# $k$-means, hierarchical, and spectral clustering

Matthew S. Shotwell, Ph.D.

Department of Biostatistics
Vanderbilt University School of Medicine
Nashville, TN, USA

April 13, 2020

# k-means clustering

- Partition data into $k$ non-overlapping clusters
- Must specify $k$
- k-means algorithm creates $k$ clusters with smallest total within-cluster variance
- Thus Euclidean distance measures similarity
- This difficult combinatorial problem, but iterative algorithm does pretty good

# k-means algorithm

**Algorithm 10.1** *$K$-Means Clustering*

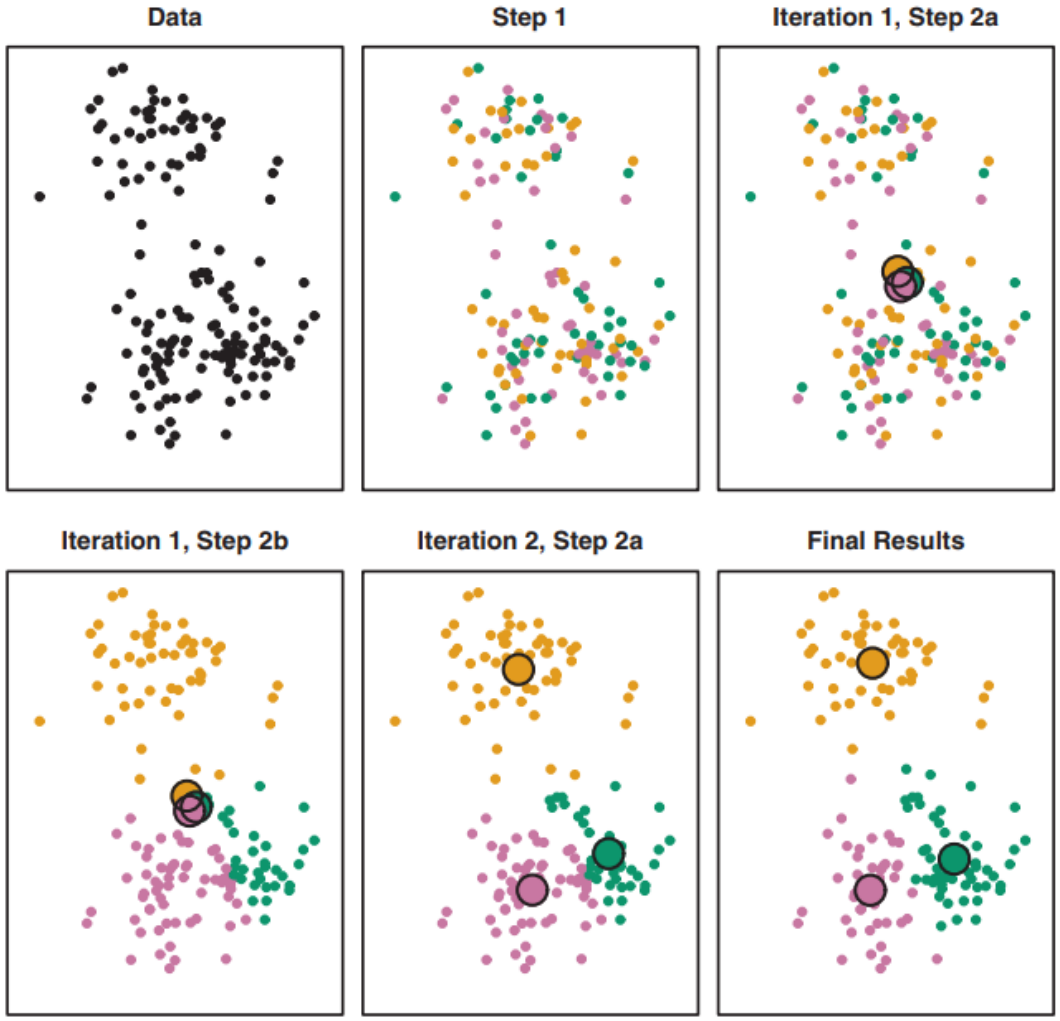
1. Randomly assign a number, from 1 to $K$, to each of the observations. These serve as initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing:

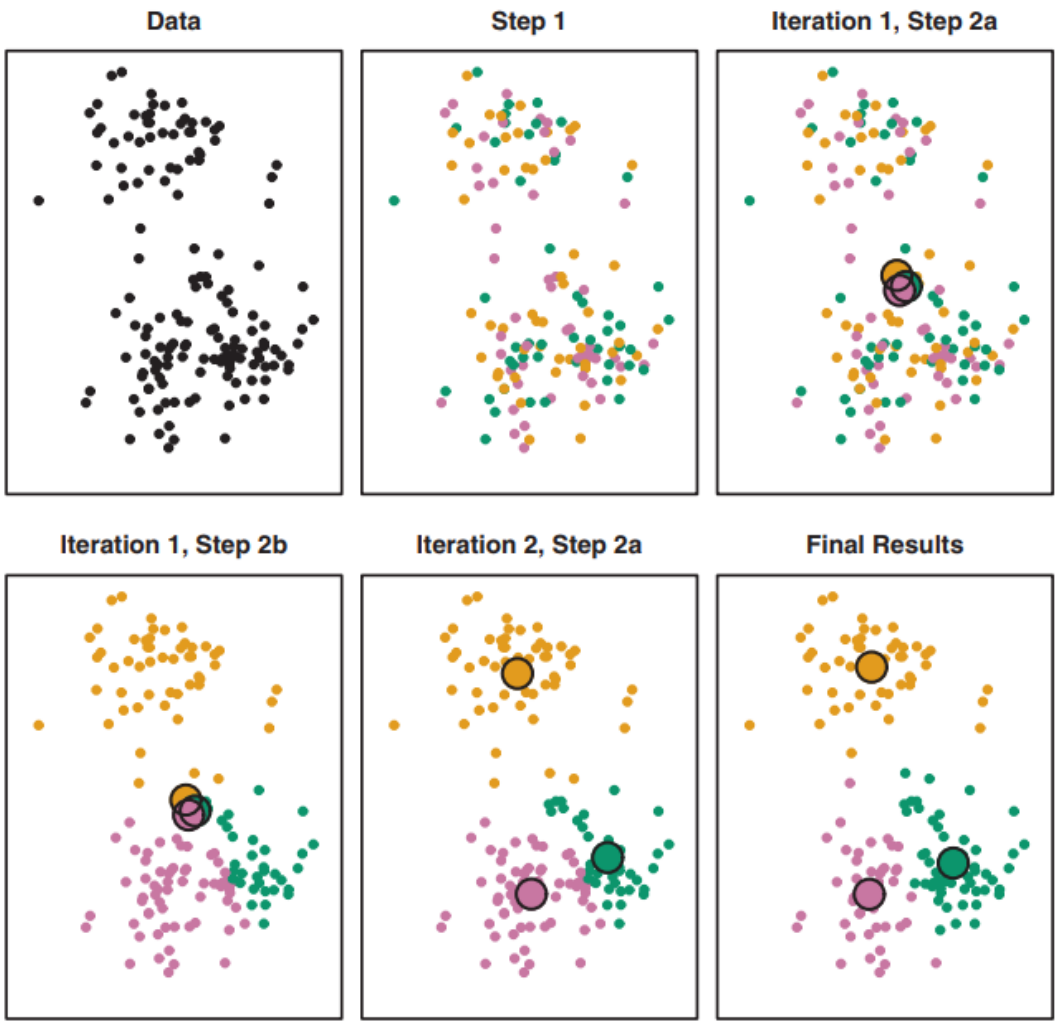
   (a) For each of the $K$ clusters, compute the cluster *centroid*. The $k$th cluster centroid is the vector of the $p$ feature means for the observations in the $k$th cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

# k-means algorithm

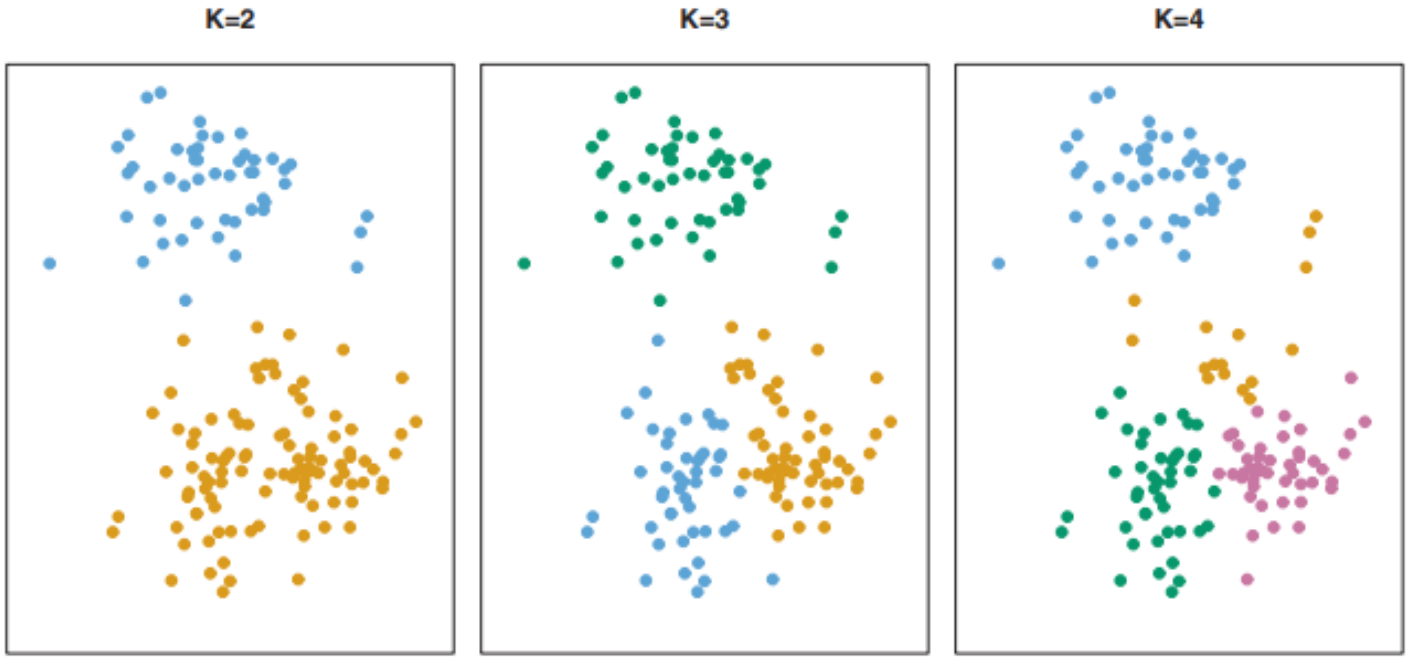


# k-means starting values




Need to do k-means several times, pick best

# k-means value of $k$



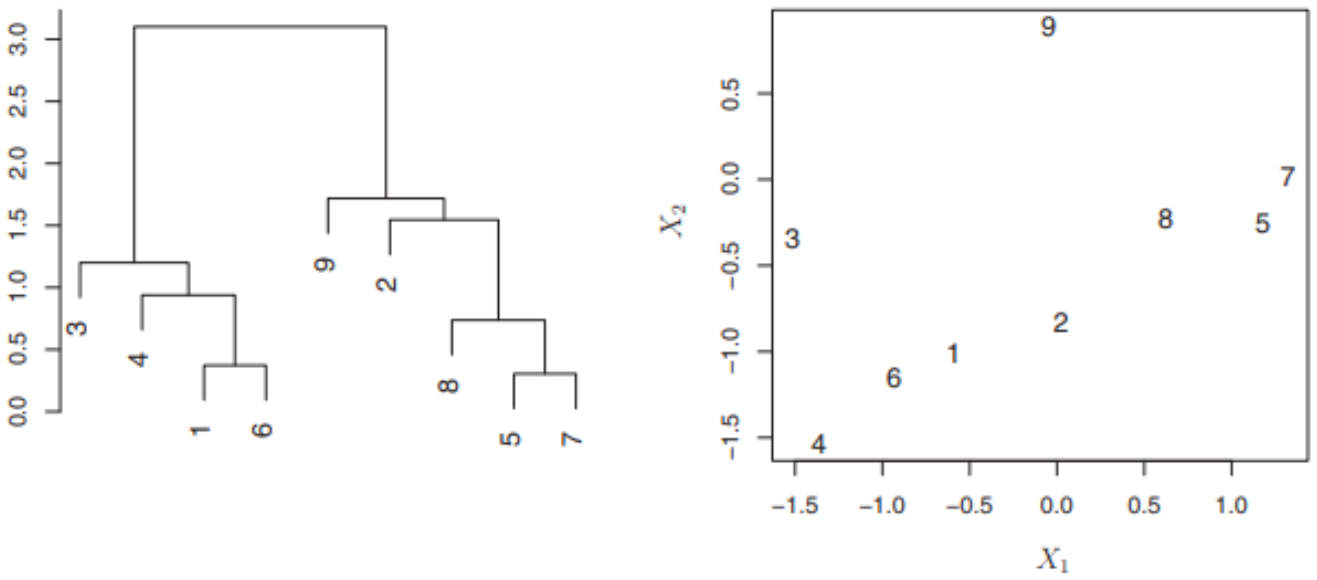
# k-means value of $k$

- As $k$ increases total within-cluster variance decreases
- Use domain-specific considerations
- Use the elbow rule if no outside info

# Hierarchical clustering

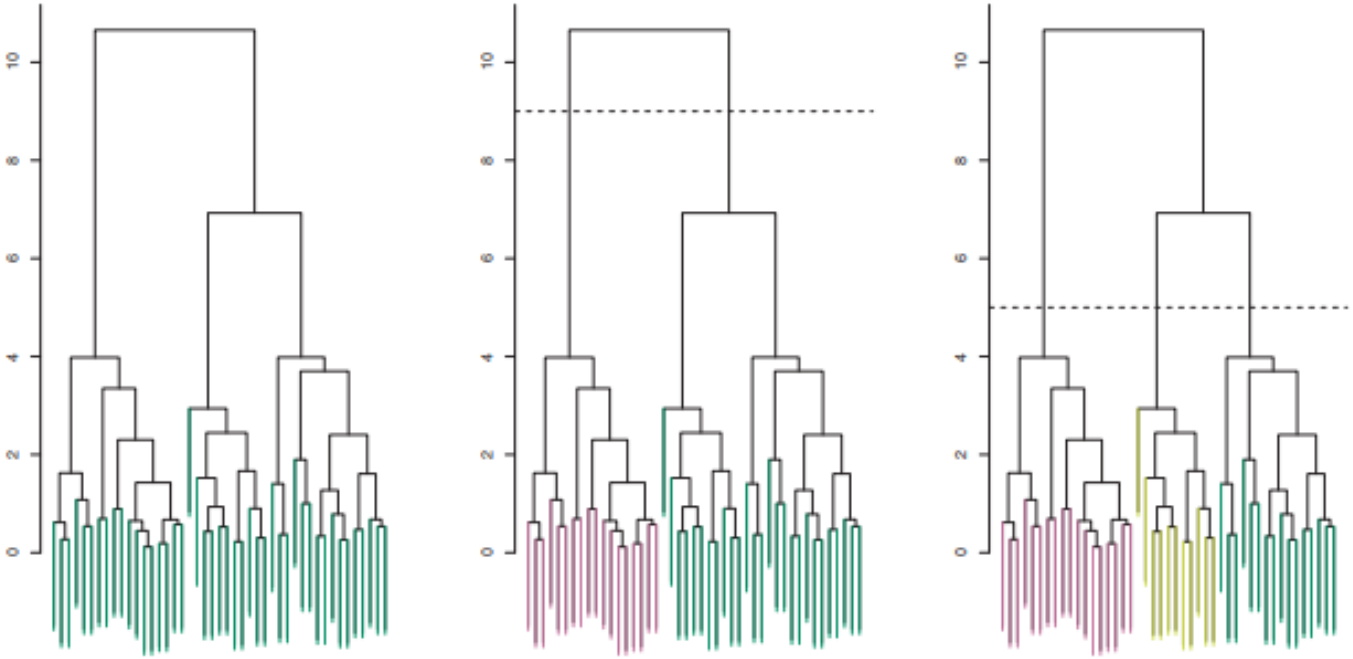
- Generates a sequence of clusters
- Split or merging clusters at each step
- Does not require prespecification of $k$
- Has tree-based representation: *dendrogram*
- Top-down and bottom-up (agglomerative) versions
- Top-down: start with 1 big cluster, split until N clusters
- Bottom-up: start with N clusters, merge until 1 cluster
- Bottom-up most common

# Hierarchical clustering: dendrogram




- *Leaf* at bottom is single observation
- Most similar observations merged to form cluster
- Most similar clusters merged to form new clusters
- Vertical axis is dissimilarity of merged clusters

# Hierarchical clustering: dendrogram clusters

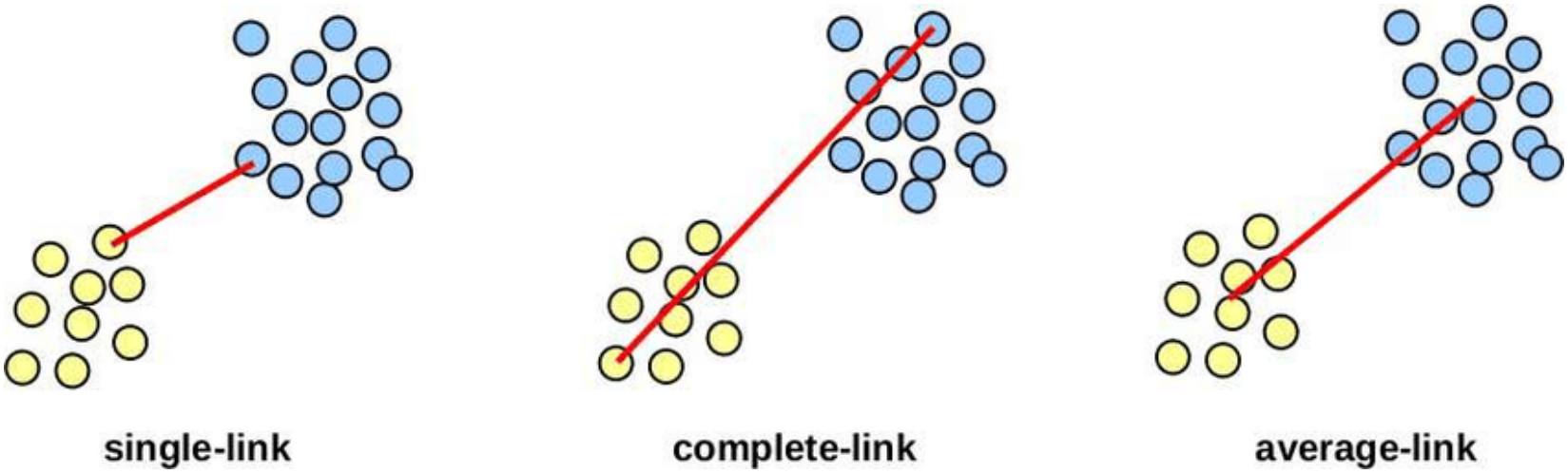



- *Cut* the dendrogram to make clusters
- Where you cut determines $k$
- Can apply elbow rule to dendrogram

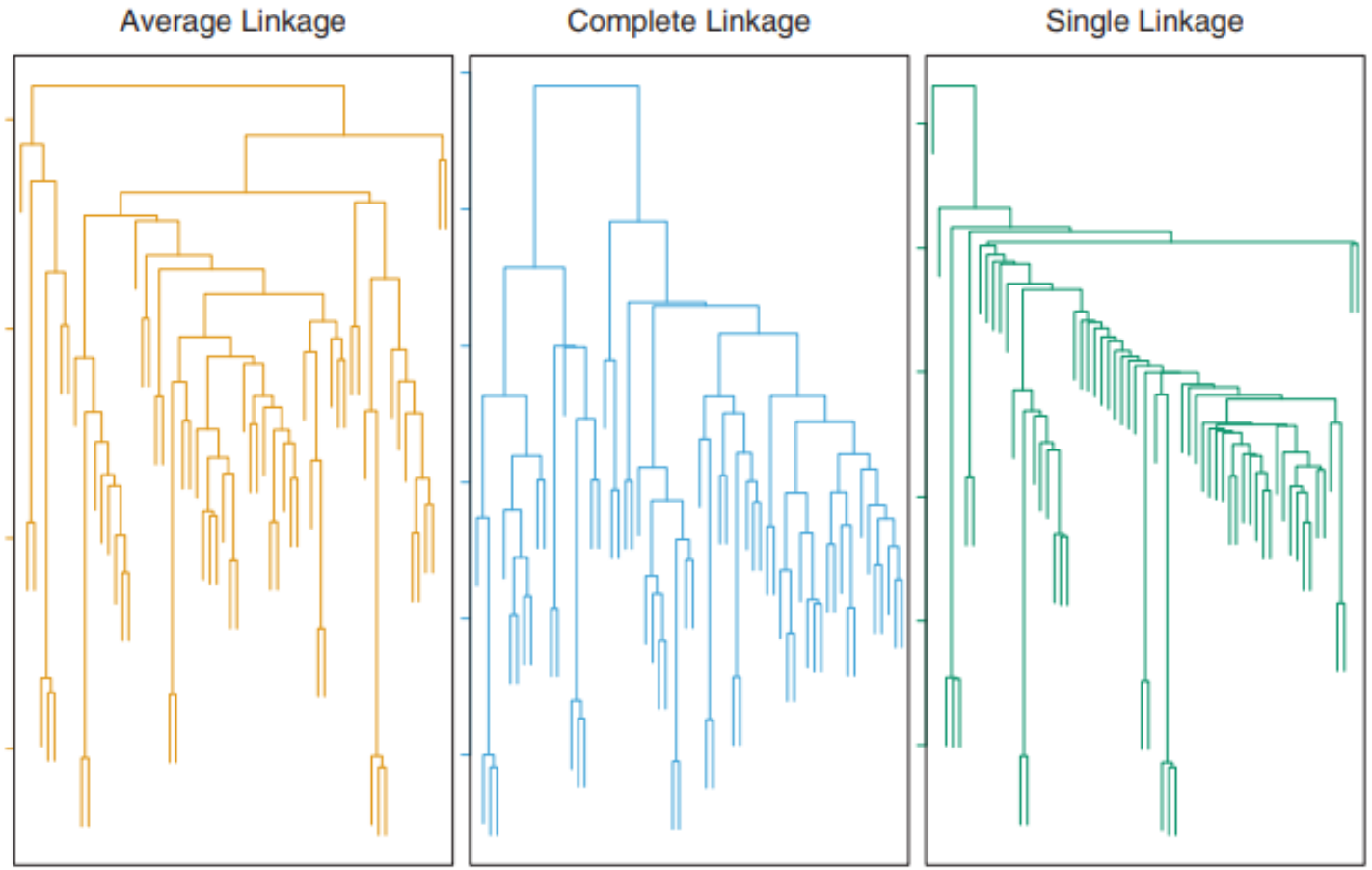
# Hierarchical clustering: dissimilarity and linkage

- Must define dissimilarity between observations
- Euclidean distance is common
- Must define dissimilarity or *linkage* between clusters:
- *Complete* - Maximum intercluster dissimilarity
- *Single* - Minimal intercluster dissimilarity
- *Average* - Mean intercluster dissimilarity
- *Centroid* - Dissimilarity between centroids
- Different dissimilarity and linkage results in different dendrograms
- Can try a variety and see if patterns consistently emerge

# Hierarchical clustering: linkage



# Hierarchical clustering: linkage



# Hierarchical clustering: bottom-up algorithm

**Algorithm 10.2** *Hierarchical Clustering*

1. Begin with $n$ observations and a measure (such as Euclidean distance) of all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities. Treat each observation as its own cluster.

2. For $i = n, n-1, \ldots, 2$:

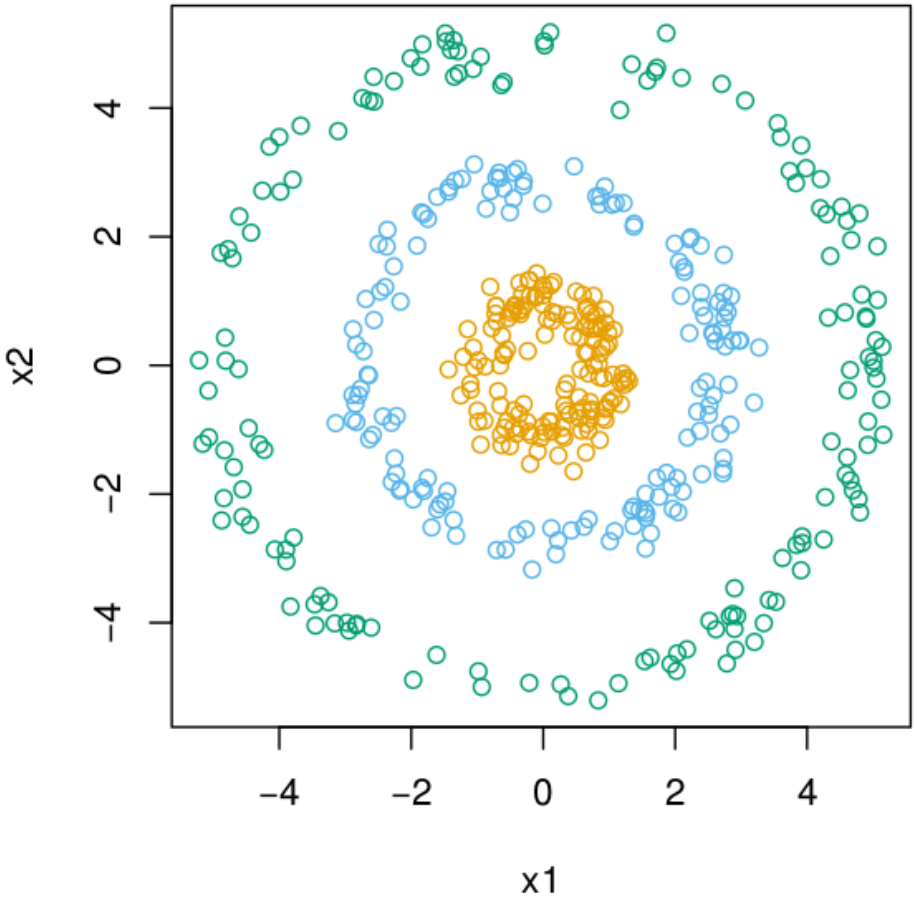
   (a) Examine all pairwise inter-cluster dissimilarities among the $i$ clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

# Spectral clustering

- unsupervised learning
- useful for unusual clusters

# Toy example



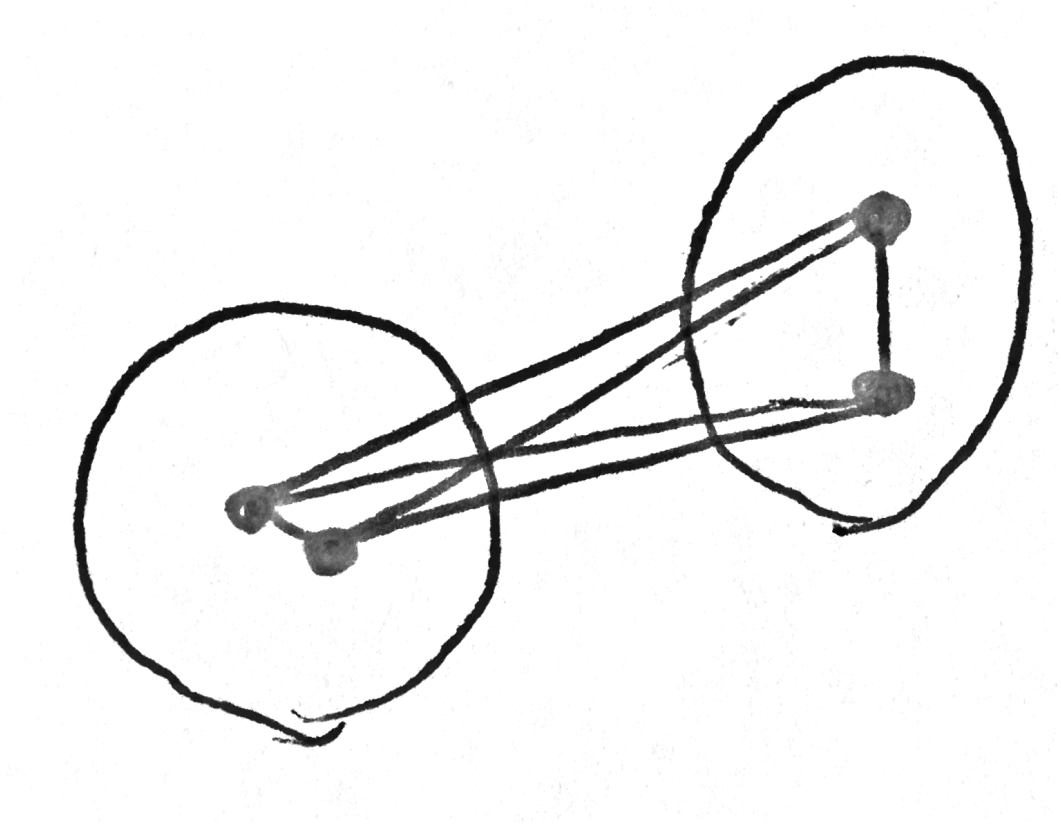
# Spectral clustering

- k-means will fail
- normal mixtures might work
- hierarchical methods might work (linkage?)
- useful for unusual clusters

# Spectral clustering

- main idea is to transform data, then cluster
- start with $N \times N$ matrix of pairwise similarities $s_{ii'} \geq 0$
- organize observations into graph with $N$ nodes connected by edges with length $1/s_{ii'}$
- clustering is then a graph-partition problem; partition graph such that edges that exit clusters have long lengths (small similarity), and edges within clusters have short lengths (large similarity)

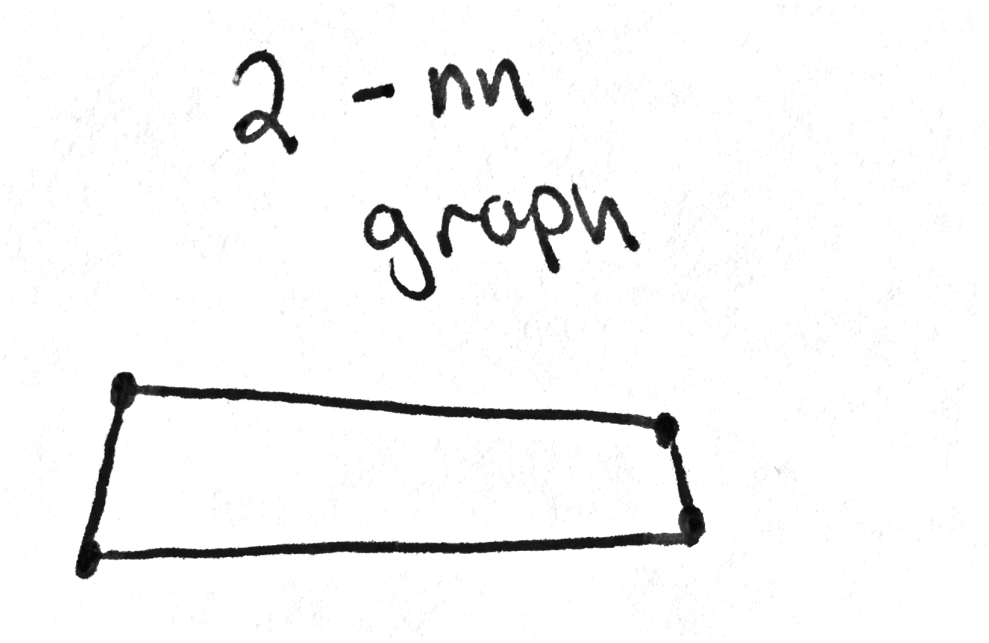
# Graph partition problem



# Spectral clustering

- consider $x_i \in \mathbb{R}^p$
- $s_{ii'} = \exp(-||x_i - x_{i'}||^2/c)$ (radial kernel)
- many ways to encode graph based on $s_{ii'}$
- e.g., fully connected
- e.g., K-nearest-neighbor graph
- define $\mathcal{N}_K$ to be the set of nearby pairs of points
- obs $(i, i')$ is in $\mathcal{N}_K$ if $i$ is K-nn of $i'$ or vice versa
- only $(i, i')$ in $\mathcal{N}_K$ are connected with weight $w_{ii'} = s_{ii'}$, otherwise no edge ($w_{ii'} = 0$).
- the K-nn graph is a kind of pre-processing; hard thresholding of similarities)

# 2-NN Graph



# Spectral clustering

- *adjacency matrix* of weights (similarities; $N \times N$): $W = \{w_{ii'}\}$
- *degree* of node $i$: $g_i = \sum_{i'} w_{ii'}$
- let $G = \mathrm{diag}(g_1, \ldots, g_N)$
- define *graph Laplacian*:

$$L = G - W$$

- spectral clustering:
  1. find $m$ eigenvectors $Z$ ($N \times m$) corresponding to the $m$ **smallest** eigenvalues of $L$ (ignoring trivial constant eigenvector)
  2. use standard clustering method to cluster rows of $Z$, which serve as the transformed data

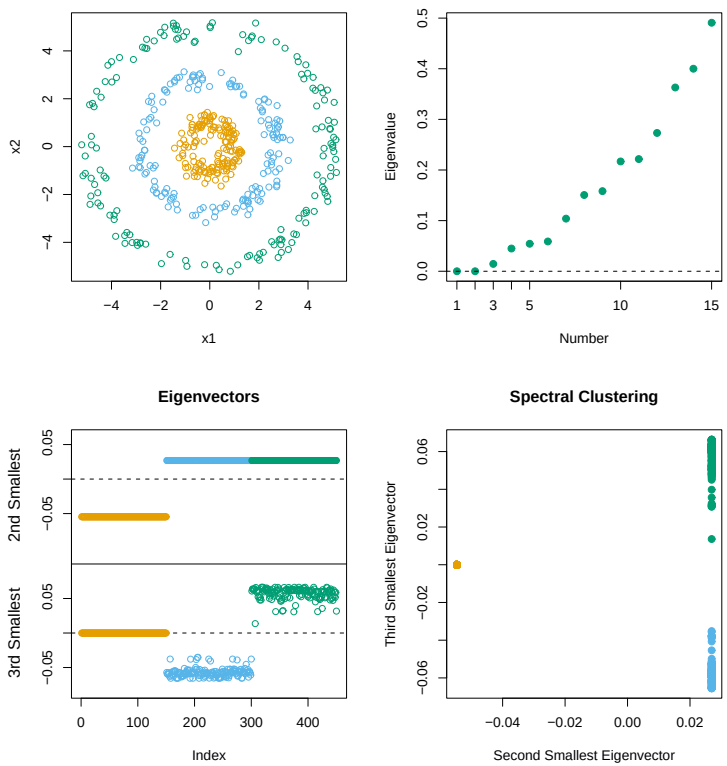



**FIGURE 14.29.** *Toy example illustrating spectral clustering. Data in top left are* 450 *points falling in three concentric clusters of* 150 *points each. The points are uniformly distributed in angle, with radius* $1, 2.8$ *and* 5 *in the three groups, and Gaussian noise with standard deviation* 0.25 *added to each point. Using a* $k = 10$ *nearest-neighbor similarity graph, the eigenvector corresponding to the second and third smallest*

# Spectral clustering: Why it works.

- consider tranforming $(N \times p)$ data $X$ into $(N \times 1)$ vector $f$ for the purpose of clustering
- for items that are similar (large $s_{ii'}$ and $w_{ii'}$), $f_i$ should have a similar value to $f_{i'}$ and vice versa
- if we consider the following quantity

$$
\begin{aligned}
f^T L f &= \sum_i g_i f_i^2 - \sum_i \sum_{i'} w_{ii'} f_i f_{i'} \\
&= \sum_i \sum_{i'} w_{ii'} f_i^2 - \sum_i \sum_{i'} w_{ii'} f_i f_{i'} \\
&= \frac{1}{2} \sum_i \sum_{i'} w_{ii'} (f_i - f_{i'})^2
\end{aligned}
$$

- finding $f_i$ similar to $f_{i'}$ for large $w_{ii'}$ implies small $f^T L f$
- now if restrict $f$ to be an eigenvector of $L$, then $f^T L f$ is the eigenvalue, and we should focus on those vectors with smallest values
- constant vector $(1^T)$ is eigenvector of $L$ with eigenvalue zero