Lecture 15: Clustering

Statistical Learning and Data Mining

Xingye Qiao
Department of Mathematical Sciences
Binghamton University

E-mail: qiao@math.binghamton.edu

Read: ELSII Ch. 14.3, and ISLR 10.3

Outline

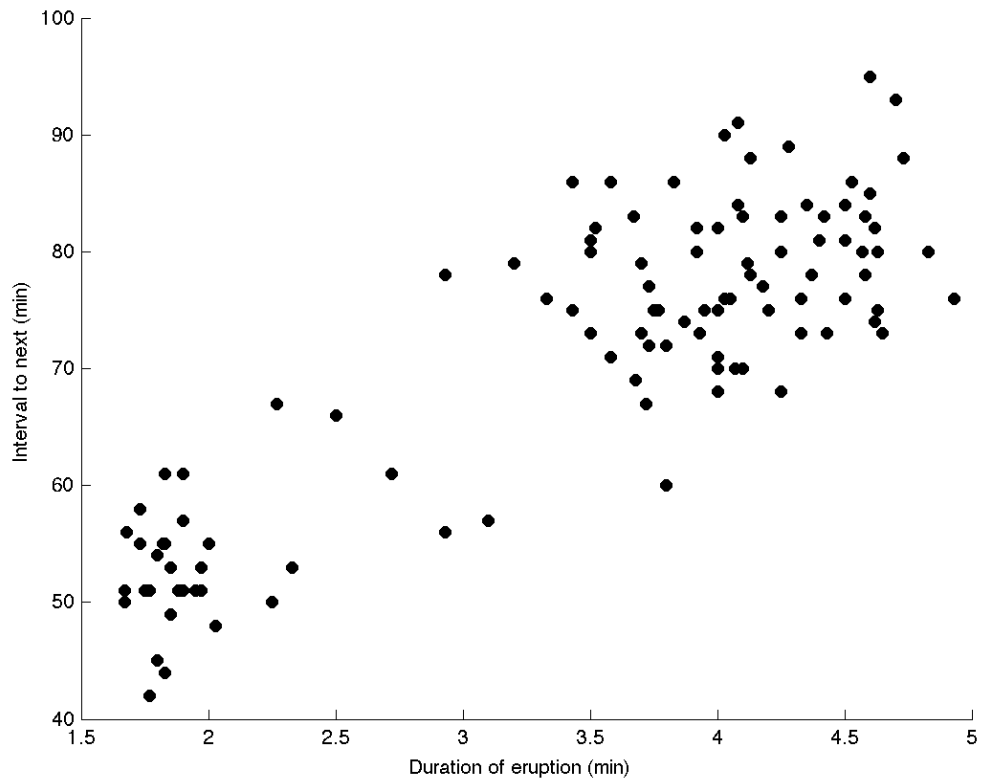
Clustering

- Divide the data $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ into groups
- Groups consist of similar objects (observations)
- Contrast to classification (discrimination)
  1. Classification has predetermined (predefined) classes (supervised, label Y is available)
  2. Clustering is to determine unknown classes (unsupervised)
- "Unsupervised learning": data segmentation, class discovery–examples include
  1. Marketers use demographics and consumer profiles to segment the marketplace into small, homogeneous groups
  2. Physicians use medical records to cluster patients for personalized treatment
  3. `Pandora` and `Netflix` use viewing history to group viewers/listeners to recommend next songs and movies
- Challenging:
  - What is a meaningful cluster?
  - How do we validate clustering results?


© Ron Niebrugge / WildNatureImages.com

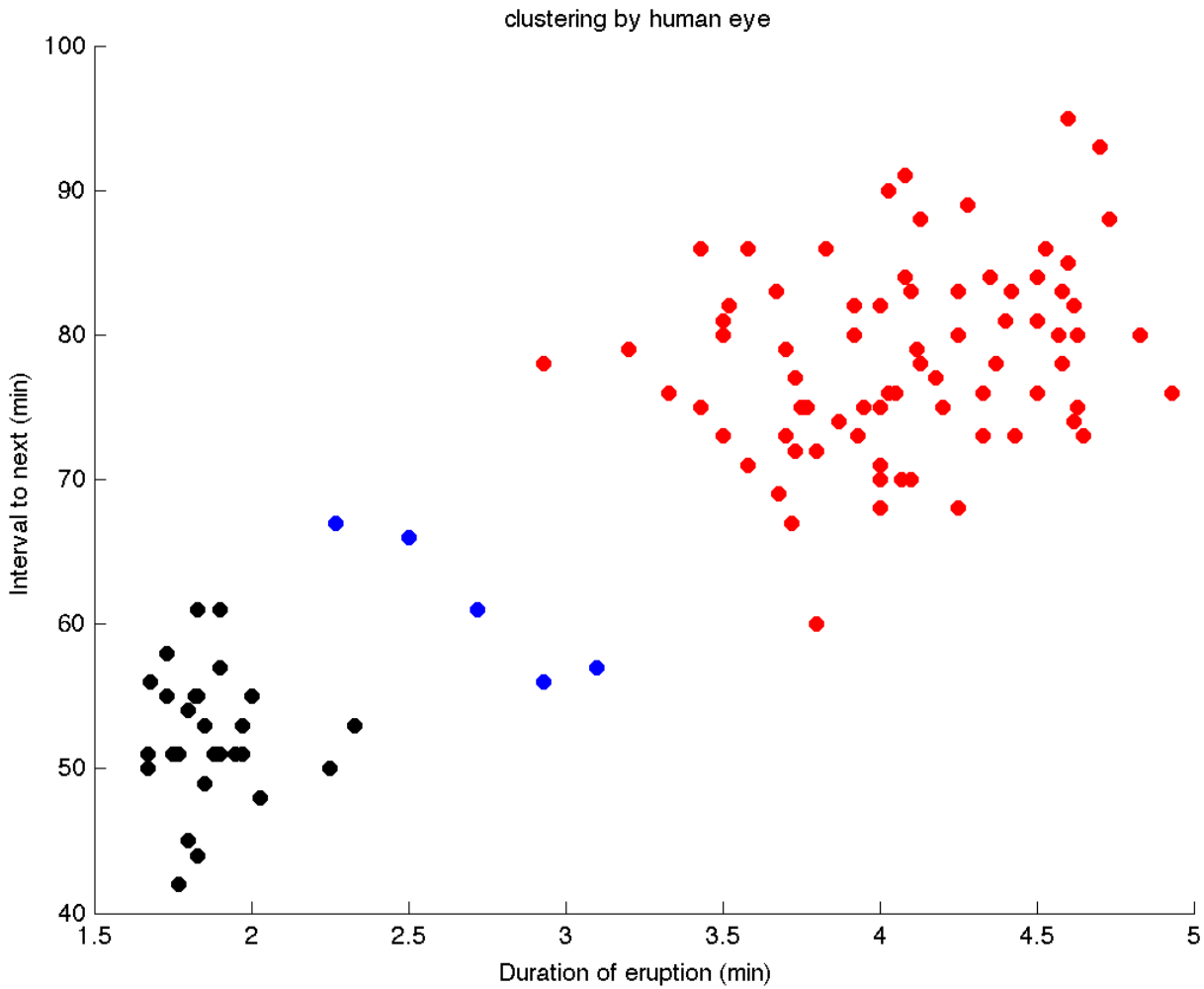
Example: Old Faithful Geyser

- Section 12.1.2 Izenman
- 107 bivariate observation for duration of eruption (X_1) and the waiting time until the next eruption (X_2).
- Can this dataset be divided into two or three sub-groups?



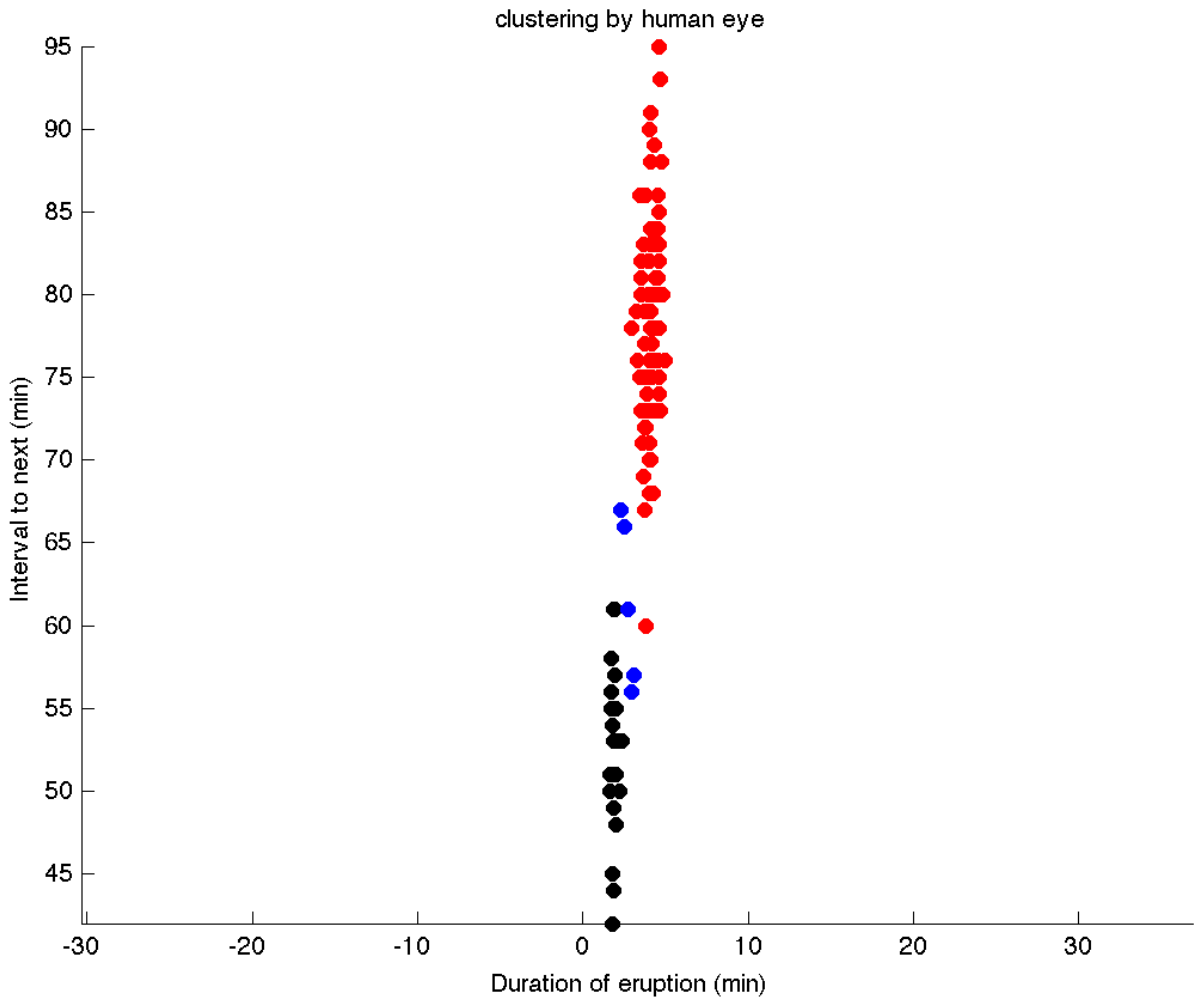
Old Faithful Geyser

- Human perception is excellent??



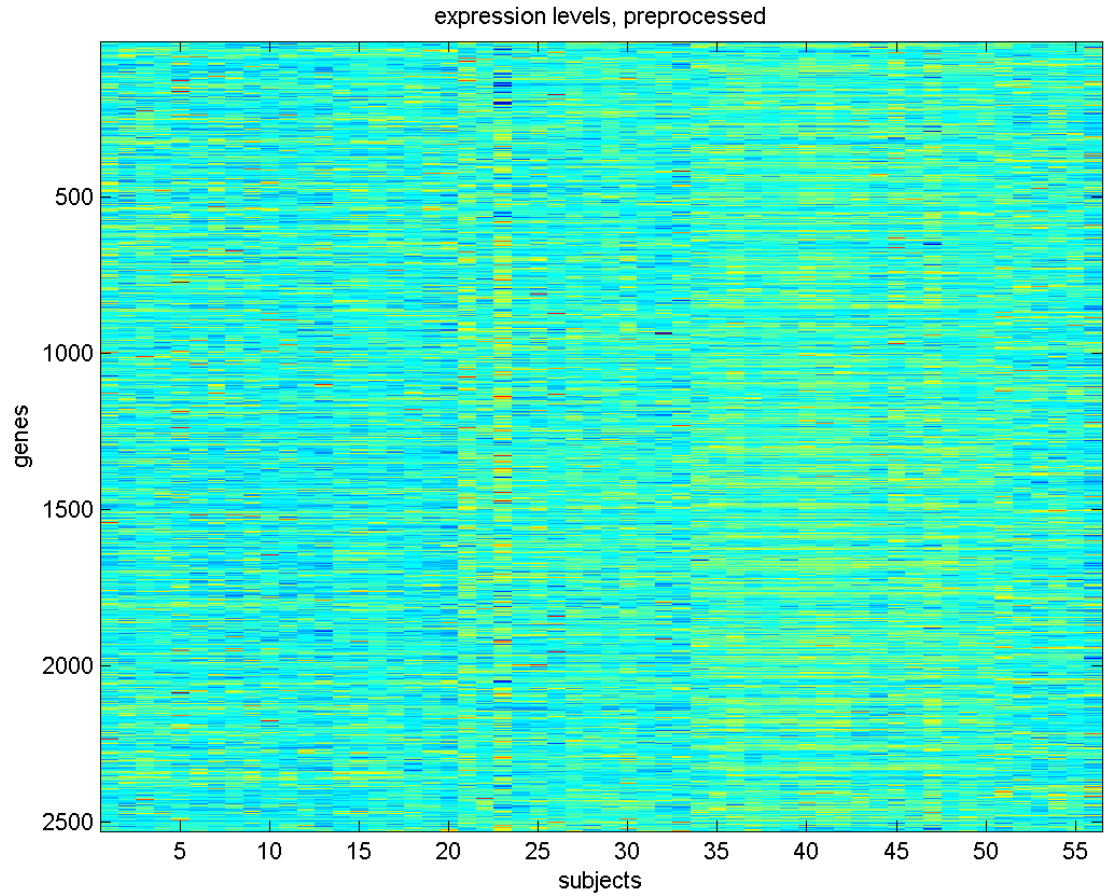
Old Faithful Geyser

- Same data, same clustering, but with different axis.
- Is human perception really excellent?? (Depend on visual.)



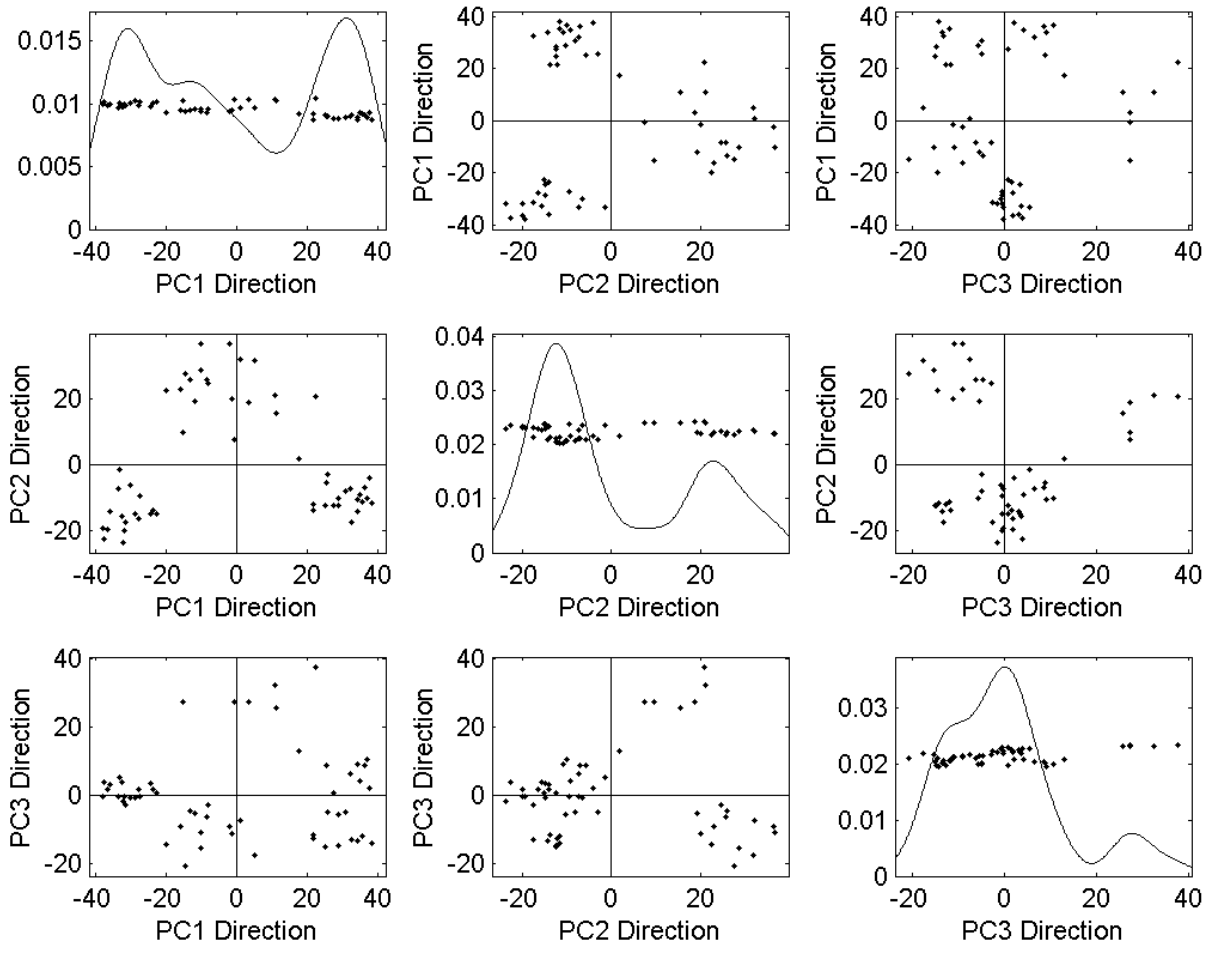
Example: mRNA expression profiling

- Bhattacharjee et al (2001) PNAS
- Preprocessed gene expressions with $d = 2530$ genes and $n = 56$ subjects with lung cancer.
- Subgroup for different types of lung cancers?



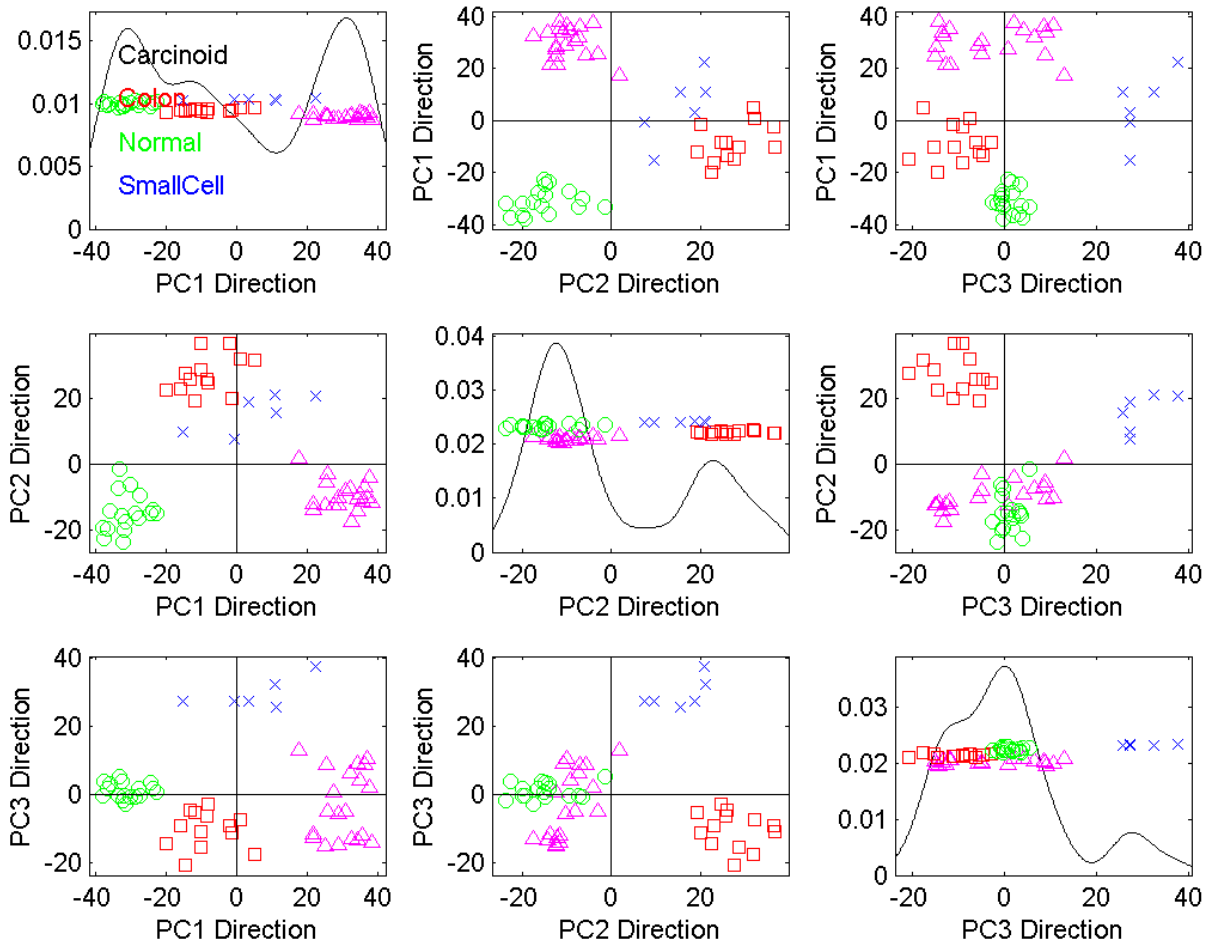
mRNA expression profiling

- Clustering by human eyes requires a good way to visualize the data: Use PCA (scatterplot matrix for PC scores 1–3)



mRNA expression profiling

- Again, is human perception excellent?
- *Yes*, for this data, compare to true subgroups



Clustering Concepts:

- Hard vs. Soft Clustering.
- Model-Based vs. Algorithmic.
- Flat vs. Nested.
- Clustering observations (most common) vs. Clustering features vs. Clustering both (Biclustering).

Ingredients for clustering

- Need a *distance* to measure **similarity** or **dissimilarity** between different observations
- Quality of clusters, number of clusters?

Dissimilarity $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for example:

1. the usual 2-norm $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2$,
2. 1-norm $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_1$ (taxi driver's distance)
3. p-norm $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_p = \left(\sum_{j=1}^{d}(x_j - y_j)^p\right)^{1/p}$, $p > 0$.
4. Correlation between $\boldsymbol{x}$ and $\boldsymbol{y}$.
   Note: if inputs are standardized, then $\|\boldsymbol{x} - \boldsymbol{y}\|^2 \propto 1 - \rho(\boldsymbol{x}, \boldsymbol{y})$

Clustering algorithms

1. Combinatorial algorithm
2. K-means and related:
   K-means, K-medoids, Partitioning around medoids, Fuzzy Analysis, NMF for soft-clustering, Model-based soft-clustering (Gaussian mixture)
3. Hierarchical clustering and related:
   agglomerative, divisive. Biclustering - Cluster-Heatmap. Convex Clustering & Convex Biclustering.

All methods (with exception of a few) allow to use only dissimilarity measures. For now, assume data are quantitative, i.e., $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$

The next section would be . . . . . .

1. Combinatorial algorithm
2. K-means and related methods
3. Hierarchical clustering
4. Other topics

Dissimilarity and within-cluster scatter

Given data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$,

- Clustering is an assignment function

$$c(i) : \{1, \ldots, n\} \to \{1, \ldots, K\},$$

  where K is the number of clusters.

- Within-cluster scatter:

$$W(c) = \frac{1}{2} \sum_{k=1}^{K} \frac{1}{n_k} \sum_{i:c(i)=k} \sum_{j:c(j)=k} d(i,j),$$

  where $n_k = \#\{i : c(i) = k\}$ number of points in cluster k.

- Small $W(c)$ is better.

Combinatorial algorithm

- One needs to minimize W over all possible assignments of n points to K clusters
- The number of distinct assignments is

$$A(n, K) = \frac{1}{K!} \sum_{k=1}^{K} (-1)^{K-k} \binom{n}{k} k^n \sim \frac{K^n}{K!}.$$

- Not so easy for large K and n
  $n = 25$ observations, $K = 4$ clusters: $A(n, K) \geq 10^{13}$
- It calls for more efficient algorithm: may not be optimal but reasonably good sub-optimal solutions

Side note: how was the number of assignments counted

- Stirling numbers of the second kind:
  the number of ways to partition a set of n objects into k non-empty subsets and is often denoted by $S(n, m)$ or $\left\{ {n \atop m} \right\}$.

$$\left\{ {n \atop m} \right\} = \frac{1}{m!} \sum_{j=0}^{m} (-1)^j \binom{m}{j} (m-j)^n.$$

The next section would be . . . . . .

1. Combinatorial algorithm
2. K-means and related methods
   - K-means
   - K-medoids
   - How to Choose K?
   - Related Algorithms
3. Hierarchical clustering
4. Other topics

The next section would be . . . . . .

K-means algorithm: motivation

- In need of an efficient algorithm to (approximately) minimize W among all possible clusterings.
- Another look at W, with $d(i,j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$ (squared Euclidean distance):

$$\frac{1}{2}\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i:c(i)=k}\sum_{j:c(j)=k}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 = \sum_{k=1}^{K}\sum_{i:c(i)=k}\left\|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_{(k)}\right\|_2^2$$

where $\bar{\boldsymbol{x}}_{(k)} = \dfrac{1}{n_k}\displaystyle\sum_{i:c(i)=k}\boldsymbol{x}_i$ (average of all points in cluster k)

K-means algorithm: motivation

- Note: for each fixed k, given clustering $c(\cdot)$, $\bar{\boldsymbol{x}}_{(k)}$ satisfies

$$\bar{\boldsymbol{x}}_{(k)} = \operatorname*{argmin}_{m_k} \sum_{i:c(i)=k} \|\boldsymbol{x}_i - m_k\|_2^2$$

  Thus

$$W(c) = \min_{m_1,\ldots,m_K} \sum_{k=1}^{K} \sum_{i:c(i)=k} \|\boldsymbol{x}_i - m_k\|_2^2 ,$$

- Idea on computing: minimize the modified criterion $L(c|m_1, \ldots, m_K) = \sum_{k=1}^{K} \sum_{i:c(i)=k} \|\boldsymbol{x}_i - m_k\|_2^2$ by alternately minimizing over c (given m_j's) and over $m_1, \ldots, m_K$ (given c).

K-means algorithm

The algorithmic iteration begins with an initial guess for K cluster centers $(m_1, \ldots, m_K)$,

1. Minimize over c: For each $\boldsymbol{x}_i$ $(i = 1, \ldots, n)$, find the cluster center m_k closest to $\boldsymbol{x}_i$, then update $c(i) = k$.
2. Minimize over $m_1, \ldots, m_K$: For each cluster, update m_k by the new average of points in cluster k.
3. Iterate Steps 1 and 2 until $L(c|\boldsymbol{m})$ does not change.

Variation on K-means algorithm: When $\rho(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is used rather than the squared Euclidean distance.

- in Step 1, *closest* cluster center is found by ρ
- in Step 2, *average* is appropriately defined by ρ

When m_k's are fixed,

$$\sum_{k=1}^{K} \sum_{i:c(i)=k} \|\boldsymbol{x}_i - m_k\|_2^2$$

$$= \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{1}_{[c(i)=k]} \|\boldsymbol{x}_i - m_k\|_2^2$$

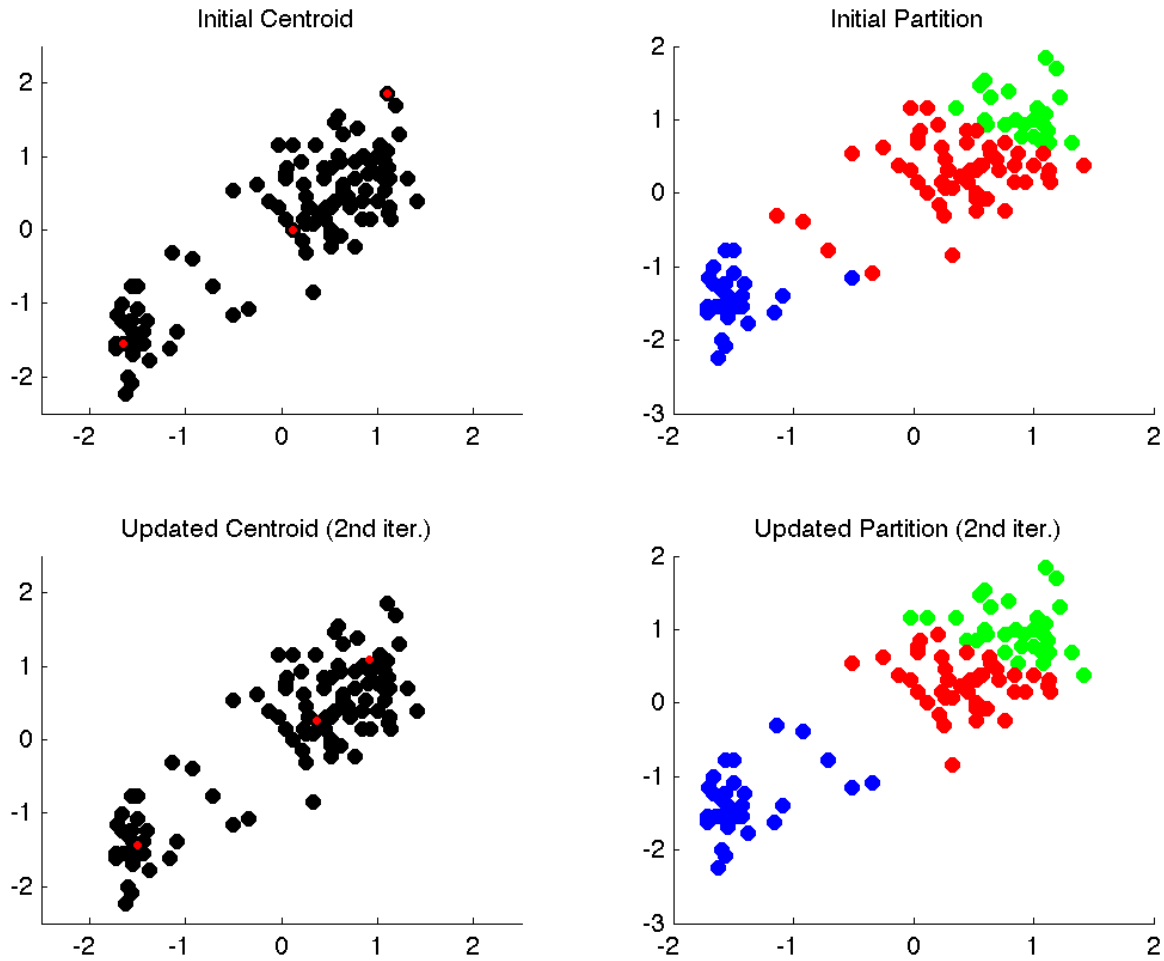
which is separable for $\boldsymbol{x}_i$'s. Therefore, we only need to find the best $c(i)$ for each i separately so that

$$\sum_{k=1}^{K} \mathbb{1}_{[c(i)=k]} \|\boldsymbol{x}_i - m_k\|_2^2$$

is minimized. Clearly we should choose $c(i) = \operatorname{argmin}_k \|\boldsymbol{x}_i - m_k\|_2$

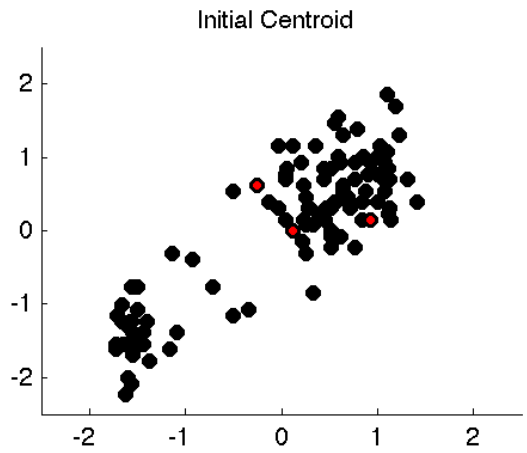
K-means example

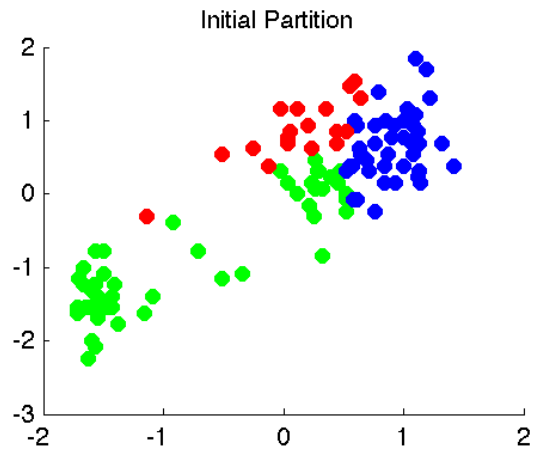
- First two iterations of the algorithm

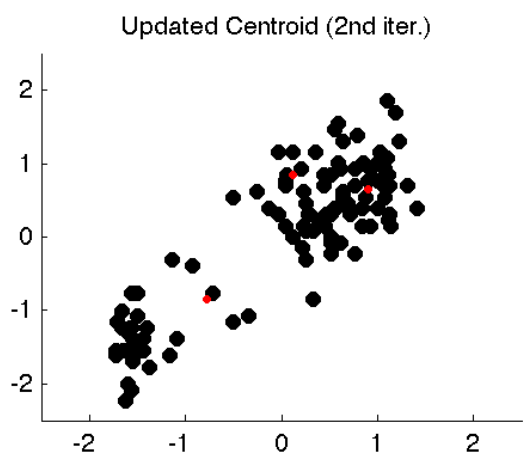


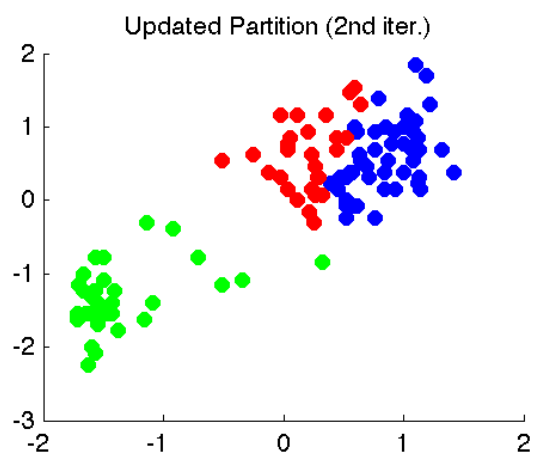
K-means example

- First two iterations of the algorithm (*with different initial*)



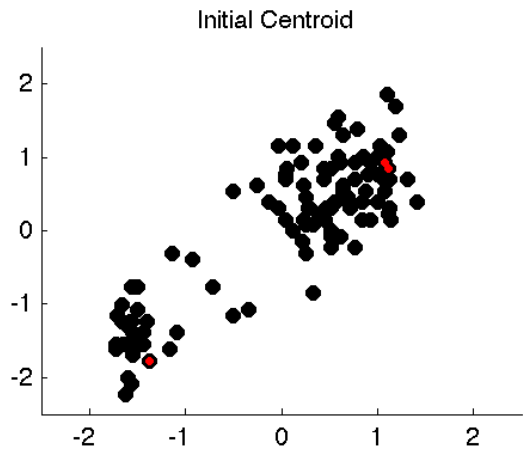


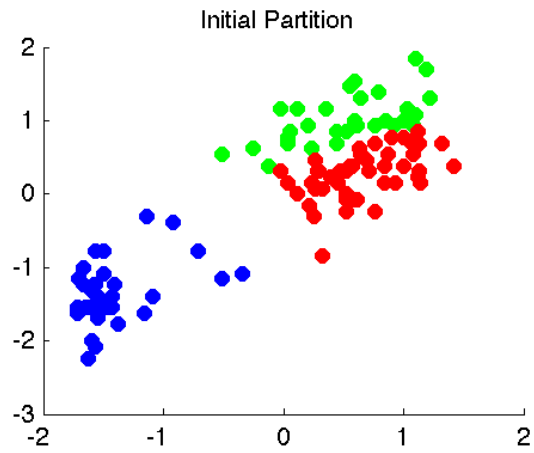


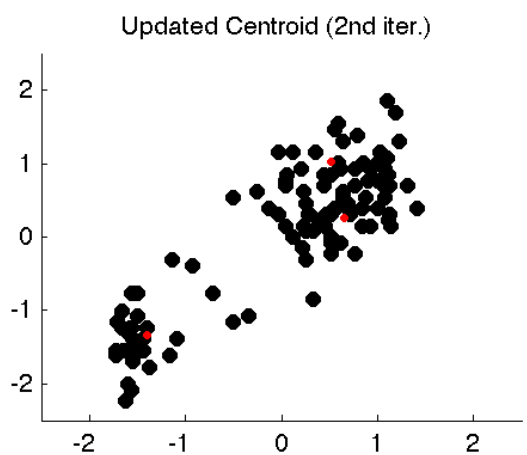


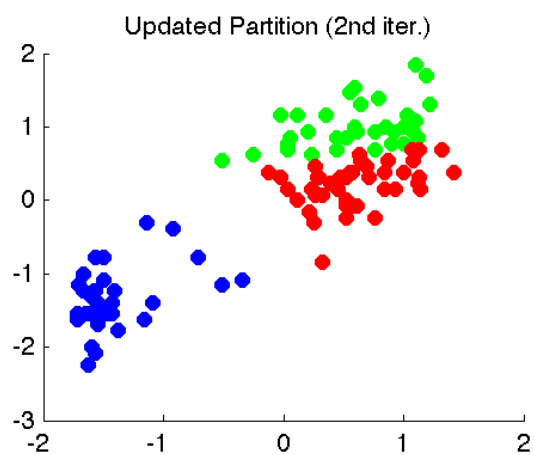
K-means example

- First two iterations of the algorithm (*another different initial*)









K-means algorithm

Colors between figures do not matter!

- Each iteration always makes $L(c|m_1, \ldots, m_K)$ smaller
- Iteration always finishes (converges)
- Different initial values may lead to different solutions.
- K-means is typically run multiple times, with a random initial value for each run. Final solution as the one with the smallest within-cluster scatters
- Still sub-optimal compared to the combinatorial method
- Works well for quantitative variables

The next section would be . . . . . .

Country Dissimilarities

- The average dissimilarity scores are given.
- K-means clustering could not be applied because we have only distances rather than raw observations.

TABLE 14.3. *Data from a political science survey: values are average pairwise dissimilarities of countries from a questionnaire given to political science students.*

	BEL	BRA	CHI	CUB	EGY	FRA	IND	ISR	USA	USS	YUG
BRA	5.58										
CHI	7.00	6.50									
CUB	7.08	7.00	3.83								
EGY	4.83	5.08	8.17	5.83							
FRA	2.17	5.75	6.67	6.92	4.92						
IND	6.42	5.00	5.58	6.00	4.67	6.42					
ISR	3.42	5.50	6.42	6.42	5.00	3.92	6.17				
USA	2.50	4.92	6.25	7.33	4.50	2.25	6.33	2.75			
USS	6.08	6.67	4.25	2.67	6.00	6.17	6.17	6.92	6.17		
YUG	5.25	6.83	4.50	3.75	5.75	5.42	6.08	5.83	6.67	3.67	
ZAI	4.75	3.00	6.08	6.67	5.00	5.58	4.83	6.17	5.67	6.50	6.92

When K-means is not prefered

- In K-means, each cluster is represented by the centroid (the average of all points in kth cluster)
- In some applications
  1. we want each cluster represented by one of the points in the data (instead of some averaged point which may be meaningless).
  2. we only have pairwise dissimilarities d_{ij} but do not have actual points (thus no averaging)
- This is where *K-medoids* comes in (two slides later)
- **Country Dissimilarities example**: Kaufman and Rousseeuw (1990)

  A study in which political science students were asked to provide pairwise dissimilarity measures for 12 countries: Belgium, Brazil, Chile, Cuba, Egypt, France, India, Israel, United States, Union of Soviet Socialist Republics, Yugoslavia and Zaire (Data next slide)

K-medoids algorithm

K-medoids is similar to K-means, but searches for K *representative objects* (medoids)

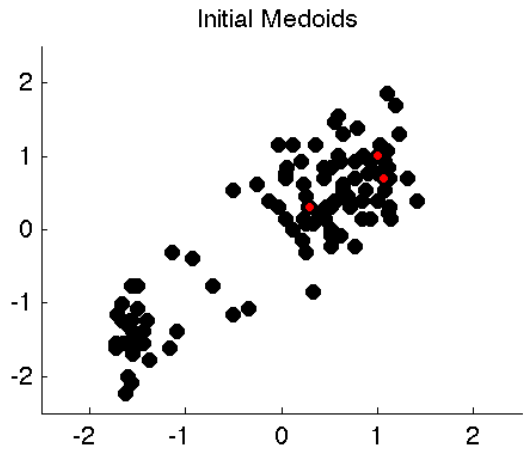
K-medoids

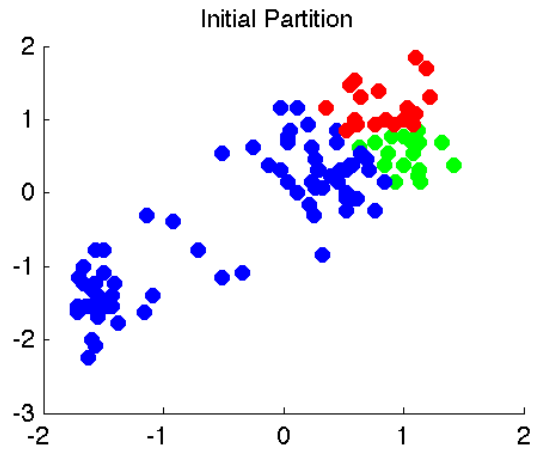
The algorithmic iteration begins with an initial guess for K cluster medoids ($m_i \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$),

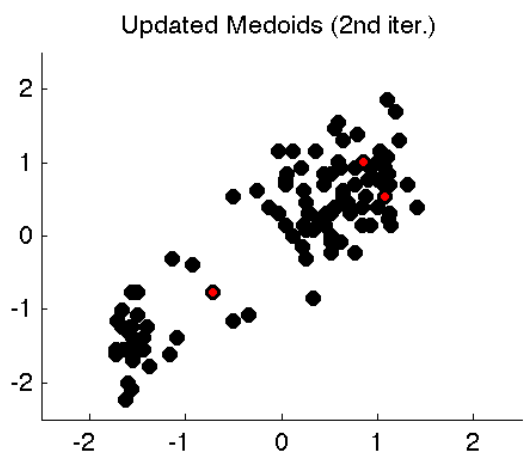
1. Minimize over c: For each $\boldsymbol{x}_i$ $(i = 1, \ldots, n)$, find the cluster medoids m_k closest to $\boldsymbol{x}_i$, then update $c(i) = k$.
2. Minimize over $m_1, \ldots, m_K$: Locate the medoid for each cluster. The medoid of the kth cluster is defined as the item in the kth cluster that minimizes the total dissimilarity to all other items within that cluster.
3. Iterate Steps 1 and 2 until $L(c|m_k)$ does not change.

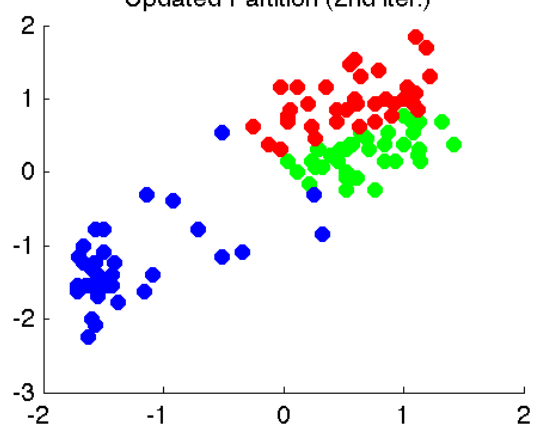
K-medoids example (Geyser)

- First two iterations of the algorithm, using sq. Euc. distance as dissimilarity
- New set of medoids consists of original observations



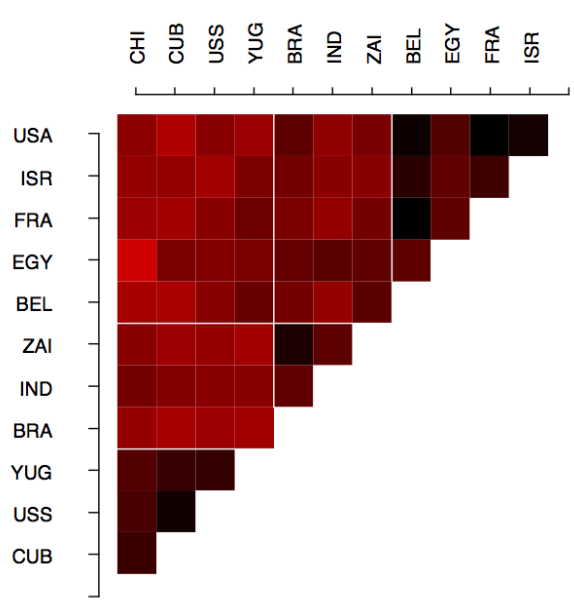




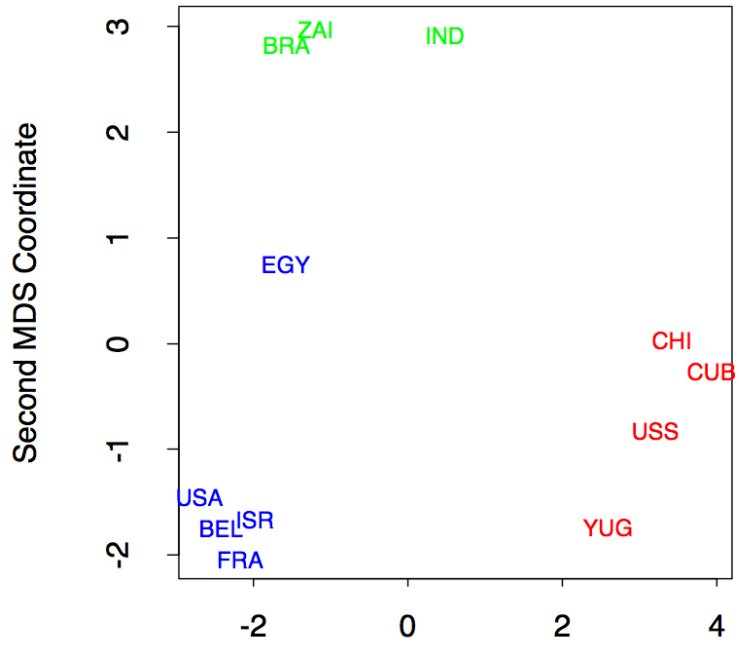


K-medoids example (Country)

Survey of country dissimilarities. (Left panel:) dissimilarities reordered and blocked according to 3-medoid clustering. Heat map is coded from most similar (dark red) to least similar (bright red). (Right panel:) two-dimensional multidimensional scaling plot, with 3-medoid clusters indicated by different colors.



Reordered Dissimilarity Matrix



First MDS Coordinate

More partitioning methods

- Izenman discusses two other methods `pam` and `fanny`
- `pam` (partitioning around medoids) is a variation of K-medoids, by allowing swapping of medoids
  - Viewed as a realization for k-mediod.
  - Slow for large data
- `fanny` (fuzzy clustering): instead of assigning clusters by the clustering function $c(i)$, strength of membership (like probabilities) u_{ik} (of the ith point belonging to kth cluster) are assigned.

  The solution for probabilities minimizes

$$\sum_{k=1}^{K} \frac{\sum_i \sum_j u_{ik}^2 u_{jk}^2 d_{ij}}{2 \sum_\ell u_{\ell k}^2}$$

The next section would be . . . . . .

How many clusters?

What is the value of K?

Using K-means, K-medoids, or hierarchical clustering (next section), attempts at formulating formal criteria to decide on the number of clusters have not been successful, by and large.

There are situations where the value of K is pre-determined, e.g.,

- Theory / domain knowledge suggests existence of K clusters
- Segmenting a client database into K clusters for K salesman

Scatter decomposition

Focus on squared Euclidean distance.
Recall **within-cluster scatter**

$$W(c) = \sum_{k=1}^{K} \sum_{i:c(i)=k} \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}_{(k)} \right\|^2 .$$

- smaller W is better (combinatorial method)
- WCS W keeps decreasing for larger K (No use in decision of right K)

Consider ANOVA-like decomposition of **total scatter**

$$\sum_i \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}\|^2 = T = W(c) + B(c),$$

where $B(c) = \sum_{k=1}^{K} n_k \left\| \bar{\boldsymbol{x}}_{(k)} - \bar{\boldsymbol{x}} \right\|^2$ is the between-cluster scatter.

Scatter decomposition

Between-cluster scatter

$$B(c) = \sum_{k=1}^{K} n_k \left\| \bar{\boldsymbol{x}}_{(k)} - \bar{\boldsymbol{x}} \right\|^2 .$$

- larger B is better (large gaps between clusters)
- BCS W keeps decreasing for larger K
- No use in determining the right K

Cluster Index

Cluster index is the standardized within-cluster scatter

$$CI(K, c) = W(K, c)/T.$$

- $\in (0, 1)$, Unit free
- Still increasing for large K

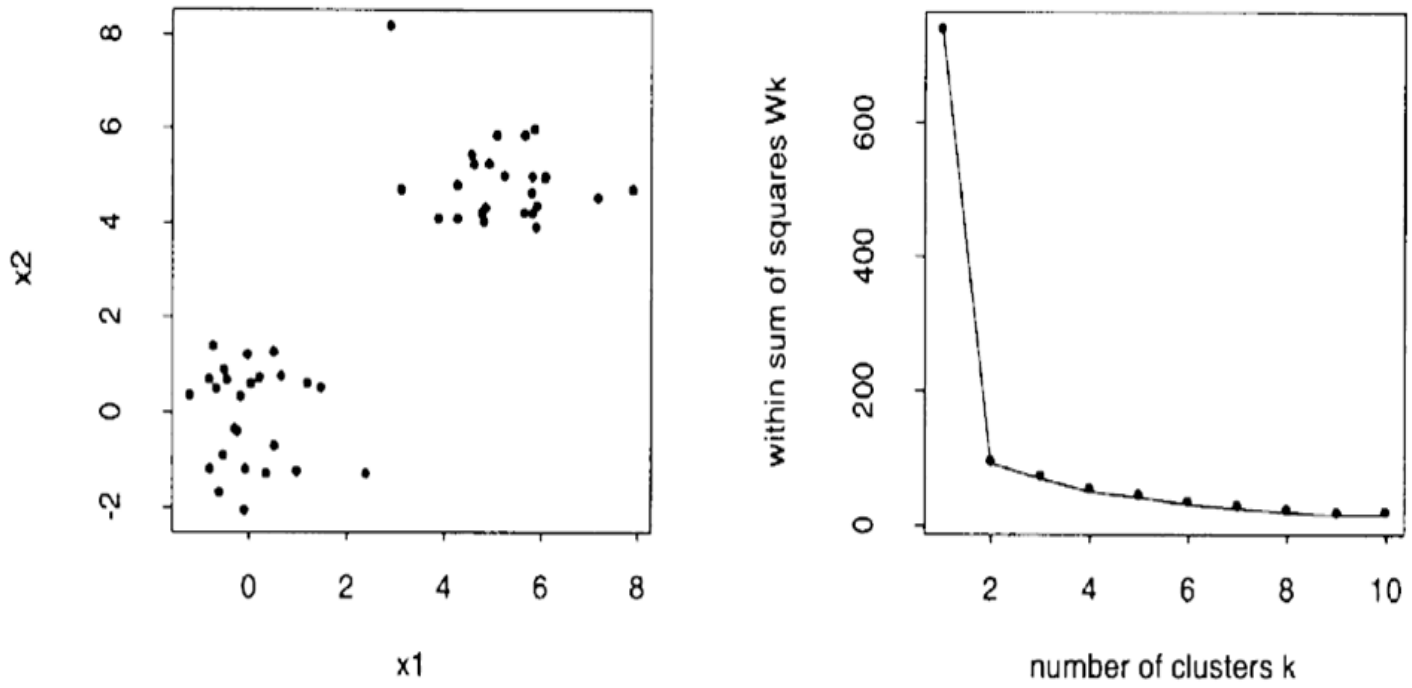
CH Index

$$CH(K, c) = \frac{B(K, c)/(K-1)}{W(K, c)/(n-K)}.$$

- A large $CH \iff$ a small W and a large B
- Not monotonic in K
- Can choose a K with largest CH index
- (Problem: no way choosing $K = 1$)

Scatter decomposition

One may look for an elbow in a scree plot of $W(K)$ or $CI(K)$ (as was did in PCA)



$W_1 \gg W_2$ since natural groups are assigned to separate clusters
Smaller decrease W_k to W_{k+1} $(k \geq 2)$, as natural groups are partitioned

How to Choose K?

- Gap Statistic.
- Silhouette Statistic.
- Cluster Prediction Strength.
- Cluster Stability.

Gap statistic

- Measures how much $W(K)$ drops compared to a null case
- The observed WCS $W(K)$ is compared with the expected WCS when there was only one cluster (null distribution):

$$Gap(K) = E^*[\log W_0(K)] - \log W(K)$$

- $E^*[\log W_0(K)]$ is simulated by a one-cluster distribution (each dimension is uniform) as null distribution.
- The greater $Gap(K)$, the better. As function of K, first increase, then decrease.
- The standard deviation $s(K)$ of $\log W_0(K)$ is also computed

Gap statistic

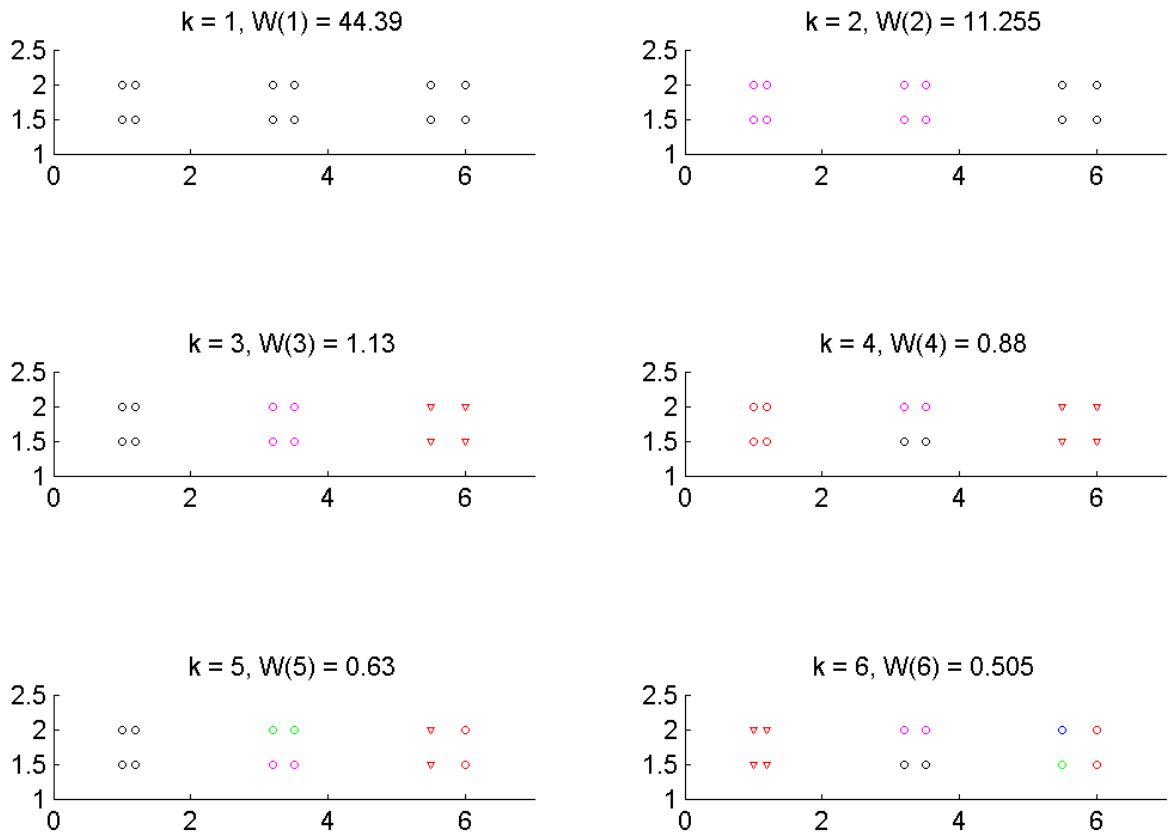
- We then choose the smallest K such that

$$Gap(K) \geq Gap(K+1) - s(K+1).$$

  Overall we find the K with the greatest gap. The "$-s(K+1)$" part is used to offset some random perturbation. Tends to be conservative (K a little less).
- R: `clusGap` in cluster package.

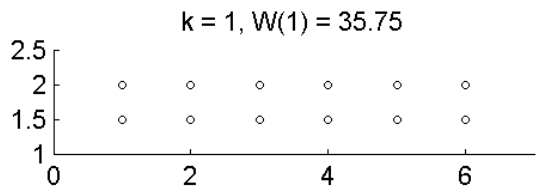
Gap statistic: Toy example

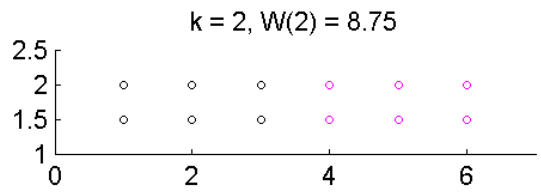
- Data with *true* $K = 3$.
- Results from K-means algorithm.

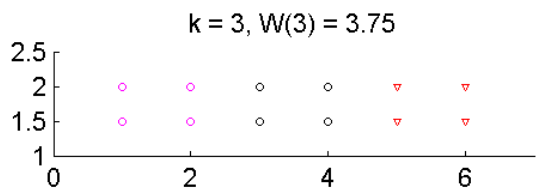


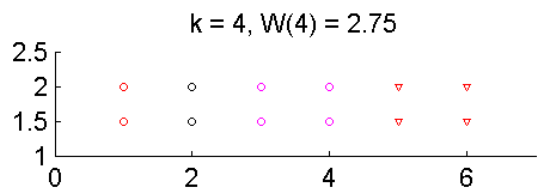
Gap statistic: Toy example

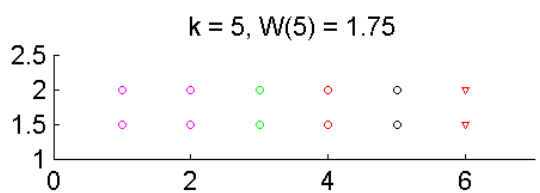
- What would be the value of $W(k)$ if there is only one cluster?
- Reference sampled from uniform, then $W(\text{reference})$ computed from K-means algorithm.

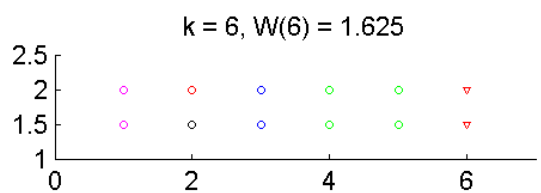






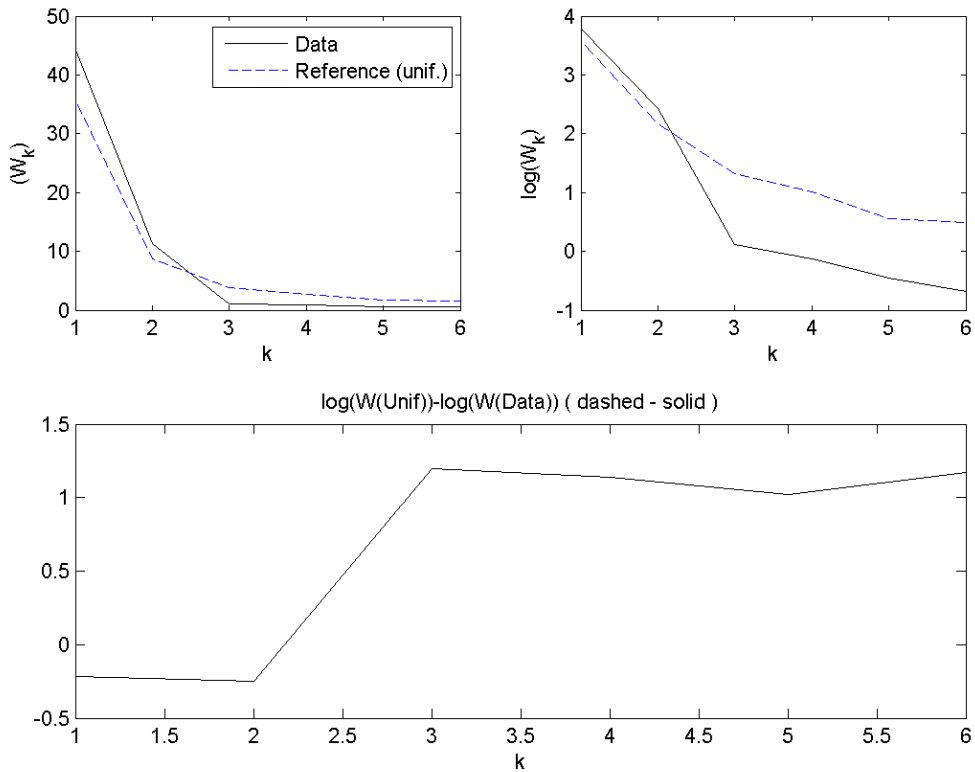






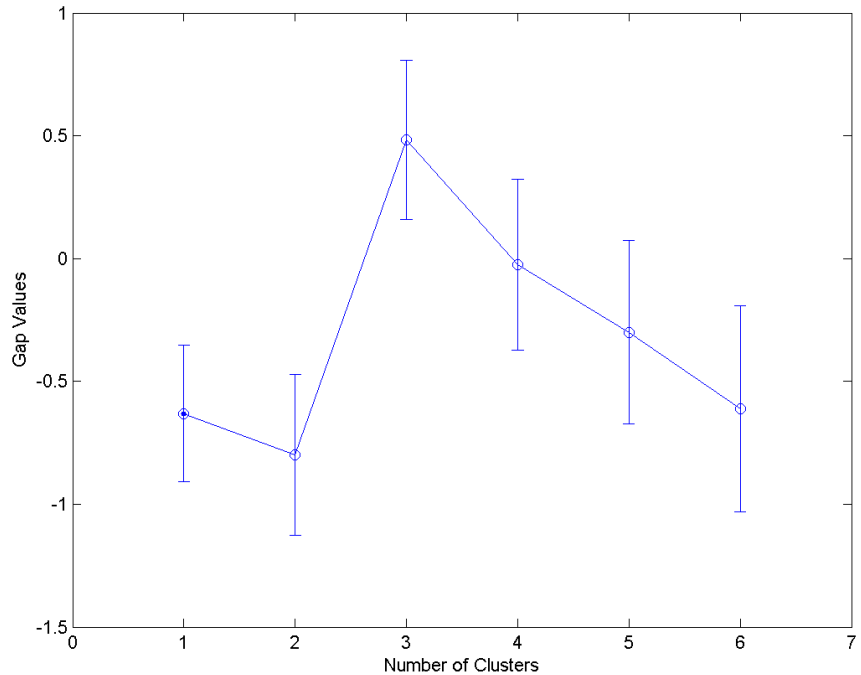
Gap statistic: Toy example

- Compare the amount of decrease of $W(k)$ in 'data' and 'reference' (Gap between `log(W(ref.))` and `log(W(data))`)
- Remember that this is just **one** realization of reference distribution.



Gap statistic: Toy example

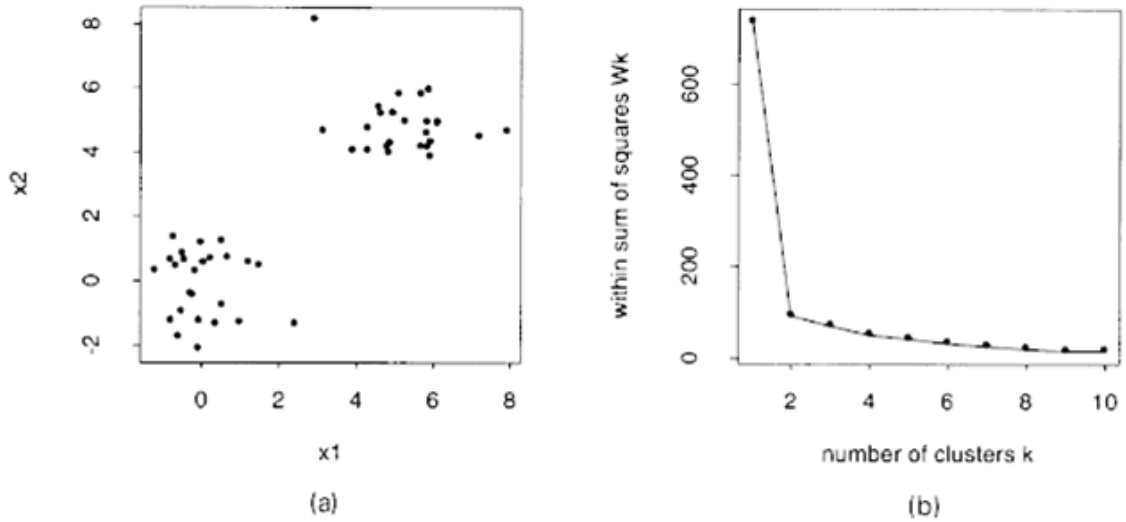
- Take the mean and standard deviation of many `log(W(ref.))` to obtain $E^*[\log W_0(K)]$ and $s(K)$
- The smallest K with $Gap(K) \geq Gap(K+1) - s(K+1)$ is 1?
- In cases where there are smaller subclusters within larger well-separated clusters, Gap curve can exhibit non-monotone behavior; *Important to examine the entire gap curve*

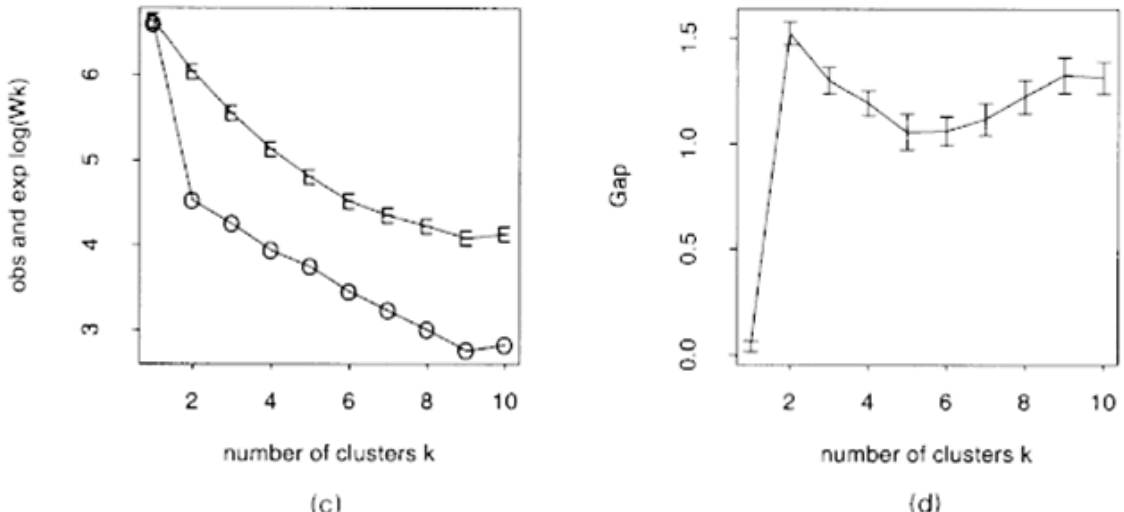


Gap statistic– 2-cluster example

$(K = 1:)$ $Gap(1) < Gap(2) - s(2)$, move on to $K = 2$

$(K = 2:)$ $Gap(2) > Gap(3) - s(3)$. $\hat{K} = 2$.





Gap statistic– 1-cluster example

$(K = 1:)$ $Gap(1) > Gap(2) - s(2)$, $\hat{K} = 1$.

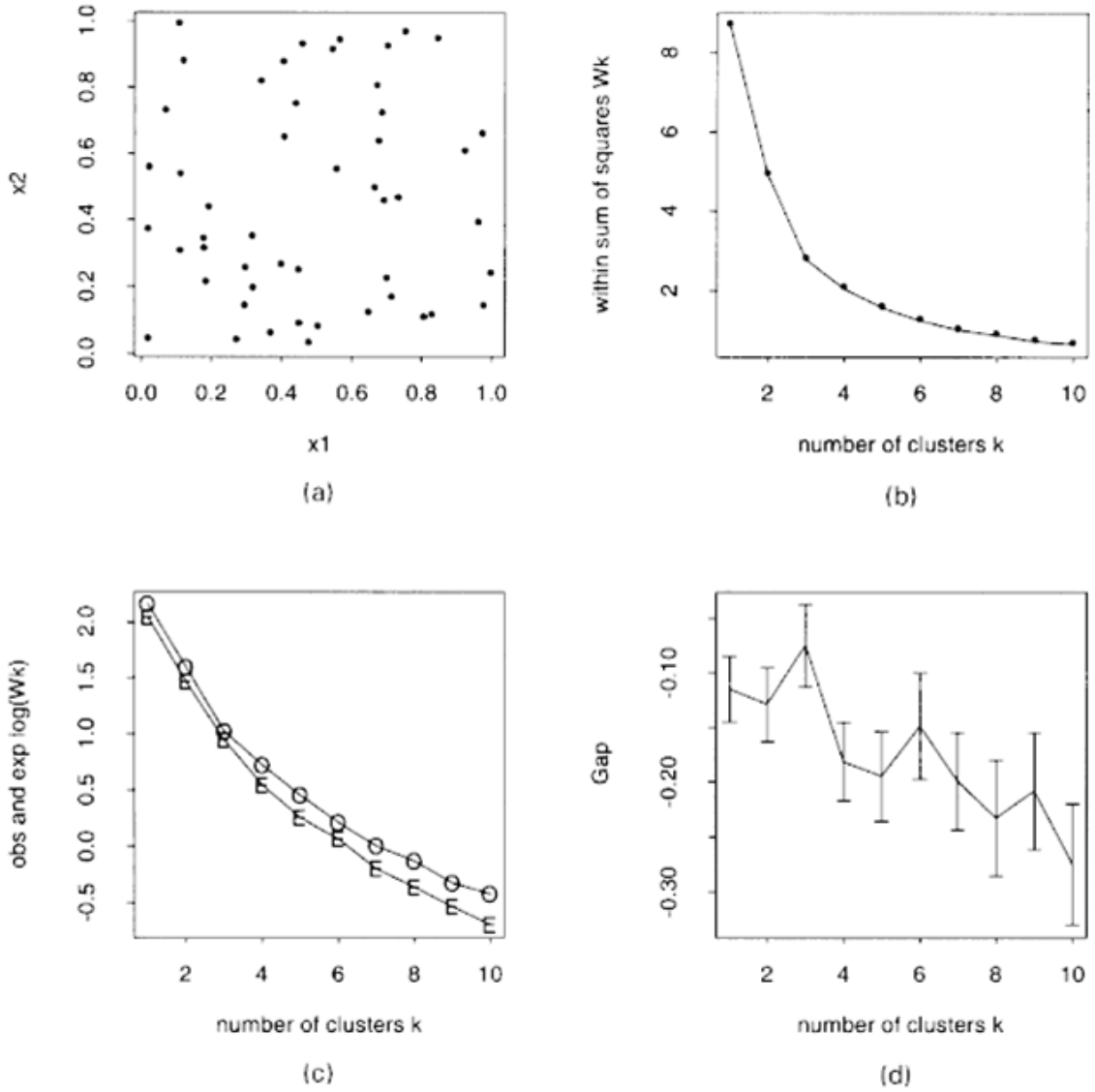


Fig. 2. Results for the uniform data example: (a) data; (b) within sum of squares function W_k; (c) functions

Silhouette Statistic

Silhouette Statistic:

- a_i - mean within-cluster dissimilarity with observation i.
- b_i - mean between-cluster dissimilarity with observation i.
- Silhouette Statistic: $S_i = \frac{b_i - a_i}{max(b_i, a_i)}$
- S_i close to 1 = good clustering.
- S_i close to -1 = bad clustering.
- Choose K that maximizes average S_i.
- R: `silhouette` in cluster package.

Other ways to choose k

Prediction Strength:

- Split into training set and test set; k-means for each; measure overlap between clusters; repeat and average

Cluster Stability:

- Perturb data (bootstrap; sub-sampling; etc.).
- Choose K where cluster assignments are most stable over perturbations.

Metrics to measure overlap between cluster assignments:

- Rand Index; R: rand indep in clusteval package.
- Jaccard Index; R: jaccard indep in clusteval package.

Summary - K-means

Strengths:

- Fast.
- Simple.
- Others?

Weaknesses:

- Local solution - highly depends on initialization.
- High-dimensional settings? ($p \gg n$ - more features than observations)
- Others?

The next section would be . . . . . .

Soft Clustering: Mixture Models

- Mixture of k distributions.
- Assign each observation a probability of arising from distribution j.
- Most Common: Gaussian Mixture model.
- Algorithm: EM (Expectation-Maximization).
  1. E-step: Cluster probabilities for each observation.
  2. M-Step: Given soft-cluster assignments, maximize likelihood estimation for each distribution.

`mclust` package in R.

Gaussian Mixture and EM algorithm

Model-based clustering:
Consider a mixture model density in $\mathbb{R}^p$

$$g(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k g_k(\boldsymbol{x}),$$

where g_k is the pdf for the kth cluster $N_p(\mu_k, \sigma^2 \mathbb{I}_p)$, $\pi_k > 0$, $\sum_k \pi_k = 1$.

- For data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \sim g$, clustering the data $\approx$ estimation of parameters
- EM algorithm is a soft version of K-means algorithm with a underlying model.

EM algorithm

- Designed for estimating parameter with missing data
- Gaussian mixture (and hence clustering) can be understood as a missing data problem

Main idea:

- $\mathcal{D} = \{\mathcal{D}_{obs}, \mathcal{D}_{mis}\}$
- Full likelihood: $L(\theta \mid \mathcal{D})$ is not available
- Instead, likelihood based on observed data:

$$L(\theta|\mathcal{D}_{obs}) = \int f(\mathcal{D}_{obs}, \mathcal{D}_{mis} \mid \theta) d\mathcal{D}_{mis}$$

  should be maximized $\leftarrow$ too difficult.
- EM algorithm is a two-step iterative algorithm to solve this.

EM algorithm (general form)

1. $\widehat{\theta}^{(0)}$: first guess on the parameter
2. For $m = 0, 1, \ldots,$ iterate between the following two steps
   (a) E-step: compute

$$Q(\theta \mid \widehat{\theta}^{(m)}) := \mathrm{E}\{\ell(\theta \mid \mathcal{D}) \mid \mathcal{D}_{obs}, \widehat{\theta}^{(m)}\}$$

   as a function of θ.
   (b) M-step: Find $\widehat{\theta}^{(m+1)} := \operatorname{argmax}_\theta Q(\theta \mid \theta^{(m)})$
3. Stop when converged.

- The expectation is with respect to the conditional distribution of $\mathcal{D}_{mis}$ given $\mathcal{D}_{obs}$ and the current estimate $\widehat{\theta}^{(m)}$
- Always converges
- May converge to a local maximum
- May be slow (for too many missing data or too large data.)

Why the approximation in the E step?

$$f(\mathcal{D}_{mis}|\mathcal{D}_{obs},\theta) = \frac{f(\mathcal{D}_{mis},\mathcal{D}_{obs}|\theta)}{f(\mathcal{D}_{obs}|\theta)}$$

$$\log\left[f(\mathcal{D}_{mis}|\mathcal{D}_{obs},\theta)\right] = \log[L(\theta|\mathcal{D}_{mis},\mathcal{D}_{obs})] - \log[L(\theta|\mathcal{D}_{obs})]$$

$$\log\left[f(\mathcal{D}_{mis}|\mathcal{D}_{obs},\theta)\right] = \ell(\theta|\mathcal{D}_{mis},\mathcal{D}_{obs}) - \ell(\theta|\mathcal{D}_{obs})$$

Take the expected value with respect to the distribution of $\mathcal{D}_{mis}$ given $\mathcal{D}_{obs}$ and θ'

$$H(\theta|\theta') = Q(\theta|\theta') - \ell(\theta|\mathcal{D}_{obs})$$

$$\ell(\theta|\mathcal{D}_{obs}) = Q(\theta|\theta') - H(\theta|\theta')$$

Here $H(\theta|\theta') := \int \log\left[f(\mathcal{D}_{mis}|\mathcal{D}_{obs},\theta)\right] f(\mathcal{D}_{mis}|\mathcal{D}_{obs},\theta')d\mathcal{D}_{mis}$

Compare $H(\theta'|\theta')$ and $H(\theta|\theta')$

$$
\begin{aligned}
& H(\theta'|\theta') - H(\theta|\theta') \\
= & \int \log \left[\frac{f(\mathcal{D}_{mis}|\mathcal{D}_{obs}, \theta')}{f(\mathcal{D}_{mis}|\mathcal{D}_{obs}, \theta)} \right] f(\mathcal{D}_{mis}|\mathcal{D}_{obs}, \theta') d\mathcal{D}_{mis} \geq 0 \\
& \text{(KL divergence)}
\end{aligned}
$$

Now between (m) and $(m+1)$ steps,

$$
\begin{aligned}
& \ell(\theta^{m+1}|\mathcal{D}_{obs}) - \ell(\theta^m|\mathcal{D}_{obs}) \\
= & [Q(\theta^{m+1}|\theta^m) - Q(\theta^m|\theta^m)] \\
& \quad - [H(\theta^{m+1}|\theta^m) - H(\theta^m|\theta^m)] \\
\geq & Q(\theta^{m+1}|\theta^m) - Q(\theta^m|\theta^m) \geq 0
\end{aligned}
$$

So after each iteration, $\ell(\cdot|\mathcal{D}_{obs})$ is already increased.

Clustering (Gaussian Mixture) as missing data (for $K = 2$ case)

See Section 8.5 ESL.

$$\delta_i = 0 \Rightarrow \text{Cluster 1} \sim N(\mu_1, \sigma_1^2)$$
$$\delta_i = 1 \Rightarrow \text{Cluster 2} \sim N(\mu_2, \sigma_2^2)$$
$$\Pr(\delta_i = 1) = \pi, \ \Pr(\delta_i = 0) = 1 - \pi$$
$$X_i \sim (1 - \delta_i) N(\mu_1, \sigma_1^2) + \delta_i N(\mu_2, \sigma_2^2).$$

- parameters $\theta = (\pi, \theta_1, \theta_2) = (\pi, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2)$
- δ_i is missing
- X_i is observed
- (X_i, δ_i) – full data

Clustering: find $\{\delta_i\}$ via estimating $(\pi, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2)$

- Full data log-likelihood:

$$\begin{aligned}\ell(\theta \mid \mathcal{D}) &= \ell(\theta \mid (X_i, \delta_i)\ i = 1, \ldots, n) \\ &= \log\{\prod_i [\phi_{\theta_1}(x_i)]^{1-\delta_i} [\phi_{\theta_2}(x_i)]^{\delta_i} \pi^{\delta_i} (1-\pi)^{1-\delta_i}\} \\ &= \sum_i \{(1-\delta_i)\log[\phi_{\theta_1}(x_i)] + \delta_i \log[\phi_{\theta_2}(x_i)] \\ &\quad + \delta_i \log \pi + (1-\delta_i)\log(1-\pi)\}\end{aligned}$$

- To find conditional expectation of the full likelihood above, need $\bar{\delta}_i := \mathsf{E}[\delta_i \mid \widehat{\theta}, X_i = x_i]$ (due to linearity)

$$\begin{aligned}\bar{\delta}_i :=& \mathsf{E}[\delta_i \mid \widehat{\theta}, X_i = x_i] = \Pr(\delta_i = 1 \mid \widehat{\theta}, X_i = x_i) \\ =& \frac{\phi_{\hat{\theta}_2}(x_i)\hat{\pi}}{\phi_{\hat{\theta}_2}(x_i)\hat{\pi} + \phi_{\hat{\theta}_1}(x_i)(1-\hat{\pi})} (\text{Bayes theorem})\end{aligned}$$

- E-step: plug $\bar{\delta}_i$ into δ_i in the full data likelihood to obtain $Q(\theta \mid \widehat{\theta})$
- M-step: maximize $Q(\theta \mid \widehat{\theta})$ over $\theta \leftarrow$ standard procedure; easy since $\pi, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2$ are all separated

Given $\bar{\delta}_i$, $i = 1, \ldots, n$, the solution to the M step are:

- $\hat{\pi} = \dfrac{\sum_{i=1}^{n} \bar{\delta}_i}{n}$
- $\hat{\mu}_1 = \dfrac{\sum_{i=1}^{n} (1 - \bar{\delta}_i) \boldsymbol{x}_i}{\sum_{i=1}^{n} (1 - \bar{\delta}_i)}$ $\leftarrow$ soft version of sample mean of those deemed to be cluster 1 ($\bar{\delta}_i = 0$)
- $\hat{\mu}_2 = \dfrac{\sum_{i=1}^{n} \bar{\delta}_i \boldsymbol{x}_i}{\sum_{i=1}^{n} \bar{\delta}_i}$ $\leftarrow$ soft version of sample mean of those deemed to be cluster 2 ($\bar{\delta}_i = 1$)
- ...
- ...

Compare K-means and Mixture Model

- K-means: for fixed centroids m_k's, assign x_i to the cluster with the closest centroid; prob. that it belongs to a class $= 0$ or 1.
- Mixture Model / EM: for fixed centroids $(\pi, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2)$, estimate δ_i by its conditional expectation (through Bayes theorem);prob. that it belongs to a class $\in (0, 1)$

Soft-Clustering: NMF

$$\mathbf{X}_{n\times p} \approx \mathbf{W}_{n\times q}\mathbf{H}_{q\times p}$$

- Clusters: Each column of $\mathbf{W}$.
- Soft-Cluster Assignments: $\mathbf{W}_j = \underbrace{(0, 0.4, 1, 0, 0, 2.1)^T}_{n}$.
- Observations can be assigned non-zero weights to more than one cluster.
- E.g. a news article can be a Trump news, a 2nd amendment news or an international politic news.
- Hard-Cluster Assignment: cluster i to the cluster j with greatest value of W_{ij}
- Features that help to explain cluster j: Row vector $\mathbf{H}_j$

`NMF` package in R.

The next section would be . . . . . .

1. Combinatorial algorithm
2. K-means and related methods
3. Hierarchical clustering
4. Other topics

Hierarchical clustering

Partitioning methods (K-means, K-medoids):

- fit K clusters, for pre-determined number K of clusters.
- Results of clustering depend on the choice of initial cluster centers
- No relation between clusterings from 2-means and those from 3-means.

Hierarchical clustering:

- does not depend on initial values – one and unique solution,
- gives clustering assignments for all $K = 1, \ldots, n$.
- has clear relationship between $(K-1)$-cluster clusterings and K-cluster clustering (nested)

Agglomerative vs divisive

Two types of hierarchical clustering algorithms

Agglomerative (bottom-up) - more popular

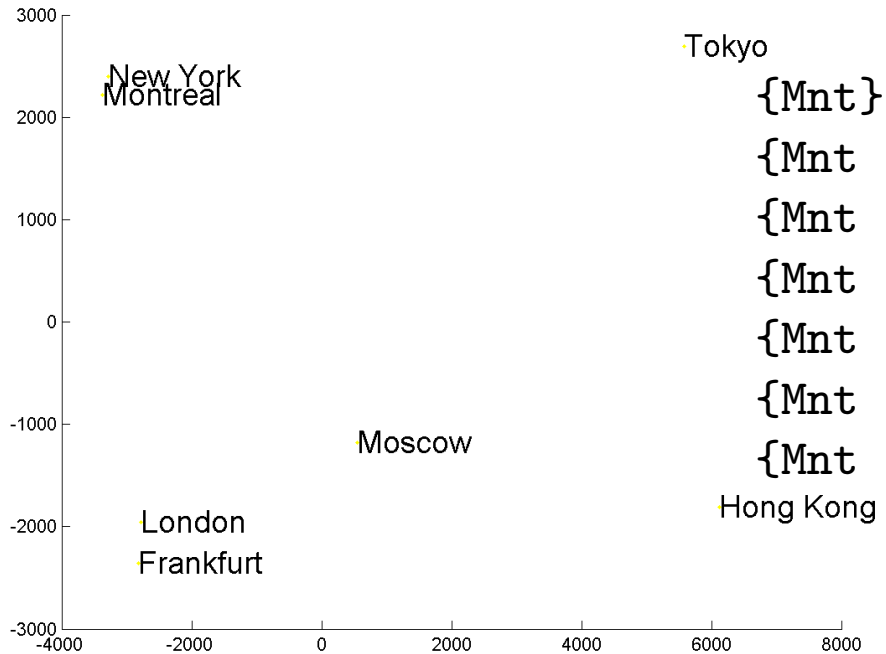
- Start with all points in their own groups
- Until there is only one cluster, repeatedly: merge the two groups that have the smallest dissimilarity

Divisive (top-down)- less popular

- Start with all points in one big cluster
- Until all points are in their own clusters, repeatedly: split the group into two, resulting in the biggest dissimilarity

Example: hierarchical clustering

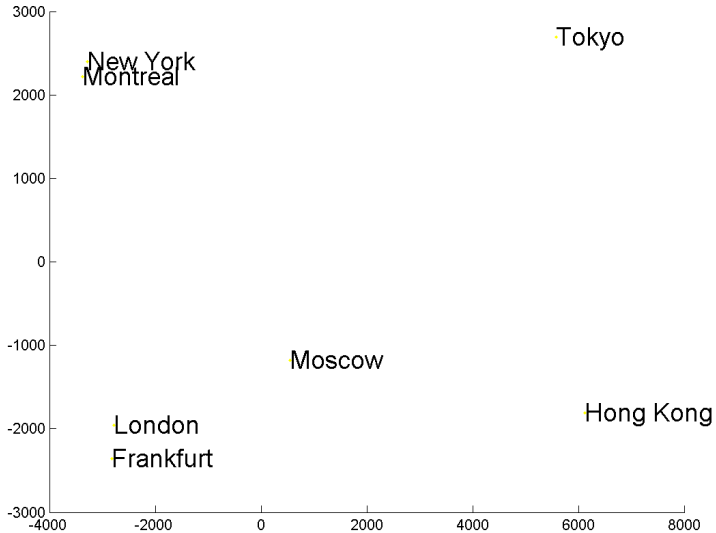
Given airline distances in miles between seven major cities ($n = 7$, dissimilarity is the airline distance), a hierarchical clustering gives a clustering sequence:

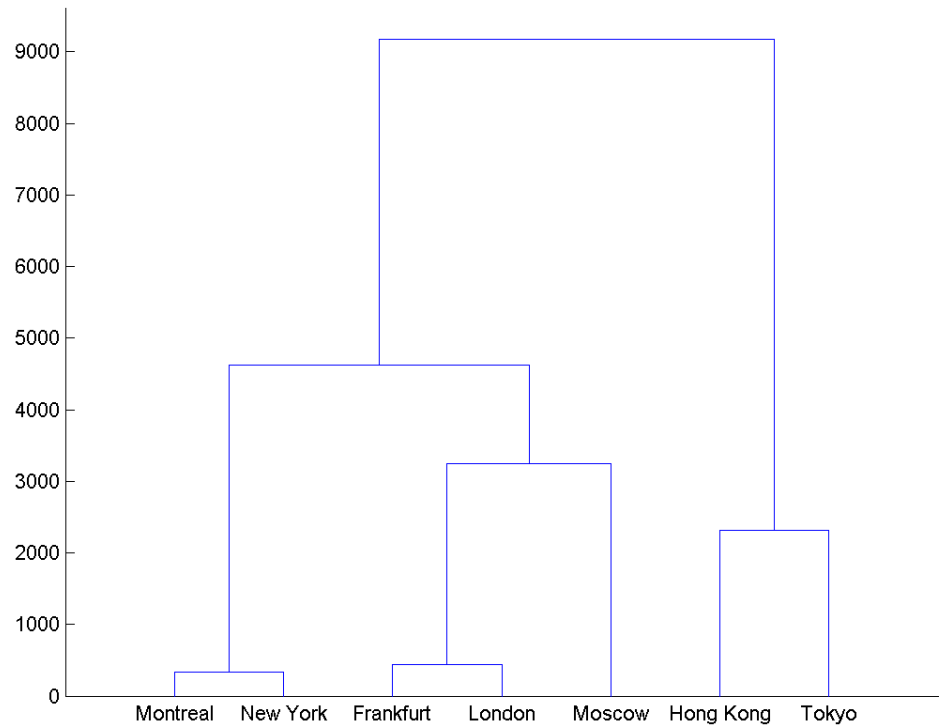



```
{Mnt} {NY} {Fr} {Lnd} {Mos} {HK} {Tk}
{Mnt   NY} {Fr} {Lnd} {Mos} {HK} {Tk}
{Mnt   NY} {Fr   Lnd} {Mos} {HK} {Tk}
{Mnt   NY} {Fr   Lnd} {Mos} {HK   Tk}
{Mnt   NY} {Fr   Lnd   Mos} {HK   Tk}
{Mnt   NY   Fr   Lnd   Mos} {HK   Tk}
{Mnt   NY   Fr   Lnd   Mos   HK   Tk}
```

Example: hierarchical clustering

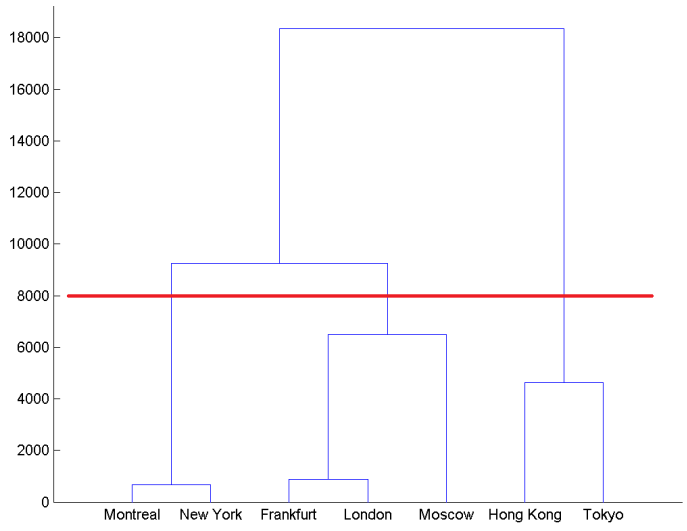
The sequence of clustering assignments is visually represented by a *dendrogram*:

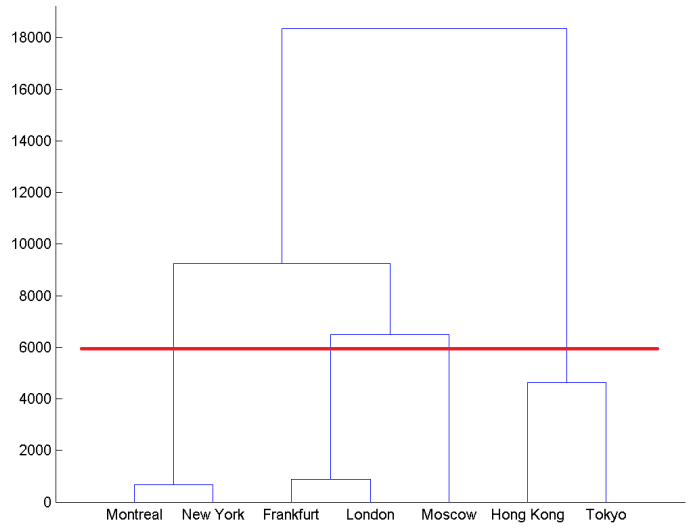




Note that cutting the dendrogram horizontally partitions the data points into clusters.

Dendrogram

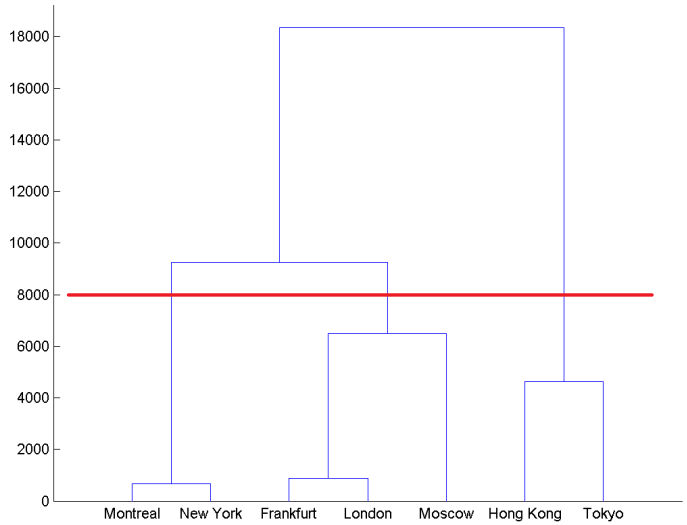


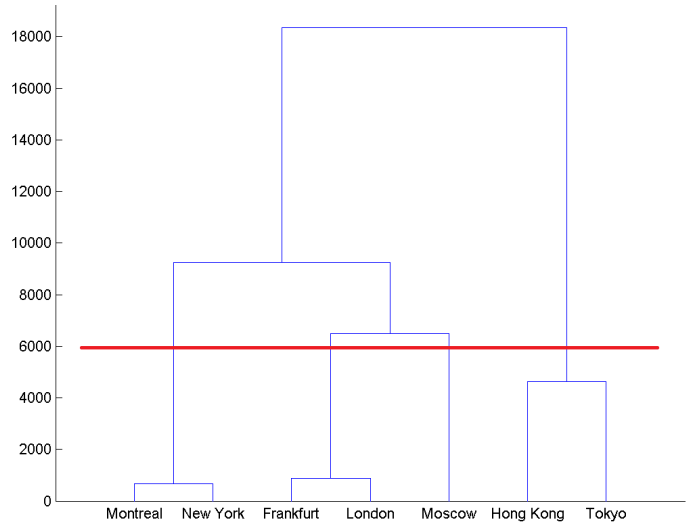


Dendrogram: Graphical representation of hierarchical sequence of clustering assignments.

- Vertical axis: **distance** *between clusters*
- Horizontal axis: observations
- Dendrogram is a binary tree where
  - Each node represents a cluster
  - Each leaf node is an observation
  - Root node is the whole data with all observations.

Dendrogram





Number of Clusters

Tree-Cuts: use `cutree` function in R.

distance between clusters

Agglomerative (bottom-up) hierarchical clustering needs a measure of distance between two clusters.

- We have dissimilarities d_{ij} between any pair of observations i and j.
- Clusters $G_1 = \{1, 2, 4, 6\}$ and $G_2 = \{3, 5\}$ (an example)
- *Linkage*: function $d(G_1, G_2)$ that takes two groups G_1, G_2 and returns a dissimilarity score between them
  - [Single linkage (nearest-neighbor linkage)]

$$d(G_1, G_2) = \min_{i \in G_1, j \in G_2} d_{ij}$$

  - [Complete linkage (furthest-neighbor linkage)]

$$d(G_1, G_2) = \max_{i \in G_1, j \in G_2} d_{ij}$$

  - [Average linkage]

$$d(G_1, G_2) = \text{Average}_{i \in G_1, j \in G_2} d_{ij} = \sum_{i \in G_1, j \in G_2} d_{ij} / (|G_1| \cdot |G_2|)$$

Agglomerative hierarchical clustering

Agglomerative hierarchical clustering algorithm

Input $D = (d_{ij})$, the $n \times n$ (symmetric) matrix of dissimilarities $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ between the n clusters, given a linkage $d(G, H)$.

1. Merge the two clusters G and H whose $d(G, H)$ is the smallest.
2. With the new cluster (GH) and remaining clusters, repeat Step 1 until there is only one cluster.

A worked example: Airline distances

$n = 7$. Consider using Single Linkage:

`D(7) =`

	Fr	HK	Lnd	Mnt	Mos	NY	Tk
Fr	0	8277	400	3640	1253	3851	9776
HK	8277	0	8252	10345	6063	10279	1788
Lnd	400	8252	0	3251	1557	3456	9536
Mnt	3640	10345	3251	0	5259	330	8199
Mos	1253	6063	1557	5259	0	5620	4667
NY	3851	10279	3456	330	5620	0	8133
Tk	9776	1788	9536	8199	4667	8133	0

Merge two clusters `Mnt` and `NY` as `d(Mnt, NY)` smallest
Compute new $(n-1) \times (n-1)$ dissimilarity matrix

A worked example: Airline distances

Compute new 6×6 dissimilarity matrix with d(`MntNY`, $\cdot$) being the single linkage

`D(6) =`

	MntNY	Fr	HK	Lnd	Mos	Tk
MntNY	0	3640	10279	3251	5259	8133
Fr	3640	0	8277	400	1253	9776
HK	10279	8277	0	8252	6063	1788
Lnd	3251	400	8252	0	1557	9536
Mos	5259	1253	6063	1557	0	4667
Tk	8133	9776	1788	9536	4667	0

Merge two clusters `Fr` and `Lnd` as `d(Fr, Lnd)` smallest

A worked example: Airline distances

Compute new 5×5 dissimilarity matrix

`D(6) =`

	FrLnd	MntNY	HK	Mos	Tk
FrLnd	0	3251	8252	1253	9536
MntNY	3251	0	10279	5259	8133
HK	8252	10279	0	6063	1788
Mos	1253	5259	6063	0	4667
Tk	9536	8133	1788	4667	0

Merge two clusters `FrLnd` and `Mos` as `d(FrLnd, Mos)` smallest

A worked example: Airline distances

Compute new 4×4 dissimilarity matrix

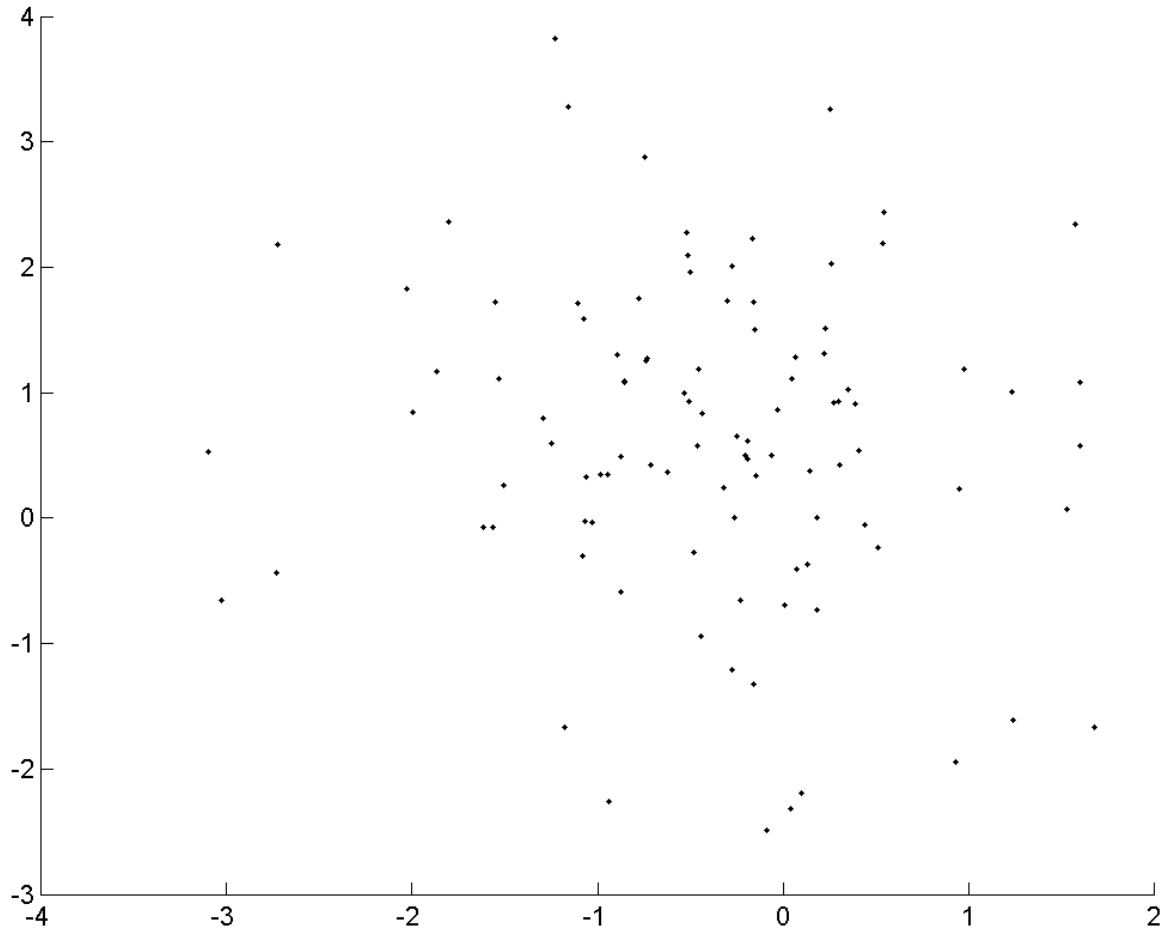
D(4) =

	FrLndMo	MntNY	HK	Tk
FrLndMo	0	3251	6063	4667
MntNY	3251	0	10279	8133
HK	6063	10279	0	1788
Tk	4667	8133	1788	0

Repeat until there is only one cluster.

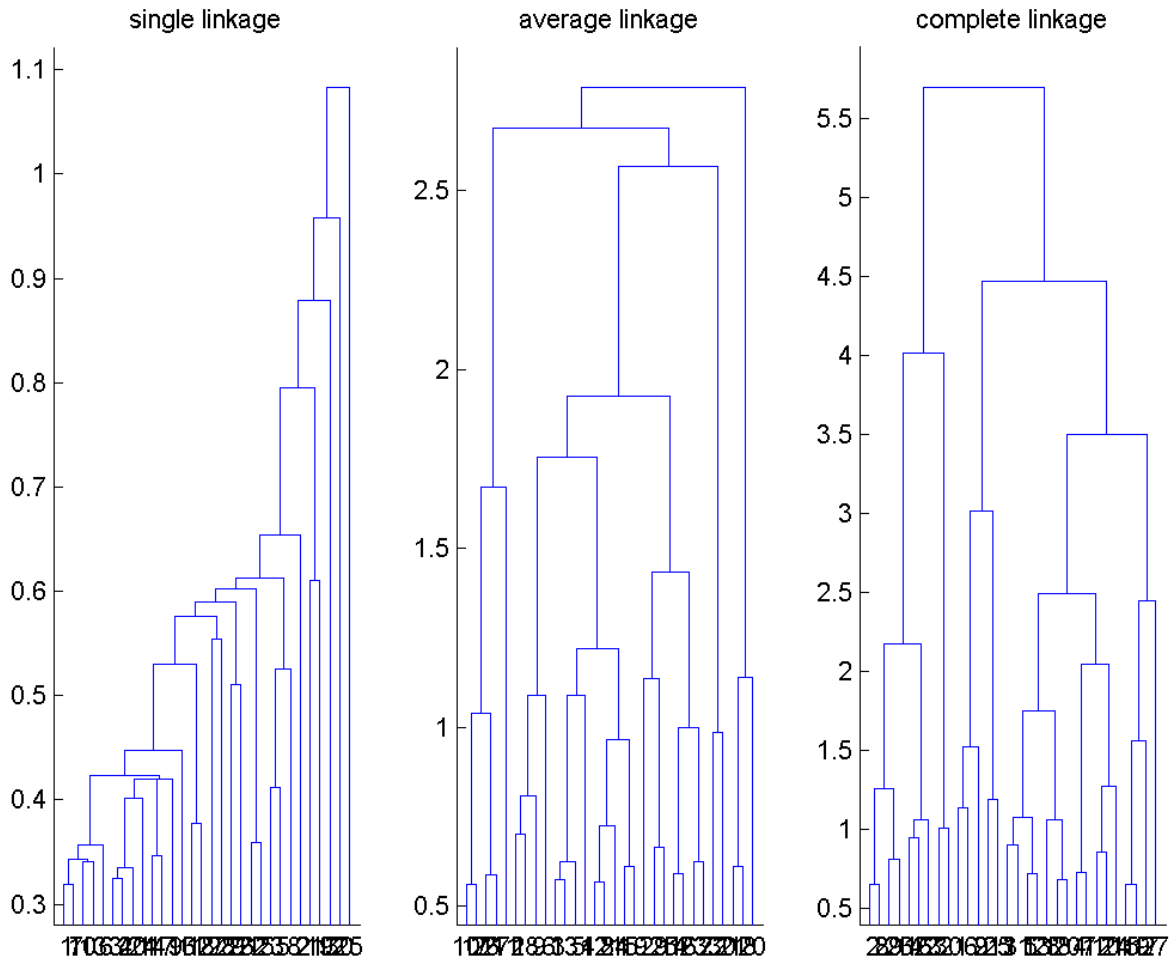
Another example

- Randomly generated data



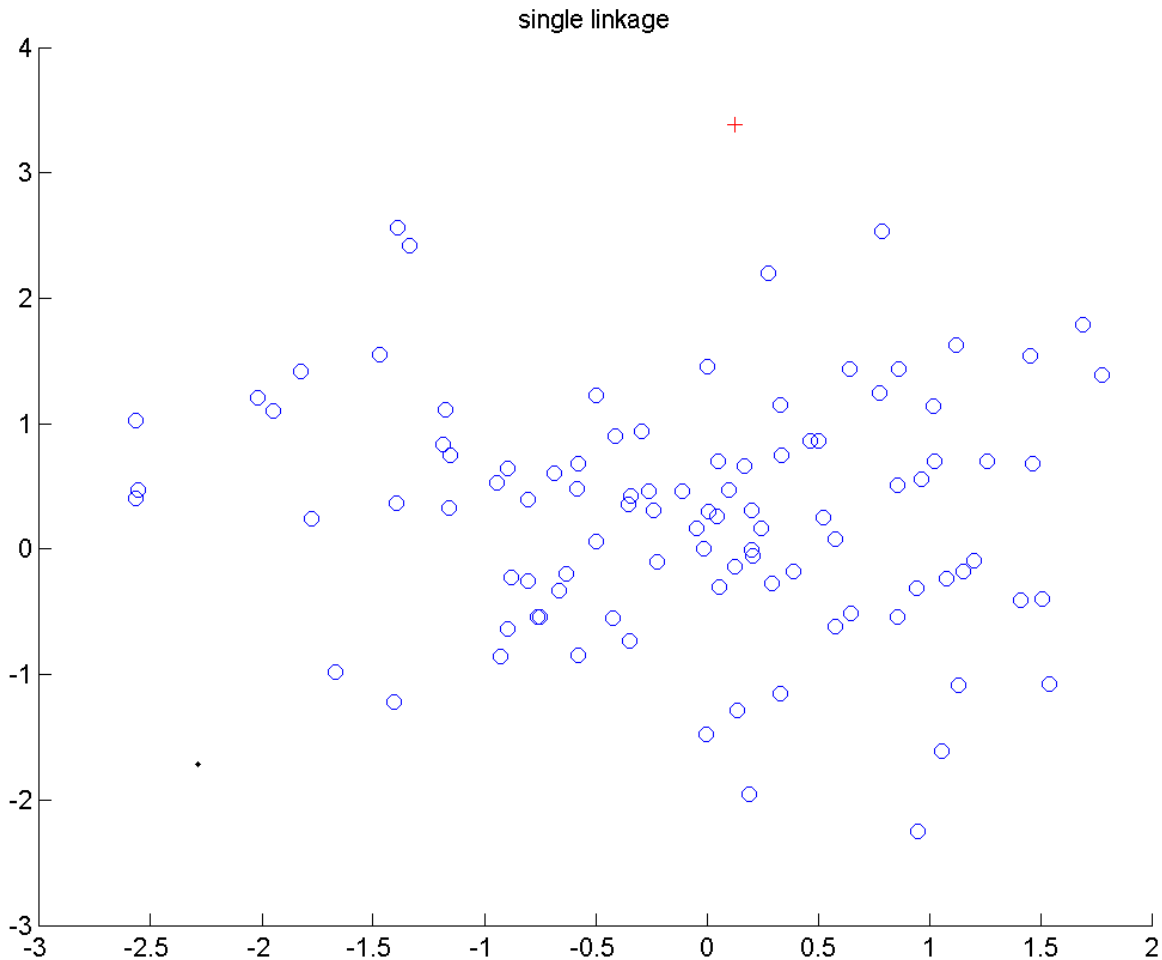
Another example: Dendrograms

- Three different linkage–single, average and complete
- Compare cluster assignments with three clusters



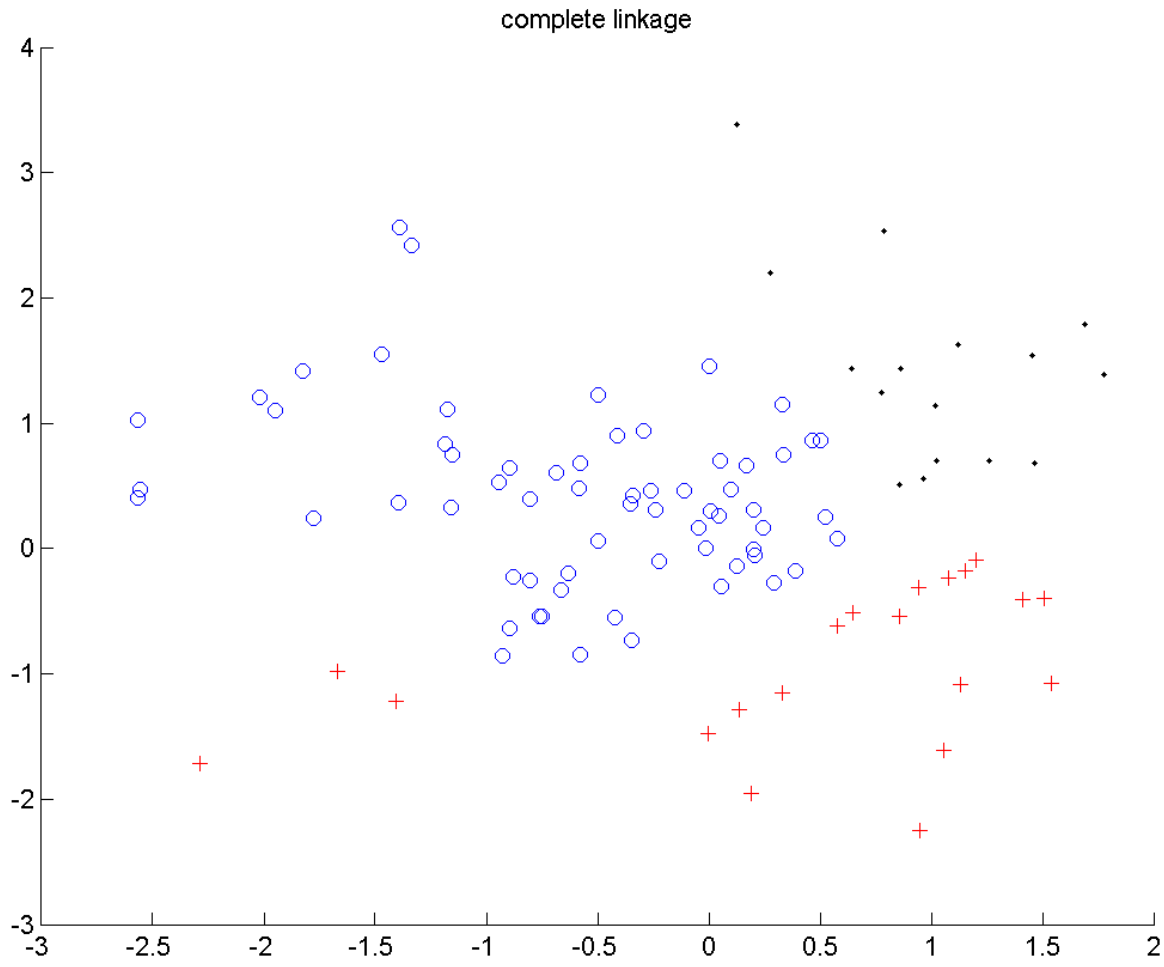
Cluster assignments by Single Linkage

- Tends to leave single points as clusters
- Suffers from *chaining* (clusters spread out, not compact)



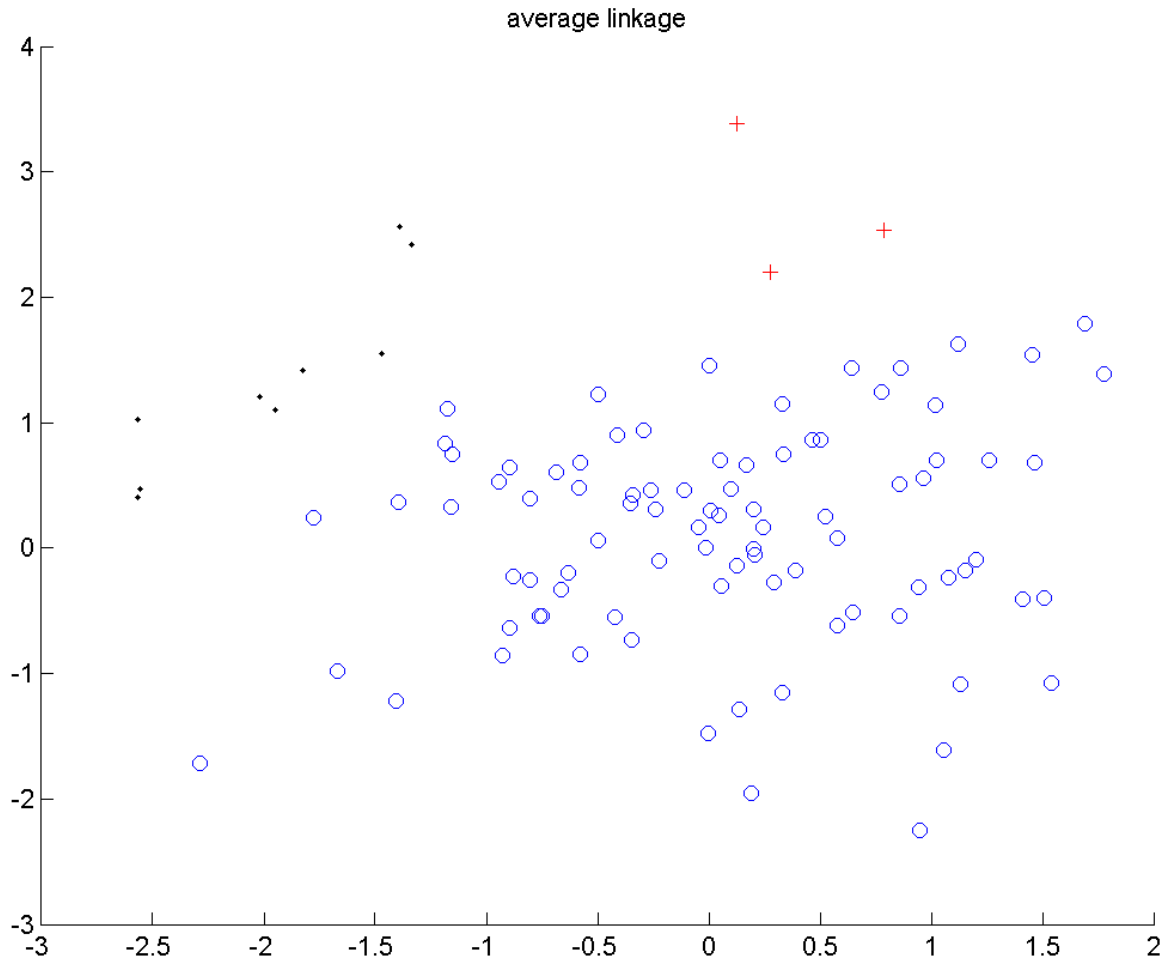
Cluster assignments by Complete Linkage

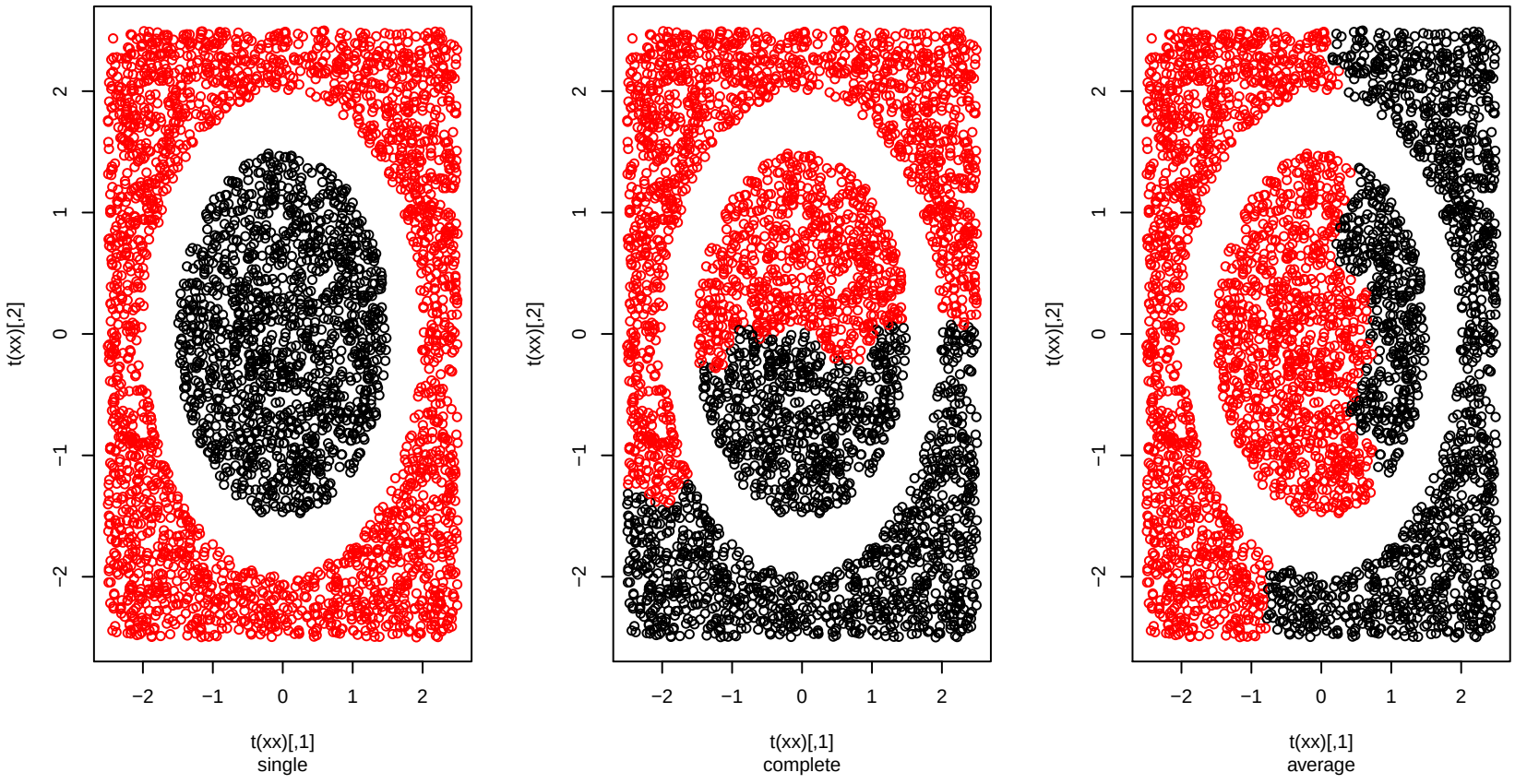
- Can have a disjoint cluster
- Suffers from *crowding* (a point can be closer to points in other clusters than to points in its own cluster)



Average Linkage

- A good balance – relatively compact, relatively far apart




t(xx)[,2]
t(xx)[,1]
single
t(xx)[,2]
t(xx)[,1]
complete
t(xx)[,2]
t(xx)[,1]
average

Linkage functions

When deciding which cluster to merge

- Single: short sighted.
- Complete: long sighted.
- Average: average.

Discussion:

- When are different linkages appropriate?
- More robust?

More Dissimilarities

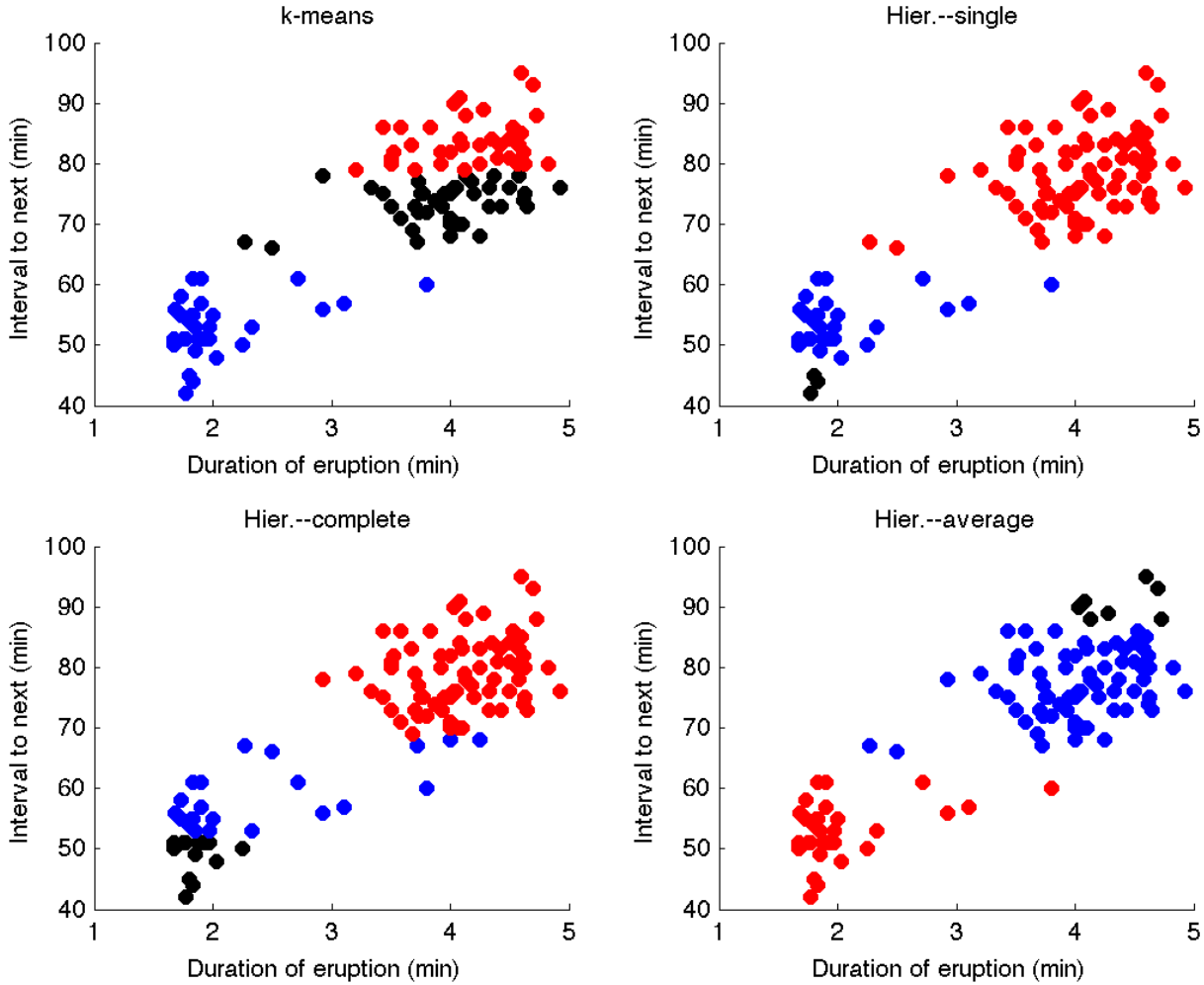
For quantitative variables. For $\boldsymbol{x}, \boldsymbol{y} \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^d$,

- p-norm $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_p = \left(\sum_{i=1}^d \|x_i - y_i\|^p\right)^{1/p}$, $p > 0$.
- Standardized distance $d^2(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^d \frac{\|x_i - y_i\|^2}{s_i^2} = (\boldsymbol{x} - \boldsymbol{y})' \mathbf{D}^{-1} (\boldsymbol{x} - \boldsymbol{y})$, where s_i is the standard deviation of ith measurements and $\mathbf{D}$ is the diagonal matrix consisting of diagonal elements of (sample) covariance matrix $\mathbf{S}$.
- Mahalanobis distance $d^2(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} - \boldsymbol{y})' \mathbf{S}^{-1} (\boldsymbol{x} - \boldsymbol{y})$.
- Many others...

Different distances lead to different clustering, as seen in the next few slides.

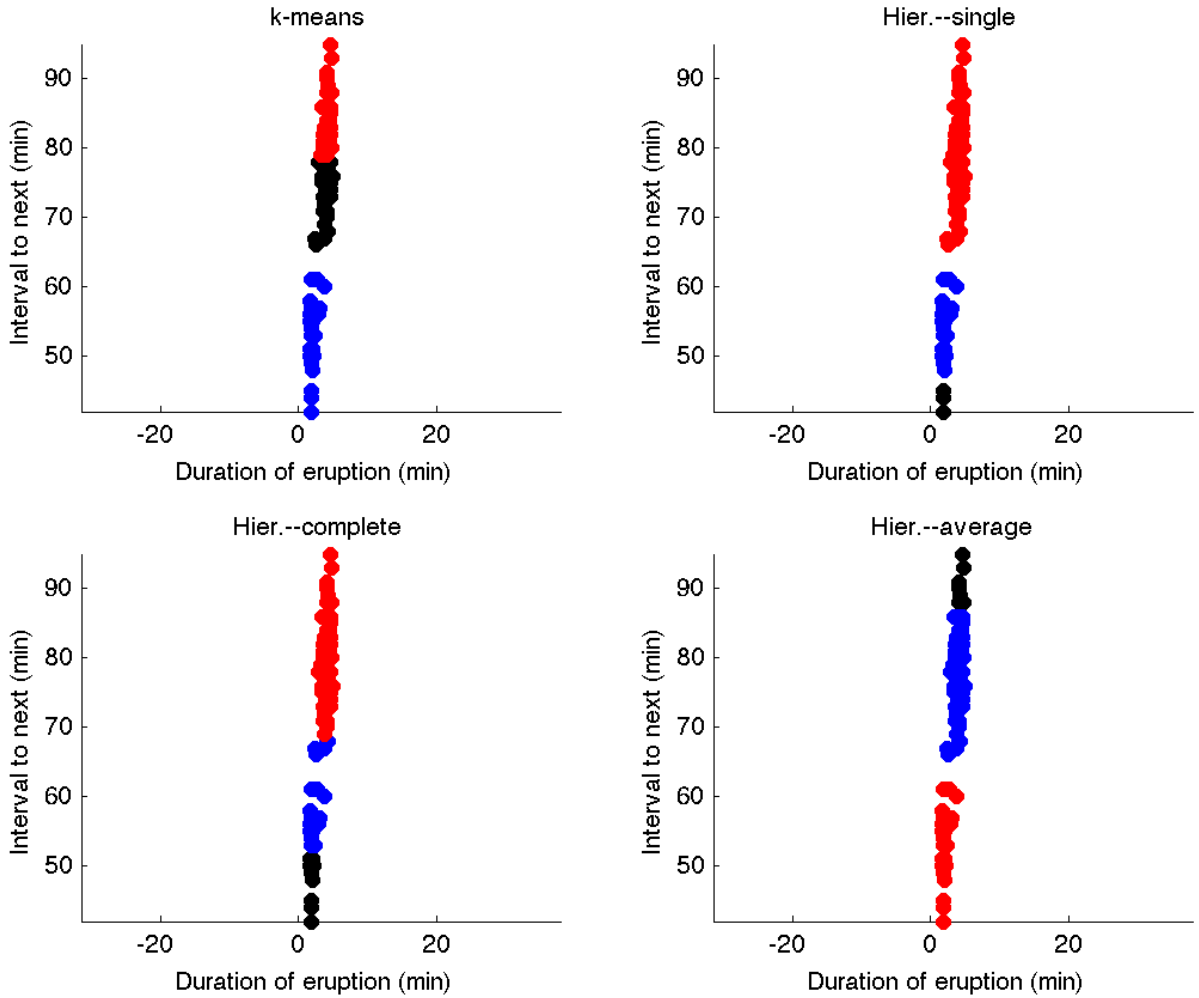
Choice of Dissimilarities (Geyser)

Using squared 2-norm:



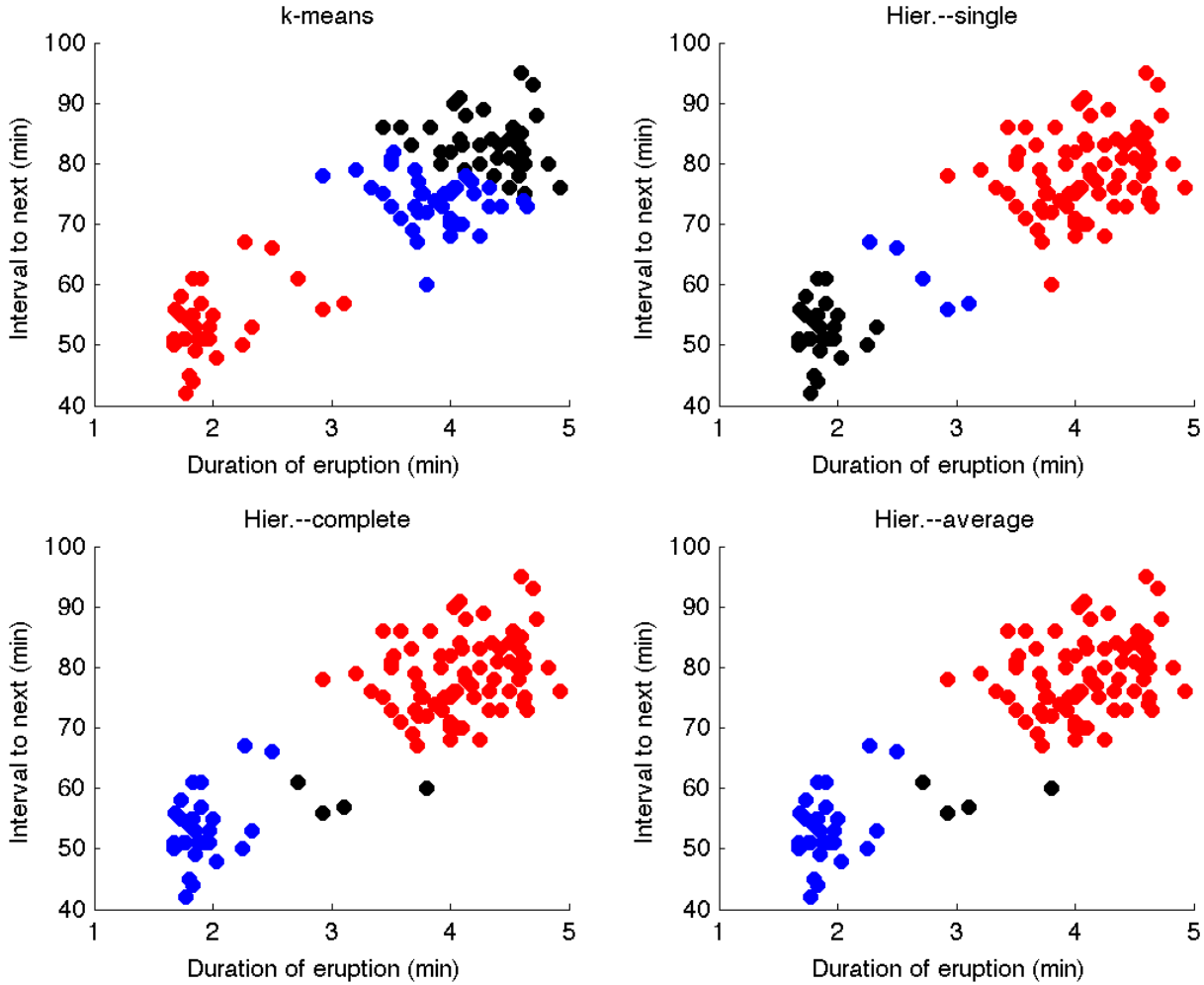
Choice of Dissimilarities (Geyser)

Using squared 2-norm (squared Euclidean distance)



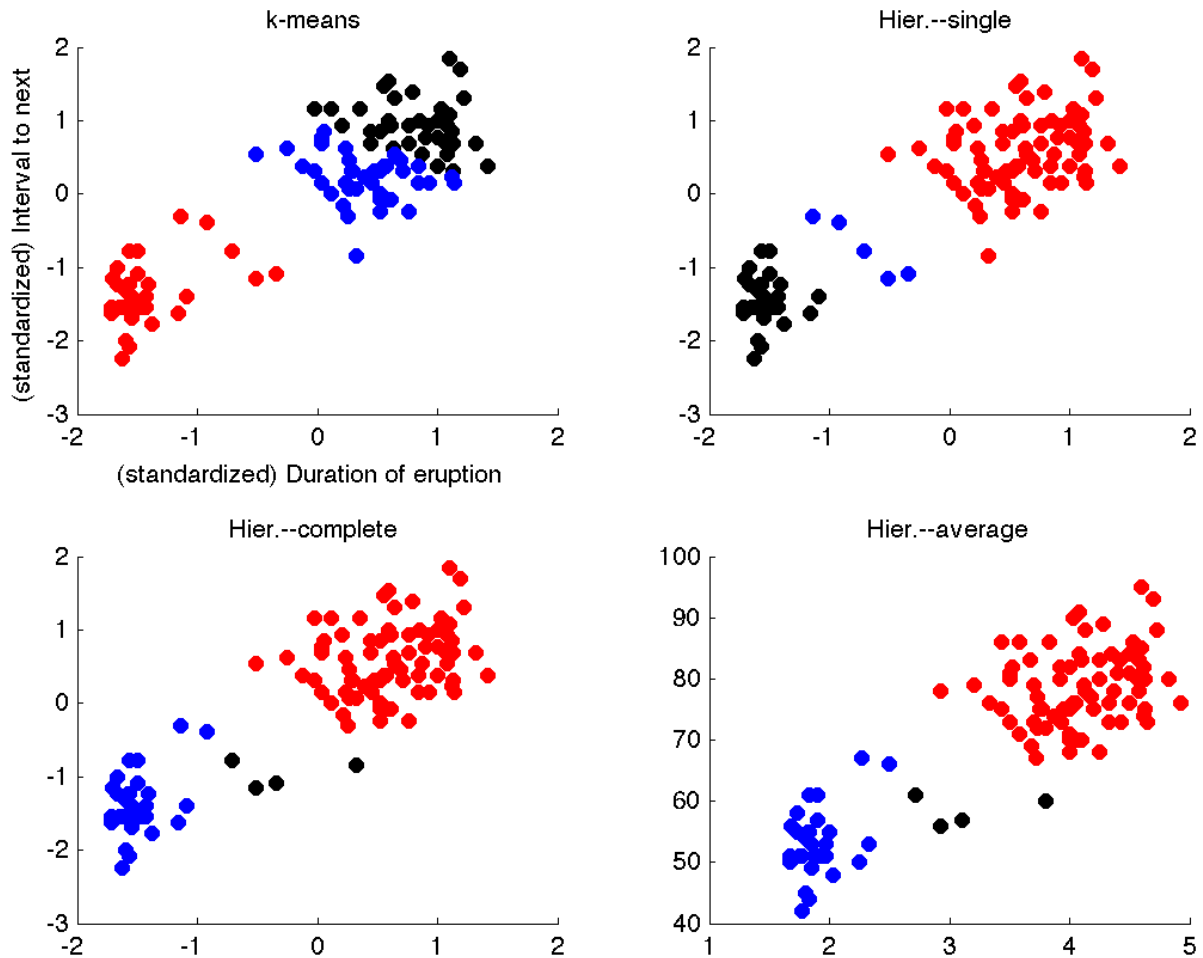
Choice of Dissimilarities (Geyser)

Using Standardized distance:



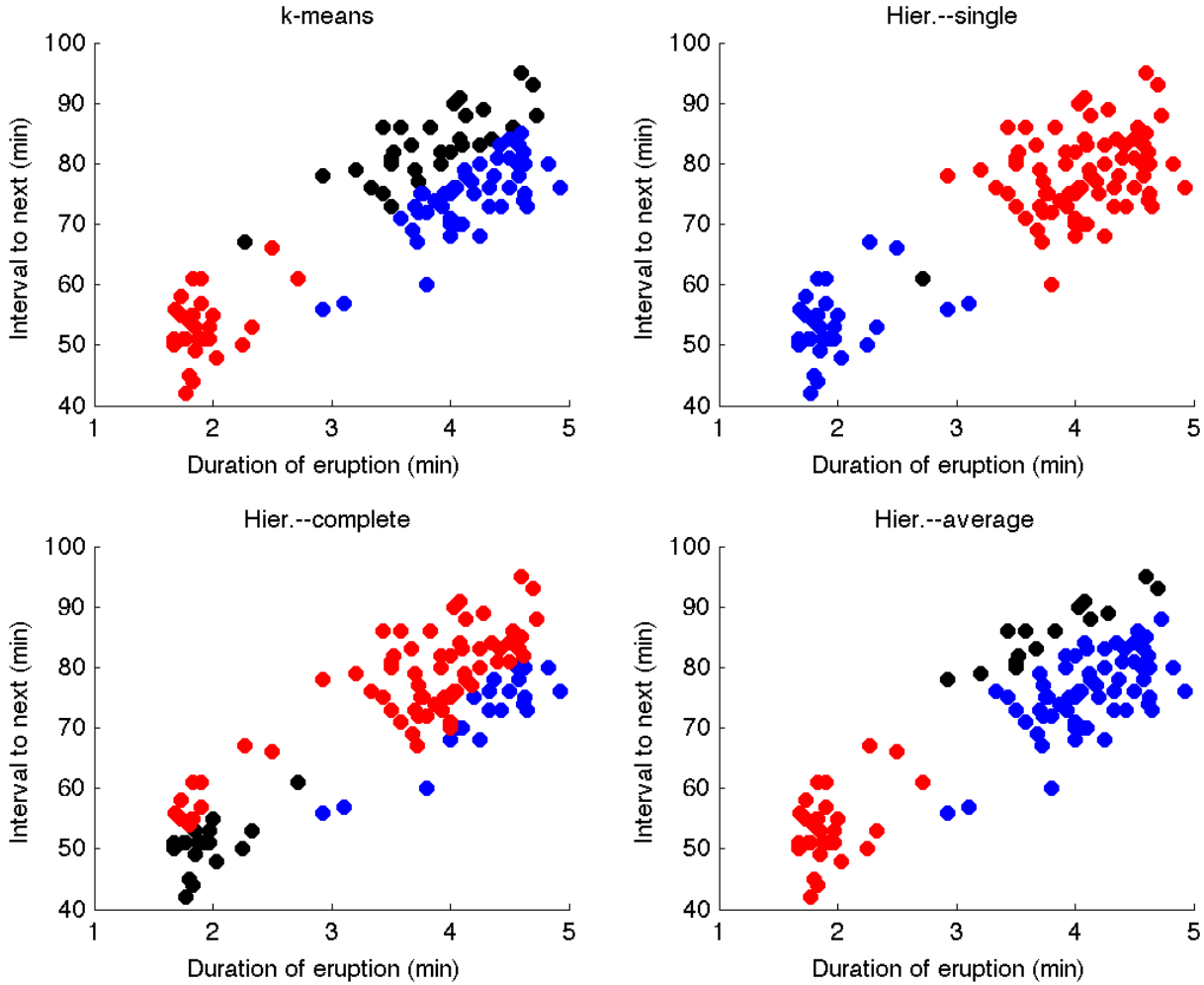
Choice of Dissimilarities (Geyser)

Using Standardized distance – equivalent to using Euclidean distance for standardized variables



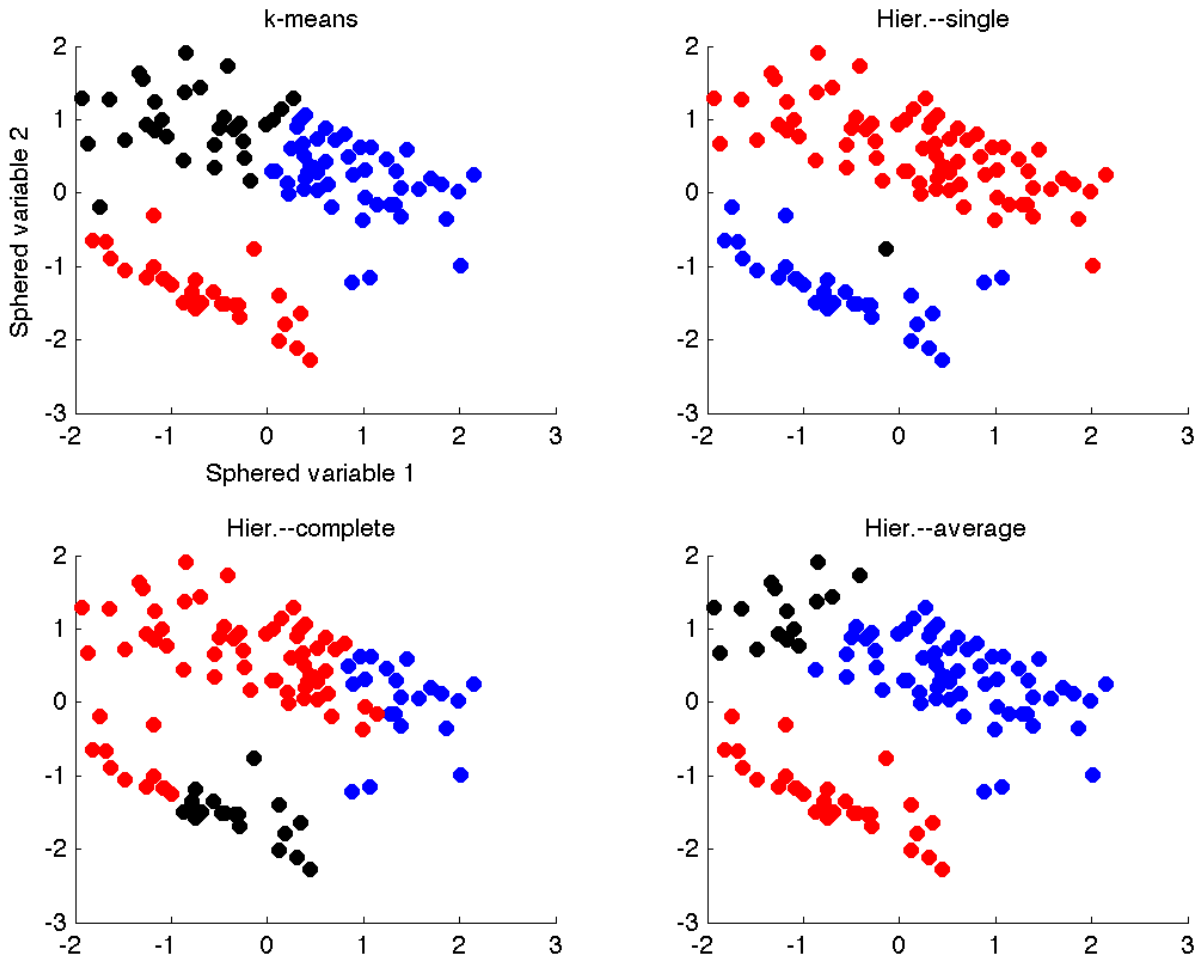
Choice of Dissimilarities (Geyser)

Using Mahalanobis distance:



Choice of Dissimilarities (Geyser)

Using Mahalanobis distance – equivalent to using squared Euclidean distance for *sphered* variables



Summary

Strengths:

- Simple / intuitive.
- Visualization.
- Family of possible clusterings (nested).

Extremely popular!!

Weaknesses:

- Unstable Solution (small perturbation to data leads to big difference).
- Depends heavily on type of linkage.
- No optimization criterion - purely algorithmic.

The next section would be . . . . . .

1. Combinatorial algorithm
2. K-means and related methods
3. Hierarchical clustering
4. Other topics

Clustering variables

So far we have focused on clustering subjects (or individuals). Variables (Measurements) can also be grouped into several clusters. We only need to have dissimilarity between variables. Common choices are:

- 1−correlation:

$$d(V_i, V_j) = 1 - \rho_{ij}$$

  where ρ_{ij} is the correlation coefficient between r.vs V_i and V_j.

- 1−squared-correlation:

$$d(V_i, V_j) = 1 - \rho_{ij}^2.$$

One can then use the dissimilarity-based clustering algorithms with dissimilarity matrix $D_{(p\times p)} = (d_{ij})$, $d_{ij} = d(V_i, V_j)$.

Bi-clustering

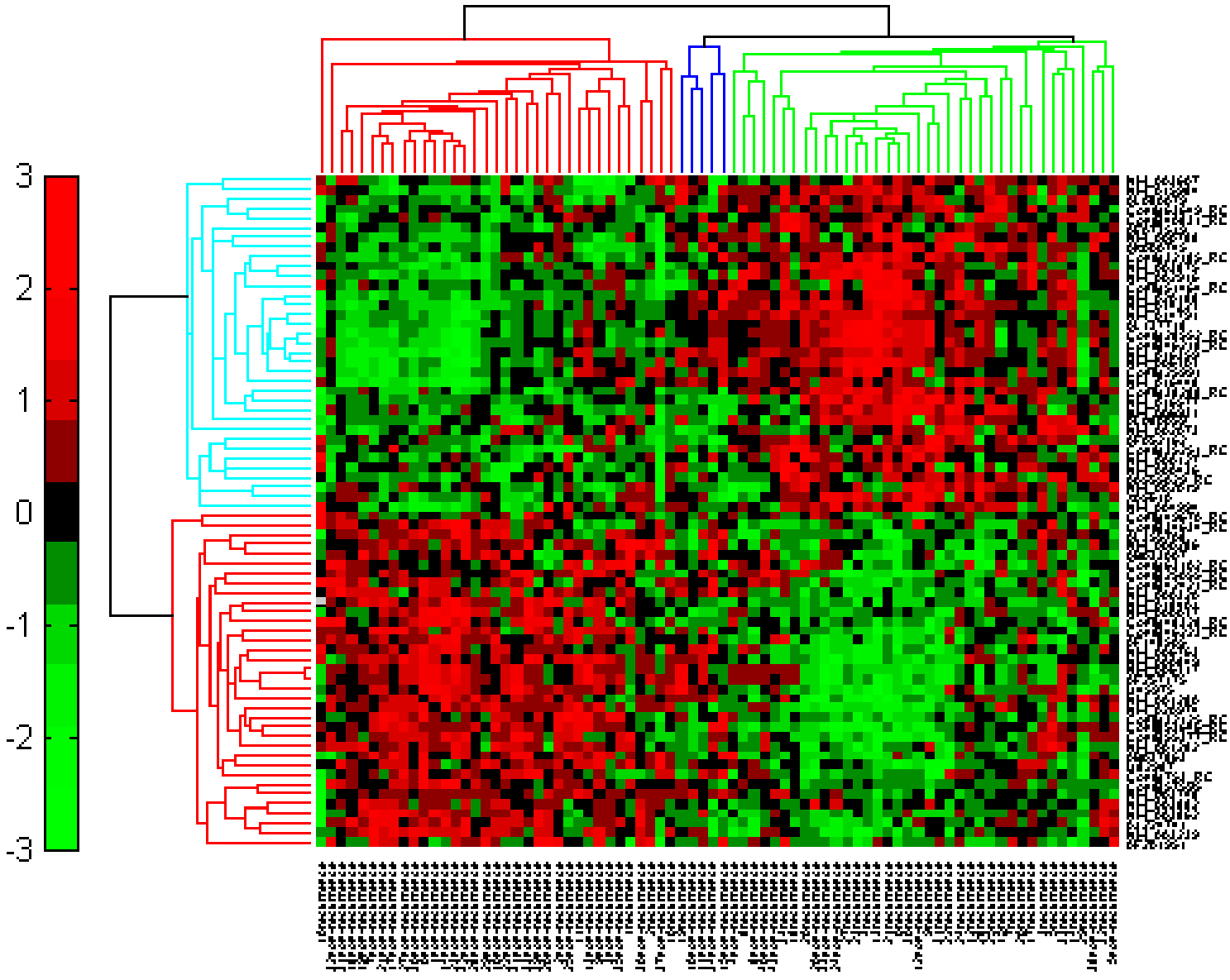
- data matrix p by n
- Biclustering is a technique used to simultaneously cluster both the rows and columns of a data matrix.
- determines a subset of columns that are distinguished, by some measure, on a subset of rows.
- Example: some genes are active for a certain group of patients; while another set of genes are active for second group of patients. Gene sets and patient groups may even have overlaps.
- Can do variable clustering and observation cluster separately. Works for some cases.
- Section 12.8 Izenman

Bi-clustering

Two main types:

- Overlapping Biclusters.
  - Plaid models & Sparse SVD models.
- Non-overlapping Biclusters (Checkerboard mean).
  - Cluster heatmap. (`heatmap` in R)

Hierarchical Clustering Separately on Rows & Columns. Reorder the rows and columns and use heatmap to represent data.

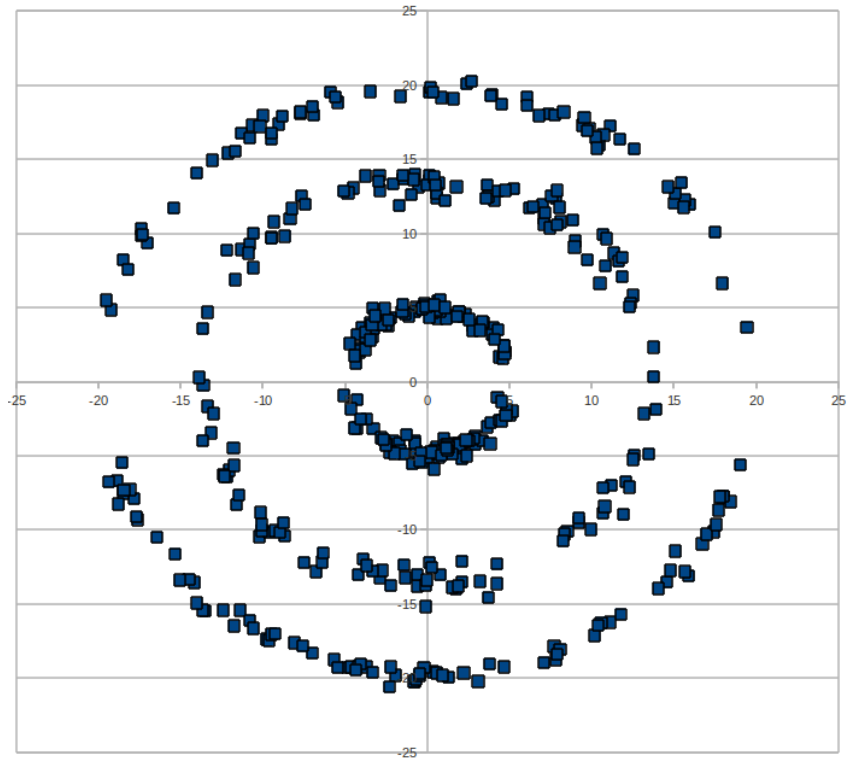


Biclustering - Applications

- Biomedicine - 'omics' data
  - Cancer genomics: Finding subtypes. Find groups of patients (subtypes) and groups of genes (genomic signatures) that separate subtypes.
- Text mining.
  - Word-Document associations.
- Collaborative Filtering.
  - Find users who highly rate particular products: Netflix, Amazon.

Spectral Clustering

- Useful when spherical metric fails.
- Suitable when the clusters are non-convex; such as donuts



- Adjacency matrix is a n by n matrix with elements that describe the neighbors of an observation and how close they are to it.
- Let $\mathbf{W}$ be the adjacency matrix and $\mathbf{J}$ is the degree matrix. Define $\mathbf{L}_{n\times n} = \mathbf{J} - \mathbf{W}$ as the graph Laplacian
- Choose the first m eigenvectors of $\mathbf{L}$ corresponding to a few smallest eigenvalues, $\mathbf{Z}_{n\times m}$; Then use standard clustering algorithms to cluster the rows of $\mathbf{Z}_{n\times m}$
- Motivation: recall that $\boldsymbol{z}'\mathbf{L}\boldsymbol{z}$ is the eigenvalue. It turns out that $\boldsymbol{z}'\mathbf{L}\boldsymbol{z} = \frac{1}{2}\sum_{i=1}^{n}\sum_{i'=1}^{n} w_{ii'}(z_i - z_{i'})^2$. Thus if this value is small, it must be that observations i and i' with large adjacency $w_{ii'}$ happen to have small $|z_i - z_{i'}|$. Therefore we cluster z_i's so that the z_i's in the same cluster correspond to observations with large adjacancy.
- Read ESL Section 14.5.3

Convex Clustering & Biclustering

Can we formulate a convex method for clustering that will yield a **unique & global** solution?

- Convex Clustering

$$\min_{\boldsymbol{u}} \ \frac{1}{2}\sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{u}_i\|_2^2 + \lambda \sum_{i<j} w_{ij} \|\boldsymbol{u}_i - \boldsymbol{u}_j\|_2$$

- Convex Biclustering

$$\min_{\boldsymbol{u}} \ \frac{1}{2}\|\mathbf{X}-\mathbf{U}\|_F^2 + \lambda \left(\sum_{i<j} w_{ij} \|\mathbf{U}_{i\cdot} - \mathbf{U}_{j\cdot}\|_2 + \sum_{i<j} \tilde{w}_{ij} \|\mathbf{U}_{\cdot i} - \mathbf{U}_{\cdot j}\|_2 \right)$$

`cvxclustr` and `cvxbiclustr` in R.

Summary

Strengths:

- Unique, global solution.
- Stable solution.
- Fast algorithm.
- Family of clustering solutions.
- One tuning parameter.

Weaknesses:

- Performance can depend on weights.

Computation (R)

kmeans and agglomerative hierarchical algorithms are part of base R distribution. For analysis of gap statistics, use `clusGap` in `cluster` package

```
k <- 3
kmeansobj<-kmeans(iris[1:4],k)

d = dist(iris[1:4])
tree.avg = hclust(d, method="average")
plot(tree.avg)
membership <- cutree(tree.avg, k = 3)

library(cluster)
gap <- clusGap(iris[1:4], FUN = kmeans, K.max = 8)
plot(gap)
```

EM: use Mclust

```
library(mclust)                          # load mclust library
x1 = rnorm(n=20, mean=1, sd=1)           # get 20 normal distribute
y1 = rnorm(n=20, mean=1, sd=1)           # get 20 normal distribute
x2 = rnorm(n=20, mean=5, sd=1)           # get 20 normal distribute
y2 = rnorm(n=20, mean=5, sd=1)           # get 20 normal distribute
rx = range(x1,x2)                        # get the axis x range
ry = range(y1,y2)                        # get the axis y range
plot(x1, y1, xlim=rx, ylim=ry)           # plot the first class poi
points(x2, y2)                           # plot the second class po
mix = matrix(nrow=40, ncol=2)            # create a dataframe matri
mix[,1] = c(x1, x2)                      # insert first class point
mix[,2] = c(y1, y2)                      # insert second class poin
mixclust = Mclust(mix)                   # initialize EM with hiera
plot(mixclust, data = mix)               # plot the two distinct cl
```

```
> mixclust$classification
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2
> mixclust$parameters
$Vinv
NULL
$pro
[1] 0.4996775 0.5003225
$mean
         [,1]     [,2]
[1,] 1.122319 4.886524
[2,] 1.088972 5.210812
$variance
$variance$modelName
[1] "EII"
$variance$d
[1] 2
$variance$G
[1] 2
```

Computation (Matlab)

kmeans and agglomerative hierarchical algorithms are part of Statistics Toolbox. For gap statistics and other methods of evaluating number of clusters, use the very recent version (R2013b).