Code Associated with: Hierarchical Bayesian models for small area estimation of county-level private forest landowner population

Neil R. Ver Planck, Andrew O. Finley, Emily S. Huff

August 1, 2017

Introduction

This `R Markdown` file gives a reproducible example for a single iteration of the small area estimation (SAE) models described in *Hierarchical Bayesian models for small area estimation of county-level private forest landowner population* for Montana. The code can easily be modified and rerun for the example iteration of New Jersey by assigning `NJ` to the object `state`.

The `MCMCpack` and `coda` packages will need to be loaded in order to access the inverse gamma distribution and summarize the Markov chain Monte Carlo (MCMC) posterior samples. We will also need to **source** the SAE model `R` scripts for the Fay-Herriot (`FH.R`) and Fay-Herriot conditional autogressive (`FHCAR.R`) models.

The data image (`data.RData`) loads the following objects:

- `dat`: data frame with columns for
  - `STATE`: character, `MT` for Montana and `NJ` for New Jersey
  - `COUNTYNUMB`: integer, county identification number
  - `NAME`: character, county name
  - `iterate`: integer, iteration number of the 4000 simulation runs
  - `n`: numeric, county sample size
  - `Y.i`: numeric, county-level population direct estimate
  - `sigma.sq.i`: numeric, sampling error of direct estimate
- `R.mt`: matrix, neighborhood for Montana lattice
- `R.nj`: matix, neighborhood for New Jersey lattice
- `X`: data frame with columns for
  - `STATE`: as defined for `dat`
  - `COUNTYNUMB`: as defined for `dat`
  - `POPDEN2010`: numeric, 2010 decennial Census population density (people / km^2)
  - `Forest.ha`: numeric, total forest area in hectares
- `Ytrue`: data frame with columns for
  - `STATE`: as defined for `dat`
  - `COUNTYNUMB`: as defined for `dat`
  - `Y.T`: numeric, GIS derived true private forest ownerships

```
library(MCMCpack)  ##function for inverse gamma distribution
library(coda)

# source the two SAE models
source("FH.R")
source("FHCAR.R")

# load the data from the R image
load("data.RData")
```

Constants

Each state is analyzed separately, so the first step is to choose the state to be analyzed either `MT` (for Montana) or `NJ` (for New Jersey) and the respective neighborhood matrix (i.e., `R.mt` or `R.nj`) defined as `R`. Three MCMC chains are run for each model with a length (`G`) of 3100 and burn-in period (`B`) of 100. The object `sub` is then created to extract post burn-in samples.

```
# Choose the state to analyze (i.e., MT or NJ)
state <- "MT"
#state <- "NJ"

# Select the correct neighborhood matrix (i.e., R.mt or R.nj)
R <- R.mt
#R <- R.nj

# length of each chain
G <- 3100
# burn-in
B <- 100

# post burn-in samples were not thinned
sub <- seq(B+1, G, by = 1)

# set the random seed for reproducibility
set.seed(19)
```

Distributions and additional functions

The multivariate normal distribution is used for drawing from the respective parameters in the SAE model `R` scripts. The function `quant` is defined for summarizing the 95% credible intervals for parameters of interest.

```
# Multivariate Normal Distribution
rmvn <- function(n, mu=0, V = matrix(1)){
  p <- length(mu)
  if(any(is.na(match(dim(V),p))))
    stop("Dimension problem!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)%*%D + rep(mu,rep(n,p)))
}

# Median; 95% Credible Interval
quant <- function(x){quantile(x, prob=c(0.5, 0.025, 0.975))}
```

Subset Data for a Single State

We now subset the `dat`, `Ytrue`, and `X` data frames for the state of interest. The object `m` is also defined for the number of counties for the state of interest. The following additional objects are also created for running the SAE models:

- `logy`: the log transformed direct estimates
- `tilde.sigma.sq`: the transformed sampling errors by the delta method
- `lm.beta`: linear model with `POPDEN2010` and `Forest.ha` covariates
- `beta.start`: beta starting values for the first chain based on `lm.beta`

```r
dat.sub <- dat[dat$STATE == state,]

Ytrue <- Ytrue[Ytrue$STATE == state,]

# number of counties
m <- nrow(dat.sub)

# Confirms the covariates are ordered sequentially by county number
X <- X[X$STATE == state, ]
X <- X[order(X$COUNTYNUMB, decreasing = FALSE),]
X <- as.matrix(X[,c("POPDEN2010", "Forest.ha")], nrow = m)
# Design Matrix
X <- cbind(1, X)

y <- as.matrix(dat.sub$Y.i, nrow = m)
logy <- log(y)
tilde.sigma.sq <- as.numeric(dat.sub$sigma.sq.i / y^2)

lm.beta <- lm(logy ~ -1 + X)
beta.start <- as.matrix(coef(lm.beta), nrow = 3)
```

SAE Models

We now run three chains for the FH SAE model with the following inputs:

- `y`: direct estimates (in this case, log transformed)
- `X`: design matrix of covariates
- `m`: number of small areas (i.e., counties)
- `G`: the length of the chain
- `beta`: starting values for β
- `a0`: prior shape parameter for inverse-gamma distribution
- `b0`: prior scale parameter for inverse-gamma distribution
- `sigma.sq.v`: starting value for random effect variance

```r
# Runs three chains for the Fay-Herriot (FH) SAE model
chain1 <- FH.model(y = logy, X = X, m = m,
                   sigma.sq = tilde.sigma.sq, G = G,
                   beta = beta.start,
                   a0 = 2, b0 = mean(tilde.sigma.sq),
                   sigma.sq.v = 1)

chain2 <- FH.model(y = logy, X = X, m = m,
                   sigma.sq = tilde.sigma.sq, G = G,
                   beta = as.matrix(c(1, -1, 1), nrow = 3),
                   a0 = 2, b0 = mean(tilde.sigma.sq),
                   sigma.sq.v = 0.1)

chain3 <- FH.model(y = logy, X = X, m = m,
                   sigma.sq = tilde.sigma.sq, G = G,
                   beta = as.matrix(c(10, -0.1, 0.1), nrow = 3),
                   a0 = 2, b0 = mean(tilde.sigma.sq),
                   sigma.sq.v = 0.5)
```

```
# a list combining the three chains
FH.chains <- list(chain1, chain2, chain3)
```

We now run three chains for the FHCAR SAE model with similar inputs to the FH model with the addition of the starting value for the autocorrelation parameter (`lambda`) and the neighborhood matrix (`R`).

```
# Runs three chains for the FHCAR SAE model
chain1 <- FHCAR.model(y = logy, X = X, m = m,
                      sigma.sq = tilde.sigma.sq, G = G, lambda = 0.5,
                      beta = beta.start,
                      a0 = 2, b0 = mean(tilde.sigma.sq),
                      sigma.sq.v = 1, R = R)

chain2 <- FHCAR.model(y = logy, X = X, m = m,
                      sigma.sq = tilde.sigma.sq, G = G, lambda = 0.8,
                      beta = as.matrix(c(1,-1,1), nrow = 3),
                      a0 = 2, b0 = mean(tilde.sigma.sq),
                      sigma.sq.v = 0.1, R = R)

chain3 <- FHCAR.model(y = logy, X = X, m = m,
                      sigma.sq = tilde.sigma.sq, G = G, lambda = 0.1,
                      beta = as.matrix(c(10,-0.1,0.1), nrow = 3),
                      a0 = 2, b0 = mean(tilde.sigma.sq),
                      sigma.sq.v = 0.5, R = R)

# a list combining the three chains
FHCAR.chains <- list(chain1, chain2, chain3)
```

We can additionally output the chains for separate model summaries or continue with the summaries that follow.

```
# save output for summarizing the models
save(list = c("FH.chains","FHCAR.chains"),
     file = paste(state,"-chains.RData", sep = ''))
```

Fay-Herriot (FH) Model Summary

Re-create objects `chain1`, `chain2`, `chain3` for the FH model.

```
# three chains of FH model
chain1 <- FH.chains[[1]]
chain2 <- FH.chains[[2]]
chain3 <- FH.chains[[3]]
```

The following creates the post burn-in samples for SAE parameters θ_i (`tsamps`), β (`bsamps`), σ_v^2 (`ssamps`), and the weighting between direct estimates and linking model (`gsamps`).

```
# MCMC samples
# Chain 1
theta.mat <- chain1[[1]]
beta.mat <- chain1[[2]]
sigma.sq.v.vec <- chain1[[3]]
g.mat <- chain1[[4]]

# Chain 2
theta.mat2 <- chain2[[1]]
```

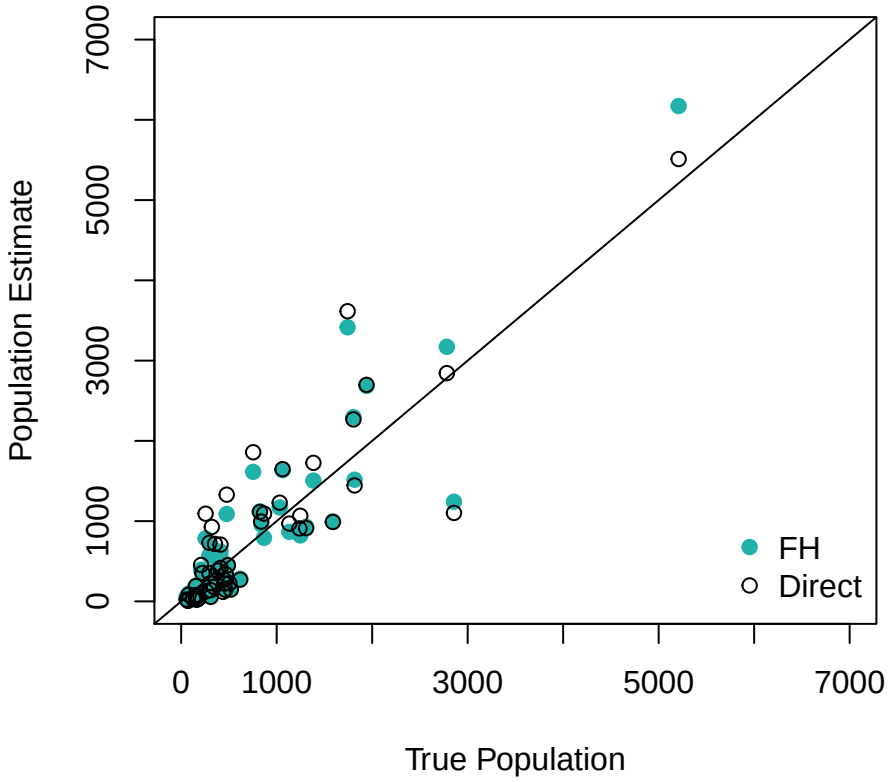
```
beta.mat2 <- chain2[[2]]
sigma.sq.v.vec2 <- chain2[[3]]
g.mat2 <- chain2[[4]]

# Chain 3
theta.mat3 <- chain3[[1]]
beta.mat3 <- chain3[[2]]
sigma.sq.v.vec3 <- chain3[[3]]
g.mat3 <- chain3[[4]]

# mcmc objects
tsamps.list <- mcmc.list(mcmc(t(theta.mat[,sub])), mcmc(t(theta.mat2[,sub])),
                         mcmc(t(theta.mat3[,sub])))
bsamps.list <- mcmc.list(mcmc(t(beta.mat[,sub])), mcmc(t(beta.mat2[,sub])),
                         mcmc(t(beta.mat3[,sub])))
ssamps.list <- mcmc.list(mcmc(sigma.sq.v.vec[sub]), mcmc(sigma.sq.v.vec2[sub]),
                         mcmc(sigma.sq.v.vec3[sub]))
gsamps.list <- mcmc.list(mcmc(t(g.mat[,sub])), mcmc(t(g.mat2[,sub])),
                         mcmc(t(g.mat3[,sub])))

# Post burn-in samples
tsamps <- rbind(tsamps.list[[1]], tsamps.list[[2]], tsamps.list[[3]])
bsamps <- rbind(bsamps.list[[1]], bsamps.list[[2]], bsamps.list[[3]])
ssamps <- c(ssamps.list[[1]], ssamps.list[[2]], ssamps.list[[3]])
gsamps <- rbind(gsamps.list[[1]], gsamps.list[[2]], gsamps.list[[3]])
```

The posterior samples are now summarized for the posterior mean of θ_i (`theta.mean`), posterior median and 95% credible interval of θ_i (`theta.hat`); for the additional parameters of β, σ_v^2, and weighting between direct estimates and linking model. The SAE parameter θ_i at individual county and state-level, as noted in code comments, is assessed by the following metrics: bias, mean squared error, root mean squared error, 95% empirical coverage rate, and 95% credible interval width.

```
# Summarize posteriors
theta.mean <- apply(exp(tsamps), 2, mean)
theta.mean <- round(theta.mean, digits = 0)

theta.hat <- apply(exp(tsamps), 2, quant)

beta.mean <- apply(bsamps, 2, mean)
beta.hat <- apply(bsamps, 2, quant)

sigma.sq.v.mean <-  apply(as.matrix(ssamps), 2, mean)
sigma.sq.v.hat <- apply(as.matrix(ssamps), 2, quant)

gamma.hat <- apply(gsamps, 2, quant)

# bias
theta.bias <- round(theta.mean - Ytrue$Y.T, digits = 0)
y.bias <- round(dat.sub$Y.i - Ytrue$Y.T, digits = 0)

# Mean squared error (MSE) at state-level
theta.mse <- round(mean((theta.mean - Ytrue$Y.T)^2), digits = 0)
y.mse <- round(mean((dat.sub$Y.i - Ytrue$Y.T)^2), digits = 0)
```

```
# Root mean squared error (RMSE) at state-level
theta.rmse <- sqrt(theta.mse)
y.rmse <- sqrt(y.mse)

# coverage at state-level
state.cover <- sum(ifelse(Ytrue$Y.T >= theta.hat[2,] &
                          Ytrue$Y.T <= theta.hat[3,], 1, 0)) / m
state.cover <- round(100 * state.cover, digits = 1)

# 95% Credible interval widths for each county
theta.ciw <- round(theta.hat[3,] - theta.hat[2,], digits = 0)
```

The following figure assesses the accuracy of the direct estimates (black circles) and FH model estimates (light seagreen circles) against the GIS-derived true population.



Fay-Herriot Conditional Autoregressive (FHCAR) Model Summary

Overwrite the three chains of the FH model with the FHCAR chains.

```
# three chains of FH model
chain1 <- FHCAR.chains[[1]]
chain2 <- FHCAR.chains[[2]]
chain3 <- FHCAR.chains[[3]]
```

The following creates the post burn-in samples for SAE parameters θ_i (`tsamps`), β (`bsamps`), λ (`lsamps`), σ_v^2 (`ssamps`), and the Metropolis-Hastings acceptance (`csamps`).

```
# MCMC samples
# Chain 1
```

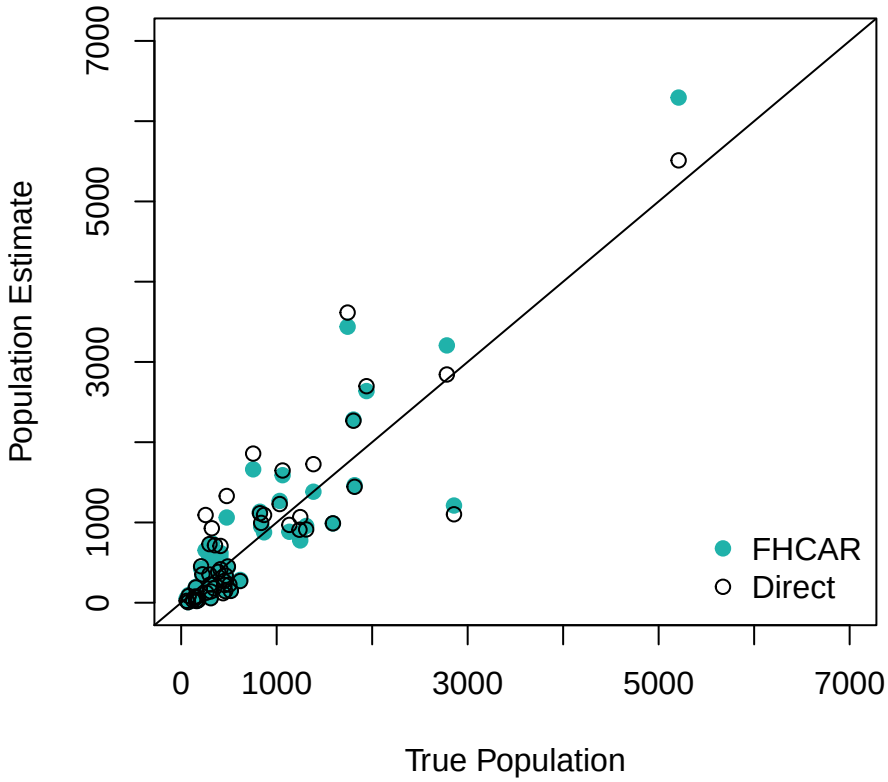
```
theta.mat <- chain1[[1]]
beta.mat <- chain1[[2]]
lambda.vec <- chain1[[3]]
sigma.sq.v.vec <- chain1[[4]]
count.vec <- chain1[[5]]

# Chain 2
theta.mat2 <- chain2[[1]]
beta.mat2 <- chain2[[2]]
lambda.vec2 <- chain2[[3]]
sigma.sq.v.vec2 <- chain2[[4]]
count.vec2 <- chain2[[5]]

# Chain 3
theta.mat3 <- chain3[[1]]
beta.mat3 <- chain3[[2]]
lambda.vec3 <- chain3[[3]]
sigma.sq.v.vec3 <- chain3[[4]]
count.vec3 <- chain3[[5]]

# mcmc objects
tsamps.list <- mcmc.list(mcmc(t(theta.mat[,sub])),
                         mcmc(t(theta.mat2[,sub])),
                         mcmc(t(theta.mat3[,sub])))
bsamps.list <- mcmc.list(mcmc(t(beta.mat[,sub])),
                         mcmc(t(beta.mat2[,sub])),
                         mcmc(t(beta.mat3[,sub])))
lsamps.list <- mcmc.list(mcmc(lambda.vec[sub]),
                         mcmc(lambda.vec2[sub]),
                         mcmc(lambda.vec3[sub]))
ssamps.list <- mcmc.list(mcmc(sigma.sq.v.vec[sub]),
                         mcmc(sigma.sq.v.vec2[sub]),
                         mcmc(sigma.sq.v.vec3[sub]))
count.list <- mcmc.list(mcmc(count.vec[sub]),
                        mcmc(count.vec2[sub]),
                        mcmc(count.vec3[sub]))

# Post burn-in samples
tsamps <- rbind(tsamps.list[[1]], tsamps.list[[2]], tsamps.list[[3]])
bsamps <- rbind(bsamps.list[[1]], bsamps.list[[2]], bsamps.list[[3]])
lsamps <- c(lsamps.list[[1]], lsamps.list[[2]], lsamps.list[[3]])
ssamps <- c(ssamps.list[[1]], ssamps.list[[2]], ssamps.list[[3]])
csamps <- c(count.list[[1]], count.list[[2]], count.list[[3]])
```

The posterior samples are now summarized for the posterior mean of θ_i (`theta.mean`), posterior median and 95% credible interval of θ_i (`theta.hat`); for the additional parameters of $\beta, \lambda, \sigma_v^2$, and Metropolis-Hastings acceptance (`count.hat`). The SAE parameter θ_i at individual county and state-level, as noted in code comments, is assessed by the following metrics: bias, mean squared error, root mean squared error, 95% empirical coverage rate, and 95% credible interval width.

```
# Summarize posteriors
theta.mean <- apply(exp(tsamps), 2, mean)
theta.mean <- round(theta.mean, digits = 0)
```

```
theta.hat <- apply(exp(tsamps), 2, quant)

beta.mean <- apply(bsamps, 2, mean)
beta.hat <- apply(bsamps, 2, quant)

lambda.mean <- apply(as.matrix(lsamps), 2, mean)
lambda.hat <- apply(as.matrix(lsamps), 2, quant)

sigma.sq.v.mean <-  apply(as.matrix(ssamps), 2, mean)
sigma.sq.v.hat <- apply(as.matrix(ssamps), 2, quant)

count.hat <- apply(as.matrix(csamps), 2, sum) / (3*length(sub))

# bias
theta.bias <- round(theta.mean - Ytrue$Y.T, digits = 0)
y.bias <- round(dat.sub$Y.i - Ytrue$Y.T, digits = 0)

# Mean squared error (MSE) at state-level
theta.mse <- round(mean((theta.mean - Ytrue$Y.T)^2), digits = 0)
y.mse <- round(mean((dat.sub$Y.i - Ytrue$Y.T)^2), digits = 0)

# Root mean squared error (RMSE) at state-level
theta.rmse <- sqrt(theta.mse)
y.rmse <- sqrt(y.mse)

# coverage at state-level
state.cover <- sum(ifelse(Ytrue$Y.T >= theta.hat[2,] &
                            Ytrue$Y.T <= theta.hat[3,], 1, 0)) / m
state.cover <- round(100 * state.cover, digits = 1)

# 95% Credible interval widths for each county
theta.ciw <- round(theta.hat[3,] - theta.hat[2,], digits = 0)
```

The following figure assesses the accuracy of the direct estimates (black circles) and FHCAR model estimates (light seagreen circles) against the GIS-derived true population.


Population Estimate
0
1000
3000
5000
7000
True Population
0
1000
3000
5000
7000
FHCAR
Direct