CALIBRATION OF ZULF'S MARKOV MORAL THEORY FOR HUMAN RACE

ZULFIKAR MOINUDDIN AHMED

Our model is a 55 parameter model for Human Moral Values. The lower left triangular values of a 10 by 10 transition Probability Matrix are the parameters to be fit.

Here we provide *working* code.

```
# load data
wvs7<-readRDS("wvs7.rds")
vars<-c("Q290","Q46", "Q57","Q177","Q178","Q179","Q180","Q181","Q182","Q183","Q184","Q185","Q

# Create ethnicity table
# by mapping various detailed
# country-based ethnicities to
# broader groups
library(haven)
# Problem is WVS Q290 has too many
# gradations when we want to deal
# with a few classes that can allow
# us to overcome ethnic prejudices
ethnicities<-unique(as.character(as_factor(wvs7$Q290)))

reduced_ethnicities<-function(){
  mapeth<-rep("Other",length(ethnicities))
  mapeth[c(1,7,15,33,38,43,65,107,154,210,214)]<-"White"
  mapeth[c(3,223,222,51,52,53,54,55,56,45,39,36,11)]<-"Black"
  mapeth[c(6,13,19,20,21,22,67,70)]<-"Indian"
  mapeth[c(4,17,48,97,98,99,100)]<-"Arab"
  mapeth[c(5,14,16,62,63,64,68,72,73,74,75,76,77,
           78,79,80,81,82,83)]<-"East Asian"
  data.frame(key=ethnicities,val=mapeth)
}

ethnicity_map<-function( v ){
  emap<-reduced_ethnicities()
  w<-sapply(as.character(as_factor(v)),function(x) { out<-emap[which(emap$key==x),]$val; if(i
  #print(head(w))
  #w<-append(w,"Other")
  as.factor(unlist(w))
}
```

Date: May 5, 2021.

```
polv<-na.omit(wvs7[,vars])
polv$eth<-ethnicity_map(as_factor(polv$Q290))

# Fit P to distributions to Q177-Q195 in WVS 7
library(markovchain)

t10 <-1:10
nrm<-function(v)v/sum(v)

transitionMatrixFromPars<-function(x){
  P<-matrix(0,nrow=10,ncol=10)
  P[1,1]<-x[1]
  P[1:2,2]<-x[2:3]
  P[1:3,3]<-x[4:6]
  P[1:4,4]<-x[7:10]
  P[1:5,5]<-x[11:15]
  P[1:6,6]<-x[16:21]
  P[1:7,7]<-x[22:28]
  P[1:8,8]<-x[29:36]
  P[1:9,9]<-x[37:45]
  P[1:10,10]<-x[46:55]
  #symmetrize
  for (k in 2:10){
    P[k,1:k]<-P[1:k,k]
  }
  for (k in 1:10){
    P[k,]<-nrm(P[k,])
  }
  P
}

parsFromTransitionMatrix<-function(P){
  x<-c()
  for (k in 1:10){
    x<-append(x, P[1,1:k])
  }
  x
}

# We let x be the parameters of the subdiagonal
# of P in form appropriate for an optimiser
moral_markov_obj<-function( x ){

  P <- transitionMatrixFromPars( x )
  states <- as.character(1:10)
```

```
mcVals = new("markovchain",
             states = states,
             transitionMatrix = P,
             name = "Vals")
print('initialised markov chain')
Imtx <- matrix( 0, nrow=18,ncol=10)
Jmtx <- matrix( 0, nrow=18, ncol=10)
I1<-nrm(table(polv[,"Q177"]))
Imtx[1,] <- I1
print('set init distribution for Y1')
M<-500
Y1<-sample(1:10,M,replace=TRUE,prob=I1)
print('sampled Y1')
all.Y<-matrix(0,nrow=M,ncol=18)
all.Y[,1]<-Y1
for (kk in 1:M) {
  #print(kk)
  outs <- rmarkovchain(n = 17, object = mcVals, what = "list")
  all.Y[kk,2:18]<-outs
}

for ( r in 2:18 ){
  iv<-nrm(table(polv[,vars[r]]))
  jv<-nrm(table(all.Y[,r]))
  ivn<-rep(0,10)
  jvn<-rep(0,10)
  for ( kc in 1:10 ){
    a<-iv[as.character(kc)]
    b<-jv[as.character(kc)]
    if (!is.null(a)){
        ivn[kc]<-a
    }
    if (!is.null(b)){
      jvn[kc]<-b
    }

  }
  Imtx[r,]<-ivn
  Jmtx[r,]<-jvn
}

print('calculating l2 distance')
l2.dist <- 0
hits<-0
for (r in 1:18){
  d1<- norm(abs(Imtx[r,]),type="2")
  d2<- norm(abs(Jmtx[r,]),type="2")
  d <- norm( abs(Imtx[r,] - Jmtx[r,] ), type="2")
```

```
    #print(paste("r=",r,"d1=",d1,"d2=",d2,"d=",d))
    if (!is.na(d)){
      hits<-hits+1
      l2.dist <- l2.dist + d^2
    }
  }
  l2.dist<-sqrt(l2.dist)
  print(paste('hits=',hits))
  print(l2.dist)
  l2.dist
}




library(nloptr)
# Get an init value
lambda <- 0.52
p11 <- 0.6
P <- matrix(0,nrow = 10, ncol=10)
P[1,1] = p11
for (k in 2:10){
  for (r in 1:k){
    P[k,r] <- exp(-lambda*(k+r))*p11
    P[r,k] <- P[k,r]
  }
}
for ( r in 1:10){
  P[r,]<-P[r,]/sum(P[r,])
}
P0<-P

x0<-parsFromTransitionMatrix(P0)
xlen <- length(x0)
l0 <- rep(0,xlen)
u0 <- rep(1,xlen)

eval_g0<-function( x ){
  P1<-transitionMatrixFromPars(x)
  out<-0
  for (k in 1:10){
    out <- out + (sum(P1[k,]))^2-1.0
  }
  out
}

# Solve using NLOPT_LN_COBYLA without gradient information
res1 <- nloptr( x0=x0,
                eval_f=moral_markov_obj,
                lb = l0,
                ub = u0,
                eval_g_ineq = eval_g0,
                opts = list("algorithm"="NLOPT_LN_COBYLA",
                            "xtol_rel"=1.0e-8,"print_level"=1))
print( res1 )
```

1. Some Provisional Results

```
> res1

Call:

nloptr(x0 = x0, eval_f = moral_markov_obj, lb = l0, ub = u0,
    eval_g_ineq = eval_g0, opts = list(algorithm = "NLOPT_LN_COBYLA",
        xtol_rel = 1e-08, print_level = 1))


Minimization using NLopt version 2.4.2

NLopt solver status: 5 ( NLOPT_MAXEVAL_REACHED: Optimization
stopped because maxeval (above) was reached. )

Number of Iterations....: 100
Termination conditions:  xtol_rel: 1e-08
Number of inequality constraints:  1
Number of equality constraints:    0
Current value of objective function:  0.944810719116277
Current value of controls: 0.998691 0.6017886 0.1230545 0.6382924 0.1523181 0.06756478
0.6521802 0.1243201 0.07645729 0.03616285 0.6788139 0.1106364
0.05256054 0.04208854 0.02963721 0.6604343 0.1345903
0.07503765 0.03138417 0.04702849 0.01516008 0.9375384
0.1186496 0.05600062 0.03639242 0.02682043 0.01248168
0.01596603 0.6248498 0.1129891 0.0487273 0.04372557
0.03241158 0.02317385 0.01029844 0.00726488 0.6085975
0.09877426 0.06007426 0.0613663 0.02351311 0.01476094
0.00509009 0.007064568 0.001769594 0.6950101 0.1412725
0.08852981 0.03363069 0.03376865 0.01791395 0.01243466
0.007366109 0.003654178 0.002536741
```