CSE 527: Computational Biology
Fall 2005, Professor Larry Ruzzo
Notes by Tom Greene

Clustering 101

I. What is it?
- Group similar objects together
- Data Exploratory tool: not statistically rigorous, looks for similar structures

II. Why cluster?
- Tissue classification—group tissues that behave similarly over time
- Finding biologically relevant genes
  --For example: coordinated expression of genes in a pathway would have similar expression data
- Hypothesis generation: generates questions like, "Are these genes being expressed together because they have similar function, similar regulation, or both?"

III. Algorithms to Cluster Expression Data
- Partitional: Top down; Start with all data and then separate into smaller clusters
  -Ex: k-means
- Heiarchical: Down up; Start with each data point as a cluster and then begin joining together
  -Ex: single link, average link, complete link
- Why so many methods?
  -NP-hard problem: an exact solution is almost impossible because the problem is so complex (high dimensionality and high noise in array data).
  -There is no best method. Method depends on data and goal.

IV. How Define Similarity?
- Compute some type of quantitative number
  -Correlation coefficient: Measures direction of change or parallel patterns in the data.
  -Euclidean distance: Magnitude and direction.

V. Clustering Algorithms
- Inputs for algorithms:
  -Raw data matrix or similarity matrix
  -Number of clusters or some other endpoint parameter
- Different Classification of Clustering Algorithms
  -Hierarchical vs. Partitional
  -Heuristic based vs. Model based

-Soft (some flexibility, partial membership to cluster) vs. Hard (rigid assignment of each point to one cluster)

VI. Hierarchical Clustering

An agglomerative, bottom-up approach

*Algorithm

Initialize: Each item is a cluster
Iterate: Merge the two most similar items
Halt: When required # of clusters is reached

- Types/Variations

a. Single Link: "Any friend of yours is a friend of mine"

-Cluster similarity is the similarity of the two most similar members. May result in long and skinny clusters. Computationally fast. Does not work well with gene expression data. Clusters defined by the algorithm.

b. Complete Link:

-Cluster similarity is defined by the similarity of the two least similar members. May result in tight, round clusters. Computationally slow.

c. Average Link:

-Cluster similarity is the average similarity of all pairs. Tight clusters. Computationally slow.

d. Centroid Link (not to be confused because sometimes called average link)

-Cluster centroid is the average of all points in the cluster. Cluster similarity is the distance between the centroids. This method discards shape and orientation of a cluster. Computationally faster due to fewer distances to compute.

-Free software to perform this clustering: Treeview
http://rana.lbl.gov/EisenSoftware.htm

e. Patitional, K-Means

Initialize: Pick k number of points at random and call them the centroids of clusters.
Iterate: Assign each data point to the "closest" or "most similar" centroid.
Find new center of points and re-define centroid.
Halt: Keep going until center doesn't change; convergence

-Tends to result in spherical, equal sized clusters. Computationally fast. Converges to a local minimum, not necessarily a global minimum. Related to a model-based approach (see later lectures).

VII. Running Time of Algorithms

- Single Link:

  Build Similarity Matrix: (n genes)(p raw data elements)
  Find minimum: n^2 computations
  Merge and Update: n computations

  *We can think of these operations as calculating the minimum of each column.

  *Order = n^3 (approximately, we do not worry about constant terms because we just want the big pictures)

However, if we re-think the algorithm we see that really we can carry the columns from one clustering step over to the re-defined matrix at the next clustering step. Then we only have to re-check one column for the minimum value. Thus, the number of computations to find the minimum lowers from n^2 to n.

- Complete Link

  Build Similarity Matrix
  Find maximum

Now, when we find the maximum, the minimum value can disappear. Thus, we would have to re-calculate the column minimum. If you use a heap data structure you can lower the number of calculations from n^2 to nlog(n). This log(n) factor is what makes this algorithm computationally slower than the single link.