SpGPCW: Spatially Varying Gaussian Process Modeling for Critical Window Estimation

SpGPCW_Example

[1] Simulate data from the proposed model:

- Setting the reproducibility seed and initializing packages for data simulation:

```
set.seed(2365)

library(SpGPCW)
library(mnormt)  #Multivariate normal distribution
library(boot)  #Inverse logit transformation
library(spdep)  #Creating a grid
```

```
## Loading required package: sp
```

```
## Loading required package: Matrix
```

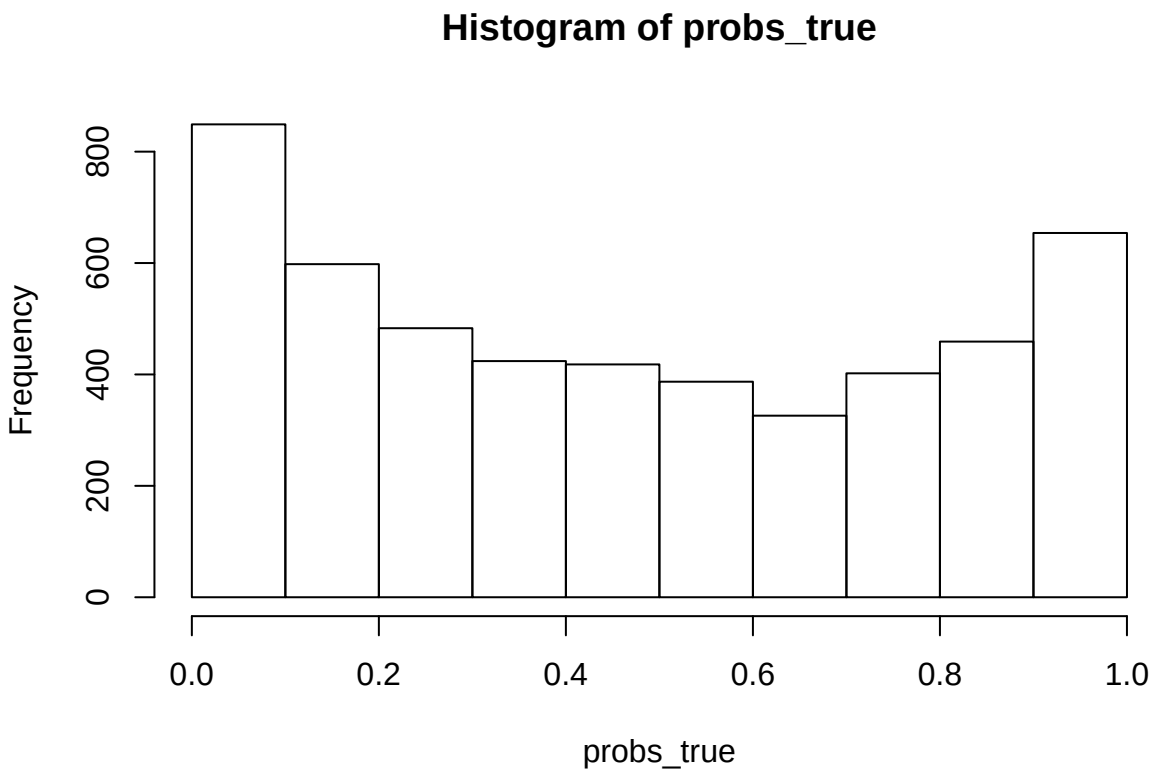
```
## Loading required package: spData
```

```
## To access larger datasets in this package, install the spDataLarge
## package with: `install.packages('spDataLarge',
## repos='https://nowosad.github.io/drat/', type='source'))`
```

- Setting the global data values:

```
n<-5000  #Sample size
m<-30  #Number of exposure time periods
g<-5  #Size of square spatial grid
s<-g^2  #Number of spatial locations

grid<-cell2nb(nrow=g,
              ncol=g,
              type="rook",
              torus=FALSE)  #Evenly spaced grid
neighbors<-nb2mat(grid,
                  zero.policy=TRUE,
                  style="B")  #Adjacency matrix
MCAR<-diag(rowSums(neighbors)) -
      neighbors
site_id<-rep(s, times=n)
for(j in 1:s){
   site_id[(1 + floor(n/s)*(j-1)):(floor(n/s)*j)]<-j
   }

z<-matrix(0, nrow=n, ncol=m)
for(j in 1:s){
   z[(site_id == j),]<-matrix(rnorm(n=sum(site_id == j)),
                              nrow=sum(site_id == j),
                              ncol=m,
                              byrow=TRUE)  #Exposure design matrices
   }
for(j in 1:m){
   z[,j]<-(z[,j] - median(z[,j]))/IQR(z[,j])  #Data standardization (interquartile range)
   }
```

```r
x<-matrix(1,
          nrow=n,
          ncol=2)  #Covariate design matrix
x[,2]<-rnorm(n)

beta_true<- c(-0.30, 0.50)
sigma2_theta_true<-0.75
sigma2_eta_true<-0.20
phi_true<-0.01
Sigma_theta_true<-sigma2_theta_true*chol2inv(chol(temporal_corr_fun(m, phi_true)[[1]]))
Sigma_eta_true<-sigma2_eta_true*chol2inv(chol(temporal_corr_fun(m, phi_true)[[1]]))
theta_true<-rmnorm(n=1,
                   mean=rep(0, times=m),
                   varcov=Sigma_theta_true)
theta_true<-theta_true -
            mean(theta_true)
rho_true<-0.85
eta_true<-rmnorm(n=1,
                 mean=rep(0, times=(m*s)),
                 varcov=chol2inv(chol(kronecker((rho_true*MCAR + (1 - rho_true)*diag(s)),
                                                chol2inv(chol(Sigma_eta_true))))))
eta_true<-eta_true -
          mean(eta_true)

logit_p_true<-rep(0, times=n)
for(j in 1:s){
   logit_p_true[site_id == j]<-x[(site_id == j),]%*%beta_true +
                               z[(site_id == j),]%*%theta_true +
                               z[(site_id == j),]%*%eta_true[(1 + (j-1)*m):(j*m)]
   }
probs_true<-inv.logit(logit_p_true)
hist(probs_true)
```



- Simulating the analysis dataset:

```
y<-rbinom(n=n,
          size=1,
          prob=probs_true)
```

[2] Fit SpGPCW to estimate spatially varying critical windows of susceptibility:

```
results<-SpGPCW(mcmc_samples = 10000,
                y = y, x = x, z = z, site_id = site_id, neighbors = neighbors,
                metrop_var_rho_trans = 1.00,
                metrop_var_phi_trans = 0.07)
```

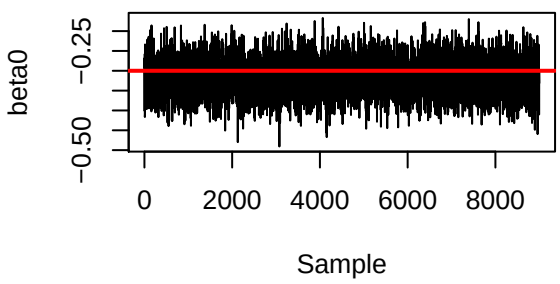
```
## Progress: 10%
## rho Acceptance: 31%
## phi Acceptance: 25%
## ******************
## Progress: 20%
## rho Acceptance: 30%
## phi Acceptance: 26%
## ******************
## Progress: 30%
## rho Acceptance: 30%
## phi Acceptance: 25%
## ******************
## Progress: 40%
## rho Acceptance: 29%
## phi Acceptance: 25%
```

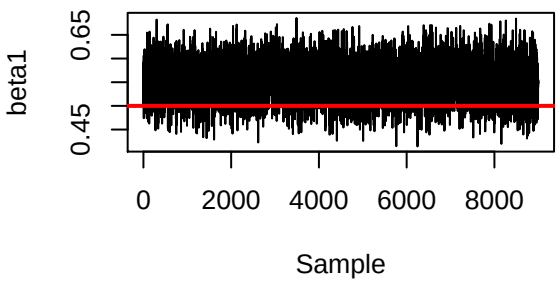
```
## ********************
## Progress: 50%
## rho Acceptance: 29%
## phi Acceptance: 25%
## ********************
## Progress: 60%
## rho Acceptance: 29%
## phi Acceptance: 25%
## ********************
## Progress: 70%
## rho Acceptance: 29%
## phi Acceptance: 25%
## ********************
## Progress: 80%
## rho Acceptance: 29%
## phi Acceptance: 25%
## ********************
## Progress: 90%
## rho Acceptance: 29%
## phi Acceptance: 25%
## ********************
## Progress: 100%
## rho Acceptance: 29%
## phi Acceptance: 25%
## ********************
```

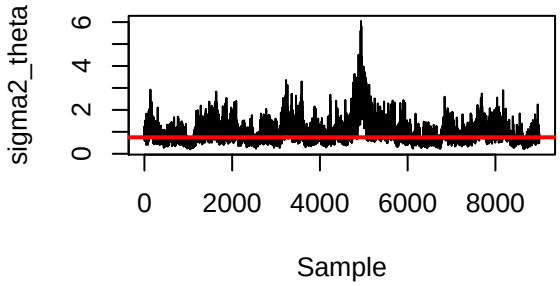
[3] Analyzing Output:

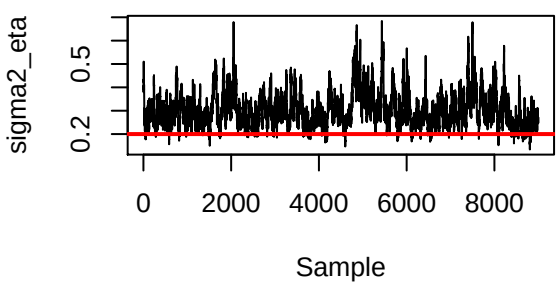
```
par(mfrow=c(2,2))
plot(results$beta[1, 1001:10000],
     type="l",
     ylab="beta0",
     xlab="Sample")
abline(h=beta_true[1],
       col="red",
       lwd=2)  #True value
plot(results$beta[2, 1001:10000],
     type="l",
     ylab="beta1",
     xlab="Sample")
abline(h=beta_true[2],
       col="red",
       lwd=2)  #True value
plot(results$sigma2_theta[1001:10000],
     type="l",
     ylab="sigma2_theta",
     xlab="Sample")
abline(h=sigma2_theta_true,
       col="red",
       lwd=2)  #True value
plot(results$sigma2_eta[1001:10000],
     type="l",
     ylab="sigma2_eta",
     xlab="Sample")
abline(h=sigma2_eta_true,
```

```
       col="red",
       lwd=2)  #True value
```










```
par(mfrow=c(2,2))
plot(results$rho[1001:10000],
     type="l",
     ylab="rho",
     xlab="Sample")
abline(h=rho_true,
       col="red",
       lwd=2)  #True value
plot(results$phi[1001:10000],
     type="l",
     ylab="phi",
     xlab="Sample")
abline(h=phi_true,
       col="red",
       lwd=2)  #True value
eta<-simplify2array(results$eta)
plot(rowMeans(results$theta[,1001:10000] +
              eta[1,,1001:10000]),
     pch=16,
     ylab="theta + eta1",
     xlab="Time",
     ylim=c(0.00, 2.00))
points(theta_true + eta_true[1:m],
```

```
       pch=16,
       col="red")  #True values
plot(rowMeans(results$theta[,1001:10000] +
              eta[8,,1001:10000]),
     pch=16,
     ylab="theta + eta8",
     xlab="Time",
     ylim=c(-0.65, 1.00))
points(theta_true + eta_true[(7*(m + 1)):(8*m)],
       pch=16,
       col="red")  #True values
```

```
## Warning in theta_true + eta_true[(7 * (m + 1)):(8 * m)]: longer object
## length is not a multiple of shorter object length
```