Linear Models of Regression

K Sri Rama Murty

IIT Hyderabad

ksrm@ee.iith.ac.in

January 28, 2022

Regression

Regression

- Predict target variable(s) $t \in \mathbb{R}$ given D-dimensional input vector $\mathbf{x}$

Regression

- Predict target variable(s) $t \in \mathbb{R}$ given D-dimensional input vector $\mathbf{x}$
- E.g. Weight estimation, Share market prediction, 3D image from 2D

Regression

- Predict target variable(s) $t \in \mathbb{R}$ given D-dimensional input vector $\mathbf{x}$
- E.g. Weight estimation, Share market prediction, 3D image from 2D
- Target can be estimated as a linear combination of inputs

$$\hat{t} = y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots w_D x_D = \mathbf{w}^\mathsf{T}\mathbf{x}$$

$$\mathbf{x} = [1\ x_1\ x_2\ \cdots\ x_D]^\mathsf{T} \qquad \mathbf{w} = [w_0\ w_1\ w_2\ \cdots w_D]^\mathsf{T}$$

Regression

- Predict target variable(s) $t \in \mathbb{R}$ given D-dimensional input vector $\mathbf{x}$
- E.g. Weight estimation, Share market prediction, 3D image from 2D
- Target can be estimated as a linear combination of inputs

$$\hat{t} = y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots w_D x_D = \mathbf{w}^\mathsf{T}\mathbf{x}$$

$$\mathbf{x} = [1\ x_1\ x_2\ \cdots\ x_D]^\mathsf{T} \qquad \mathbf{w} = [w_0\ w_1\ w_2\ \cdots w_D]^\mathsf{T}$$

- Determine the model parameters $\mathbf{w}$ to minimize error on labeled training data

$$\mathcal{S} = \{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \cdots (\mathbf{x}_N, t_N)\}$$

Regression

- Predict target variable(s) $t \in \mathbb{R}$ given D-dimensional input vector $\mathbf{x}$
- E.g. Weight estimation, Share market prediction, 3D image from 2D
- Target can be estimated as a linear combination of inputs

$$\hat{t} = y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots w_D x_D = \mathbf{w}^\mathsf{T}\mathbf{x}$$

$$\mathbf{x} = [1 \ x_1 \ x_2 \ \cdots \ x_D]^\mathsf{T} \qquad \mathbf{w} = [w_0 \ w_1 \ w_2 \ \cdots w_D]^\mathsf{T}$$

- Determine the model parameters $\mathbf{w}$ to minimize error on labeled training data

$$\mathcal{S} = \{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \cdots (\mathbf{x}_N, t_N)\}$$

- Need to define a loss function for optimizing model parameters $\mathbf{w}$

Least Squares Criterion to Determine $\mathbf{w}$

Least Squares Criterion to Determine $\mathbf{w}$

- Estimated target of n^{th} example

$$\hat{t}_n = y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_n$$

Least Squares Criterion to Determine $\mathbf{w}$

- Estimated target of n^{th} example

$$\hat{t_n} = y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_n$$

- Error in estimation

$$e_n = t_n - y_n \quad n = 1, 2, \cdots N$$

Least Squares Criterion to Determine $\mathbf{w}$

- Estimated target of n^{th} example

$$\hat{t}_n = y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_n$$

- Error in estimation

$$e_n = t_n - y_n \quad n = 1, 2, \cdots N$$

- Overall error on training set

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} e_n^2$$

Least Squares Criterion to Determine w

- Estimated target of n^{th} example

$$\hat{t_n} = y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_n$$

- Error in estimation

$$e_n = t_n - y_n \quad n = 1, 2, \cdots N$$

- Overall error on training set

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} e_n^2$$

- Estimate **w** to minimize $J(\mathbf{w})$

$$\mathbf{w}_* = \arg\min_{w} J(\mathbf{w})$$

Least Squares Criterion to Determine $\mathbf{w}$

- Estimated target of n^{th} example

$$\hat{t}_n = y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_n$$

- Error in estimation

$$e_n = t_n - y_n \quad n = 1, 2, \cdots N$$

- Overall error on training set

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} e_n^2$$

- Estimate $\mathbf{w}$ to minimize $J(\mathbf{w})$

$$\mathbf{w}_* = \arg\min_{w} J(\mathbf{w})$$

- Formulating in matrix notation

$$\mathbf{y}_{N\times 1} = \mathbf{X}_{N\times D+1}\mathbf{w}_{D+1x1}$$
$$\mathbf{X} = [\mathbf{x}_1\,\mathbf{x}_2\ \cdots\ \mathbf{x}_N]^\mathsf{T}$$

Least Squares Criterion to Determine $\mathbf{w}$

- Estimated target of n^{th} example

$$\hat{t_n} = y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_n$$

- Error in estimation

$$e_n = t_n - y_n \quad n = 1, 2, \cdots N$$

- Overall error on training set

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} e_n^2$$

- Estimate $\mathbf{w}$ to minimize $J(\mathbf{w})$

$$\mathbf{w}_* = \arg\min_{w} J(\mathbf{w})$$

- Formulating in matrix notation

$$\mathbf{y}_{N\times 1} = \mathbf{X}_{N\times D+1}\mathbf{w}_{D+1x1}$$
$$\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_N]^\mathsf{T}$$

- LSE can be expressed as

$$J(\mathbf{w}) = \frac{1}{2}\mathrm{Tr}[(\mathbf{t}-\mathbf{y})(\mathbf{t}-\mathbf{y})^\mathsf{T}]$$

Least Squares Criterion to Determine $\mathbf{w}$

- Estimated target of n^{th} example

$$\hat{t}_n = y(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_n$$

- Error in estimation

$$e_n = t_n - y_n \quad n = 1, 2, \cdots N$$

- Overall error on training set

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} e_n^2$$

- Estimate $\mathbf{w}$ to minimize $J(\mathbf{w})$

$$\mathbf{w}_* = \arg\min_{w} J(\mathbf{w})$$

- Formulating in matrix notation

$$\mathbf{y}_{N\times 1} = \mathbf{X}_{N\times D+1}\mathbf{w}_{D+1x1}$$
$$\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \cdots\ \mathbf{x}_N]^\mathsf{T}$$

- LSE can be expressed as

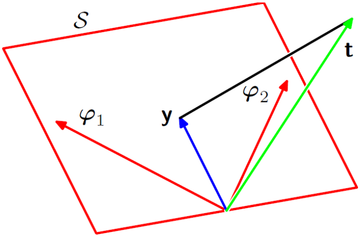
$$J(\mathbf{w}) = \frac{1}{2}\mathrm{Tr}[(\mathbf{t}-\mathbf{y})(\mathbf{t}-\mathbf{y})^\mathsf{T}]$$

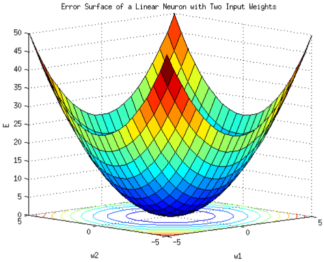
- Equating derivative w.r.t $\mathbf{w}$ to 0

$$\nabla_{\mathbf{w}} J(\mathbf{w}) = \nabla_{\mathbf{y}} J(\mathbf{w})\ \nabla_{\mathbf{w}}\mathbf{y}$$
$$= \mathbf{X}^\mathsf{T}(\mathbf{t} - \mathbf{X}\mathbf{w}) = \mathbf{0}$$

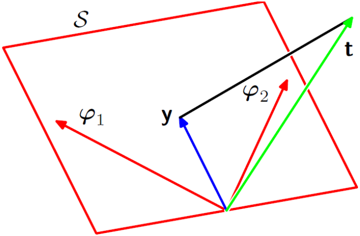
$$\boxed{\mathbf{w} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{t}}$$

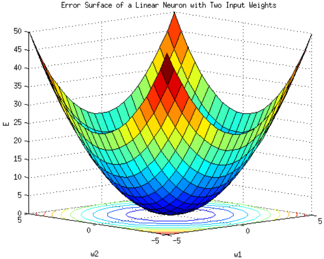
Geometric Interpretation of Least Squares





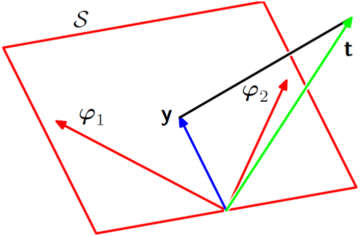
Geometric Interpretation of Least Squares

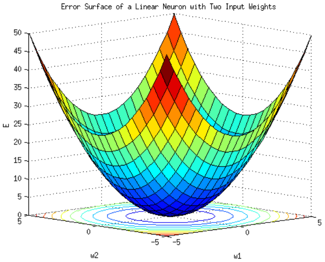




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$

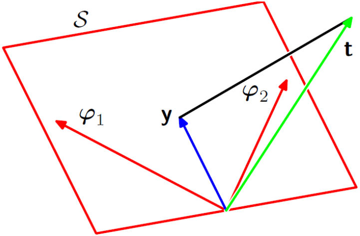
Geometric Interpretation of Least Squares

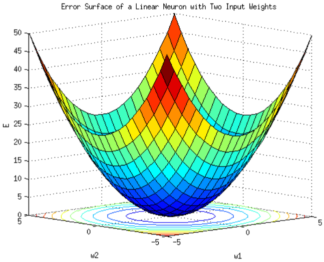




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$
- Let $\mathcal{S}$ denote a subspace spanned by columns of X in N-dim space

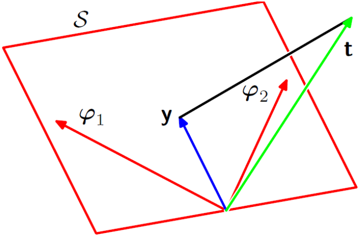
Geometric Interpretation of Least Squares

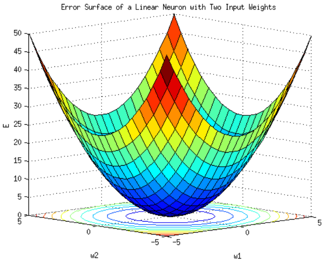




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$
- Let $\mathcal{S}$ denote a subspace spanned by columns of X in N-dim space
- $\mathbf{y} = \mathbf{Xw} \in \mathcal{S}$, being a linear combination of columns of $\mathbf{X}$

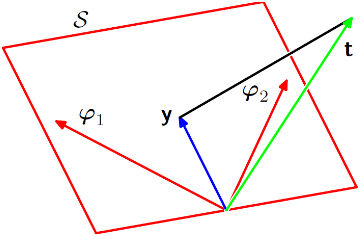
Geometric Interpretation of Least Squares

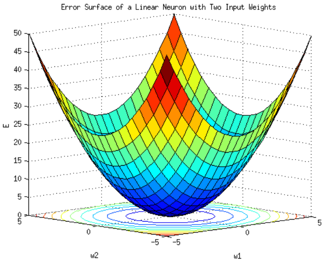




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$
- Let $\mathcal{S}$ denote a subspace spanned by columns of X in N-dim space
- $\mathbf{y} = \mathbf{X}\mathbf{w} \in \mathcal{S}$, being a linear combination of columns of $\mathbf{X}$
- For the LS optimality criterion

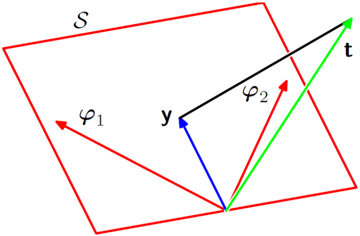
Geometric Interpretation of Least Squares

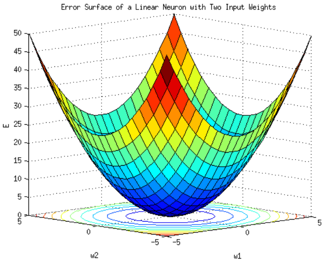




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$
- Let $\mathcal{S}$ denote a subspace spanned by columns of X in N-dim space
- $\mathbf{y} = \mathbf{Xw} \in \mathcal{S}$, being a linear combination of columns of $\mathbf{X}$
- For the LS optimality criterion
  - $\mathbf{y}$ is orthogonal projection of $\mathbf{t}$ on $\mathcal{S}$

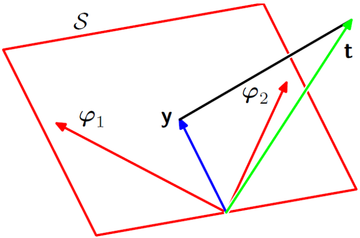
Geometric Interpretation of Least Squares

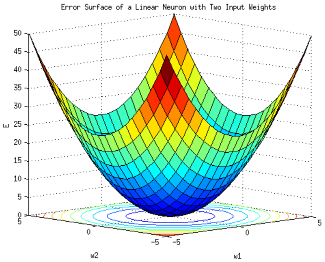




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$
- Let $\mathcal{S}$ denote a subspace spanned by columns of X in N-dim space
- $\mathbf{y} = \mathbf{X}\mathbf{w} \in \mathcal{S}$, being a linear combination of columns of $\mathbf{X}$
- For the LS optimality criterion
  - $\mathbf{y}$ is orthogonal projection of $\mathbf{t}$ on $\mathcal{S}$
  - Error surface $J(\mathbf{w})$ is convex

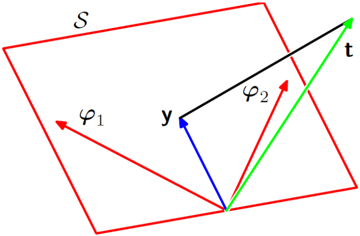
Geometric Interpretation of Least Squares

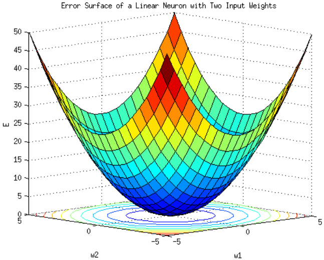




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$
- Let $\mathcal{S}$ denote a subspace spanned by columns of X in N-dim space
- $\mathbf{y} = \mathbf{X}\mathbf{w} \in \mathcal{S}$, being a linear combination of columns of $\mathbf{X}$
- For the LS optimality criterion
  - $\mathbf{y}$ is orthogonal projection of $\mathbf{t}$ on $\mathcal{S}$
  - Error surface $J(\mathbf{w})$ is convex
  - Sim. to Wiener filter: $\mathbf{w} = \mathbf{R}_{xx}^{-1} \mathrm{r}_{xt}$

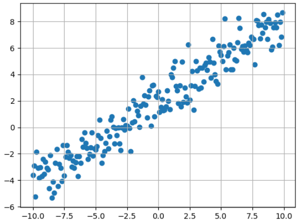
Geometric Interpretation of Least Squares

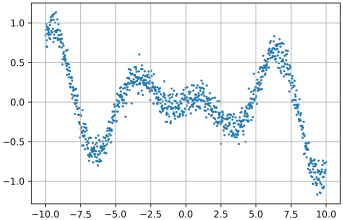




- Given N examples, the target vector $\mathbf{t} \in \mathbb{R}^N$ and columns of $\mathbf{X} \in \mathbb{R}^N$
- Let $\mathcal{S}$ denote a subspace spanned by columns of X in N-dim space
- $\mathbf{y} = \mathbf{X}\mathbf{w} \in \mathcal{S}$, being a linear combination of columns of $\mathbf{X}$
- For the LS optimality criterion
  - $\mathbf{y}$ is orthogonal projection of $\mathbf{t}$ on $\mathcal{S}$
  - Error surface $J(\mathbf{w})$ is convex
  - Sim. to Wiener filter: $\mathbf{w} = \mathbf{R}_{xx}^{-1} \mathrm{r}_{xt}$
  - Also referred to as pseudo inverse sol.

Nonlinear Input-Output Relations



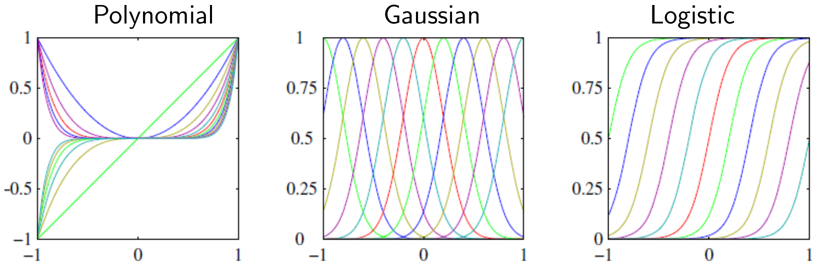


- Polynomial curve fitting can be used to model ninlinear i/o relation

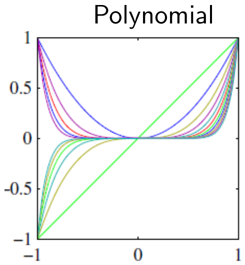
$$\begin{aligned} \hat{t} = y(x, \mathbf{w}) &= w_0 + w_1 x + w_2 x^2 + \cdots + w_M x^M \\ &= \mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}) \qquad \text{(Model is linear in } \mathbf{w}\text{)} \end{aligned}$$

- $\phi(.) : \mathbb{R}^1 \to \mathbb{R}^M$ - nonlinear transformation to higher dim. space

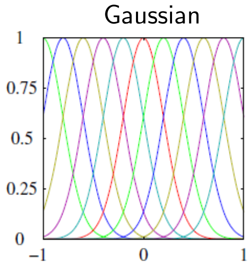
Kernel Examples



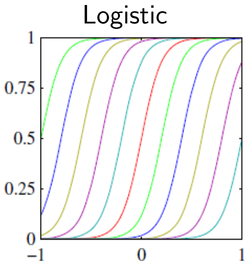
Kernel Examples



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

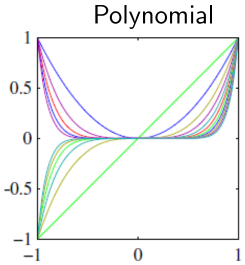


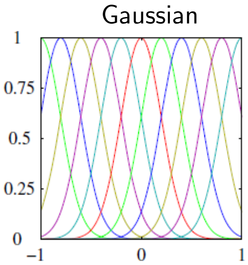
$$\phi_j(x) = \exp\left(-\frac{x-\mu_j}{2\sigma^2}\right)$$

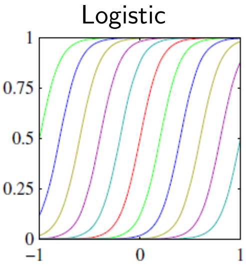


$$\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right)$$

Kernel Examples





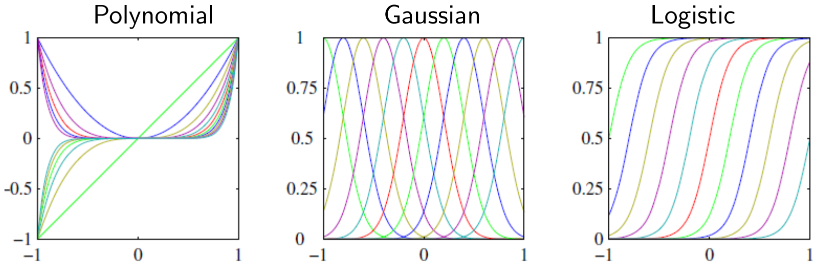


$\phi_j(x) = x^j$

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

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

Kernel Examples



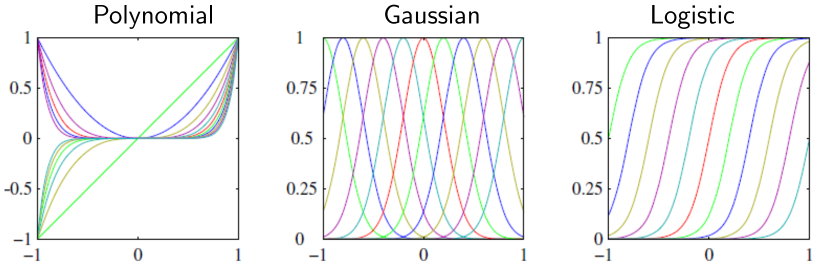
$\phi_j(x) = x^j$

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

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

- Explicit vs Implicit kernels

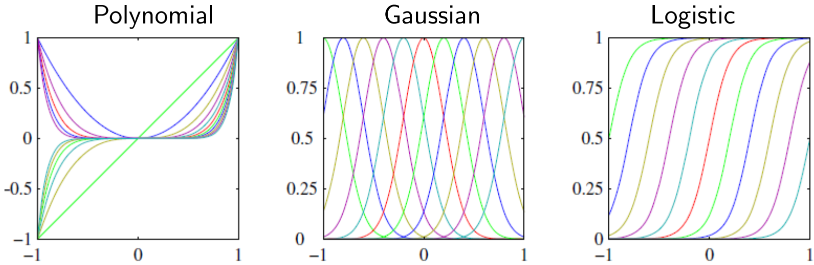
Kernel Examples



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

- Explicit vs Implicit kernels
  - Explicit representation for $\phi(\mathbf{x})$ is available or not

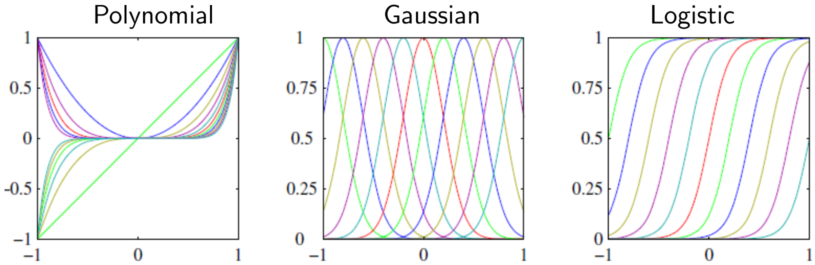
Kernel Examples



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

- Explicit vs Implicit kernels
    - Explicit representation for $\phi(\mathbf{x})$ is available or not
- Global vs Local kernels

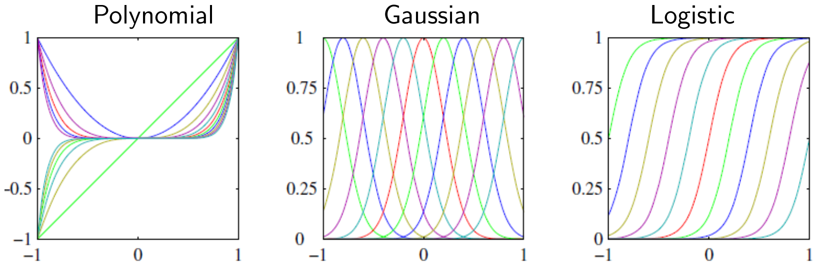
Kernel Examples



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

- Explicit vs Implicit kernels
  - Explicit representation for $\phi(\mathbf{x})$ is available or not
- Global vs Local kernels
  - Changes in one region of input space affect all other regions

Kernel Examples



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

- Explicit vs Implicit kernels
  - Explicit representation for $\phi(\mathbf{x})$ is available or not
- Global vs Local kernels
  - Changes in one region of input space affect all other regions
  - Local kernels are preferable for functions with varying characteristics

Least Squares Regression in Kernel Space

Least Squares Regression in Kernel Space

- If t_n is nonlinearly related to $\mathbf{x}_n$, perform regression in kernel space.

Least Squares Regression in Kernel Space

- If t_n is nonlinearly related to $\mathbf{x}_n$, perform regression in kernel space.
- Let $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M, \quad M > D$ is a nonlinear kernel mapping

Least Squares Regression in Kernel Space

- If t_n is nonlinearly related to $\mathbf{x}_n$, perform regression in kernel space.
- Let $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M, \quad M > D$ is a nonlinear kernel mapping
- $\mathbf{x}_n = [x_{n1} \ x_{n2}]^\mathsf{T} \in \mathbb{R}^2$ can be mapped using 2nd order polynomial kernel as $\phi(\mathbf{x}_n) = [1 \ x_{n1} \ x_{n2} \ x_{n1}^2 \ x_{n2}^2 \ x_{n1}x_{n2}]^\mathsf{T} \in \mathbb{R}^6$

Least Squares Regression in Kernel Space

- If t_n is nonlinearly related to $\mathbf{x}_n$, perform regression in kernel space.
- Let $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M, \quad M > D$ is a nonlinear kernel mapping
- $\mathbf{x}_n = [x_{n1} \ x_{n2}]^\mathsf{T} \in \mathbb{R}^2$ can be mapped using 2[nd] order polynomial kernel as $\phi(\mathbf{x}_n) = [1 \ x_{n1} \ x_{n2} \ x_{n1}^2 \ x_{n2}^2 \ x_{n1}x_{n2}]^\mathsf{T} \in \mathbb{R}^6$
- The target t_n is regressed from the kernel representation $\phi(\mathbf{x}_n)$ as

$$\hat{t}_n = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n) \qquad \mathbf{w} \in \mathbb{R}^M$$

Least Squares Regression in Kernel Space

- If t_n is nonlinearly related to $\mathbf{x}_n$, perform regression in kernel space.
- Let $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M, \quad M > D$ is a nonlinear kernel mapping
- $\mathbf{x}_n = [x_{n1} \; x_{n2}]^\mathsf{T} \in \mathbb{R}^2$ can be mapped using 2^{nd} order polynomial kernel as $\phi(\mathbf{x}_n) = [1 \; x_{n1} \; x_{n2} \; x_{n1}^2 \; x_{n2}^2 \; x_{n1}x_{n2}]^\mathsf{T} \in \mathbb{R}^6$
- The target t_n is regressed from the kernel representation $\phi(\mathbf{x}_n)$ as

$$\hat{t}_n = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n) \qquad \mathbf{w} \in \mathbb{R}^M$$

- The regression coefficients are given by $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$

Least Squares Regression in Kernel Space

- If t_n is nonlinearly related to $\mathbf{x}_n$, perform regression in kernel space.
- Let $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M, \quad M > D$ is a nonlinear kernel mapping
- $\mathbf{x}_n = [x_{n1}\ x_{n2}]^\mathsf{T} \in \mathbb{R}^2$ can be mapped using 2^{nd} order polynomial kernel as $\phi(\mathbf{x}_n) = [1\ x_{n1}\ x_{n2}\ x_{n1}^2\ x_{n2}^2\ x_{n1}x_{n2}]^\mathsf{T} \in \mathbb{R}^6$
- The target t_n is regressed from the kernel representation $\phi(\mathbf{x}_n)$ as

$$\hat{t}_n = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n) \qquad \mathbf{w} \in \mathbb{R}^M$$

- The regression coefficients are given by $\mathbf{w}_* = (\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\mathsf{T}\mathbf{t}$
- DNNs can be used to learn data-dependent nonlinear transf. $\boldsymbol{\phi(\mathbf{x}_n)}$

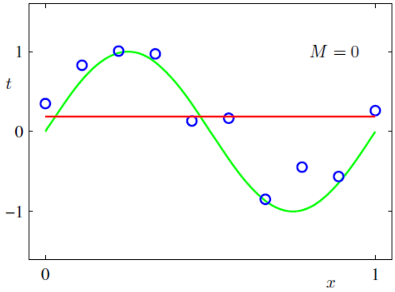
Least Squares Regression in Kernel Space

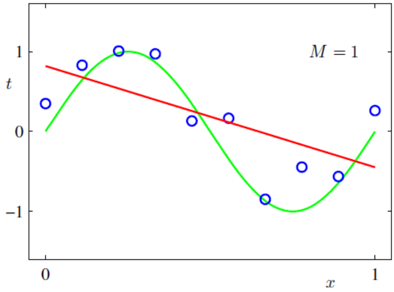
- If t_n is nonlinearly related to $\mathbf{x}_n$, perform regression in kernel space.
- Let $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^M, \quad M > D$ is a nonlinear kernel mapping
- $\mathbf{x}_n = [x_{n1} \ x_{n2}]^\mathsf{T} \in \mathbb{R}^2$ can be mapped using 2[nd] order polynomial kernel as $\phi(\mathbf{x}_n) = [1 \ x_{n1} \ x_{n2} \ x_{n1}^2 \ x_{n2}^2 \ x_{n1}x_{n2}]^\mathsf{T} \in \mathbb{R}^6$
- The target t_n is regressed from the kernel representation $\phi(\mathbf{x}_n)$ as

$$\hat{t}_n = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n) \qquad \mathbf{w} \in \mathbb{R}^M$$

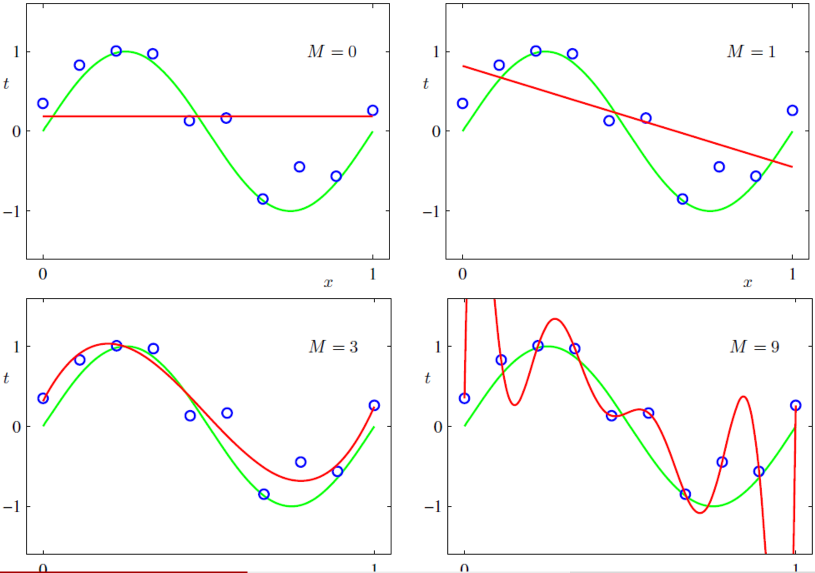
- The regression coefficients are given by $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
- DNNs can be used to learn data-dependent nonlinear transf. $\phi(\boldsymbol{x_n})$
- The last layer of DNNs typically performs linear regression on $\phi(\mathbf{x}_n)$

Effect of Model Order M: $t = \sin(\pi x) + \epsilon$

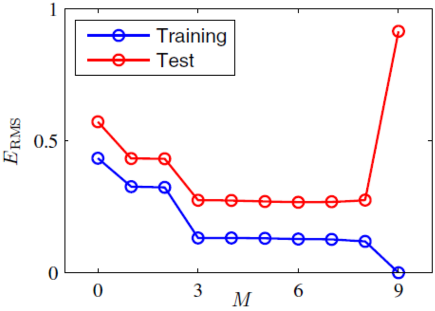




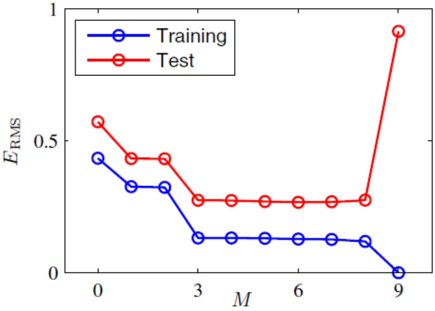
Effect of Model Order M: $t = \sin(\pi x) + \epsilon$



Model Validation

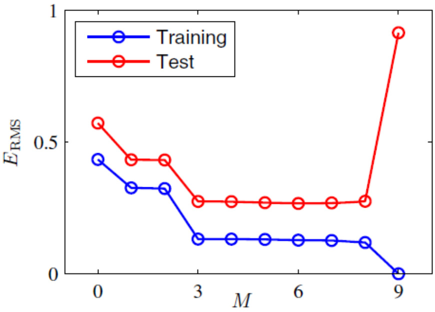


Model Validation



- Training & test error diverge for higher model orders
- Model 'overfits' to the noise in the training data

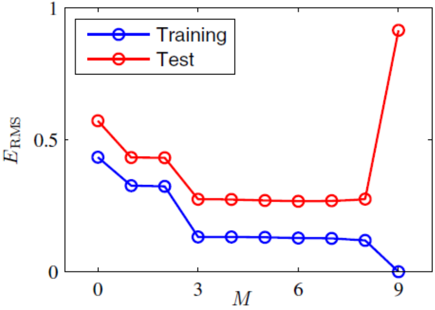
Model Validation



	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	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

- Training & test error diverge for higher model orders
- Model 'overfits' to the noise in the training data

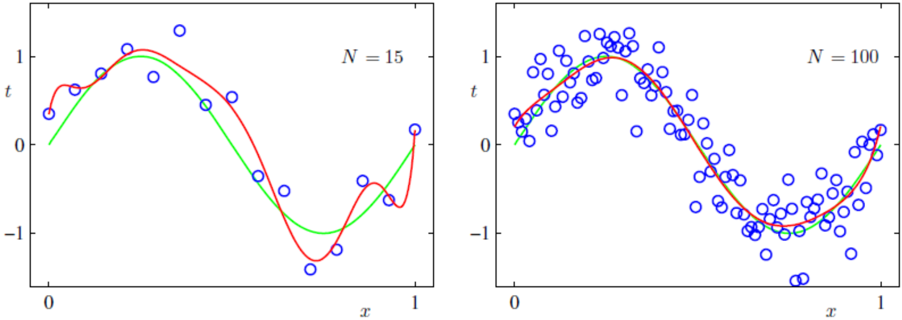
Model Validation



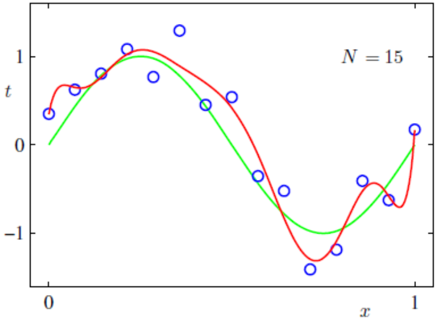
	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

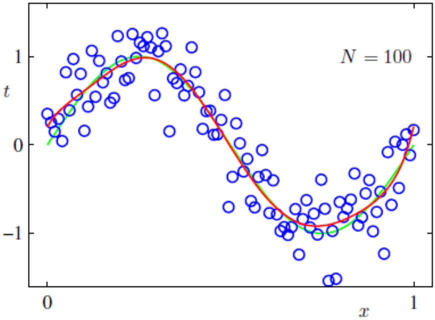
- Training & test error diverge for higher model orders
- Model 'overfits' to the noise in the training data
- Large amplitude weights with alternating polarity.
- $(\mathbf{\Phi}^T\mathbf{\Phi})$ may be ill conditioned

Amount of Training Data ($M = 9$)



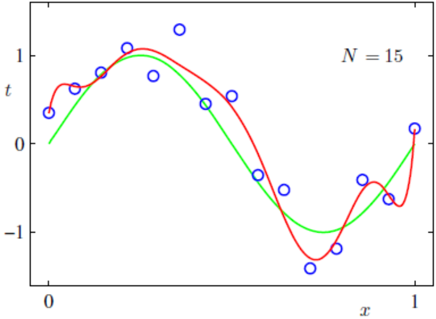
Amount of Training Data ($M = 9$)

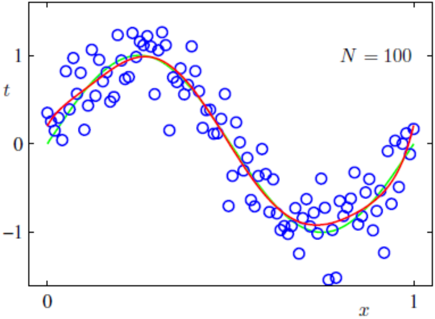




- Overfitting is less severe with increased amount of data.

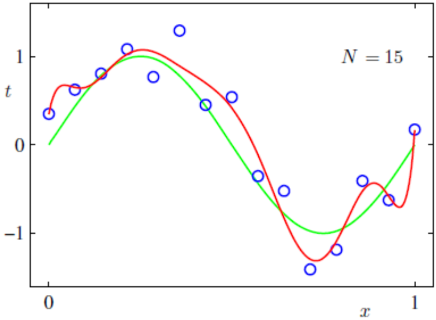
Amount of Training Data ($M = 9$)

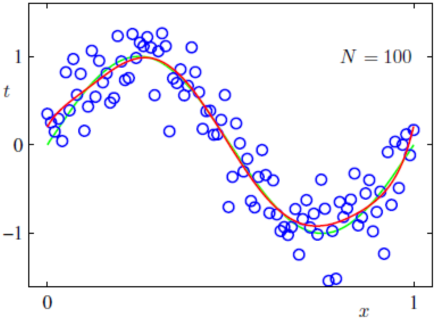




- Overfitting is less severe with increased amount of data.
- Model order cannot be limited by the amount of data available!

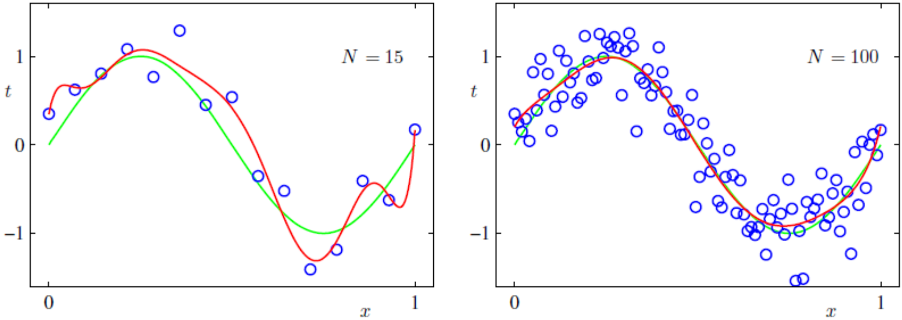
Amount of Training Data ($M = 9$)





- Overfitting is less severe with increased amount of data.
- Model order cannot be limited by the amount of data available!
- Model order should be based on complexity of task/pattern!

Amount of Training Data ($M = 9$)



- Overfitting is less severe with increased amount of data.
- Model order cannot be limited by the amount of data available!
- Model order should be based on complexity of task/pattern!
- A way forward: arrest the growth of the model weights

Regularized Least Squares

Regularized Least Squares

- Add a penalty term to the error term to discourage weight growth

$$J(\mathbf{w}) = \underbrace{E_D(\mathbf{w})}_{\text{Data Term}} + \underbrace{\lambda E_W(\mathbf{w})}_{\text{Regularization Term}}$$

Regularized Least Squares

- Add a penalty term to the error term to discourage weight growth

$$J(\mathbf{w}) = \underbrace{E_D(\mathbf{w})}_{\text{Data Term}} + \underbrace{\lambda E_W(\mathbf{w})}_{\text{Regularization Term}}$$

- λ controls relative importance of the terms (bias vs variance)

Regularized Least Squares

- Add a penalty term to the error term to discourage weight growth

$$J(\mathbf{w}) = \underbrace{E_D(\mathbf{w})}_{\text{Data Term}} + \underbrace{\lambda E_W(\mathbf{w})}_{\text{Regularization Term}}$$

- λ controls relative importance of the terms (bias vs variance)
- Sum of squares error function with a quadratic regularizer

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

Regularized Least Squares

- Add a penalty term to the error term to discourage weight growth

$$J(\mathbf{w}) = \underbrace{E_D(\mathbf{w})}_{\text{Data Term}} + \underbrace{\lambda E_W(\mathbf{w})}_{\text{Regularization Term}}$$

- λ controls relative importance of the terms (bias vs variance)
- Sum of squares error function with a quadratic regularizer

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

- Equating $\nabla_{\mathbf{w}} J(\mathbf{w}) = \mathbf{0} \implies -\mathbf{\Phi}^\mathsf{T}(\mathbf{t} - \mathbf{\Phi}\mathbf{w}) + \lambda\mathbf{w} = \mathbf{0}$

$$\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$$

Regularized Least Squares

- Add a penalty term to the error term to discourage weight growth

$$J(\mathbf{w}) = \underbrace{E_D(\mathbf{w})}_{\text{Data Term}} + \underbrace{\lambda E_W(\mathbf{w})}_{\text{Regularization Term}}$$

- λ controls relative importance of the terms (bias vs variance)
- Sum of squares error function with a quadratic regularizer

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

- Equating $\nabla_{\mathbf{w}} J(\mathbf{w}) = \mathbf{0} \implies -\mathbf{\Phi}^T\left(\mathbf{t} - \mathbf{\Phi}\mathbf{w}\right) + \lambda\mathbf{w} = \mathbf{0}$

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

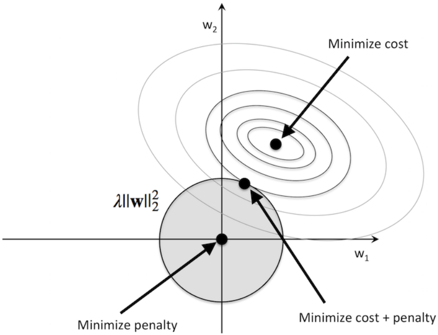
- Regularization term conditions the autocorrelation matrix!

Modified Error Surface

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\Big(t_n - \mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n)\Big)^2 + \lambda\|\mathbf{w}\|_p$$

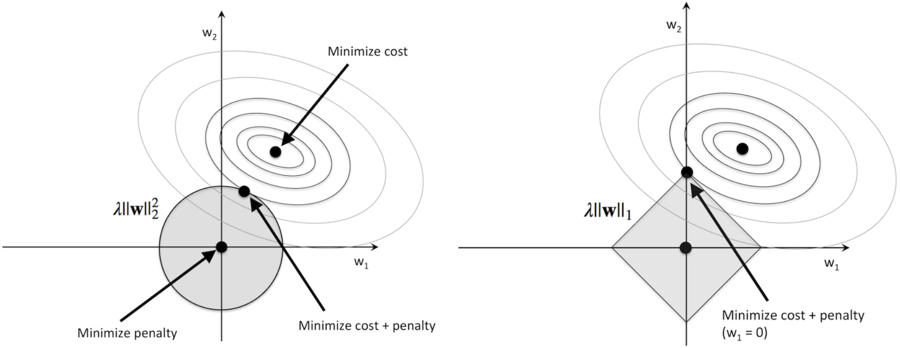
Modified Error Surface

$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\Big(t_n - \mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n)\Big)^2 + \lambda\|\mathbf{w}\|_p$$



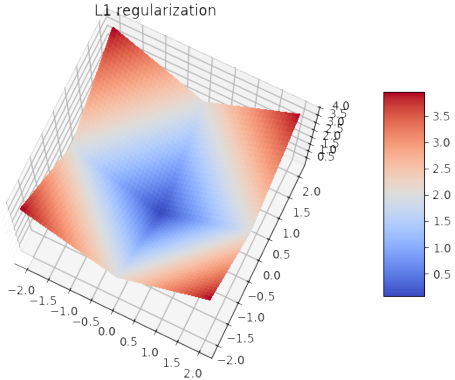
Modified Error Surface

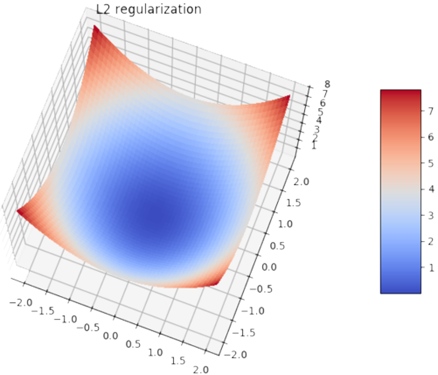
$$J(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(t_n - \mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n)\right)^2 + \lambda\|\mathbf{w}\|_p$$



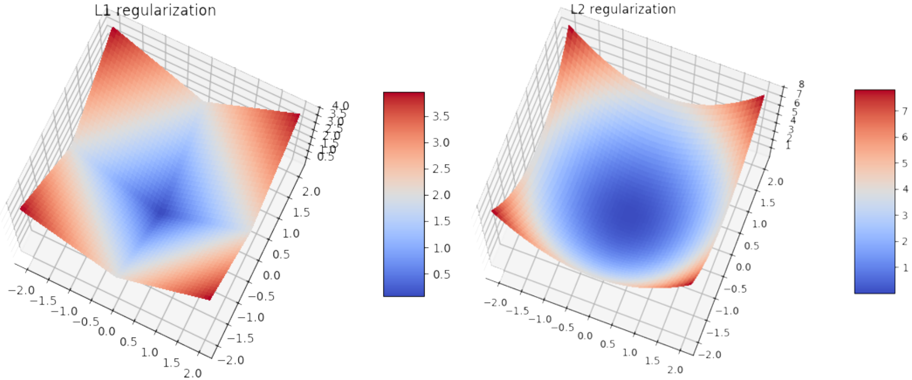
L_2 Regularizer　　L_1 Regularizer

L_1 vs L_2


L1 regularization
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0
0.5 1.0 1.5 2.0 2.5 3.0 3.5

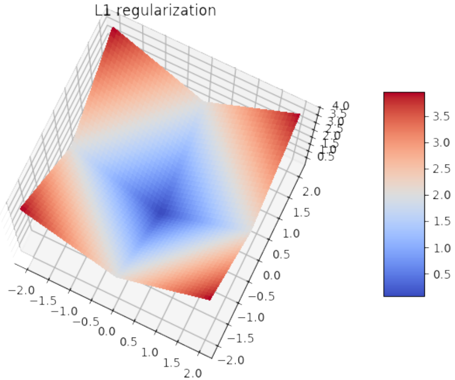

L2 regularization
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0
1 2 3 4 5 6 7 8

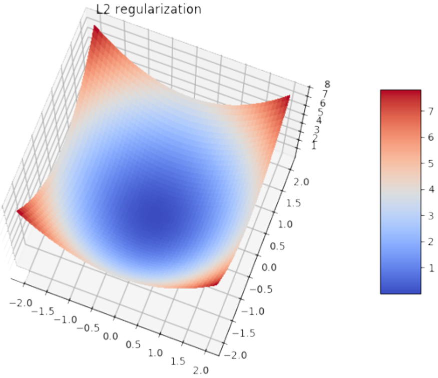
L_1 vs L_2



- L_1 regularization promotes sparser solutions

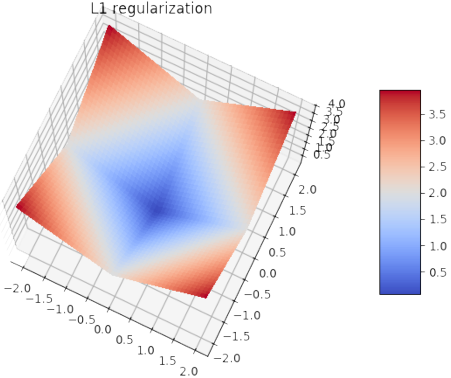
L_1 vs L_2

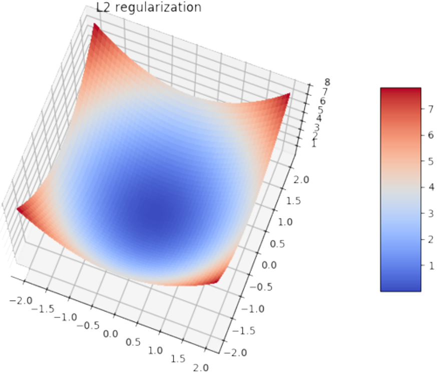




- L_1 regularization promotes sparser solutions
- L_1 regularization $\implies$ Laplacian priors

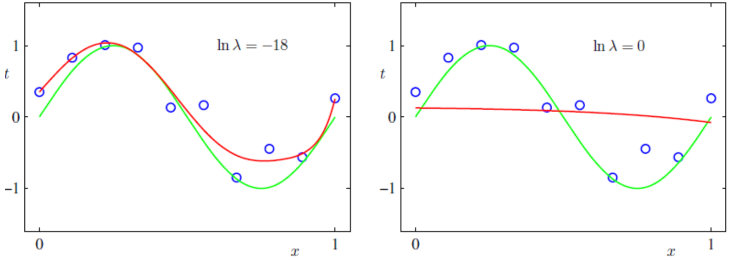
L_1 vs L_2



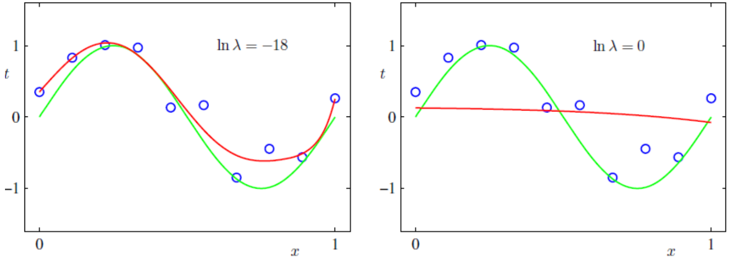


- L_1 regularization promotes sparser solutions
- L_1 regularization $\implies$ Laplacian priors
- L_2 regularization $\implies$ Gaussian priors

Effect of Regularization ($N = 10, M = 9$)

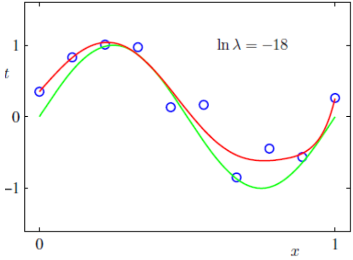


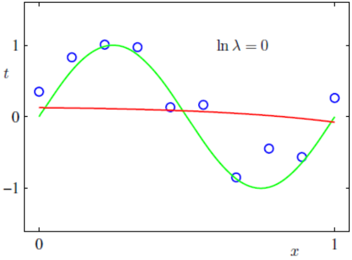
Effect of Regularization ($N = 10$, $M = 9$)



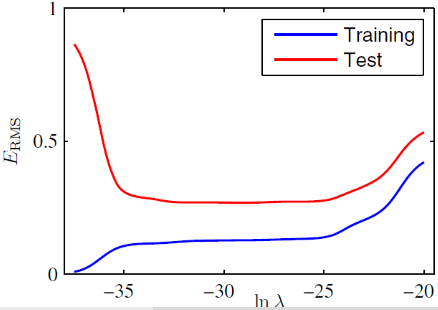
	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$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

Effect of Regularization ($N = 10, M = 9$)





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



Sequential Learning

Sequential Learning

- LS approach involves considering entire training set in one go.

Sequential Learning

- LS approach involves considering entire training set in one go.
- For HD data the matrix $(\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi})$ may be poorly conditioned

Sequential Learning

- LS approach involves considering entire training set in one go.
- For HD data the matrix $(\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi})$ may be poorly conditioned
- Iteratively update $\mathbf{w}^{(\tau+1)}$ by adding a correction factor

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

Sequential Learning

- LS approach involves considering entire training set in one go.
- For HD data the matrix $(\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi})$ may be poorly conditioned
- Iteratively update $\mathbf{w}^{(\tau+1)}$ by adding a correction factor

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

- Apply correction factor in the negative direction of gradient of $J(\mathbf{w}^{(\tau)})$

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta\nabla J(\mathbf{w}^{(\tau)})$$

Sequential Learning

- LS approach involves considering entire training set in one go.
- For HD data the matrix $(\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi})$ may be poorly conditioned
- Iteratively update $\mathbf{w}^{(\tau+1)}$ by adding a correction factor

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

- Apply correction factor in the negative direction of gradient of $J(\mathbf{w}^{(\tau)})$

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta\nabla J(\mathbf{w}^{(\tau)})$$

- Choose a random batch of points $\mathcal{B}$ to update $\mathbf{w}$. $J(\mathbf{w}^{(\tau)}) = \frac{1}{2}\sum_{n\in\mathcal{B}} e_n^2$

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta\sum_{n\in\mathcal{B}}\left(t_n - \mathbf{w}^{(\tau)\mathsf{T}}\phi(\mathbf{x}_n)\right)\phi(\mathbf{x}_n)$$

Sequential Learning

- LS approach involves considering entire training set in one go.
- For HD data the matrix $(\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})$ may be poorly conditioned
- Iteratively update $\mathbf{w}^{(\tau+1)}$ by adding a correction factor

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)}$$

- Apply correction factor in the negative direction of gradient of $J(\mathbf{w}^{(\tau)})$

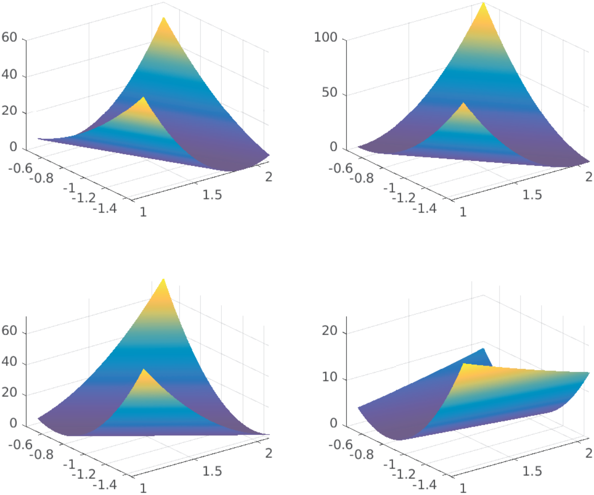
$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta\nabla J(\mathbf{w}^{(\tau)})$$

- Choose a random batch of points $\mathcal{B}$ to update $\mathbf{w}$. $J(\mathbf{w}^{(\tau)}) = \frac{1}{2}\sum_{n\in\mathcal{B}} e_n^2$

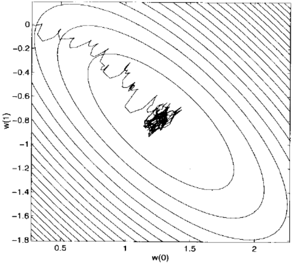
$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta\sum_{n\in\mathcal{B}}\left(t_n - \mathbf{w}^{(\tau)\mathsf{T}}\phi(\mathbf{x}_n)\right)\phi(\mathbf{x}_n)$$

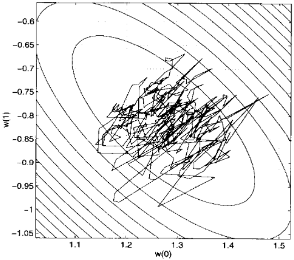
- $|\mathcal{B}| = N$: Steepest descent $\quad |\mathcal{B}| = 1$: LMS $\quad$ Otherwise: SGD.

SGD Error Dynamics: $w_1 = 1.6, w_2 = -0.5$

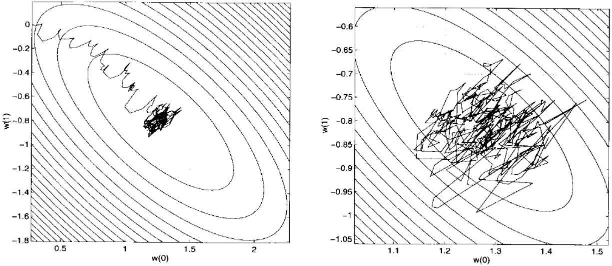


Convergence of SGD





Convergence of SGD

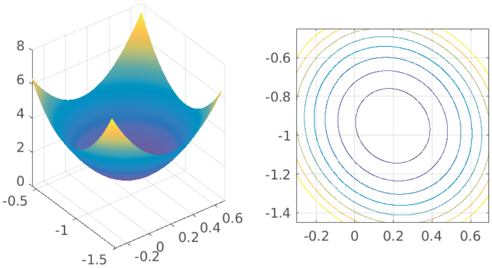


- SGD algorithm converges in mean:

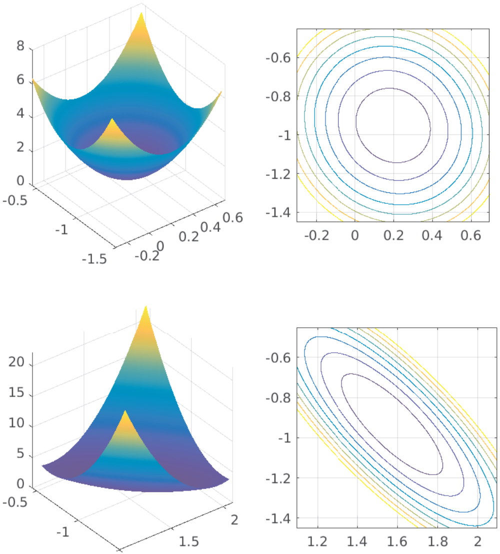
$$\lim_{k\to\infty} \mathbb{E}[\mathbf{w}_k] \to (\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\mathsf{T}\mathbf{t} \qquad \eta \text{ is small enough}$$

- Expectation over multiple runs (k) converges to true solution for convex error surfaces, provided η is sufficiently small

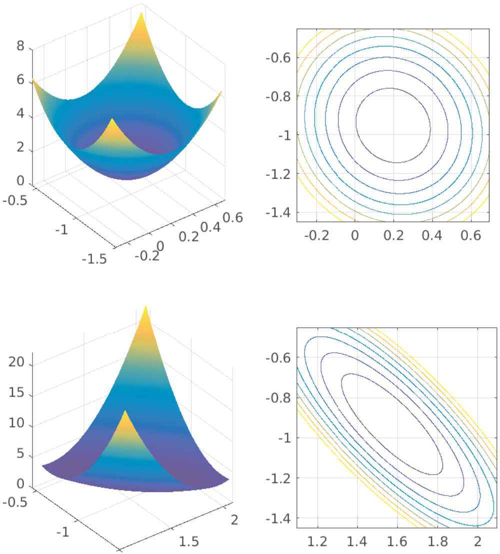
Geometry of Error Surface vs Convergence Rate



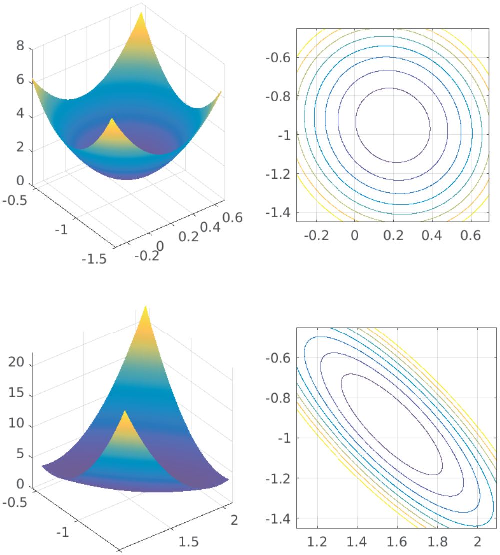
Geometry of Error Surface vs Convergence Rate



Geometry of Error Surface vs Convergence Rate

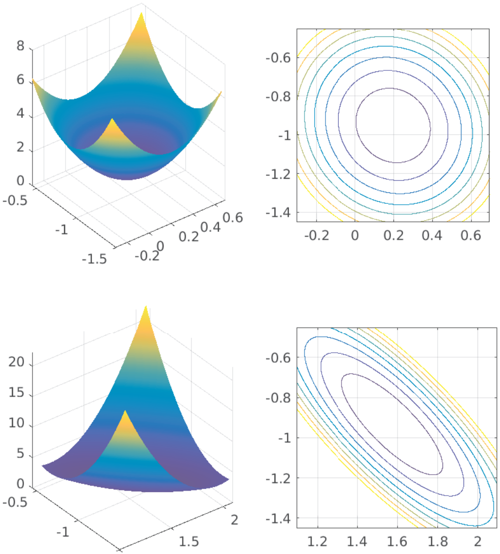


Geometry of Error Surface vs Convergence Rate



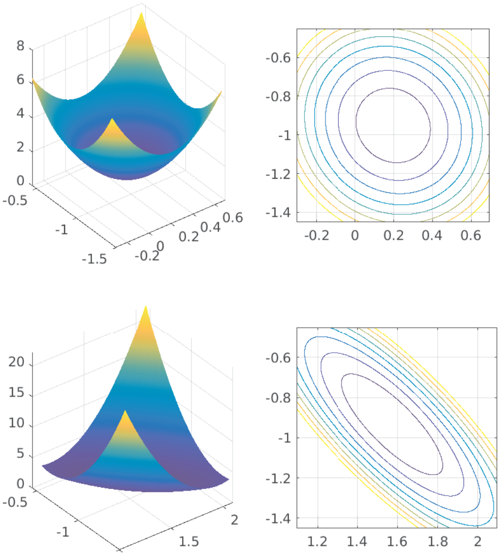
- Gradient magnitude depends on direction!

Geometry of Error Surface vs Convergence Rate



- Gradient magnitude depends on direction!
- η has to be fixed based on steepest direction.

Geometry of Error Surface vs Convergence Rate



- Gradient magnitude depends on direction!
- η has to be fixed based on steepest direction.
- Convergence along flatter dimension is too slow!

Newtons Method

Newtons Method

- The weights of the model are updated as

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \Delta \mathbf{w}$$

Newtons Method

- The weights of the model are updated as

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \Delta\mathbf{w}$$

- Expanding the objective function using Taylor series

$J(\mathbf{w}_{n+1}) = J(\mathbf{w}_n + \Delta\mathbf{w}) =$

Newtons Method

- The weights of the model are updated as

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \Delta\mathbf{w}$$

- Expanding the objective function using Taylor series

$$J(\mathbf{w}_{n+1}) = J(\mathbf{w}_n + \Delta\mathbf{w}) = J(\mathbf{w}_n) + \Delta\mathbf{w}^\mathsf{T}\nabla J(\mathbf{w}_n) + \frac{1}{2}\Delta\mathbf{w}^\mathsf{T}\nabla^2 J(\mathbf{w}_n)\Delta\mathbf{w}$$

Newtons Method

- The weights of the model are updated as

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \Delta\mathbf{w}$$

- Expanding the objective function using Taylor series

$$J(\mathbf{w}_{n+1}) = J(\mathbf{w}_n + \Delta\mathbf{w}) = J(\mathbf{w}_n) + \Delta\mathbf{w}^{\mathsf{T}} \nabla J(\mathbf{w}_n) + \frac{1}{2}\Delta\mathbf{w}^{\mathsf{T}} \nabla^2 J(\mathbf{w}_n)\Delta\mathbf{w}$$

- Estimate $\Delta\mathbf{w}$ s.t $J(\mathbf{w}_n + \Delta\mathbf{w})$ is minimized

$$\frac{\partial}{\partial \Delta\mathbf{w}}\left(J(\mathbf{w}_n) + \Delta\mathbf{w}^{\mathsf{T}} \nabla J(\mathbf{w}_n) + \frac{1}{2}\Delta\mathbf{w}^{\mathsf{T}} \nabla^2 J(\mathbf{w}_n)\Delta\mathbf{w}\right) = 0$$

Newtons Method

- The weights of the model are updated as

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \Delta\mathbf{w}$$

- Expanding the objective function using Taylor series

$$J(\mathbf{w}_{n+1}) = J(\mathbf{w}_n + \Delta\mathbf{w}) = J(\mathbf{w}_n) + \Delta\mathbf{w}^{\mathsf{T}}\nabla J(\mathbf{w}_n) + \frac{1}{2}\Delta\mathbf{w}^{\mathsf{T}}\nabla^2 J(\mathbf{w}_n)\Delta\mathbf{w}$$

- Estimate $\Delta\mathbf{w}$ s.t $J(\mathbf{w}_n + \Delta\mathbf{w})$ is minimized

$$\frac{\partial}{\partial\Delta\mathbf{w}}\left(J(\mathbf{w}_n) + \Delta\mathbf{w}^{\mathsf{T}}\nabla J(\mathbf{w}_n) + \frac{1}{2}\Delta\mathbf{w}^{\mathsf{T}}\nabla^2 J(\mathbf{w}_n)\Delta\mathbf{w}\right) = 0$$

- Optimal update is given by $\Delta\mathbf{w} = -\frac{\nabla J(\mathbf{w}_n)}{\nabla^2 J(\mathbf{w}_n)}$

$$\mathbf{w}_{n+1} = \mathbf{w}_n - \mathbf{H}^{-1}(\mathbf{w}_n)\nabla J(\mathbf{w}_n) \qquad \mathbf{H}(\mathbf{w}_n) = \nabla^2 J(\mathbf{w}_n)$$

Homework - 1

- Apply Newtons method to steepest-descent algorithm to the optimal step size η, and check how many iterations are required for convergence.

$$\mathbf{w}^{new} = \mathbf{w}^{old} + \eta \, \mathbf{X}^{\mathsf{T}}(\mathbf{t} - \mathbf{X}\mathbf{w})\Big|_{\mathbf{w}=\mathbf{w}^{old}}$$

Homework - 2

- Suppose you are experimenting with L_1 and L_2 regularization. Further, imagine that you are running gradient descent and at some iteration your weight vector is $w = [1, \epsilon] \in \mathbb{R}^2$ where $\epsilon > 0$ is very small. With the help of this example explain why L_2 norm does not encourage sparsity i.e., it will not try to drive ϵ to 0 to produce a sparse weight vector. Give mathematical explanation.

Homework - 3

- Till now we have been considering a scalar target t from a vector of input observations $\mathbf{x}$. How do you extend this approach for regressing a vector of targets $\mathbf{t} = (t_1, t_2, \cdots t_P)$. Derive the closed form solutions and write sequential update equations using SGD.

Probabilistic Approach to Regression

Probabilistic Approach to Regression

- Predict target variable(s) $t \in \mathbb{R}$ given the observation vector $\mathbf{x} \in \mathbb{R}^D$

Probabilistic Approach to Regression

- Predict target variable(s) $t \in \mathbb{R}$ given the observation vector $\mathbf{x} \in \mathbb{R}^D$
- Target variable is estimated as a deterministic function $y(\mathbf{x}, \mathbf{w})$ with some error e.

$$t = y(\mathbf{x}, \mathbf{w}) + e_n$$

Probabilistic Approach to Regression

- Predict target variable(s) $t \in \mathbb{R}$ given the observation vector $\mathbf{x} \in \mathbb{R}^D$
- Target variable is estimated as a deterministic function $y(\mathbf{x}, \mathbf{w})$ with some error e.

$$t = y(\mathbf{x}, \mathbf{w}) + e_n$$

- Assume that the error is Gaussian distributed: $e \sim \mathcal{N}(0, \beta^{-1})$

Probabilistic Approach to Regression

- Predict target variable(s) $t \in \mathbb{R}$ given the observation vector $\mathbf{x} \in \mathbb{R}^D$
- Target variable is estimated as a deterministic function $y(\mathbf{x}, \mathbf{w})$ with some error e.

$$t = y(\mathbf{x}, \mathbf{w}) + e_n$$

- Assume that the error is Gaussian distributed: $e \sim \mathcal{N}(0, \beta^{-1})$
- Hence, the conditional distribution of target t is given by

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

Probabilistic Approach to Regression

- Predict target variable(s) $t \in \mathbb{R}$ given the observation vector $\mathbf{x} \in \mathbb{R}^D$
- Target variable is estimated as a deterministic function $y(\mathbf{x}, \mathbf{w})$ with some error e.

$$t = y(\mathbf{x}, \mathbf{w}) + e_n$$

- Assume that the error is Gaussian distributed: $e \sim \mathcal{N}(0, \beta^{-1})$
- Hence, the conditional distribution of target t is given by

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

- Gaussian noise $\implies$ Guassian conditional density on targets

Probabilistic Approach to Regression

- Predict target variable(s) $t \in \mathbb{R}$ given the observation vector $\mathbf{x} \in \mathbb{R}^D$
- Target variable is estimated as a deterministic function $y(\mathbf{x}, \mathbf{w})$ with some error e.

$$t = y(\mathbf{x}, \mathbf{w}) + e_n$$

- Assume that the error is Gaussian distributed: $e \sim \mathcal{N}(0, \beta^{-1})$
- Hence, the conditional distribution of target t is given by

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

- Gaussian noise $\implies$ Guassian conditional density on targets
- We need to estimate $\mathbf{w}$ (and β) to maximize $p(t/\mathbf{x}, \mathbf{w}, \beta)$

Maximum Likelihood (ML)

Maximum Likelihood (ML)

- Training data: $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2) \cdots (\mathbf{x}_n, t_n) \cdots (\mathbf{x}_N, t_N)\}$

Maximum Likelihood (ML)

- Training data: $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2) \cdots (\mathbf{x}_n, t_n) \cdots (\mathbf{x}_N, t_N)\}$
- Let the target be estimated as $\hat{t}_n = \mathbf{y}(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n)$

Maximum Likelihood (ML)

- Training data: $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2) \cdots (\mathbf{x}_n, t_n) \cdots (\mathbf{x}_N, t_N)\}$
- Let the target be estimated as $\hat{t}_n = \mathbf{y}(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n)$
- Assuming Gaussian errors: $p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1})$

Maximum Likelihood (ML)

- Training data: $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2) \cdots (\mathbf{x}_n, t_n) \cdots (\mathbf{x}_N, t_N)\}$
- Let the target be estimated as $\hat{t}_n = \mathbf{y}(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n)$
- Assuming Gaussian errors: $p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1})$
- Assuming the data points are drawn independently and identically

$$p(t_1, t_2, \cdots t_N/\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n/\mathbf{x}_n, \mathbf{w}, \beta)$$

Maximum Likelihood (ML)

- Training data: $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2) \cdots (\mathbf{x}_n, t_n) \cdots (\mathbf{x}_N, t_N)\}$
- Let the target be estimated as $\hat{t}_n = \mathbf{y}(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n)$
- Assuming Gaussian errors: $p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1})$
- Assuming the data points are drawn independently and identically

$$p(t_1, t_2, \cdots t_N/\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n/\mathbf{x}_n, \mathbf{w}, \beta)$$

$$\log p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta) = \sum_{n=1}^{N} \log \mathcal{N}(t/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1})$$

Maximum Likelihood (ML)

- Training data: $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2) \cdots (\mathbf{x}_n, t_n) \cdots (\mathbf{x}_N, t_N)\}$
- Let the target be estimated as $\hat{t}_n = \mathbf{y}(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^\mathsf{T} \phi(\mathbf{x}_n)$
- Assuming Gaussian errors: $p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1})$
- Assuming the data points are drawn independently and identically

$$p(t_1, t_2, \cdots t_N/\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n/\mathbf{x}_n, \mathbf{w}, \beta)$$

$$\begin{aligned}\log p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta) &= \sum_{n=1}^{N} \log \mathcal{N}(t/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1}) \\ &= \frac{N}{2}\log\beta - \frac{N}{2}\log(2\pi) - \frac{\beta}{2}\sum_{n=1}^{N}(t_n - \mathbf{w}^\mathsf{T}\phi(x_n))^2\end{aligned}$$

Maximum Likelihood (ML)

- Training data: $\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2) \cdots (\mathbf{x}_n, t_n) \cdots (\mathbf{x}_N, t_N)\}$
- Let the target be estimated as $\hat{t}_n = \mathbf{y}(\mathbf{x}_n, \mathbf{w}) = \mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}_n)$
- Assuming Gaussian errors: $p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t/\mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}_n), \beta^{-1})$
- Assuming the data points are drawn independently and identically

$$p(t_1, t_2, \cdots t_N/\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n/\mathbf{x}_n, \mathbf{w}, \beta)$$

$$\log p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta) = \sum_{n=1}^{N} \log \mathcal{N}(t/\mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}_n), \beta^{-1})$$

$$= \frac{N}{2} \log \beta - \frac{N}{2} \log(2\pi) - \frac{\beta}{2} \sum_{n=1}^{N} (t_n - \mathbf{w}^{\mathsf{T}} \phi(x_n))^2$$

- $\mathbf{w}$ and β can be estimated to maximize likelihood $p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta)$

Understanding Likelihood

Understanding Likelihood

- Likelihood function is not probability for continuous RV.

Understanding Likelihood

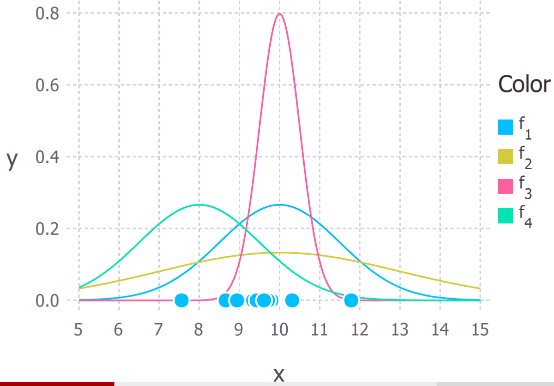
- Likelihood function is not probability for continuous RV.
- Likelihood can be greater than one.

Understanding Likelihood

- Likelihood function is not probability for continuous RV.
- Likelihood can be greater than one.
- In ML, the parameters $\mathbf{w}$ are adjusted to maximize the likelihood of the observed data $\mathbf{t}$. $\mathcal{L}(\mathbf{w}/\mathbf{t}, \mathbf{X}) = p(\mathbf{t}/\mathbf{X}, \mathbf{w})$

Understanding Likelihood

- Likelihood function is not probability for continuous RV.
- Likelihood can be greater than one.
- In ML, the parameters $\mathbf{w}$ are adjusted to maximize the likelihood of the observed data $\mathbf{t}$. $\mathcal{L}(\mathbf{w}/\mathbf{t}, \mathbf{X}) = p(\mathbf{t}/\mathbf{X}, \mathbf{w})$



ML $\iff$ Least Squares

ML $\iff$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

ML $\iff$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

- ML approach assigns a probability density to the estimated target

$$p(t/\mathbf{x}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(t/y(\mathbf{x}, \mathbf{w}_{ML}), \beta_{ML}^{-1})$$

ML $\iff$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

- ML approach assigns a probability density to the estimated target

$$p(t/\mathbf{x}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(t/y(\mathbf{x}, \mathbf{w}_{ML}), \beta_{ML}^{-1})$$

$$\mathbb{E}[t/\mathbf{x}] = \int tp(t/\mathbf{x})dt =$$

ML $\iff$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

- ML approach assigns a probability density to the estimated target

$$p(t/\mathbf{x}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(t/y(\mathbf{x}, \mathbf{w}_{ML}), \beta_{ML}^{-1})$$

$$\mathbb{E}[t/\mathbf{x}] = \int tp(t/\mathbf{x})dt = y(\mathbf{x}, \mathbf{w}_{ML})$$

ML $\iff$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

- ML approach assigns a probability density to the estimated target

$$p(t/\mathbf{x}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(t/y(\mathbf{x}, \mathbf{w}_{ML}), \beta_{ML}^{-1})$$

$$\mathbb{E}[t/\mathbf{x}] = \int t p(t/\mathbf{x}) dt = y(\mathbf{x}, \mathbf{w}_{ML})$$

- ML with Laplacian conditional density assumption is same as LAD

ML $\Longleftrightarrow$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

- ML approach assigns a probability density to the estimated target

$$p(t/\mathbf{x}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(t/y(\mathbf{x}, \mathbf{w}_{ML}), \beta_{ML}^{-1})$$

$$\mathbb{E}[t/\mathbf{x}] = \int tp(t/\mathbf{x})dt = y(\mathbf{x}, \mathbf{w}_{ML})$$

- ML with Laplacian conditional density assumption is same as LAD
- ML & LS rely on point estimates of model parameters $\mathbf{w}$

ML $\Longleftrightarrow$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

- ML approach assigns a probability density to the estimated target

$$p(t/\mathbf{x}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(t/y(\mathbf{x}, \mathbf{w}_{ML}), \beta_{ML}^{-1})$$

$$\mathbb{E}[t/\mathbf{x}] = \int tp(t/\mathbf{x})dt = y(\mathbf{x}, \mathbf{w}_{ML})$$

- ML with Laplacian conditional density assumption is same as LAD
- ML & LS rely on point estimates of model parameters **w**
- Point estimates cannot be exact with finite number of samples

ML $\iff$ Least Squares

- ML with Gaussian conditional density assumption is same as LS

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

- ML approach assigns a probability density to the estimated target

$$p(t/\mathbf{x}, \mathbf{w}_{ML}, \beta_{ML}) = \mathcal{N}(t/y(\mathbf{x}, \mathbf{w}_{ML}), \beta_{ML}^{-1})$$

$$\mathbb{E}[t/\mathbf{x}] = \int tp(t/\mathbf{x})dt = y(\mathbf{x}, \mathbf{w}_{ML})$$

- ML with Laplacian conditional density assumption is same as LAD
- ML & LS rely on point estimates of model parameters $\mathbf{w}$
- Point estimates cannot be exact with finite number of samples
- Instead, estimate the distribution of $\mathbf{w}$

Maximum A Posteriori (MAP) Estimate

Maximum A Posteriori (MAP) Estimate

- Given a set of N datapoints, the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}) \propto p(\mathbf{w})p(\mathbf{t}/\mathbf{w}, \mathbf{X})$$

Maximum A Posteriori (MAP) Estimate

- Given a set of N datapoints, the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}) \propto p(\mathbf{w})p(\mathbf{t}/\mathbf{w}, \mathbf{X})$$

- Let the prior distribution of $\mathbf{w}$ be Gaussian: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I})$

Maximum A Posteriori (MAP) Estimate

- Given a set of N datapoints, the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}) \propto p(\mathbf{w})p(\mathbf{t}/\mathbf{w}, \mathbf{X})$$

- Let the prior distribution of $\mathbf{w}$ be Gaussian: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I})$
- Let the conditional distribution of target be Gaussian

$$p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t_n/\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n), \beta^{-1})$$

Maximum A Posteriori (MAP) Estimate

- Given a set of N datapoints, the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}) \propto p(\mathbf{w})p(\mathbf{t}/\mathbf{w}, \mathbf{X})$$

- Let the prior distribution of $\mathbf{w}$ be Gaussian: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I})$
- Let the conditional distribution of target be Gaussian

$$p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t_n/\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n), \beta^{-1})$$

- The posterior distribution of $\mathbf{w}$ is given by

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}, \alpha, \beta) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I}) \prod_{n=1}^{N} \mathcal{N}(t_n/\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n), \beta^{-1})$$

Maximum A Posteriori (MAP) Estimate

- Given a set of N datapoints, the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}) \propto p(\mathbf{w})p(\mathbf{t}/\mathbf{w}, \mathbf{X})$$

- Let the prior distribution of $\mathbf{w}$ be Gaussian: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I})$
- Let the conditional distribution of target be Gaussian

$$p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t_n/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1})$$

- The posterior distribution of $\mathbf{w}$ is given by

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}, \alpha, \beta) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I}) \prod_{n=1}^{N} \mathcal{N}(t_n/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1})$$

$$\log p(\mathbf{w}/\mathbf{t}) = -\frac{\beta}{2}\sum_{n=1}^{N}\left(t_n - \mathbf{w}^\mathsf{T}\phi(x_n)\right)^2 - \frac{\alpha}{2}\mathbf{w}^\mathsf{T}\mathbf{w} + \text{const}$$

Maximum A Posteriori (MAP) Estimate

- Given a set of N datapoints, the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}) \propto p(\mathbf{w})p(\mathbf{t}/\mathbf{w}, \mathbf{X})$$

- Let the prior distribution of $\mathbf{w}$ be Gaussian: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I})$
- Let the conditional distribution of target be Gaussian

$$p(t_n/\mathbf{x}_n, \mathbf{w}, \beta) = \mathcal{N}(t_n/\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n), \beta^{-1})$$

- The posterior distribution of $\mathbf{w}$ is given by

$$p(\mathbf{w}/\mathbf{t}, \mathbf{X}, \alpha, \beta) = \mathcal{N}(\mathbf{w}/\mathbf{0}, \alpha^{-1}\mathbf{I}) \prod_{n=1}^{N} \mathcal{N}(t_n/\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n), \beta^{-1})$$

$$\log p(\mathbf{w}/\mathbf{t}) = -\frac{\beta}{2}\sum_{n=1}^{N}\left(t_n - \mathbf{w}^{\mathsf{T}}\phi(x_n)\right)^2 - \frac{\alpha}{2}\mathbf{w}^{\mathsf{T}}\mathbf{w} + \text{const}$$

- Estimate w to maximize $\log p(\mathbf{w}/\mathbf{t})$

MAP $\iff$ Regularized Least Squares

MAP $\iff$ Regularized Least Squares

- MAP estimation is equivalent to RLS with $\lambda = \frac{\alpha}{\beta}$

MAP $\Longleftrightarrow$ Regularized Least Squares

- MAP estimation is equivalent to RLS with $\lambda = \frac{\alpha}{\beta}$
- MAP estimate of $\mathbf{w}$ is given by

$$\mathbf{w}_{MAP} = (\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^{\mathsf{T}}\mathbf{t}$$

MAP $\iff$ Regularized Least Squares

- MAP estimation is equivalent to RLS with $\lambda = \frac{\alpha}{\beta}$
- MAP estimate of **w** is given by

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

- Gaussain priors $\iff$ L_2 regularizer

MAP $\iff$ Regularized Least Squares

- MAP estimation is equivalent to RLS with $\lambda = \frac{\alpha}{\beta}$
- MAP estimate of $\mathbf{w}$ is given by

$$\mathbf{w}_{MAP} = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$$

- Gaussain priors $\iff$ L_2 regularizer
- Laplacian priors $\iff$ L_1 regularizer

Evaluating Posterior Density $p(\mathbf{w}/\mathbf{t})$

Evaluating Posterior Density $p(\mathbf{w}/\mathbf{t})$

- Let the prior distribution of $\mathbf{w}$ be $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \mathbf{\Sigma}_0)$

Evaluating Posterior Density $p(\mathbf{w}/\mathbf{t})$

- Let the prior distribution of $\mathbf{w}$ be $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \boldsymbol{\Sigma}_0)$
- Assuming linear model with Gaussian errors, the likelihood is given by

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

Evaluating Posterior Density $p(\mathbf{w}/\mathbf{t})$

- Let the prior distribution of $\mathbf{w}$ be $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \mathbf{\Sigma}_0)$
- Assuming linear model with Gaussian errors, the likelihood is given by

$$p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1}) = \mathcal{N}(\mathbf{t}/\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I})$$

Evaluating Posterior Density $p(\mathbf{w}/\mathbf{t})$

- Let the prior distribution of $\mathbf{w}$ be $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \boldsymbol{\Sigma}_0)$
- Assuming linear model with Gaussian errors, the likelihood is given by

$$p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1}) = \mathcal{N}(\mathbf{t}/\boldsymbol{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I})$$

- The posterior density after observing 'N' samples is given by

$$p(\mathbf{w}/\mathbf{t}) \propto \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \boldsymbol{\Sigma}_0)\ \mathcal{N}(\mathbf{t}/\boldsymbol{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I})$$

Evaluating Posterior Density $p(\mathbf{w}/\mathbf{t})$

- Let the prior distribution of **w** be $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \mathbf{\Sigma}_0)$
- Assuming linear model with Gaussian errors, the likelihood is given by

$$p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n/\mathbf{w}^{\mathsf{T}}\phi(\mathbf{x}_n), \beta^{-1}) = \mathcal{N}(\mathbf{t}/\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I})$$

- The posterior density after observing 'N' samples is given by

$$p(\mathbf{w}/\mathbf{t}) \propto \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \mathbf{\Sigma}_0)\, \mathcal{N}(\mathbf{t}/\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_N, \mathbf{\Sigma}_N)$$

Evaluating Posterior Density $p(\mathbf{w}/\mathbf{t})$

- Let the prior distribution of $\mathbf{w}$ be $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \mathbf{\Sigma}_0)$
- Assuming linear model with Gaussian errors, the likelihood is given by

$$p(\mathbf{t}/\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n/\mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n), \beta^{-1}) = \mathcal{N}(\mathbf{t}/\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I})$$

- The posterior density after observing 'N' samples is given by

$$p(\mathbf{w}/\mathbf{t}) \propto \mathcal{N}(\mathbf{w}/\mathbf{m}_0, \mathbf{\Sigma}_0)\ \mathcal{N}(\mathbf{t}/\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_N, \mathbf{\Sigma}_N)$$

- $\mathbf{m}_N$ and $\mathbf{\Sigma}_N$ can be evaluated by completing quadratic term of exp()

$$\mathbf{m}_N = \mathbf{\Sigma}_N \left(\mathbf{\Sigma}_0^{-1}\mathbf{m}_0 + \beta\mathbf{\Phi}^\mathsf{T}\mathbf{t} \right)$$

$$\mathbf{\Sigma}_N^{-1} = \mathbf{\Sigma}_0^{-1} + \beta\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}$$

Bayesian Sequential Estimates

Bayesian Sequential Estimates

- Let the posterior distribution of **w** after observing n samples be

$$p(\mathbf{w}/\mathbf{t}_{1:n}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_n, \mathbf{\Sigma}_n)$$

Bayesian Sequential Estimates

- Let the posterior distribution of $\mathbf{w}$ after observing n samples be

$$p(\mathbf{w}/\mathbf{t}_{1:n}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_n, \mathbf{\Sigma}_n)$$

- In sequential update, $p(\mathbf{w}/\mathbf{t}_{1:n})$ is used as prior for $(n+1)^{th}$ sample

Bayesian Sequential Estimates

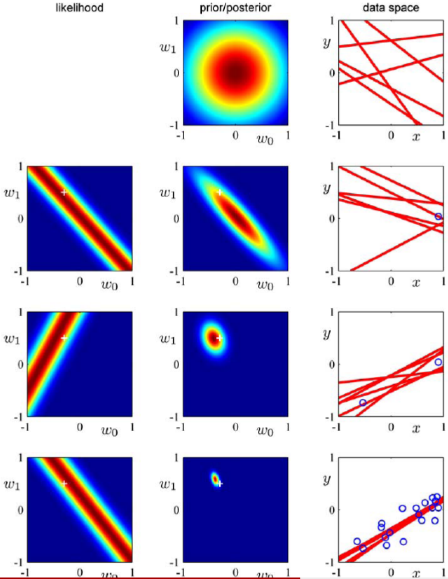
- Let the posterior distribution of $\mathbf{w}$ after observing n samples be

$$p(\mathbf{w}/\mathbf{t}_{1:n}) = \mathcal{N}(\mathbf{w}/\mathbf{m}_n, \boldsymbol{\Sigma}_n)$$

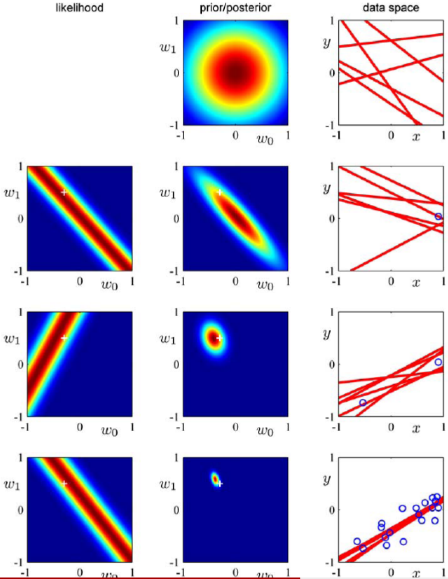
- In sequential update, $p(\mathbf{w}/\mathbf{t}_{1:n})$ is used as prior for $(n+1)^{th}$ sample
- The posterior stats can be updated after observing $(\mathbf{x}_{n+1}, t_{n+1})$ as

$$\mathbf{m}_{n+1} = \boldsymbol{\Sigma}_{n+1}\left(\boldsymbol{\Sigma}_n^{-1}\mathbf{m}_n + \beta\phi(\mathbf{x}_{n+1})t_{n+1}\right)$$
$$\boldsymbol{\Sigma}_{n+1}^{-1} = \boldsymbol{\Sigma}_n^{-1} + \beta\phi(\mathbf{x}_{n+1})\phi^{\mathsf{T}}(\mathbf{x}_{n+1})$$

Bayes Updates Illustration: $t = a_0 + a_1 x + \epsilon$



Bayes Updates Illustration: $t = a_0 + a_1 x + \epsilon$

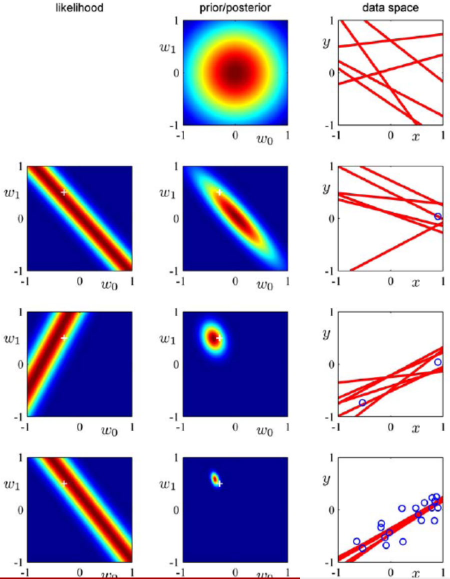


- Actual targets are generated as

$$t = 0.5x - 0.3 + \epsilon$$

$$x \in \mathcal{U}[-1\ 1] \qquad \epsilon \in \mathcal{N}(0, 0.2^2)$$

Bayes Updates Illustration: $t = a_0 + a_1 x + \epsilon$



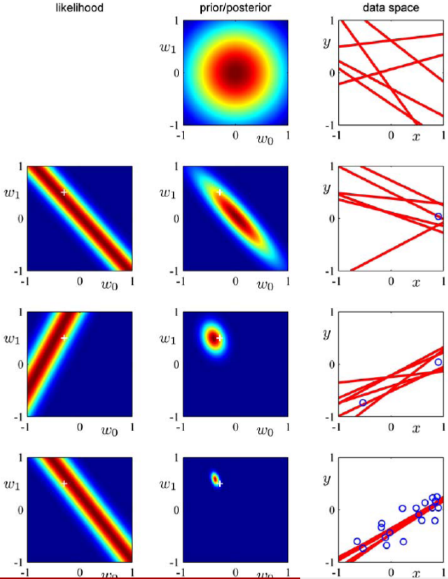
- Actual targets are generated as

$$t = 0.5x - 0.3 + \epsilon$$

$$x \in \mathcal{U}[-1\ 1] \qquad \epsilon \in \mathcal{N}(0, 0.2^2)$$

- Assume: $y(x, \mathbf{w}) = w_1 x + w_0$

Bayes Updates Illustration: $t = a_0 + a_1 x + \epsilon$



- Actual targets are generated as

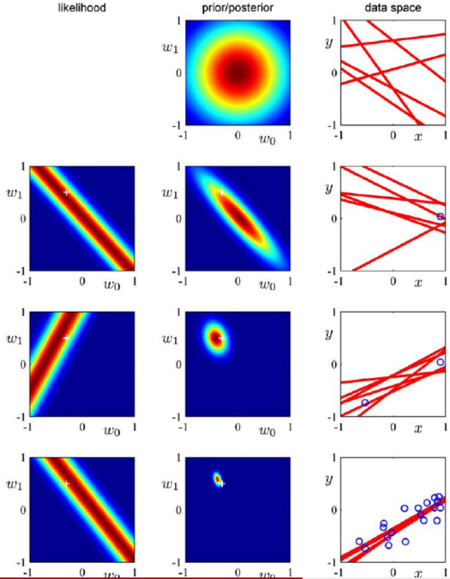
$$t = 0.5x - 0.3 + \epsilon$$

$$x \in \mathcal{U}[-1\ 1] \qquad \epsilon \in \mathcal{N}(0, 0.2^2)$$

- Assume: $y(x, \mathbf{w}) = w_1 x + w_0$
- Assume noise variance is known

$$\beta = \frac{1}{0.2^2} \qquad \alpha = 2.0$$

Bayes Updates Illustration: $t = a_0 + a_1 x + \epsilon$



- Actual targets are generated as

$$t = 0.5x - 0.3 + \epsilon$$

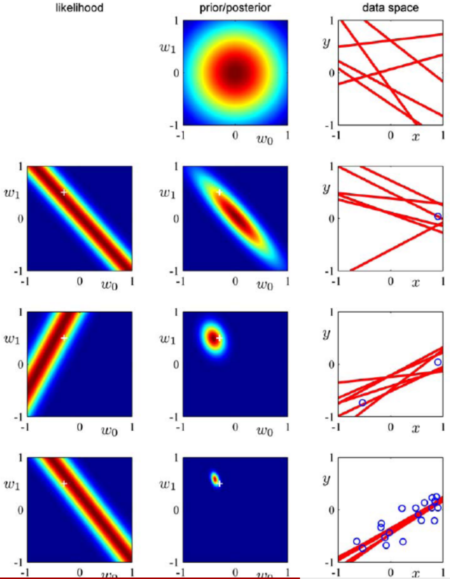
$$x \in \mathcal{U}[-1\ 1] \qquad \epsilon \in \mathcal{N}(0, 0.2^2)$$

- Assume: $y(x, \mathbf{w}) = w_1 x + w_0$
- Assume noise variance is known

$$\beta = \frac{1}{0.2^2} \qquad \alpha = 2.0$$

- Seq. update posterior $p(\mathbf{w}/\mathbf{t})$

Bayes Updates Illustration: $t = a_0 + a_1 x + \epsilon$



- Actual targets are generated as

$$t = 0.5x - 0.3 + \epsilon$$

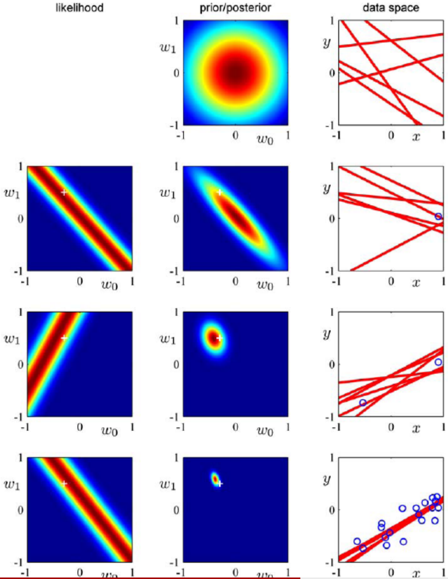
$$x \in \mathcal{U}[-1\ 1] \qquad \epsilon \in \mathcal{N}(0, 0.2^2)$$

- Assume: $y(x, \mathbf{w}) = w_1 x + w_0$
- Assume noise variance is known

$$\beta = \frac{1}{0.2^2} \qquad \alpha = 2.0$$

- Seq. update posterior $p(\mathbf{w}/\mathbf{t})$
- Draw random samples from $p(\mathbf{w}/\mathbf{t})$ and plot y$= w_1 x + w_0$

Bayes Updates Illustration: $t = a_0 + a_1 x + \epsilon$



- Actual targets are generated as

$$t = 0.5x - 0.3 + \epsilon$$

$$x \in \mathcal{U}[-1\ 1] \qquad \epsilon \in \mathcal{N}(0, 0.2^2)$$

- Assume: $y(x, \mathbf{w}) = w_1 x + w_0$
- Assume noise variance is known

$$\beta = \frac{1}{0.2^2} \qquad \alpha = 2.0$$

- Seq. update posterior $p(\mathbf{w}/\mathbf{t})$
- Draw random samples from $p(\mathbf{w}/\mathbf{t})$ and plot y= $w_1 x + w_0$
- Lines converge as data increase

Homework

- Derive the statistics of the posterior distribution $p(\mathbf{w}/\mathbf{t})$ by completing the quadratic term of exp(.)
- Given a Gaussian marginal distribution for $\mathbf{x}$ and a Gaussian conditional distribution for y in the form

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}/\mu, \mathbf{\Lambda})$$
$$p(\mathbf{y}/\mathbf{x}) = \mathcal{N}(\mathbf{y}/\mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{L}^{-1})$$

show that the marginal distribution of $\mathbf{y}$ and conditional distribution of $\mathbf{x}$ are given by

$$p(\mathbf{y}) = \mathcal{N}\left(\mathbf{y}/\mathbf{A}\mu + \mathbf{b}, \mathbf{L}^{-1} + \mathbf{A}\mathbf{\Lambda}^{-1}\mathbf{A}^{\mathsf{T}}\right)$$
$$p(\mathbf{x}/\mathbf{y}) = \mathcal{N}\left(\mathbf{x}/\mathbf{\Sigma}\left(\mathbf{A}^{\mathsf{T}}\mathbf{L}(\mathbf{y}-\mathbf{b}) + \mathbf{\Lambda}\mu\right), \mathbf{\Sigma}\right)$$

where $\mathbf{\Sigma} = \left(\mathbf{\Lambda} + \mathbf{A}^{\mathsf{T}}\mathbf{L}\mathbf{A}\right)^{-1}$

Predictive Distributions

Predictive Distributions

- Given a training set of N points $(\mathbf{x}_{1:N}, t_{1:N})$, predict target distribution for a new input $\mathbf{x}_0$

$$\begin{aligned} p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) &= \int p(t_0, \mathbf{w}/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w} \\ &= \int p(t_0/\mathbf{w}, \mathbf{x}_0, \beta) p(\mathbf{w}/\mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w} \end{aligned}$$

Predictive Distributions

- Given a training set of N points $(\mathbf{x}_{1:N}, t_{1:N})$, predict target distribution for a new input $\mathbf{x}_0$

$$\begin{aligned} p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) &= \int p(t_0, \mathbf{w}/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w} \\ &= \int p(t_0/\mathbf{w}, \mathbf{x}_0, \beta) p(\mathbf{w}/\mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w} \end{aligned}$$

- The predictive distribution is Gaussian and is given by

$$\begin{aligned} p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) &= \mathcal{N}\left(t_0/\mathbf{m}_N^T \phi(\mathbf{x}_0), \sigma_N^2(\mathbf{x}_0)\right) \\ \sigma_N^2(\mathbf{x}_0) &= \frac{1}{\beta} + \phi^{\mathsf{T}}(\mathbf{x}_0) \mathbf{\Sigma}_N \phi(\mathbf{x}_0) \end{aligned}$$

Predictive Distributions

- Given a training set of N points $(\mathbf{x}_{1:N}, t_{1:N})$, predict target distribution for a new input $\mathbf{x}_0$

$$p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) = \int p(t_0, \mathbf{w}/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w}$$
$$= \int p(t_0/\mathbf{w}, \mathbf{x}_0, \beta) p(\mathbf{w}/\mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w}$$

- The predictive distribution is Gaussian and is given by

$$p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}\left(t_0/\mathbf{m}_N^T \phi(\mathbf{x}_0), \sigma_N^2(\mathbf{x}_0)\right)$$
$$\sigma_N^2(\mathbf{x}_0) = \frac{1}{\beta} + \phi^{\mathsf{T}}(\mathbf{x}_0) \mathbf{\Sigma}_N \phi(\mathbf{x}_0)$$

- Predictive distribution gets narrower with additional training points

$$\sigma_{N+1}^2(\mathbf{x}_0) \leq \sigma_N^2(\mathbf{x}_0)$$

Predictive Distributions

- Given a training set of N points $(\mathbf{x}_{1:N}, t_{1:N})$, predict target distribution for a new input $\mathbf{x}_0$

$$p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) = \int p(t_0, \mathbf{w}/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w}$$
$$= \int p(t_0/\mathbf{w}, \mathbf{x}_0, \beta) p(\mathbf{w}/\mathbf{X}, \mathbf{t}, \alpha, \beta) d\mathbf{w}$$

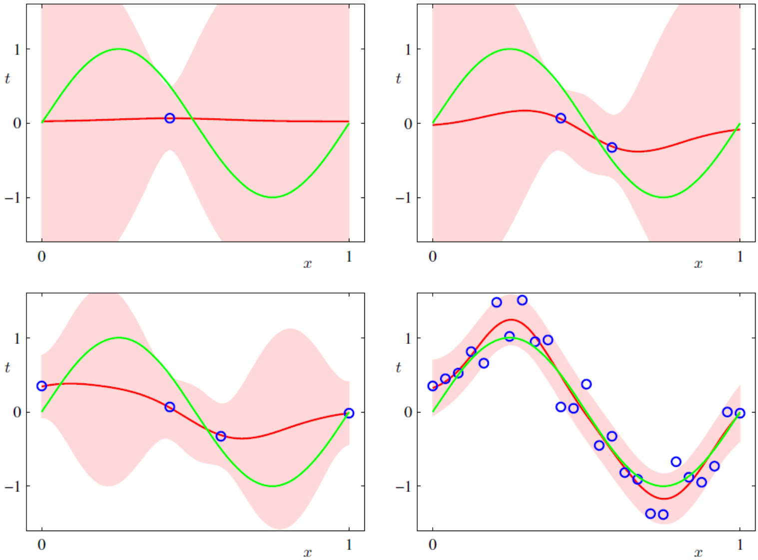
- The predictive distribution is Gaussian and is given by

$$p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}\left(t_0/\mathbf{m}_N^T \phi(\mathbf{x}_0), \sigma_N^2(\mathbf{x}_0)\right)$$
$$\sigma_N^2(\mathbf{x}_0) = \frac{1}{\beta} + \phi^\mathsf{T}(\mathbf{x}_0)\mathbf{\Sigma}_N \phi(\mathbf{x}_0)$$

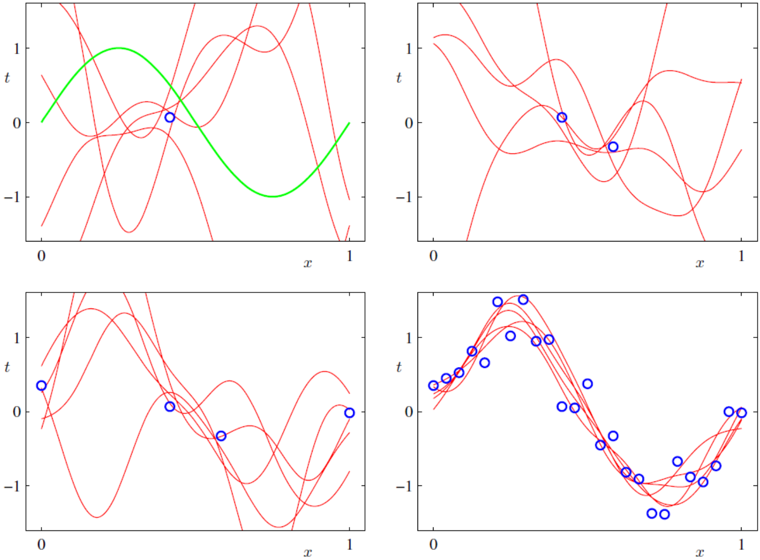
- Predictive distribution gets narrower with additional training points

$$\sigma_{N+1}^2(\mathbf{x}_0) \leq \sigma_N^2(\mathbf{x}_0) \qquad \lim_{N\to\infty} \sigma_N^2(\mathbf{x}_0) \to \frac{1}{\beta}$$

Predictive Distribution: $t = \sin(2\pi x) + \epsilon$



Curves $y(x, \mathbf{w})$ Sampled from Posterior $p(\mathbf{w}/\mathbf{t})$



Summary of Linear Models of Regression

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
    - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
    - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
    - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
    - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
    - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
    - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
  - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Regularize $\mathbf{w}$ to avoid over-fitting: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$

Summary of Linear Models of Regression

- Linear in model parameters **w**.
  - If **x** and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Regularize **w** to avoid over-fitting: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
  - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^{\mathsf{T}}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^{\mathsf{T}}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\boldsymbol{\Phi}^{\mathsf{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathsf{T}}\mathbf{t}$
  - Regularize $\mathbf{w}$ to avoid over-fitting: $\mathbf{w}_* = (\boldsymbol{\Phi}^{\mathsf{T}}\boldsymbol{\Phi} + \lambda\mathbf{I})^{-1}\boldsymbol{\Phi}^{\mathsf{T}}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning
- Probabilistic interpretation to regression

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
  - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^{\mathsf{T}}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^{\mathsf{T}}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathsf{T}}\mathbf{t}$
  - Regularize $\mathbf{w}$ to avoid over-fitting: $\mathbf{w}_* = (\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^{\mathsf{T}}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning
- Probabilistic interpretation to regression
  - Point estimate of target does not hold for one-to-many maps

Summary of Linear Models of Regression

- Linear in model parameters **w**.
  - If **x** and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Regularize **w** to avoid over-fitting: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning
- Probabilistic interpretation to regression
  - Point estimate of target does not hold for one-to-many maps
  - ML estimation assigns a distribution to the target $p(t_n/y(\mathbf{w}, \mathbf{x}_n), \beta^{-1})$

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
  - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Regularize $\mathbf{w}$ to avoid over-fitting: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning
- Probabilistic interpretation to regression
  - Point estimate of target does not hold for one-to-many maps
  - ML estimation assigns a distribution to the target $p(t_n/y(\mathbf{w}, \mathbf{x}_n), \beta^{-1})$
  - Parameters $\mathbf{w}$ depend on training set - point estimate not enough

Summary of Linear Models of Regression

- Linear in model parameters **w**.
  - If **x** and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^\mathsf{T}\mathbf{t}$
  - Regularize **w** to avoid over-fitting: $\mathbf{w}_* = (\boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi} + \lambda\mathbf{I})^{-1}\boldsymbol{\Phi}^\mathsf{T}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning
- Probabilistic interpretation to regression
  - Point estimate of target does not hold for one-to-many maps
  - ML estimation assigns a distribution to the target $p(t_n/y(\mathbf{w}, \mathbf{x}_n), \beta^{-1})$
  - Parameters **w** depend on training set - point estimate not enough
  - MAP estimation assigns a distribution to **w**: $p(\mathbf{w}/\mathbf{t}, \mathbf{X}, \alpha, \beta)$

Summary of Linear Models of Regression

- Linear in model parameters $\mathbf{w}$.
  - If $\mathbf{x}$ and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Regularize $\mathbf{w}$ to avoid over-fitting: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning
- Probabilistic interpretation to regression
  - Point estimate of target does not hold for one-to-many maps
  - ML estimation assigns a distribution to the target $p(t_n/y(\mathbf{w}, \mathbf{x}_n), \beta^{-1})$
  - Parameters $\mathbf{w}$ depend on training set - point estimate not enough
  - MAP estimation assigns a distribution to $\mathbf{w}$: $p(\mathbf{w}/\mathbf{t}, \mathbf{X}, \alpha, \beta)$
  - Predict target distribution for a test-point x_0: $p(t_0/x_0, \mathbf{t}, \mathbf{X}, \alpha, \beta)$

Summary of Linear Models of Regression

- Linear in model parameters **w**.
  - If **x** and t are linearly related $\hat{t} = \mathbf{w}^\mathsf{T}\mathbf{x}$
  - If relationship is not linear: $\hat{t} = \mathbf{w}^\mathsf{T}\phi(\mathbf{x})$
  - LS criterion leads to pseudo-inverse solution: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Regularize **w** to avoid over-fitting: $\mathbf{w}_* = (\mathbf{\Phi}^\mathsf{T}\mathbf{\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{t}$
  - Gradient descent algorithms can be used for sequential learning
- Probabilistic interpretation to regression
  - Point estimate of target does not hold for one-to-many maps
  - ML estimation assigns a distribution to the target $p(t_n/y(\mathbf{w}, \mathbf{x}_n), \beta^{-1})$
  - Parameters **w** depend on training set - point estimate not enough
  - MAP estimation assigns a distribution to **w**: $p(\mathbf{w}/\mathbf{t}, \mathbf{X}, \alpha, \beta)$
  - Predict target distribution for a test-point x_0: $p(t_0/x_0, \mathbf{t}, \mathbf{X}, \alpha, \beta)$
  - Predictive uncertainty depends on x_0 and is smallest in the neighborhood of train data points.

Homework

- For Gaussian likelihood and Gaussian posterior, prove that the he predictive distribution is Gaussian and is given by

$$p(t_0/\mathbf{x}_0, \mathbf{X}, \mathbf{t}, \alpha, \beta) = \mathcal{N}\left(t_0/\mathbf{m}_N^T \phi(\mathbf{x}_0), \sigma_N^2(\mathbf{x}_0)\right)$$
$$\sigma_N^2(\mathbf{x}_0) = \frac{1}{\beta} + \boldsymbol{\phi}^{\mathsf{T}}(\mathbf{x}_0)\boldsymbol{\Sigma}_N \phi(\mathbf{x}_0)$$

- Prove that the predictive uncertainty deceases with increase in training data, i.e., predictive distribution gets narrower with additional training points

$$\sigma_{N+1}^2(\mathbf{x}_0) \leq \sigma_N^2(\mathbf{x}_0)$$

Thank You!