JOHNS HOPKINS
SCHOOL of MEDICINE

Quantitative Molecular Neurogenomics 2022
Module 10 - Clustering

ME.440.825
Loyal Goff - Course Director
Alina Spiegel - TA
Richard Sriworarat - TA

JOHNS HOPKINS
SCHOOL of MEDICINE

WHY DO WE CLUSTER?

Why Cluster?

- Reveal hidden patterns/groupings/associations in gene expression data
- Understand the hierarchical relationships across data points
- Identify technical sources of variation/error
- Confirm biological relationships prior to discovery
- Identify patterns/networks of gene co-regulation/expression
- Estimate dimensionality of datasets

A note about clustering

- More of an art form
  - Parameters must be informed by biology of the system
- No clustering method/solution should be interpreted as sacrosanct
- Clustering results depend on ***context*** (ie. Selected space and dimensionality)
  - Different clustering results can & should be returned based on choice of genes/pathways/samples used.
- Biology is hierarchical

HIERARCHICAL CLUSTERING

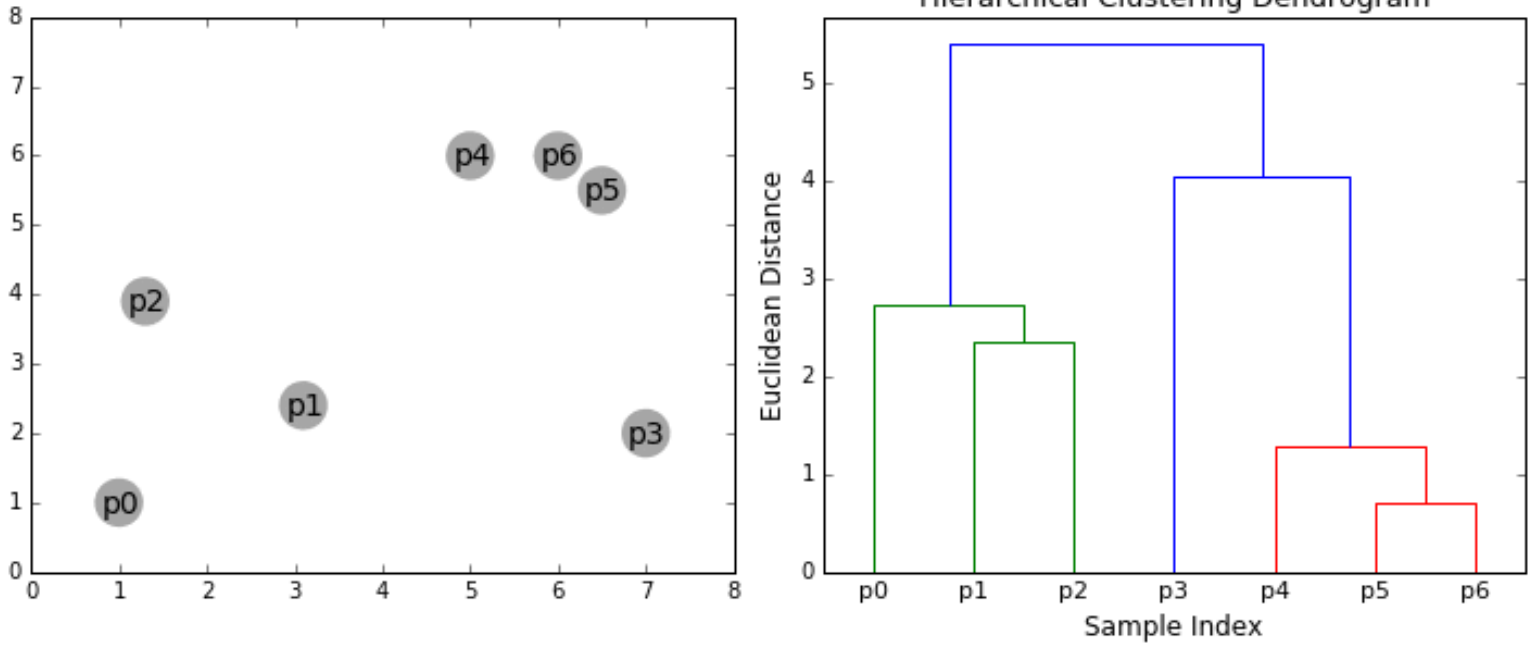
Hierarchical Clustering

- HC is a recursive partitioning of a set of points at increasingly finer granularity
- Input: Set of points with a score that represents the pairwise similarity (or dissimilarity) of the points
  - Usually a ‘distance’ matrix
- Goal: Output a tree (dendrogram) with leaves representing individual data points and internal nodes (branches) representing groups (clusters) of data points

HC Strategies

- Agglomerative:
  - “Bottom up”
  - Each data point starts as it’s own cluster and pairs are merged into new clusters at each iteration
- Divisive:
  - “Top down”
  - All points are treated as belonging to a single cluster and with each iteration, the preceding clusters are split until only individual data points remain
  - (Less often used in bioinformatics)

Hierarchical clustering in action

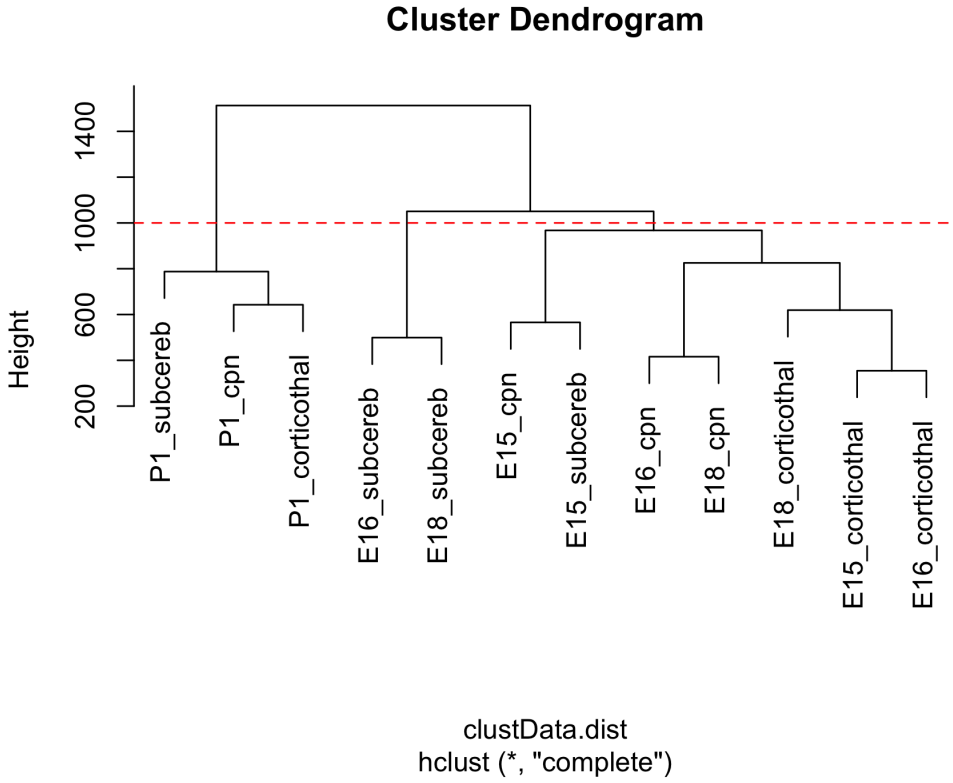


HC in R

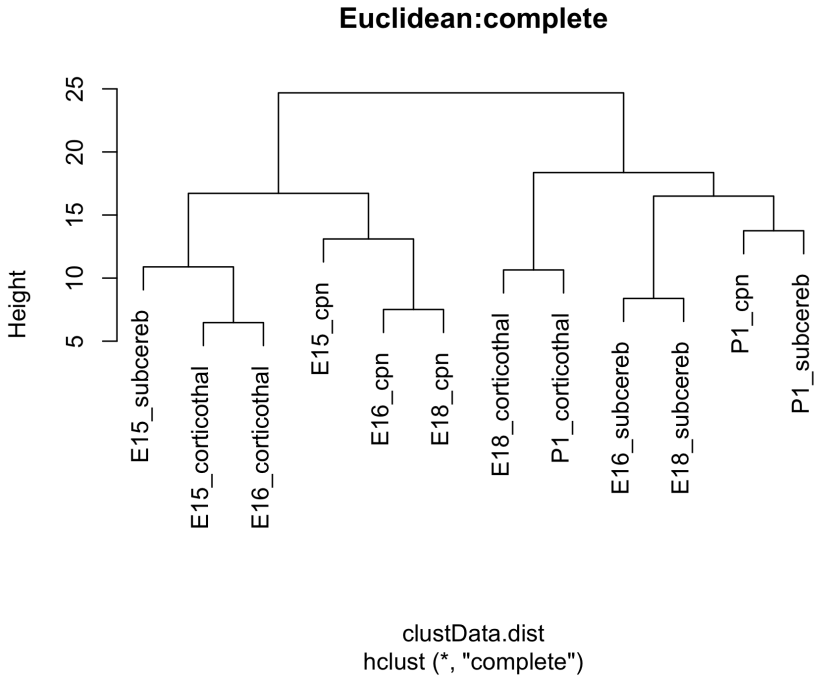
- Step 1:
  - Organize your input data so that the items you want to cluster are rows (observations) and the dimensions along which you want to cluster are columns (variables)
- Step 2:
  - Calculate the pairwise distance between all rows with dist()
- Step 3:
  - Perform hierarchical clustering with hclust()
- Step 4 (optional):
  - Determine where the dendrogram should be partitioned to return the number of clusters & cluster assignments

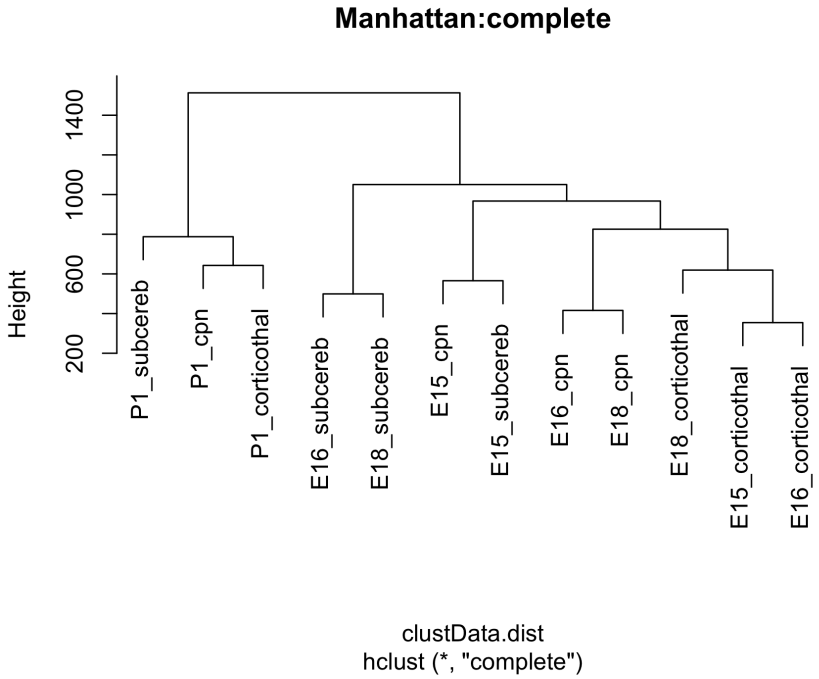
Using HC to partition leaves

- cutree() is used to partition a dendrogram into discrete clusters
- Partition by choosing k
- Partition by choosing threshold value on distance metric



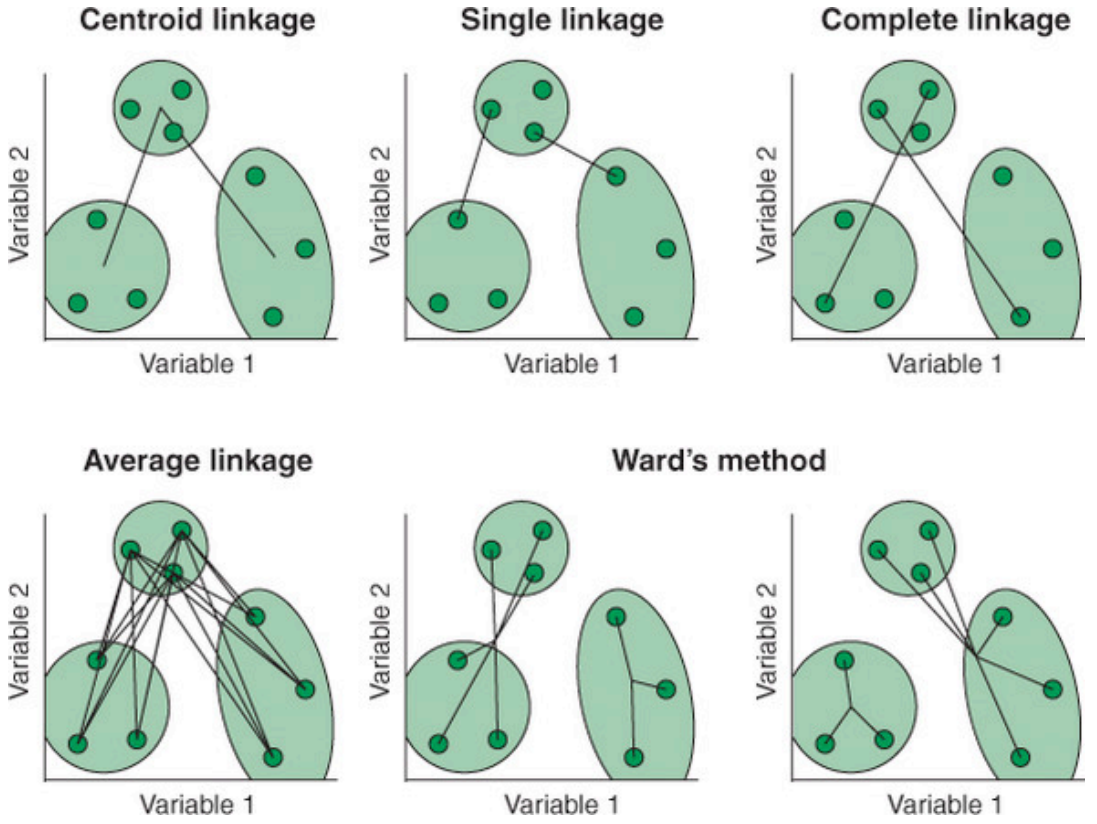
HC is sensitive to choice of distance metric





HC is sensitive to the choice of linkage

- How are nodes/ clusters joined together?
- Choice of linkage can dramatically affect dendrogram
- Default is usually complete linkage



machine-learning-with-r-the-tidyverse-and-mlr

Pros & Cons of HC

- Pros
  - Good for small n of leaves
  - Shows all relationships across conditions at different levels
    - Hierarchical organization
  - Can be quickly leveraged for visualizing relationships amongst samples in a given gene space
- Cons
  - Pairwise distance calculation is a time-consuming operation with large time complexity $O(n^3)$
  - Interpretations of many data points can get crowded/unwieldy
  - Sensitive to choice of distance metric & dimensionality of data

PARTITIONING METHODS

K-means clustering

- Objective: *Partition n observations into k clusters with each observation belonging to the cluster with the nearest mean. With each iteration, goal is to minimize within-cluster distance to mean*

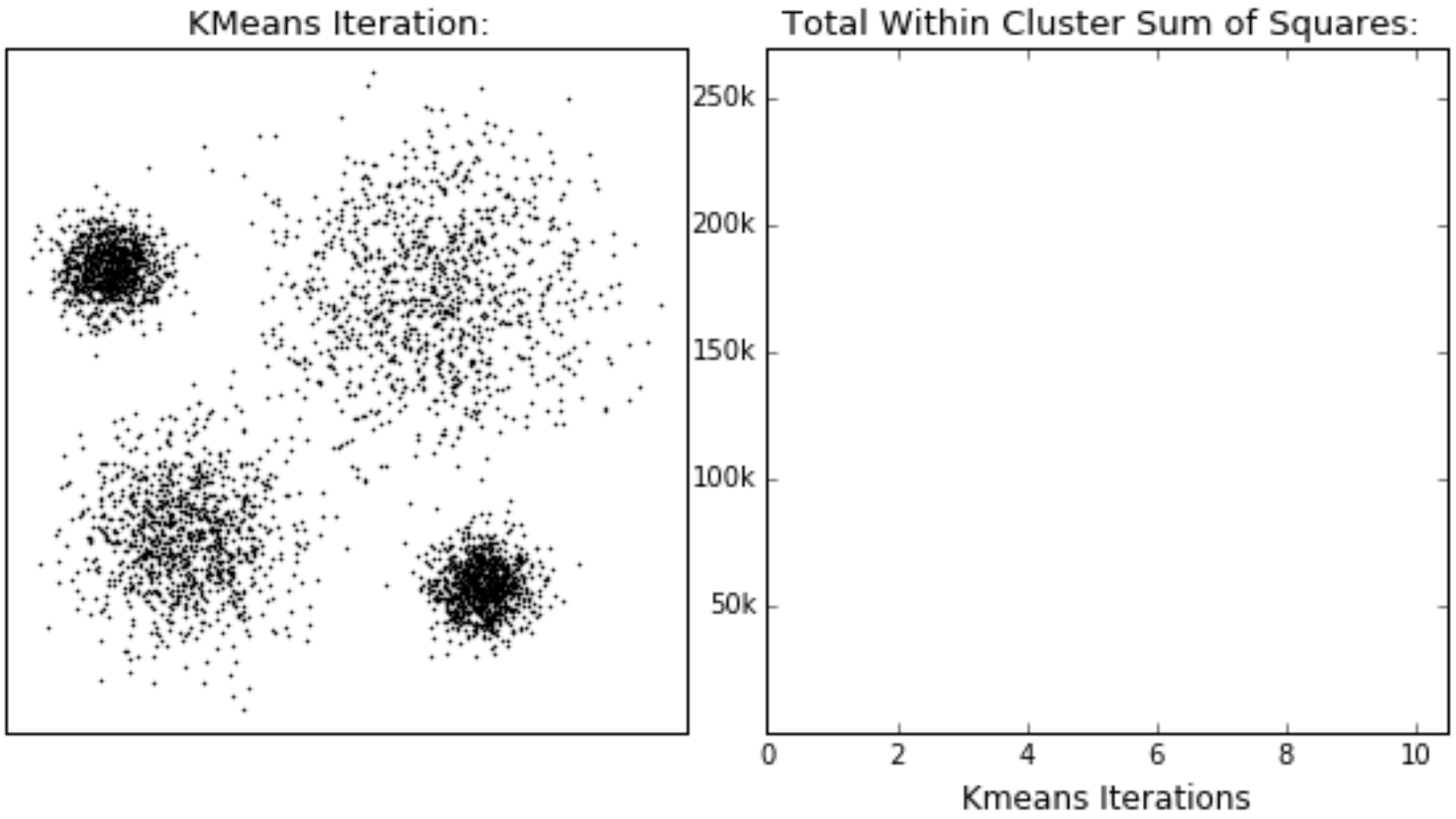
- Input: data.frame of observations (rows) to be partitioned based on their distance to a cluster mean in variable (column) space

- Output: a partitioning of observations into k clusters

K-means is an iterative algorithm

- Step 1:
  - Randomly seed k points
- Step 2:
  - Calculate distance from each observation to each k point
- Step 3:
  - (Re)assign points to closest k seed point
- Step 4:
  - Move k to new point based on summary value of newly-assigned observations
    - e.g. kmeans moves k to be the mean of all points assigned to a particular k
- Repeat steps 2-4
  - Fixed number of iterations OR
  - Some threshold level of convergence reached
- After stopping, report final assigned k as cluster assignment

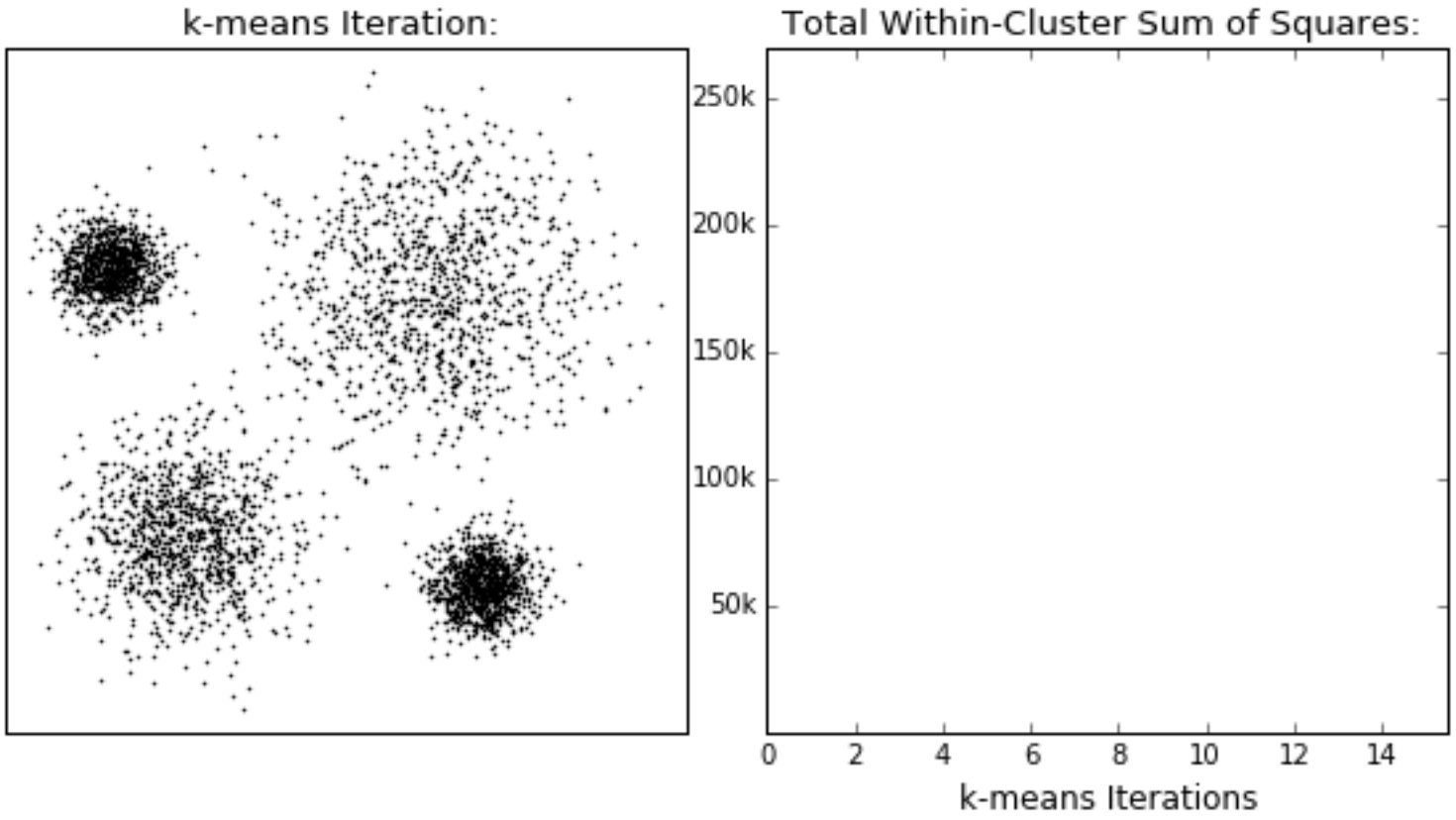
K-means clustering



Kmeans in R

- Step 1: prepare and organize data so that the dimension you would like to cluster are rows in your input data
- Step 2: choose a value for k to fix
- Step 3: perform kmeans clustering with kmeans(<input data>,k)

Kmeans clustering is sensitive to start positions



Kmeans is sensitive to the choice of k

- Kmeans is (almost) guaranteed to converge on a solution
- Since k is fixed, samples ***will*** be partitioned into k groups
- Overfitting with a larger k is very easy
- Underfitting can aggregate and mask true differences between sets of values

Kmeans pros and cons

- Pros
  - Simple to implement
  - Scales to larger datasets than HC
  - Less time complexity than HC

- Cons
  - Manual choice of k is a drawback
    - Can be ‘optimized’ with specific metrics
  - Starting values can impact quality of partitioning
  - Naive kmeans has problems with clusters of different size/density
    - Can be generalized to solve
  - Centroids can be affected by outliers
  - NP-hard problem
    - Newer methods for parallelization of kmeans help to resolve this
  - Nonlinear relationships are not captured well

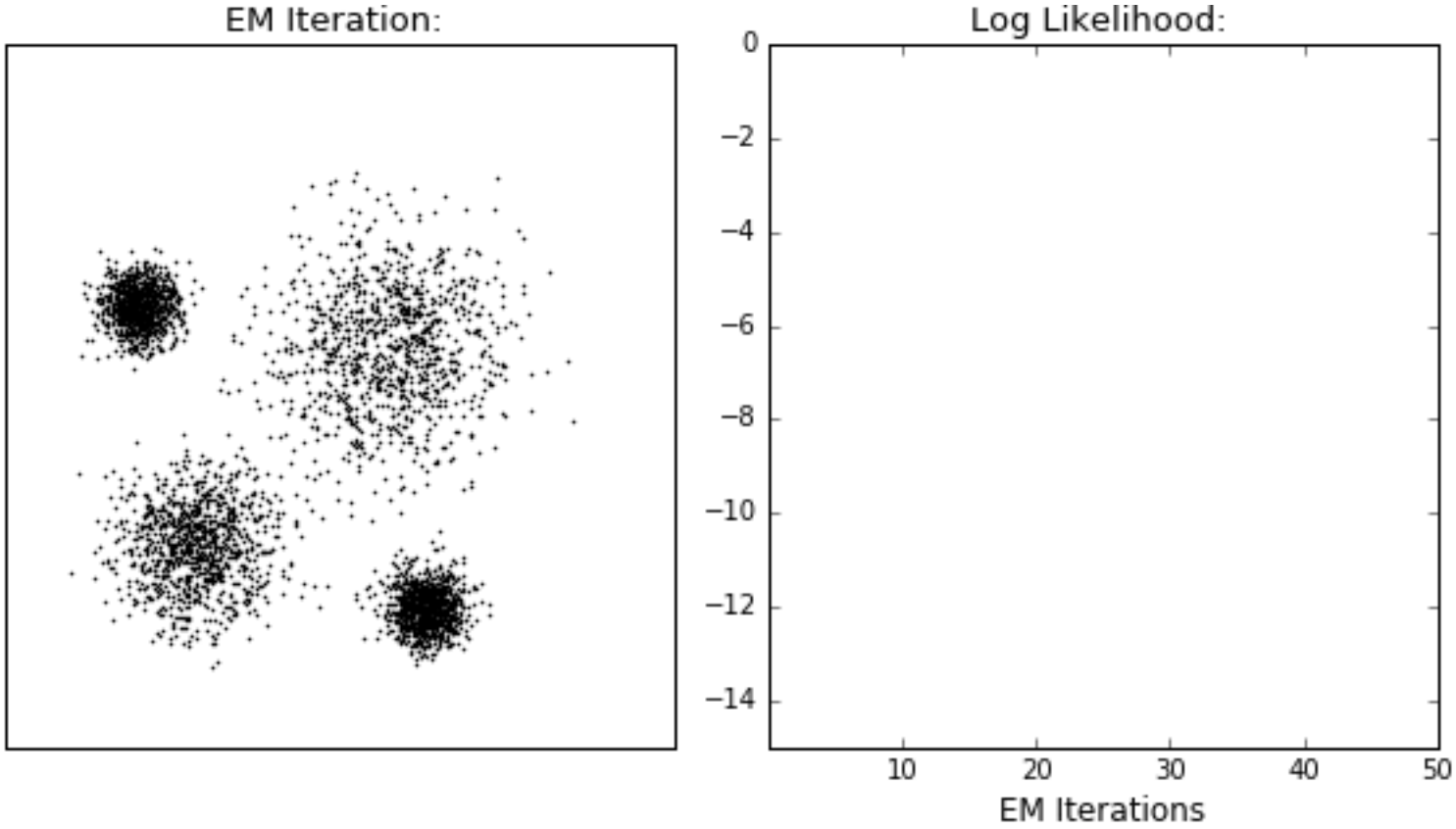
JOHNS HOPKINS
SCHOOL of MEDICINE

MODELING METHODS

General Expectation Maximization (EM)

- Choose model to describe data
- Choose initial starting parameters
- Iterate over:
  - Estimate latent variables of the model
    - (e.g. mean and cov of multinomial normal (Gaussian MM)
  - Evaluate log-likelihood and adjust model parameters to attempt to maximize with each iteration.

Gaussian Mixture Modeling



The Curse of Dimensionality

- When the dimensionality increases, the volume of the space increases so fast that the available data become sparse.
- Objects appear to be sparse and dissimilar in many ways, which prevents common data organization strategies from being efficient.
- TL;DR
  - Distance metrics become less meaningful in higher dimensions, as all points are farther away from each other.