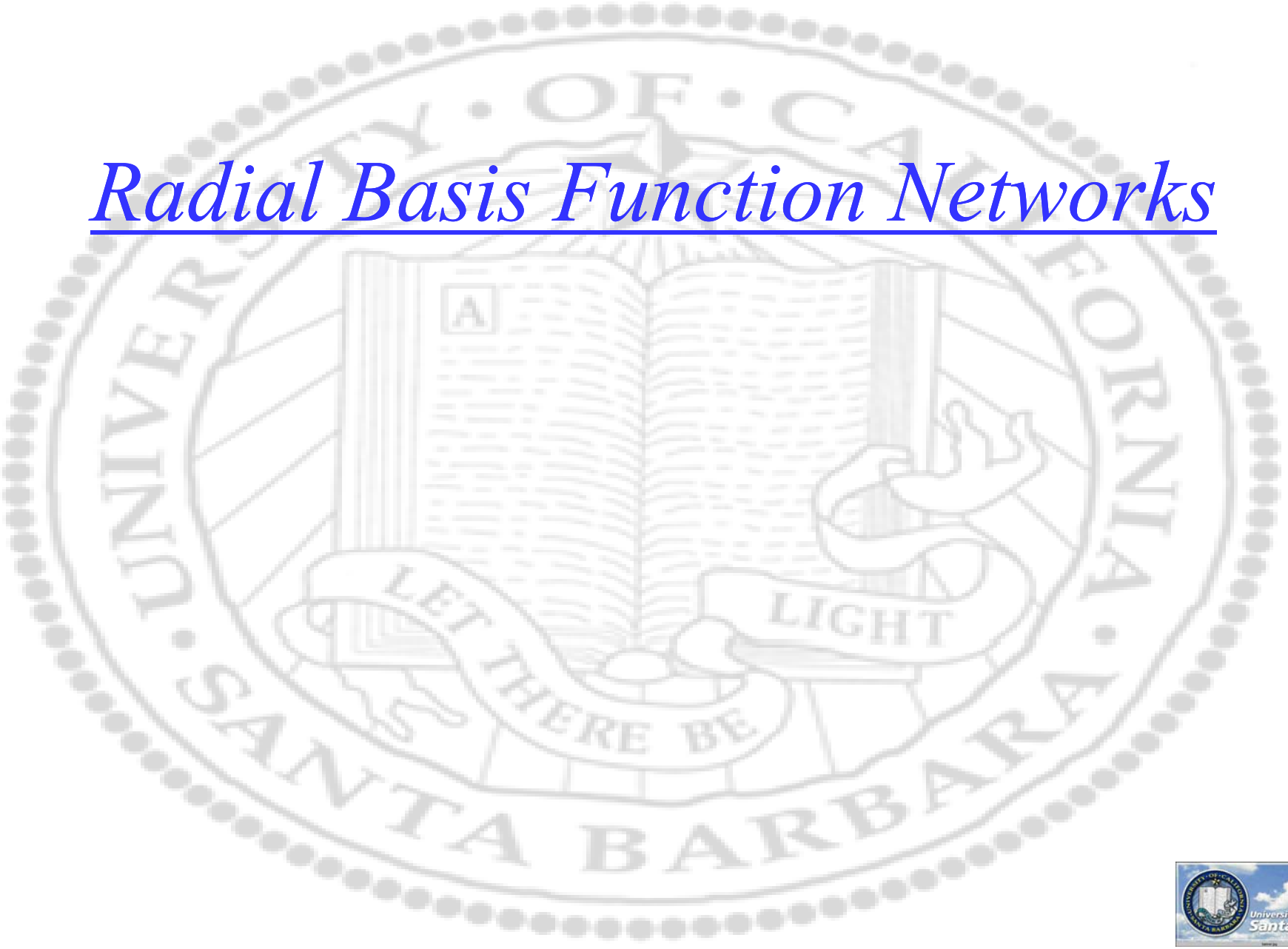
Radial Basis Function Networks



Radial Basis Function Networks

- A special types of ANN that have three layers
  - Input layer
  - Hidden layer
  - Output layer
- Mapping from input to hidden layer is *nonlinear*
- Mapping from hidden to output layer is *linear*

Comparison

Multi-layer perceptron

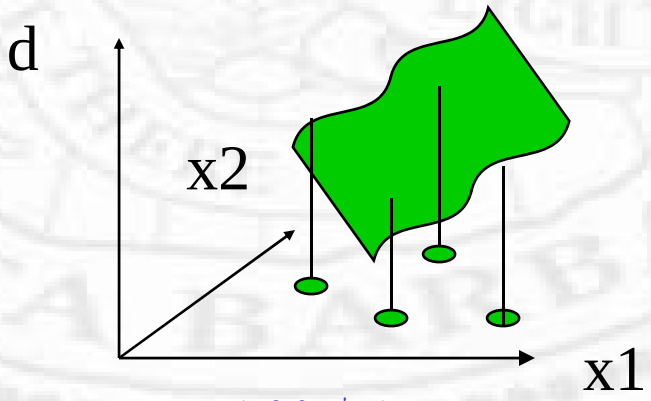
- Multiple hidden layers
- Nonlinear mapping
- W: inner product
- Global mapping
- Warp classifiers
- *Stochastic approximation*

RBF Networks

- Single hidden layer
- Nonlinear + linear
- W: distance
- Local mapping
- Warp data
- *Curve fitting*

Another View: Curve Fitting

- We try to estimate a mapping from patterns into classes *f(patterns)->classes*, *f(**X**)->d*
- Patterns are represented as feature vector **X**
- Classes are decisions *d*
- Training samples: $f(\mathbf{X}_i) \rightarrow d_i$, $i=1,\ldots, n$
- Interpolation of the *f* based on samples

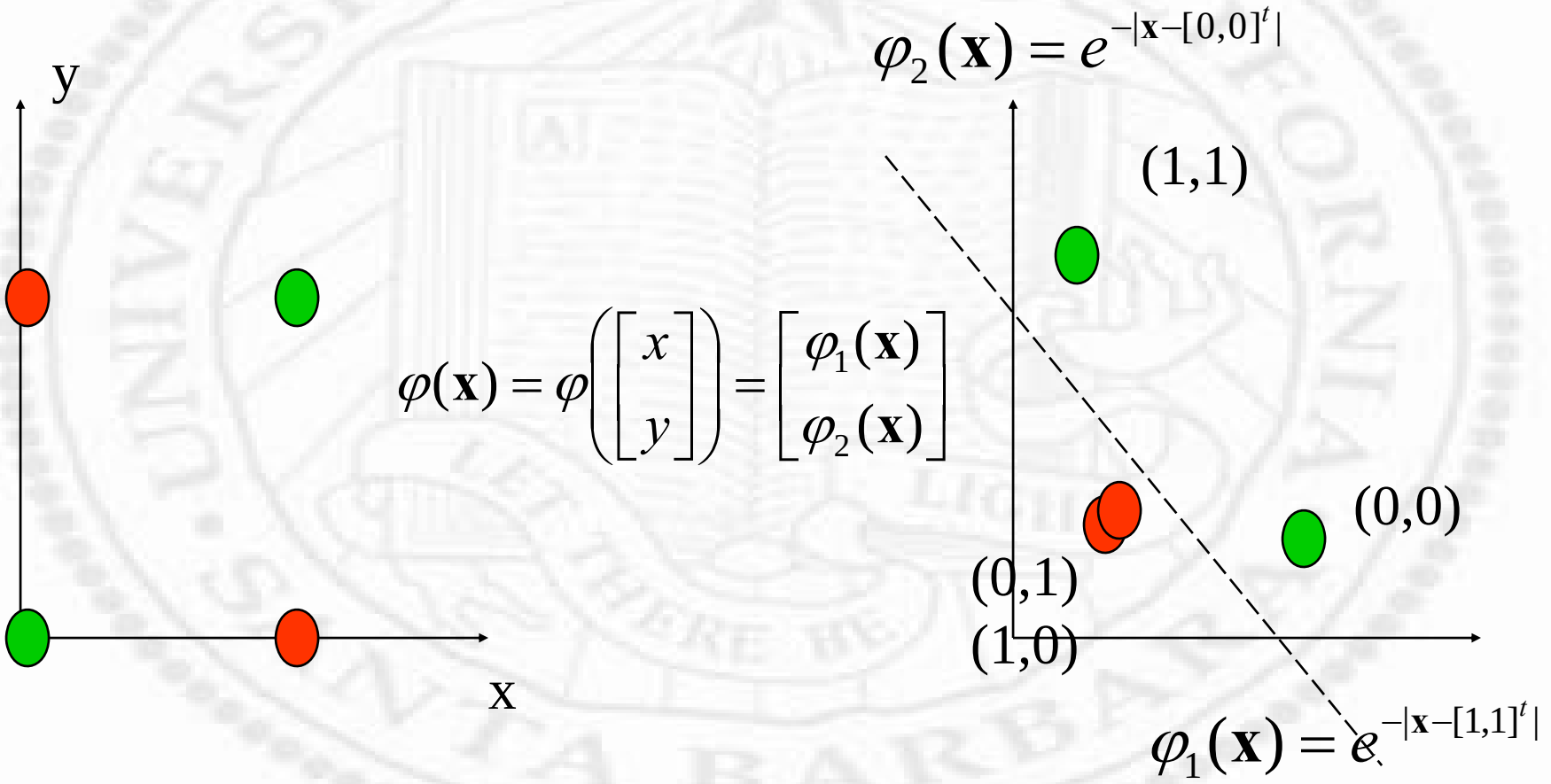


Yet Another View: Warping Data

- If the problem is not linearly separable, MLP will use multiple neurons to define complicated decision boundaries (warp classifiers)
- Another alternative is to warp data into higher dimensional space that they are much more likely to be linearly separable (single perceptron will do)
- This is very similar to the idea of Support Vector Machine

Example

- XOR
- Warpped XOR



More Example

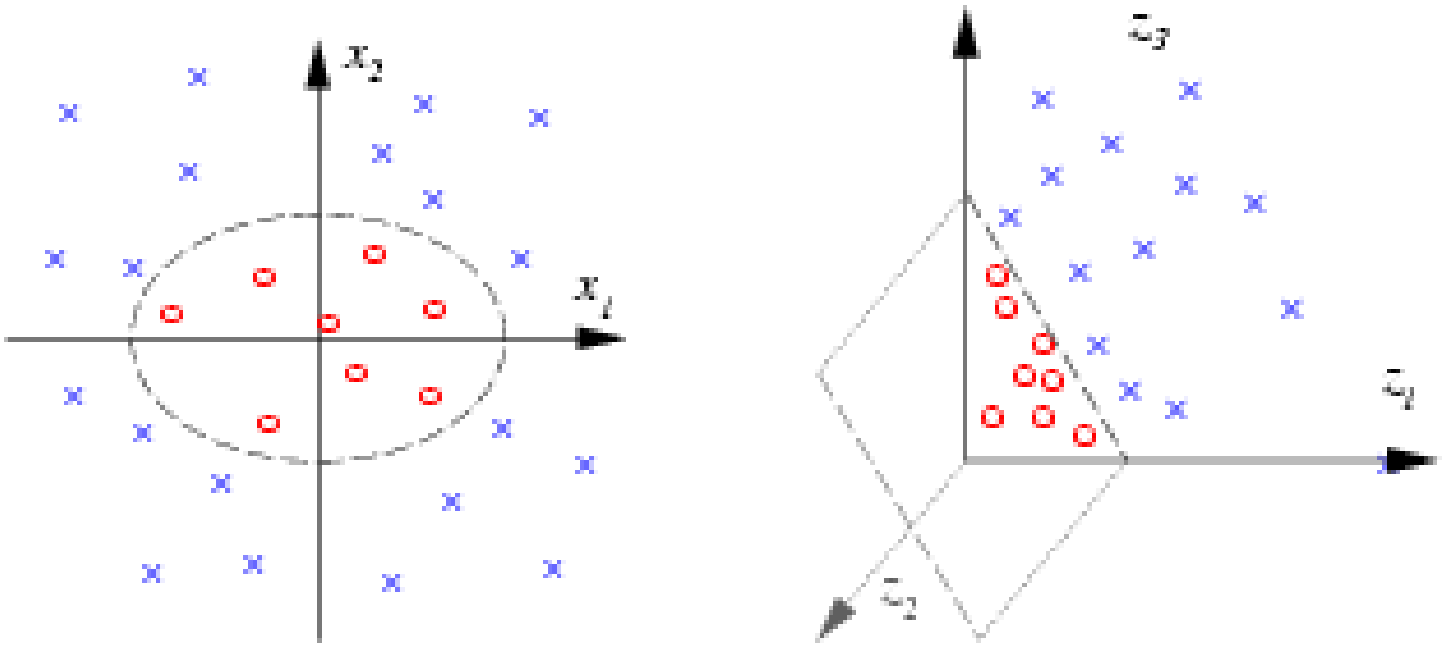


Fig. 4. Two dimensional classification example. Using the second order monomials $x_1^2, \sqrt{2}x_1x_2$ and x_2^2 as features a separation in feature space can be found using a *linear* hyperplane (right). In input space this construction corresponds to a *non-linear* ellipsoidal decision boundary (left) (figure from [48]).

A Pure Interpolation Approach

- Given: $(\mathbf{X}_i, d_i)$, i=1, …, n
- Desired: $f(\mathbf{X}_i)= d_i$
- Solution: $f(\mathbf{X})$, with $f(\mathbf{X}_i)= d_i$
- Radial basis function solution
  - $\phi(\mathbf{X},\mathbf{X}_i)$ – general form
  - ϕ is *shift* and *rotation* invariant
  - Shift invariant requires $\mathbf{X}-\mathbf{X}_i$
  - Rotation invariant requires $\| \mathbf{X}-\mathbf{X}_i \|$
- Example
  - Multiquadrics
  - Inserve Multiquadrics
  - Gaussan

$$f(\mathbf{X}) = \sum_i w_i \varphi(\|\mathbf{X} - \mathbf{X}_i\|)$$

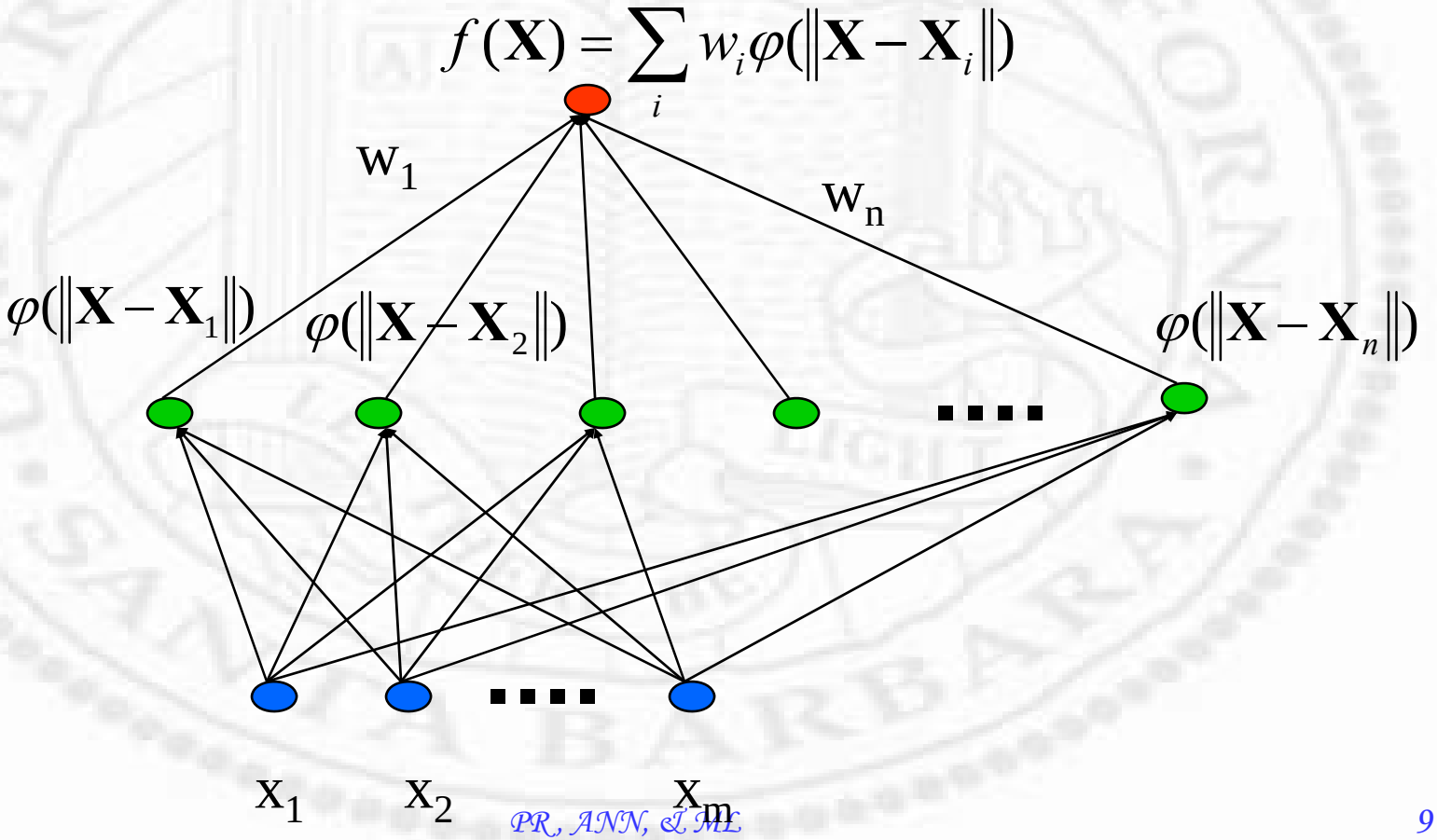
$$\varphi(r) = \sqrt{r^2 + c^2}$$

$$\varphi(r) = \frac{1}{\sqrt{r^2 + c^2}}$$

$$\varphi(r) = e^{-\frac{r^2}{2\sigma^2}}$$

Graphical Interpretation

- Each neuron responds based on the distance to the center of its receptive field
- The bottom level is a nonlinear mapping
- The top level is a linear weighted sum

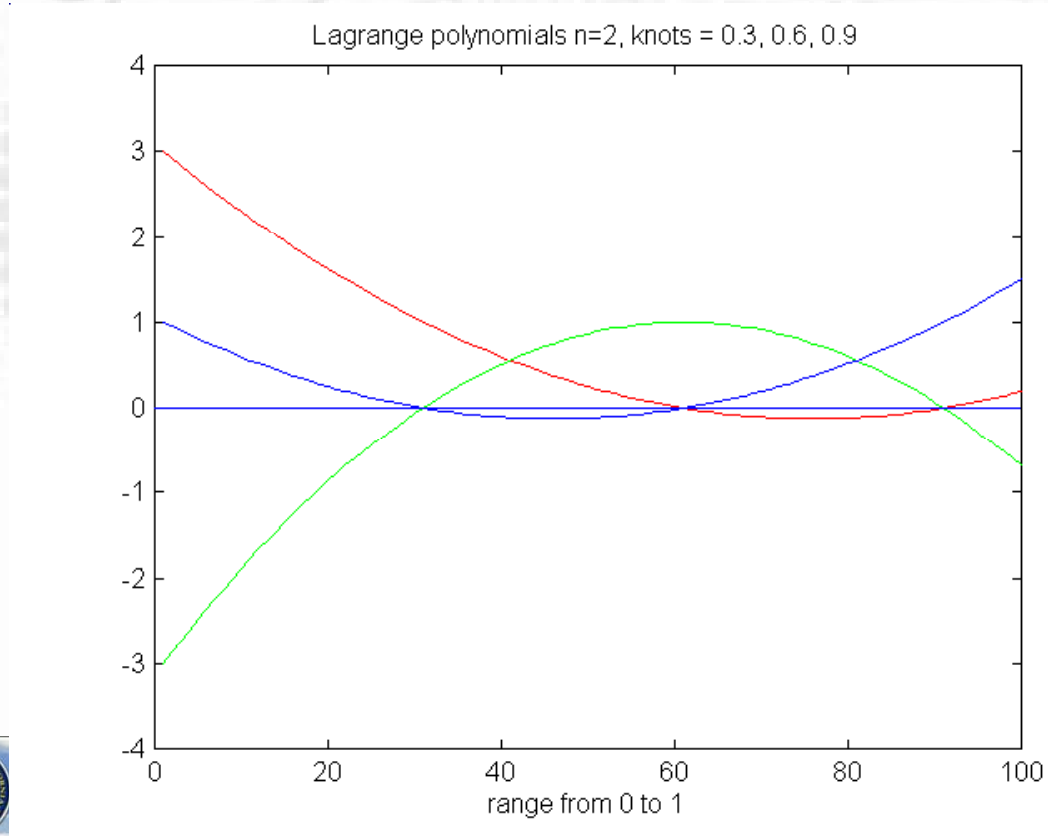


Other Alternatives: Global

- Lagrange polynomials

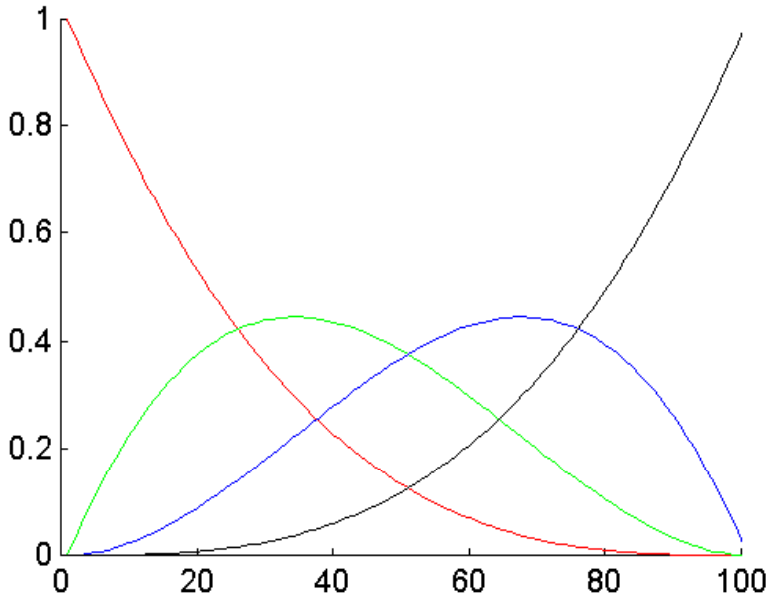
$$y = f(x) = \sum_{k=0}^{n} y_k L_{n,k}$$

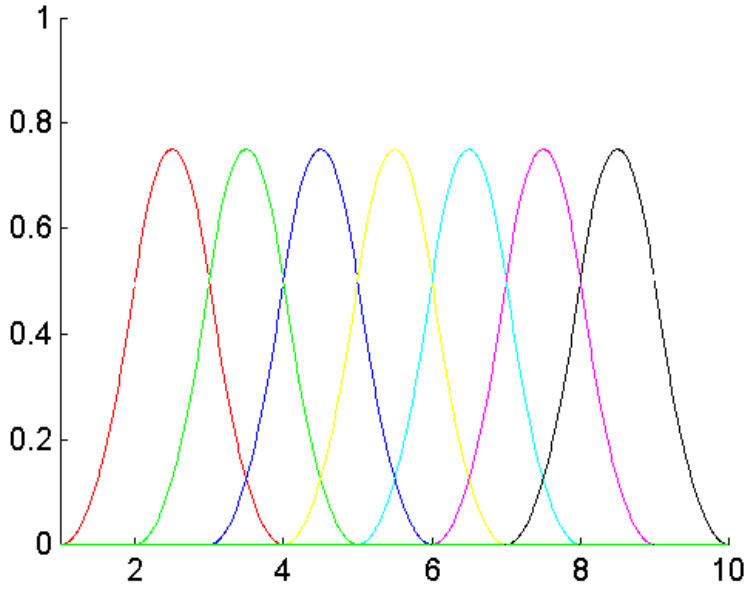
$$L_{n,k} = \frac{(x - x_o)(x - x_1)\cdots(x - x_{k-1})(x - x_{k+1})\cdots(x - x_n)}{(x_k - x_o)(x_k - x_1)\cdots(x_k - x_{k-1})(x_k - x_{k+1})\cdots(x_k - x_n)}$$



Other Alternatives: Local

- Bezier Basis
- B-spline basis





B-Spline Interpolation

- A big subject in mathematics
- Used in many disciplines
  - Approximation
  - Pattern recognition
  - Computer graphics
- As far as pattern recognition is concerned
  - Determine order of spline (DOFs)
    - Knot vectors (partition into intervals)
    - Fitting in each interval

Interpolation Solution

$$f(\mathbf{X}) = \sum_i w_i \varphi(\mathbf{X}, \mathbf{X}_i)$$

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1n} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ \varphi_{n1} & \varphi_{n2} & \cdots & \varphi_{nn} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} \qquad \varphi_{ij} = \varphi(\mathbf{X}_i, \mathbf{X}_j)$$

$$\mathbf{\Phi W} = \mathbf{D}$$

$$\mathbf{W} = \mathbf{\Phi}^{-1}\mathbf{D}$$

- $\mathbf{\Phi}$ is symmetrical
- $\mathbf{\Phi}$ is invertable (if all $\mathbf{X}_i$'s are distinct)

Practical Issue: Accuracy (cont.)

- The Φ function represents the Green's function for a certain differential operator
- When it is *shift* and *rotational* invariant, we can write $\Phi(\mathbf{X}, \mathbf{X}_i)$ as $G(\|\mathbf{X}-\mathbf{X}_i\|)$, again, Gaussian Kernel is a popular choice here

Practical Issues

- Accuracy
  - How about data are noisy?
- Speed
  - How about there are many sample points?
- Training
  - What is the training procedure?

Practical Issue: Accuracy

- When data are noisy, pure interpolation represents a form of "overfitting"
- Need a stabilizing (or smoothing, regularization) term
- The solution should achieve two things
  - Good fitting
  - Smoothness

Practical Issue: Accuracy (cont.)

$$\xi = \frac{1}{2}\sum_{i=1}^{n}(f(\mathbf{X}_i) - d_i)^2 + \frac{1}{2}\lambda \,|\, Df \,|^2$$

$$\xi = \frac{1}{2}\sum_{i=1}^{n}(d_i - \sum_{i=1}^{m} w_j G(\|\mathbf{X}_i - \mathbf{T}_j\|))^2 + \frac{1}{2}\lambda \,|\, Df^* \,|^2$$

$$(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{G}_o)\mathbf{W} = \mathbf{G}^T\mathbf{D}$$

$$\mathbf{W} = (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{G}_o)^{-1}\mathbf{G}^T\mathbf{D}$$

- The solution is rooted in the regularization theory, which is way beyond the scope of this course (read the papers on the class Web sites for more details)
- Try to minimize error as a weighted sum of two terms which impose the fitting and the smoothness constraints

Sidebar I

- It can be proven that MAP estimator (Baysian rule) gives the same results as regularized RBF solution
- Un-regularized (fitting) solution assumes the same prior

$$\xi = \frac{1}{2}\sum_{i=1}^{n}(f(\mathbf{X}_i) - d_i)^2 + \frac{1}{2}\lambda \| Df \|^2$$

$$P(f \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid f)P(f)}{P(\mathbf{X})}$$

$$-\log P(f \mid \mathbf{X}) = -\log P(\mathbf{X} \mid f) + (-\log P(f)) + c$$

Sidebar II

- Regularization is also similar to (or call) ridge regression in statistics
- The problem here is to fit a model to data without overfitting
- In linear case, we have

$$\mathbf{w}^{ridge} = \arg\min_{\mathbf{w}} \left\{ \sum_i (y_i - w_o - \sum_j x_{ij} w_j)^2 + \lambda \sum_j w_j^2 \right\}$$

$$\mathbf{w}^{ridge} = \arg\min_{\mathbf{w}} \left\{ \sum_i (y_i - w_o - \sum_j x_{ij} w_j)^2 \right\}$$

$$subject to \lambda \sum_j w_j^2 \le s$$

Intuition

- When variables x_i are highly correlated, their coefficients become poorly determined with high variance
  - E.g. widely large positive coefficient on one can be canceled by a similarly large negative coefficient on its correlated cousin
  - Size constraint is helpful
  - Caveat: constraint is problem dependent

Solution to Ridge Regression

- Similar to regularization

$$\mathbf{w}^{ridge} = \arg\min_{\mathbf{w}} \left\{ \sum_i (y_i - w_o - \sum_j x_{ij} w_j)^2 + \lambda \sum_j w_j^2 \right\}$$

$$\mathbf{w}^{ridge} = \arg\min_{\mathbf{w}} (\mathbf{XW} - \mathbf{Y})^T (\mathbf{XW} = \mathbf{Y}) + \lambda \mathbf{W}^T \mathbf{W}$$

$$\Rightarrow \frac{d(\mathbf{XW} - \mathbf{Y})^T (\mathbf{XW} - \mathbf{Y}) + \lambda \mathbf{W}^T \mathbf{W}}{d\mathbf{w}} = 0$$

$$\Rightarrow \mathbf{X}^T (\mathbf{XW} - \mathbf{Y}) + \lambda \mathbf{W} = 0$$

$$\Rightarrow (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}) \mathbf{W} = \mathbf{X}^T \mathbf{Y}$$

$$\Rightarrow \mathbf{w}^{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{Y}$$

Ugly Math

$$\mathbf{w}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$$

$$\mathbf{Y} = \mathbf{X}\mathbf{w}^{ridge} = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$$

$$= \mathbf{U}\boldsymbol{\Sigma}V^T(\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}V^T + \lambda\mathbf{I})^{-1}\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{Y}$$

$$= \mathbf{U}\boldsymbol{\Sigma}(V^{-T})^{-1}(\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}V^T + \lambda\mathbf{I})^{-1}(\mathbf{V}^{-1})^{-1}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{Y}$$

$$= \mathbf{U}\boldsymbol{\Sigma}(\mathbf{V}^{-1}\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}V^TV^{-T} + \mathbf{V}^{-1}\lambda\mathbf{I}V^{-T})^{-1}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{Y}$$

$$= \mathbf{U}\boldsymbol{\Sigma}(\boldsymbol{\Sigma}^T\boldsymbol{\Sigma} + \lambda\mathbf{I})^{-1}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{Y}$$

$$= \sum_i \mathbf{u}_i \frac{\sigma_i^2}{\sigma_i^2 + \lambda}\mathbf{u}_i{}^T\mathbf{Y}$$

Physical Interpretation

- Singular values of **X** represents the spread of data along different body-fitting dimensions
- To estimate $\mathbf{Y(=Xw^{ridge})}$ regularization minimizes the contribution from less spread-out dimensions
  - Less spread-out dimensions usually have much larger variance (high dimension eigen modes)
  - Trace $X(X^TX+\lambda I)^{-1}X^T$ is called effective degrees of freedom

More Details

- Trace $X(X^TX+\lambda I)^{-1}X^T$ is called effective degrees of freedom
  - Controls how many eigen modes are actually used or active
- Different methods are possible
  - Shrinking smoother: contributions are scaled
  - Projection smoother: contributions are used (1) or not used (0)

Practical Issue: Speed

- When there are many training samples, **G** and **Φ** matrices are of size n by n
- Inverting such a matrix is of $O(n^3)$
- Reducing the number of bases used

Practical Issue: Speed (cont.)

m<n

$$f^*(\mathbf{X}) = \sum_{j=1}^{m} w_j G(\|\mathbf{X} - \mathbf{T}_j\|)$$

$$\xi = \frac{1}{2}\sum_{i=1}^{n} (d_i - \sum_{i=1}^{m} w_j G(\|\mathbf{X}_i - \mathbf{T}_j\|))^2 + \frac{1}{2}\lambda \mid Df^* \mid^2$$

$$(\mathbf{G}^T\mathbf{G} + \lambda \mathbf{G}_o)\mathbf{W} = \mathbf{G}^T\mathbf{D}$$

$$\mathbf{G} = \begin{bmatrix} G(\mathbf{X}_1, \mathbf{T}_1) & G(\mathbf{X}_1, \mathbf{T}_2) & \cdots & G(\mathbf{X}_1, \mathbf{T}_m) \\ G(\mathbf{X}_2, \mathbf{T}_1) & G(\mathbf{X}_2, \mathbf{T}_2) & \cdots & G(\mathbf{X}_2, \mathbf{T}_m) \\ \cdots & \cdots & \cdots & \cdots \\ G(\mathbf{X}_n, \mathbf{T}_1) & G(\mathbf{X}_n, \mathbf{T}_2) & \cdots & G(\mathbf{X}_n, \mathbf{T}_m) \end{bmatrix}_{n \times m}$$

$$\mathbf{G}_o = \begin{bmatrix} G(\mathbf{T}_1, \mathbf{T}_1) & G(\mathbf{T}_1, \mathbf{T}_2) & \cdots & G(\mathbf{T}_1, \mathbf{T}_m) \\ G(\mathbf{T}_2, \mathbf{T}_1) & G(\mathbf{T}_2, \mathbf{T}_2) & \cdots & G(\mathbf{T}_2, \mathbf{T}_m) \\ \cdots & \cdots & \cdots & \cdots \\ G(\mathbf{T}_n, \mathbf{T}_1) & G(\mathbf{T}_n, \mathbf{T}_2) & \cdots & G(\mathbf{T}_n, \mathbf{T}_m) \end{bmatrix}_{m \times m}$$

Practical Issue: Training

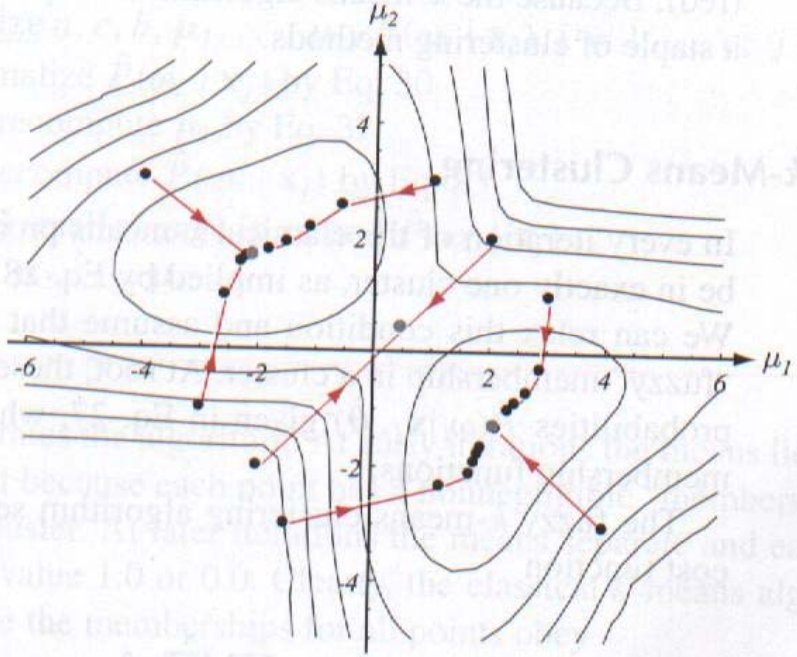
- How can the center of radial basis functions for the reduced basis set be determined?
- Chosen randomly
- Training involves finding $w_{i,}$ using SVD

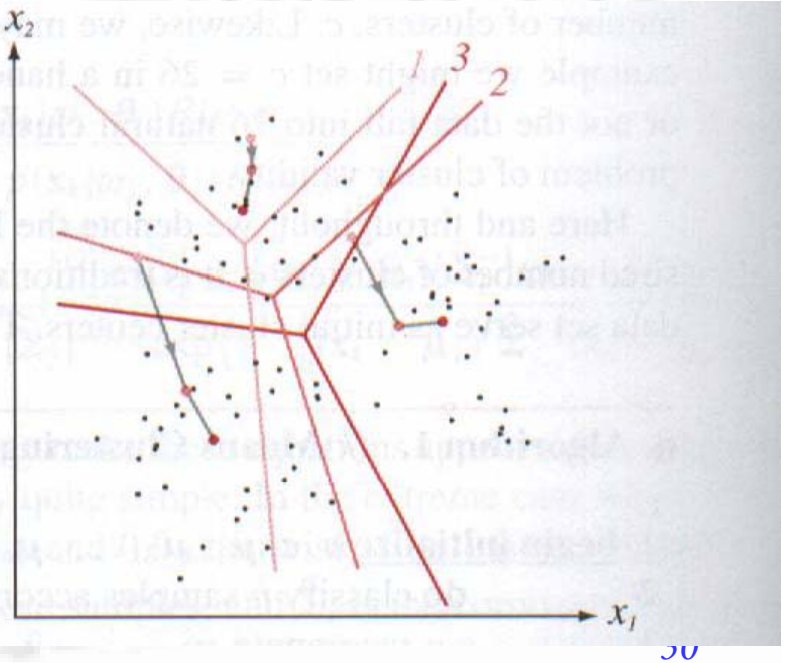
Training with K-mean

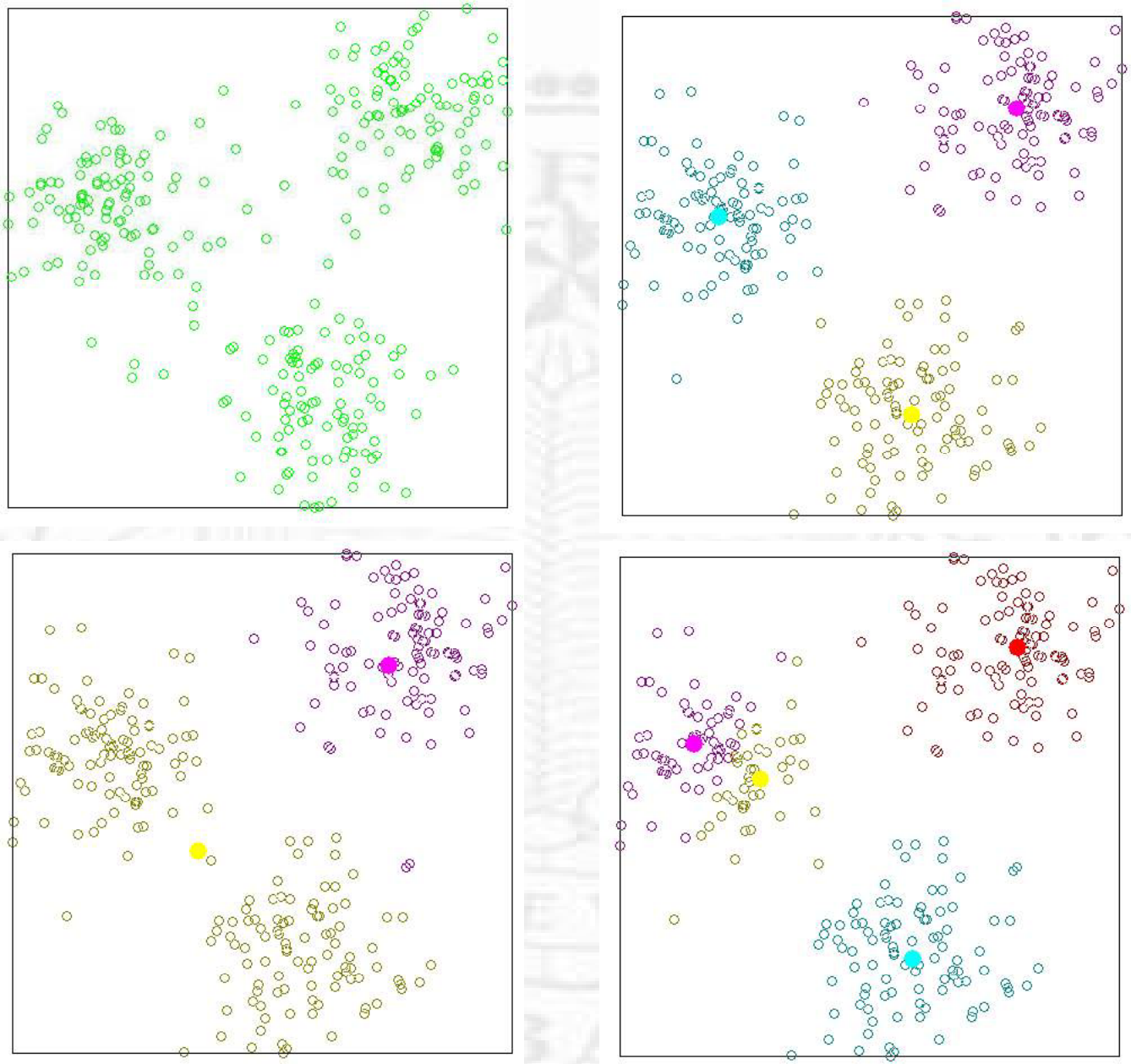
- Using unsupervised clustering
- Find where data are clustered – that is where the radial basis functions should be placed
- With k-mean

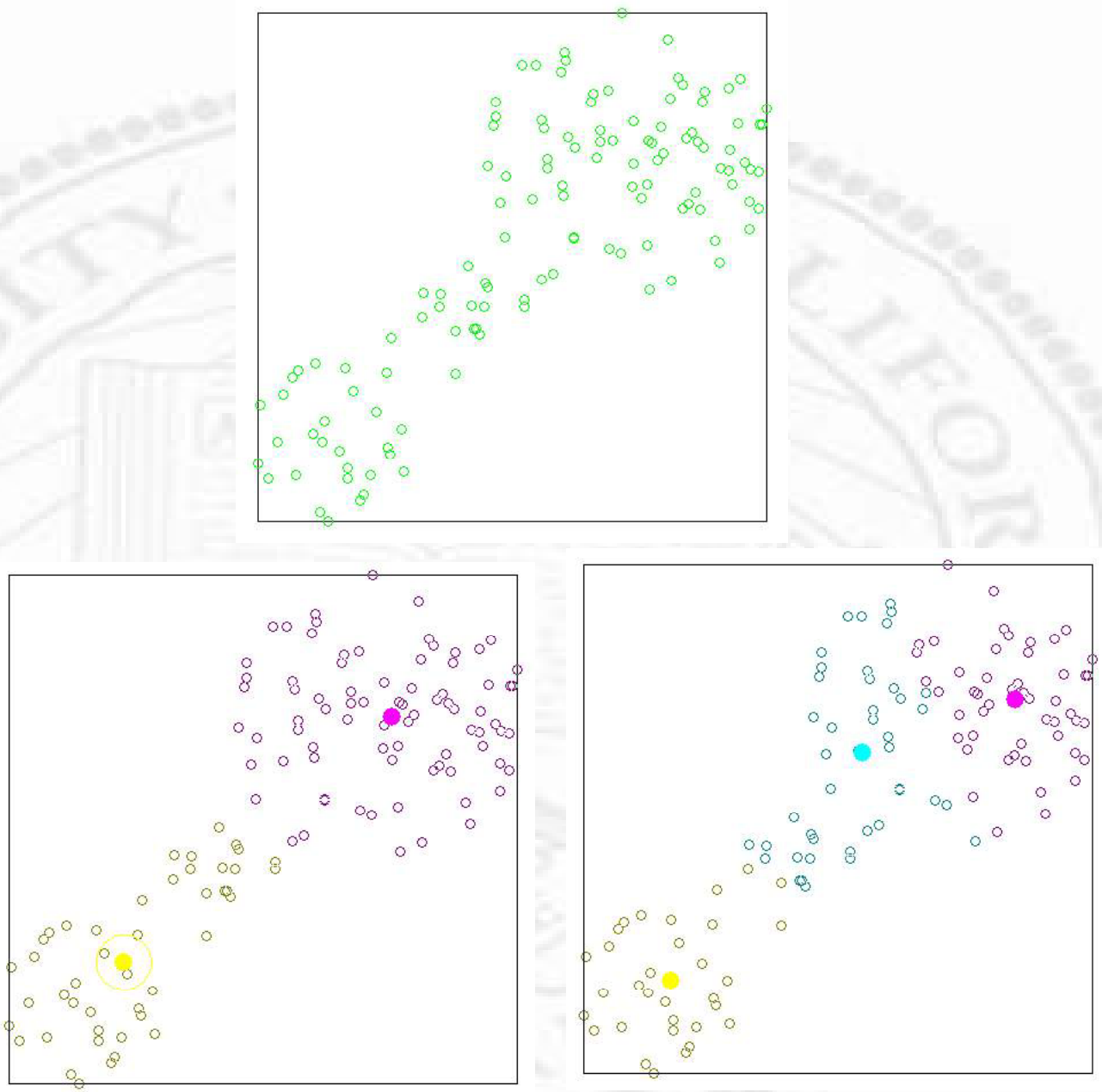
K-Means Algorithm (fixed # of clusters)

- Arbitrarily pick *N* cluster centers, assign samples to nearest center
- Compute sample mean of each cluster
- Reassign samples to clusters with the nearest mean (for all samples)
- Repeat if there are changes, otherwise stop


μ_2
μ_1
4
2
-2
-4
-6
-4
-2
2
4
6


x_2
x_1
1
3
2





Training with Gradient Decent

- Error Expression

$$\xi^{(n)} = \frac{1}{2}\sum_{i=1}^{n}(d_i - \sum_{j=1}^{m} w_j^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|))^2$$

- Free variables in the error expression are
  - Weight
  - Center location
  - Basis spread

Effect of Weights

$$\xi^{(n)} = \frac{1}{2}\sum_{i=1}^{n}(d_i - \sum_{j=1}^{m} w_j^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|))^2$$

$$\xi_i^{(n)} = d_i - \sum_{j=1}^{m} w_j^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|)$$

$$\frac{\partial \xi^{(n)}}{\partial w_j^{(n)}} = \sum_{i=1}^{n} \xi_i^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|)$$

$$w_j^{(n+1)} = w_j^{(n)} - \eta_w \frac{\partial \xi^{(n)}}{\partial w_j^{(n)}}$$

Effect of Center Positions

$$\xi^{(n)} = \frac{1}{2}\sum_{i=1}^{n}(d_i - \sum_{j=1}^{m} w_j^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|))^2$$

$$\xi_i^{(n)} = d_i - \sum_{j=1}^{m} w_j^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|)$$

$$\frac{\partial \xi^{(n)}}{\partial \mathbf{T}_j^{(n)}} = 2 w_j^{(n)} \sum_{i=1}^{n} \xi_i^{(n)} G'(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|) \mathbf{\Sigma}^{-1} (\mathbf{X}_i - \mathbf{T}_j^{(n)})$$

$$\mathbf{T}_j^{(n+1)} = \mathbf{T}_j^{(n)} - \eta_{\mathbf{T}} \frac{\partial \xi^{(n)}}{\partial \mathbf{T}_j^{(n)}}$$

Effect of Basis Spread

$$\xi^{(n)} = \frac{1}{2}\sum_{i=1}^{n}(d_i - \sum_{j=1}^{m} w_j^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|))^2$$

$$\xi_i^{(n)} = d_i - \sum_{j=1}^{m} w_j^{(n)} G(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|)$$

$$\frac{\partial \xi^{(n)}}{\partial \mathbf{\Sigma}_j^{-1\,(n)}} = -w_j^{(n)} \sum_{i=1}^{n} \xi_i^{(n)} G'(\left\| \mathbf{X}_i - \mathbf{T}_j^{(n)} \right\|) \mathbf{Q}_{ij}^{(n)}$$

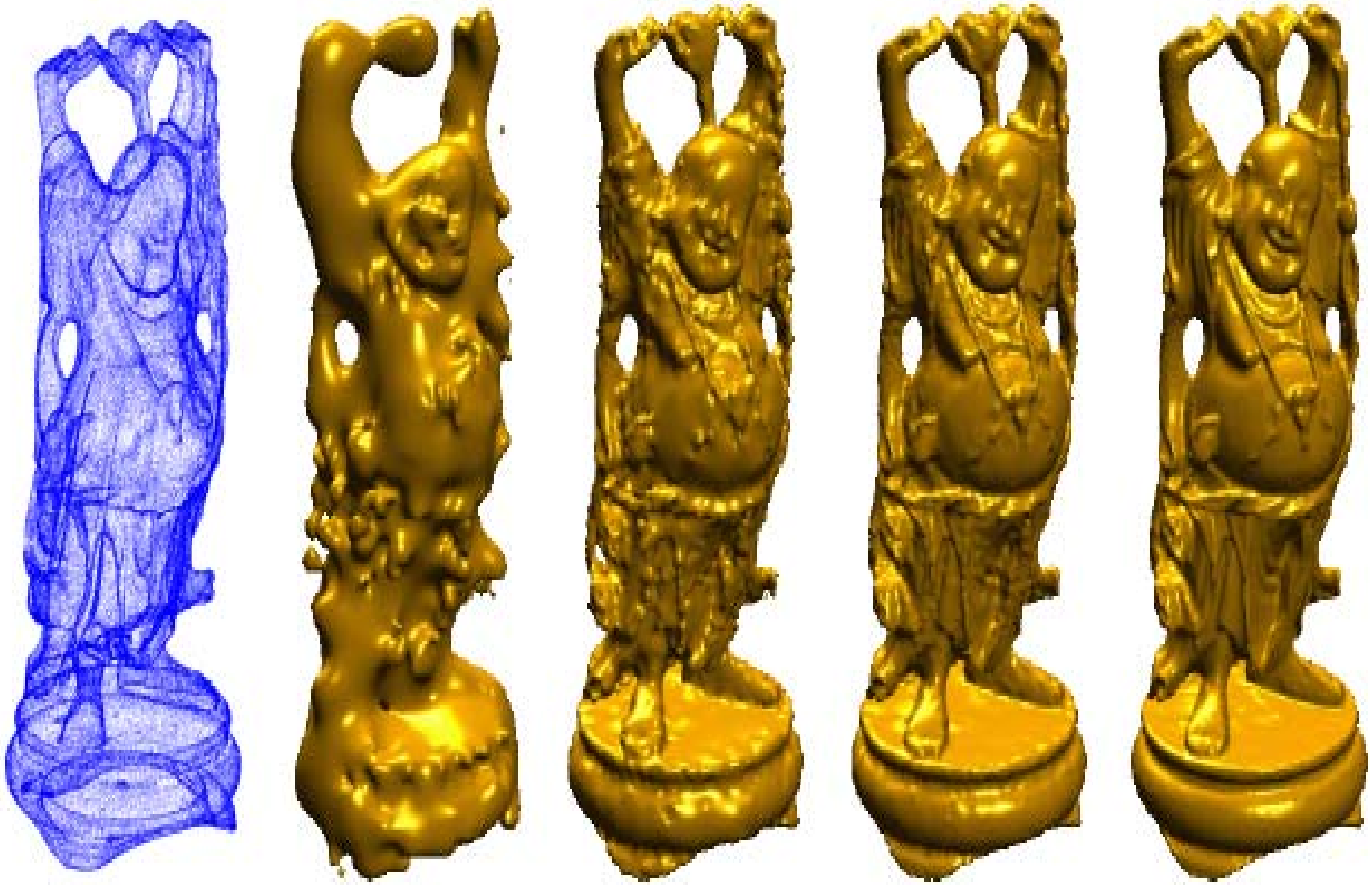
$$\mathbf{Q}_{ij}^{(n)} = (\mathbf{X}_i - \mathbf{T}_j^{(n)})(\mathbf{X}_i - \mathbf{T}_j^{(n)})^T$$

$$\mathbf{\Sigma}_j^{-1(n+1)} = \mathbf{\Sigma}_j^{-1(n)} - \eta_{\mathbf{\Sigma}_j^{-1}} \frac{\partial \xi^{(n)}}{\partial \mathbf{\Sigma}_j^{-1\,(n)}}$$

Details

- A lot of theoretical development results are omitted here
  - E.g., relation to kernel regression and SVM
- A lot of tuning considerations are not covered here
  - E.g., how to determine λ?
- This is an active research area

Examples



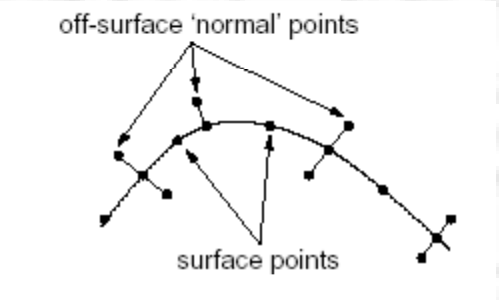
544,000 data points w. 80,000 centers
Accuracy of 1.4×10^{-6} for all data points

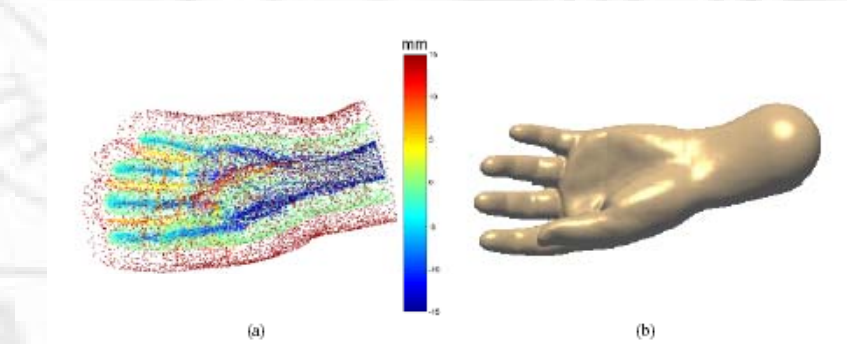
Problem Definition

- Given a point cloud of data
  - From laser range scanner, or
  - CT, MR, etc.
- Find a *single* analytical surface approximation
- Or an inside-outside function
  - Range data are $s(\mathbf{X})=0$
  - Outside is $s(\mathbf{X})>0$
  - Inside is $s(\mathbf{X})<0$
- Just sample data $s(\mathbf{X})=0$ is not enough
  - s can be a trivial zero function
  - Need off-surface data generation

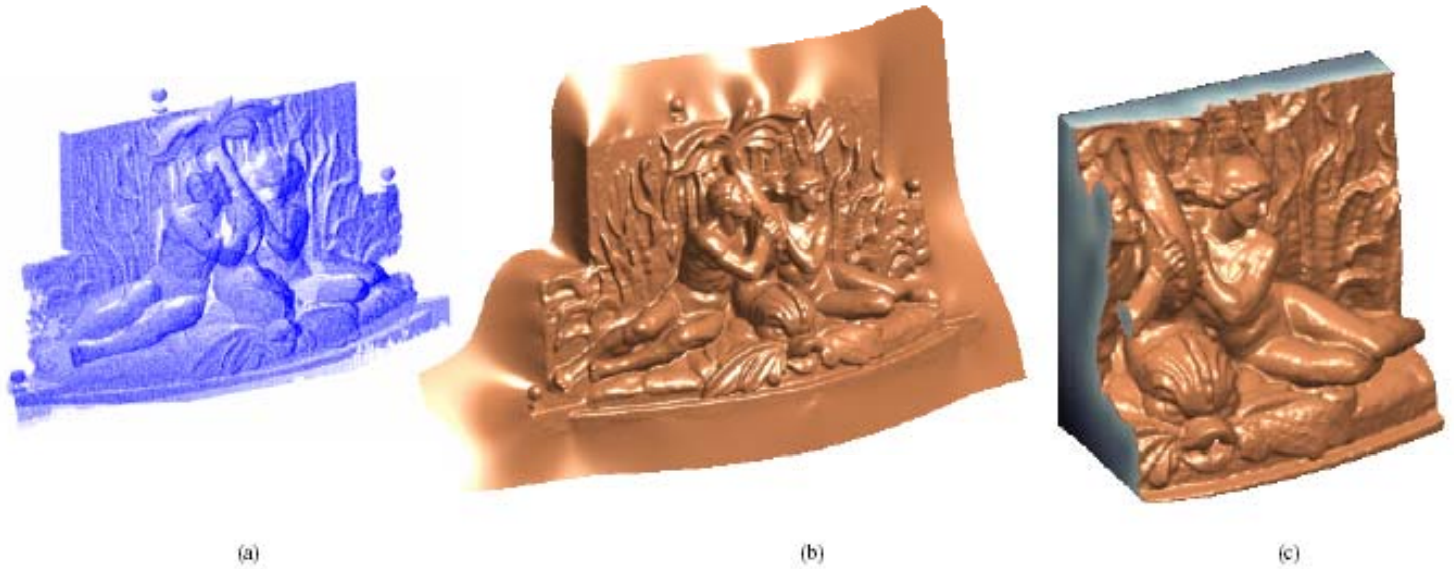
Procedures

1. Off surface data generation
2. Choose a subset from the interpolation node $\mathbf{x}_i$ and fit an RBF only to these
3. Evaluate the resideual $e_i = \mathbf{f}_i - s(\mathbf{x}_i)$
4. If $\max(e_i)$<accuracy, then stop
5. Else append new centers where e_i is large
6. Re-fit RBF and go back to step 2





More Results



(a) (b) (c)

Figure 8: RBF approximation of noisy LIDAR data. (a) 350,000 point-cloud, (b) the smooth RBF surface approximates the original point-cloud data, (c) cut-away view illustrating the RBF distance field and the preservation of the gap between the arm and the torso.

(a) (b) (c)

Less smoothing ← → More smoothing