

CS545: Linear Models (Nonlinear Inputs; Probabilistic)

Chuck Anderson

Department of Computer Science
Colorado State University

Fall, 2009

Outline

CS545: Linear
Models (Nonlinear
Inputs;
Probabilistic)

Chuck Anderson

Bayesian
Regression

Derivation

Application to Auto
MPG Data

Application to 1-D
Data

Bayesian Regression

Derivation

Application to Auto MPG Data

Application to 1-D Data

Bayesian Regression

- The full Bayesian approach to regression does not solve for a single \mathbf{w} value. Instead, an expression for the probability of a model, given the data, is formulated. Then, a sum is taken over **all possible models** of the prediction value for a given model weighted by the probability of that model.

Bayesian Regression

- The full Bayesian approach to regression does not solve for a single \mathbf{w} value. Instead, an expression for the probability of a model, given the data, is formulated. Then, a sum is taken over **all possible models** of the prediction value for a given model weighted by the probability of that model.
- So

$$\begin{aligned} p(t_n | \mathbf{x}_n, \mathbf{X}, \mathbf{T}) &= \int p(t_n, \text{model} | \mathbf{x}_n, \mathbf{X}, \mathbf{T}) d\text{model} \\ &= \int p(t_n | \text{model}, \mathbf{x}_n, \mathbf{X}, \mathbf{T}) p(\text{model} | \mathbf{X}, \mathbf{T}) d\text{model} \end{aligned}$$

Bayesian Regression

- The full Bayesian approach to regression does not solve for a single \mathbf{w} value. Instead, an expression for the probability of a model, given the data, is formulated. Then, a sum is taken over **all possible models** of the prediction value for a given model weighted by the probability of that model.
- So

$$\begin{aligned} p(t_n | \mathbf{x}_n, \mathbf{X}, \mathbf{T}) &= \int p(t_n, \text{model} | \mathbf{x}_n, \mathbf{X}, \mathbf{T}) d\text{model} \\ &= \int p(t_n | \text{model}, \mathbf{x}_n, \mathbf{X}, \mathbf{T}) p(\text{model} | \mathbf{X}, \mathbf{T}) d\text{model} \end{aligned}$$

- As before, let's choose the model to be $y(\mathbf{x}_n, \mathbf{w}) = \phi(\mathbf{x}_n)\mathbf{w} + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \beta^{-1})$. With this choice

$$\begin{aligned} p(t_n | \text{model}, \mathbf{x}_n, \mathbf{X}, \mathbf{T}) &= p(t_n | \mathbf{w}, \phi, \mathbf{x}_n, \mathbf{X}, \mathbf{T}, \beta) \\ &= \mathcal{N}(t_n | \phi(\mathbf{x}_n)\mathbf{w}, \beta^{-1}) \end{aligned}$$

- Bayes Theorem tells us

$$p(\text{model}|\mathbf{X}, \mathbf{T}) \propto p(\mathbf{T}|\mathbf{X}, \text{model})p(\text{model})$$

or

$$p(\mathbf{w}|\phi, \mathbf{X}, \mathbf{T}, \beta) \propto p(\mathbf{T}|\mathbf{w}, \phi, \mathbf{X}, \beta)p(\mathbf{w})$$

- Bayes Theorem tells us

$$p(\text{model}|\mathbf{X}, \mathbf{T}) \propto p(\mathbf{T}|\mathbf{X}, \text{model})p(\text{model})$$

or

$$p(\mathbf{w}|\phi, \mathbf{X}, \mathbf{T}, \beta) \propto p(\mathbf{T}|\mathbf{w}, \phi, \mathbf{X}, \beta)p(\mathbf{w})$$

- We will model the data likelihood function as a Gaussian,

$$\begin{aligned} p(\mathbf{T}|\mathbf{w}, \phi, \mathbf{X}, \beta) &= \mathcal{N}(\mathbf{T}|\Phi\mathbf{w}, \beta^{-1}I) \\ &= \prod_{n=1}^N \mathcal{N}(t_n|\phi(\mathbf{x}_n)\mathbf{w}, \beta^{-1}) \end{aligned}$$

- Bayes Theorem tells us

$$p(\text{model}|\mathbf{X}, \mathbf{T}) \propto p(\mathbf{T}|\mathbf{X}, \text{model})p(\text{model})$$

or

$$p(\mathbf{w}|\phi, \mathbf{X}, \mathbf{T}, \beta) \propto p(\mathbf{T}|\mathbf{w}, \phi, \mathbf{X}, \beta)p(\mathbf{w})$$

- We will model the data likelihood function as a Gaussian,

$$\begin{aligned} p(\mathbf{T}|\mathbf{w}, \phi, \mathbf{X}, \beta) &= \mathcal{N}(\mathbf{T}|\Phi\mathbf{w}, \beta^{-1}I) \\ &= \prod_{n=1}^N \mathcal{N}(t_n|\phi(\mathbf{x}_n)\mathbf{w}, \beta^{-1}) \end{aligned}$$

- Again, choose the prior distribution of the weights to be the zero-mean Gaussian

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

- Now the original equation becomes

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

- Now the original equation becomes

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

- Let's work on the last two terms in the integral. They involve products of Gaussians, which involve products of exponential terms with base e . These products are formed by adding the exponents, so let's focus on the value of the exponent of e . The sum we get is

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

- Continuing working with the sum, we get

$$\begin{aligned} & -\frac{1}{2}(\beta \sum_{n=1}^N (t_n - \phi(\mathbf{x}_n)\mathbf{w})^2 + \mathbf{w}^T \alpha / \mathbf{w}) \\ &= -\frac{1}{2}\beta \sum_{n=1}^N t_n^2 + \sum_{n=1}^N t_n \phi(\mathbf{x}_n)\mathbf{w} - \frac{1}{2}(\sum_{n=1}^N \phi(\mathbf{x}_n)\mathbf{w})^2 - \frac{1}{2}\mathbf{w}^T \alpha / \mathbf{w} \\ &= -\frac{1}{2}\beta \mathbf{T}^T \mathbf{T} + \beta \mathbf{T}^T \Phi \mathbf{w} - \frac{1}{2}\beta \mathbf{w}^T \Phi^T \Phi \mathbf{w} - \frac{1}{2}\mathbf{w}^T \alpha / \mathbf{w} \\ &= -\frac{1}{2}\mathbf{w}^T (\beta \Phi^T \Phi + \alpha I) \mathbf{w} + \mathbf{w}^T \beta \Phi^T \mathbf{T} - \frac{1}{2}\beta \mathbf{T}^T \mathbf{T} \end{aligned}$$

- Continuing working with the sum, we get

$$\begin{aligned} & -\frac{1}{2}(\beta \sum_{n=1}^N (t_n - \phi(\mathbf{x}_n)\mathbf{w})^2 + \mathbf{w}^T \alpha / \mathbf{w}) \\ &= -\frac{1}{2}\beta \sum_{n=1}^N t_n^2 + \sum_{n=1}^N t_n \phi(\mathbf{x}_n)\mathbf{w} - \frac{1}{2}(\sum_{n=1}^N \phi(\mathbf{x}_n)\mathbf{w})^2 - \frac{1}{2}\mathbf{w}^T \alpha / \mathbf{w} \\ &= -\frac{1}{2}\beta \mathbf{T}^T \mathbf{T} + \beta \mathbf{T}^T \Phi \mathbf{w} - \frac{1}{2}\beta \mathbf{w}^T \Phi^T \Phi \mathbf{w} - \frac{1}{2}\mathbf{w}^T \alpha / \mathbf{w} \\ &= -\frac{1}{2}\mathbf{w}^T (\beta \Phi^T \Phi + \alpha I) \mathbf{w} + \mathbf{w}^T \beta \Phi^T \mathbf{T} - \frac{1}{2}\beta \mathbf{T}^T \mathbf{T} \end{aligned}$$

- Notice the terms are grouped by ones quadratic and linear in \mathbf{w} . This allows us to figure out what the Gaussian distribution is. After all, a product of two Gaussians must be another Gaussian.

Reverse engineering a Gaussian

- Look at expression (2.71) in Bishop's book and the text around it.

$$\begin{aligned} -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}) = \\ -\frac{1}{2}\mathbf{x}^T \boldsymbol{\Sigma}^{-1}\mathbf{x} + \mathbf{x}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \text{const} \end{aligned}$$

Reverse engineering a Gaussian

- Look at expression (2.71) in Bishop's book and the text around it.

$$\begin{aligned} -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}) = \\ -\frac{1}{2}\mathbf{x}^T \boldsymbol{\Sigma}^{-1}\mathbf{x} + \mathbf{x}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \text{const} \end{aligned}$$

- Comparing with our previous expression

$$-\frac{1}{2}\mathbf{w}^T(\beta\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \alpha I)\mathbf{w} + \mathbf{w}^T\beta\boldsymbol{\Phi}^T\mathbf{T} - \frac{1}{2}\beta\mathbf{T}^T\mathbf{T}$$

Reverse engineering a Gaussian

- Look at expression (2.71) in Bishop's book and the text around it.

$$\begin{aligned} -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}) = \\ -\frac{1}{2}\mathbf{x}^T \boldsymbol{\Sigma}^{-1}\mathbf{x} + \mathbf{x}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \text{const} \end{aligned}$$

- Comparing with our previous expression

$$-\frac{1}{2}\mathbf{w}^T (\beta \boldsymbol{\Phi}^T \boldsymbol{\Phi} + \alpha I) \mathbf{w} + \mathbf{w}^T \beta \boldsymbol{\Phi}^T \mathbf{T} - \frac{1}{2} \beta \mathbf{T}^T \mathbf{T}$$

- we see that

$$\boldsymbol{\Sigma}^{-1} = \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi} + \alpha I$$

- Unfortunately, with this definition, the linear terms do not match. To make them match, we can introduce $\Sigma^{-1}\Sigma = I$ at the right place, so

$$\mathbf{w}^T \beta \Phi^T \mathbf{T} \text{ becomes } \mathbf{w}^T \Sigma^{-1} \Sigma \beta \Phi^T \mathbf{T}$$

- Unfortunately, with this definition, the linear terms do not match. To make them match, we can introduce $\Sigma^{-1}\Sigma = I$ at the right place, so

$$\mathbf{w}^T \beta \Phi^T \mathbf{T} \text{ becomes } \mathbf{w}^T \Sigma^{-1} \Sigma \beta \Phi^T \mathbf{T}$$

- making the full exponent

$$\begin{aligned} & -\frac{1}{2} \mathbf{w}^T (\beta \Phi^T \Phi + \alpha I) \mathbf{w} + \\ & \mathbf{w}^T (\beta \Phi^T \Phi + \alpha I) (\beta \Phi^T \Phi + \alpha I)^{-1} \beta \Phi^T \mathbf{T} - \\ & \frac{1}{2} \beta \mathbf{T}^T \mathbf{T} \end{aligned}$$

- Unfortunately, with this definition, the linear terms do not match. To make them match, we can introduce $\Sigma^{-1}\Sigma = I$ at the right place, so

$$\mathbf{w}^T \beta \Phi^T \mathbf{T} \text{ becomes } \mathbf{w}^T \Sigma^{-1} \Sigma \beta \Phi^T \mathbf{T}$$

- making the full exponent

$$\begin{aligned} & -\frac{1}{2} \mathbf{w}^T (\beta \Phi^T \Phi + \alpha I) \mathbf{w} + \\ & \mathbf{w}^T (\beta \Phi^T \Phi + \alpha I) (\beta \Phi^T \Phi + \alpha I)^{-1} \beta \Phi^T \mathbf{T} - \\ & \frac{1}{2} \beta \mathbf{T}^T \mathbf{T} \end{aligned}$$

- Now we can identify the mean

$$\boldsymbol{\mu} = \beta \Sigma \Phi^T \mathbf{T}.$$

- Renaming the mean and covariance matrix of this distribution of \mathbf{w} to be

$$S_w = \Sigma = (\beta \Phi^T \Phi + \alpha I)^{-1}$$

$$\mu_w = \mu = \beta S_w \Phi^T \mathbf{T},$$

- Renaming the mean and covariance matrix of this distribution of \mathbf{w} to be

$$S_w = \Sigma = (\beta \Phi^T \Phi + \alpha I)^{-1}$$
$$\mu_w = \mu = \beta S_w \Phi^T \mathbf{T},$$

- We have now identified the resulting Gaussian to be

$$\mathcal{N}(\mu_w, S_w).$$

Our book uses variable names μ_N and S_N for μ_w and S_w .

- The initial integral

$$\begin{aligned} p(t_n | \mathbf{x}_n, \mathbf{X}, \mathbf{T}) &= \int \mathcal{N}(t_n | \phi(\mathbf{x}_n) \mathbf{w}, \beta^{-1}) \\ &\prod_{n=1}^N \mathcal{N}(t_n | \phi(\mathbf{x}_n) \mathbf{w}, \beta^{-1}) \mathcal{N}(\mathbf{w} | 0, \alpha^{-1} I) d\mathbf{w} \end{aligned}$$

is now

$$p(t_n | \mathbf{x}, \mathbf{X}, \mathbf{T}) = \int \mathcal{N}(t_n | \phi(\mathbf{x}_n) \mathbf{w}, \beta^{-1}) \mathcal{N}(\mathbf{w} | \mu_w, S_w) d\mathbf{w}$$

Sheesh! Another Gaussian Product

- To deal with the integral of the product of these two Gaussians, we will first form the joint distribution of t and w and develop an expression for the mean and covariance matrix of the joint distribution. First, form a vector containing both t_n and \mathbf{w} and call it \mathbf{z} :

$$\mathbf{z} = \begin{pmatrix} t_n \\ \mathbf{w} \end{pmatrix}$$

Sheesh! Another Gaussian Product

- To deal with the integral of the product of these two Gaussians, we will first form the joint distribution of t and w and develop an expression for the mean and covariance matrix of the joint distribution. First, form a vector containing both t_n and \mathbf{w} and call it z :

$$z = \begin{pmatrix} t_n \\ \mathbf{w} \end{pmatrix}$$

- Since t_n and \mathbf{w} are independent $p(z) = p(t_n)p(\mathbf{w})$ and when we multiply these two Gaussians, we just add their exponents.

- $p(t_n)p(\mathbf{w})$ is

$$\begin{aligned} & -\frac{1}{2}\beta t_n^2 + \beta t_n \phi(\mathbf{x}_n) \mathbf{w} - \frac{1}{2} \mathbf{w}^T \phi(\mathbf{w})^T \phi(\mathbf{w}_n) \mathbf{w} \\ & - \frac{1}{2} \mathbf{w}^T S_w^{-1} \mathbf{w} + \mathbf{w}^T S_w^{-1} \mu_w - \frac{1}{2} \mu_w^T S_w^{-1} \mu_w \\ & = -\frac{1}{2} \begin{pmatrix} t_n \\ \mathbf{w} \end{pmatrix}^T \begin{pmatrix} \beta & \phi(\mathbf{x}_n) \\ \phi(\mathbf{x}_n) & \phi(\mathbf{x}_n) \phi(\mathbf{x}_n)^T - S_w^{-1} \end{pmatrix} \begin{pmatrix} t_n \\ \mathbf{w} \end{pmatrix} \\ & \quad + \begin{pmatrix} t_n \\ \mathbf{w} \end{pmatrix}^T \begin{pmatrix} 0 \\ S_w^{-1} \mu_w \end{pmatrix} + \text{const} \end{aligned}$$

- At this point, we have followed the derivation in Section 2.3.3 up to Equation 2.106. Continuing this derivation would result in (meaning I don't have slides for this) determining that

$$\begin{aligned} p(t_n | \mathbf{x}, \mathbf{X}, \mathbf{T}) &= \int \mathcal{N}(t_n | \phi(\mathbf{x}_n) \mathbf{w}, \beta^{-1}) \mathcal{N}(\mathbf{w} | \mu_w, S_w) d\mathbf{w}, \\ &= \mathcal{N}(t_n | \mu_t, S_t) \end{aligned}$$

where

$$\begin{aligned} \mu_t &= \phi(\mathbf{x}) \mu_w \\ S_t &= \frac{1}{\beta} + \phi(\mathbf{x})^T S_w \phi(\mathbf{x}). \end{aligned}$$

- At this point, we have followed the derivation in Section 2.3.3 up to Equation 2.106. Continuing this derivation would result in (meaning I don't have slides for this) determining that

$$\begin{aligned} p(t_n | \mathbf{x}, \mathbf{X}, \mathbf{T}) &= \int \mathcal{N}(t_n | \phi(\mathbf{x}_n) \mathbf{w}, \beta^{-1}) \mathcal{N}(\mathbf{w} | \mu_w, S_w) d\mathbf{w}, \\ &= \mathcal{N}(t_n | \mu_t, S_t) \end{aligned}$$

where

$$\begin{aligned} \mu_t &= \phi(\mathbf{w}) \mu_w \\ S_t &= \frac{1}{\beta} + \phi(\mathbf{x})^T S_w \phi(\mathbf{x}). \end{aligned}$$

- To use this result, we build our model by first calculating μ_w and S_w given some training data \mathbf{X} and \mathbf{T} .

- At this point, we have followed the derivation in Section 2.3.3 up to Equation 2.106. Continuing this derivation would result in (meaning I don't have slides for this) determining that

$$\begin{aligned} p(t_n | \mathbf{x}, \mathbf{X}, \mathbf{T}) &= \int \mathcal{N}(t_n | \phi(\mathbf{x}_n) \mathbf{w}, \beta^{-1}) \mathcal{N}(\mathbf{w} | \mu_w, S_w) d\mathbf{w}, \\ &= \mathcal{N}(t_n | \mu_t, S_t) \end{aligned}$$

where

$$\begin{aligned} \mu_t &= \phi(\mathbf{w}) \mu_w \\ S_t &= \frac{1}{\beta} + \phi(\mathbf{x})^T S_w \phi(\mathbf{x}). \end{aligned}$$

- To use this result, we build our model by first calculating μ_w and S_w given some training data \mathbf{X} and \mathbf{T} .
- Then, for every data sample \mathbf{x} we predict the target value $t_{\text{predicted}} = \phi(\mathbf{x}) \mu_w$, an expression very much like our frequentist solution. In addition, the Bayesian approach gives us the variance S_t of the prediction.

Application to Auto MPG Data

- Calculate μ_w and S_w , after choosing some hyperparameters.

$$S_w = \Sigma = (\beta \Phi^T \Phi + \alpha I)^{-1}$$

$$\mu_w = \mu = \beta S_w \Phi^T \mathbf{T},$$

```
> beta <- 10
> alpha <- 10
> M <- 7
> S.w <- solve(beta * t(X1) %**% X1 + alpha * diag(1,M+1))
> mu.w <- beta * S.w %**% t(X1) %**% T
```

- Then, for the test data, calculate the predicted MPG distribution parameters.

$$\mu_t = \phi(\mathbf{w})\mu_w$$
$$S_t = \frac{1}{\beta} + \phi(\mathbf{x})^T S_w \phi(\mathbf{x}).$$

```
> pred.mu <- Xtest1 %*% mu.w
> pred.var <- c()
> for (i in 1:nrow(Xtest1)) {
+   x <- Xtest1[i,,drop=FALSE]
+   pred.var <- c(pred.var, 1/beta + x %*% S.w %*% t(x)
+ }
> for (i in 1:10) {cat(pred.mu[i], "+-", sqrt(pred.var[i]), "\n") }
13.88899 +- 0.2022685
14.79652 +- 0.2033407
10.91201 +- 0.2054595
18.77275 +- 0.2021981
20.99154 +- 0.2025548
22.23494 +- 0.2024484
8.29176 +- 0.2082009
22.83277 +- 0.2029982
25.06364 +- 0.2018831
29.37667 +- 0.2036604
```

- Then, for the test data, calculate the predicted MPG distribution parameters.

$$\mu_t = \phi(\mathbf{w})\mu_w$$

$$S_t = \frac{1}{\beta} + \phi(\mathbf{x})^T S_w \phi(\mathbf{x}).$$

```
> pred.mu <- Xtest1 %*% mu.w
> pred.var <- c()
> for (i in 1:nrow(Xtest1)) {
+   x <- Xtest1[i,,drop=FALSE]
+   pred.var <- c(pred.var, 1/beta + x %*% S.w %*% t(x)
+ }
> for (i in 1:10) {cat(pred.mu[i], "+-", sqrt(pred.var[i]), "\n") }
```

13.88899 +- 0.2022685
14.79652 +- 0.2033407
10.91201 +- 0.2054595
18.77275 +- 0.2021981
20.99154 +- 0.2025548
22.23494 +- 0.2024484
8.29176 +- 0.2082009
22.83277 +- 0.2029982
25.06364 +- 0.2018831
29.37667 +- 0.2036604

- Will calculate S_T in matrix form a few slides later.

Now something a little easier to understand

- Let's try fitting a sine curve using radial basis functions. First generate the x values and target values.

```
Xall <- matrix(c(0.4,0.6,0.1, runif(50)))
nRBFs <- 9
rbfWidth <- 0.2
xrange <- range(Xall)
rbfCenters <- seq(xrange[1],xrange[2],len=nRBFs)
XallRBF <- rbfnize1D(Xall, centers=rbfCenters, widths=rbfWidth)
beta <- 25
### Assume we know beta for target distribution.
Tall <- matrix(sin(2*pi*Xall)+rnorm(nrow(Xall),0,sqrt(1/beta)))
```

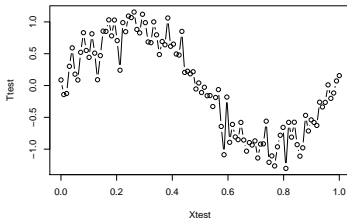
Now something a little easier to understand

- Let's try fitting a sine curve using radial basis functions. First generate the x values and target values.

```
Xall <- matrix(c(0.4,0.6,0,1, runif(50)))  
nRBFs <- 9  
rbfWidth <- 0.2  
xrange <- range(Xall)  
rbfCenters <- seq(xrange[1],xrange[2],len=nRBFs)  
XallRBF <- rbfize1D(Xall, centers=rbfCenters, widths=rbfWidth)  
beta <- 25  
### Assume we know beta for target distribution.  
Tall <- matrix(sin(2*pi*Xall)+rnorm(nrow(Xall),0,sqrt(1/beta)))
```

- Then the test data.

```
nTest <- 100  
Xtest <- matrix(seq(0,1,len=nTest))  
XtestRBF <- rbfize1D(Xtest, centers=rbfCenters, widths=rbfWidth)
```

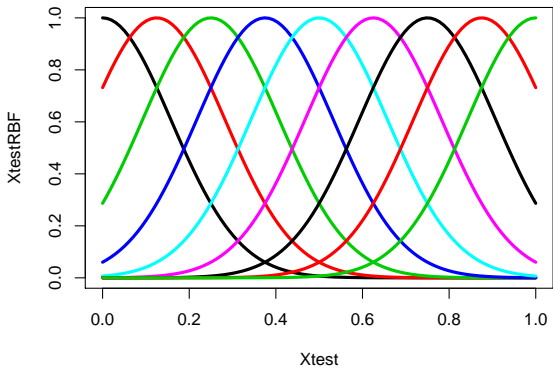


- What do the RBFs look like?

```
matplot(Xtest,XtestRBF,type="l",lwd=3,lty=1)
```

- What do the RBFs look like?

```
matplotlib(Xtest,XtestRBF,type="l",lwd=3,lty=1)
```



- Now, let's perform Bayesian Regression to form a model.

$$S_w = \Sigma = (\beta \Phi^T \Phi + \alpha I)^{-1}$$

$$\mu_w = \mu = \beta S_w \Phi^T \mathbf{T}$$

$$\mu_t = \phi(\mathbf{w}) \mu_w$$

$$S_t = \frac{1}{\beta} + \phi(\mathbf{x})^T S_w \phi(\mathbf{x}).$$

Remember, the last two equations are for a single sample \mathbf{x} , but we want the R code to handle multiple samples.

```
makeBayesReg <- function(X, T, alpha, beta) {
  X1 <- cbind(1,X)
  M <- ncol(X1)
  S.w <- solve(beta * t(X1) %*% X1 + alpha * diag(1,M))
  mu.w <- beta * S.w %*% t(X1) %*% T
  list (mu.w = mu.w, S.w = S.w, alpha = alpha, beta = beta)
}

useBayesReg <- function(model, X) {
  X1 <- cbind(1,X)
  mu.t <- X1 %*% model$mu.w
  S.t <- 1/model$beta + rowSums(X1 %*% model$S.w * X1)
  append(list(mean = mu.t, variance = S.t), model)
}
```

- Now use these for different sizes of training sets to see how this affects the results. We are generating Figure 3.8 in Bishop's text.

```
alpha <- 0.1 ## other precision parameter we need
def.par <- par(mar=c(2,2,3,1))
layout(matrix(c(1,2,3,4),2,2, byrow=TRUE)) \textcolor{red}{## read ?layout}

for (nTrain in c(1,2,4,25)) {

  Xtrain <- Xall[1:nTrain,,drop=FALSE]
  XtrainRBF <- XallRBF[1:nTrain,,drop=FALSE]
  Ttrain <- Tall[1:nTrain,,drop=FALSE]

  model <- makeBayesReg(XtrainRBF,Ttrain,alpha,beta)
  predictions <- useBayesReg(model,XtestRBF)

  plot.BayesRegression1D(model,Xtrain,Ttrain,Xtest, predictions ,nRBFs,rbfWidth)
}
par(def.par)
```

- That plot function is below. Uses `polygon`. Read examples at end of `?polygon`.

```
plot.BayesRegression1D <- function(model,Xtrain,Ttrain,Xtest,predictions,nRBFs,rbfWidth) {  
  upper <- predictions$mean + sqrt(predictions$variance)  
  lower <- predictions$mean - sqrt(predictions$variance)  
  ## See demo(graphics) to see how to draw shaded region  
  plot(c(Xtest,rev(Xtest)),c(upper,rev(lower)),type="n",xlab="x",ylab="t",  
        ylim=c(-2,2))  
  polygon(c(Xtest,rev(Xtest)),c(upper,rev(lower)),col="pink",border=NA)  
  points(Xtrain,Ttrain,col="blue")  
  lines(Xtest,predictions$mean,col="red")  
  lines(Xtest,sin(2*pi*Xtest),col="green")  
  title(paste("nRBFs =",nRBFs," rbfWidth =",rbfWidth,  
             " nTrain =",length(Xtrain),"\n",  
             " beta =",model$beta,  
             " alpha =",model$alpha))  
}
```

Green line is target curve.

Red line is model output mean.

Pink region shows model output variance.

