PDEEC – Machine Learning 2018/19

Lecture
Unsupervised Learning:
SOMs, principal components, multidimensional scaling

Jaime S. Cardoso
jaime.cardoso@inesctec.pt

INESC TEC and Faculdade Engenharia, Universidade do Porto

Dec. 13, 2018

Empirical Learning

Supervised learning

Supervised learning is concerned with predicting the values of one or more outputs or response variables Y for a given set of input or predictor variables X.

- the predictions are based on the training sample $(\mathbf{x}_i, \mathbf{y}_i)$ of previously solved cases
- if (X, Y) are random variables with joint probability density $Pr(X, Y)$, then supervised learning can be characterized as a density estimation problem where one is concerned with determining properties of the conditional density $Pr(Y|X)$.
- distint approaches to solving the decision problem:
  1. generative models: first solve the inference problem of determining the joint density $p(x, y)$ (or $p(x, \mathcal{C}_k)$). Then normalize to obtain the posterior probabilities $p(y|x)$. Finally the conditional mean (or class membership) for each new input x.
  2. first solve the inference problem of determining the conditional density $p(y|x)$ and then subsequently calculate the conditional mean
  3. find a regression (or a discriminant) function directly from the training data.

Empirical Learning

Unsupervised learning

Unsupervised learning is concerned with inferring properties of the probability density $p(x)$ without the help of a supervisor or teacher providing correct answers or a degree-of-error for each observation. The goal is to characterize x-values, or collections of such values, where $p(x)$ is relatively large.

- principal components, multidimensional scaling, self-organizing maps, principal curves, etc, attempt to identify low-dimensional manifolds within the x-space that represent high data density.
- **cluster analysis** attempts to find multiple (convex) regions of the x-space that contain modes of $p(x)$.
- **association rules** attempt to construct simple descriptions (conjunctive rules) that describe regions of high density in the special case of very high dimensional binary-valued data.

Empirical Learning

Unsupervised learning

Self-organizing map (SOM)

- can be viewed as a constrained version of K-means clustering, in which the prototypes are encouraged to lie in a one- or two-dimensional manifold (A manifold is a mathematical space in which every point has a neighborhood which resembles Euclidean space, but in which the global structure may be more complicated.) in the feature space.
- The map seeks to preserve the topological properties of the input space.
- This makes SOM useful for visualizing low-dimensional views of high-dimensional data, akin to multidimensional scaling. The model was first described as an artificial neural network by the Finnish professor Teuvo Kohonen, and is sometimes called a Kohonen map.

Empirical Learning

Self-organizing map (SOM)

- Starting point:
  - set of high dimension data points $\mathbf{x}_i \in \mathbf{R}^d$
  - set of K prototypes in the same space $p_i \in \mathbf{R}^d$
- Analogy to K-means clustering
  - find "nearest" prototype for current data point $\rightarrow$ "winning prototype" (competitive learning)
  - associate data point to that prototype
  - update prototype to resemble data
  - minimize average quantization error $\frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} ||x_n - p_k||^2$

Empirical Learning

Self-organizing map (SOM)

SOM specific extensions:

- introduction of neighbourhoods for the prototypes
  - K-Means $\rightarrow$ leads to single unrelated clusters
  - relationships on superior cluster level not captured
- Improvement of the SOM
  - neighbourhoods reflect similarity of clusters
  - similar clusters become automatically adjacent
  - highly dissimilar clusters are spatially separated

Empirical Learning

Self-organizing map (SOM)

Neighbourhoods:

- prototypes are assigned to distinct points (coordinates) on a low dimensional structure (manifold) of target space, e.g. a 2 dimensional or 3D grid
- neighbourhood is defined as a function h() on this manifold
- Note: "adjacent prototypes" do not need to be similar (e.g. measured by Euclidian distance)
- Produced Effect:
  - "mapping" of low dimensional structure into data space

Empirical Learning

Self-organizing map (SOM)

(online) training:

1. Choose new data sample x_n
2. Compute "nearest" prototype vector p^* satisfying $dist(x_n, p^*) = \min_k \{dist(x, p_k)\}$
3. Update step:

$$p_k(t+1) \leftarrow p_k(t) + \alpha(t) \underbrace{h(t, p^*, p_k)}_{neighbourhood} (x_n - p_k(t))$$

4. Repeat steps 1-3 while cycling repeatively through the trainings set

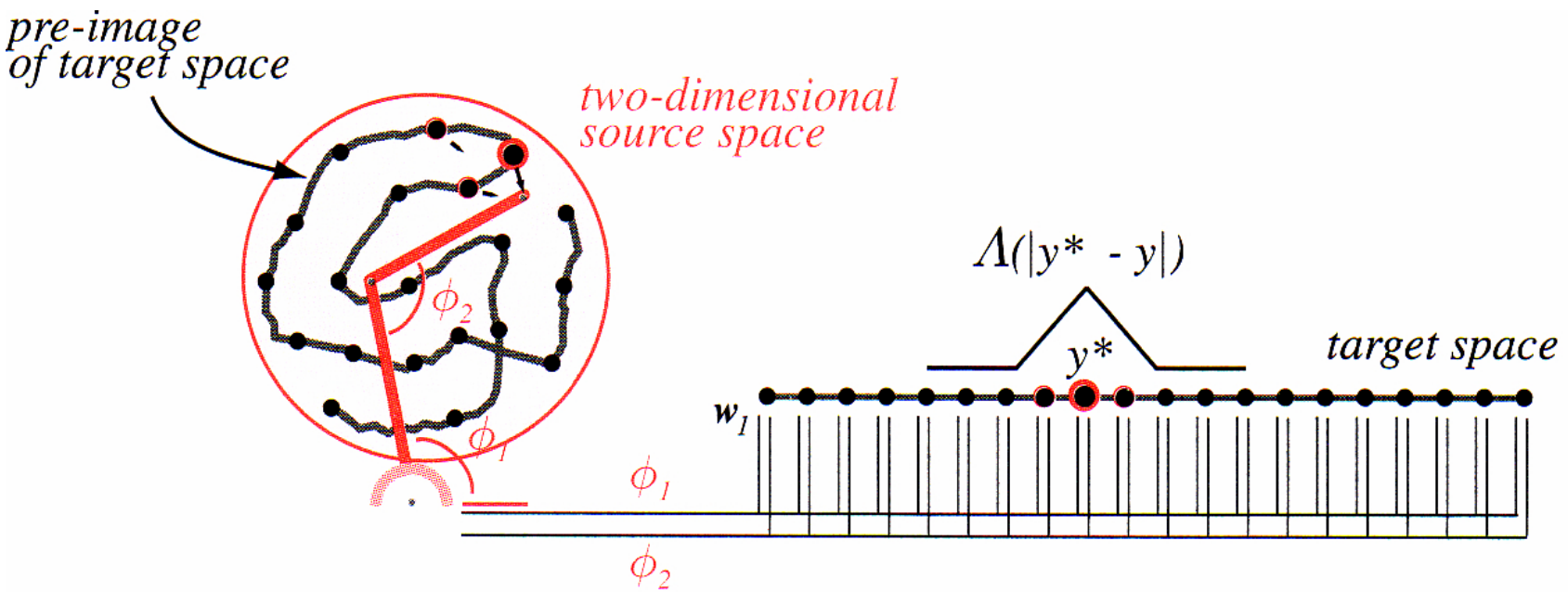
Effects: moves prototypes closer to the data while maintaining a smooth relationship between prototypes

Empirical Learning

Self-organizing map (SOM)

Example:

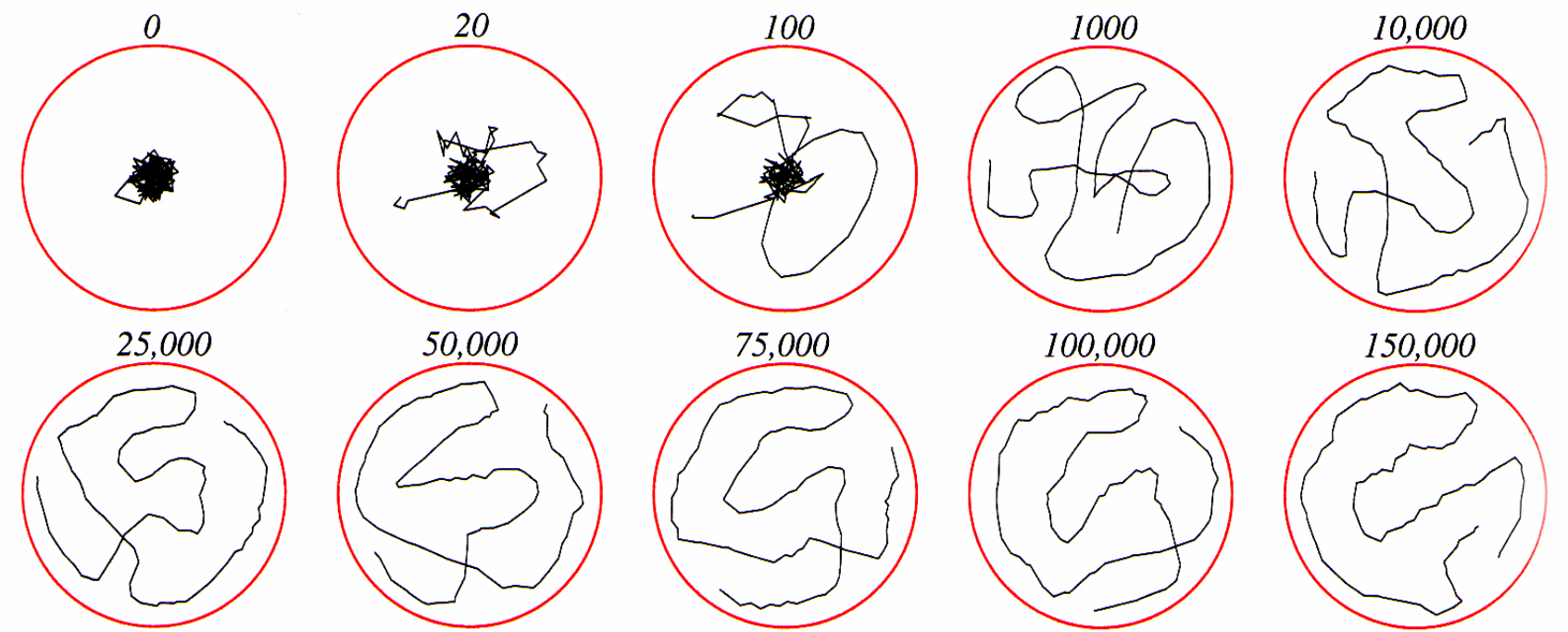
- 1 dimensional target structure (a chain or line)
- simple neighbourhoods $\rightarrow$ "next one left" AND "next one right"
- data from angles of a robot's arm



Empirical Learning

Self-organizing map (SOM)

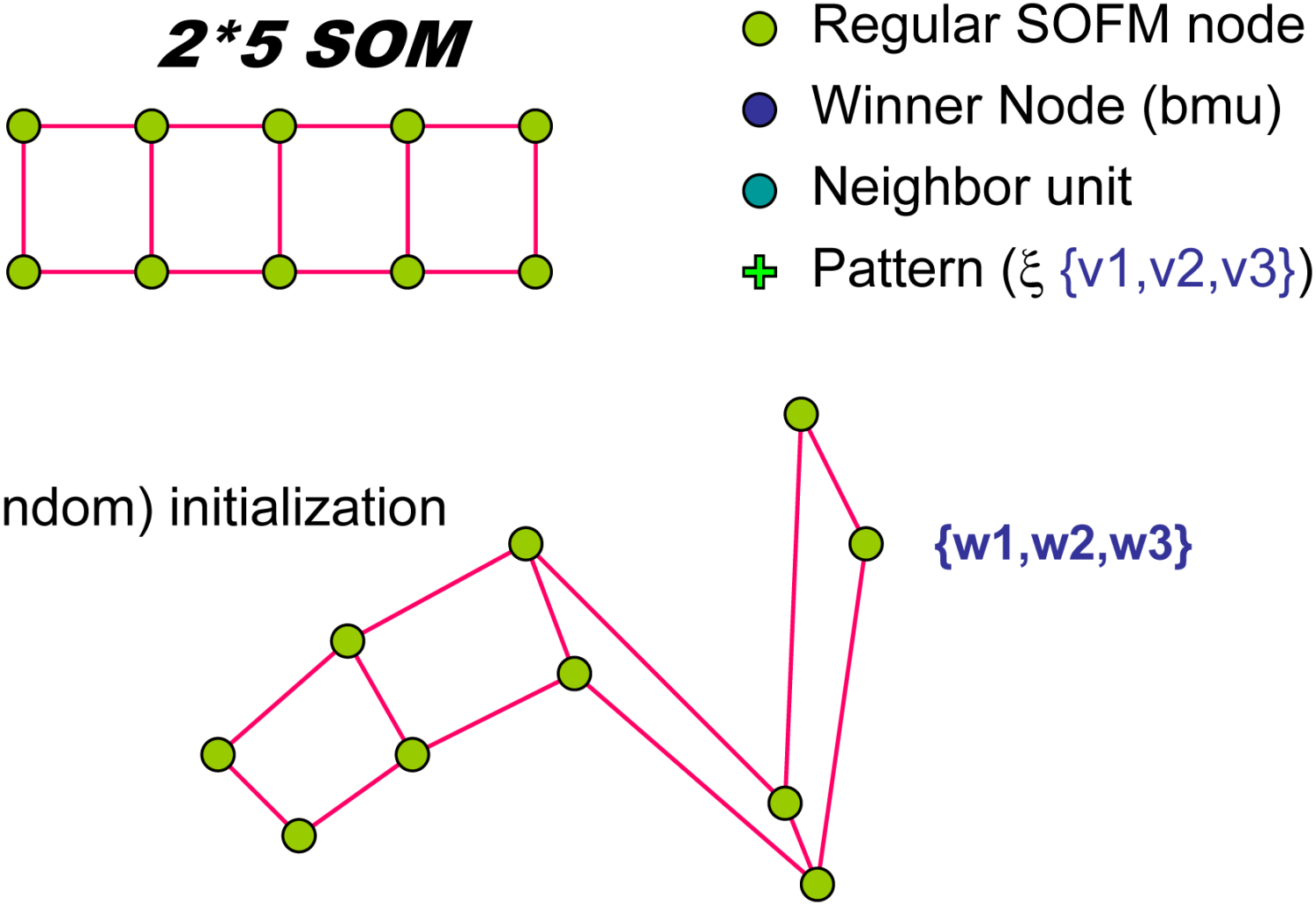
Example: development of the map in the course of training



Empirical Learning

Self-organizing map (SOM)

Example2:



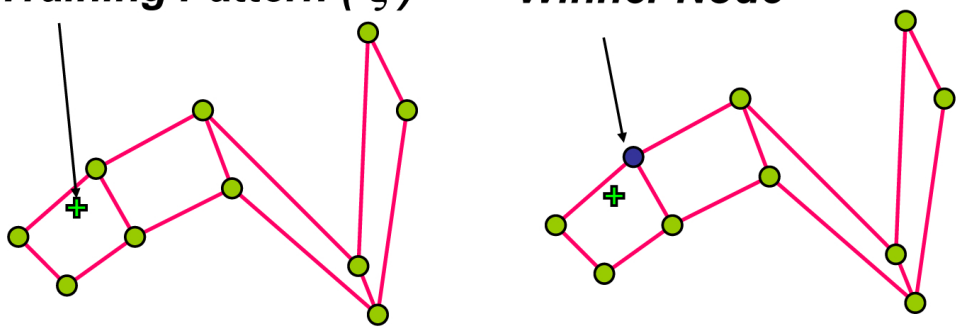
Empirical Learning

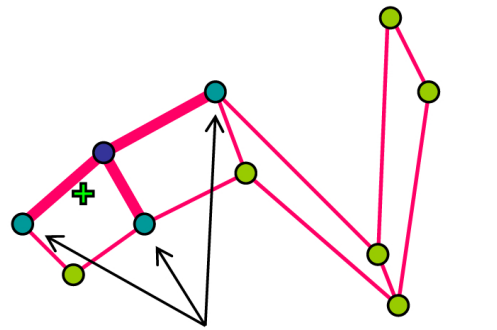
Self-organizing map (SOM)

Example2:

(Similarity measure: Euclidean distance)

Training Pattern (ξ) ***Winner Node***



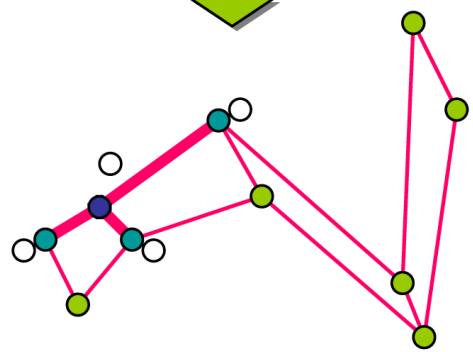


Neighborhood (N(t))

$\alpha(t)$: **adaptation strength** $\alpha(t) = \alpha_0 \cdot \frac{1}{\sqrt{(1+t)}}$

Learning Rule:

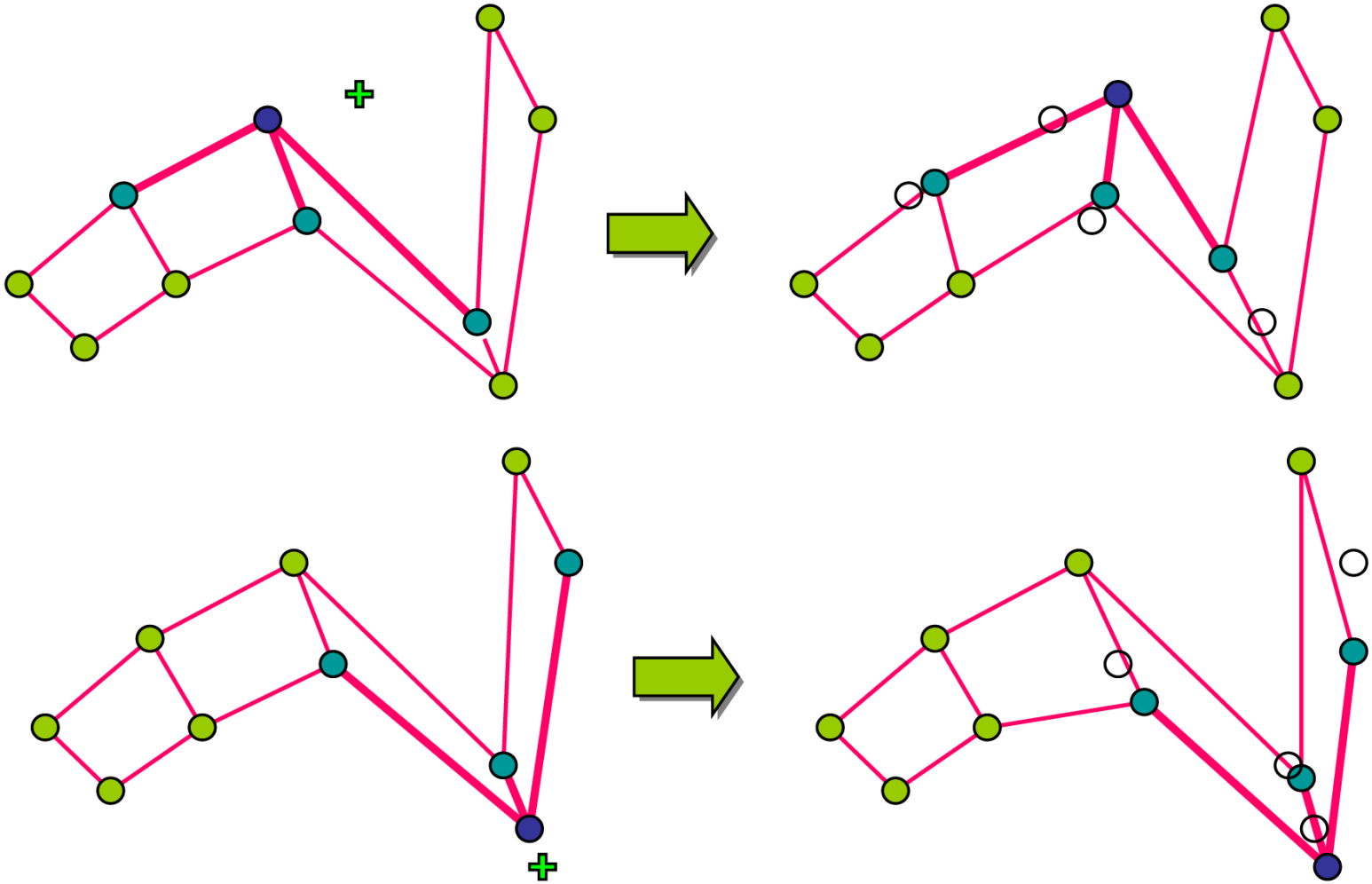
$$\begin{cases} w^i(t+1) = w^i(t) + \alpha(t) \cdot \left[\xi - w^i(t)\right] \text{ if } i \in N(t)_{winner} \\ w^i(t+1) = w^i(t) \text{ if } i \notin N(t)_{winner} \end{cases}$$



Empirical Learning

Self-organizing map (SOM)

Example2:



Empirical Learning

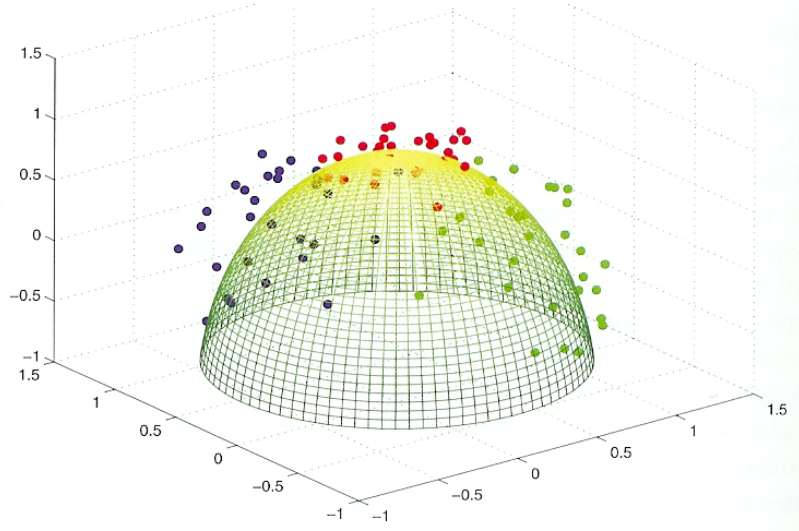
Self-organizing map (SOM)

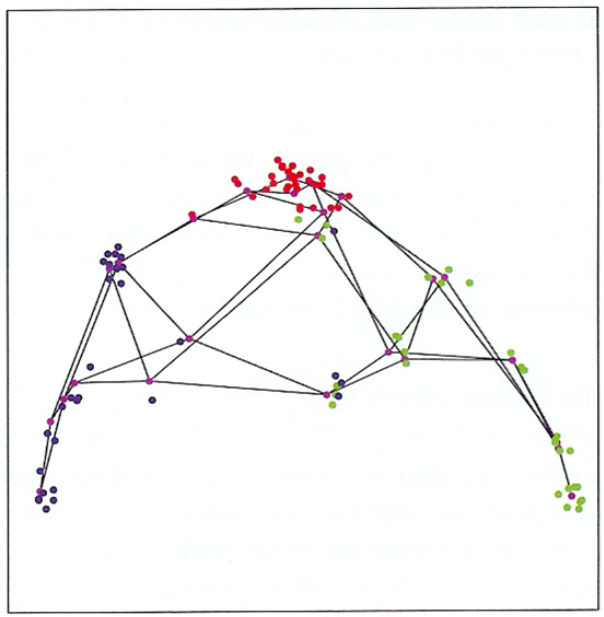
Comments:

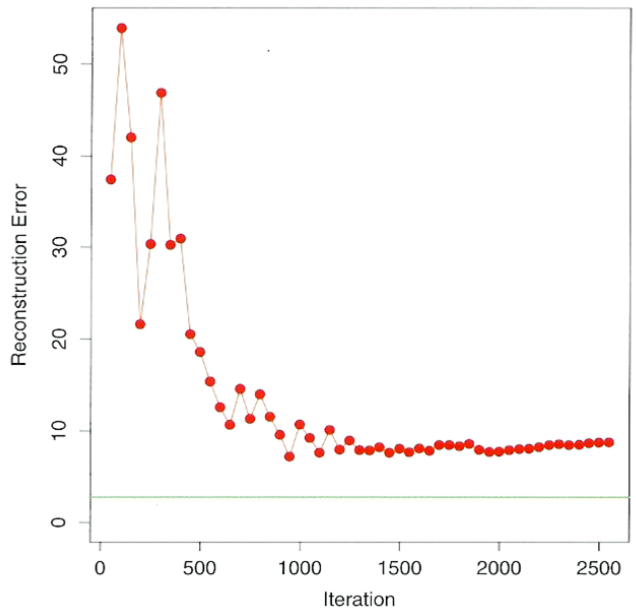
- initialization influences results
- principal component axes $\rightarrow$ improved performance
- fold-back problem $\rightarrow$ reinitialize, change parameters
- Parameters:
  - Learning rate α: decrease over time $1.0 \rightarrow 0.0$
  - Neighbourhood threshold: "radius" of considered neighbourhood: decreased over time
  - distance measure
  - number of prototypes

Empirical Learning

Self-organizing map (SOM)







Empirical Learning

Self-organizing map (SOM)

- If neighbourhood is small enough, each neighbourhood contains only one point $\rightarrow$ spatial connection between prototypes is lost $\rightarrow$ converges at a local minima of K-means clustering
- Need to check if the constraint is reasonable: compute and compare reconstruction error $\epsilon = ||x - m||^2$ for both methods (SOM's ϵ is bigger, but should be similar)

Empirical Learning

Unsupervised learning

Self-organizing map (SOM)

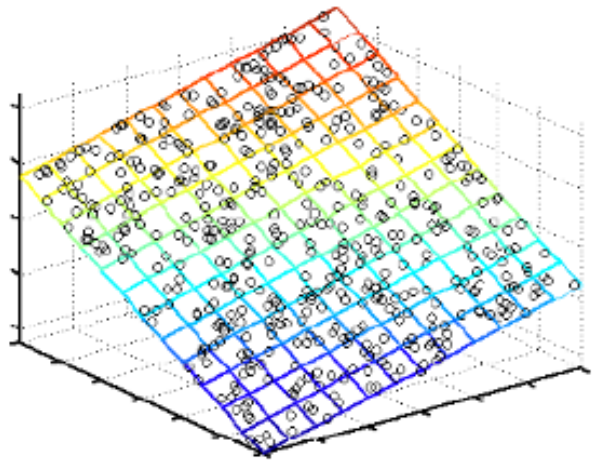
- There are two ways to interpret a SOM. Because in the training phase weights of the whole neighborhood are moved in the same direction, similar items tend to excite adjacent neurons. Therefore, SOM forms a semantic map where similar samples are mapped close together and dissimilar apart.
- The other way is to think of neuronal weights as pointers to the input space. They form a discrete approximation of the distribution of training samples. More neurons point to regions with high training sample concentration and fewer where the samples are scarce.
- SOM may be considered a nonlinear generalization of principal component analysis.

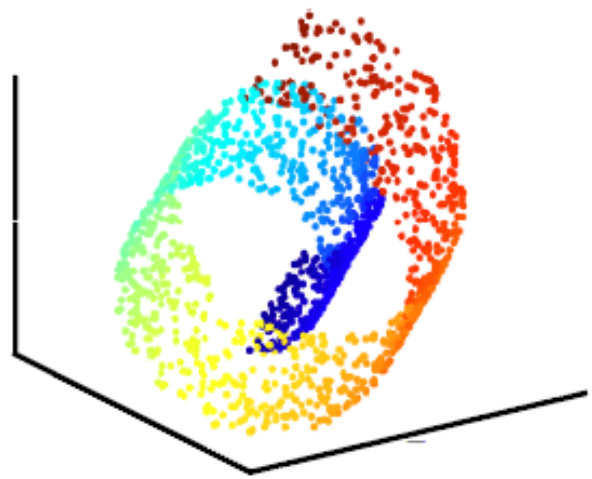
Latent Feature Extraction

Combinations of observed features provide more efficient representation, and capture underlying relations that govern the data.

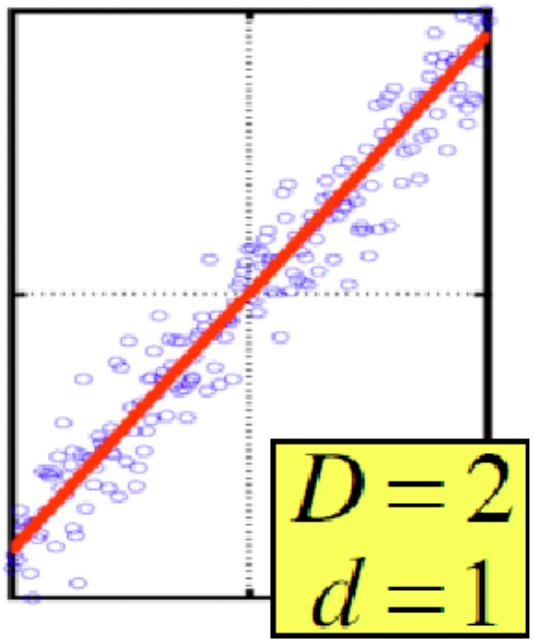
Often may not have physical meaning.

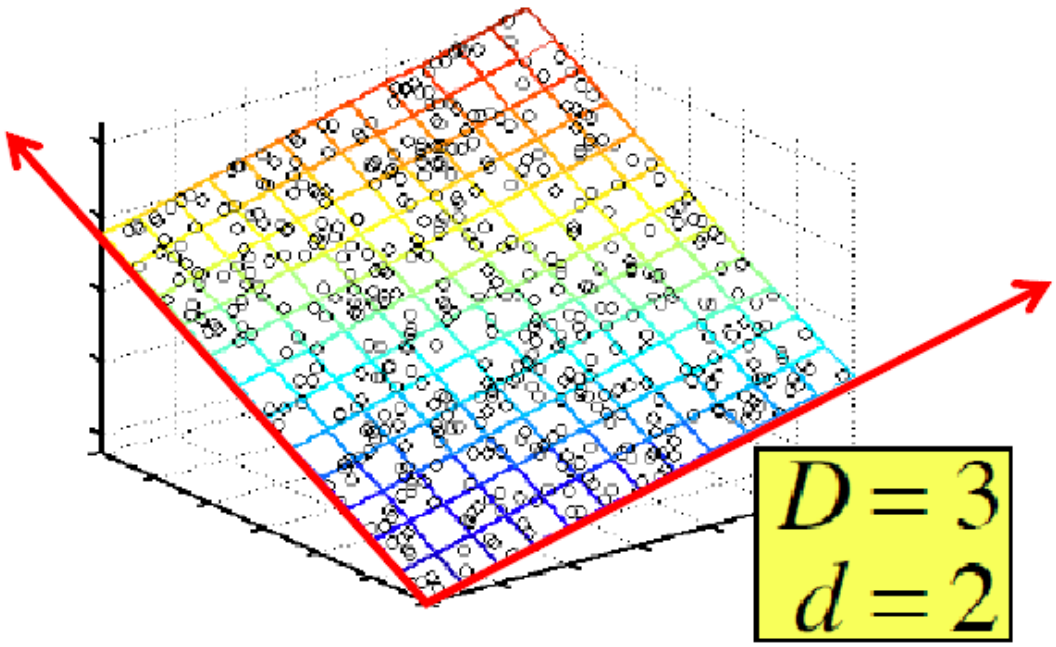
- Linear (Principal Component Analysis (PCA), Factor Analysis, etc)
- Nonlinear (multidimensional scaling, ISOMAP, Local Linear Embedding (LLE), etc)





Principal Component Analysis (PCA)





Assumption: Data lies on or near a low d-dimensional linear subspace.

Axes of this subspace are an effective representation of the data

Identifying the axes is known as Principal Components Analysis, and can be obtained by Eigen or Singular value decomposition

Empirical Learning

Unsupervised learning

Principal component analysis PCA

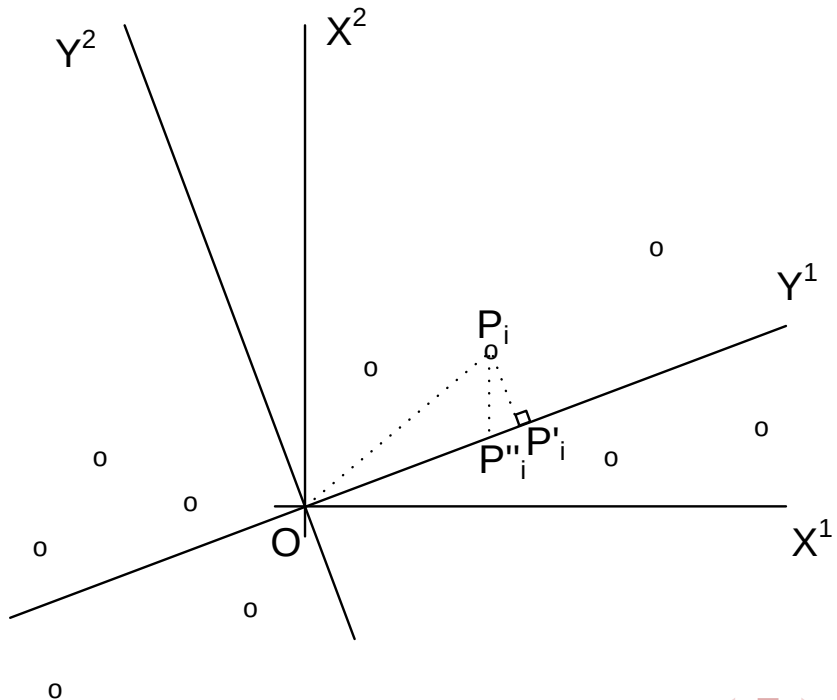
- the principal components of a set of data in $\mathbf{R}^d$ provide a sequence of best linear approximations to that data
- consider a set of N observations $\mathbf{x}_n \in \mathbf{R}^d$
- consider the center of the referencial on the mean value of the data: $\bar{\mathbf{x}} = \mathbf{0}$

Empirical Learning

Principal component analysis PCA

Example

- (X^1, X^2) is the original referencial; (Y^1, Y^2) is a new system
- the goal of PCA is to find Y^1 such that the error of approximating P_i by P_i' is minimum: $\min \sum_n \overline{P_n P_n'}^2$
- Note the difference to standard regression, where the goal in to $\min \sum_n \overline{P_n P_n''}^2$

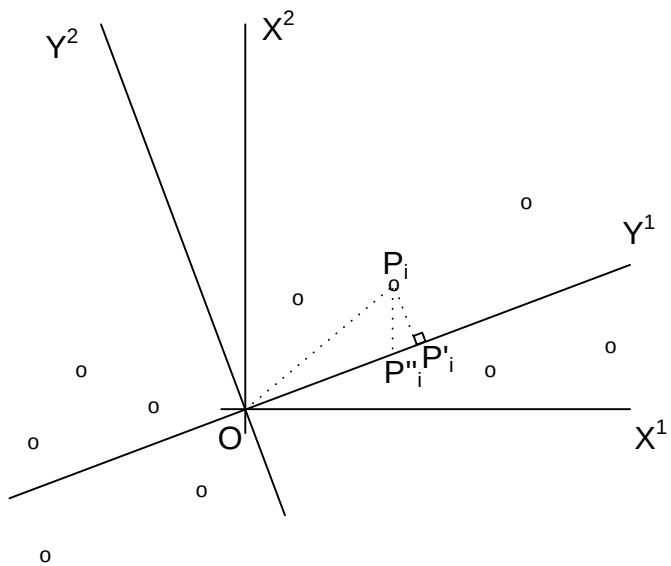


Empirical Learning

Principal component analysis PCA

Example

- $\overline{OP_i}^2 = \overline{OP_i'}^2 + \overline{P_iP_i'}^2$
  $\frac{1}{N}\sum_i \overline{OP_i}^2 = \frac{1}{N}\sum_i \overline{OP_i'}^2 + \frac{1}{N}\sum_i \overline{P_iP_i'}^2$
- minimize $\sum_i \overline{P_iP_i'}^2$ is equivalent to maximize $\sum_i \overline{OP_i'}^2$
- since the average of the set of sample is zero, $\sum_i \overline{OP_i'}^2$ is the variance over Y^1



Empirical Learning

Principal component analysis PCA

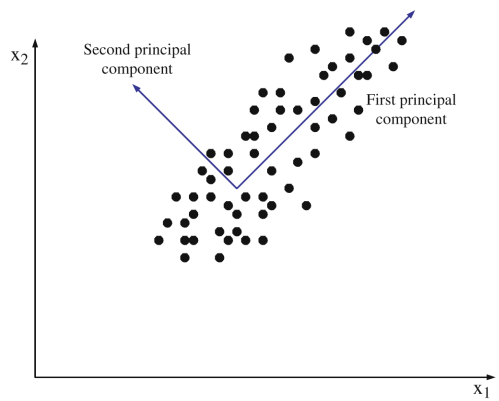
- Let $\mathbf{x}$ be a variable in R^d with covariance matrix $\mathbf{C_x}$.
- The goal is to find $\mathbf{y}$ as a linear combination of $\mathbf{x}$, $\mathbf{y} = \mathbf{w}^T.\mathbf{x}$, such that the components are pairwise uncorrelated and variances decrease from the first to the last, subject to $\| \mathbf{w} \| = 1$.
- the solution to this problem is given by the unitary eigenvectors of the $\mathbf{C}_X$ matrix. The order of the eigenvectors is such that the corresponding eigenvalues $\lambda_1, \cdots, \lambda_n$ satisfy $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$.
- The variance corresponding to variable $\mathbf{y}_i$ is given by the corresponding eigenvalues λ_i. Therefore the j-principal component explains $\frac{\lambda_j}{\sum_{i=1}^{n} \lambda_i}$ of the total variance.
- When keeping only the first q principal components, only $\frac{\sum_{i=1}^{q} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$ of the total variance of the data is preserved.

Empirical Learning

Principal component analysis PCA

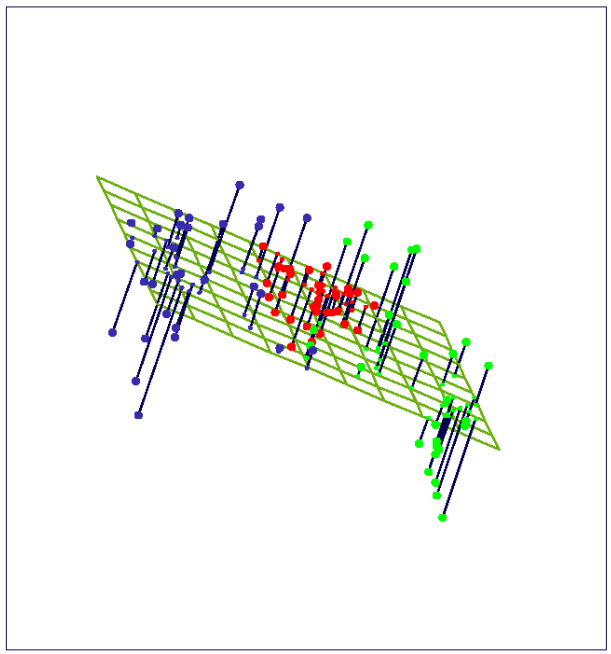
Example

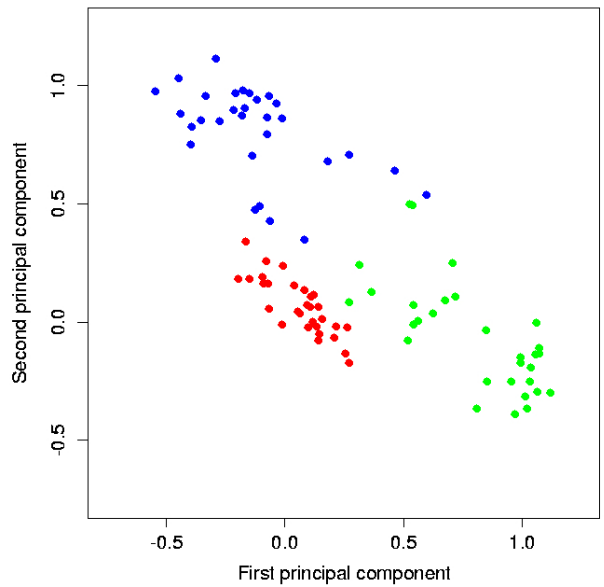
- $u_1^T u_1 = 1$
- the variance of the projected data is given by $\frac{1}{N}\sum_n \{u_1^T \mathbf{x}_n - u_1^T \bar{\mathbf{x}}\}^2 = u_1^T S u_1$, with $S = \frac{1}{N}\sum_n (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T$
- maximize the projected variance with respect to u_1, subject to $u_1^T u_1 = 1$. using the Lagrange multiplier technique, we maximize the unconstrained problem $u_1^T S u_1 + \lambda_1(1 - u_1^T u_1)$. This gives $S u_1 = \lambda_1 u_1$, with variance given by λ_1
- So the variance will be maximum when we set u_1 equal to the eigenvector having the largest eigenvalue λ_1



Empirical Learning

Principal Curves and surfaces

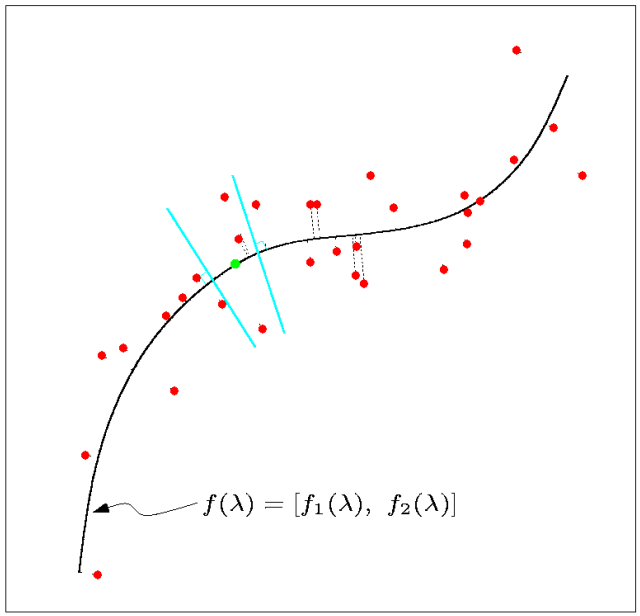




Empirical Learning

Principal Curves and surfaces

- Principal curves generalize the principal component line.
- provide a smooth one-dimensional curved approximation to a set of data points
- a principal surface provides a curved manifold approximation of dimension 2
- principal surfaces of dimension greater than two are rarely used, since the visualization aspect is less attractive, as is smoothing in high dimensions



Empirical Learning

Multidimensional scaling

- SOMs, PCA, principal curves and surfaces map data points in $\mathbf{R}^d$ to a lower dimension manifold
- Multidimensional scaling has a similar goal but approaches the problem in a different way:
  - starting from a set of observations $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbf{R}^d$ MDS seeks values $z_1, z_2, \ldots, z_N \in \mathbf{R}^k, k < d$ with the same pairwise distance values. The z values are find by minimizing

$$S_D(z_1, z_2, ..., z_N) = \sum_{i \neq i'} (||x_i - x_{i'}|| - ||z_i - z_{i'}||)^2$$

  - This is known as least square MDS or Kruskal-Shephard scaling
  - note that only the distance pairs of starting x-values are needed, not the data points x_i
  - A gradient descent algorithm is used to minimize S_D.

Empirical Learning

Multidimensional scaling

- Sammon mapping minimizes

$$S_D(z_1, z_2, ..., z_N) = \sum_{i \neq i'} \frac{(||x_i - x_{i'}|| - ||z_i - z_{i'}||)^2}{||x_i - x_{i'}||}$$

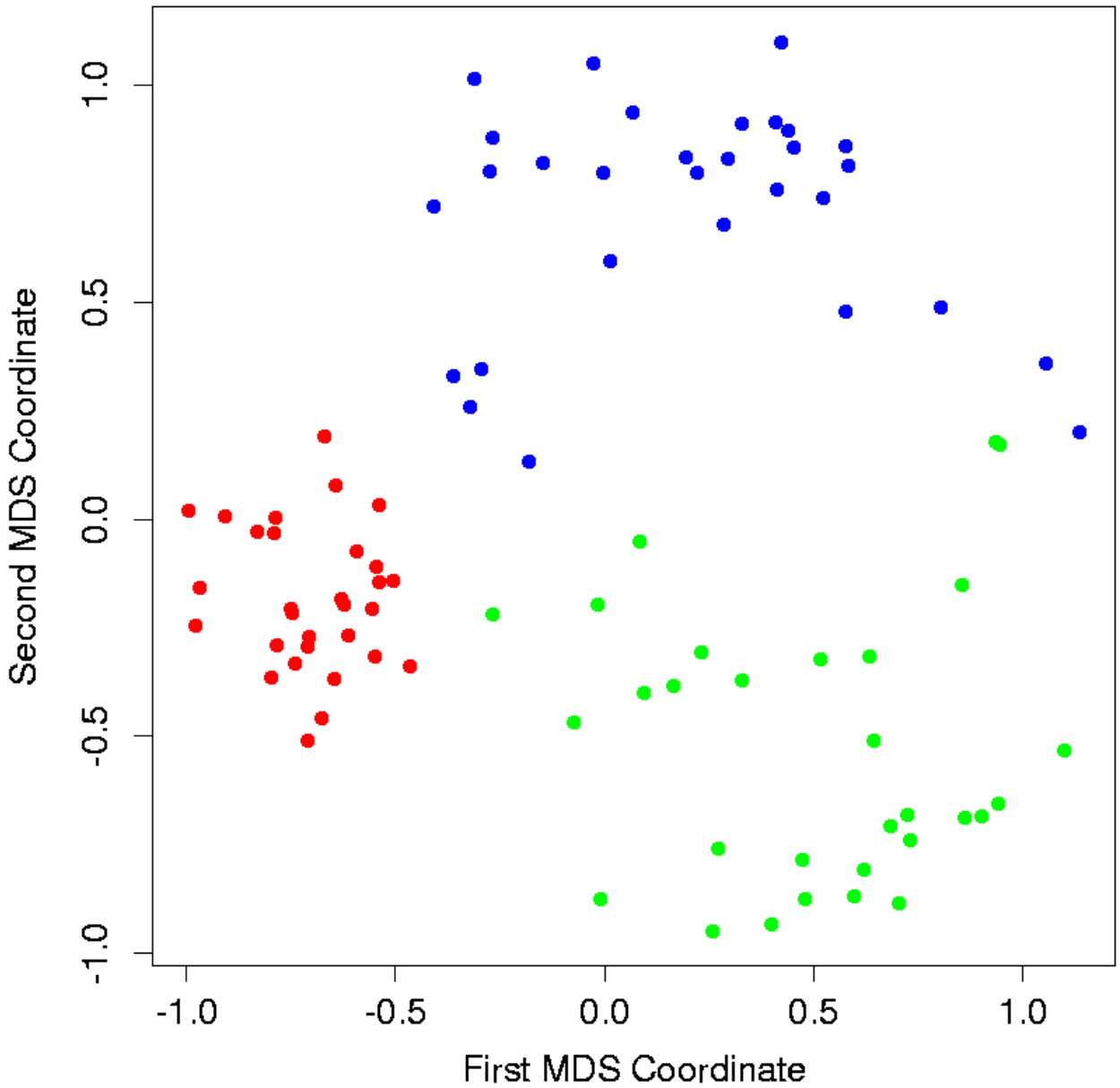
  Here more emphasis is put in preserving smaller pairwise distances

- classical scaling is based on similarities $s_{ii'}$. Often one uses centered inner product: $s_{ii'} =< x_i - \bar{x}, x_{i'} - \bar{x} >$. The problem is to minimize

$$\sum_{i \neq i'} (s_{ii'} - < z_i - \bar{z}, z_{i'} - \bar{z} >)^2$$

Empirical Learning

Multidimensional scaling



Empirical Learning

ISOMAP – isometric feature mapping

- Construct neighborhood graph
- Compute shortest paths
- Construct d-dimensional embedding

Key insight: For neighboring points, input space distance provides a good approximation to geodesic distance. For faraway points, geodesic distance can be approximated by adding up a sequence of "short hops" between neighboring points. These approximations are computed efficiently by finding shortest paths in a graph with edges connecting neighboring data points.

Empirical Learning

Locally Linear Embedding

- Compute the neighbors of each data point
- Compute the weights that best reconstruct each data point from its neighbors, minimizing the cost by constrained linear fits
- Compute the vectors best reconstructed by the weights minimizing a quadratic form

Key insight: The same weights that reconstruct the data point in D-dimensions should also reconstruct its embedded manifold coordinates in d-dimensions

References

Bishop

Pattern recognition and machine learning,

Book.

Trevor Hastie and Robert Tibshirani and Jerome Friedman

The elements of statistical learning,

Springer.

Mathworks

Matlab help,

????.