CALIBRATION OF ZULF'S MARKOV MORAL THEORY FOR HUMAN RACE TO R-SQUARED PROXY 0.91

ZULFIKAR MOINUDDIN AHMED

1. Markov Moral Theory as a Statistical Model

Social Science is not physics. We are not going to get accuracy to $\epsilon = 1e - 12$ anytime soon in social science. Instead we are interested in parsimonious models with fewer parameters than the data, and cross our fingers, hope, pray, sing beautiful songs to the spirits of the sun and stars, cry to the forests and rivers, and do various rituals so that any sort of deity gives us their favours. While we are sometimes quite hardnosed after the fact, I am not shy about saying that I have all sorts of chocolates and fruits reserved for Athena, one of my favourites, just in case. We don't take chances in science when we are looking for results you see.

Anyway our model is 55 parameter model and we are trying to fit 150 data points. Now if we just fit linear least-square lines, that would be 15 vectors of length 10, and we'd fit 2 parameters each. Here we're fitting $55/15 = 3.67$ parameters per line effectively. Therefore we're not going to be expecting an error of zero because we'll have to account for noise.

Let's quickly go over the most simple intuition about

$$R^2 = 1 - \text{residualsumsquared/datasumsquared}. \tag{1}$$

Most of us have fit so many linear univariate fits that we have all sorts of calluses in our mind with nodding about how $R^2 = 0.6$ is pretty good for finance predictive models and how $R^2 = 0.95$ in a finance predictive model is trying to bamboozle you out of your money. But things are a little different when dealing with data about Human Nature Morals.

I devised an R-squared proxy by simple analogy to the linear situation. For each of the vectors I just record (1) for each line directly and report their average over the 15 vector fits. The average is a marvelous $R^2 = 0.91$.

It's a proxy R-squared, because it's hard to know what to make of the objective function which is just an L^2-norm whose meaning is obscure.

2. The R Code

Usually people present the code in the appendix but it's actually more revealing of what I am doing so it's worth just putting it here.

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

# Create ethnicity table
```

Date: May 6, 2021.

```
# 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))
}



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")
  Imtx <- matrix( 0, nrow=18,ncol=10)
  Jmtx <- matrix( 0, nrow=18, ncol=10)
  I<-nrm(table(polv[,"Q177"]))
  Imtx[1,] <- I
  M<-10000
  Y1<-sample(1:10,M,replace=TRUE,prob=I)
  all.Y<-matrix(0,nrow=M,ncol=18)
  all.Y[,1]<-Y1
  for (kk in 1:M) {
    #print(kk)
    outs <- markovchainSequence(n = 17, markovchain = mcVals,
                                t0 = all.Y[kk,1],
                                include.t0 = TRUE )
    all.Y[kk,1:18]<-outs
  }
```

```
for ( r in 1:18 ){
  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
l2.ldist <- 0
sld.dist<-0
rsq.proxy<-0
hits<-0
for (r in 1:18){
  d1<- norm(abs(Imtx[r,]),type="2")
  d2<- norm(abs(Jmtx[r,]),type="2")
  A <-log(Imtx[r,]+1e-6) - log(Jmtx[r,]+1e-6)
  d <- norm( Imtx[r,] - Jmtx[r,], type="2")
  suplogd <-norm( as.matrix(A), type="I")
  dl <- suplogd + 10*d
  #print(paste("r=",r,"d1=",d1,"d2=",d2,"d=",d))
  if (!is.na(d)){
    hits<-hits+1
    sld.dist <- sld.dist + suplogd
    l2.ldist <- l2.ldist + dl
    l2.dist <- l2.dist + d^2
    rsqp <- 1 - d^2/d1^2
    rsq.proxy <- rsq.proxy + rsqp
  }
}
#l2.ldist<-l2.ldist
print(paste("sld=",sld.dist))
print(l2.ldist)
l2.dist<-sqrt(l2.dist)
print(l2.dist)
```

```
  rsq.proxy <- rsq.proxy/15
  print(paste('rsq.proxy=',rsq.proxy))
  l2.ldist
}




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,
                opts = list("algorithm"="NLOPT_LN_NELDERMEAD",
                            "xtol_rel"=1.0e-6,
                            "maxeval"=5000,
```

```
                                "print_level"=1))
print( res1 )
```

3. The fitted model

We fit a Markov Transition Probability Matrix.

	1	2	3	4	5	6	7	8	9	10
1	15.03	14.05	14.91	12.19	13.57	8.51	6.02	1.96	1.49	12.27
2	65.04	9.90	0.72	0.62	6.46	1.23	2.33	10.98	1.12	1.60
3	64.60	0.68	4.29	4.18	5.84	3.73	2.64	6.13	5.13	2.79
4	71.85	0.79	5.68	9.78	0.34	3.07	2.41	2.60	1.62	1.86
5	67.97	6.99	6.75	0.29	3.24	2.39	3.75	1.94	3.79	2.89
6	72.46	2.27	7.33	4.44	4.06	2.23	2.45	1.50	2.79	0.47
7	65.49	5.47	6.62	4.45	8.13	3.12	1.29	2.22	2.31	0.89
8	26.84	32.45	19.36	6.04	5.30	2.40	2.80	1.81	1.74	1.26
9	31.19	5.07	24.79	5.75	15.85	6.86	4.45	2.66	1.41	1.97
10	84.58	2.38	4.43	2.17	3.98	0.38	0.56	0.63	0.65	0.23

This is a beautiful thing: a Markov Chain Generator for Moral Value levels. The data are Q177–Q195. They are all structured as "Justified traditionally immoral such and such". I am not interested in details. I consider all of the moral values roughly homogeneous, even ambiguous ones such as 'divorce' because I believe there is something simpler and deeper in the way that human beings have their moral convictions. I also don't think ethnicity, religion, etc. matter all that much since I am seeking some evolutionary human nature effects that dominate the data.

4. Opimisation Results

```
Call:

nloptr(x0 = x0, eval_f = moral_markov_obj, lb = l0, ub = u0,
    opts = list(algorithm = "NLOPT_LN_NELDERMEAD", xtol_rel = 1e-06,
        maxeval = 5000, print_level = 1))


Minimization using NLopt version 2.4.2

NLopt solver status: 4 ( NLOPT_XTOL_REACHED: Optimization
stopped because xtol_rel or xtol_abs (above) was reached. )

Number of Iterations....: 2734
Termination conditions:  xtol_rel: 1e-06 maxeval: 5000
Number of inequality constraints:  0
Number of equality constraints:    0
Optimal value of objective function:  35.8706392521016
Optimal value of controls: 0.9781533 0.914155 0.139106 0.9700959 0.01016634 0.06449197
0.7933091 0.008725645 0.06271482 0.1079745 0.8828269
0.09084231 0.08765224 0.003739942 0.04211452 0.5534433
0.01730511 0.05597271 0.03390994 0.0310179 0.01704735
```

```
0.3917288 0.03269303 0.03958918 0.0266307 0.04865909
0.01868915 0.007710896 0.1276933 0.1543736 0.09210397
0.02874771 0.02519741 0.01143723 0.01329967 0.008617431
0.09693486 0.01576467 0.07705055 0.0178849 0.04925389
0.02132651 0.01384717 0.008257145 0.004396937 0.7981451
0.02246566 0.04183687 0.02051317 0.03752682 0.003599973
0.00530332 0.005971533 0.006108878 0.002156556
```

I use nloptr, the R binding of the high quality nlopt software. I won't get too much into details about that.

What is worth mentioning is that I found it useful to use the unconventional objective

$$f(x, y) = \|log(x) - log(y)\|_\infty + 10\|x - y\|_2$$

But I reported the least-square distance. There are various heuristics I used and I don't want to get distracted by what mixture of norms can have what good convergence properties. This scheme has value in some more general situations and there might be some theory for that. I used it heuristically.

5. Markov Moral Theory

My theory is that there exists a Markov Generator that can act as a transition probability for all moral values, not just the 15 I have used. What we can see with R-squared proxy being 0.91 is that statistically this is a very good fit, and so my theory is vindicated.