Abstract

Radial basis functions have now become a popular model for classification and prediction tasks. Most algorithms for their design, however, are basically iterative and lead to irreproducible results. In this tutorial, we present an innovative new approach (Shin-Goel algorithm) for the design and evaluation of the RBF model. It is based on purely algebraic concepts and yields reproducible designs. Use of this algorithm is demonstrated on some benchmark data sets, and data mining applications in software engineering and cancer class prediction are described.
Outline

1. Problems of classification and prediction
2. RBF model structure
3. Brief overview of RBF design methods
4. Algebraic algorithm of Shin and Goel
5. RBF center selection algorithm
6. Benchmark data classification modeling
7. Data mining and knowledge discovery applications
8. Summary

Problems of Classification and Prediction
Classification and Prediction

- Classification and prediction encompass a wide range of tasks of great practical significance in science and engineering, ranging from speech recognition to classifying sky objects. These are collectively called pattern recognition tasks. Humans are good at some of these, such as speech recognition, while machines are good at others, such as bar code reading.
- The discipline of building these machines is the domain of pattern recognition.
- Traditionally, statistical methods have been used for such tasks but recently neural nets are increasing employed since they can handle very large problems, and are less restrictive than statistical methods. Radial basis function is one such type of neural network.

Radial Basis Function

- RBF model is currently very popular for pattern recognition problems.
- RBF has nonlinear and linear components which can be treated separately. Also, RBF possesses significant mathematical properties of universal and best approximation. These features make RBF models attractive for many applications.
- Range of fields in which RBF model has been employed is very impressive and includes geophysics, signal processing, meteorology, orthopedics, computational fluid dynamics, and cancer classification.
Problem Definition

• The pattern recognition task is to construct a model that captures an unknown input-output mapping on the basis of limited evidence about its nature. The evidence is called the *training sample*. We wish to construct the “best” model that is as close as possible to the true but unknown mapping function. This process is called *training or modeling*.
• The training process seeks model parameters that provide a good fit to the training data and also provide good predictions on future data.

Problem Definition (cont.)

• Formally, we are given data set
  \[ D = \{(x_i, y_i) : x_i \in \mathbb{R}^d, y_i, i = 1, \ldots, n\}, \]
  in which both inputs and their corresponding outputs are made available and the outputs \(y_i\) are continuous or discrete values.
• Problem is to find a mapping function from the \(d\)-dimensional input space to the 1-dimensional output space based on the data.
Modeling Issues

• The objective of training or modeling is to determine model parameters so as to minimize the squared estimation error that can be decomposed into bias squared and variance. However, both cannot be simultaneously minimized. Therefore, we seek parameter values that give the best compromise between small bias and small variance.

• In practice, the bias squared and the variance cannot be computed because the computation requires knowledge of the true but unknown function. However, their trend can be analyzed from the shapes of the training and validation error curves.

Modeling Issues (cont.)

• Idealized relationship of these errors is shown below. Here we see the conceptual relationship between the expected training and validation errors, the so-called bias-variance dilemma.
Modeling Issues (cont.)

- Here, training error decreases with increasing model complexity; validation error decreases with model complexity up to a certain point and then begins to increase.
- We seek a model that is neither too simple nor too complex. A model that is too simple will suffer from underfitting because it does not learn enough from the data and hence provides a poor fit. On the other hand, a model that is too complicated would learn details including noise and thus suffers from overfitting. It cannot provide good generalization on unseen data.
- In summary, we seek a model that is
  - Not too simple: underfitting; not learn enough
  - Not too complicated: overfitting; not generalize well

RBF Model Structure
Function Approximation

• Suppose $D = \{(x_i, y_i): x_i \in \mathbb{R}^d, y_i \in \mathbb{R}, i = 1, \ldots, n\}$ where the underlying true but unknown function is $f^0$.

• Then, for given $D$, how to find a “best” approximating function $f^*$ for $f^0$?
  – Function approximation problem

• In practice, $F$, a certain class of functions, is assumed.
  – Approximation problem is to find a best approximation for $f^0$ from $F$.
  – An approximating function $f^*$ is called a best approximation from $F = \{f^1, f^2, \ldots, f^p\}$ if $f^*$ satisfies the following condition:

$$\|f^* - f^0\| \leq \|f^j - f^0\|, j = 1, \ldots, p$$

RBF Model for Function Approximation

• Assume
  – $F$ is a class of RBF models
  – $f^* \in F$

• Why RBF?
  – Mathematical properties
    • Universal approximation property
    • Best approximation property
  – Fast learning ability due to separation of nonlinearity and linearity during training phase (model development).
RBF Model

\[ \hat{y}_i = f(x_i) = \sum_{j=1}^{m} w_j \phi(x_i) = \sum_{j=1}^{m} \phi \left( \|x_i - \mu_j\| / \sigma_j \right) \]

- Here
  - \( \phi(\cdot) \) is a basis function
  - \( w_j \): weight
  - \( \mu_j \): center
  - \( \sigma_j \): width of basis function
  - \( m \): number of basis functions

- Choices of basis function

<table>
<thead>
<tr>
<th>Basis Function</th>
<th>( \phi(r) = \phi(|x - \mu| / \sigma) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>( \exp(-r^2/2) )</td>
</tr>
<tr>
<td>Thin plate spline</td>
<td>( r^2 \log r )</td>
</tr>
<tr>
<td>Inverse multiquadratic</td>
<td>( (r^2 + c^2)^{-1/2} )</td>
</tr>
<tr>
<td>Multiquadratic</td>
<td>( (r^2 + c^2)^{1/2} )</td>
</tr>
<tr>
<td>Cubic</td>
<td>( r^3 )</td>
</tr>
<tr>
<td>Linear</td>
<td>( r )</td>
</tr>
</tbody>
</table>

Radial Basis Function Network

\[ X = (x_1, x_2, ..., x_n)^T \in R^{nod} \]

\[ \phi_j(x) = \phi(\|x - \mu_j\|), \ j = 1, 2, ..., m \]

Input layer \quad Hidden layer of \quad Output layer
\[ m \] radial basis functions
RBF Interpolation: Sine Example

SINE EXAMPLE

- Consider sine function (Bishop, 1995) and its interpolation
  \[ h(x) = 0.5 + 0.4 \sin(2\pi x). \]
- Compute five values of \( h(x) \) at equal intervals of \( x \) in \((0, 1)\),
  add random noise from normal with mean = 0, variance = 0.25

<table>
<thead>
<tr>
<th>i</th>
<th>( x_i )</th>
<th>( h(x_i) )</th>
<th>( y_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.00</td>
<td>0.50</td>
<td>0.5582</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>0.90</td>
<td>0.9313</td>
</tr>
<tr>
<td>3</td>
<td>0.50</td>
<td>0.50</td>
<td>0.5038</td>
</tr>
<tr>
<td>4</td>
<td>0.75</td>
<td>0.10</td>
<td>0.1176</td>
</tr>
<tr>
<td>5</td>
<td>1.00</td>
<td>0.50</td>
<td>0.4632</td>
</tr>
</tbody>
</table>
- Interpolation problem: Determine Gaussian RBF \( f(x_i) \) such that
  \[ f(x_i) = y_i, \quad i = 1, 2, \ldots, 5. \]
SINE EXAMPLE (cont.)

- Construct interpolation matrix with five basis functions centered at $x$'s (assume $\sigma = 0.4$) and compute $G$:

$$
G = \begin{bmatrix}
1.0000 & 0.8226 & 0.4578 & 0.1724 & 0.0439 \\
0.8226 & 1.0000 & 0.8226 & 0.4578 & 0.1724 \\
0.4578 & 0.8226 & 1.0000 & 0.8226 & 0.4578 \\
0.1724 & 0.4578 & 0.8226 & 1.0000 & 0.8226 \\
0.0439 & 0.1724 & 0.4578 & 0.8226 & 1.0000
\end{bmatrix}
$$

- In above, e.g., $g_2$ is obtained as:

$$
g_2 = \begin{bmatrix}
\phi_2(x_1) \\
\phi_2(x_2) \\
\phi_2(x_3) \\
\phi_2(x_4) \\
\phi_2(x_5)
\end{bmatrix} = \begin{bmatrix}
\exp\left(-\frac{|x_1-0.25|^2}{2(0.4)^2}\right) \\
\exp\left(-\frac{|x_2-0.25|^2}{2(0.4)^2}\right) \\
\exp\left(-\frac{|x_3-0.25|^2}{2(0.4)^2}\right) \\
\exp\left(-\frac{|x_4-0.25|^2}{2(0.4)^2}\right) \\
\exp\left(-\frac{|x_5-0.25|^2}{2(0.4)^2}\right)
\end{bmatrix} = \begin{bmatrix}
0.8226 \\
1.0000 \\
0.8226 \\
0.4578 \\
0.1724
\end{bmatrix}
$$

SINE EXAMPLE (cont.)

Basis function centered at $\mu_2 = 0.25$
SINE EXAMPLE (cont.)

- The weights are computed from G and $y_i$ and we get
  \[
  \hat{y} = Gw.
  \]
  \[
  \hat{y}_i = -2.0978 \exp\left(-\frac{\|x_i - 0.0\|^2}{2(0.4)^2}\right) + 3.8689 \exp\left(-\frac{\|x_i - 0.25\|^2}{2(0.4)^2}\right) 
  - 0.0605 \exp\left(-\frac{\|x_i - 0.50\|^2}{2(0.4)^2}\right) - 3.6329 \exp\left(-\frac{\|x_i - 0.75\|^2}{2(0.4)^2}\right) 
  + 2.9063 \exp\left(-\frac{\|x_i - 1.0\|^2}{2(0.4)^2}\right)
  \]

- Each term is a weighted basis function

SINE EXAMPLE (cont.)

Plots of weighted basis functions outputs
SINE EXAMPLE (cont.)

Plots of true, observed and estimated values by RBF model
Brief Overview of RBF Design Methods
Brief Overview of RBF Design

• Model Parameters $P = (\mu, \sigma, w, m)$ where
  
  $\mu = [\mu_1, \mu_2, \ldots, \mu_m]$
  
  $\sigma = [\sigma_1, \sigma_2, \ldots, \sigma_m]$
  
  $w = [w_1, w_2, \ldots, w_m]$

• Design problem of RBF model
  – How to determine $P$?

• Some design approaches
  – Clustering
  – Subset selection
  – Regularization

Clustering

• Assume some value $k$, the number of basis functions is given
• Construct $k$ clusters with randomly selected initial centers
• The parameters are taken to be
  $\mu_j : j$th cluster center
  $\sigma_j : \text{average distance of each cluster to } P\text{-nearest clusters or individual distances}$
  $w_j : \text{weight}$
• Because of randomness in training phase, the design suffers from inconsistency
Subset Selection

- Assume some value of $\sigma$
  - $\mu_j$: a subset of $j$ input vectors that most contribute to output variance
  - $m$: number of basis functions that provides output variance enough to cover a prespecified threshold value
  - $w_j$: weight

Regularization

$m$: data size, i.e., number of input vectors
$\mu_j$: input vectors ($x_i$)
$w_j$: least squares method with regularized term
- Regularization parameter ($\lambda$) controls the smoothness and the degree of fit
- Computationally demanding
Algebraic Algorithm of Shin and Goel

Our Objective

- Derive a mathematical framework for design and evaluation of RBF model
- Develop an objective and systematic design methodology based on this mathematical framework
**Four Step RBF Modeling Process of SG Algorithm**

- **Step 1**: \( \sigma, \delta, D \)
- **Step 2**: Interpolation matrix, Singular value decomposition (SVD)
- **Step 3**: QR factorization with column pivoting
- **Step 4**: Pseudo inverse

\( w \)

**SG algorithm** is a learning or training algorithm to determine the values for the number of basis functions \( m \), their centers \( \mu \), widths \( \sigma \) and weights \( w \) to the output layer on the basis of the data set.

#### Design Methodology

- \( m = \text{rank} \left( G, s_1 \times \frac{\delta}{100} \right) \) where
  - \( G \): Gaussian interpolation matrix
  - \( s_1 \): first singular value of \( G \)
  - \( \delta : 100(1 - \delta\% \text{ RC of } G) \)
- \( \mu \): a subset of input vectors
  - Which provides a good compromise between structural stabilization and residual minimization
  - By QR factorization with column pivoting
- \( w : \Phi^{+}y \)
  - Where \( \Phi^{+} \) is pseudo-inverse of design matrix \( \Phi \)
RBF Model Structure

- For $D = \{(x_i, y_i): x_i \in \mathbb{R}^d, y_i \in \mathbb{R}\}$
  - input layer: $n \times d$ input matrix
  - hidden layer: $n \times m$ design matrix
  - output layer: $n \times 1$ output vector

\[
\begin{bmatrix}
  x_1 \\
  \vdots \\
  x_n \\
\end{bmatrix}
\begin{bmatrix}
  \phi_1(x_1) \\
  \vdots \\
  \phi_m(x_n) \\
\end{bmatrix}
\begin{bmatrix}
  w_1 \\
  \vdots \\
  w_m \\
\end{bmatrix}
= \begin{bmatrix}
  \phi_1(x_1)w_1 + \cdots + \phi_m(x_n)w_m \\
  \vdots \\
  \vdots \\
\end{bmatrix}
= \begin{bmatrix}
  y_1 \\
  \vdots \\
  y_n \\
\end{bmatrix}
\]

- $\Phi$ is called design matrix

- For, $\phi_j(x_i) = \phi(||x_i - \mu_j|| / \sigma_j)$, $i = 1, \ldots, n$, $j = 1, \ldots, m$
  - If $m = n$ and $\mu_j = x_j$, $j = 1, \ldots, n$ then, $\Phi$ is called interpolation matrix
  - If $m << n$, Design Matrix

Basic Matrix Properties

- Subspace spanned by a matrix
  - Given a matrix $A = [a_1 \ a_2 \ \ldots \ a_n] \in \mathbb{R}^{m \times n}$, the set of all linear combinations of these vectors builds the subspace $A$ of $\mathbb{R}^n$, i.e.,

  \[
  A = \text{span}\{a_1, a_2, \ldots, a_n\} = \{\sum_{j=1}^{n} c_j a_j : c_j \in \mathbb{R}\}
  \]

  - Subspace $A$ is said to be spanned by the matrix $A$

- Dimension of subspace
  - Let $A$ be the subspace spanned by $A$. If $\exists$ independent basis vectors $b_1, b_2, \ldots, b_k \in A$ such that

  \[
  A = \text{span}\{b_1, b_2, \ldots, b_k\}
  \]

  - Then the dimension of the subspace $A$ is $k$, i.e., $\dim(A) = k$
Basic Matrix Properties (cont.)

• Rank of a matrix
  – Let \( A \in \mathbb{R}^{m \times n} \) and \( A \) be the subspace spanned by the matrix \( A \). Then, rank of \( A \) is defined by the dimension of \( A \), the subspace spanned by \( A \). In other words,
    \[
    \text{rank}(A) = \dim(A)
    \]
• Rank deficiency
  – A matrix \( A \in \mathbb{R}^{m \times n} \) is rank-deficient if \( \text{rank}(A) < \min\{m, n\} \)
  – Implies that
    • \( \exists \) some redundancy among its column or row vectors

Characterization of Interpolation Matrix

• Let \( G = [g_1, g_2, \ldots, g_n] \in \mathbb{R}^{n \times n} \) be an interpolation matrix.
  – Rank of \( G \) = dimension of its column space
  – If column vectors are linearly independent,
    • \( \text{Rank}(G) = \text{number of column vectors} \)
  – If column vectors are linearly dependent,
    • \( \text{Rank}(G) < \text{number of column vectors} \)
• Rank deficiency of \( G \)
  – It becomes rank-deficient if \( \text{rank}(G) < n \)
  – It happens
    • When two basis function outputs are collinear to each other, i.e., if two or more input vectors are very close to each other, then the outputs of the basis functions centered at those input vectors would be collinear
Characterization of Interpolation Matrix (cont.)

- In such a situation, we do not need all the column vectors to represent the subspace spanned by \( G \).
- Any one of those collinear vectors can be computed from other vectors.

- In summary, if \( G \) is rank-deficient, it implies that
  - the intrinsic dimensionality of \( G \) < number of columns (\( n \)).
  - the subspace spanned by \( G \) can be described by a smaller number (\( m < n \)) of independent column vectors.

Rank Estimation Based on SVD

- The most popular rank estimation technique for dealing with large matrices in practical applications is Singular Value Decomposition (Golub, 1996).
  - If \( G \) is a real \( n \times n \) matrix, then \( \exists \) orthogonal matrices
    \[
    U \in [u_1, u_2, \ldots, u_n] \in \mathbb{R}^{n \times n}, \quad V \in [v_1, v_2, \ldots, v_n] \in \mathbb{R}^{n \times n}, \quad \text{such that}
    \]
    \[
    U^TGV = \text{diag}(s_1, s_2, \ldots, s_n) = S \in \mathbb{R}^{n \times n}
    \]
    where \( s_1 \geq s_2 \geq \ldots \geq s_n \geq 0 \).
  - \( s_i \): \( i \)th singular value
  - \( u_i \): \( i \)th left singular vector
  - \( v_i \): \( i \)th right singular vector
  - If we define \( r \) by \( s_1 \geq \ldots \geq s_r \geq s_{r+1} = \ldots = s_n = 0 \), then
    \[
    \text{rank}(G) = r \quad \text{and} \quad G = \sum_{i=1}^{r} s_i u_i v_i^T
    \]
Rank Estimation Based on SVD (cont.)

- In practice, data tend to be noisy
  - Interpolation matrix $G$ generated from data is also noisy
  - Thus, the computed singular values from $G$ are noisy and real rank of $G$ should be estimated
- It is suggested to use effective rank ($\varepsilon$-rank) of $G$
- Effective rank $r_\varepsilon = \text{rank}(G, \varepsilon)$, for $\varepsilon > 0$ such that
  $$s_1 \geq s_2 \geq \ldots \geq \varepsilon \geq \ldots \geq s_n$$
- How to determine $\varepsilon$?
  - We introduce RC (Representational Capability)

Representational Capability (RC)

- Definition: RC of $G_m$
  - Let $G$ be an interpolation matrix of size $n \times n$, and SVD of $G$ be given as above. If $m \leq n$ and $G_m = \sum_{i=1}^{m} s_i u_i v_i^T$, then RC of $G_m$ is given by:
    $$\text{RC}(G_m) = 1 - \frac{\|G - G_m\|_2}{\|G\|_2}$$
- Properties of RC
  - Corollary 1: Let SVD of $G = \text{diag}(s_1, s_2, \ldots, s_n)$ and $G_m = \sum_{i=1}^{m} s_i u_i v_i^T$
    Then, for $m < n$
    $$\text{RC}(G_m) = 1 - \frac{s_{m+1}}{s_1}$$
  - Corollary 2: Let $r = \text{rank}(G)$ for $G \in \mathbb{R}^{n \times n}$. If $m < r$, $\text{RC}(G_m) < 1$. Otherwise, $\text{RC}(G_m) = 1$
Determination of $m$ based on RC Criterion

- For an interpolation matrix $G \in \mathbb{R}^{n \times n}$, the number of basis functions which provides $100(1 - \delta)\%$ RC of $G$ is given as

$$m = \text{rank} \left( G, s_i \times \frac{\delta}{100} \right)$$

SVD and $m$:
Sine Example
Singular Value Decomposition (SVD)

- SVD of the interpolation matrix produces three matrices, $U$, $S$, and $V$ ($\sigma = 0.4$)

$$U = \begin{bmatrix}
0.3532 & -0.5708 & 0.5814 & -0.4173 & 0.1929 \\
0.4827 & -0.4173 & -0.1270 & 0.5708 & -0.5009 \\
0.5334 & -0.0000 & -0.5401 & -0.0000 & 0.6510 \\
0.4827 & 0.4173 & -0.1270 & -0.5708 & -0.5009 \\
0.3532 & 0.5708 & 0.5814 & 0.4173 & 0.1929
\end{bmatrix}$$

$$S = \begin{bmatrix}
3.0950 & 0 & 0 & 0 & 0 \\
0 & 1.4314 & 0 & 0 & 0 \\
0 & 0 & 0.4012 & 0 & 0 \\
0 & 0 & 0 & 0.6568 & 0 \\
0 & 0 & 0 & 0 & 0.0055
\end{bmatrix}$$

$$V = \begin{bmatrix}
0.3532 & -0.5708 & 0.5814 & -0.4173 & 0.1929 \\
0.4827 & -0.4173 & -0.1270 & 0.5708 & -0.5009 \\
0.5334 & -0.0000 & -0.5401 & 0.0000 & 0.6510 \\
0.4827 & 0.4173 & -0.1270 & -0.5708 & -0.5009 \\
0.3532 & 0.5708 & 0.5814 & 0.4173 & 0.1929
\end{bmatrix}$$

Singular Value Decomposition (SVD) (cont.)

- Effective rank of $G$ is obtained for several $\sigma$ values

<table>
<thead>
<tr>
<th>Width ($\sigma$)</th>
<th>$s_1$</th>
<th>$s_2$</th>
<th>$s_3$</th>
<th>$s_4$</th>
<th>$s_5$</th>
<th>Effective Rank $r_e$ ($\epsilon = 0.01$)</th>
<th>($\epsilon = 0.001$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>0.20</td>
<td>1.85</td>
<td>1.44</td>
<td>0.94</td>
<td>0.52</td>
<td>0.26</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>0.40</td>
<td>3.10</td>
<td>1.43</td>
<td>0.40</td>
<td>0.0067</td>
<td>0.0006</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>0.70</td>
<td>4.05</td>
<td>0.86</td>
<td>0.08</td>
<td>0.0004</td>
<td>0.0001</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1.00</td>
<td>4.47</td>
<td>0.51</td>
<td>0.02</td>
<td>0.0005</td>
<td>0.0000</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
RC of the Matrix $G_m$

$$G_m = \sum_{i=1}^{m} s_i u_i v_i^T.$$  

$$RC(G_m) = 1 - \frac{s_{m+1}}{s_1}.$$  

- Consider $\sigma = 0.4$; then for $m = 1, 2, 3, 4, 5$, the RC is

<table>
<thead>
<tr>
<th>$m$</th>
<th>$RC(G_m) = (1 - s_{m+1}/s_1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.5375</td>
</tr>
<tr>
<td>2</td>
<td>0.8704</td>
</tr>
<tr>
<td>3</td>
<td>0.9784</td>
</tr>
<tr>
<td>4</td>
<td>0.9983</td>
</tr>
<tr>
<td>5</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

---

RC of the Matrix $G_m$ (cont.)

- Determine $m$ for $RC \geq 80\%$ or $\delta \leq 20\%$

$$s_{m+1} \leq s_1 \times \frac{\delta}{100}.$$  

$$s_{m+1} \leq (3.095) \times (0.20),$$  

$$s_{m+1} \leq 0.619.$$  

$$s_3 = s_{2+1} = 0.4012 < 0.618,$$  

so we get $m = 2$.  

RBF Center Selection Algorithm

Center Selection Algorithm

- Given an interpolation matrix and the number of designed basis functions $m$, two questions are
  - Which columns should be chosen as the column vectors of the design matrix?
  - What criteria should be used?
- We use compromise between
  - Residual minimization for better approximation
  - Structural stabilization for better generalization
Center Selection Algorithm (cont.)

1. Compute the SVD of $G$ to obtain matrices $U$, $S$, and $V$.

2. Partition matrix $V$ and apply the QR factorization with column pivoting to $[V_{11} V_{21}^T]$ and obtain a permutation matrix $P$ as follows:

$$V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix}^{n \times m}.$$

$$Q^T[V_{11}^T V_{21}^T]P = [R_{11} R_{12}].$$

Center Selection Algorithm (cont.)

3. Compute $GP$ and obtain the design matrix $\Phi$ by

$$GP = \begin{bmatrix} \Phi & \Phi' \\ \end{bmatrix}^{m \times m}.$$

4. Compute $X^TP = (x_{k1}, x_{k2}, \ldots, x_{km})$ and determine $m$ centers as

$$\mu_1, \mu_2, \ldots, \mu_m \equiv (x_{k1}, x_{k2}, \ldots, x_{km}).$$
Center Selection: Sine Example

SG Center Selection Algorithm

Step 1: Compute the SVD of $G$ and obtain matrices $U$, $S$, and $V$.

Step 2: Partition $V$ as follows: ($\sigma = 0.4$)

$$V = \begin{bmatrix} 0.3532 & -0.5708 & 0.5814 & -0.4173 & 0.1929 \\ 0.4827 & -0.4173 & -0.1270 & 0.5708 & -0.5009 \\ 0.5334 & -0.0000 & -0.5401 & 0.0000 & 0.6510 \\ 0.4827 & 0.4173 & -0.1270 & -0.5708 & -0.5009 \\ 0.3532 & 0.5708 & 0.5814 & 0.4173 & 0.1929 \end{bmatrix} = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix}$$

where $V_{11}$ is $4 \times 4$, $V_{12}$ is $4 \times 1$, $V_{21}$ is $1 \times 4$, and $V_{22}$ is $1 \times 1$.

Then we apply QR factorization with column pivoting to the following matrix.

$$[V_{11}^T V_{21}^T] = \begin{bmatrix} 0.3532 & 0.4827 & 0.5334 & 0.4827 & 0.3532 \\ -0.5708 & -0.4173 & -0.0000 & 0.4173 & 0.5708 \\ 0.5814 & -0.1270 & -0.5401 & -0.1270 & 0.5814 \\ -0.4173 & 0.5708 & 0.0000 & -0.5708 & 0.4173 \end{bmatrix}$$
SG Center Selection Algorithm (cont.)

This results in Q, R, and P.

\[
Q = \begin{bmatrix}
-0.3600 & 0.3742 & 0.4789 & 0.7078 \\
-0.5817 & -0.5597 & 0.4888 & -0.3308 \\
-0.5925 & 0.6159 & -0.2910 & -0.4300 \\
-0.4253 & -0.4092 & -0.6686 & 0.4524
\end{bmatrix}
\]

\[
R = \begin{bmatrix}
-0.9812 & 0.0379 & -0.0985 & -0.0985 & 0.1280 \\
0 & 0.9805 & 0.1024 & 0.1024 & -0.1331 \\
0 & 0 & 0.8538 & -0.3175 & 0.4126 \\
0 & 0 & 0 & 0.7926 & 0.6098
\end{bmatrix}
\]

\[
P = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

SG Center Selection Algorithm (cont.)

Step 3: Compute GP.

\[
GP = \begin{bmatrix}
0.0439 & 1.0000 & 0.1724 & 0.8226 & 0.4578 \\
0.1724 & 0.8226 & 0.4578 & 1.0000 & 0.8226 \\
0.4578 & 0.4578 & 0.8226 & 0.8226 & 1.0000 \\
0.8226 & 0.1724 & 1.0000 & 0.4578 & 0.8226 \\
1.0000 & 0.0439 & 0.8226 & 0.1724 & 0.4578
\end{bmatrix}
\]

This gives the design matrix \( \Phi \) as

\[
\Phi = \begin{bmatrix}
0.0439 & 1.0000 & 0.1724 & 0.8226 \\
0.1724 & 0.8226 & 0.4578 & 1.0000 \\
0.4578 & 0.4578 & 0.8226 & 0.8226 \\
0.8226 & 0.1724 & 1.0000 & 0.4578 \\
1.0000 & 0.0439 & 0.8226 & 0.1724
\end{bmatrix}
\]
SG Center Selection Algorithm (cont.)

Step 4: Compute $X^T P$ and determine $m = 4$ centers as the first four elements in $X^T P$.

$$X^T P = \begin{bmatrix} 0 & 0.25 & 0.50 & 0.75 & 1.00 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1.00 & 0 & 0.75 & 0.25 & 0.50 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

This gives the four centers as $\mu_1 = 1.00, \mu_2 = 0, \mu_3 = 0.75$ and $\mu_4 = 0.25$.

Structural Stabilization

- Structural stabilization criterion is used for better generalization property of the designed RBF model
- Five possible combinations and potential design matrices are $\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5$

<table>
<thead>
<tr>
<th>Design Matrix</th>
<th>$\Phi_1$</th>
<th>$\Phi_2$</th>
<th>$\Phi_3$</th>
<th>$\Phi_4$</th>
<th>$\Phi_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\phi_1, \phi_2, \phi_3, \phi_4)$</td>
<td>$(\phi_1, \phi_2, \phi_3, \phi_4)$</td>
<td>$(\phi_1, \phi_2, \phi_3, \phi_4)$</td>
<td>$(\phi_1, \phi_2, \phi_3, \phi_4)$</td>
<td>$(\phi_1, \phi_2, \phi_3, \phi_4)$</td>
<td></td>
</tr>
<tr>
<td>Weights</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 inputs</td>
<td>30 inputs</td>
<td>5 inputs</td>
<td>30 inputs</td>
<td>5 inputs</td>
<td>30 inputs</td>
</tr>
<tr>
<td>-4.8236</td>
<td>-4.9756</td>
<td>-3.4517</td>
<td>-3.6433</td>
<td>-2.0789</td>
<td>-2.1835</td>
</tr>
</tbody>
</table>

Euclidean distance between weight vectors:

| 0.2479 | 0.2665 | 0.2081 | 0.2724 | 0.2363 |
Structural Stabilization

- Simulate additional 30 (x, y) data
- Compute 5 design matrices for \( \Phi_I, \Phi_{II}, \Phi_{III}, \Phi_{IV}, \Phi_{V} \)
- Compute weights and compare
- Use euclidean distance

---

Residual Size

<table>
<thead>
<tr>
<th>Design Matrix</th>
<th>( \Phi_I )</th>
<th>( \Phi_{II} )</th>
<th>( \Phi_{III} )</th>
<th>( \Phi_{IV} )</th>
<th>( \Phi_{V} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residual size 5 data points</td>
<td>0.0067</td>
<td>0.0016</td>
<td>0.0000</td>
<td>0.0018</td>
<td>0.0035</td>
</tr>
<tr>
<td>Residual size 30 data points</td>
<td>0.1402</td>
<td>0.1025</td>
<td>0.0086</td>
<td>0.0986</td>
<td>0.1084</td>
</tr>
</tbody>
</table>
Benchmark Data Classification Modeling

Benchmark Classification Problems

• Benchmark data for classifier learning are important for evaluating or comparing algorithms for learning from examples
• Consider two sets from Proben 1 database (Prechelt, 1994) in the UCI repository of machine learning databases:
  – Diabetes
  – Soybean
Diabetes Data: 2 Classes

- Determine if diabetes of Pima Indians is positive or negative based on description of personal data such as age, number of times pregnant, etc.
- 8 inputs, 2 outputs, 768 examples and no missing values in this data set
- The 768 example data is divided into 384 examples for training, 192 for validation and 192 for test
- Three permutations of data to generate three data sets: diabetes 1, 2, 3
- Error measure

\[
\text{Classification error} = \frac{\# \text{incorrectly classified patients}}{\# \text{total patients}}
\]

Description of Diabetes Input and Output Data

<table>
<thead>
<tr>
<th>Attribute No.</th>
<th>No. of Attributes</th>
<th>Attribute Meaning</th>
<th>Values and Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Number of times pregnant</td>
<td>0.17 → 0.1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Plasma glucose concentration after 2 hours in an oral glucose tolerance test</td>
<td>0.199 → 0.1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>Diastolic blood pressure (mm Hg)</td>
<td>0.122 → 0.1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>Triceps skin fold thickness (mm)</td>
<td>0.99 → 0.1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2-hour serum insulin (mu U/ml)</td>
<td>0.846 → 0.1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>Body mass index (weight in kg/(height in m)^2)</td>
<td>0.671 → 0.1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>Diabetes pedigree function</td>
<td>0.078..2.42 → 0.1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>Age (years)</td>
<td>21.81 → 0.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output (1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
### RBF Models for Diabetes 1

#### $\delta = 0.01$

<table>
<thead>
<tr>
<th>Model</th>
<th>$m$</th>
<th>$\sigma$</th>
<th>Classification Error (CE), %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Training</td>
</tr>
<tr>
<td>A</td>
<td>12</td>
<td>0.6</td>
<td>20.32</td>
</tr>
<tr>
<td>B</td>
<td>9</td>
<td>0.7</td>
<td>21.88</td>
</tr>
<tr>
<td>C</td>
<td>9</td>
<td>0.8</td>
<td><strong>22.66</strong></td>
</tr>
<tr>
<td>D</td>
<td>8</td>
<td>0.9</td>
<td>22.92</td>
</tr>
<tr>
<td>E</td>
<td>8</td>
<td>1.0</td>
<td>23.44</td>
</tr>
<tr>
<td>F</td>
<td>7</td>
<td>1.1</td>
<td>26.04</td>
</tr>
<tr>
<td>G</td>
<td>6</td>
<td>1.2</td>
<td>25.78</td>
</tr>
<tr>
<td>H</td>
<td>5</td>
<td>1.3</td>
<td>25.26</td>
</tr>
</tbody>
</table>

#### Plots of Training and Validation Errors for Diabetes 1 ($\delta = 0.01$)

- **Figure (a)**: 3D plot showing the classification error (CE) percentage as a function of $m$ and $\sigma$.
- **Figure (b)**: 2D plot showing the classification error (CE) percentage as a function of $m$.

- **Legend**:
  - **Red**: Training
  - **Blue**: Validation
Observations

- As model $\sigma$ decreases (bottom to top)
  - Model complexity ($m$) increases
  - Training CE decreases
  - Validation CE decreases and then increases
  - Test CE decreases and then increases
- CE behavior as theoretically expected
- Choose model B with minimum validation CE
- Test CE is 23.44%
- Different models for other $\delta$ values
- Best model for each data set is given next

RBF Classification Models for Diabetes 1, Diabetes 2 and Diabetes 3

<table>
<thead>
<tr>
<th>Problem</th>
<th>$\delta$</th>
<th>$m$</th>
<th>$\sigma$</th>
<th>Classification Error (CE), %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Training</td>
</tr>
<tr>
<td>diabetes1</td>
<td>0.001</td>
<td>10</td>
<td>1.2</td>
<td>22.66</td>
</tr>
<tr>
<td>diabetes2</td>
<td>0.005</td>
<td>25</td>
<td>0.5</td>
<td>18.23</td>
</tr>
<tr>
<td>diabetes3</td>
<td>0.001</td>
<td>15</td>
<td>1.0</td>
<td>18.49</td>
</tr>
</tbody>
</table>

Diabetes 1, 2, and 3
- Test error varies considerably
- Average about 24.7%
Comparison with Prechelt Results [1994]

- Linear Network (LN)
  - No hidden nodes, direct input-output connection
  - The error values are based on 10 runs
- Multilayer Network (MN)
  - Sigmoidal hidden nodes
  - 12 different topologies
  - “Best” test error reported

Diabetes Test CE for LN, MN and SG-RBF

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>Test CE %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mean</td>
</tr>
<tr>
<td>diabetes1</td>
<td>LN</td>
<td>25.83</td>
</tr>
<tr>
<td></td>
<td>MN</td>
<td>24.57</td>
</tr>
<tr>
<td></td>
<td>SG (model C)</td>
<td>23.96</td>
</tr>
<tr>
<td>diabetes2</td>
<td>LN</td>
<td>24.69</td>
</tr>
<tr>
<td></td>
<td>MN</td>
<td>25.91</td>
</tr>
<tr>
<td></td>
<td>SG (model C)</td>
<td>25.52</td>
</tr>
<tr>
<td>diabetes3</td>
<td>LN</td>
<td>22.92</td>
</tr>
<tr>
<td></td>
<td>MN</td>
<td>23.06</td>
</tr>
<tr>
<td></td>
<td>SG (model B)</td>
<td>23.01</td>
</tr>
<tr>
<td>Average</td>
<td>LN/MN/SG</td>
<td>24.48/24.46/24.20</td>
</tr>
</tbody>
</table>

Compared to Prechelt, almost as good as best reported
RBF-SG results are fixed; no randomness
Soybean Disease Classification: 19 Classes

- Inputs (35): Description of bean, plant, plant history, etc
- Output: One of 19 disease types
- 683 examples: 342 training, 171 validation, 170 test
- Three permutations to generate Soybean 1, 2, 3
- $\sigma$: 1.1(0.2)2.5
- $\delta$: 0.001, 0.005, 0.01

Description of One Soybean Data Point

```
Attribute number          Data value
  1  2  3    4   5  6  7  8  9  10  11 12  13  14  15  16  17  18
0  1  0  0  0  0  0  5  0  1  0  0  0  3  3  3  3  3  3

1  2  3  4  5  6  7  8  9  10  11 12  13  14  15  16  17  18
1  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2

19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
abnorm yes below-soil dk-brown-blk absent

000000000000000000001000

88 phytophthora-rot
```
RBF Models for Soybean1 ($\delta = 0.01$)

The 683 example data set is divided into 342 examples for training set, 171 for validation set and 170 for test set.

<table>
<thead>
<tr>
<th>model</th>
<th>$m$</th>
<th>$\sigma$</th>
<th>CE %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Training</td>
<td>Val.</td>
</tr>
<tr>
<td>A</td>
<td>249</td>
<td>1.1</td>
<td>0.88</td>
</tr>
<tr>
<td>B</td>
<td>202</td>
<td>1.3</td>
<td>2.27</td>
</tr>
<tr>
<td>C</td>
<td>150</td>
<td>1.5</td>
<td>2.05</td>
</tr>
<tr>
<td>D</td>
<td>107</td>
<td>1.7</td>
<td>2.92</td>
</tr>
<tr>
<td>E</td>
<td>73</td>
<td>1.8</td>
<td>4.09</td>
</tr>
<tr>
<td>F</td>
<td>56</td>
<td>2.1</td>
<td>4.68</td>
</tr>
<tr>
<td>G</td>
<td>46</td>
<td>2.3</td>
<td>4.97</td>
</tr>
<tr>
<td>H</td>
<td>39</td>
<td>2.5</td>
<td>7.60</td>
</tr>
</tbody>
</table>

The minimum validation CE equals 4.68% for two models C and D. Since, we generally prefer a simpler model, i.e., a model with smaller $m$, we choose model $D$.

Plots of CE Training and Validation Errors for Soybean1 ($\delta = 0.01$)

Training error decreases from models $H$ to $A$ as $m$ increases. The validation error, however, decreases up to a point and then begins to increase.
### Soybean CE for LN, MN and SG-RBF

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>Test CE %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>mean</td>
</tr>
<tr>
<td>soybean1</td>
<td>LN</td>
<td>9.47</td>
</tr>
<tr>
<td></td>
<td>MN</td>
<td>9.06</td>
</tr>
<tr>
<td></td>
<td>SG (model F)</td>
<td>7.65</td>
</tr>
<tr>
<td>soybean2</td>
<td>LN</td>
<td>4.24</td>
</tr>
<tr>
<td></td>
<td>MN</td>
<td>5.84</td>
</tr>
<tr>
<td></td>
<td>SG (model G)</td>
<td>4.71</td>
</tr>
<tr>
<td>soybean3</td>
<td>LN</td>
<td>7.00</td>
</tr>
<tr>
<td></td>
<td>MN</td>
<td>7.27</td>
</tr>
<tr>
<td></td>
<td>SG (model E)</td>
<td>4.12</td>
</tr>
<tr>
<td>Average</td>
<td>LN/MN/SG</td>
<td>6.90/7.39/5.49</td>
</tr>
</tbody>
</table>

The SG-RBF classifiers have *smaller errors* for soybean1 and soybean3. Overall better average error and no randomness.

---

**Data Mining and Knowledge Discovery**
Knowledge Discovery: Software Engineering

- KDD is the nontrivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data
- KDD includes data mining as a critical phase of the KDD process; activity of extracting patterns by employing a specific algorithm
- Currently KDD is used for, e.g., text mining, sky surveys, customer relations managements, etc
- We discuss knowledge discovery about criticality evaluation of software modules

KDD Process

- KDD refers to all activities from data collection to use of the discovered knowledge
- Typical steps in KDD
  - Learning the application domain: prior knowledge; study objectives
  - Creating dataset: identification of relevant variables or factors
  - Data cleaning and preprocessing: removal of wrong data and outliers, consistency checking, methods for dealing with missing data fields, and preprocessing
  - Data reduction and projection: finding useful features for data representation, data reduction and appropriate transformations
  - Choosing data mining function: decisions about modeling goal such as classification or prediction
KDD Process (cont.)

- Choosing data mining algorithms: algorithm selection for the task chosen in the previous step
- Data mining: actual activity of searching for patterns of interest such as classification rules, regression or neural network modeling as well as validation and accuracy assessment
- Interpretation and use of discovered knowledge: presentation of discovered knowledge; and taking specific steps consistent with the goals of knowledge discovery

KDD Goals: SE

- Software development is very much like an industrial production process consisting of several overlapping activities, formalized as life-cycle models
- Aim of collecting software data is to perform knowledge discovery activities to seek useful information
- Some typical questions of interest to software engineers and managers are
  - What features (metrics) are indicators of high quality systems
  - What metrics should be tracked to assess system readiness
  - What patterns of metrics indicate potentially high defect modules
  - What metrics can be related to software maturity during development
- Hundreds of such questions are of interest in SE
List of Metrics from NASA Metrics Database

<table>
<thead>
<tr>
<th>( x_7 )</th>
<th>Faults</th>
<th># of faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_9 )</td>
<td>Function Calls from This Component</td>
<td>Design metrics ( x_9, x_{10}, x_{18} )</td>
</tr>
<tr>
<td>( x_{10} )</td>
<td>Function Calls to This Component</td>
<td>Coding metrics ( x_{13}, x_{14}, x_{15} )</td>
</tr>
<tr>
<td>( x_{11} )</td>
<td>Input/Output Statements</td>
<td>Module level product metrics</td>
</tr>
<tr>
<td>( x_{12} )</td>
<td>Total Statements</td>
<td></td>
</tr>
<tr>
<td>( x_{13} )</td>
<td>Size of Component in Number of Program Lines</td>
<td></td>
</tr>
<tr>
<td>( x_{14} )</td>
<td>Number of Comment Lines</td>
<td></td>
</tr>
<tr>
<td>( x_{15} )</td>
<td>Number of Decisions</td>
<td></td>
</tr>
<tr>
<td>( x_{16} )</td>
<td>Number of Assignment Statements</td>
<td></td>
</tr>
<tr>
<td>( x_{17} )</td>
<td>Number of Format Statements</td>
<td></td>
</tr>
<tr>
<td>( x_{18} )</td>
<td>Number of Input/Output Parameters</td>
<td></td>
</tr>
<tr>
<td>( x_{19} )</td>
<td>Number of Unique Operators</td>
<td></td>
</tr>
<tr>
<td>( x_{20} )</td>
<td>Number of Unique Operands</td>
<td></td>
</tr>
<tr>
<td>( x_{21} )</td>
<td>Total Number of Operators</td>
<td></td>
</tr>
<tr>
<td>( x_{22} )</td>
<td>Total Number of Operands</td>
<td></td>
</tr>
</tbody>
</table>

KDD Process for Software Modules

- **Application domain:** Early identification of critical modules which are subjected to additional testing, etc. to improve system quality
- **Database:** NASA metrics DB; 14 metrics; many projects; select 796 modules
- **Transformation:** Normalize metrics to (0, 1); class is +1 if number of faults exceeds five; -1 otherwise; ten permutation with (398 training; 199 validation; 199 test)
- **Function:** RBF classifiers
- **Data Mining:** Classification modeling for design; coding; fourteen metrics
- **Interpretation:** Compare accuracy; determine relative adequacy of different sets of metrics
### Classification: Design Metrics

<table>
<thead>
<tr>
<th>Permutation</th>
<th>m</th>
<th>Training</th>
<th>Validation</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>27.1</td>
<td>29.2</td>
<td>21.6</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>25.2</td>
<td>23.6</td>
<td>24.6</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>25.6</td>
<td>21.1</td>
<td>26.1</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>24.9</td>
<td>26.6</td>
<td>22.6</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>21.6</td>
<td>27.6</td>
<td>28.1</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>24.1</td>
<td>25.1</td>
<td>24.6</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>22.6</td>
<td>26.6</td>
<td>24.6</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>24.4</td>
<td>28.6</td>
<td>24.1</td>
</tr>
<tr>
<td>9</td>
<td>7</td>
<td>24.4</td>
<td>28.6</td>
<td>24.1</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>23.1</td>
<td>24.6</td>
<td>27.1</td>
</tr>
</tbody>
</table>

### Design Metrics (cont.)

[Bar chart showing classification error across different permutations.]
Test Error Results

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Average</th>
<th>SD</th>
<th>90 %</th>
<th>95 %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Metrics</td>
<td>24.95</td>
<td>1.97</td>
<td>{23.81, 26.05}</td>
<td>{23.60, 26.40}</td>
</tr>
<tr>
<td>Coding Metrics</td>
<td>23.00</td>
<td>3.63</td>
<td>{20.89, 25.11}</td>
<td>{21.40, 25.80}</td>
</tr>
<tr>
<td>Fourteen Metrics</td>
<td>24.35</td>
<td>2.54</td>
<td>{22.89, 25.81}</td>
<td>{22.55, 26.15}</td>
</tr>
</tbody>
</table>

Confidence bound:

$$\left\{\text{Avg } TE \pm t_{9,\alpha/2} \left(\text{SD of } TE\right)/\sqrt{10}\right\}$$

Summary of Data Mining Results

- Predictive error on test data about 23%
- Very good for software engineering data where low accuracy is common; errors can be as high 60% or more
- Classification errors are similar for design metrics, coding metrics, all (14) metrics
- However, design metrics are available in early development phases and are preferred for developing classification models
- Knowledge discovered
  - good classification accuracy
  - can use design metrics for criticality evaluation of software modules
- What next
  - KDD on other projects using RBF
Empirical Data Modeling in Software Engineering Project Effort Prediction

Software Effort Modeling

- Accurate estimation of soft project effort is one of the most important empirical modeling tasks in software engineering as indicated by the large number of models developed over the past twenty years.
- Most of the popularly used models employ a regression type equation relating effort and size, which is then calibrated for local environment.
- We use NASA data to develop RBF models for effort (Y) based on Developed Lines (DL) and Methodology (ME).
- DL is KLOC; ME is composite score; Y is Man-months.
RBF Based on DL

• Simple problem; for illustration
• Our goal is to seek a parsimonious model which provides a good fit and exhibits good generalization capability
• Modeling steps
  – Select $d = 1\%, 2\%, \text{ and } 0.1\% \text{ and a range of } \sigma \text{ values}$
  – For each $\sigma$, determine the value of $m$ which satisfies $d$
  – Determine parameters $\mu$ and $w$ according to the SG algorithm
  – Compute training error for the data on 18 projects
  – Use LOOCV technique to compute generalization error
  – Select the model which has minimum generalization error and small training error
  – Repeat above for each $d$ and select the most appropriate model
Two Error Measures

- MMRE = \( \frac{1}{n} \sum_{i=1}^{n} \frac{|Y_i - \hat{Y}_i|}{Y_i} \)

- PRED(25) = Percentage of predictions falling within 25% of the actual known values

RBF Designs and Performance Measure for (DL-Y) Models (\( \delta = 1\% \))

<table>
<thead>
<tr>
<th>Model</th>
<th>( \sigma )</th>
<th>( m )</th>
<th>MMRE Training</th>
<th>MMRE Generalization</th>
<th>PRED(25) Training</th>
<th>PRED(25) Generalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.25</td>
<td>5</td>
<td>0.1932</td>
<td>0.2556</td>
<td>72.22</td>
<td>61.11</td>
</tr>
<tr>
<td>B</td>
<td>0.30</td>
<td>4</td>
<td>0.2662</td>
<td>0.3345</td>
<td>55.56</td>
<td>44.44</td>
</tr>
<tr>
<td>C</td>
<td>0.35</td>
<td>4</td>
<td>0.2680</td>
<td>0.3372</td>
<td>55.56</td>
<td>44.44</td>
</tr>
<tr>
<td>D</td>
<td>0.40</td>
<td>3</td>
<td>0.1647</td>
<td>0.1978</td>
<td>83.33</td>
<td>66.67</td>
</tr>
<tr>
<td>E</td>
<td>0.45</td>
<td>3</td>
<td>0.1613</td>
<td>0.1930</td>
<td>83.33</td>
<td>72.22</td>
</tr>
<tr>
<td>F</td>
<td>0.50</td>
<td>3</td>
<td>0.1592</td>
<td>0.1899</td>
<td>88.89</td>
<td>72.22</td>
</tr>
<tr>
<td>G</td>
<td>0.55</td>
<td>3</td>
<td>0.1579</td>
<td>0.1870</td>
<td>88.89</td>
<td>72.22</td>
</tr>
<tr>
<td>H</td>
<td>0.60</td>
<td>3</td>
<td>0.1678</td>
<td>0.1982</td>
<td>88.89</td>
<td>77.78</td>
</tr>
<tr>
<td>I</td>
<td>0.65</td>
<td>3</td>
<td>0.1774</td>
<td>0.2090</td>
<td>88.89</td>
<td>72.22</td>
</tr>
<tr>
<td>J</td>
<td>0.70</td>
<td>3</td>
<td>0.1860</td>
<td>0.2187</td>
<td>88.89</td>
<td>66.67</td>
</tr>
<tr>
<td>K</td>
<td>0.75</td>
<td>2</td>
<td>0.1863</td>
<td>0.2093</td>
<td>77.78</td>
<td>77.78</td>
</tr>
</tbody>
</table>
A Graphical Depiction of MMRE Measures for Candidate Models

RBF Models for (DL-Y) Data

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>$\sigma$</th>
<th>m</th>
<th>MMRE Training</th>
<th>MMRE Generalization</th>
<th>PRED(25) Training</th>
<th>PRED(25) Generalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>0.55</td>
<td>3</td>
<td>0.1579</td>
<td>0.1870</td>
<td>88.89</td>
<td>72.22</td>
</tr>
<tr>
<td>2%</td>
<td>0.55</td>
<td>3</td>
<td>0.1579</td>
<td>0.1870</td>
<td>88.89</td>
<td>72.22</td>
</tr>
<tr>
<td>0.1%</td>
<td>0.25</td>
<td>6</td>
<td>0.1450</td>
<td>0.1881</td>
<td>77.78</td>
<td>72.22</td>
</tr>
</tbody>
</table>
Estimation Model

\[
\hat{Y}_N = -0.3652 \exp \left( -\frac{1}{2(0.55)^2} \cdot (DL_N - 0.0)^2 \right) + 0.3709 \exp \left( -\frac{1}{2(0.55)^2} \cdot (DL_N - 0.5309)^2 \right) + 0.6907 \exp \left( -\frac{1}{2(0.55)^2} \cdot (DL_N - 1.0)^2 \right)
\]

where

\[
DL_N = \frac{DL - 2.1}{100.8 - 2.1} \quad \text{and} \quad Y_N = \frac{DL - 5.0}{138.3 - 5.0}
\]

Plot of the Fitted RBF Estimation Model and Actual Effort as a Function of DL
Models for DL and ME

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>m</th>
<th>$\sigma$</th>
<th>MMRE Training</th>
<th>MMRE Generalization</th>
<th>PRED(25) Training</th>
<th>PRED(25) Generalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>4</td>
<td>0.85</td>
<td>.1448</td>
<td>0.2584</td>
<td>88.89</td>
<td>72.22</td>
</tr>
<tr>
<td>2%</td>
<td>4</td>
<td>0.75</td>
<td>.1470</td>
<td>0.2474</td>
<td>88.89</td>
<td>72.22</td>
</tr>
<tr>
<td>0.1%</td>
<td>7</td>
<td>0.75</td>
<td>.0870</td>
<td>.1907</td>
<td>77.78</td>
<td>72.22</td>
</tr>
</tbody>
</table>

Plot of the Fitted RBF Estimation Model and Actual Effort as a Function DL and ME
Plot of the Fitted RBF Estimation Model and Actual Effort as a Function DL and ME (cont.)

KDD: Microarray Data Analysis
MICROARRAY DATA AND ANALYSIS
GOALS

Data*
• A matrix of gene expression values $X_{n\times d}$
• Cancer class vector $y=1$ (ALL), $y=0$ (AML), $Y_{n\times d}$
• Training set $n=38$, Test set $n=34$
• Two data sets with number of genes $d=7129$ and $d=50$

MICROARRAY DATA AND ANALYSIS

GOALS (cont.)

Classification Goal
• Develop classification models to predict leukemia class (ALL or AML) based on training set
• Use Radial Basis Function (RBF) model and employ recently developed Shin-Goel (SG) design algorithm

Model selection
• Choose the model that achieves the best balance between fitting and model complexity
• Use tradeoffs between classification errors on training and test sets as model selection criterion

BACKGROUND

• Advances in microarray technology are producing very large datasets that require proper analytical techniques to understand the complexities of gene functions. To address this issue, presentations at CAMDA2000 conference* discussed analyses of the same data sets using different approaches
• Golub et al’s dataset (one of two at CAMDA) involves classification into acute lymphoblastic (ALL) or acute myeloid (AML) leukemia based on 7129 attributes that correspond to human gene expression levels

* Critical Assessment of Microarray Data; for papers see Lin, S. M. and Johnson, K. E (Editors), Methods of Microarray Data Analysis, Kluwer, 2002
BACKGROUND (cont.)

- In this study, we formulate the classification problem as a two step process. First we construct a radial basis function model using a recent algorithm of Shin and Goel**. Then model performance is evaluated on test set classification.

---

Shin, M, Goel, A. L. Radial Basis Function Model Development and Analysis Using the SG Algorithm (Revised), Technical Report, Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY, 2002

CLASSIFICATION MODELING

- Data of Golub et al* consists of 38 training samples (27 ALL, 11 AML) and 34 test samples (20 ALL, 14 AML). Each sample corresponds to 7129 genes. They also selected 50 most informative genes and used both sets for classification studies.
- We develop several RBF classification models using the SG algorithm and study their performance on training and test data sets.
- Classifier with best compromise between training and test errors is selected.

---

CLASSIFICATION MODELING (cont.)

Summary of Results
• For specified RC and $\sigma$, SG algorithm first computes minimum $m$ and then the centers and weights
• We use RC = 99% and 99.5%
• 7129 gene set: $\sigma = 20(2)32$,
  50 gene set: $\sigma = 2(0.4)4$
• Table 1 lists the “Best” RBF models

<table>
<thead>
<tr>
<th>Data Set</th>
<th>RC</th>
<th>m</th>
<th>$\sigma$</th>
<th>Correct Classification</th>
<th>Classification error %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>training</td>
<td>test</td>
</tr>
<tr>
<td>7129 genes</td>
<td>99.0%</td>
<td>29</td>
<td>26</td>
<td>38</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>99.5%</td>
<td>35</td>
<td>30</td>
<td>38</td>
<td>29</td>
</tr>
<tr>
<td>50 genes</td>
<td>99.0%</td>
<td>6</td>
<td>3.2</td>
<td>38</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>99.5%</td>
<td>13</td>
<td>3.2</td>
<td>38</td>
<td>33</td>
</tr>
</tbody>
</table>
SENSITIVITY ANALYSES
(7129 Gene Data)

RC=99%; \( \sigma=20(2)32 \)
- SG algorithm computes minimum m (no. of basis functions) that satisfies RC
- Table 2 and Figure 4, show models and their performance on training and test sets
- “Best” model is D: \( m=29, \sigma=26 \)
- Correctly classifies 38/38 training samples; only 29/34 test samples
- Models A and B represent underfitting, F and G overfitting; Figure 1 shows underfit-overfit realization

Classification results
(7129 Genes, RC=99%)
(38 training, 34 test samples)

<table>
<thead>
<tr>
<th>Model</th>
<th>( \sigma )</th>
<th>m</th>
<th>Correct classification</th>
<th>Classification error %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>training</td>
<td>test</td>
</tr>
<tr>
<td>A</td>
<td>32</td>
<td>12</td>
<td>36</td>
<td>25</td>
</tr>
<tr>
<td>B</td>
<td>30</td>
<td>15</td>
<td>37</td>
<td>27</td>
</tr>
<tr>
<td>C</td>
<td>28</td>
<td>21</td>
<td>37</td>
<td>28</td>
</tr>
<tr>
<td>D</td>
<td>26</td>
<td>29</td>
<td>38</td>
<td>29</td>
</tr>
<tr>
<td>E</td>
<td>24</td>
<td>34</td>
<td>38</td>
<td>29</td>
</tr>
<tr>
<td>F</td>
<td>22</td>
<td>38</td>
<td>38</td>
<td>28</td>
</tr>
<tr>
<td>G</td>
<td>20</td>
<td>38</td>
<td>38</td>
<td>28</td>
</tr>
</tbody>
</table>
Classification Errors
(7129 genes; RC=99%)

SENSITIVITY ANALYSES (cont.)
(50 Gene Data)

- Table 3 and Figure 5 show several RBF models and their performance on 50 gene training and test data
- Model C (m=6, \( \sigma = 3.2 \)) seems to be the best one with 38/38 correct classification on training data and 33/34 on test data
- Model A represents underfit and models D, E and F seem to be unnecessarily complex, with no gain in classification accuracy
Classification Results  
(50 Genes RC=99%)  
(38 Training, 34 Test Sets)

<table>
<thead>
<tr>
<th>Model</th>
<th>$\sigma$</th>
<th>Basis Functions (m)</th>
<th>Correct classification</th>
<th>Classification error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>training</td>
<td>test</td>
</tr>
<tr>
<td>A</td>
<td>4.0</td>
<td>4</td>
<td>37</td>
<td>31</td>
</tr>
<tr>
<td>B</td>
<td>3.6</td>
<td>5</td>
<td>37</td>
<td>32</td>
</tr>
<tr>
<td>C</td>
<td>3.2</td>
<td>6</td>
<td>38</td>
<td>33</td>
</tr>
<tr>
<td>D</td>
<td>2.8</td>
<td>9</td>
<td>38</td>
<td>33</td>
</tr>
<tr>
<td>E</td>
<td>2.4</td>
<td>13</td>
<td>38</td>
<td>33</td>
</tr>
<tr>
<td>F</td>
<td>2.0</td>
<td>18</td>
<td>38</td>
<td>33</td>
</tr>
</tbody>
</table>

Classification Errors (50 genes; RC = 99%)
REMARKS

- This study used Gaussian RBF model and the SG algorithm for the cancer classification problem of Golub et. al. Here we present some remarks about our methodology and future plans.
- RBF models have been used for classification in a broad range of applications, from astronomy to medical diagnosis and from stock market to signal processing.
- Current algorithms, however, tend to produce inconsistent results due to their ad-hoc nature.
- The SG algorithm produces consistent results, has strong mathematical underpinnings, primarily involves matrix computations and no search or optimization. It can be almost totally automated.

Summary

In this tutorial, we discussed the following issues:
- Problems of classification and prediction; and the modeling considerations involved.
- Structure of the RBF model and some design approaches.
- Detailed coverage of the new (Shin-Goel) SG algebraic algorithm with illustrative examples.
- Classification modeling using the SG algorithm for two benchmark data sets.
- KDD and DM issues using RBF/SG in software engineering and cancer class prediction.
Selected References

• C. M. Bishop, Neural Network for Pattern Recognition, Oxford, 1995.

Selected References (cont.)