Producing a TangleGram in R

bh

October 9, 2014

First, we need to load up the required libraries, in this case:

library(dendextend)

##   
## Welcome to dendextend version 0.17.5  
##   
## Type ?dendextend to access the overall documentation and  
## vignette('dendextend') for the package vignette.  
## You can execute a demo of the package via: demo(dendextend)  
##   
## More information is available on the dendextend project web-site:  
## https://github.com/talgalili/dendextend/  
##   
## Contact: <tal.galili@gmail.com>  
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues  
##   
## To suppress the this message use:  
## suppressPackageStartupMessages(library(dendextend))  
##   
##   
## Attaching package: 'dendextend'  
##   
## The following object is masked from 'package:stats':  
##   
## cutree

library(RColorBrewer)

It is also a good idea to verify our working directory, since we'll be loading data from files located here:

getwd()

## [1] "/Users/bh/Dropbox/0 - Publications\_bh/Shiny/EpiQuant/data/demo\_data"

Now we can get our feet wet with some actual R code: Load up the data from our tab-delimited text files:

epi <- read.table('Epi\_Sim\_Data\_58.txt', sep = '\t', header = T)  
cgf <- read.table("Hex-SimTable\_58.txt", sep='\t', header = T)

Now, we set up the data to include both row and column names, and transform the matrices into dendrograms using the base functions dist() and hclust().

rownames(epi)<-colnames(epi)  
rownames(cgf)<-colnames(cgf)  
  
d\_cgf<- as.dendrogram(hclust(dist(cgf)))  
d\_epi<- as.dendrogram(hclust(dist(epi)))

An important reminder to using dendextend is that we have to make sure our labels are in the character format.

labels(d\_cgf) <- as.character(labels(d\_cgf))  
labels(d\_epi) <- as.character(labels(d\_epi))

Now, comes the fun part(s). We have our two trees, but we need to untangle them, or, configure their layout so that the terminal leaves line up as close as possible. The first step to this is by redrawing the trees at random 150 times. This can take a few seconds, so be patient.

dendo\_random <- untangle\_random\_search(d\_cgf, d\_epi, R = 150)

Then, we fix one of the trees, and only rotate the second tree to fit the first.

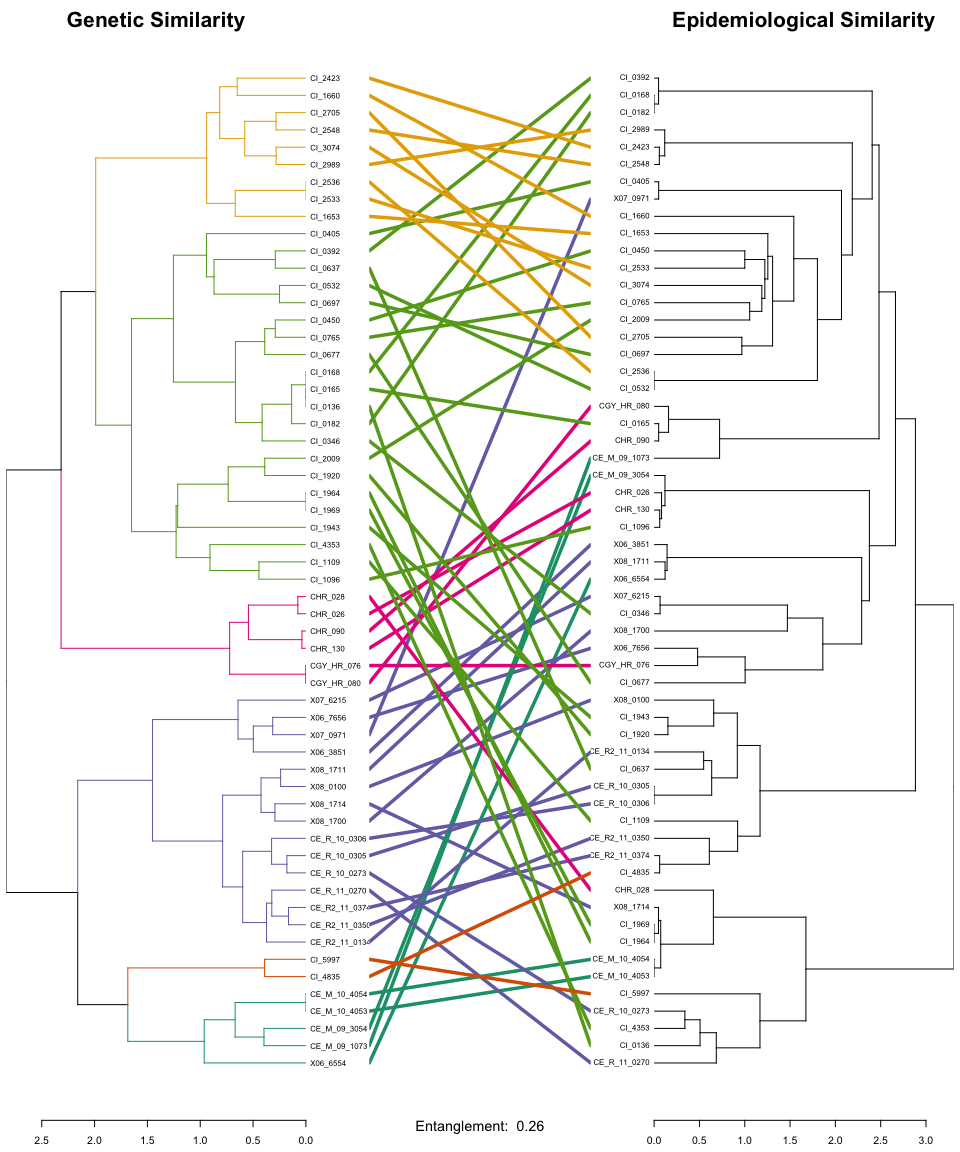
dend\_heights <- heights\_per\_k.dendrogram(dendo\_random[[2]])  
dendo <- untangle\_step\_rotate\_1side(dend1 = dendo\_random[[2]], dend2\_fixed = dendo\_random[[1]], dendextend\_heights\_per\_k.dendrogram= (dend\_heights)[[1]], k\_seq = 2:4 )

Now, we color the branches and the connecting lines according to the "Dark2" palette in RColorBrewer.

dendo[[1]] <- color\_branches(dendo[[1]], 6, col = brewer.pal(6, "Dark2"))  
col\_lines\_left2 <- brewer.pal(6, "Dark2")[cutree(dendo[[1]], 6, order\_clusters\_as\_data = F, sort\_cluster\_numbers = T)]

Finally, we can draw the actual plot:

tangleplot<- tanglegram(dendo[[1]], dendo[[2]], margin\_inner = 5,  
 color\_lines = col\_lines\_left2,   
 lab.cex = 0.75,   
 main\_left = "Genetic Similarity",   
 main\_right = "Epidemiological Similarity",   
 sub = paste("Entanglement: ", round(entanglement(dendo[[1]], dendo[[2]]), 2)),  
 cex\_sub = 0.9)



```