Estimation of Models—Applied

Theodore Dounias

11/5/2018

Gibbs Sampler for the County Models

Simple County Mixed-Effects Model

```r
## Gibbs sampler in R
a.update <- function(){
  a.new <- rep (NA, J)
  for (j in 1:J){
    n.j <- sum (model_dt$county==cnt_vec[j])
    y.bar.j <- mean (model_dt$turnout[model_dt$county==cnt_vec[j]])
    a.hat.j <- ((n.j/sigma.y^2)*y.bar.j + (1/sigma.a^2)*mu.a)/
               (n.j/sigma.y^2 + 1/sigma.a^2)
    V.a.j <- 1/(n.j/sigma.y^2 + 1/sigma.a^2)
    a.new[j] <- rnorm (1, a.hat.j, sqrt(V.a.j))
  }
  return (a.new)
}
mu.a.update <- function(){
  mu.a.new <- rnorm (1, mean(a), sigma.a/sqrt(J))
  return (mu.a.new)
}
sigma.y.update <- function(){
  sigma.y.new <- sqrt(sum((model_dt$turnout-a[model_dt$county])^2)/rchisq(1,703))
  return (sigma.y.new)
}
sigma.a.update <- function(){
  sigma.a.new <- sqrt(sum((a-mu.a)^2)/rchisq(1,J-1))
  return (sigma.a.new)
}

J <- 64
n.chains <- 3
n.iter <- 1000
sims <- array (NA, c(n.iter, n.chains, J+3))
dimnames (sims) <- list (NULL, NULL, c (paste ("a[", 1:J, "]", sep=""), "mu.a",
   "sigma.y", "sigma.a"))

for (m in 1:n.chains){
  mu.a <- rnorm (1, mean(model_dt$turnout), sd(model_dt$turnout))
  sigma.y <- runif (1, 0, sd(model_dt$turnout))
  sigma.a <- runif (1, 0, sd(model_dt$turnout))
  for (t in 1:n.iter){
    a <- a.update ()
    mu.a <- mu.a.update ()
    sigma.y <- sigma.y.update ()
    sigma.a <- sigma.a.update ()
    sims[t,m,] <- c (a, mu.a, sigma.y, sigma.a)
  }
}

md_1 <- lmer(data = model_dt, turnout ~ 1 + (1|county))

arm::display(md_1)
```

```
## lmer(formula = turnout ~ 1 + (1 | county), data = model_dt)
## coef.est  coef.se
##     0.47     0.01
##
## Error terms:
##  Groups   Name        Std.Dev.
##  county   (Intercept) 0.04
##  Residual             0.20
## ---
## number of obs: 704, groups: county, 64
## AIC = -236.1, DIC = -257.2
## deviance = -249.7
```

```
#Get Values after convergance
vals <- data.frame(rep(0,3))
for(i in 1:67){
  for(j in 1:3){
    vals[j,i] <- mean(sims[-c(1:500),j,i])
  }
}

vals <- vals %>%
  summarise_all(mean)

names(vals) <- c (paste ("a[", 1:J, "]", sep=""), "mu.a",
   "sigma.y", "sigma.a")

vals[1,65:67]
```

```
##        mu.a   sigma.y    sigma.a
## 1 0.4689634 0.1999873 0.03922495
```

Note that these values are approximately the same as the ones outputed by the model!

```
# ## Gibbs sampler for a multilevel model w/ predictors
# a.update <- function(){
#   y.temp <- y - X%*%b - U[county]%*%g
#   eta.new <- rep (NA, J)
#   for (j in 1:J){
#     n.j <- sum (county==j)
#     y.bar.j <- mean (y.temp[county==j])
#     eta.hat.j <- ((n.j/sigma.y^2)*y.bar.j/
#                   (n.j/sigma.y^2 + 1/sigma.a^2))
#     V.eta.j <- 1/(n.j/sigma.y^2 + 1/sigma.a^2)
#     eta.new[j] <- rnorm (1, eta.hat.j, sqrt(V.eta.j))
#   }
#   a.new <- U%*%g + eta.new
#   return (a.new)
# }
# b.update <- function(){
#   y.temp <- y - a[county]
#   lm.0 <- lm (y.temp ~ X)
#   b.new <- sim (lm.0, n.sims=1)
#   return (b.new)
# }
# g.update <- function(){
#   lm.0 <- lm (a ~ U)
#   g.new <- sim (lm.0, n.sims=1)
#   return (g.new)
# }
# sigma.y.update <- function(){
#   sigma.y.new <- sqrt(sum((y-a[county]-X%*%b)^2)/rchisq(1,n-1))
#   return (sigma.y.new)
# }
# sigma.a.update <- function(){
#   sigma.a.new <- sqrt(sum((a-U%*%g)^2)/rchisq(1,J-1))
#   return (sigma.a.new)
# }
```

A First Pass at a Simple Logistic Model