R code used for estimation, bootstrapping, and simulation in "Estimating Within-Household Contact Networks from Egocentric Data"

The Annals of Applied Statistics

Gail E. Potter, Mark S. Handcock,
Ira M. Longini, Jr., and M. Elizabeth Halloran

```
make.mat = function(v)  {
## When sent a vector v, returns a symmetric square matrix with those elements
## We will use this function to convert a vector of probability parameters into
## a matrix of conditional contact probabilities.
upper = matrix(0, nrow=num.age.cats, ncol=num.age.cats)
upper[1, 1:5] = v[1:5]
upper[2,2:5] = v[6:9]
upper[3,3:5] = v[10:12]
upper[4,4:5] = v[13:14]
upper[5,5] = v[15]
lower=t(upper)
mat=upper+lower
diag(mat)=diag(upper)
return(mat)
}


lik.net = function(pz, dat) {

## takes as input a vector (dat) which contains the age category of the respondent,
## the numbers of family members in each age category, and the numbers of observed
## contacts to family members in each age category, as well as
## pz = matrix of age-specific contact probabilities

## returns the likelihood of the observed network (under the independence assumption
## assuming that all the household members in dat are at home.

p.age = as.numeric(dat[1]) ## index associated with participant age category
n.family.members = dat[2:6]  ## numbers of family members in each age category
n.contacts = dat[7:11]  ## numbers of contacts in each age category
n.noncontacts = dat[12:16]  ## numbers of non-contacts in each age category

probs=pz[p.age,]  ## probs of age cat of resp, contacting other age cats

lik = prod(dbinom(n.contacts, size=n.family.members, p=probs))
#prod(probs^n.contacts * (1-probs)^n.noncontacts)

return(lik)

}

lik.i = function(i, pz, ph) {

## takes as input an index (i) of the data frame, whose row contains the age category
## of the respondent,
## the numbers of family members in each age category, and the numbers of observed
## contacts to family members in each age category, as well as
# pz = matrix of age-specific contact probabilities
# ph = probabilities of being "at home"

## returns the likelihood of the data reported by respondent i.
## This is computed by summing over all possible combinations of underlying contact network
## and household members being home or away from home, which are consistent with the
## observed data.

dat = latent.data[i,]
dat

p.age = as.numeric(dat[2])  ## index associated with participant age category
n.family.members = as.numeric(dat[3:7])   ## numbers of family members in each age category
n.contacts = as.numeric(dat[8:12])   ## numbers of contacts in each age category
n.noncontacts = as.numeric(dat[13:17])   ## numbers of non-contacts in each age category

if (sum(n.contacts)==0) any.reported.contacts = 0 else any.reported.contacts=1

n.family.members
n.contacts
n.noncontacts

## generate a matrix of all combinations of household members being away from home

if (sum(n.noncontacts[2:5])>0) nrep1 = prod((n.noncontacts[2:5]+1)) else nrep1=1
if (sum(n.noncontacts[3:5])>0) nrep2 = prod((n.noncontacts[3:5]+1)) else nrep2=1
if (sum(n.noncontacts[4:5])>0) nrep3= prod((n.noncontacts[4:5]+1)) else nrep3=1
if (sum(n.noncontacts[5])>0) nrep4= prod((n.noncontacts[5]+1)) else nrep4=1

away = cbind(
rep(0:n.noncontacts[1], nrep1),
rep(sort(rep(0:n.noncontacts[2], (n.noncontacts[1]+1))), nrep2),
rep(sort(rep(0:n.noncontacts[3], prod((n.noncontacts[1:2]+1)))),nrep3),
rep(sort(rep(0:n.noncontacts[4], prod(n.noncontacts[1:3]+1))), nrep4),
sort(rep(0:n.noncontacts[5], prod(n.noncontacts[1:4]+1))))

away

## loop through the matrix of all possible combinations; for each one generate the two possible
##  underlying networks
## for each respondent who is away, the underlying contact network could have a 0 or a 1 and
## compute the likelihood of each.

lik = 0
for (a in 1:dim(away)[1]) {
n.home = n.family.members - away[a,]
n.away=away[a,]

## create matrix of all possible underlying contact networks for given away members

## only members w 0 reported contacts can be away
## underlying contact network contains, at a minimum, all reported contacts
## its maximum is the reported contacts plus all those who are away.

## extra.contacts is a matrix with all possible combinations of additional (unreported)
## contacts in the underlying network - which are unobserved due to the family member
## being away.  It has ncol=# age categories, and nrow=prod(n.away+1), since
## if there are k members away in age group a, we have k+1 away
## status possibilities (0:k+1)

extra.contacts = cbind(
rep(0:n.away[1], prod(n.away[2:5]+1)),
rep(sort(rep(0:n.away[2], n.away[1]+1)), prod(n.away[3:5]+1)),
rep(sort(rep(0:n.away[3], prod(n.away[1:2]+1))), prod(n.away[4:5]+1)),
rep(sort(rep(0:n.away[4], prod(n.away[1:3]+1))), prod(n.away[5]+1)),
sort(rep(0:n.away[5], prod(n.away[1:4]+1))) )

all.possible.underlying.contact.nets = matrix(rep(n.contacts,
prod(n.away+1)),ncol=5,byrow=T) + extra.contacts
for (underlying.net in 1:prod(n.away+1)) {
underlying.contacts = all.possible.underlying.contact.nets[underlying.net,]
underlying.noncontacts = n.noncontacts - extra.contacts[underlying.net,]

dat.a = c(p.age, n.family.members, underlying.contacts, underlying.noncontacts)
lik = lik + lik.net(pz, dat.a)* prod(dbinom(n.home,
size=n.family.members, p=ph))*ph[p.age]
   }
}

## We're applying the Law of Total Probability:
## P(observed sub-network) = P(obs subnet | resp home)P(resp home) +
## P(obs subnet|resp away)P(resp away)
## When at least one contact is reported, P(obs subnet | resp away)=0, so the likelihood
## is the first term only.
## When no contacts are reported, we need to add the second term.

if (any.reported.contacts==0) lik=lik+(1-ph[p.age])

return(lik)

}


logit=function(p) log(p/(1-p))
expit=function(x) exp(x)/(exp(x)+1)

latent.loglik = function(x) {
## compute the likelihood of the entire data set
## takes as input a vector of age-specific contact probabilities, transforms
## these to a symmetric matrix.
## also takes as input a vector of age-specific probabilities of being at home.
p=expit(x)
#print(round(p,2))

pz.vec=p[1:15]
pz = make.mat(pz.vec)
ph=p[16:20]

loglik = 0

for (i in 1:dim(latent.data)[1]) {
loglik = loglik + log (lik.i(i, pz, ph))
```

```
}
return(loglik)
}



load("latent.data")

## this data set takes the following form:
## column 1 - ID, col 2 - participant age
## cols 3-7: number of household members in each age category in respondent's households
## cols 8-12: numbers members in each age category who were contacted by respondent
## cols 13-17:  numbers of family members in each age category who were not contacted by
## respondent
## col 18: day of week
## col 19: holiday?
## col 20: household size

## The code below can also be used to do the estimate for small households
## (with 2-3 members), by first subsetting with this line:
## latent.data = latent.data[latent.data$hh.size < 4,-(18:20)]

## We could also run the code for households with 4+ members:
## latent.data = latent.data[latent.data$hh.size >= 4,-(18:20)]

## Holiday, nonholiday, weekend, and weekday estimates are done similarly.

## for overall estimate, need to remove cols 18-20
latent.data = original.latent.data[,-(18:20)]

attach(latent.data)

num.age.cats = 5
nr.trace=1
nr.maxit=50
nr.method="BFGS"
nr.reltol=sqrt(.Machine$double.eps)
start.val = logit(rep(.95,20))

mle=optim(start.val, f=latent.loglik,
  method=nr.method,
  control=list(trace=nr.trace,REPORT=1,fnscale=-1,maxit=nr.maxit,reltol=nr.reltol))

## later repeat analysis for small households (mlehh4) and large households (mlehh5)
x=mle$par
p=expit(x)
round(p,2)

## contact probs
make.mat(round(p[1:15],2))

## at home probs
round(p[16:20],2)
```

```
## likelihood under null hypothesis
lik.null = mle$value

## bootstrap standard errors
nboot=500
boot=matrix(nrow=nboot, ncol=20)

original.data = latent.data

for (b in 1:nboot) {
latent.data=original.data[sample(1:dim(original.data)[1], replace=T),]
mle=optim(start.val,f=latent.loglik,
  method=nr.method,
  control=list(trace=nr.trace,REPORT=1,fnscale=-1,maxit=nr.maxit,reltol=nr.reltol))
boot[b,]=mle$par
}

low = apply(expit(boot), 2, quantile, 0.025)
up = apply(expit(boot), 2, quantile, 0.975)

contact.prob.estimates=make.mat(round(p[1:15],2))
agecats = c("0-5","6-11","12-18","19-35","36+")
row.names(contact.prob.estimates)=agecats
colnames(contact.prob.estimates)=agecats

home.probs=round(p[16:20],2)
names(home.probs)=agecats

contact.prob.estimates
home.probs

contact.prob.low=make.mat(round(low[1:15],2))
row.names(contact.prob.low)=agecats
colnames(contact.prob.low)=agecats

contact.prob.up=make.mat(round(up[1:15],2))
row.names(contact.prob.up)=agecats
colnames(contact.prob.up)=agecats

home.probs.low=round(low[16:20],2)
names(home.probs.low)=agecats

home.probs.up=round(up[16:20],2)
names(home.probs.up)=agecats

contact.prob.estimates
home.probs
contact.prob.low
home.probs.low
contact.prob.up
home.probs.up

# Do LRT for household size effect:
loglik.restricted = mle$value  # max over restricted space
loglik.unrestricted = mle.big$value + mle.small$value

## (log lik of data from 4+ sized hhs plus log lik of data from 2-3 member hhs)
## so log of product of those two likelihoods

chi.stat = -2*(loglik.restricted - loglik.unrestricted)
chi.stat

pchisq(chi.stat, df=17, lower.tail=F)

## simulations to assess weak identifiability of model

p.mle = expit(mle$par)
contact.probs=p.mle[1:15]
num.age.cats=5
home.probs = c(1, .9, .8, .7, .6)

p=c(contact.probs, home.probs)
contact.mat=make.mat(contact.probs)

n.members = latent.data[,3:7]
n=dim(latent.data)[1]

## simulate n.samples data sets from model
n.samples=1000
p.mle.sim = matrix(nrow=n.samples,ncol= 20)
converged = NULL
n=dim(latent.data)[1]
w=matrix(NA,nrow=n,ncol=5)
attach(latent.data)

for (s in 1:n.samples) {

for (i in 1:n) {
## sample who is at home
## first sample respondent at home:
resp.age.cat = as.numeric(part.age[i])
resp.home = rbinom(n=1,p=home.probs[resp.age.cat],size=1)

if (resp.home==0) w[i,]=rep(0,5) else {

## indicate which age categories have members in hh
have.members = n.members[i,]!=0

at.home = rep(0,5)
n.obs=sum(have.members)
size=n.members[i,][have.members]
probs=home.probs[have.members]

at.home[have.members] = rbinom(n.obs,size,probs)
## conditional on "at home" members, sample contacts

contact.probs.i = contact.mat[resp.age.cat,]
w[i,]=rep(0,5)
size=at.home[have.members]
probs=contact.probs.i[have.members]

w[i,have.members] = rbinom(n.obs, size, probs)
} # end of else statement
}

latent.data[,8:12]=w
latent.data[,13:17]=latent.data[,3:7]-w

## fit latent variable model
est = optim(logit(rep(.95,20)),f=latent.loglik,
  method=nr.method,
  control=list(trace=nr.trace,REPORT=1,fnscale=-1,maxit=nr.maxit,reltol=nr.reltol))

p.mle.sim[s,] = expit(est$par)
converged[s] = est$convergence
}

table(converged)

## compare means from fitted "at home" probs to the true "at home" probs.

compare.mat =round( cbind(
apply(p.mle.sim, 2, mean),
apply(p.mle.sim, 2, quantile, 0.025),
apply(p.mle.sim,2,quantile,0.025, 0.975),
p.mle),2)

compare.mat
```