Clustering Genes Trees into Topological Classes

To uncover evolutionary relationships between species, we can no longer assume that all genetic loci in a genomic dataset support the same underlying tree topology: effects such as incomplete lineage sorting, introgression and horizontal transfer can cause incongruence to occur between gene trees. Here, we explore ways to identify multiple topologies present in the data by clustering trees reconstructed from individual loci into classes with common underlying topologies.

Using simulation and empirical data, we investigated which is the best combination of distance metric (Robinson-Foulds, Weighted Robinson-Foulds and Felsenstein’s Branch Length Distance) and clustering method (single-linkage, complete-linkage, UPGMA and Ward’s method) to partition the genes into topological classes . On real data, where the true partition is unknown, we computed trees for each cluster from concatenations of the constituent loci. We then used the sum of log-likelihoods of the resulting cluster trees as a scoring function to assess the relative performance of each method.

Finally, we explored ways of estimating the optimal number of topologies, using information criteria and other model selection techniques.

To build on these results we propose an iterative optimisation procedure, using the sum of log-likelihoods as an objective function, to further improve the partitioning.

def plot\_dendrogram(self, compound\_key,top=0.9,bottom=0.1,left=0.125,right=0.9):

"""

Extracts data from clustering to plot dendrogram

"""

partition = self.partitions[compound\_key]

(linkmat, names, threshold) = self.plotting\_info[compound\_key]

fig = plt.figure(figsize=(5.85,4.15))

fig.subplots\_adjust(top=top,bottom=bottom,left=left,right=right)

dendrogram(

linkmat,

color\_threshold=threshold,

leaf\_font\_size=8,

leaf\_rotation=90,

leaf\_label\_func=lambda leaf: names[leaf][:4]+names[leaf][5:],

count\_sort=True

)

plt.suptitle('Dendrogram', fontsize=16)

plt.title('Distance metric: {0} Linkage method: {1} Number of classes: {2}'.format(compound\_key[0],

compound\_key[1], compound\_key[2]), fontsize=8)

plt.axhline(threshold, color='grey', ls='dashed')

plt.xlabel('Gene',fontsize=16)

plt.ylabel('Distance',fontsize=16)

return fig