GabrielReder_BMI203_HW2

February 7, 2017

Gabe Reder BMI 203 - HW 2
2/7/17

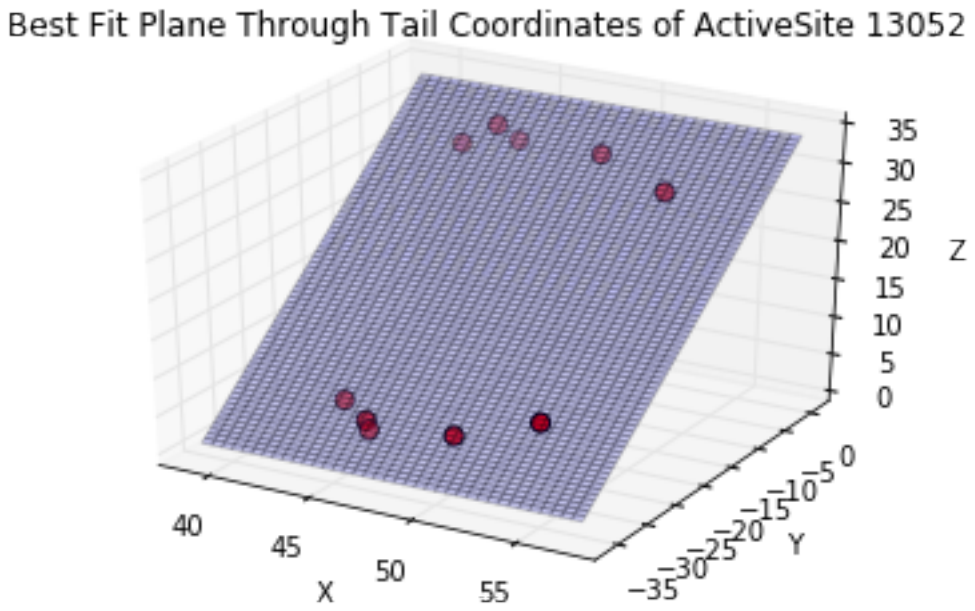
```
In [10]: import gabe_hw2.cluster as cluster
         from gabe_hw2.io import read_active_sites, write_clustering, write_mult_cl
         import numpy as np
         import scipy.linalg
         from mpl_toolkits.mplot3d import Axes3D
         import matplotlib.pyplot as plt
         import warnings
```

0.1 Similarity Metric

The similarity score between two sites is calculated as follows. For each site, the list of 'tail' atoms are collected from the site's residues. For each residue in the site, the tail atom is defined as the last item in that residue's list of atoms. The idea is that these will be atoms at the end of the residue (furthest from the backbone). For each active site, the three-dimensional coordinates of the tail atoms are taken and a best-fit linear plane through them is calculated. An example of this is shown below for input active site 13052 from the data set.

```
In [2]: warnings.filterwarnings("ignore")

        # From ActiveSite 13052
        coords = [(44.823, -2.971, 29.019), (42.833, -5.824, 28.879),
                  (52.569, -5.324, 26.958), (48.911, -3.174, 29.192),
                  (44.226, -4.544, 31.342), (45.826, -32.172, 6.023),
                  (43.807, -29.015, 6.069), (53.45, -29.469, 8.665),
                  (49.953, -31.688, 6.064), (45.4, -30.33, 3.61)]

        coords = np.asarray(coords)
        x_max = np.amax(coords[ : , 0])
        x_min = np.amin(coords[ : , 0])
        y_max = np.amax(coords[ : , 1])
        y_min = np.amin(coords[ : , 1])
        z_max = np.amax(coords[ : , 2])
        z_min = np.amin(coords[ : , 2])

        X,Y = np.meshgrid(np.arange(x_min - 4.0, x_max + 4.0, 0.5),
```

```
                    np.arange(y_min - 4.0, y_max + 4.0, 0.5))
XX = X.flatten()
YY = Y.flatten()

# best-fit linear plane
A = np.c_[coords[:,0], coords[:,1], np.ones(coords.shape[0])]
C,sum_residual,_,_ = scipy.linalg.lstsq(A, coords[:,2])    # coefficients

avg_dist = sum_residual / len(coords)
# evaluate it on grid
Z = C[0]*X + C[1]*Y + C[2]


fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot_surface(X, Y, Z, rstride=1, cstride=1, alpha=0.2)
ax.scatter(coords[:,0],  coords[:,1], coords[:,2], c='r', s=50)
plt.xlabel('X')
plt.ylabel('Y')
ax.set_zlabel('Z')
ax.axis('equal')
ax.axis('tight')
plt.title('Best Fit Plane Through Tail Coordinates of ActiveSite 13052')
plt.show()
```



For both active sites, the average distance from the best fit plane is calculated as the sum of the squared residuals between the tail atoms and the best fit plane through them. This gives us a single number for each active site. We then take these two average distances and find the absolute

value of the difference between them. Then this difference is converted to logorithmic scale since the range can vary significantly (differences that are less than 1.0 are simply called 0.0). Finally, all of these logorithmic distances are normalized by the maximum difference found between all active sites in the data set and the resulting normalized value is subtracted from 1. This gives us a similarity score in the range [0, 1] for all active sites in the data set.

Biologically, this is an attempt to measure the similarity in terms of the geometry of the two active sites. My idea was that the tail coordinates of the residue roughly outline the interaction surface with the ligand that will dock at the active site. By looking at the best fit plan and the distance to that plane, I could get an idea of how planaar the orientation of these tail atoms are for a given active site. Two active sites whos tail atoms are both very planaar might be presenting a similar sort of docking surface to the ligand.

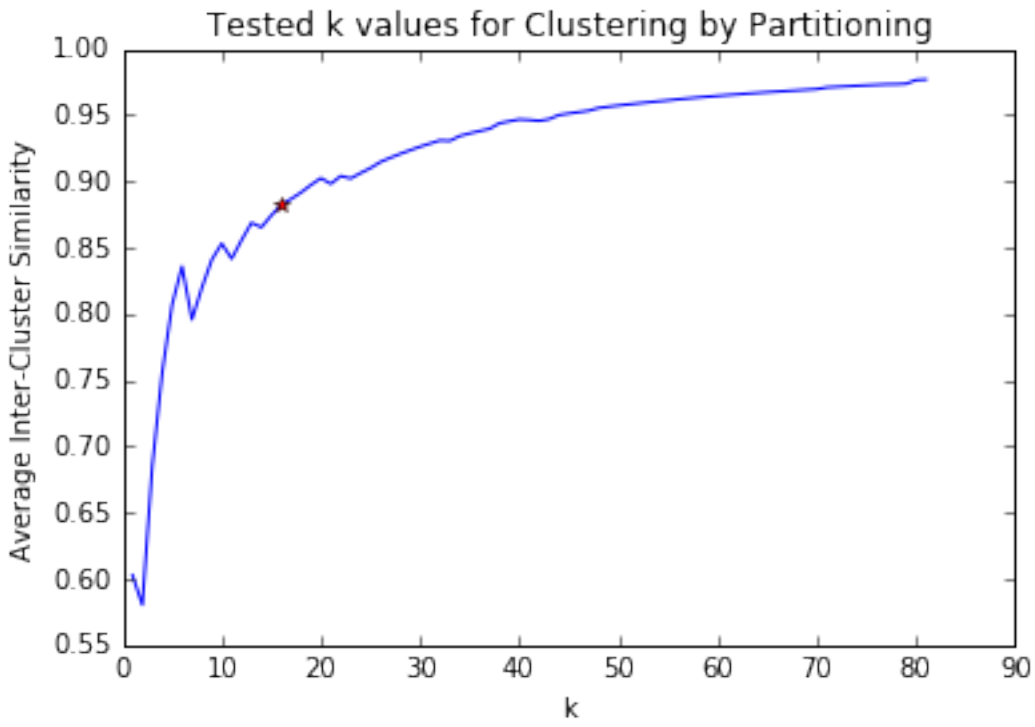
0.2 Partitioning Algorithm

For the partioning algorithm, I implemented k-means clusterings. First, to determine a good choice for k, I run k-means over a range of possible k values (from 1 up to 3/5 of the total number of active sites). For each resulting clustering, I have a resulting average intra-cluster similarity (defined as the average similarity within clusters averaged across all clusters for a given clustering). This intra-cluster similarity should converge to 1 as the k values approach the number of active sites since this approaches the clustering in which every active site is assigned to its own cluster (resulting in an average intra-cluster similarity of 1.0). Wanting a high average intra-cluster similarity but also wanting to avoid k singleton clusters, I set the k value to be the lowest value at which the average intra-cluster similarity is 90% of the max value found over all the range of k values tested. The k-values tested and the chosen optimal k values are shown below for the input data set of active sites.

After determing a k value to use, I run k-means clustering with the given k value. At each iteration, for cluster centers I use the single active site in the cluster that is most similar to all other members of the same cluster. I do this instead of finding the cluster centres as averages of all the cluster members because for these active sites, I thought that defining an average between multiple sites would undermine biological relevance.

```
In [3]: active_sites = read_active_sites('data')
        sim_matrix = cluster.get_sim_matrix(active_sites)
        cluster.get_optimal_k(active_sites, sim_matrix, plot = True)
```

```
Read in 136 active sites
```



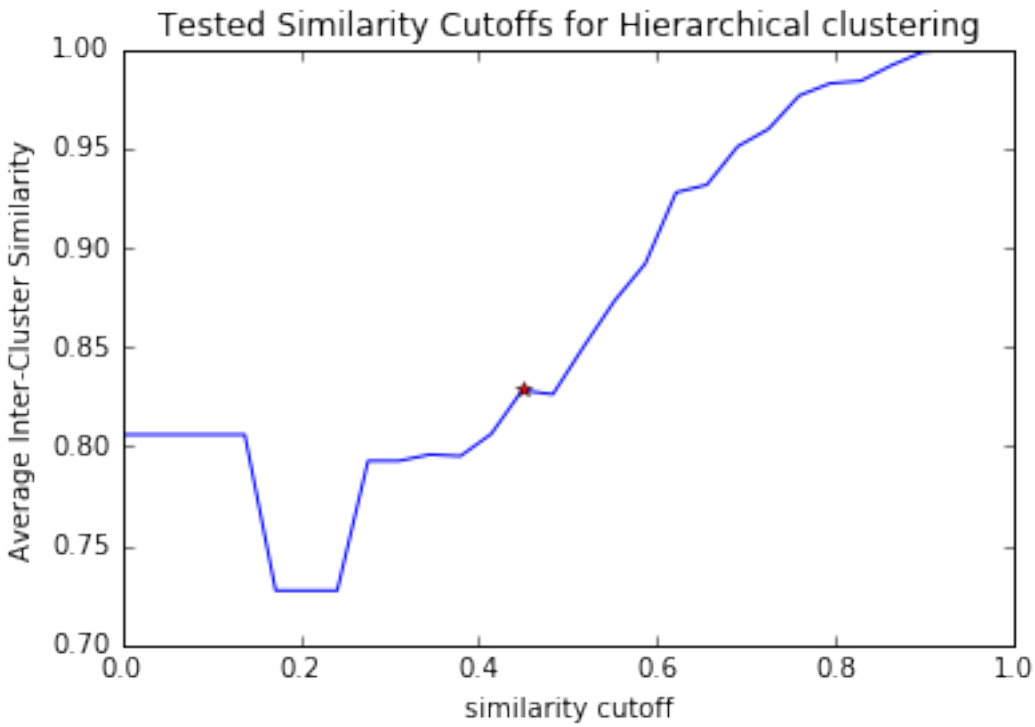
Out[3]: 16

0.3 Hierarchical Algorithm

For hierarchical clustering, I implemented an agglomerative approach. The algorithm begins by assigning each active site to its own cluster. Then at each iteration, it joins the two clusters between which the similarity score amongst the two lowest scoring members are the highest. That is, the joined clusters have the best furthest neighbor distance between them. I keep doing this until the furthest neighbor similarity between the two best clusters is lower than a given similarity cutoff threshold.

To determine the similarity cutoff threshold to use, I first run hiearchical clustering on a range of input similiarty cutoff thresholds (from 0.0 to 1.0) and, similar to the partioning algorithm, plot the resulting average intra-cluster similarities. Then to determine a goood similarity cutoff to use, I look for local maxima in average intra-cluster similarities (similarity cutoff values that produce higher intra-cluster similarities than both the next higher and lower similarity cutoff values). Of these local maxima, I save the one for which the average distance between the resulting intra-cluster similarities and the two neighboring intra-cluster similarities is the highest. The idea here was to pick a similarity cutoff value which was providing the highest marginal benefit in terms of intra-cluster similarity compared to its neighbors. The tested similarity cutoff values and results, together with the chosen similarity cutoff value, are shown below.

```
In [4]: # active_sites = read_active_sites('data')
        # sim_matrix = cluster.get_sim_matrix(active_sites)
        cluster.get_sim_cutoff(active_sites, sim_matrix, plot = True)
```




```
Out[4]: 0.44827586206896552
```

0.4 Quality Metric / Comparing Clusterings

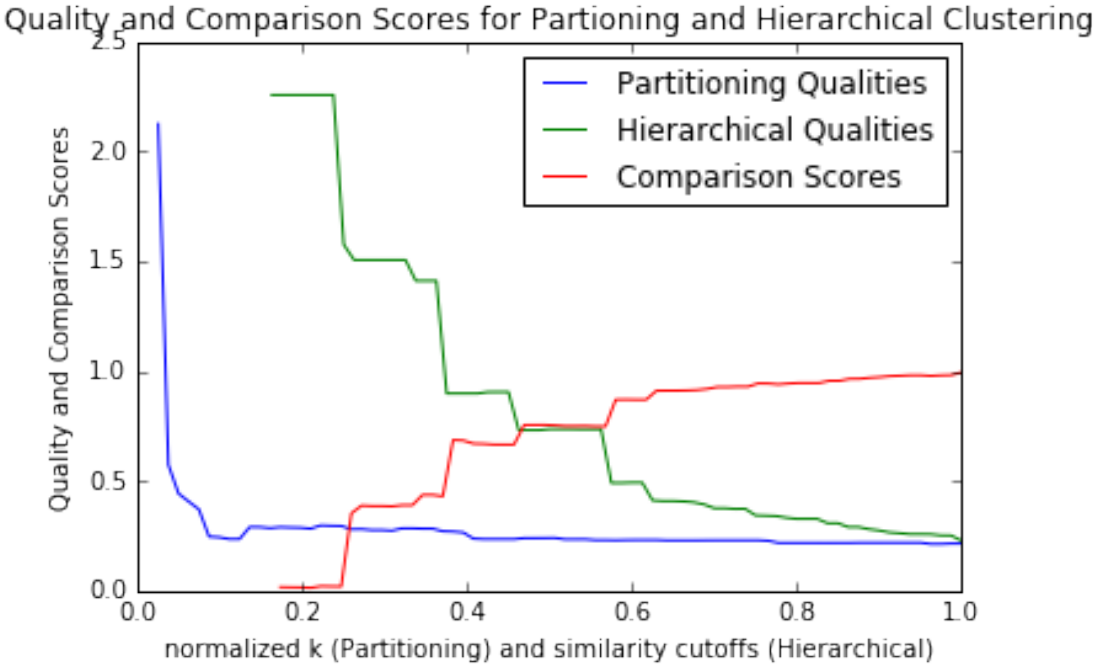
For a quality score of a clustering, I used the average silhouette score across the clustering. For a given ActiveSite, the silhouette score, s, in my case is defined as:

```
s = (a - b) / (min(a, b))
```

where a is the average similarity of the ActiveSite with all other members of its assinged cluster and b is the average similarity between the ActiveSite and all others members of the closest cluster (defined as having the highest average similarity). The idea is to get a sense for how appropriately the ActiveSite is situated. The silhouette score should be higher when the ActiveSite is in a cluster with other very similar sites and there are no similar sites in other clusters. The average silhouette score across a clustering gives a sense of how well all the ActiveSites fit in to their respective clusters across the board.

To compare two clusterings, I take the average silhouette scores of the two clusterings and find the absolute value of the difference between them. I then normalize this value to the range [0,1] where 0 is the worst possible comparison score between any two clusterings and 1 is the best possible comparison score between any two clusterings. A plot of partioning and hierarchical quality scores across a range of normalized k values and similarity cutoffs are shown below. Also shown are the comparison scores where each comparison score is taken between the hiearchical and partioning clustering found at the respective normalized k value (for partitioning) or similarity cutoff (for hierarchical).

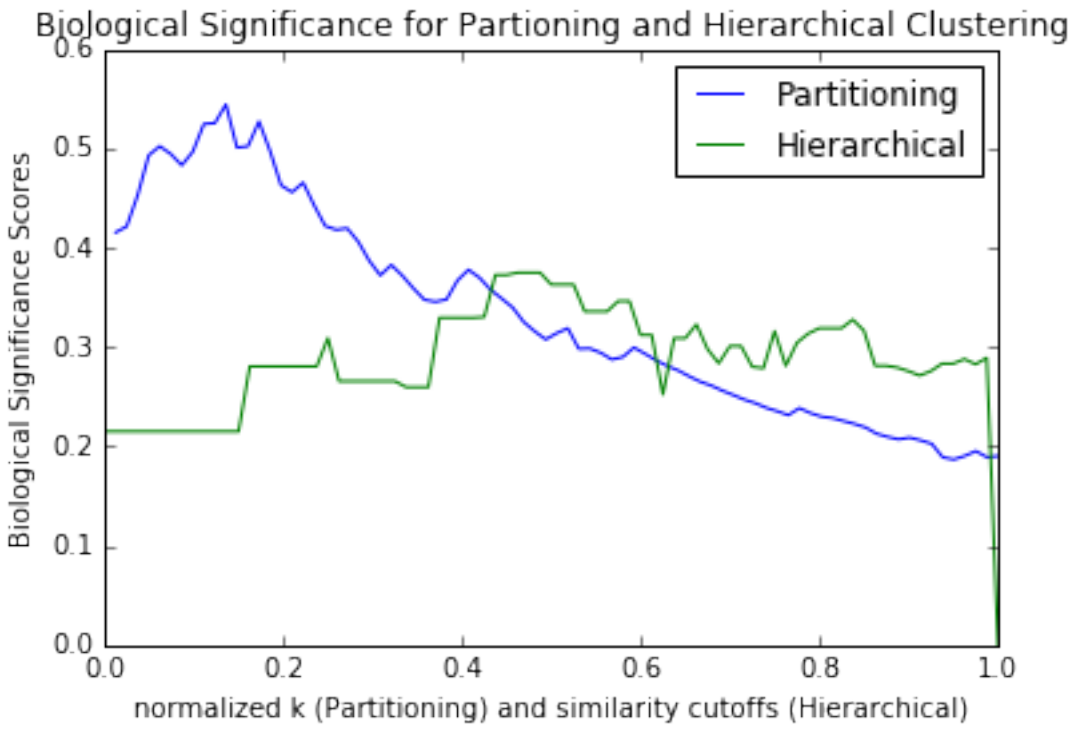
```
In [5]: hierarchical_clustering = cluster.cluster_hierarchically(active_sites, sim_
        partioning_clustering = cluster.cluster_by_partitioning(active_sites, sim_m
        cluster.compare(hierarchical_clustering, partioning_clustering, active_site
```



0.5 Biological Meaning

To get a sense of the biological significance of the clusterings, I have a biological score metric. For a single cluster, the metric is calculated as follows. For every pair of active sites in the cluster, compute a sequence similarity score between the two by dividing the number of residues the two sequences have in common (ignoring position in sequence) by the length of the longer sequence. This produces a number in the range [0, 1] for every pair of active sites. Average this score across the entire cluster and return as the biological score of the cluster. The biological significance score of the entire clustering is calculated as the average biological score of all the clusters in the clustering. The biological significance score of the clusterings produced by a range of k values and similarity cutoffs for partioning and hierarchical clustering are shown below.

```
In [6]: cluster.compare_biology_significance(hierarchical_clustering,
                                              partioning_clustering,
                                              active_sites, sim_matrix)
```


Biological Significance for Partioning and Hierarchical Clustering
Partitioning
Hierarchical
Biological Significance Scores
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.0
0.2
0.4
0.6
0.8
1.0
normalized k (Partitioning) and similarity cutoffs (Hierarchical)