Spillover: Spatial Change Point Estimation Due to Spillover from a Point Source

Spillover_Example

[1] Simulate data from the proposed model:

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

```
set.seed(7541)

library(Spillover)
library(geoR)  #Spatial covariance functions
```

```
## Warning: package 'geoR' was built under R version 4.1.1
```

```
## --------------------------------------------------------------
##  Analysis of Geostatistical Data
##  For an Introduction to geoR go to http://www.leg.ufpr.br/geoR
##  geoR version 1.8-1 (built on 2020-02-08) is now loaded
## --------------------------------------------------------------
```

```
library(mnormt)  #Multivariate normal distribution
library(boot)  #Inverse logit transformation
```

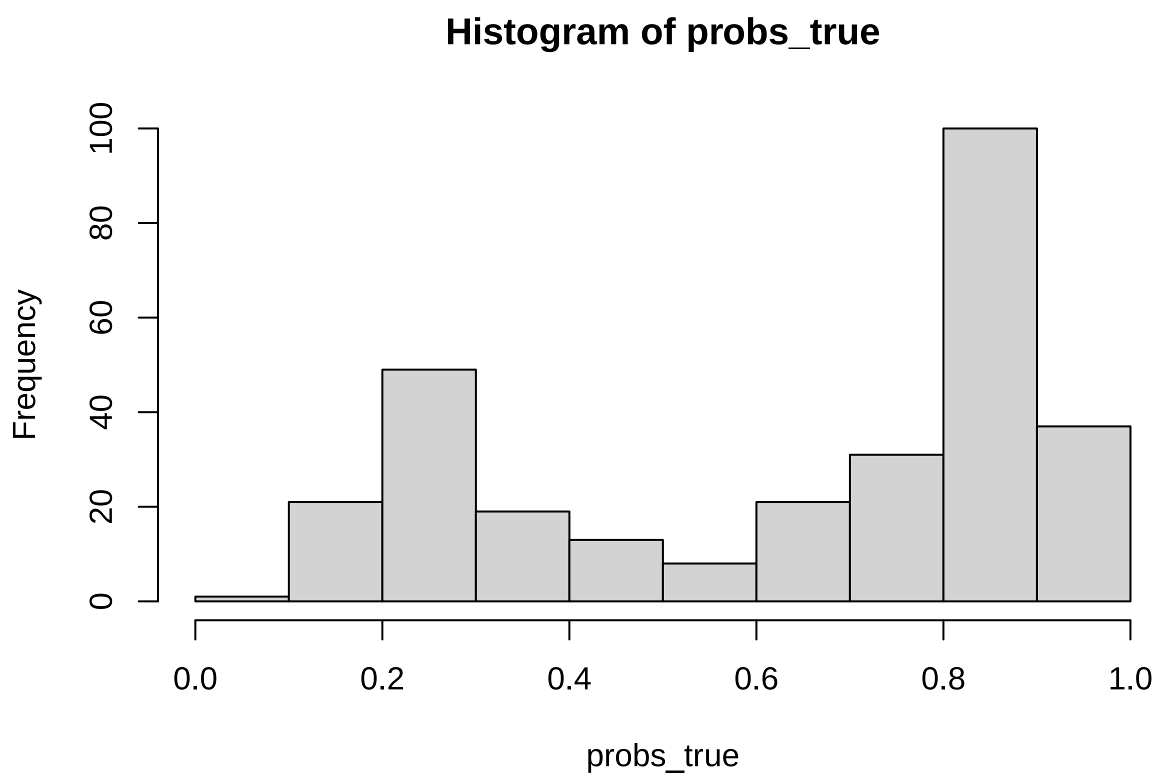
- Setting the global data values:

```
n<-300  #Sample size
m<-100  #Number of unique spatial locations

unique_locations<-matrix(runif(2*m),
                         nrow=m,
                         ncol=2)
ps_location<-unique_locations[1,]
spatial_dists<-as.matrix(dist(unique_locations,
                              diag=TRUE,
                              upper=TRUE))
z<-matrix(0, nrow=n, ncol=m)
distance_to_ps<-rep(0, times=n)
for(j in 1:n){
   loc<-sample(c(1:m),
               size=1)
   if(j <= round(0.10*n)){  #~10% located at the point source
     loc<-1
     }
   z[j, loc]<-1  #Spatial random effect design matrix
   distance_to_ps[j]<-spatial_dists[1, loc]
   }
x<-matrix(1, nrow=n, ncol=2)
x[,2]<-rnorm(n)  #Covariate design matrix
```

- Setting the values for the statistical model parameters:

```
beta_true<-c(-0.50, 0.30)
lambda_true<-2.00

theta_true<-0.50
x_full_true<-cbind(x,
```

```
                   as.numeric(distance_to_ps <= theta_true)*exp(-(distance_to_ps^2)))

phi_true<-0.70
spatial_corr_true<-spatial_corr<-cov.spatial(spatial_dists,
                                             cov.model="spherical",
                                             cov.pars=c(1, (1/phi_true)))
sigma2_w_true<-0.75
w_true<-c(rmnorm(n=1,
                 mean=rep(0, times=m),
                 varcov=(sigma2_w_true*spatial_corr_true)))
w_true<-w_true - mean(w_true)

logit_p_true<-x_full_true%*%c(beta_true, lambda_true) +
              z%*%w_true

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 Spillover to the Data:

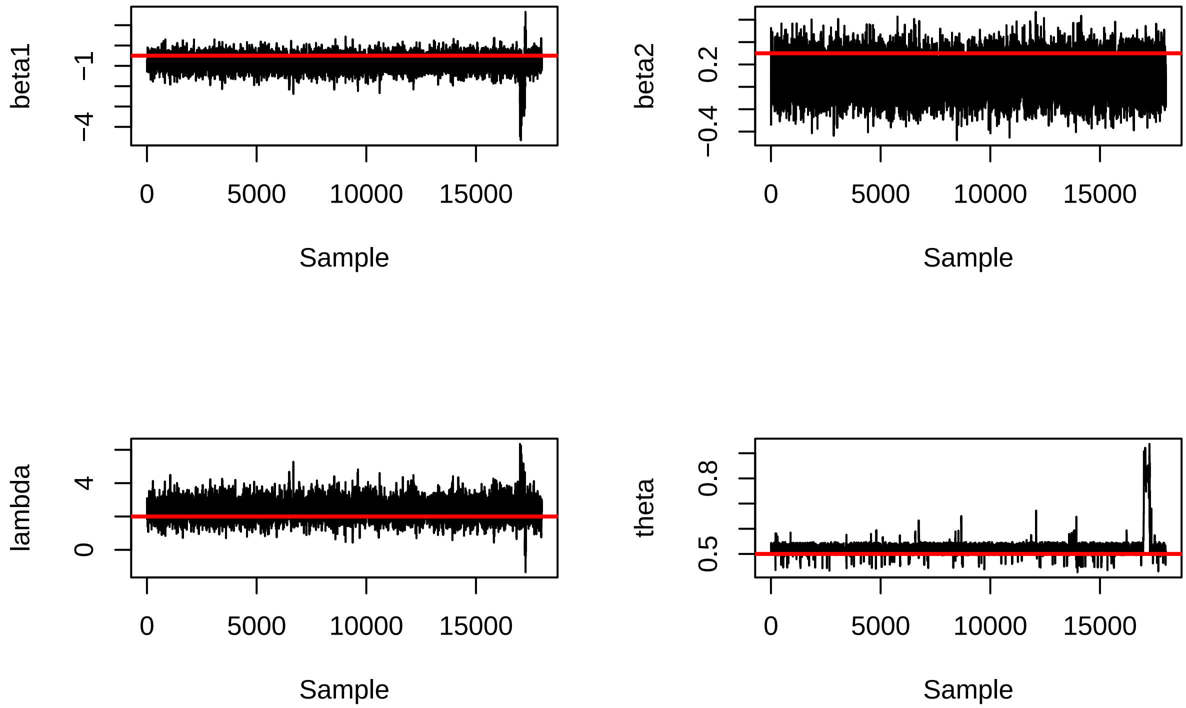
```r
results<-Spillover(mcmc_samples = 20000,
                   spillover_covar_def = 3, #1: Change point; 2: Exponential; 3: Gaussian
                   y = y,
                   x = x,
                   distance_to_ps = distance_to_ps,
                   z = z,
                   spatial_dists = spatial_dists,
                   metrop_var_phi_trans = 0.35,
                   metrop_var_theta_trans = 0.20)
```

```
## ********************
## Gaussian Spillover
## Progress: 10%
## phi Acceptance: 27%
## theta Acceptance: 22%
## ********************
## Gaussian Spillover
## Progress: 20%
## phi Acceptance: 26%
## theta Acceptance: 22%
## ********************
## Gaussian Spillover
## Progress: 30%
## phi Acceptance: 26%
## theta Acceptance: 21%
## ********************
## Gaussian Spillover
## Progress: 40%
## phi Acceptance: 26%
## theta Acceptance: 21%
## ********************
## Gaussian Spillover
## Progress: 50%
## phi Acceptance: 27%
## theta Acceptance: 21%
## ********************
## Gaussian Spillover
## Progress: 60%
## phi Acceptance: 26%
## theta Acceptance: 20%
## ********************
## Gaussian Spillover
## Progress: 70%
## phi Acceptance: 27%
## theta Acceptance: 20%
## ********************
## Gaussian Spillover
## Progress: 80%
## phi Acceptance: 27%
## theta Acceptance: 20%
## ********************
## Gaussian Spillover
## Progress: 90%
## phi Acceptance: 27%
```

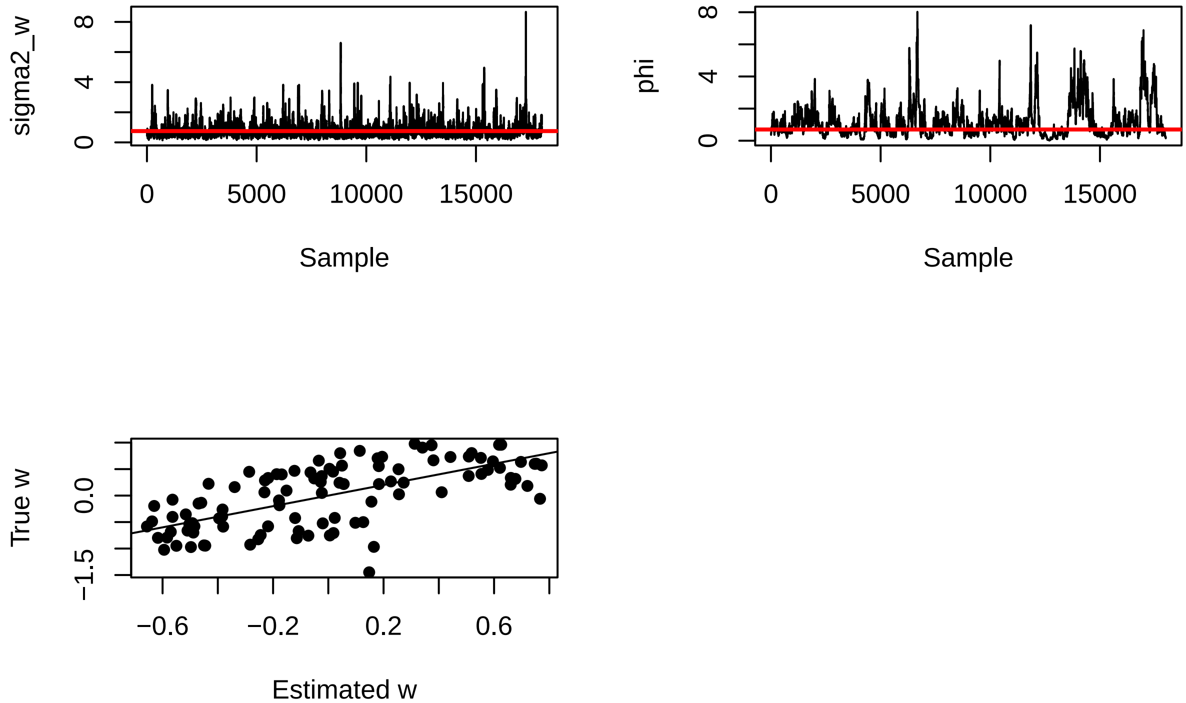
```
## theta Acceptance: 20%
## ********************
## Gaussian Spillover
## Progress: 100%
## phi Acceptance: 26%
## theta Acceptance: 21%
```

[3] Analyzing Output:

```
par(mfrow=c(2,2))
plot(results$beta[1, 2001:20000],
     type="l",
     ylab="beta1",
     xlab="Sample")
abline(h=beta_true[1],
       col="red",
       lwd=2)  #True value
plot(results$beta[2, 2001:20000],
     type="l",
     ylab="beta2",
     xlab="Sample")
abline(h=beta_true[2],
       col="red",
       lwd=2)  #True value
plot(results$lambda[2001:20000],
     type="l",
     ylab="lambda",
     xlab="Sample")
abline(h=lambda_true,
       col="red",
       lwd=2)  #True value
plot(results$theta[2001:20000],
     type="l",
     ylab="theta",
     xlab="Sample")
abline(h=theta_true,
       col="red",
       lwd=2)  #True value
```




```
par(mfrow=c(2,2))
plot(results$sigma2_w[2001:20000],
     type="l",
     ylab="sigma2_w",
     xlab="Sample")
abline(h=sigma2_w_true,
       col="red",
       lwd=2)  #True value
plot(results$phi[2001:20000],
     type="l",
     ylab="phi",
     xlab="Sample")
abline(h=phi_true,
       col="red",
       lwd=2)  #True value
plot(rowMeans(results$w[,2001:20000]), w_true,
     pch=16,
     ylab="True w",
     xlab="Estimated w")
abline(a=0,
       b=1)
```


sigma2_w
0
4
8
0
5000
10000
15000
Sample
phi
0
4
8
0
5000
10000
15000
Sample
True w
−1.5
0.0
−0.6
−0.2
0.2
0.6
Estimated w