Clustering

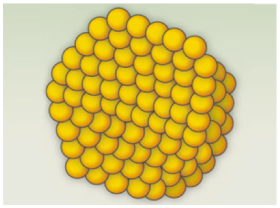
Ishwar K Sethi

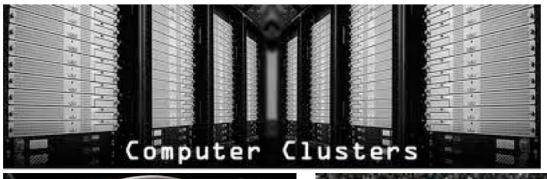
What is a Cluster?

clus·ter
/ˈkləstər/

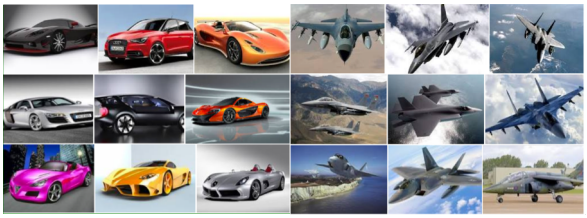
noun

1. a group of similar things or people positioned or occurring closely together.
   "clusters of creamy-white flowers"
   synonyms: bunch, clump, mass, knot, group, clutch, bundle, truss More





What is Clustering?



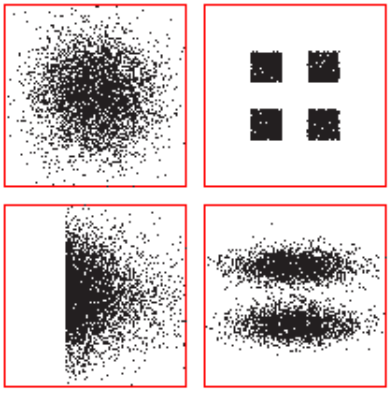
Clustering is the process of organizing objects into groups whose members are similar in some way.

Clustering is also a way of learning, for example being able to decide whether an object should be placed in group one or group two. Since there is no teacher to illustrate examples from different groups (no class labels), the learning in clustering is often called unsupervised learning.

Unsupervised Learning

- Unsupervised learning refers to situations where we have a collection of data but each data instance is unlabeled/unmarked. In such situations, the best we can do is to organize data into groups for further analysis
- The process of grouping data is known as *Clustering*. Clustering has wide applications and is known through a variety of names - *unsupervised classification, Q analysis, typology, numerical taxonomy, and market segmentation*

Why Clustering?



These four data sets have identical first-order and second-order statistics. We need to find other ways of modeling their structure. Clustering is an alternative way of describing the data in terms of groups of patterns.

Clustering Example

Market segmentation to group customers into different groups

Clustering Example

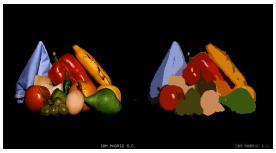
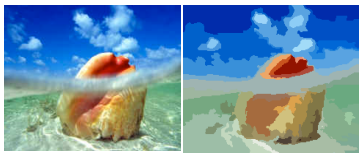


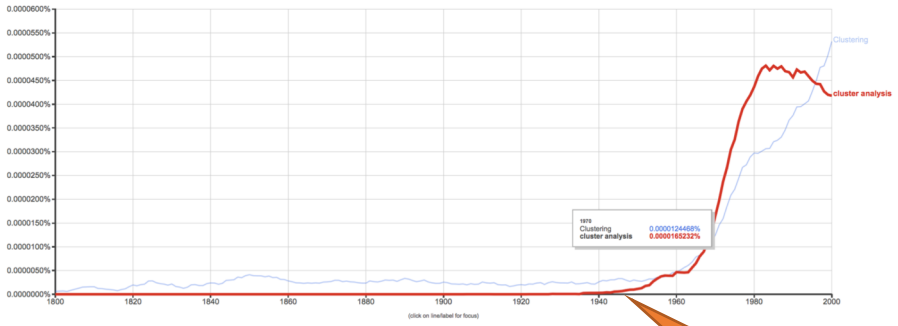
Image segmentation to group pixels into meaningful regions



Clustering Example

Clustering news stories

History of Clustering



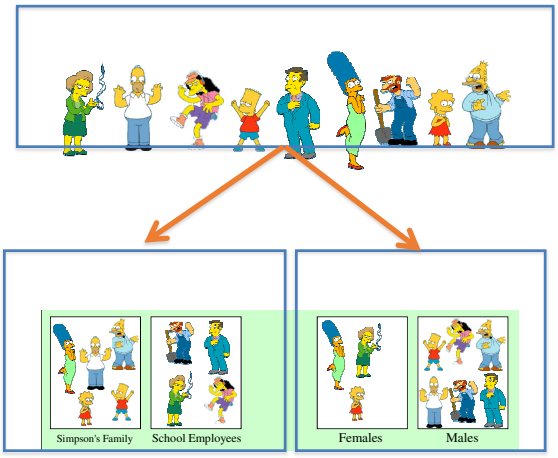
Google Books Ngram Viewer

Beginning of Clustering

Books.google.com/ngrams

How many clusters?

Multiple Clustering Results are Possible



Results must be validated against knowledge about the application domain.

Similarity Measures

The basis of clustering lies in measuring similarity between a pair of objects

- A general class of metrics for d-dimensional patterns is the *Minkowski metric*

$$L_p(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|^p\right)^{1/p}$$

  also referred to as the L_p *norm*.

- The *Euclidean distance* is the L_2 norm

$$L_2(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|^2\right)^{1/2}$$

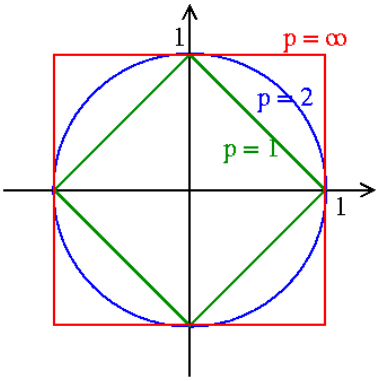
- The *Manhattan* or *city block distance* is the L_1 norm

$$L_1(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|$$

- The L_∞ norm is the maximum of the distances along individual coordinate axes

$$L_\infty(\mathbf{x},\mathbf{y}) = \max_{i=1}^{d} |\mathbf{x}_i - \mathbf{y}_i|$$

Minkowski Metric: Contours of Constant Distance

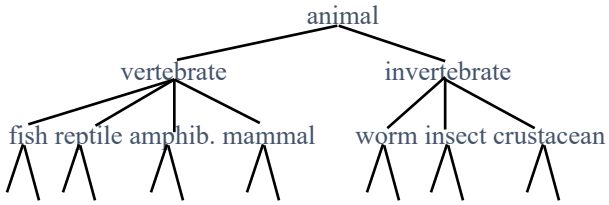


Taxonomy of Clustering Methods

- Hierarchical
  - Agglomerative
  - Divisive
- Partitional
  - Sequential or simultaneous procedures
  - Direct or indirect methods

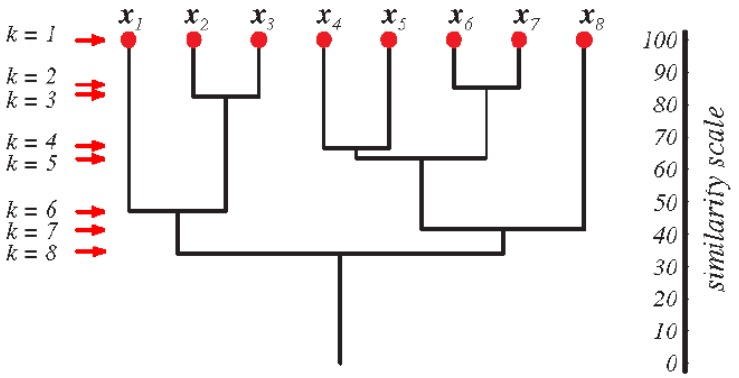
Hierarchical Clustering

- Builds a tree-based hierarchical taxonomy (*dendrogram*) from a set of examples.



- Recursive application of a standard clustering algorithm can produce hierarchical clustering.

Dendrogram



Agglomerative Vs Divisive Clustering

- *Agglomerative* (*bottom-up*) methods start with each example in its own cluster and iteratively combine them to form larger and larger clusters.
- *Divisive* (*partitional, top-down*) separates all examples immediately into clusters.

Direct Clustering Methods

- *Direct clustering* methods require a specification of the number of clusters, *k*, desired.
  - A *clustering evaluation function* assigns a real-value quality measure to a clustering.
  - The number of clusters can be determined automatically by explicitly generating clustering for multiple values of *k* and choosing the best result according to a clustering evaluation function.

How Many Clusters?

- Statistical significance of differences between clusters
- Cluster sizes
- Meaningful cluster profiles
- Aggregation or decomposition patterns of clusters at different stages of clustering

Hierarchical Agglomerative Clustering

- Assumes a *similarity function* for determining the similarity of two instances.
- Starts with all instances in separate clusters and then repeatedly joins the two clusters that are most similar until there is only one cluster.
- The history of merging forms a binary tree or hierarchy.

Cluster Similarity

- How to compute similarity of two clusters each possibly containing multiple instances?
  - Single Link: Similarity of two most similar members.
  - Complete Link: Similarity of two least similar members.
  - Group Average: Average similarity between members.

Cluster Similarity

Popular distance measures (for two clusters $\mathcal{D}_i$ and $\mathcal{D}_j$):

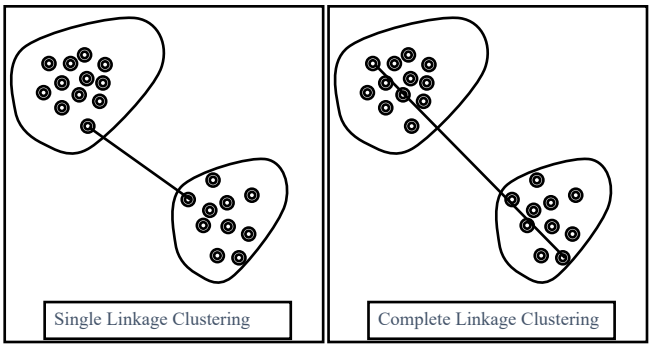
$$d_{\text{min}}(\mathcal{D}_i, \mathcal{D}_j) = \min_{\substack{\mathbf{x}\in\mathcal{D}_i \\ \mathbf{x}'\in\mathcal{D}_j}} \|\mathbf{x} - \mathbf{x}'\|$$

$$d_{\text{max}}(\mathcal{D}_i, \mathcal{D}_j) = \max_{\substack{\mathbf{x}\in\mathcal{D}_i \\ \mathbf{x}'\in\mathcal{D}_j}} \|\mathbf{x} - \mathbf{x}'\|$$

$$d_{\text{avg}}(\mathcal{D}_i, \mathcal{D}_j) = \frac{1}{\#\mathcal{D}_i \, \#\mathcal{D}_j} \sum_{\mathbf{x}\in\mathcal{D}_i} \sum_{\mathbf{x}'\in\mathcal{D}_j} \|\mathbf{x} - \mathbf{x}'\|$$

$$d_{\text{mean}}(\mathcal{D}_i, \mathcal{D}_j) = \|\mathbf{m}_i - \mathbf{m}_j\|$$

Popular Agglomerative Clustering Procedures



a.k.a. nearest neighbor clustering

a.k.a. furthest neighbor clustering

Hierarchical Clustering Example

Let us consider five examples: A, B, C, D, and E. Let the interpoint distances between these examples be given by the following distance matrix.

$$\mathbf{DM}(1) = \begin{array}{c|ccccc} & A & B & C & D & E \\ A & 0 & & & & \\ B & 1 & 0 & & & \\ C & 5 & 3 & 0 & & \\ D & 6 & 8 & 4 & 0 & \\ E & 8 & 7 & 6 & 2 & 0 \end{array}$$

Hierarchical Clustering Example

Using the *nearest neighbor* measure, also known as the *single linkage* measure, we merge A and B to form a cluster, since they are closest. Next we compute the distances between this cluster and the remaining examples. We can get these distances from the above distance matrix. The values for these are as follows:

$d_{(AB)C} = min\{d_{AC}, d_{BC}\} = d_{BC} = 3$

$d_{(AB)D} = min\{d_{AD}, d_{BD}\} = d_{AD} = 6$

$d_{(AB)E} = min\{d_{AE}, d_{BE}\} = d_{BE} = 7$

At this point we can form an updated distance matrix. This is given as:

$$\mathbf{DM}(2) = \begin{array}{c|cccc} & AB & C & D & E \\ AB & 0 & & & \\ C & 3 & 0 & & \\ D & 6 & 4 & 0 & \\ E & 7 & 6 & 2 & 0 \end{array}$$

Hierarchical Clustering Example

Since the smallest entry in above distance matrix is 2, examples D and E are merged to form another cluster. At this point, we repeat the distance calculations to obtain the following set of values:

$d_{(AB)C} = 3$

$d_{(AB)(DE)} = min\{d_{AD}, d_{AE}, d_{BD}, d_{BE}\} = d_{AD} = 6$

$d_{(DE)C} = min\{d_{CD}, d_{CE}\} = d_{CD} = 4$

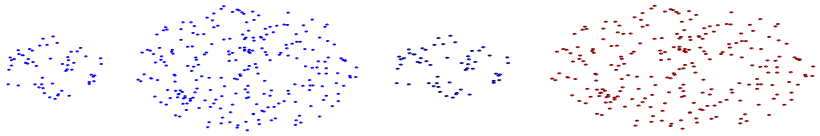
The new distance matrix thus becomes

$$\mathbf{DM}(3) = \begin{array}{c|ccc} & AB & C & DE \\ AB & 0 & & \\ C & 3 & 0 & \\ DE & 6 & 4 & 0 \end{array}$$

This matrix indicates that C should be merged with A and B. At this stage we have only two clusters left that are joined to form a single cluster of five examples.

Single Linkage Behavior

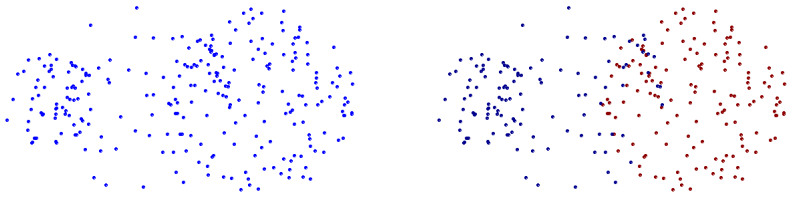
Can handle non-elliptical shapes



Original Points

Result (Two Clusters)

Single Linkage Behavior

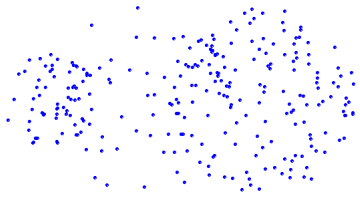


Original Points

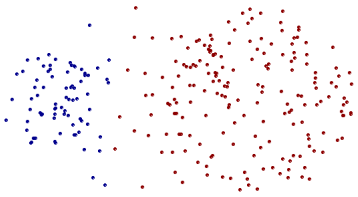
Resulting Two Clusters

Sensitive to noise and outliers

Complete Linkage Behavior



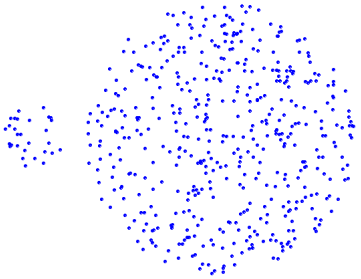
Original Points



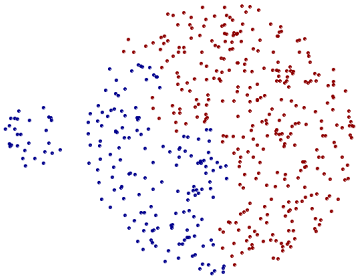
Resulting Two Clusters

Less sensitive to noise and outliers

Complete Linkage Behavior



Original Points



Two Clusters

Tends to break large clusters

How to Choose the Number of Clusters in Hierarchical Clustering

- Lifetime Method
  - The *lifetime* of a cluster is defined as the absolute value of the difference between the dendrogram level at which it is created and the level at which it is absorbed into a larger cluster. Using lifetime as a criterion, a user can search for cluster that have a large lifetime.
- Self-similarity Measure Method
  - This method uses a function *h(C)* that measures the dissimilarity between the vectors of the same cluster *C*. A cutoff value for the selected measure can be used to control the number of clusters.
  - Examples of possible functions are

    *h(C)= max{d(***x***,* **y***), For all* **x**, **y** *from cluster C}*

    *h(C)= med{d(***x***,* **y***), For all* **x**, **y** *from cluster C}*

Direct Clustering : K-Means Clustering

- Assumes instances are real-valued vectors.
- Clusters based on *centroids, center of gravity*, or mean of points in a cluster, *c*:

$$\vec{\mu}(c) = \frac{1}{|c|}\sum_{\vec{x}\in c}\vec{x}$$

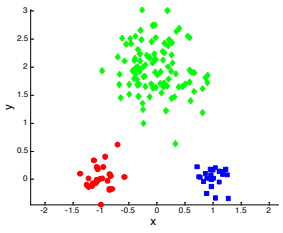
- Reassignment of instances to clusters is based on distance to the current cluster centroids.

What can you do if attributes are not real valued?

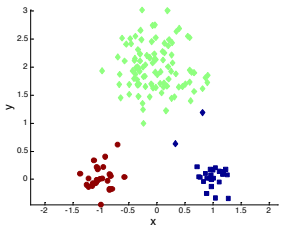
K-Means Clustering : Seeds Choice

- Results can vary based on random seed selection.
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clustering.
- Select good seeds using a heuristic or the results of another method.

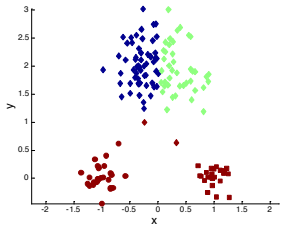
K-Means Clustering Can Yield Different Results



Original Points

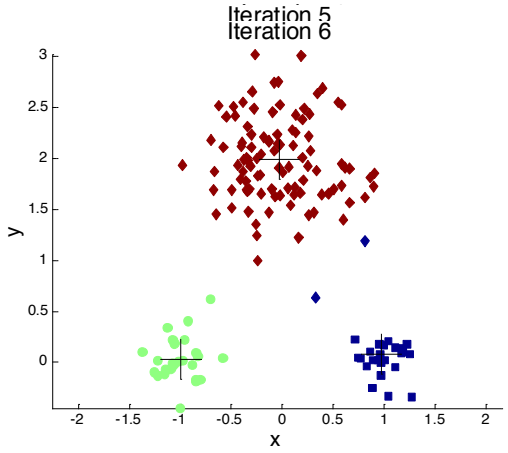


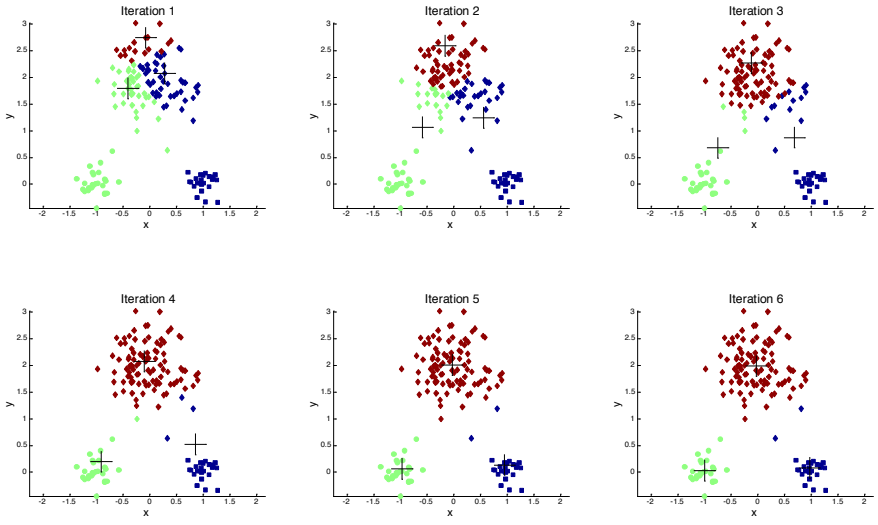
Optimal Clustering



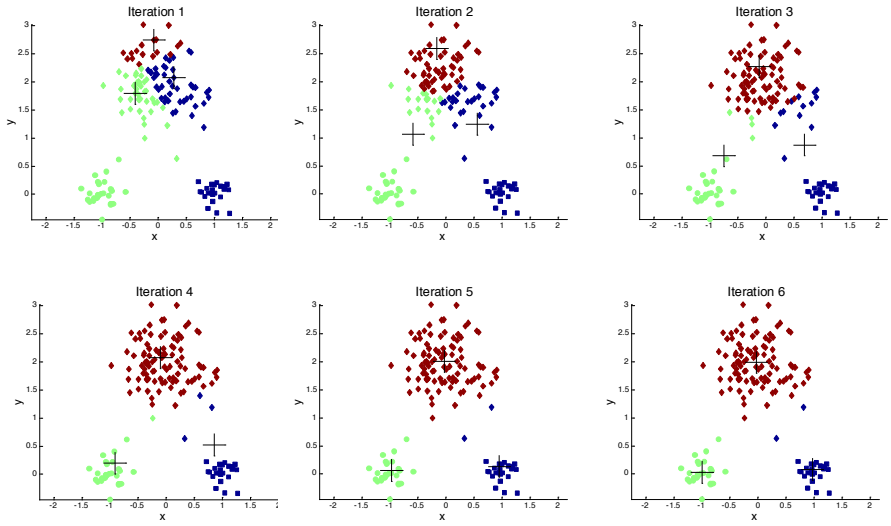
Sub-optimal Clustering

Importance of Choosing Initial Centroids





Results with Different Seed Points



Evaluating K-Means Clusters

- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

  - Given two clusters, we can choose the one with the smallest error
  - One easy way to reduce SSE is to increase K, the number of clusters. A good clustering with smaller K can have a lower SSE than a poor clustering with higher K

K-means Illustration

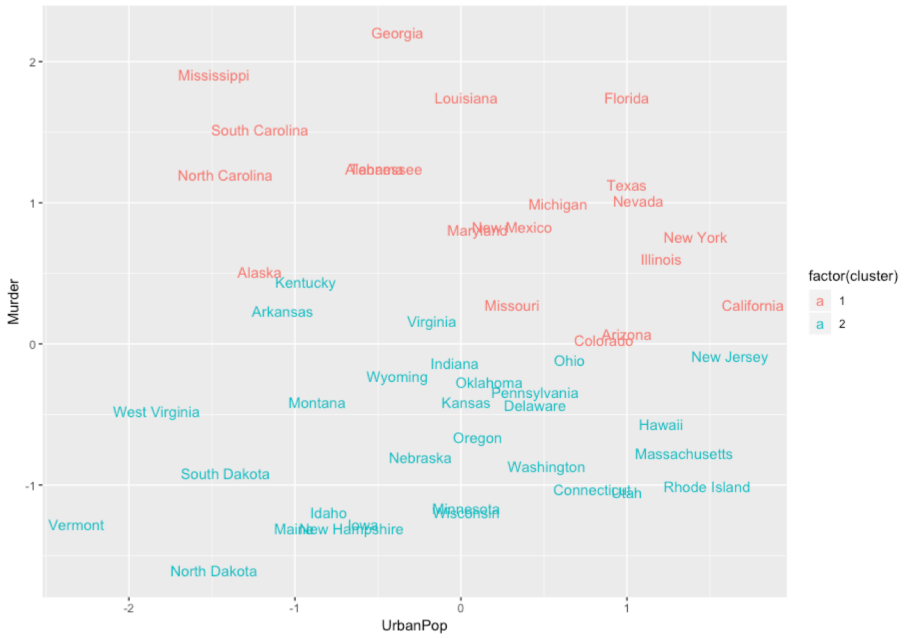
- USArrests data containing arrest information for 50 states. The information is arrest rates for three crimes per 100,000 residents. The other attribute is urban population percentage
- After scaling/normalization the data looks like as shown below

	Murder	Assault	UrbanPop	Rape
Alabama	1.24256408	0.7828393	-0.5209066	-0.003416473
Alaska	0.50786248	1.1068225	-1.2117642	2.484202941
Arizona	0.07163341	1.4788032	0.9989801	1.042878388
Arkansas	0.23234938	0.2308680	-1.0735927	-0.184916602
California	0.27826823	1.2628144	1.7589234	2.067820292
Colorado	0.02571456	0.3988593	0.8608085	1.864967207

Q. Why are some rates negative?

nstart indicates the number of initial configurations to be used

```
> k2 <- kmeans(df, centers = 2, nstart = 25)
> str(k2)
List of 9
 $ cluster     : Named int [1:50] 1 1 1 2 1 1 2 2 1 1 ...
  ..- attr(*, "names")= chr [1:50] "Alabama" "Alaska" "Arizona" "Arkansas" ...
 $ centers     : num [1:2, 1:4] 1.005 -0.67 1.014 -0.676 0.198 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:2] "1" "2"
  .. ..$ : chr [1:4] "Murder" "Assault" "UrbanPop" "Rape"
 $ totss       : num 196
 $ withinss    : num [1:2] 46.7 56.1
 $ tot.withinss: num 103
 $ betweenss   : num 93.1
 $ size        : int [1:2] 20 30
 $ iter        : int 1
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```



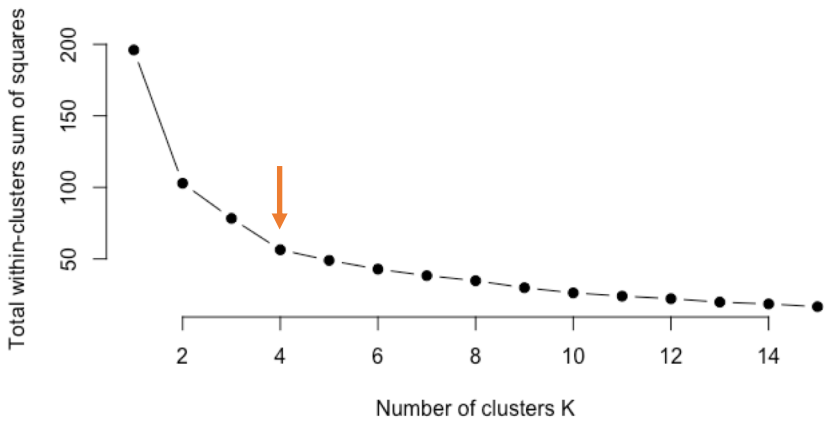

```
K-means clustering with 2 clusters of sizes 20, 30

Cluster means:
     Murder    Assault   UrbanPop       Rape
1  1.004934  1.0138274  0.1975853  0.8469650
2 -0.669956 -0.6758849 -0.1317235 -0.5646433

Clustering vector:
       Alabama         Alaska        Arizona       Arkansas     California
             1              1              1              2              1
      Colorado    Connecticut       Delaware        Florida        Georgia
             1              2              2              1              1
        Hawaii          Idaho       Illinois        Indiana           Iowa
             2              2              1              2              2
        Kansas       Kentucky      Louisiana          Maine       Maryland
             2              2              1              2              1
 Massachusetts       Michigan      Minnesota    Mississippi       Missouri
             2              1              2              1              1
       Montana       Nebraska         Nevada  New Hampshire     New Jersey
             2              2              1              2              2
    New Mexico       New York North Carolina   North Dakota           Ohio
             1              1              1              2              2
      Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
             2              2              2              2              1
  South Dakota      Tennessee          Texas           Utah        Vermont
             2              1              1              2              2
      Virginia     Washington  West Virginia      Wisconsin        Wyoming
             2              2              2              2              2

Within cluster sum of squares by cluster:
[1] 46.74796 56.11445
 (between_SS / total_SS =  47.5 %)
```

How to Choose the Correct Number of Clusters



A Hybrid Algorithm to Tackle Seed Points Selection

- Combines HAC (Hierarchical Agglomerative Clustering) and K-Means clustering.
- First randomly take a sample of instances of size $\sqrt{n}$
- Run group-average HAC on this sample, which takes only O(n) time.
- Use the results of HAC as initial seeds for K-means.
- Overall algorithm is O(n) and avoids problems of bad seed selection.

Indirect Clustering Through Iterative Optimization

- This approach works best when an initial clustering solution can be obtained through other means
- We use the Sum of Squared (SSE) criterion; other measures are possible with suitable changes in the algorithm.
- SSE Measure

$$J_e = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \mathbf{m}_i\|^2 = \sum_{i=1}^{k} J_i .$$

Suppose we take an example $\hat{\mathbf{x}}$ from cluster i and contemplate on moving it cluster j. This move implies that means of clusters i and j would change. The new means would be given as

$$\mathbf{m}_i^{new} = \mathbf{m}_i^{old} - \frac{\hat{\mathbf{x}} - \mathbf{m}_i^{old}}{n_{iold} - 1}, \text{ and}$$

As a result of adding an extra example to cluster j, its error measure would increase. Similarly, the error measure for cluster i would decrease because it would now have one less example. It is not hard to show that the new and old values of error measures for the two clusters are related by the following equations:

$$J_{inew} = J_{iold} - \frac{n_{iold}}{n_{iold} - 1} \|\hat{\mathbf{x}} - \mathbf{m}_i\|^2$$

$$J_{jnew} = J_{jold} + \frac{n_{jold}}{n_{jold} + 1} \|\hat{\mathbf{x}} - \mathbf{m}_j\|^2 .$$

We should move $\hat{\mathbf{x}}$ from cluster i to cluster j only when there is a decrease in the value of the SSE criterion function. Thus, we should move $\hat{\mathbf{x}}$ from cluster i to cluster j only when

$$\frac{n_{iold}}{n_{iold} - 1} \|\hat{\mathbf{x}} - \mathbf{m}_i\|^2 > \frac{n_{jold}}{n_{jold} + 1} \|\hat{\mathbf{x}} - \mathbf{m}_j\|^2 .$$

Iterative Optimization Steps

Based on above, we can perform indirect clustering through the following steps:

1. Obtain an initial partition of the n examples into k clusters and compute each cluster mean.
2. Select a candidate example for move from one of the clusters and check if it is profitable to move it into another cluster. Move and update cluster means.
3. Repeat Step 2 with another example until no more moves are profitable.

Cluster Validity

- How do we validate our clustering result?
  - Methods for validating the results of a clustering algorithm include:
    - Repeating the clustering procedure for different values of the parameters, and examining the resulting values of the criterion function for large jumps or stable ranges.
    - Evaluating the goodness-of-fit using measures such as the chi-squared or Kolmogorov-Smirnov statistics.
    - Formulating hypothesis tests that check whether multiple clusters found have been formed by chance, and whether the observed change in the error criterion has any significance.

Cluster Validity

- The groupings by the unsupervised clustering can also be compared to the known labels if ground truth is available.
- In an optimal result, the patterns with the same class labels in the ground truth must be assigned to the same cluster and the patterns corresponding to different classes must appear in different clusters.
- The following measures quantify how well the results of the unsupervised clustering algorithm reflect the groupings in the ground truth:
  - Entropy,
  - Rand index.

Cluster Validity

- *Entropy* is an information theoretic criterion that measures the homogeneity of the distribution of the clusters with respect to different classes.
- Given K as the number of clusters resulting from the clustering algorithm and C as the number of classes in the ground truth, let
  - h_{ck} denote the number of patterns assigned to cluster k with a ground truth class label c.
  - $h_{c.} = \sum_{k=1}^{K} h_{ck}$ denote the number of patterns with a ground truth class label c.
  - $h_{.k} = \sum_{c=1}^{C} h_{ck}$ denote the number of patterns assigned to cluster k.

Cluster Validity

- The quality of individual clusters is measured in terms of the homogeneity of the class labels within each cluster.
- For each cluster k, the cluster entropy E_k is given by

$$E_k = -\sum_{c=1}^{C} \frac{h_{ck}}{h_{.k}} \log \frac{h_{ck}}{h_{.k}}.$$

- Then, the overall cluster entropy E_{cluster} is given by a weighted sum of individual cluster entropies as

$$E_{\text{cluster}} = \frac{1}{\sum_{k=1}^{K} h_{.k}} \sum_{k=1}^{K} h_{.k} E_k.$$

Cluster Validity

- A smaller cluster entropy value indicates a higher homogeneity.
- However, the cluster entropy continues to decrease as the number of clusters increases.
- To overcome this problem, another entropy criterion that measures how patterns of the same class are distributed among the clusters can be defined.

Cluster Validity

- For each class c, the class entropy E_c is given by

$$E_c = -\sum_{k=1}^{K} \frac{h_{ck}}{h_{c.}} \log \frac{h_{ck}}{h_{c.}}.$$

- Then, the overall class entropy E_{class} is given by a weighted sum of individual class entropies as

$$E_{\text{class}} = \frac{1}{\sum_{c=1}^{C} h_{c.}} \sum_{c=1}^{C} h_{c.} E_c.$$

Cluster Validity

- Unlike the cluster entropy, the class entropy increases when the number of clusters increases.
- Therefore, the two measures can be combined for an overall entropy measure as

$$E = \beta E_{\text{cluster}} + (1 - \beta) E_{\text{class}}$$

where $\beta \in [0, 1]$ is a weight that balances the two measures.

Cluster Validity

- The *Rand index* can also be used to measure the agreement of every pair of patterns according to both unsupervised and ground truth labelings.
- The agreement occurs if
  - two patterns that belong to the same class are put into the same cluster, or
  - two patterns that belong to different classes are put into different clusters.
- The Rand index is computed as the proportion of all pattern pairs that agree in their labels.
- The index has a value between 0 and 1, where 0 indicates that the two labelings do not agree on any pair of patterns and 1 indicates that the two labelings are exactly the same.

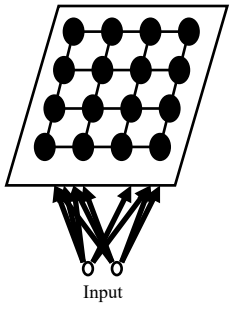
Clustering for Large Data Sets

- Desirable attributes of a clustering procedure for very large data sets
  - No more than one pass (scan) of the database
  - Should allow for incremental updates of results as more data becomes available
  - Work with limited main memory

K-Means Clustering for Large Data Sets

- Choose a random sample of data and perform k-means. Return the k-means and the quality of cluster measure
- Choose another random sample and repeat
- Compare the new result with the old and save the better of the two results
- Repeat above predetermined number of times

Self-Organizing Feature Map

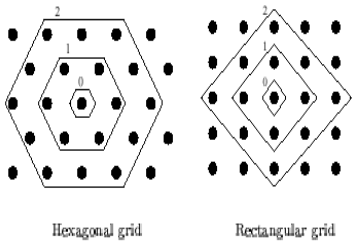


What is SOM ?

- SOM is a neural network-based method for unsupervised clustering.
- SOM maps high-dimensional data on a low dimensional (usually two-dimension) grid in such a way that similar high-dimensional data points are mapped to same or neighboring neurons.

Basic SOM

- The basic SOM consists of M neurons located on a regular low-dimensional grid, usually 1- or 2-dimensional.



SOM Algorithm

- Initialization: to construct the map, each neuron in the grid is initialized with small random weights, that is each neuron *i* is initialized as a *d*-dimensional prototype,

$$m_i = [m_{i1}, \ldots, m_{id}]$$

Training…

- Randomly select an input data vector ***x*** and apply it to all neurons;
- Distances between ***x*** and all the prototype vectors are computed to find BMU (Best-matching unit), denoted here by *b:*

$$\|x - m_b\| = \min_i \{\|x - m_i\|\}$$

SOM Algorithm

Next, the prototypes are updated:

$$m_i(t+1) = m_i(t) + \alpha(t) h_{bi}(t)[x - m_i(t)]$$

Where *t* denotes time, $\alpha(t)$ is learning rate and

$$h_{bi}(t) = e^{-\frac{\|r_b - r_i\|^2}{2\sigma^2(t)}}$$

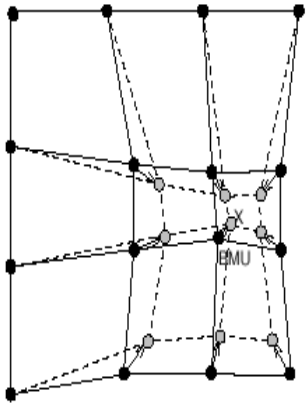
is a neighborhood kernel centered on the winner unit.

The r_i and r_b are positions of neurons *i* and *b* on the SOM gird; $\sigma(t)$ is neighborhood radius.

During training…

- The SOM behaves like a flexible net that folds onto the 'cloud' formed by the training data.
- Because of the neighborhood relations, neighboring prototypes are pulled the same direction, and thus prototype vectors of neighboring units resemble each other.

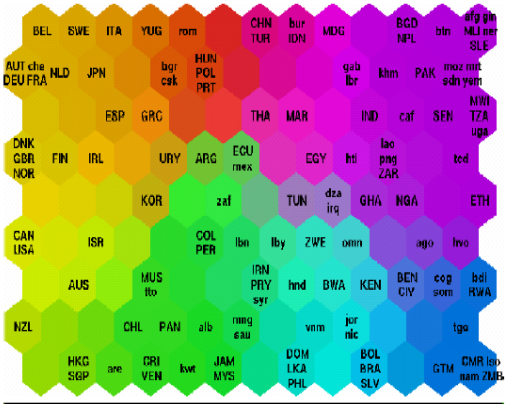
Map Behavior



SOM Contd.

- The prototype vectors are iteratively adjusted to correspond to the training data, and neighborhood relations are used in such a manner that neighboring prototype vectors become similar to each other.

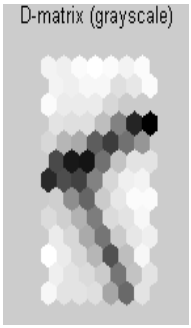
Visualization via SOFM: World Poverty Mapping

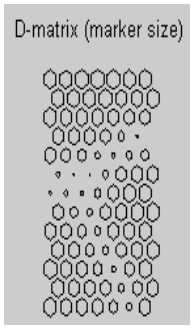


Based on 39 attributes to describe each country

Visualization of Cluster Structures with SOM

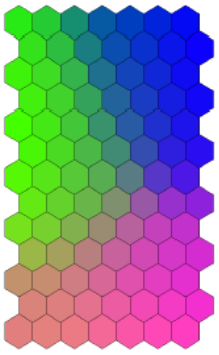
- U Matrix: a matrix of distances between neighboring map units.





Visualization of Cluster Structures

- Colormaps: Give similar colors to similar map units.

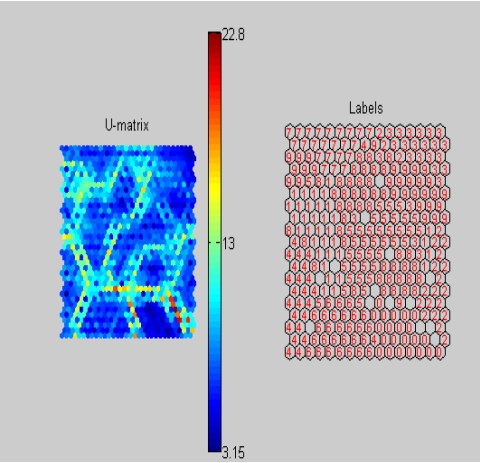


Similarity coloring

A simple example…

- Title of Database: Optical Recognition of Handwritten Digits 0…9
- Training data: 3823; Testing data: 1797
- Number of attributes of each data vector is 64

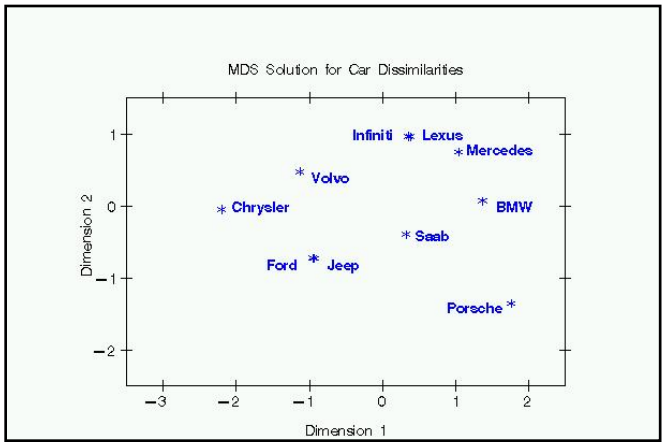
Trained Map



SOM as a Classifier

- Compute the distances between input data and each prototype vectors;
- The class the closest prototype vectors represent is what we looking for;
- Final classification accuracy is 92.37%

Multidimensional Scaling (MDS)



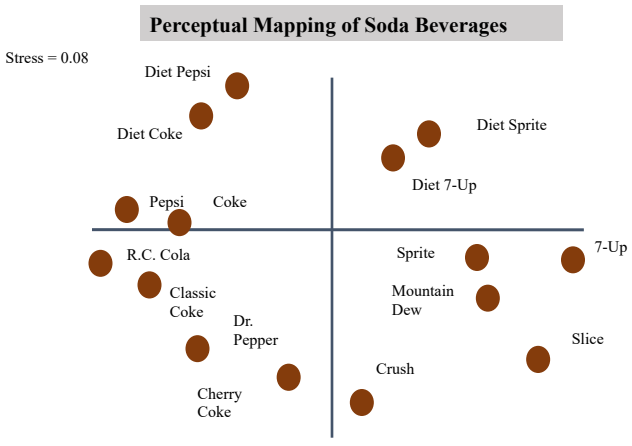
What is MDS?

- Multidimensional Scaling (MDS) is a data analysis method for mapping a group of points in a low (2-3) dimensional space while preserving as well as possible inter-point distances between the points in the original high dimensional space.
- MDS can aid data understanding/clustering results via visualization

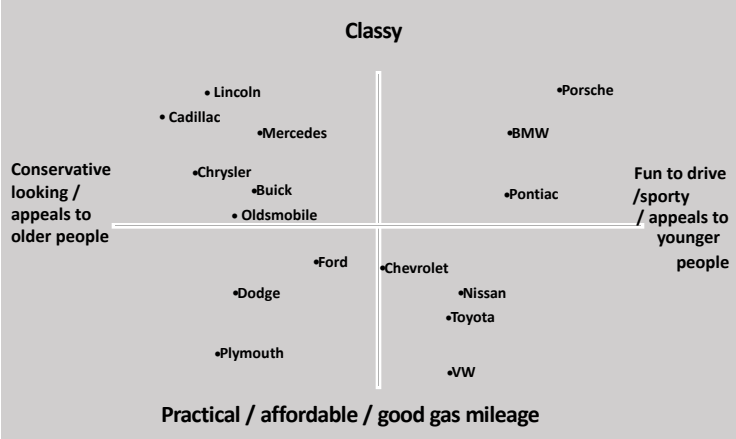
MDS Applications

- Originally developed as a tool to uncover hidden structure to account for perceived similarities among a group of objects or items.
- MDS is also known as perceptual mapping
- Widely used in marketing and product placement (Analyzing surveys)

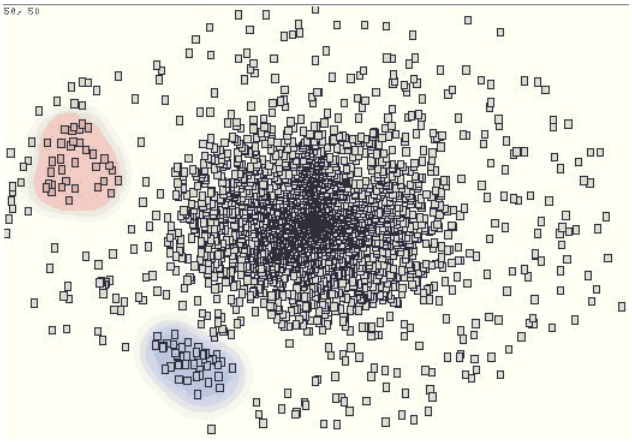
Perceptual Mapping of Soda Beverages



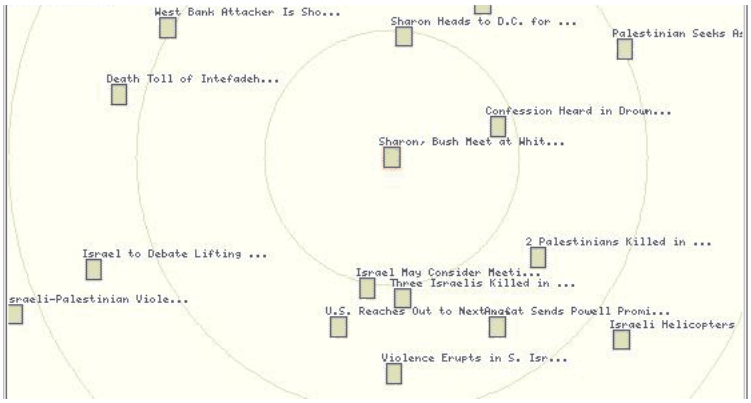
Another Perceptual Map Example



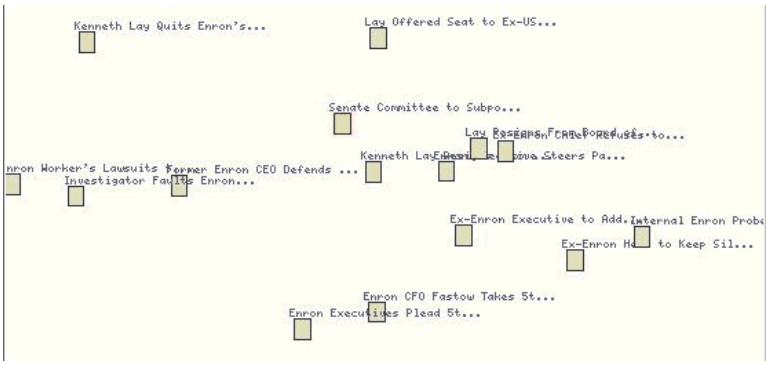
MDS Map of AP Wire Stories



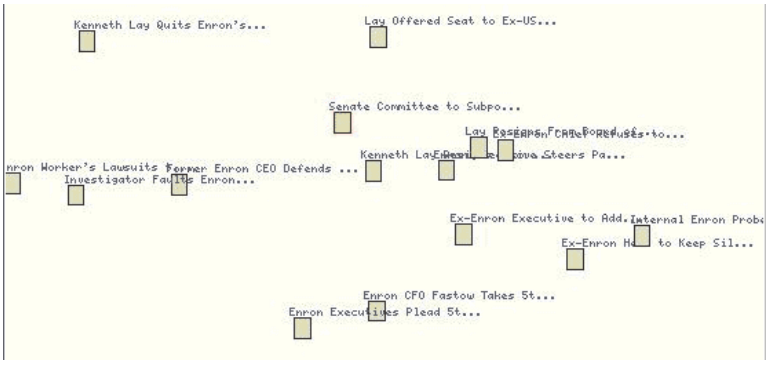
Close Up of Blue Cluster



Close Up of Red Cluster



Close Up of Red Cluster



Mathematical Representation of MDS

- Let $\delta_{ij} = Dist(\boldsymbol{X}_i, \boldsymbol{X}_j)$, and
  $d_{ij} = Dist(\boldsymbol{Y}_i, \boldsymbol{Y}_j)$
- A measure for mapping, called stress, can be

$$J_1 = \frac{1}{\sum_{i<j} \delta_{ij}^2} \sum_{i<j} (d_{ij} - \delta_{ij})^2$$

This measure emphasizes large errors. An optimization procedure like the gradient search is used to obtain the result

Mathematical Representation of MDS

- Another possible measure is the following. It emphasizes large fractional errors

$$J_2 = \sum_{i<j} \left(\frac{d_{ij} - \delta_{ij}}{\delta_{ij}}\right)^2$$

Mathematical Representation of MDS

- Yet another is as follows which is a compromise between the previous two measures

$$J_3 = \frac{1}{\sum_{i<j} \delta_{ij}} \sum_{i<j} \left(\frac{d_{ij} - \delta_{ij}}{\delta_{ij}}\right)^2$$

- The mapping performed by any of these three measures is known as metric MDS

Non-Metric MDS

- Non-metric MDS is more useful. It tries to preserve the order or rank of similarities. Let the ranked similarities be

  $$\delta_{i_1 j_1} \le \Delta \le \delta_{i_m j_m}; m = n(n-1)/2$$

- Then the mapping is obtained by finding *m* numbers that satisfy the following constraint

  $$d_{i_1 j_1} \le D \le d_{i_m j_m}$$

  and minimize the function:

$$J = \min \frac{1}{\sum_{i<j} d_{ij}^2} \sum_{i<j} (d_{ij} - d_{ij})^2$$

The above measure is invariant to translation, rotation, and dilation of the point configuration in the original space.