Assignment 3

Seungjun Lee

November 11, 2022

1. Load the file munichrents.RData and look at head(rents). The variables are

   - RentPerM2: Monthly rent per square meter in Euros
   - Year: Year the apartment was built
   - Location: District index
   - NoHotWater: Indicator variable for no hot water
   - NoCentralHeat: Indicator variable for no central heat
   - NoBathTiles: Indicator variable for no tiles in the bath
   - SpecialBathroom: Indicator variable for a special bathroom
   - SpecialKitchen: Indicator variable for a special kitchen
   - Room1-Room6: Indicator variables for corresponding number of rooms

   Fit a linear model relating rent per square meter to the covariates using least squares, and extract the coefficient estimates. You can ignore the Location variable for now since we will later treat this as a spatial random effect. Also note that the room indicator variables include one that is redundant, so treat a single room as the baseline (i.e., leave Room1 out of the model, so the intercept corresponds to a single room and coefficients for the others represent adjustments to the intercept for a different number of rooms).

```
> #head(rents)
> lm_rents = lm(RentPerM2 ~ . -Location - Room1,data=rents)
>
> # coeff
> lm_rents

Call:
lm(formula = RentPerM2 ~ . - Location - Room1, data = rents)

Coefficients:
(Intercept)             Year        NoHotWater    NoCentralHeat
  -16.01233          0.01333          -1.87051         -1.22576
NoBathTiles  SpecialBathroom    SpecialKitchen            Room2
   -0.72571          0.66037           1.46289         -1.31937
      Room3            Room4             Room5            Room6
   -1.88641         -2.46463          -2.37816         -2.48262
```

2. There are two SpatialPolygons objects associated with this dataset, districts.sp and parks.sp. The first corresponds to city districts in which apartments may be located. The second corresponds to districts with no possible apartments, such as parks or fields. Create an nb object with neighbors for the districts, defining neighbors as districts that share a common boundary. Make a plot showing the districts, then add the parks shaded a different color.

```
> nb_dist = poly2nb(districts.sp)
> coords = coordinates(districts.sp)
> png("problem2.png", width = 1000, height = 800)
> plot(districts.sp, border = "gray")
> plot(nb_dist, coords, pch = 19, cex = 0.6, add = TRUE)
> plot(parks.sp,add=TRUE,col="skyblue")
> legend("bottomleft", fill = c("skyblue"),
+        legend = c("No apartments"),
+        bty="n", cex = 1.5, y.intersp = 1.5)
> dev.off()
RStudioGD
        2
```

Figure 1: Map of city districts in which apartment may be located.

3. There are 380 districts in districts.sp, and the corresponding district numbers are indicated by the Location variable in rents. I've included a matrix H that provides a mapping between the districts as they're ordered in districts.sp and as they appear in the rents dataframe. Use H to create a new vector containing the number of observations in each district, and make a color or grayscale plot to illustrate this.

```
> dist_obs = colSums(H)
> length(dist_obs)
[1] 380
> range(dist_obs)
[1]  0 34
```

```
>
> png("problem3.png", width = 1000, height = 800)
>
> pal <- rev(grey(seq(0,1,length=6))[-1])
> q5 <- classIntervals(dist_obs, n = 5, style = "quantile")
> col <- findColours(q5, pal)
> plot(districts.sp, col = col)
> legend("bottomleft", fill = c(attr(col, "palette"),"skyblue"),
+        legend = c(names(attr(col, "table")),"No apartments"),
+        bty="n", cex = 1.5, y.intersp = 1.5)
> plot(parks.sp,add=TRUE,col="skyblue")
> dev.off()
RStudioGD
        2
```

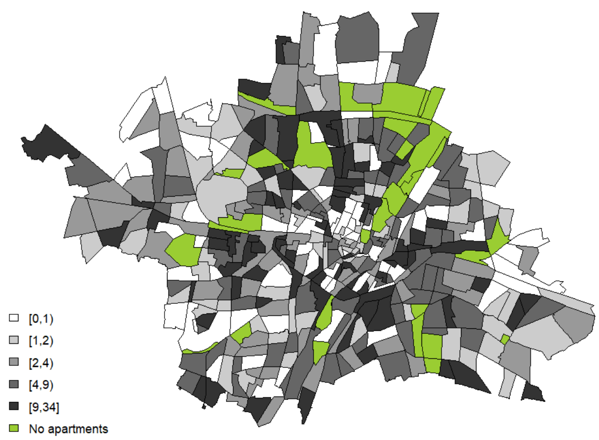

Figure 2: Map of city districts with the number of observations in each district.

4. We will now create a Gibbs sampler to sample from the posterior distribution under the following Bayesian model. Let X be the matrix of covariates, including the intercept term. Let n be the number of data points in Y and m be the number of spatial locations in η.
Data model:

$$Y|\beta, \eta, \sigma^2 \sim MVN(X\beta + H\eta, \sigma^2, I)$$

Process model:

$$p(\eta|\tau^2) \propto (\tau^2)^{-(m-1)/2} exp\left\{-\frac{1}{2\tau^2}\eta^{'}(D_w - W)\eta\right\}$$

where W is the matrix of 0 and 1 indicating the neighborhood structure from problem 2, and D_w is a diagonal matrix with diagonal entries $\Sigma_j W_{1j}, \cdots, \Sigma_j W_{nj}$. That is η follows an (improper) intrinsic autoregressive model.

Prior model: Specify independent priors for β, σ^2, and τ^2 with

$$p(\beta) \propto 1\sigma^2, \tau^2 \sim InverseGamma(0.001, 0.001)$$

The full conditional distributions for β, η, σ^2, and τ^2 are given at the end of this assignment. Construct a Gibbs sampler that cycles through each of the full conditionals and stores the results for B = 10, 000 iterations

Turn in the following:

- Your map with the neighbors from problem 2
- Your map of the apartment counts for each district
- Trace plots and ACF plots for σ^2 and τ^2
- A table with posterior means of the β and 95% credible intervals constructed using the 0.025 and 0.975 quantiles of the posterior samples
- A color or grayscale map of the posterior means for the vector η
- A color or grayscale map of the posterior standard deviations for the vector η

```
> # weight matrix
> W = nb2mat(nb_dist,style="B")
> Dw = diag(nrow(W))
> diag(Dw) = colSums(W)
>
> B = 1e+4 # iterations
> p = ncol(X)
> m = ncol(H)
>
> # placeholder
> beta = matrix(NA, B, p)
> eta = matrix(NA, B, m)
> sigma2 = matrix(NA, B, 1)
> tau2 = matrix(NA, B, 1)
>
>
> # init
> beta[1,] = lm_rents$coeff
> eta[1, ] = 1
> sigma2[1, ] = 1
> tau2[1, ] = 1
>
> # Gibbs sampler
> set.seed(1110)

> for(i in 2:B){
+     start = Sys.time()
+     inv_XtX = solve(t(X)%*%X)
+     beta[i, ] = rmvnorm(1, mu=inv_XtX%*%t(X)%*%(y-H%*%eta[i-1,]), Sigma=sigma2[i-1,]*inv_XtX)
```

```
+
+     inv_HtH = solve(t(H)%*%H/sigma2[i-1,] + (Dw - W)/tau2[i-1,])
+     eta[i, ] = rmvnorm(1, mu = inv_HtH%*%t(H)%*%(y-X%*%beta[i,])/sigma2[i-1,], Sigma=inv_HtH)
+     eta[i, ] = eta[i,] - mean(eta[i,]) # impose the constraint
+
+     sigma2[i, ] = rinvgamma(1,0.001 + n/2, 0.001 +
                            t(y - X%*%beta[i,] - H%*%eta[i,])%*%(y - X%*%beta[i,] - H%*%eta[i,])/2
+     tau2[i, ] = rinvgamma(1,0.001 + (m-1)/2, 0.001 + t(eta[i, ])%*%(Dw- W)%*%eta[i, ]/2)
+     if(i ==2) print(paste("ETS: ",round(B*as.vector(Sys.time()-start)/60,3),"mins" ))
+     if(i %% 100 == 0){print(i)}
+ }

> save(beta, eta, tau2, sigma2, file = "Gibbs_real_samples.RData")
> load("Gibbs_real_samples.RData")
```

- Trace plots and ACF plots for σ^2 and τ^2 : Both seemed to have converged.

```
> # ACF and trace plot of sigma2 and tau2
> png("problem4_tau_ACF.png", width = 1200, height = 600)
> par(mfrow=c(1,2))
> plot(tau2, type="l", xlab = "iteration", ylab = "tau2", main = "Traceplot of tau2 ")
> acf(tau2, xlab = "Lag", ylab = "ACF", main = "ACF plot of tau2 ")
> dev.off()
RStudioGD
    2
>
> png("problem4_sigma_ACF.png", width = 1200, height = 600)
> par(mfrow=c(1,2))
> plot(sigma2, type="l", xlab = "iteration", ylab = "sigma2", main = "Traceplot of sigma2 ")
> acf(sigma2, xlab = "Lag", ylab = "ACF", main = "ACF plot of sigma2 ")
> dev.off()
RStudioGD
    2
```

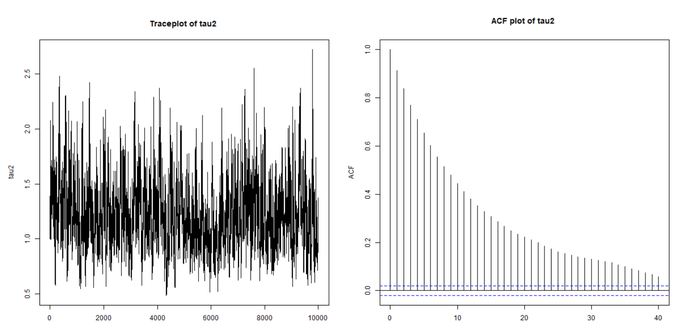

Figure 3: Trace plot and ACF plot of τ^2

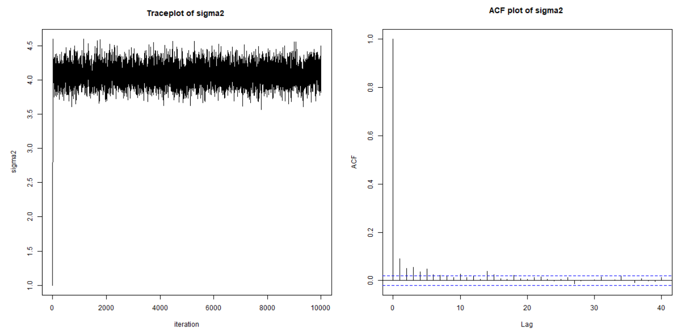

Figure 4: Trace plot and ACF plot of σ^2

- A table with posterior means of the β and 95% credible intervals constructed using the 0.025 and 0.975 quantiles of the posterior samples

```
> pmeans = matrix(colMeans(beta),nrow=1)
> colnames(pmeans) = paste0("beta",1:12)
> pmeans      # posterior means for beta
        beta1     beta2    beta3     beta4      beta5     beta6
[1,] -25.50135 0.0181245 -1.89307 -1.324123 -0.6468978 0.6216116
        beta7     beta8     beta9    beta10    beta11    beta12
[1,] 1.364515 -1.243109 -1.757587 -2.276017 -2.227121 -2.479833
>
> CI = apply(beta,2,function(x) quantile(x,probs=c(0.025,0.975)))
> colnames(CI) = paste0("beta",1:12)
> CI  # Credible Intervals for the beta
          beta1      beta2     beta3      beta4      beta5     beta6
2.5%  -34.01547 0.01378534 -2.458646 -1.7181142 -0.8830718 0.2952034
97.5% -16.98751 0.02243156 -1.325675 -0.9352396 -0.4075280 0.9492704
         beta7      beta8    beta9    beta10    beta11    beta12
2.5%  1.016511 -1.5431271 -2.05675 -2.643650 -2.886512 -3.606987
97.5% 1.719211 -0.9423278 -1.46141 -1.909155 -1.589295 -1.357552
```

- A color or grayscale map of the posterior means for the vector η

```
> # map of posterior mean of eta
> peta = colMeans(eta)
> length(peta)
[1] 380
> range(peta)
[1] -2.249682  1.179476
>
> pal <- rev(grey(seq(0,1,length=6))[-1])
> q5 <- classIntervals(peta, n = 5, style = "quantile")
> col <- findColours(q5, pal)
>
> png("problem4_post.png", width = 1000, height = 800)
>
> plot(districts.sp, col = col)
> legend("bottomleft", fill = c(attr(col, "palette"),"skyblue"),
+        legend = c(names(attr(col, "table")),"No apartments"),
```

```
+         bty="n", cex = 1.5, y.intersp = 1.5)
> plot(parks.sp,add=TRUE,col="skyblue")
> dev.off()
RStudioGD
        2
```

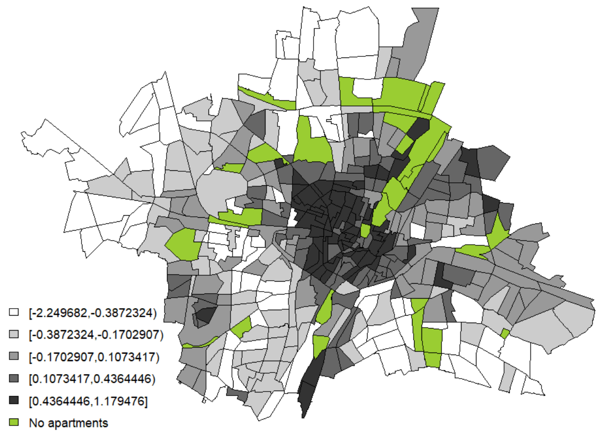

Figure 5: Gray scale map of the posterior mean for the η

- A color or grayscale map of the posterior standard deviations for the vector η

Figure 6: Gray scale map of the posterior standard deviation for the η

1 Appendix : All codes

```
setwd("C:/Users/user/desktop/jun/SpatioTemp/hw3")
load("munichrents.Rdata")

library(sp)
library(spdep)
library(classInt)
library(fields)
library(MCMCpack)

#------------------------------------------------------------#
# 1
#head(rents)
lm_rents = lm(RentPerM2 ~ . -Location - Room1,data=rents)

# coeff
lm_rents


X = as.matrix(cbind(1,rents[-c(1,3,9)]),nrow=2035)
solve(t(X)%*%X)%*%t(X)%*%as.matrix(rents[c(1)],nrow=2035)
#------------------------------------------------------------#
# 2
nb_dist = poly2nb(districts.sp)
coords = coordinates(districts.sp)
png("problem2.png", width = 1000, height = 800)
plot(districts.sp, border = "gray")
plot(nb_dist, coords, pch = 19, cex = 0.6, add = TRUE)
plot(parks.sp,add=TRUE,col="yellowgreen")
legend("bottomleft", fill = c("yellowgreen"),
       legend = c("No apartments"),
       bty="n", cex = 1.5, y.intersp = 1.5)
dev.off()

#------------------------------------------------------------#
# 3
dist_obs = colSums(H)
length(dist_obs)
range(dist_obs)

png("problem3.png", width = 1000, height = 800)

pal <- rev(grey(seq(0,1,length=6))[-1])
q5 <- classIntervals(dist_obs, n = 5, style = "quantile")
col <- findColours(q5, pal)
plot(districts.sp, col = col)
legend("bottomleft", fill = c(attr(col, "palette"),"yellowgreen"),
       legend = c(names(attr(col, "table")),"No apartments"),
       bty="n", cex = 1.5, y.intersp = 1.5)
plot(parks.sp,add=TRUE,col="yellowgreen")
dev.off()
```

```r
#----------------------------------------------------------#
# 4

# weight matrix
W = nb2mat(nb_dist,style="B")
Dw = diag(nrow(W))
diag(Dw) = colSums(W)

B = 1e+4 # iterations
p = ncol(X)
m = ncol(H)

# placeholder
beta = matrix(NA, B, p)
eta = matrix(NA, B, m)
sigma2 = matrix(NA, B, 1)
tau2 = matrix(NA, B, 1)


# init
beta[1,] = lm_rents$coeff
eta[1, ] = 1
sigma2[1, ] = 1
tau2[1, ] = 1

# Gibbs sampler
set.seed(1110)

for(i in 2:B){
  start = Sys.time()
  inv_XtX = solve(t(X)%*%X)
  beta[i, ] = rmvnorm(1, mu=inv_XtX%*%t(X)%*%(y-H%*%eta[i-1,]), Sigma=sigma2[i-1,]*inv_XtX)

  inv_HtH = solve(t(H)%*%H/sigma2[i-1,] + (Dw - W)/tau2[i-1,])
  eta[i, ] = rmvnorm(1, mu = inv_HtH%*%t(H)%*%(y-X%*%beta[i,])/sigma2[i-1,], Sigma=inv_HtH)
  eta[i, ] = eta[i,] - mean(eta[i,]) # impose the constraint

  sigma2[i, ] = rinvgamma(1,0.001 + n/2, 0.001 + t(y - X%*%beta[i,] - H%*%eta[i,])%*%(y - X%*%beta[i,] - H%*%e
  tau2[i, ] = rinvgamma(1,0.001 + (m-1)/2, 0.001 + t(eta[i, ])%*%(Dw- W)%*%eta[i, ]/2)
  if(i ==2) print(paste("ETS: ",round(B*as.vector(Sys.time()-start)/60,3),"mins" ))
  if(i %% 100 == 0){print(i)}
}

save(beta, eta, tau2, sigma2, file = "Gibbs_real_samples.RData")

load("Gibbs_real_samples.RData")

# ACF and trace plot of sigma2 and tau2
png("problem4_tau_ACF.png", width = 1200, height = 600)
par(mfrow=c(1,2))
plot(tau2, type="l", xlab = "iteration", ylab = "tau2", main = "Traceplot of tau2 ")
acf(tau2, xlab = "Lag", ylab = "ACF", main = "ACF plot of tau2 ")
```

```r
dev.off()

png("problem4_sigma_ACF.png", width = 1200, height = 600)
par(mfrow=c(1,2))
plot(sigma2, type="l", xlab = "iteration", ylab = "sigma2", main = "Traceplot of sigma2 ")
acf(sigma2, xlab = "Lag", ylab = "ACF", main = "ACF plot of sigma2 ")
dev.off()


# posterior mean and C.I

pmeans = matrix(colMeans(beta),nrow=1) # posterior means for beta
colnames(pmeans) = paste0("beta",1:12)
pmeans

CI = apply(beta,2,function(x) quantile(x,probs=c(0.025,0.975)))
colnames(CI) = paste0("beta",1:12)
CI

# map of posterior mean of eta
peta = colMeans(eta)
length(peta)
range(peta)

pal <- rev(grey(seq(0,1,length=6))[-1])
q5 <- classIntervals(peta, n = 5, style = "quantile")
col <- findColours(q5, pal)

png("problem4_post.png", width = 1000, height = 800)

plot(districts.sp, col = col)
legend("bottomleft", fill = c(attr(col, "palette"),"yellowgreen"),
       legend = c(names(attr(col, "table")),"No apartments"),
       bty="n", cex = 1.5, y.intersp = 1.5)
plot(parks.sp,add=TRUE,col="yellowgreen")
dev.off()


# map of standard deviation of eta

sd_eta = apply(eta,2,sd)
length(sd_eta)
range(sd_eta)

pal <- rev(grey(seq(0,1,length=6))[-1])
q5 <- classIntervals(sd_eta, n = 5, style = "quantile")
col <- findColours(q5, pal)

png("problem4_sd.png", width = 1000, height = 800)

plot(districts.sp, col = col)
legend("bottomleft", fill = c(attr(col, "palette"),"yellowgreen"),
       legend = c(names(attr(col, "table")),"No apartments"),
```

```
       bty="n", cex = 1.5, y.intersp = 1.5)
plot(parks.sp,add=TRUE,col="yellowgreen")
dev.off()
```