MAMCR for JB

Eric Brewe

9/7/2017

knitr::opts\_chunk$set(echo = TRUE)  
  
rm(list = ls())  
library(tidyverse)

## Loading tidyverse: ggplot2  
## Loading tidyverse: tibble  
## Loading tidyverse: tidyr  
## Loading tidyverse: readr  
## Loading tidyverse: purrr  
## Loading tidyverse: dplyr

## Conflicts with tidy packages ----------------------------------------------

## filter(): dplyr, stats  
## lag(): dplyr, stats

library(knitr)  
library(igraph)

##   
## Attaching package: 'igraph'

## The following objects are masked from 'package:dplyr':  
##   
## %>%, as\_data\_frame, groups, union

## The following objects are masked from 'package:purrr':  
##   
## %>%, compose, simplify

## The following objects are masked from 'package:tidyr':  
##   
## %>%, crossing

## The following object is masked from 'package:tibble':  
##   
## as\_data\_frame

## The following objects are masked from 'package:stats':  
##   
## decompose, spectrum

## The following object is masked from 'package:base':  
##   
## union

library(ggraph)  
  
setwd("~/Dropbox/R Work/MAMCRforJB")  
fci = read.csv("fci\_resp.csv", header = T)

First to extract the bipartite network from FCI data

fci %>%  
 filter(Session == "session-0") %>%  
 select(Q2:Q29) -> fciPre  
fci %>%  
 filter(Session == "session-1") %>%  
 select(Q2:Q29) -> fciPost  
   
#turn it into a matrix  
FCIPostNet <- matrix(data = NA, nrow = nrow(fciPost), ncol = 36)  
checkFCI <- function(x) {  
 as.numeric(x==c("1","2","3","4"))}

#takes data a spread it out into a matrix of zero and one’s  
for(k in 1:9){  
 kmin <- k\*4-3  
 kmax <- k\*4  
 FCIPostNet[,kmin:kmax] <- t(sapply(fciPost[,k],checkFCI))}  
  
#This removes items from the matrix  
#FCIPostNet = FCIPostNet[,-c(1,7,10,14,18,21,28,31,34)]

#take the matrix and turn it into a graph object  
FCIPostGr = graph.incidence(FCIPostNet)

#This is where the project happens. You have an FCI post graph and you can think of #it as a matrix of 108 people by 36 responses. We’re multiplying by the transpose. #Now we have symmetric matrices. Square matrices. This gives me the edge weights between people. Each element is a count of how many times one person agreed with another person (had same answer for same question… goes from 0 to 9)  
PeopleGr = bipartite.projection(FCIPostGr)$proj1  
QuestionGr = bipartite.projection(FCIPostGr)$proj2  
  
#Label the nodes, making life easier later.  
#V(QuestionGr)$id = c("Q2A","Q2B","Q2C","Q2D","Q3A","Q3B","Q3C","Q3D","Q6A","Q6B","Q6C","Q6D","Q7A","Q7B","Q7C","Q7D","Q8A","Q8B","Q8C","Q8D","Q12A","Q12B","Q12C","Q12D","Q14A","Q14B","Q14C","Q14D","Q27A","Q27B","Q27C","Q27D","Q29A","Q29B","Q29C","Q29D" )  
  
  
#Now remove the right answers, and node 12 which was isolated.  
#QuestionGr %>%  
 # delete\_vertices(c(1,7,10,14,18,21,28,31,34,12)) -> QuGrRed

Next up sparsification

We want to figure out whether something is a significant edge. We’re going evaluate this edge on the basis of this vector of probabilities in order to figure out if the edge is significant, within this list of possibly significant edges. Lets evaluate each edge and evaluate it with the probability that is is more than … Sparcification takes the graph that has 108 nodes and some number of edges

We’re assigning a significance values to each edge, based on

Creates a disparity filter

The probability of an edge = the edge weight / sum of all the edge weights (not including that one) across the subject/node (the sum of the elements in the row) (how important is this compared to other things)

Keep the ones where the prob is above some alpha level. “In this paper, we introduce a new method for backbone extraction that does not rely on any particular null model, but instead uses the empirical distribution of similarity weight to determine and then retain statistically significant edges”

(Method can be found at: <http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0016431>)

backboneNetwork<-function(g,alpha){  
  
 A<-get.adjacency(g,attr="weight")  
 A<-as.matrix(A)  
 #Now, convert this matrix to a probability matrix,p-matrix. The function rowSums(A) returns a vector with the sum of all the entries in a row  
 p<-A/rowSums(A)  
   
 #This is the evaluation function. It takes a vector of probabilities, Q, and compares each entry with the other entries in the vector.  
  
 F\_hat<-function(Q){  
   
 x<-vector()  
 for(j in 1:length(Q)){  
 x[j]<-length(which(Q!=0 & Q<=Q[j]))/length(which(Q>0))  
 }  
 return(x)  
 }  
 #The following produces a matrix, sigMatrix, with values 1 for the links that are to be kept and 0 for the links that we throw away.  
 sigMatrix<-matrix(nrow = length(V(g)), ncol=length(V(g)))  
 for(i in 1:length(V(g))){  
 sigMatrix[i,]<-F\_hat(p[i,])  
 }  
 sigMatrix2<-sigMatrix > 1 - alpha  
   
 mode(sigMatrix2)<-"numeric"  
 sigMatrix2[is.na(sigMatrix2)] <- 0  
 #Now multiply the original adjacency matrix with sigMatrix to get rid of the insignificant links  
 B<-sigMatrix2\*A  
 #Now create a graph from the new matrix.  
 h<-graph.adjacency(B,mode=c("upper"),weighted=TRUE) #it can be lower, upper or directed. We should experiment with different values.   
 #V(h)$id<-V(g)$id  
 return(h)  
 }  
#h is the sparsified network

#BBQuestions = backboneNetwork(QuGrRed, 0.1)  
BBPeople = backboneNetwork(PeopleGr, 0.01) #creating the sparsified network

#A = infomap.community(BBQuestions, e.weights = V(BBQuestions)$weight)  
  
B = infomap.community(BBPeople, e.weights = V(BBPeople)$weight)  
C = as.vector(membership(B))  
C = as.data.frame(C)  
  
V(PeopleGr)$mem = membership(B) # takes cluster and makes that an attribute in the people graph, so that if I wanted to plot things based on membership within the infomap network I can

When we use alpha = 0.01, we end up with 14 communities. They are listed below.

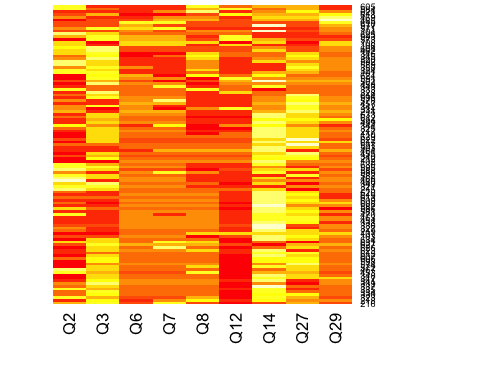
communities(B)

## $`1`  
## [1] 12 14 18 23 26 27 29 31 33 39 40 44 54 61 63 81 84  
## [18] 85 92 99 102 103 106 108  
##   
## $`2`  
## [1] 3 6 15 21 28 30 43 51 57 66 70 71 86 87 90 95 96  
##   
## $`3`  
## [1] 11 17 41 49 55 67 69 72 78 105  
##   
## $`4`  
## [1] 2 9 32 45 52 59 80 93 98 104  
##   
## $`5`  
## [1] 8 13 20 36 48 77 89 107  
##   
## $`6`  
## [1] 10 24 34 46 65 79 91  
##   
## $`7`  
## [1] 22 25 35 47 58 74 100  
##   
## $`8`  
## [1] 1 19 42 73 75  
##   
## $`9`  
## [1] 16 37 38 50 82  
##   
## $`10`  
## [1] 5 7 62 76 101  
##   
## $`11`  
## [1] 4 60 64 88  
##   
## $`12`  
## [1] 53 56 94 97  
##   
## $`13`  
## [1] 68 83

fci %>%  
 filter(Session == "session-1") %>%  
 select(Subject) -> fciPostID  
fciPostID

## Subject  
## 1 101  
## 2 102  
## 3 103  
## 4 104  
## 5 106  
## 6 107  
## 7 108  
## 8 110  
## 9 212  
## 10 214  
## 11 215  
## 12 216  
## 13 217  
## 14 218  
## 15 219  
## 16 320  
## 17 321  
## 18 323  
## 19 324  
## 20 325  
## 21 327  
## 22 328  
## 23 330  
## 24 331  
## 25 333  
## 26 334  
## 27 335  
## 28 336  
## 29 337  
## 30 338  
## 31 339  
## 32 340  
## 33 341  
## 34 342  
## 35 343  
## 36 344  
## 37 345  
## 38 346  
## 39 347  
## 40 348  
## 41 349  
## 42 350  
## 43 451  
## 44 453  
## 45 455  
## 46 457  
## 47 458  
## 48 459  
## 49 460  
## 50 462  
## 51 463  
## 52 464  
## 53 465  
## 54 467  
## 55 468  
## 56 469  
## 57 470  
## 58 502  
## 59 503  
## 60 571  
## 61 572  
## 62 573  
## 63 574  
## 64 577  
## 65 578  
## 66 581  
## 67 582  
## 68 584  
## 69 585  
## 70 586  
## 71 587  
## 72 588  
## 73 589  
## 74 591  
## 75 592  
## 76 593  
## 77 594  
## 78 595  
## 79 596  
## 80 597  
## 81 598  
## 82 604  
## 83 605  
## 84 606  
## 85 607  
## 86 608  
## 87 609  
## 88 610  
## 89 612  
## 90 613  
## 91 614  
## 92 615  
## 93 617  
## 94 618  
## 95 619  
## 96 620  
## 97 621  
## 98 622  
## 99 623  
## 100 624  
## 101 625  
## 102 626  
## 103 627  
## 104 629  
## 105 630  
## 106 631  
## 107 633  
## 108 634

fciPost = bind\_cols(fciPost,C,fciPostID)  
fciPost %>% arrange(C) %>% select(Q2:Q29) %>% as.matrix() ->a  
fciPost %>% arrange(C) %>% select(Subject) -> matRowNames  
rownames(a) <- matRowNames$Subject  
  
#this gives a heatmap of the 9 question data set.  
heatmap(a, Rowv = NA, Colv = NA)



#this gives a heatmap of the 36 question data set (all answers smeared out)  
heatmap(FCIPostNet, Rowv = NA, Colv = NA)

