# 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(4541)  
  
library(Spillover)  
library(geoR) #Spatial covariance functions

## --------------------------------------------------------------  
## Analysis of Geostatistical Data  
## For an Introduction to geoR go to http://www.leg.ufpr.br/geoR  
## geoR version 1.7-5.2.1 (built on 2016-05-02) is now loaded  
## --------------------------------------------------------------

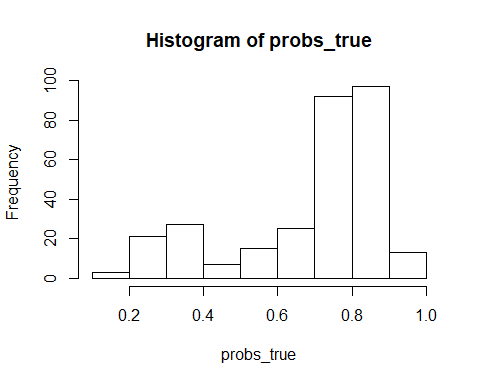
library(mnormt) #Multivariate normal distribution  
library(boot) #Inverse logit transformation

* 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)

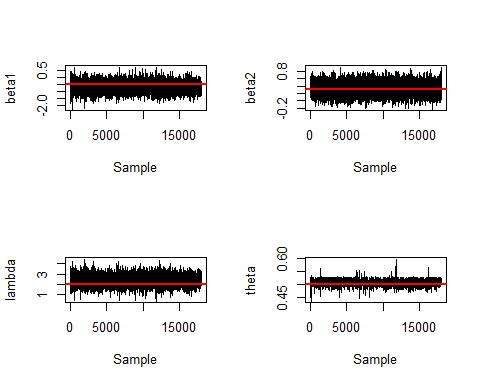
[2] Fit Spillover to the Data:

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 = 0.10)

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 10%  
## phi Acceptance: 28%  
## theta Acceptance: 22%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 20%  
## phi Acceptance: 28%  
## theta Acceptance: 22%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 30%  
## phi Acceptance: 28%  
## theta Acceptance: 22%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 40%  
## phi Acceptance: 28%  
## theta Acceptance: 22%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 50%  
## phi Acceptance: 28%  
## theta Acceptance: 22%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 60%  
## phi Acceptance: 28%  
## theta Acceptance: 22%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 70%  
## phi Acceptance: 28%  
## theta Acceptance: 23%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 80%  
## phi Acceptance: 27%  
## theta Acceptance: 23%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 90%  
## phi Acceptance: 27%  
## theta Acceptance: 23%  
## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
## Gaussian Spillover  
## Progress: 100%  
## phi Acceptance: 27%  
## theta Acceptance: 23%

[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)

