Dirichlet regression example

Sean van der Merwe, University of the Free State

In a school netball tournament, players were tracked in various ways. The data provided is a modified and anonymised subset of the game summaries. It constitutes composition vectors of the proportions of time a player spent walking, standing, or running during a netball match. Since each player plays multiple matches, we have random subject effects. The player position on the field is treated as a fixed effect of interest.

The modelling is done using the R2OpenBUGS approach. This means that the data and model are loaded in R, BUGS is called from R and given time to simulate the posterior distribution, and then the simulation results are reviewed and sent back to R for analysis.

```
# Dirichlet Regression Example - Sean van der Merwe, UFS

# Data input
setwd('C:/Users/vandermerwes/ownCloud/Work/Work2018/Research/ProportionsExample') # Change to your working directory
mydata <- read.csv('NetballData.csv')
attach(mydata)
n <- length(Position)
Props <- cbind(Standing,Walking,Running)
Ndim <- 3
Positions <- levels(Position); nPositions <- nlevels(Position)
BPosition <- Position; levels(BPosition) <- 1:nPositions; BPosition <- c(BPosition) # Change factor levels to
numbers for BUGS
Players <- levels(PlayerName); nPlayers <- nlevels(PlayerName)
BPlayers <- PlayerName; levels(BPlayers) <- 1:nPlayers; BPlayers <- c(BPlayers) # Change factor levels to numbers
for BUGS

library(R2OpenBUGS)  # Connects R to OpenBUGS. Ensure that OpenBUGS is installed, as well as the R2OpenBUGS library

DirModel <- function() { # Model written in BUGS notation. Note that parameters are specified as in BUGS, not R, for
example dnorm(mean,precision)
for (j in 1:Ndim) {
      for (k in 1:nPositions) {
            Positioneffect[k,j] ~ dnorm(hypermu,0.0001) # Vague prior for the fixed effects of interest. Replace
with informative prior if available.
      }
      for (k in 1:nPlayers) {
```

```
            Playereffect[k,j] ~ dnorm(0,tauSubj[j]) # Random effect by subject (player in this case)
      }
      tauSubj[j] ~ dgamma(0.001,0.001) # Prior on common variance for subject effect
}
for (k in 1:nPositions) {
      phiPositioneffect[k] ~ dnorm(0,0.0001) # Vague prior for fixed effect of Position in the precision model
}
precpenalty1 ~ dexp(pp1m) # Priors on penalties
precpenalty2 ~ dexp(pp2m)
for (i in 1:n) {
      Props[i,1:Ndim] ~ ddirich(alpha[i,1:Ndim])  # Likelihood. Each observed vector is specified as coming from a
Dirichlet with its own parameter vector alpha
      for (j in 1:Ndim) {
            log(alpha[i,j]) <- logalpha[i,j] # Transform parameters to log scale to get rid of the restriction that
they must be positive
            logalpha[i,j] ~ dnorm(logalphamu[i,j],precpenalty1) # Implement flexibility by saying each set of
parameters may vary slightly around their expected values
            logalphamu[i,j] <- log(mu[i,j]) + phiPositioneffect[BPosition[i]] # Split parameters into a mean model
and a precision model. The precision model here is just a set of intercepts for each precision.
            mu[i,j] <- ilogit(Positioneffect[BPosition[i],j] + Playereffect[BPlayers[i],j]) # Mean model. The logit
is needed for the means to be on the right scale. In this case the mean model is the sum of fixed and random
effects.
      }
      zdifmu[i] <- sum(mu[i,1:Ndim])-1 # Calculate discrepancy between sum of mean vector and 1
      zerodif[i] ~ dnorm(zdifmu[i],precpenalty2) # Implement penalty on this discrepancy
      for (j in 1:Ndim) {
            muadj[i,j] <- (mu[i,j]/(zdifmu[i]+1)) # Adjust mean vectors to sum to exactly 1
      }
}
}
write.model(DirModel,'DirModelNetball2018.txt') # Save the model to a text file in the format required by BUGS

# Preparing data for BUGS:
BUGSdata <-
list(n=n,BPosition=BPosition,nPositions=nPositions,Props=Props,zerodif=rep(0,n),Ndim=Ndim,pp1m=Ndim/100,pp2m=Ndim/10
00,hypermu=-log(Ndim-1),nPlayers=nPlayers,BPlayers=BPlayers)
# Preparing initial values for all parameters, for the BUGS Gibbs sampler to start from
initmu <- matrix(runif(nPositions*Ndim),nPositions,Ndim); initmu <- initmu/matrix(rowSums(initmu),nPositions,Ndim) #
Initial values for the means
```

```
# Using some randomness allows each chain to start at a different point. This makes it much easier to assess
convergence.
initPrec <- 0; initPhi <- exp(initPrec) # Initial values for the precisions
initAlpha <- initmu[BPosition,]*matrix(initPhi,n,Ndim) # Initial values for the Dirichlet parameters
inits <- function() {list( phiPositioneffect = rep(initPrec,nPositions), Positioneffect=qlogis(initmu),
logalpha=log(initAlpha) , tauSubj=apply(Props,2,function(x){0.1/var(x)}),
Playereffect=matrix(0,nPlayers,Ndim),precpenalty1=100/Ndim,precpenalty2=1000/Ndim ) } # Initial values specified as
a function that returns a list

# Now we are ready to simulate the posterior using BUGS. Call ?bugs to see the meaning of all options.
BUGSout <-
bugs(BUGSdata,inits,c('Positioneffect','muadj','phiPositioneffect'),n.iter=47000,'DirModelNetball2018.txt',debug=TRU
E,n.burnin=22000, n.chains=5, n.thin=2)
# NB: The bugs process takes a long time as I'm doing a lot of simulations. Lower the number of chains if you need
more speed.
# MORE NB: Once bugs is done (showing graphs for evaluating convergence) then close it before attempting to run more
R code.

nsims <- BUGSout$n.sims # Obtain final number of simulation vectors produced
Positioneffect <- BUGSout$mean$Positioneffect
BayesMu <- apply(BUGSout$mean$muadj,2,tapply,Position,mean) # Expected values of interest, split by playing position
BayesLower <- apply(apply(BUGSout$sims.list$muadj,2:3,quantile,0.025),2,tapply,Position,mean) # Intervals of
interest
BayesUpper <- apply(apply(BUGSout$sims.list$muadj,2:3,quantile,0.975),2,tapply,Position,mean)
rownames(Positioneffect) <- rownames(BayesMu)
colnames(Positioneffect) <- colnames(BayesMu) <- colnames(BayesLower) <- colnames(BayesUpper) <-
c('Standing','Walking','Running')

outputs <-
list(nsims=nsims,Positioneffect=Positioneffect,BayesMu=BayesMu,BayesLower=BayesLower,BayesUpper=BayesUpper)
save(outputs,file=paste('DirRegNetball ',gsub(':','_',date()),'.rdata',sep='')) # Save results for later use

arr <- c(1,2,6,7,3,4,5) # Change arrangement of positions in the graph below to make it look pretty
Labels <- gsub("(?<=\\b)([a-z])", "\\U\\1", tolower(Positions[arr]), perl=TRUE) # Make labels pretty
windows(10.5,4); par(mar=c(4,4,0.1,0.1)) # Set window size just right
plot(c(0.8,(nPositions+1.4)),c(0,0.68),type='n', xlab='Position',ylab='Proportion of time',xaxt='n') # Empty plot
with space for bars
axis(1,1:nPositions,Labels) # Nice position labels
for ( j in 1:Ndim) {
```

```
for ( i in 1:nPositions ) {
       arrows(i, BayesLower[arr[i],j], i, BayesUpper[arr[i],j],code=3,col=(5-j),lwd=3,length=0.4,angle=90) # Error
bars
       points(i, BayesMu[arr[i],j],col=(5-j),pch=4,cex=1) # Point estimates
}
}
legend('topright',c(colnames(BayesMu)),col=c(4,3,2),pch=c(4,4,4),lty=1,lwd=3)
```