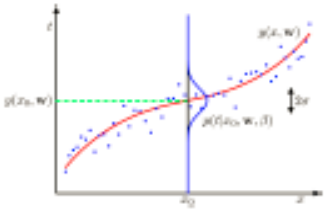
$$\mathrm{S}_N^{-1} = \alpha \mathrm{I} + \beta \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}$$

Variance of Predictive Distribution

- Predictive distribution is a Gaussian:

$$p(t \mid \boldsymbol{x}, \mathbf{t}, \alpha, \beta) = N(t \mid m_N^T \boldsymbol{\phi}(\boldsymbol{x}), \sigma_N^2(\boldsymbol{x}))$$

$$\text{where} \quad \sigma_N^2(\boldsymbol{x}) = \underbrace{\frac{1}{\beta}}_{\text{Noise in data}} + \underbrace{\boldsymbol{\phi}(\boldsymbol{x})^T S_N \boldsymbol{\phi}(\boldsymbol{x})}_{\text{Uncertainty associated with parameters } \boldsymbol{w}}$$



where $S_N^{-1} = \alpha \mathrm{I} + \beta \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}$ is the covariance of $p(\boldsymbol{w}|\alpha)$

Posterior Predictive Distribution

- Posterior distribution given training data is Gaussian:

$$p(w \mid t_{\text{train}}, x_{\text{train}}) = \mathcal{N}(w \mid m_N, S_N)$$

- Predictive distribution is then also Gaussian:

$$p(t_{\text{test}} \mid x_{\text{test}}, t_{\text{train}}, x_{\text{train}}) =$$

$$= \int \mathcal{N}(t_{\text{test}} \mid \phi(x_{\text{test}})^T w, \beta^{-1}) \mathcal{N}(w \mid m_N, S_N)\, dw$$

$$= \mathcal{N}(t_{\text{test}} \mid \phi(x_{\text{test}})^T m_N, \beta^{-1} + \phi(x_{\text{test}})^T S_N \phi(x_{\text{test}}))$$

- How does this prediction relate to the MAP estimate?

$$\hat{w} = \arg\max_w \; p(w \mid t_{\text{train}}, x_{\text{train}}) = m_N$$

$$p(t_{\text{test}} \mid x_{\text{test}}, \hat{w}) = \mathcal{N}(t_{\text{test}} \mid \phi(x_{\text{test}})^T m_N, \beta^{-1})$$

Example of Predictive Distribution

- Data generated from $\sin(2\pi x)$
- Model: nine Gaussian basis functions

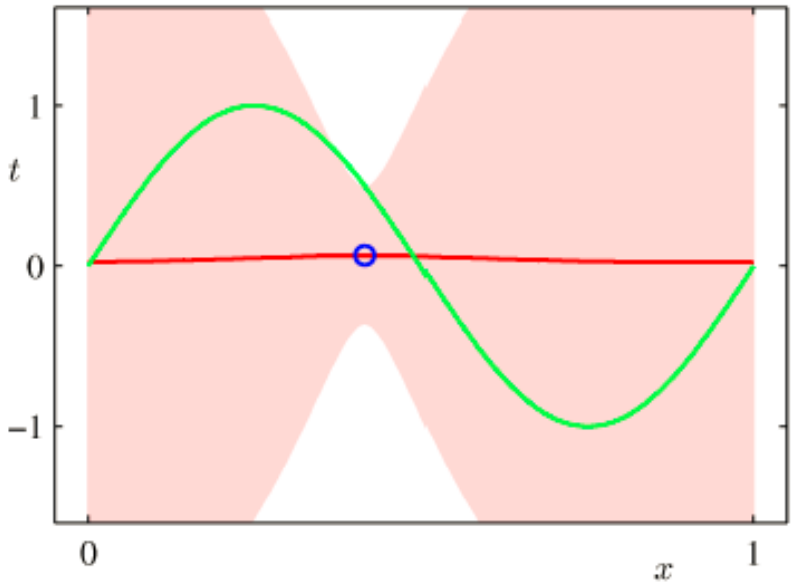
$$y(x,\mathbf{w}) = \sum_{j=0}^{8} w_j \phi_j(x) = \mathbf{w}^T \boldsymbol{\phi}(x)$$

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

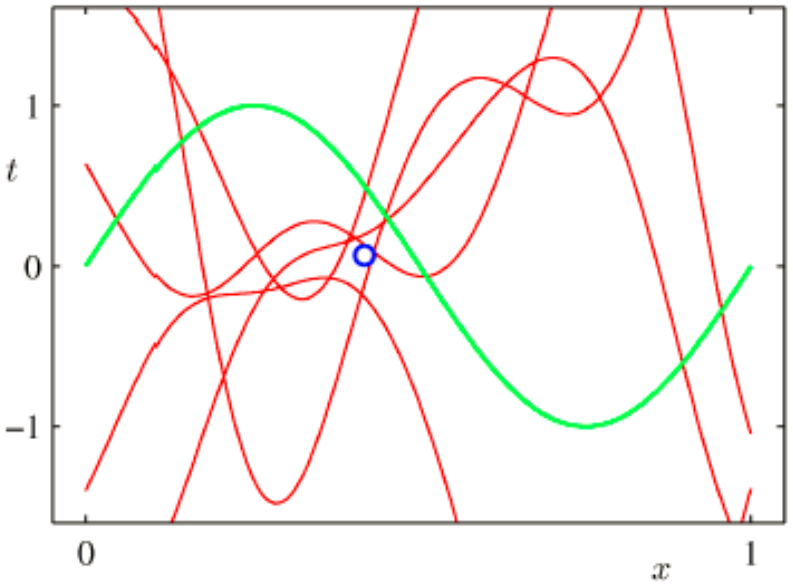
- Predictive distribution

$$p(t \mid x, \mathbf{t}, \alpha, \beta) = N(t \mid \mathbf{m}_N^T \boldsymbol{\phi}(x), \sigma_N^2(x)) \quad \text{where} \quad \sigma_N^2(x) = \frac{1}{\beta} + \boldsymbol{\phi}(x)^T \mathbf{S}_N \boldsymbol{\phi}(x)$$

Predictive Distribution: *N=1 Observation*



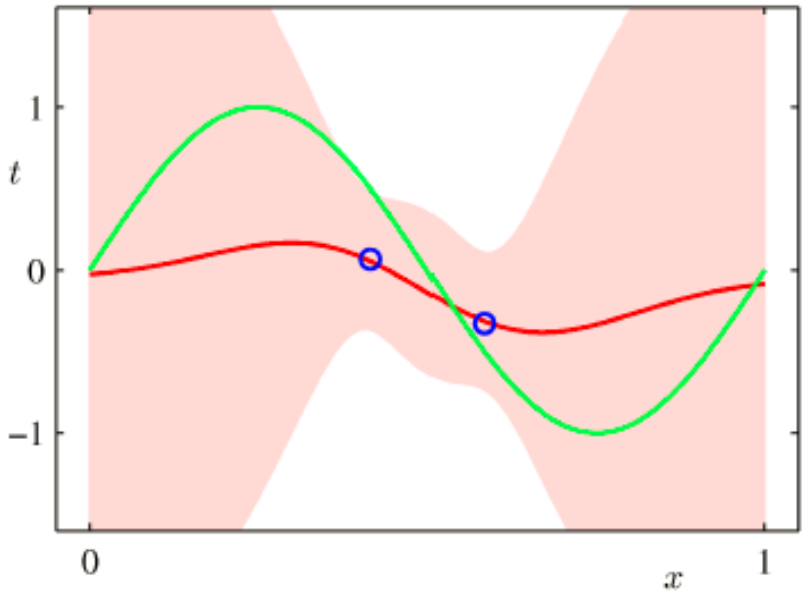
Mean plus/minus standard deviation



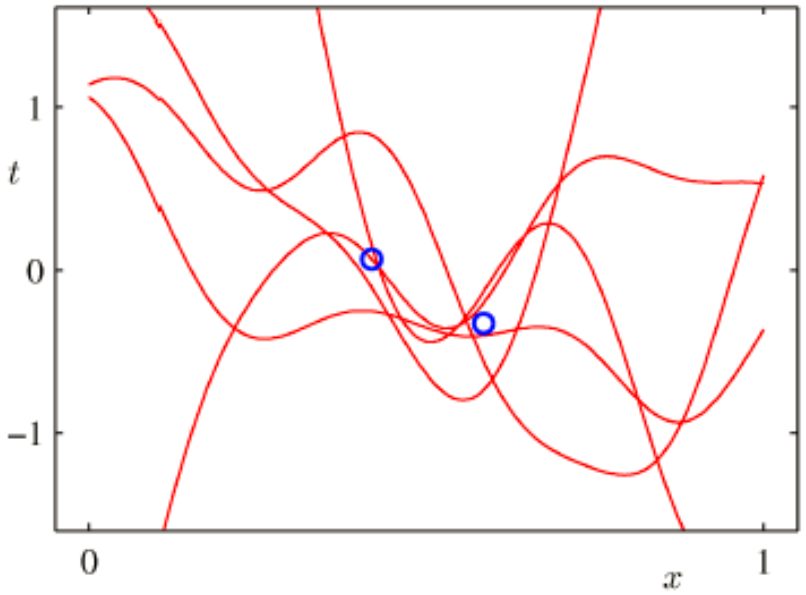
5 samples from predictive posterior

Example: Sinusoidal data, M=9 radial basis functions

Predictive Distribution: *N=2 Observations*



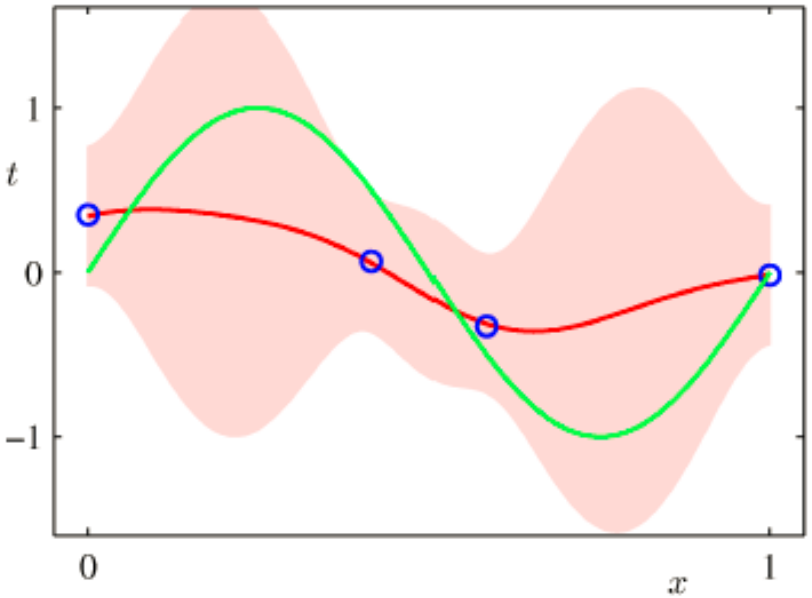
Mean plus/minus standard deviation



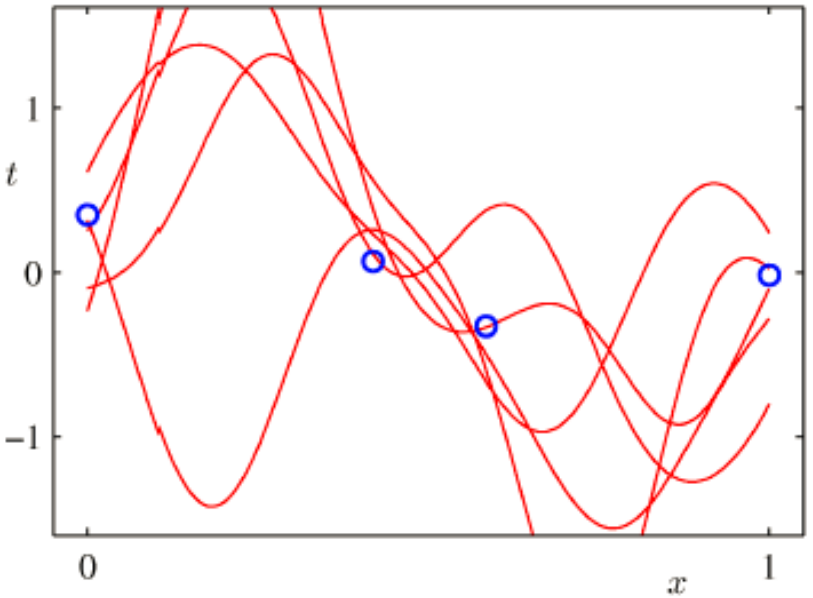
5 samples from predictive posterior

Example: Sinusoidal data, M=9 radial basis functions

Predictive Distribution: *N=4 Observations*



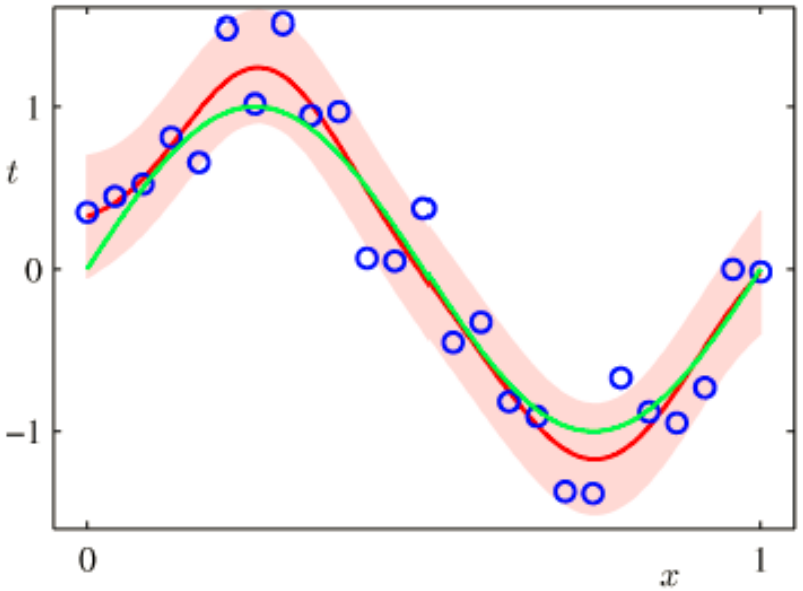
Mean plus/minus standard deviation



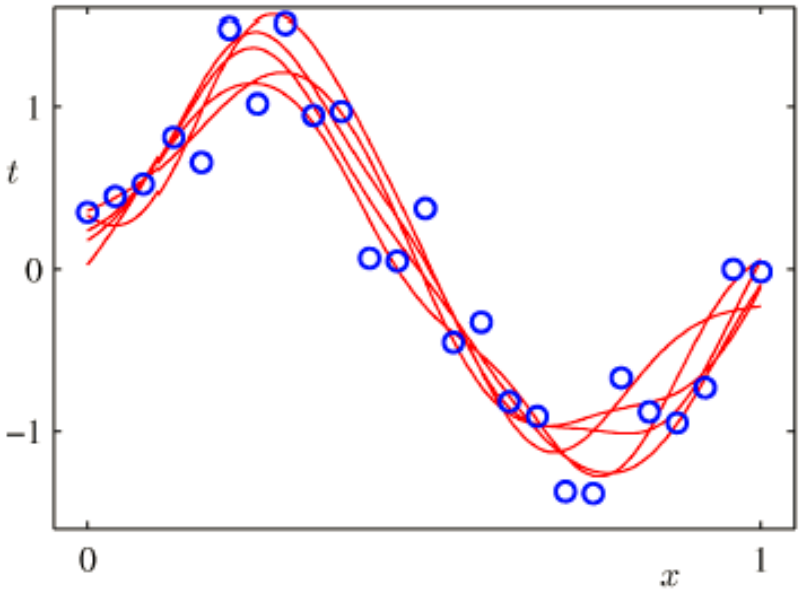
5 samples from predictive posterior

Example: Sinusoidal data, M=9 radial basis functions

Predictive Distribution: *N=25 Observations*



Mean plus/minus standard deviation



5 samples from predictive posterior

Example: Sinusoidal data, M=9 radial basis functions

本章阅读

- Bishop Ch. 3.1, 3.2