lezione 24-10-29

December 22, 2024

0.1 Esempio

Campionioniamo da una normale bivariata

$$\mathbf{X} \sim N_2(\mu, \Sigma)$$

[2]:
```
set.seed(10)
mu = c(10,20)
Sigma = matrix(c(1, 0.95,0.95,1), ncol=2 )

mu
Sigma
```

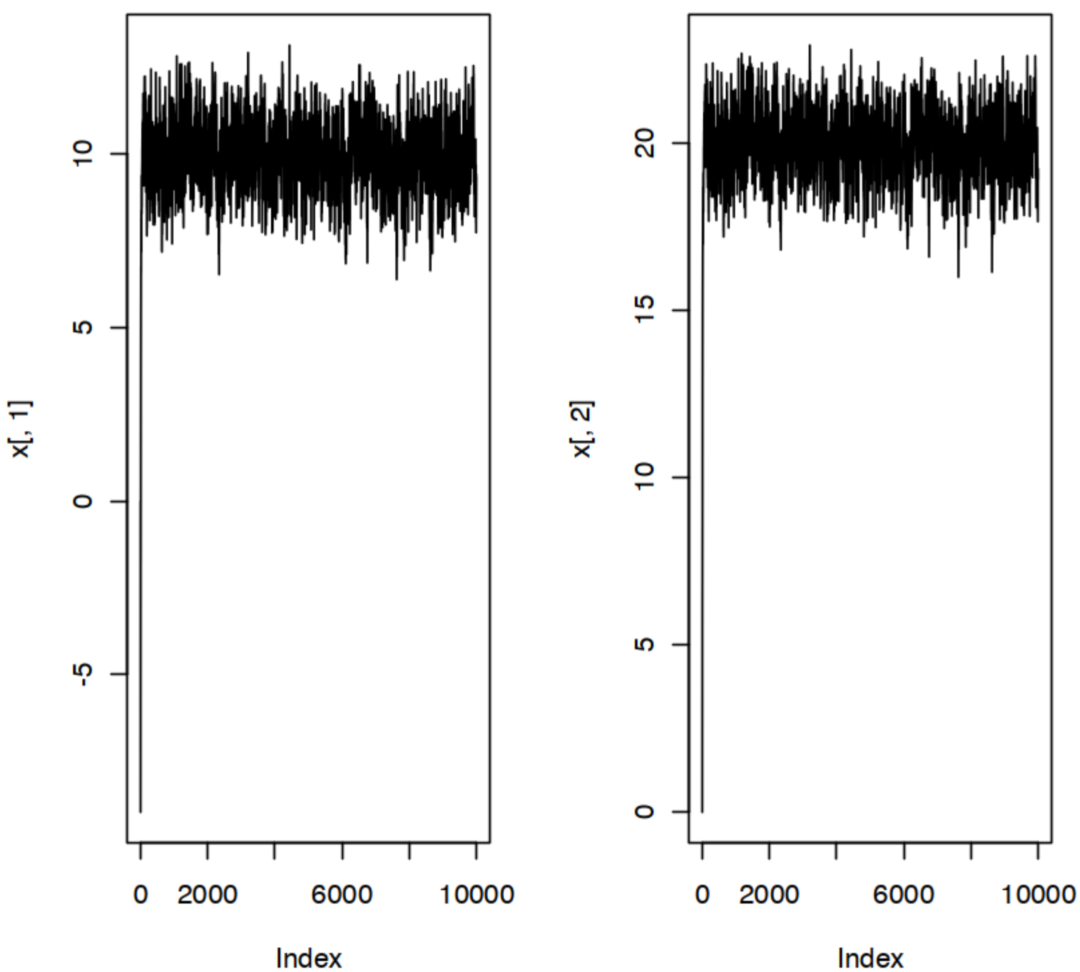
1. 10 2. 20

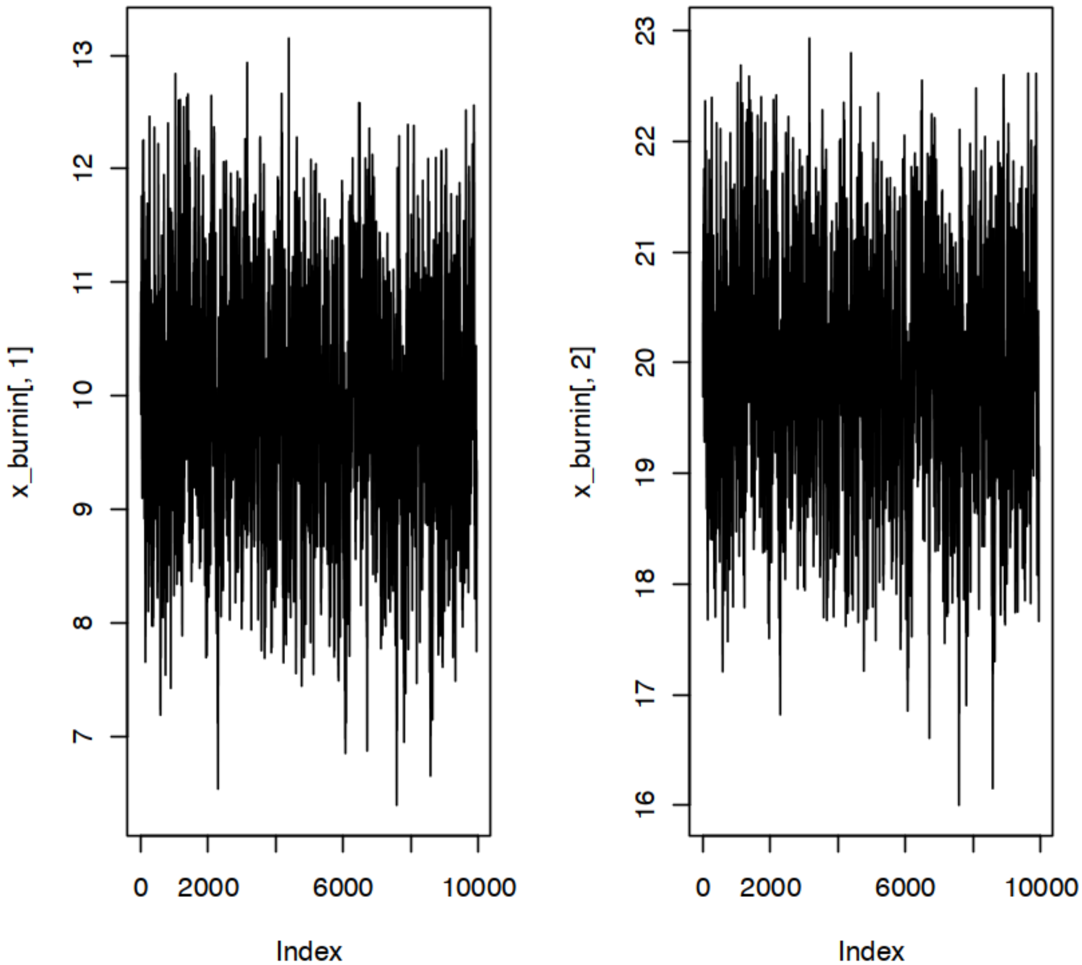
A matrix: 2 x 2 of type dbl

1.00	0.95
0.95	1.00

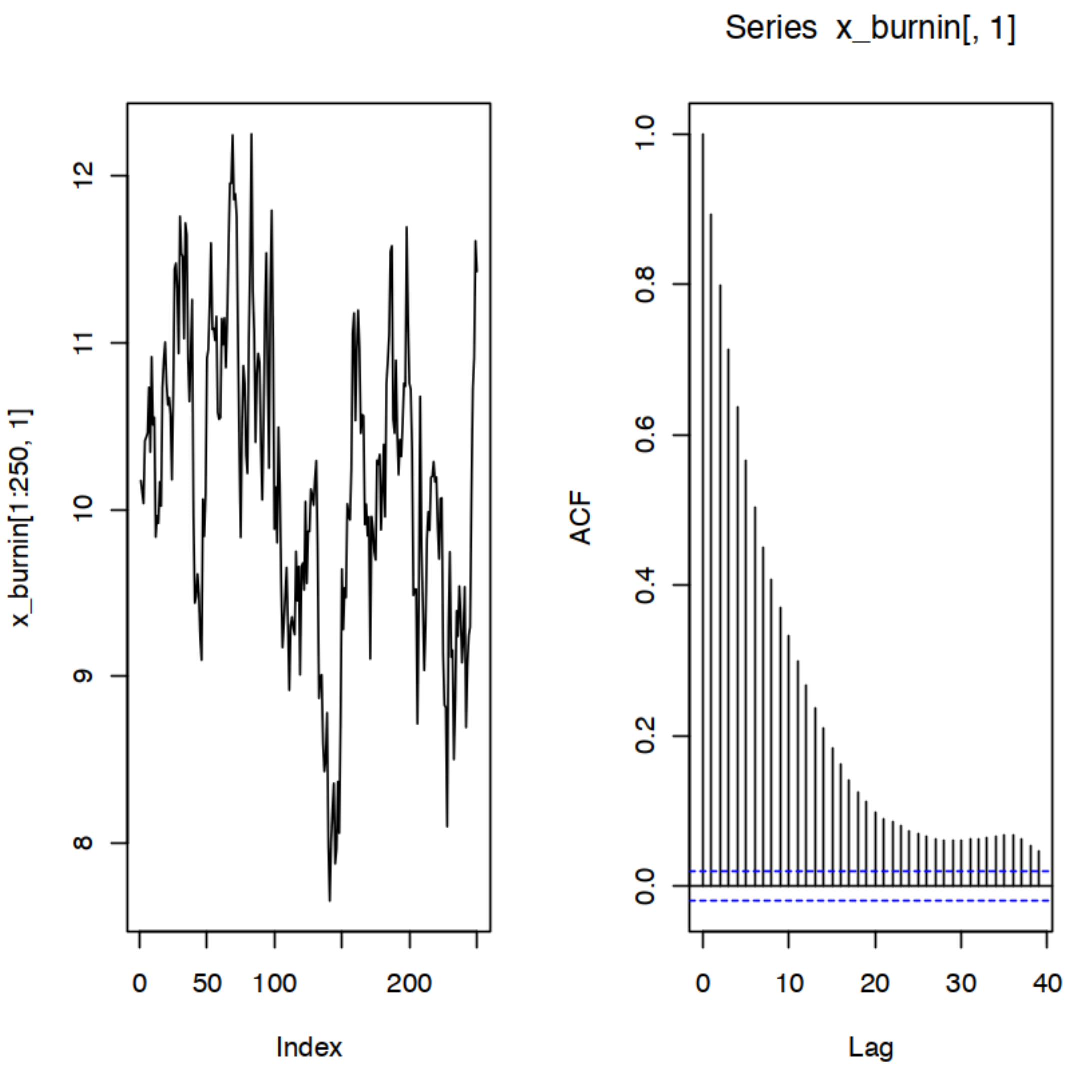
[5]:
```
B = 10000
x = matrix(NA, ncol= 2, nrow = B)
x[1, ] = c(0,0)

for(isim in 2:B)
{

  ## campioni X_A
  mean_cond = mu[1] + Sigma[1,2]*Sigma[2,2]^(-1)*(x[isim-1,2] - mu[2])
  var_cond = Sigma[1,1] - Sigma[1,2]*Sigma[2,2]^(-1)*Sigma[1,2]

  x[isim, 1] = rnorm(1, mean_cond, var_cond^0.5)

  ## campioni X_B
  mean_cond = mu[2] + Sigma[2,1]*Sigma[1,1]^(-1)*(x[isim,1] - mu[1])
  var_cond = Sigma[2,2] - Sigma[2,1]*Sigma[1,1]^(-1)*Sigma[2,1]

  x[isim, 2] = rnorm(1, mean_cond, var_cond^0.5)

}
```

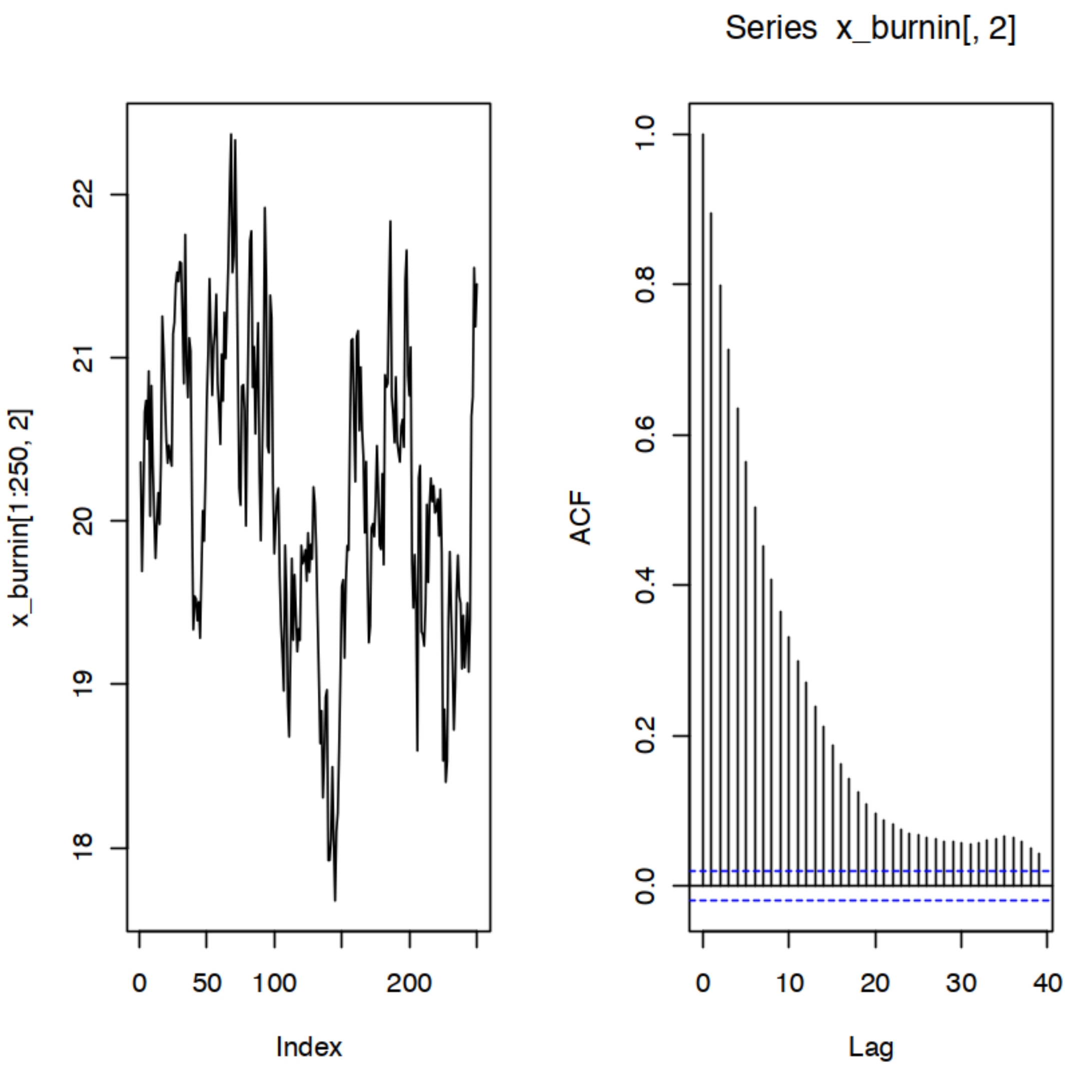
[9]:
```
par(mfrow=c(1,2))
plot(x[,1], type="l")
plot(x[,2], type="l")


x_burnin = x[-c(1:50),]
par(mfrow=c(1,2))
plot(x_burnin[,1], type="l")
plot(x_burnin[,2], type="l")
```



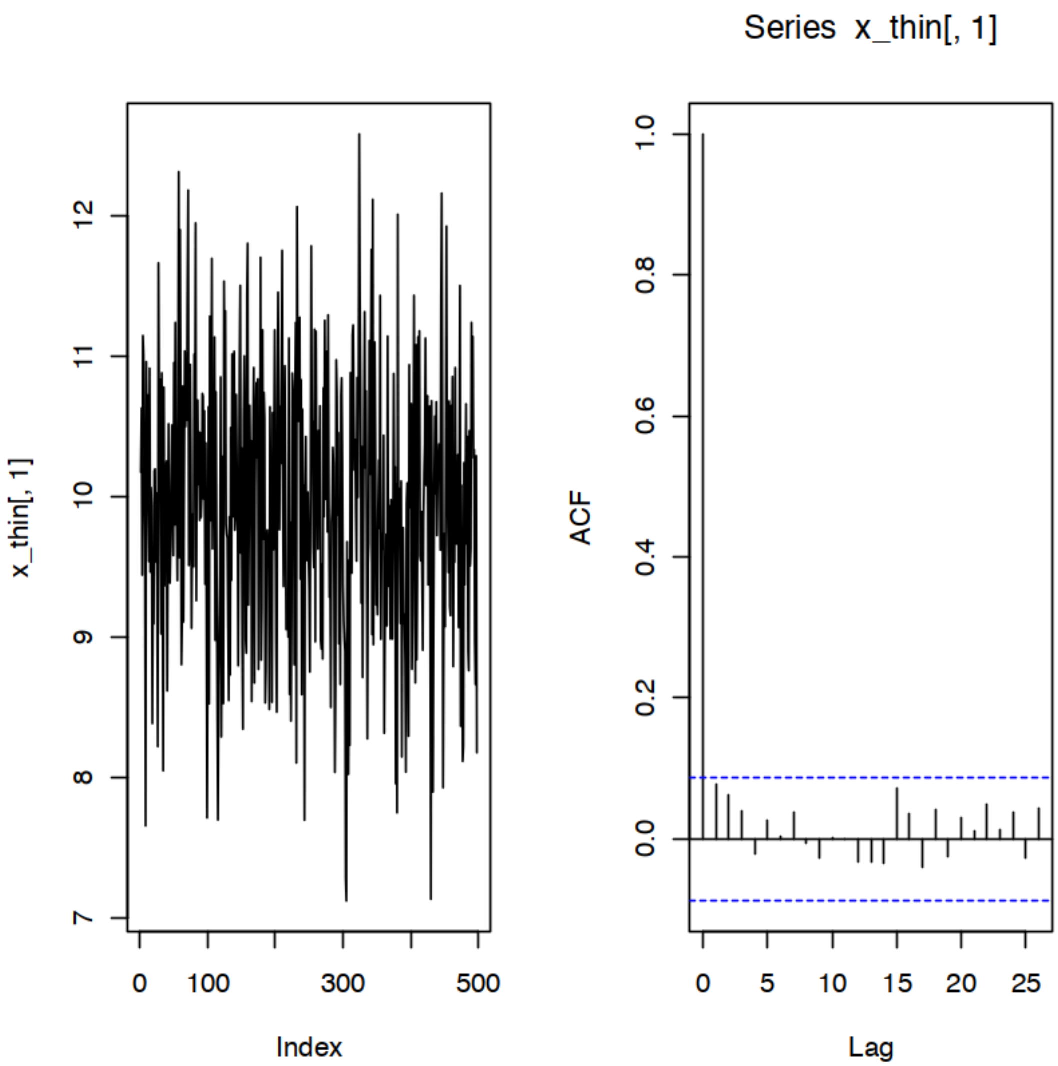


[11]:
```
par(mfrow=c(1,2))
plot(x_burnin[1:250,1], type="l")
acf(x_burnin[,1])
plot(x_burnin[1:250,2], type="l")
acf(x_burnin[,2])
```

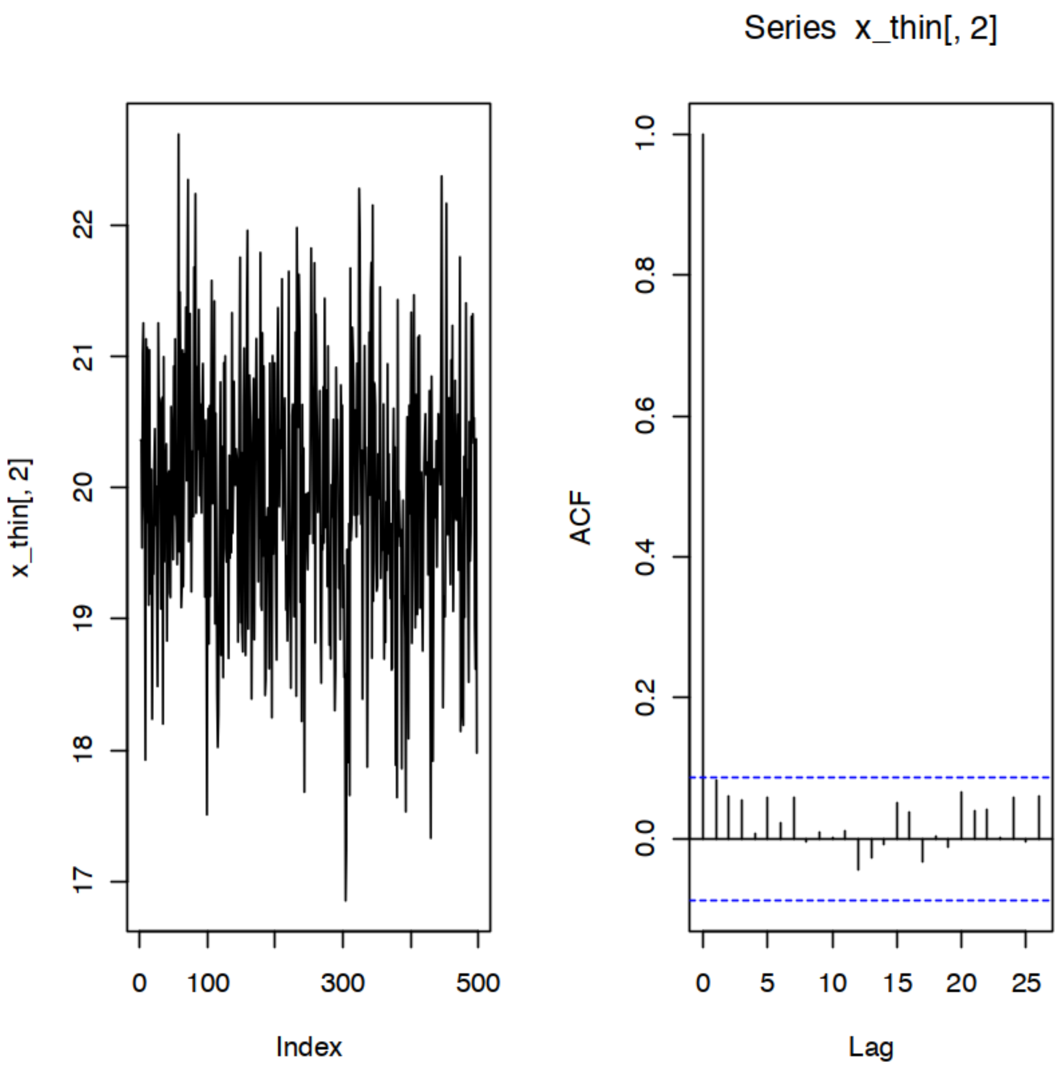

Series x_burnin[, 1]
x_burnin[1:250, 1]
Index
ACF
Lag



[13]:
```
x_thin = x_burnin[seq(1, nrow(x_burnin), by = 20),]
par(mfrow=c(1,2))
plot(x_thin[,1], type="l")
acf(x_thin[,1])
plot(x_thin[,2], type="l")
acf(x_thin[,2])
```

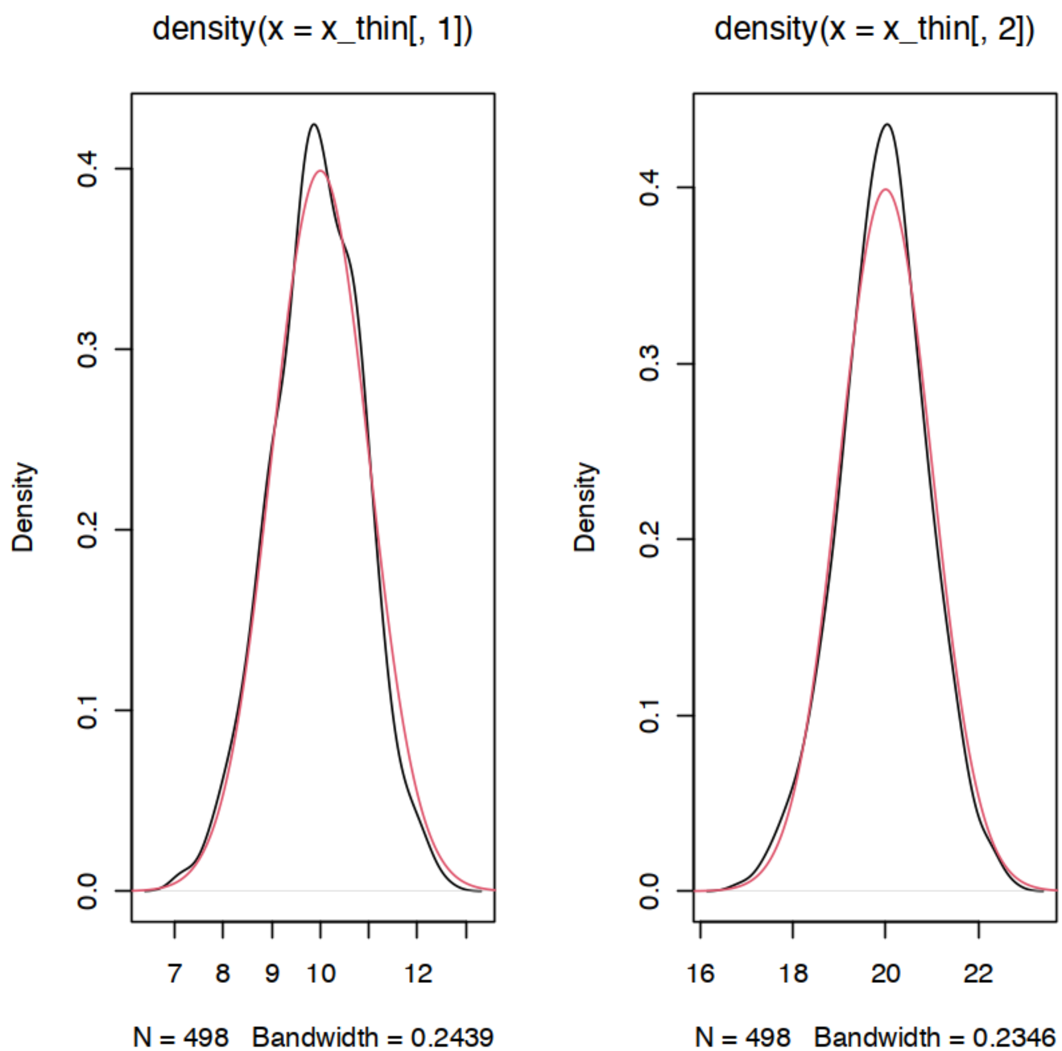


Series x_thin[, 1]
x_thin[, 1]
Index
ACF
Lag



[23]:

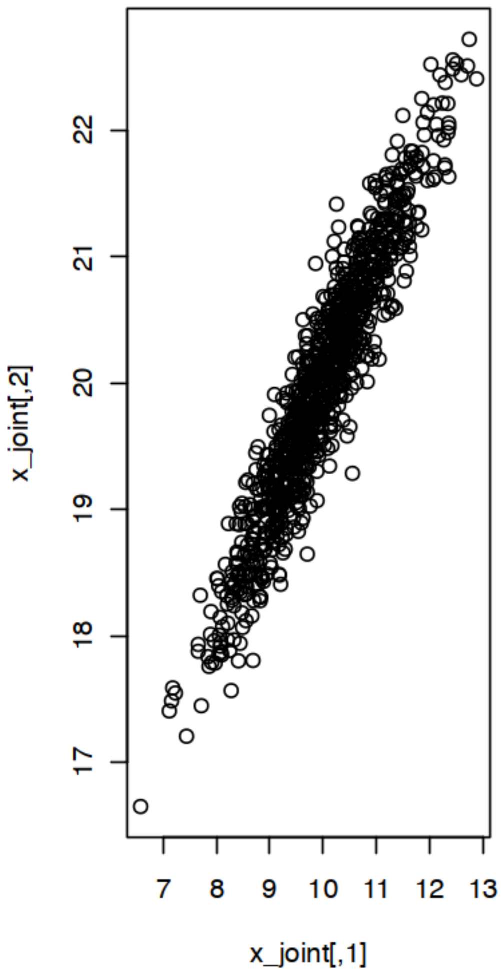
```
#plot(x_thin[,1])
#abline(h = mu[1], col=2)

par(mfrow=c(1,2))
plot(density(x_thin[,1]))
xseq = seq(5,15, by = 0.01)
lines(xseq,dnorm(xseq, mu[1], Sigma[1,1]^0.5), col=2)

plot(density(x_thin[,2]))
xseq = seq(15,25, by = 0.01)
lines(xseq,dnorm(xseq, mu[2], Sigma[2,2]^0.5), col=2)
```



[26]:

```
rmnorm=function(n = 1, mean = rep(0, d), varcov)
{
  d <- if (is.matrix(varcov))
    ncol(varcov)
  else 1
  z <- matrix(rnorm(n * d), n, d) %*% chol(varcov)
  y <- t(mean + t(z))
  return(y)
}

x_joint = rmnorm(1000, mean = mu, varcov = Sigma)
```

[27]:
```
par(mfrow=c(1,2))
plot(x_joint)
plot(x_thin)
```

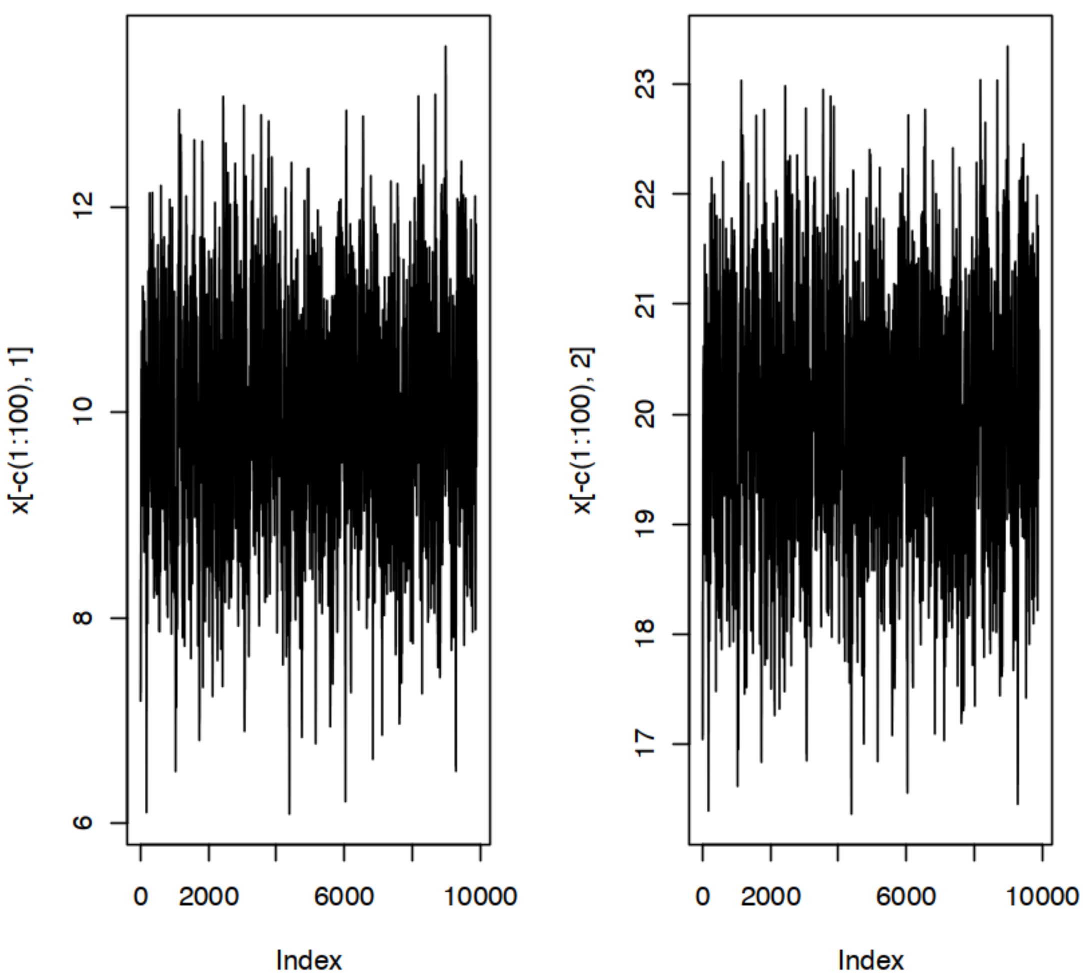


[28]:
```
B = 10000
x = matrix(NA, ncol= 2, nrow = B)
x[1, ] = c(-20000,-100000)

for(isim in 2:B)
{

  ## campioni X_A
  mean_cond = mu[1] + Sigma[1,2]*Sigma[2,2]^(-1)*(x[isim-1,2] - mu[2])
```

```
  var_cond = Sigma[1,1] - Sigma[1,2]*Sigma[2,2]^(-1)*Sigma[1,2]

  x[isim, 1] = rnorm(1, mean_cond, var_cond^0.5)

  ## campioni X_B
  mean_cond = mu[2] + Sigma[2,1]*Sigma[1,1]^(-1)*(x[isim,1] - mu[1])
  var_cond = Sigma[2,2] - Sigma[2,1]*Sigma[1,1]^(-1)*Sigma[2,1]

  x[isim, 2] = rnorm(1, mean_cond, var_cond^0.5)

}
```

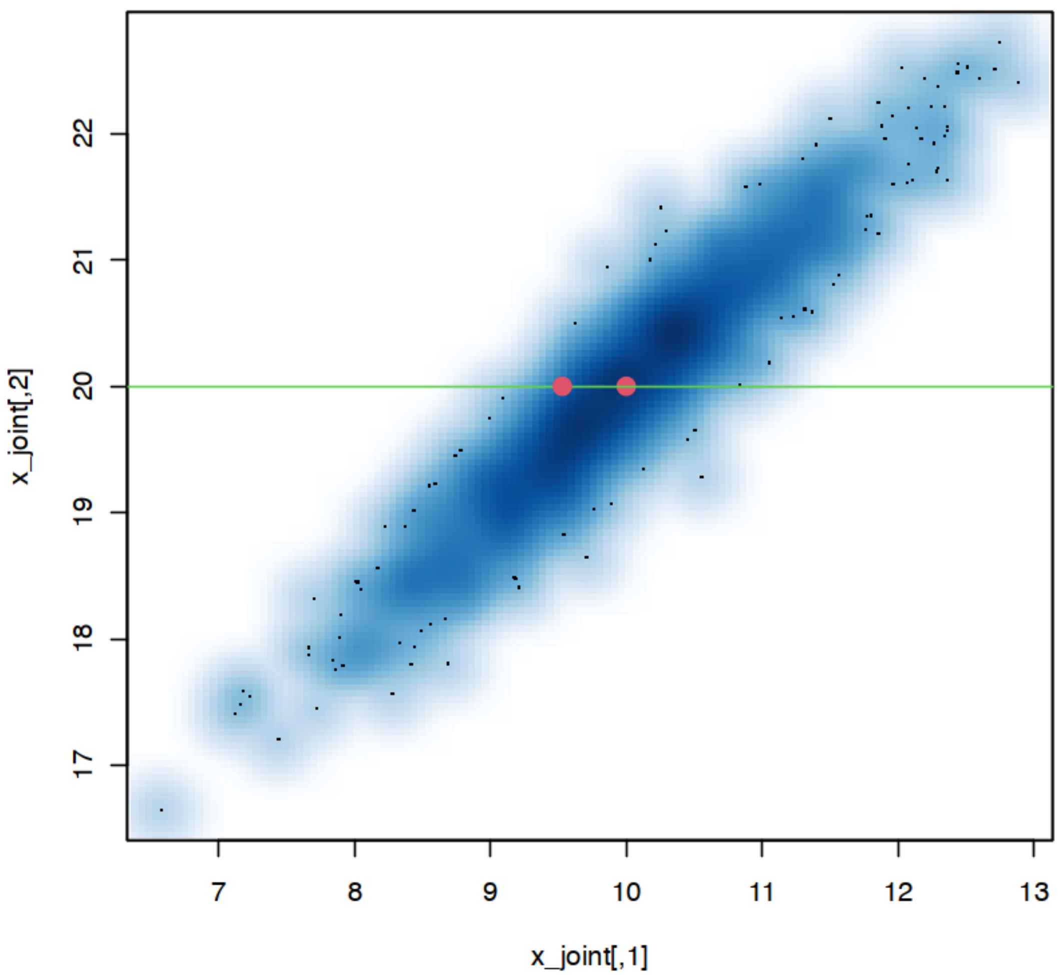
[31]:

```
par(mfrow=c(1,2))
plot(x[-c(1:100),1], type="l")
plot(x[-c(1:100),2], type="l")
```

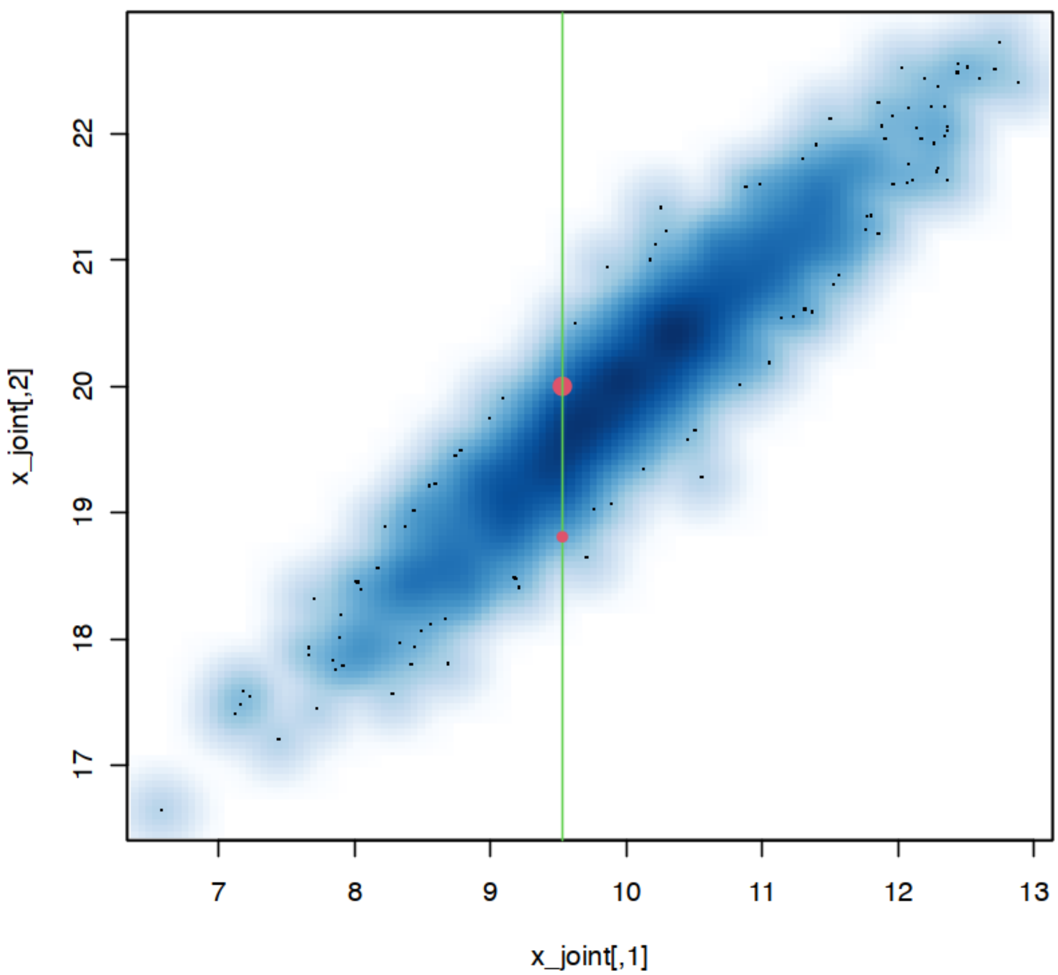


[38]:

```
## Gibbs
B = 10000
x = matrix(NA, ncol= 2, nrow = B)
x[1, ] = mu



## b = 2
isim = 2

smoothScatter(x_joint)
points(x[isim-1, 1], x[isim-1,2], pch = 20, cex = 2, col=2)
abline(h = x[isim-1,2], col=3)
mean_cond = mu[1] + Sigma[1,2]*Sigma[2,2]^(-1)*(x[isim-1,2] - mu[2])
var_cond = Sigma[1,1] - Sigma[1,2]*Sigma[2,2]^(-1)*Sigma[1,2]

x[isim, 1] = rnorm(1, mean_cond, var_cond^0.5)
points(x[isim, 1], x[isim-1,2], pch = 20, cex = 2, col=2)
```

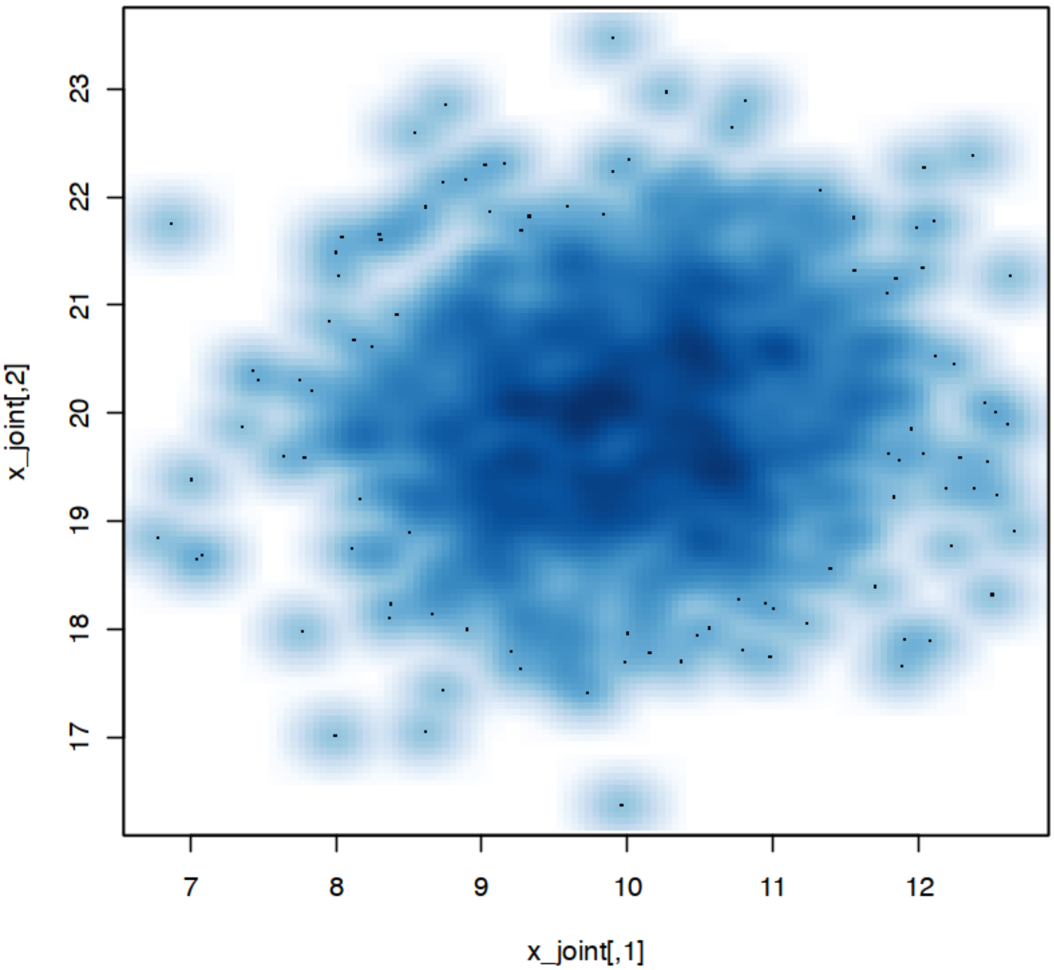


[42]:
```
smoothScatter(x_joint)
points(x[isim, 1], x[isim-1,2], pch = 20, cex = 2, col=2)
abline(v = x[isim,1], col=3)

mean_cond = mu[2] + Sigma[2,1]*Sigma[1,1]^(-1)*(x[isim,1] - mu[1])
var_cond = Sigma[2,2] - Sigma[2,1]*Sigma[1,1]^(-1)*Sigma[2,1]

x[isim, 2] = rnorm(1, mean_cond, var_cond^0.5)
points(x[isim, 1], x[isim, 2], pch = 20, col=2)
```



[45]:
```
Sigma = matrix(c(1, 0.1,0.1,1), ncol=2 )
x_joint = rmnorm(1000, mean = mu, varcov = Sigma)
```

[46]:
```
smoothScatter(x_joint)
```



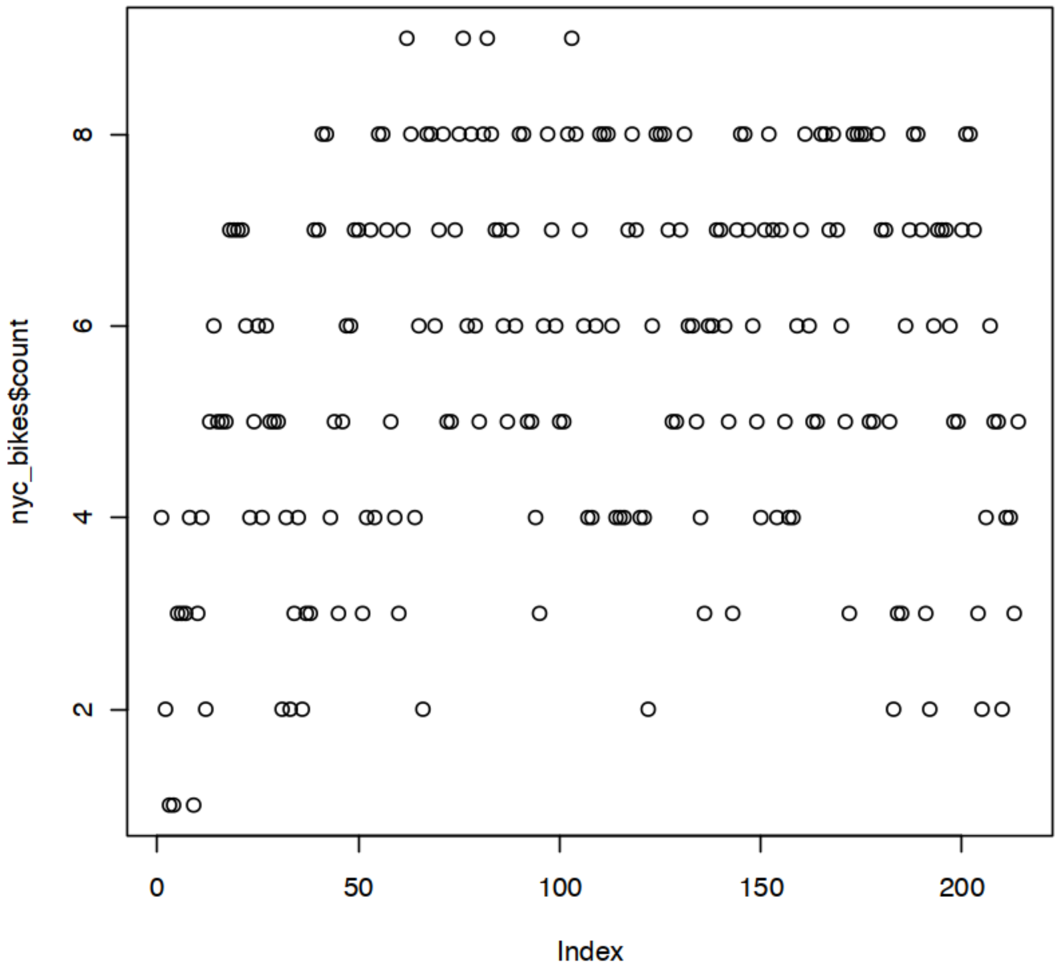
1 Bike

The data #### #### #### #### #### #### #### #### DESCRIZIONE #### Numero (in migliagia) di bici osservate passare sul ponte Williamsburg #### di New York, dal prima aprile al 31 ottobre. Ordinate temporalmente #### #### weekend: giorno della settimana #### hightemp: temperatura massima giornaliera #### lowtemp: temperatura massima giornaliera #### precip_rain: quantità di pioggia giornaliera #### precip_snow: quantità di neve giornaliera #### time: variabile che indica la differenza tra il giorno osservato e #### il 31 marzo #### count: numero di biciclette (in migliaia) #### #### #### #### ####

[48]:
```r
setwd("/Users/gianlucamastrantonio/Dropbox (Politecnico di Torino Staff)/2025 -␣
↪01OHENG - Modelli statistici_Statistica computazionale Stat - 281501/
↪datasets/")
nyc_bikes = read.csv("nyc_bike_counts.csv")
summary(nyc_bikes)
```

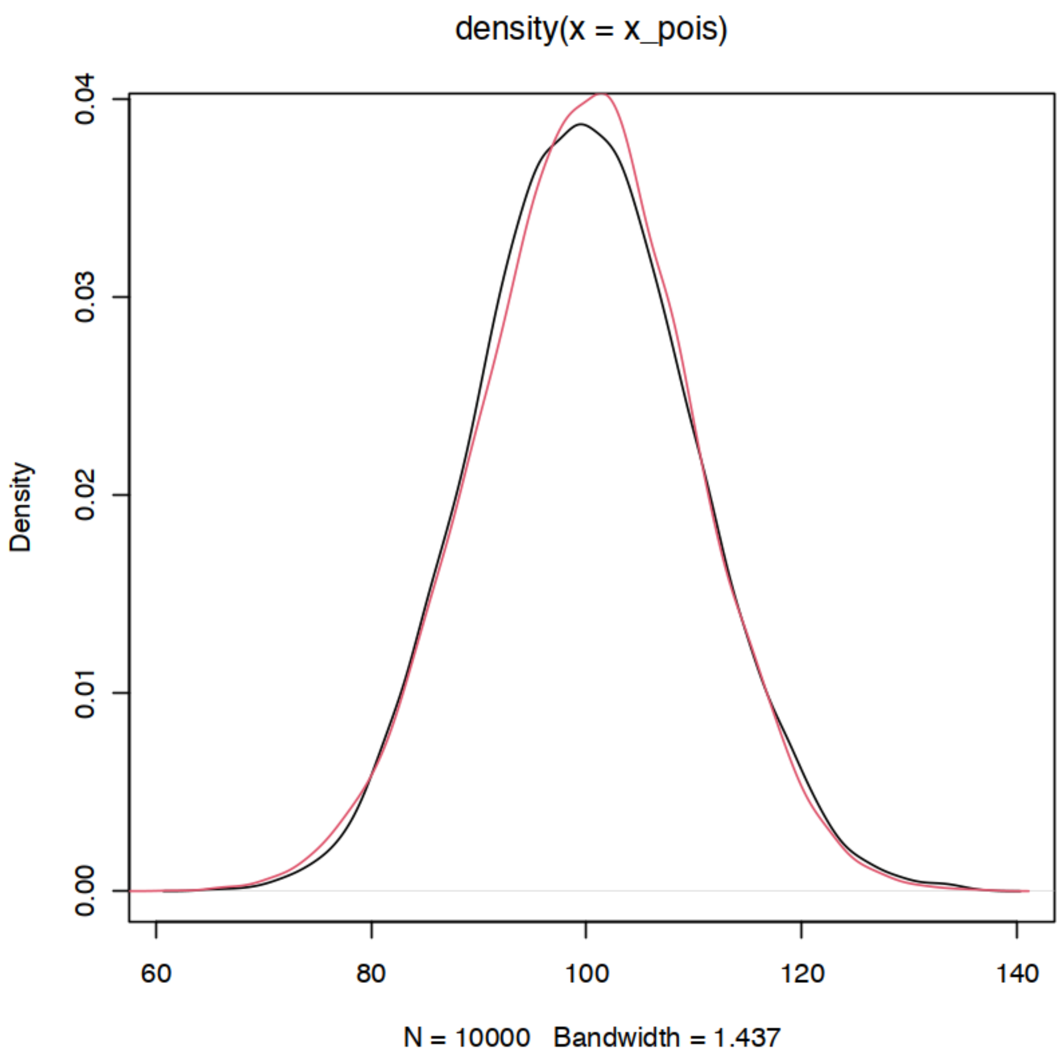
```
   weekday             hightemp        lowtemp        precip_rain
 Length:214         Min.   :39.90   Min.   :26.10   Min.   :0.0000
 Class :character   1st Qu.:66.05   1st Qu.:53.23   1st Qu.:0.0000
 Mode  :character   Median :78.10   Median :64.90   Median :0.0000
                    Mean   :74.93   Mean   :61.97   Mean   :0.1069
                    3rd Qu.:84.90   3rd Qu.:71.10   3rd Qu.:0.0400
                    Max.   :96.10   Max.   :82.00   Max.   :1.6500
  precip_snow           count            time
 Min.   :0.000000   Min.   :1.000   Min.   :  1.00
 1st Qu.:0.000000   1st Qu.:4.000   1st Qu.: 54.25
 Median :0.000000   Median :6.000   Median :107.50
 Mean   :0.002196   Mean   :5.659   Mean   :107.50
 3rd Qu.:0.000000   3rd Qu.:7.000   3rd Qu.:160.75
 Max.   :0.470000   Max.   :9.000   Max.   :214.00
```

[49]:
```r
plot(nyc_bikes$count)
```



[50]:

```
x_pois = rpois(10000, 100)
x_norm = rnorm(10000, 100, 100^0.5)

plot(density(x_pois))
lines(density(x_norm), col=2)
```



$$f(\mathbf{u}, \beta, \sigma^2|\mathbf{X}) \propto f(\mathbf{x}, \mathbf{u}|\beta, \sigma^2) f(\beta) f(\sigma^2)$$

$$f(\mathbf{x}, \mathbf{u}|\beta, \sigma^2) = \prod_{i=1}^{n} (2\pi\sigma^2)^{-0.5} \exp(-\frac{(1000x_i + u_i - \mu_i)^2}{2\sigma^2})$$

oppure

$$f(\mathbf{x}, \mathbf{u}|\beta, \sigma^2) = \prod_{i=1}^{n} (2\pi\sigma^2)^{-0.5} \exp(-\frac{(y_i - \mu_i)^2}{2\sigma^2})$$

con

$$y_i = 1000x_i + u_i$$

$$\mu_i = {}_i\beta$$

[55]:

```
Cov_mat = model.matrix( ~ weekday + time, data = nyc_bikes)
Cov_mat[1:10,]
p = ncol(Cov_mat)
n = nrow(Cov_mat)

burnin = 100
thin = 2
iterations = 10000

sample_to_save = floor((iterations-burnin)/thin)

x = nyc_bikes$count
x1000 = x*1000
sigma2_mcmc = 1
beta_mcmc = matrix(0, ncol=1, nrow=p)
u = matrix(0, ncol=1, nrow=n)
y = x1000 +u

beta_out = matrix(NA, ncol= p, nrow = sample_to_save)
sigma2_out = matrix(NA, ncol=1, nrow= sample_to_save)
u_out = matrix(NA, ncol=n, nrow= sample_to_save)

app_iter = burnin
for(isave in 1:sample_to_save)
{
  for(isim in 1:app_iter)
  {
    ## campiono beta
    var_p = solve(t(Cov_mat)%*%diag(1/sigma2_mcmc,n)%*%Cov_mat + diag(1/100,p))
    mean_p = var_p%*%(t(X)%*%diag(1/sigma2_mcmc,n)%*%z_mcmc + diag(1/
↪prior_mu_var, p )%*%matrix(prior_mu_mean, ncol=1,nrow=p))
    beta_mcmc[1:p] = rmnorm(n = 1, mean = mean_p, var_p)

    ### var
    a_post = prior_sigma2_a + n/2
    b_post = prior_sigma2_b + 0.5*sum( (z_mcmc-X%*%beta_mcmc)^2  )
    sigma2_mcmc = 1/rgamma(1, shape=a_post, rate=b_post )



    ## campioni sigma 2

    ## campiono u
    for(i in 1:n)
    {

    }
```

```
  }
  app_iter = thin
  ### Salvo i parametri
  beta_out[isave, ] = beta_mcmc
  sigma2_out[isave, ] = sigma2_mcmc
  u_out[isave, ] = u_mcmc
}
```

A matrix: 10 x 8 of type dbl

	(Intercept)	weekdayMonday	weekdaySaturday	weekdaySunday	weekdayT
1	1	0	0	0	0
2	1	0	1	0	0
3	1	0	0	1	0
4	1	1	0	0	0
5	1	0	0	0	0
6	1	0	0	0	0
7	1	0	0	0	1
8	1	0	0	0	0
9	1	0	1	0	0
10	1	0	0	1	0

```
Error in eval(expr, envir, enclos): oggetto 'u_mcmc' non trovato
Traceback:
```

```
[ ]:
```