Estimation of Models—Applied

Theodore Dounias

11/5/2018

Gibbs Sampler for the County Models

Simple County Mixed-Effects Model

```
## 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
```

```
##        a[1]      a[2]      a[3]      a[4]      a[5]      a[6]     a[7]
## 1 0.4419399 0.4652145 0.4543922 0.4572707 0.4948953 0.4730623 0.455343
##        a[8]      a[9]     a[10]     a[11]     a[12]     a[13]     a[14]
## 1 0.4630414 0.4830031 0.501924 0.4599679 0.4763479 0.4804906 0.4813186
##       a[15]     a[16]     a[17]     a[18]     a[19]     a[20]     a[21]
## 1 0.5034081 0.4710746 0.4380663 0.4740476 0.4638234 0.4431791 0.4491705
##       a[22]     a[23]     a[24]     a[25]     a[26]     a[27]     a[28]
## 1 0.4766929 0.4632769 0.4527275 0.4497076 0.4649351 0.4497059 0.5178371
##       a[29]     a[30]     a[31]     a[32]     a[33]     a[34]     a[35]
## 1 0.4844453 0.4934728 0.4630395 0.5043725 0.4803696 0.4473393 0.4474057
##       a[36]     a[37]     a[38]     a[39]     a[40]     a[41]     a[42]
## 1 0.4595093 0.4671027 0.4870102 0.4792172 0.4521485 0.5051364 0.4539349
##       a[43]     a[44]     a[45]     a[46]     a[47]    a[48]     a[49]
## 1 0.4519916 0.4698837 0.4634933 0.4641122 0.4713283 0.470712 0.4736597
##       a[50]     a[51]     a[52]     a[53]     a[54]     a[55]     a[56]
## 1 0.4465473 0.4607029 0.4538048 0.4858565 0.4662366 0.4544055 0.4595166
##       a[57]     a[58]     a[59]     a[60]     a[61]     a[62]     a[63]
## 1 0.4620142 0.4455312 0.5007491 0.4231033 0.4582347 0.5059223 0.4499308
```

```
##      a[64]      mu.a   sigma.y    sigma.a
## 1 0.480756 0.4679076 0.2004358 0.03637585
```

```r
# ## 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