Esercitazione 4

October 18, 2024

1 Statistica Bayesiana

1.1 Regressione

Abbiamo una regressione

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

con $i = 1, \dots, n$ e

$$\epsilon_i \sim N(0, 1)$$

e assumete che le prior siano

$$\beta_0, \beta_1 \sim N(0, 100)$$

Usando $n = 10$, $\beta_0 = 1$, $\beta_2 = 0.5$ e $\mathbf{x}$ è un vettore di variabili simualte da $U(0, 1)$

1. Simulate dei campioni dalla a posteriori (tramite campinamento diretto) e fate un plot delle a posteriori univariate e della bivariata
2. Stimate tramite mote carlo la media dei parametri e la correlazione.
3. Create un intervallo al 95% delle a posteriori univariate (che è chiamato **intervallo di credibilità** al 95%) e chiedetevi se c'è abbastanza evidenza che $\beta_2 > 0$
4. rifate il punto 3 assumendo $n = 1000$ e plottate le 2 a posteriori di β_2 e la a priori

SOLUZIONI

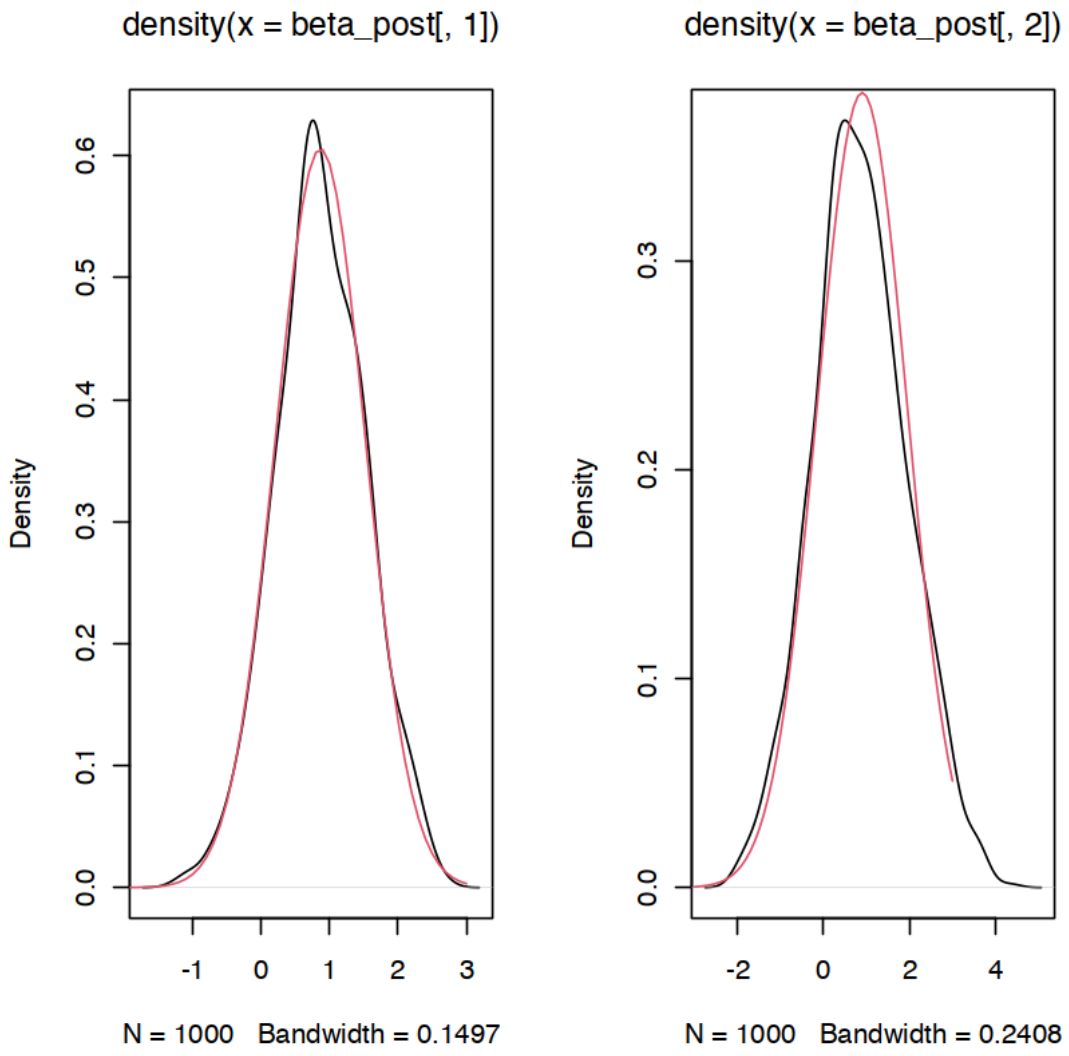
Dalla teoria sappiamo come simulare i β di una regressione con prior normali

[65]:
```
set.seed(100)
n = 10
beta0 = 1
beta1 = 0.5
sigma2 = 1
prior_mean = 0
prior_var = 100
x = runif(n, 0,1)
epsilon = rnorm(n,0, sigma2^0.5)
y = beta0+beta1*x +epsilon
```

[66]:
```
B = 1000
### simulo dalla a posteriori
X = matrix(1, nrow = n, ncol=2)
X[,2] = x
```

```
# varianza a posteriori
v_p = solve( t(X)%*%solve(diag(sigma2,n))%*%X + solve(diag(prior_var,2)))

# media a posteriori
m_p = v_p%*%(  t(X)%*%solve(diag(sigma2,n))%*%matrix(y, ncol=1) +␣
 ↪solve(diag(prior_var,2))%*%matrix(prior_mean, ncol=1, nrow=2))

v_p_chol = t(chol(v_p))

beta_post = matrix(NA, nrow = B, ncol=2)

for(b in 1:B)
{
    beta_post[b,] = m_p + v_p_chol%*%matrix(rnorm(2, 0,1), ncol=1)
}
```

[67]:
```
xseq = seq(-3,3, by = 0.1)
par(mfrow=c(1,2))
plot(density(beta_post[,1]))
lines(xseq ,dnorm(xseq, m_p[1], v_p[1,1]^0.5), col=2)
plot(density(beta_post[,2]))
lines(xseq, dnorm(xseq, m_p[2], v_p[2,2]^0.5), col=2)
par(mfrow=c(1,1))
```



[68]:
```
# punto 2
post_mean = apply(beta_post, 2, mean)
post_cor = cor(beta_post)
post_mean
post_cor
```

1. 0.875257397334247 2. 0.861478209280881

A matrix: 2 x 2 of type dbl

1.0000000	-0.8688382
-0.8688382	1.0000000

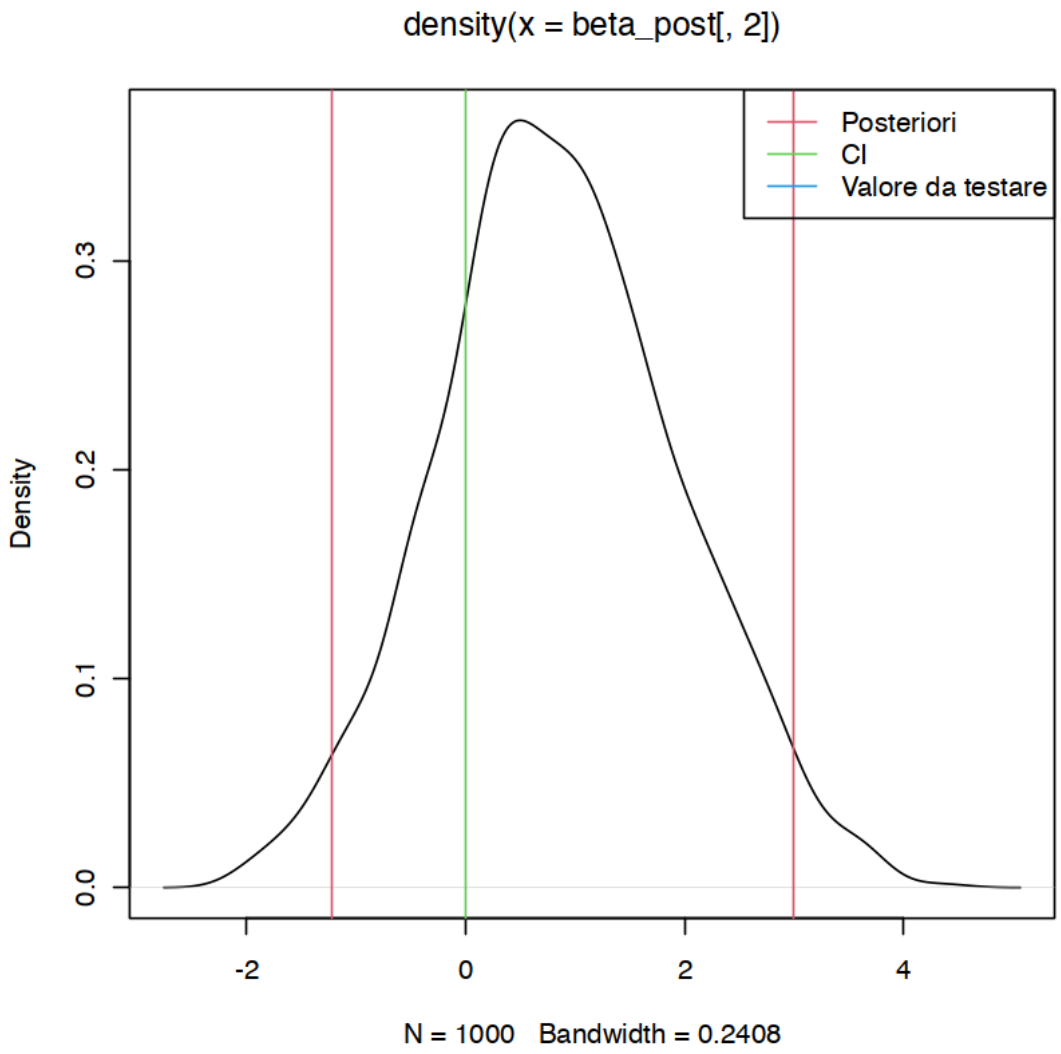
[69]:
```
# punto 3
ci_95_beta0 = quantile(beta_post[,1],prob = c(0.025, 0.975))
ci_95_beta1 = quantile(beta_post[,2],prob = c(0.025, 0.975))
```

```r
## oppure
ci95 = apply(beta_post,2, function(x) quantile(x,prob = c(0.025, 0.975)))
ci95

# valuto se l'intervallo contiene lo 0
(ci95[1,2]> 0)
plot(density(beta_post[,2]))
abline(v = ci95[,2], col=2)
abline(v = 0, col=3)
legend("topright", c("Posteriori", "CI", "Valore da testare"), col=2:4, lty = 1)
```

A matrix: 2 x 2 of type dbl

2.5%	-0.4236906	-1.228949
97.5%	2.1989150	2.991357

2.5\%: FALSE



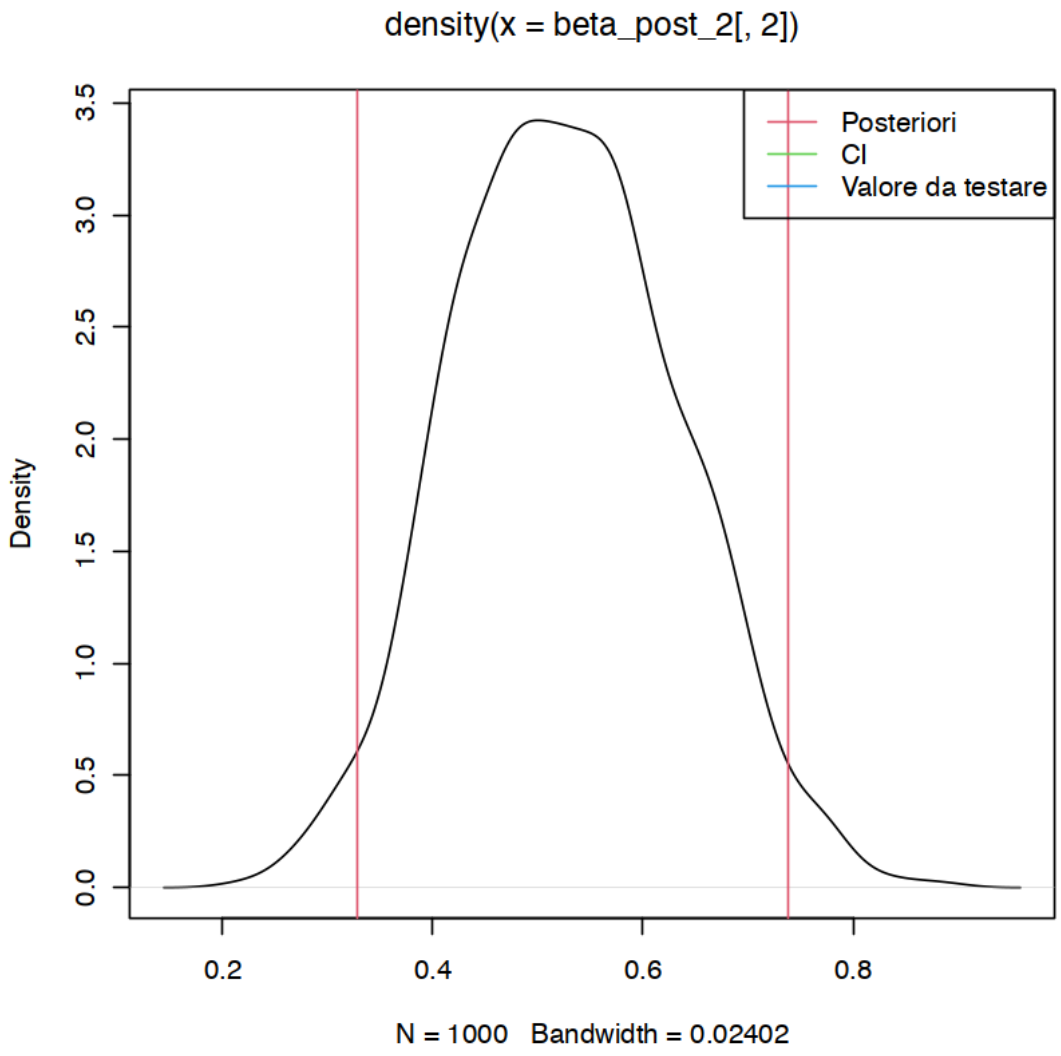
[70]:
```
# punto 4
n = 1000
beta0 = 1
beta1 = 0.5
sigma2 = 1
prior_mean = 0
prior_var = 100
x = runif(n, 0,1)
epsilon = rnorm(n,0, sigma2^0.5)
y = beta0+beta1*x +epsilon

B = 1000
### simulo dalla a posteriori
X = matrix(1, nrow = n, ncol=2)
X[,2] = x
# varianza a posteriori
v_p = solve( t(X)%*%solve(diag(sigma2,n))%*%X + solve(diag(prior_var,2)))

# media a posteriori
m_p = v_p%*%(  t(X)%*%solve(diag(sigma2,n))%*%matrix(y, ncol=1) +␣
 ↪solve(diag(prior_var,2))%*%matrix(prior_mean, ncol=1, nrow=2))

v_p_chol = t(chol(v_p))

beta_post_2 = matrix(NA, nrow = B, ncol=2)

for(b in 1:B)
{
    beta_post_2[b,] = m_p + v_p_chol%*%matrix(rnorm(2, 0,1), ncol=1)
}
```

[71]:
```
ci95_2 = apply(beta_post_2,2, function(x) quantile(x,prob = c(0.025, 0.975)))
ci95_2

# valuto se l'intervallo contiene lo 0
(ci95_2[1,2]> 0)
plot(density(beta_post_2[,2]))
abline(v = ci95_2[,2], col=2)
abline(v = 0, col=3)
legend("topright", c("Posteriori", "CI", "Valore da testare"), col=2:4, lty = 1)
```

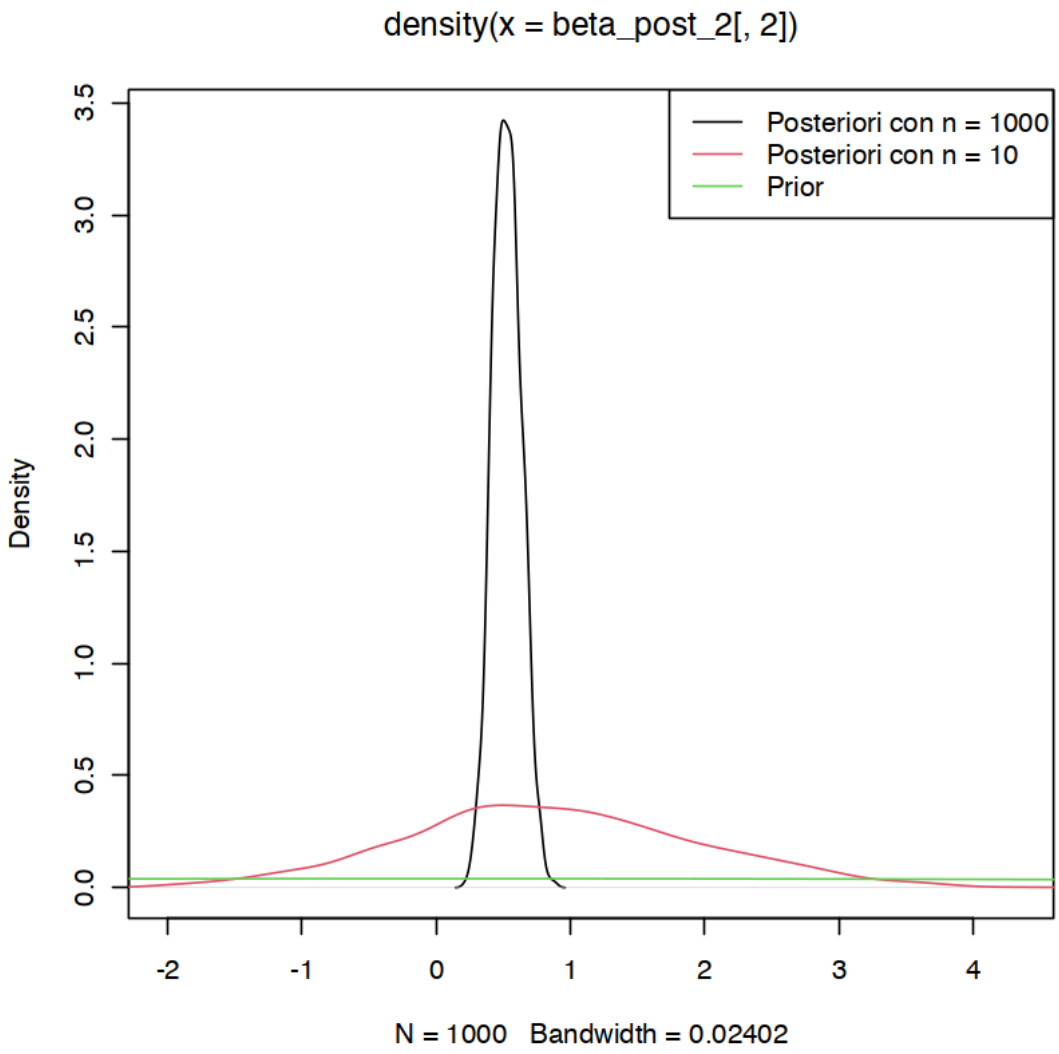
A matrix: 2 x 2 of type dbl

2.5%	0.9317481	0.3275822
97.5%	1.1696428	0.7379345

2.5\%: TRUE



[72]:
```
# plotto le due densità
xseq = seq(-5,5, by = 0.01)
plot(density(beta_post_2[,2]), xlim= range( c(beta_post_2,beta_post) ))
lines(density(beta_post[,2]), col=2)
lines(xseq, dnorm(xseq, prior_mean, prior_var^0.5), col=3)
legend("topright", c(paste("Posteriori con", c("n = 1000", "n = 10")),"Prior"),␣
↪col=1:3, lty = 1)
```



1.2 Distribuzione di Dirichlet

Abbiamo una variabile $Z_i \in \{1, 2, 3, 4, 5, 6\}$, il cui valore rappresenta uno dei valori usciti da un lancio di un dado. Abbiamo osservato le seguanti proporzioni di risultati

$$\frac{n_1}{n} = 0.1, \quad \frac{n_2}{n} = 0.2, \quad \frac{n_3}{n} = 0.1, \quad \frac{n_4}{n} = 0.2, \quad \frac{n_5}{n} = 0.2$$

Assumendo che

$$P(Z_i = j) = \pi_j$$

e che $\pi = (\pi_1, \pi_2, \pi_3, \pi_4, \pi_5, \pi_6)$. Avete che π è un vettore di probabilità e quindi, uno dei 6 elementi è funzione deterministica degli altri. Avete che la prior è

$$\pi \sim Dir(\alpha_1, \dots, \alpha_k) \qquad \alpha_j > 0$$

che ha densità

$$f(\pi) = \frac{\Gamma\left(\sum_{j=1}^{6} \alpha_j\right) \prod_{j=1}^{6} \pi_j^{\alpha_j - 1}}{\prod_{j=1}^{6} \Gamma(\alpha_j)}$$

Ci sono modi per ottenere la distribuzione Dirichlet di dimensione K:

Metodo 1

$$Y_j \sim G(\alpha_j, \theta), j = 1, \dots, K,$$

e definiamo

$$\pi_j = \frac{Y_j}{\sum_{h=1}^{K} Y_h}$$

allora

$$(\pi_1, \dots, \pi_K) \sim Dir(\alpha_1, \dots, \alpha_k)$$

Metodo 2 (stick breaking)

$$X_j \sim B(\alpha_j, \sum_{h=j+1}^{K} \alpha_h), \quad j = 1, \dots, K-1$$

Se

$$\pi_1 = X_1,$$

$$\pi_j = X_j \prod_{h=1}^{j-1}(1 - X_j) = X_j(1 - \sum_{h=1}^{j-1} \pi_h), \quad j = 2, \dots, K-1$$

$$\pi_K = 1 - \sum_{j=1}^{K-1}$$

allora

$$(\pi_1, \dots, \pi_K) \sim Dir(\alpha_1, \dots, \alpha_k)$$

Per l'esercizio, prendete valori di α_j vicini a 1

1. determinare la a posteriori in forma chiusa e ottenete dei campioni (n = 10)
2. c'è evidenza che il dato sia truccato e ci sia differenza tra i numeri pari e dispari, assumendo $n = 10$? e $n = 1000$?
3. con le stesse Z_i dei punti precedenti, valutate graficamente la a prior e la a posteriori di

   $$w_1 = \pi_1 + \pi_2, \quad w_2 = \pi_3 + \pi_4, \quad w_3 = \pi_5 + \pi_6,$$

   ricordando che su 3 componenti, solo 2 sono variabili aleatorie. Potete usare il seguente risultato (che è facile vedere se pensate alla Dirichlet come trasformazione di Gamma)

   $$(\pi_1, \pi_2, \pi_3, \dots, \pi_K) \sim Dir(\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_k)$$

   allora

   $$(\pi_1, \pi_2 + \pi_3, \dots, \pi_K) \sim Dir(\alpha_1, \alpha_2 + \alpha_3, \dots, \alpha_k)$$

SOLUZIONI

Per la a posteriori abbiamo che è

$$f(\pi|\mathbf{z}) \propto f(\mathbf{z}|\pi)f(\pi)$$

con

$$f(\mathbf{z}|\pi) = \pi_i^{n_1}\pi^{n_2} \dots \pi_6^{n_6}$$

La a posteriori è allora

$$f(\pi|\mathbf{z}) \propto \pi_i^{n_1+\alpha_1-1}\pi^{n_2+\alpha_2-1} \dots \pi_6^{n_6+\alpha_6-1}$$

e quindi

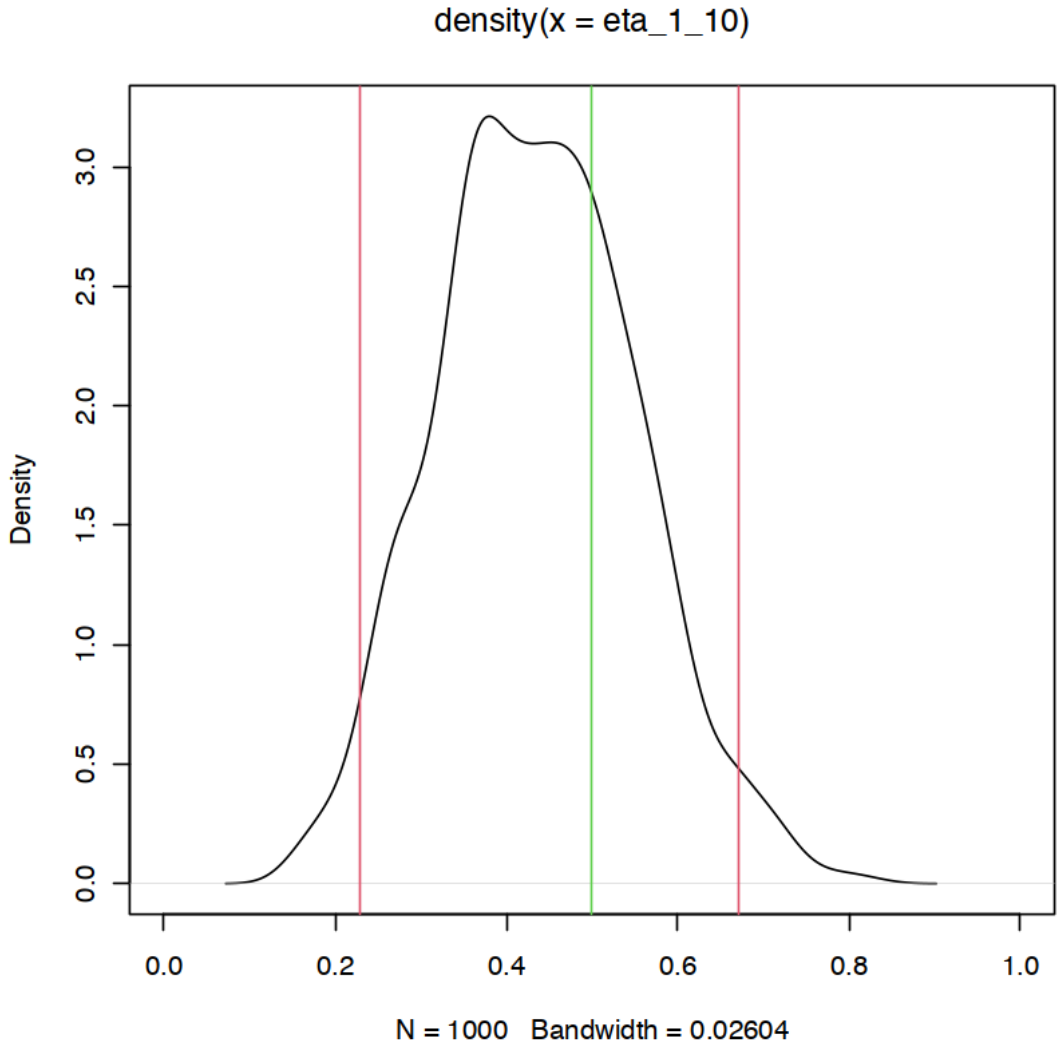
$$\pi|\mathbf{z} \sim Dir(n_1+\alpha_1, \dots, n_6+\alpha_6)$$

[73]:
```
sim_dir = function(alpha_vec)
{
    K = length(alpha_vec)
    x = rgamma(K,alpha_vec,1 )
    return(x/sum(x))
}
n = 10
alpha_vec = rep(1,6)
prop_init = c(0.1,  0.2,  0.1,   0.2,  0.2)
n_z = prop_init
n_z[6] = 1-sum(n_z)
n_z = n_z * n
```

[74]:
```
nsim = 1000
alpha_vec = c(1.2,1.2,1.2,1.2,1.2,1.2)
pi_sim_10 = matrix(NA, ncol=6, nrow = nsim)
for(isim in 1:nsim)
{
    pi_sim_10[isim,] = sim_dir(alpha_vec + n_z)
}
```

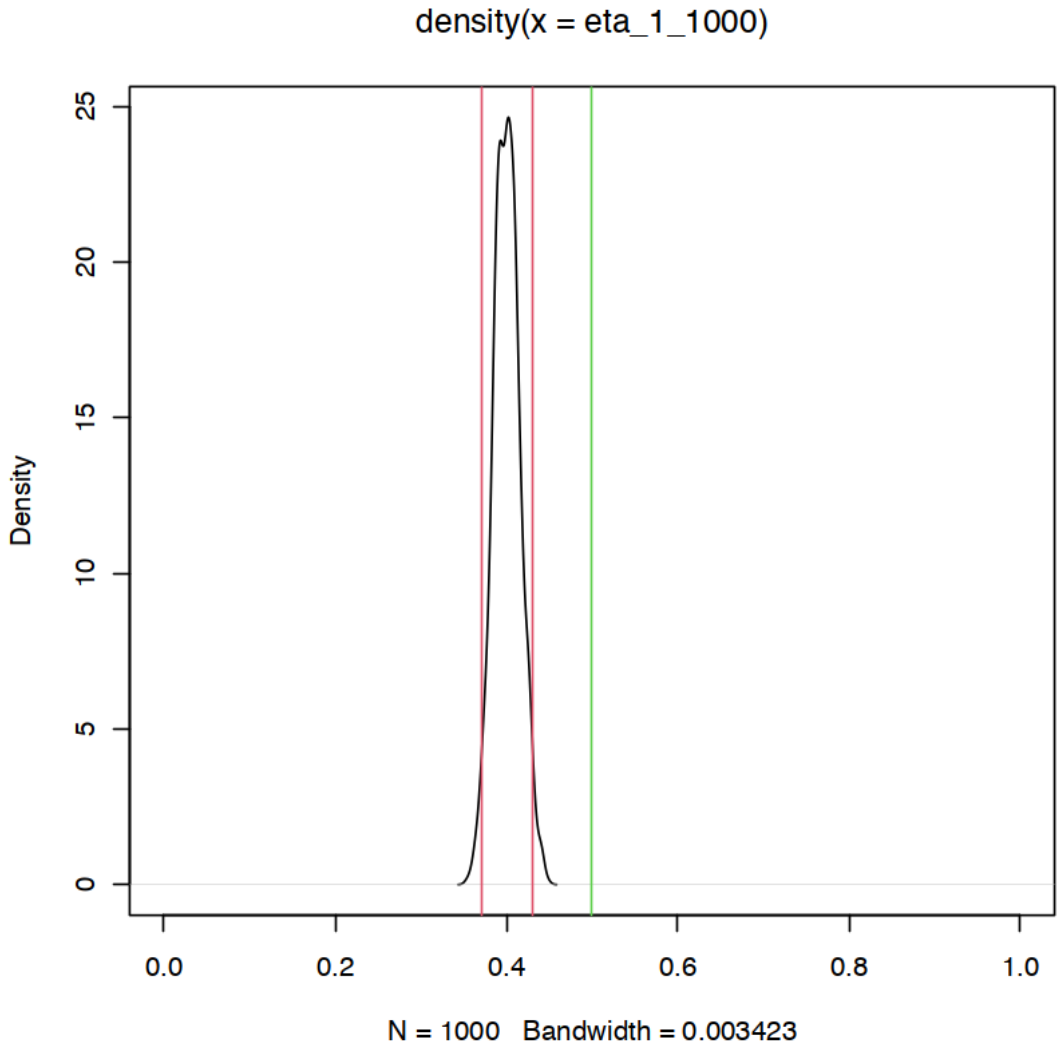
Per valutare il punto 2, possiamo valutare la distribuzione di $\eta_1 = \pi_1 + \pi_3 + \pi_5$ e $\eta_2 = \pi_2 + \pi_4 + \pi_6$. SOmmando i campioni corrispondenti, possiamo ottenere campioni diq euste sue variabili

[75]:
```
eta_1_10 = apply(pi_sim_10[,c(1,3,5)],1,sum)
eta_2_10 = apply(pi_sim_10[,c(1,3,5) +1],1,sum)
ci95_10 = quantile(eta_1_10,prob = c(0.025, 1 - 0.025))
plot(density(eta_1_10), xlim=c(0,1))
abline(v=ci95_10, col=2)
abline(v = 0.5, col=3)
```



[76]:
```r
# ripeto con n = 100
n = 1000
prop_init = c(0.1,  0.2,  0.1,   0.2,  0.2)
n_z = prop_init
n_z[6] = 1-sum(n_z)
n_z = n_z * n
nsim = 1000
alpha_vec = c(1.2,1.2,1.2,1.2,1.2,1.2)
pi_sim_1000 = matrix(NA, ncol=6, nrow = nsim)
for(isim in 1:nsim)
{
    pi_sim_1000[isim,] = sim_dir(alpha_vec+n_z)
}
```

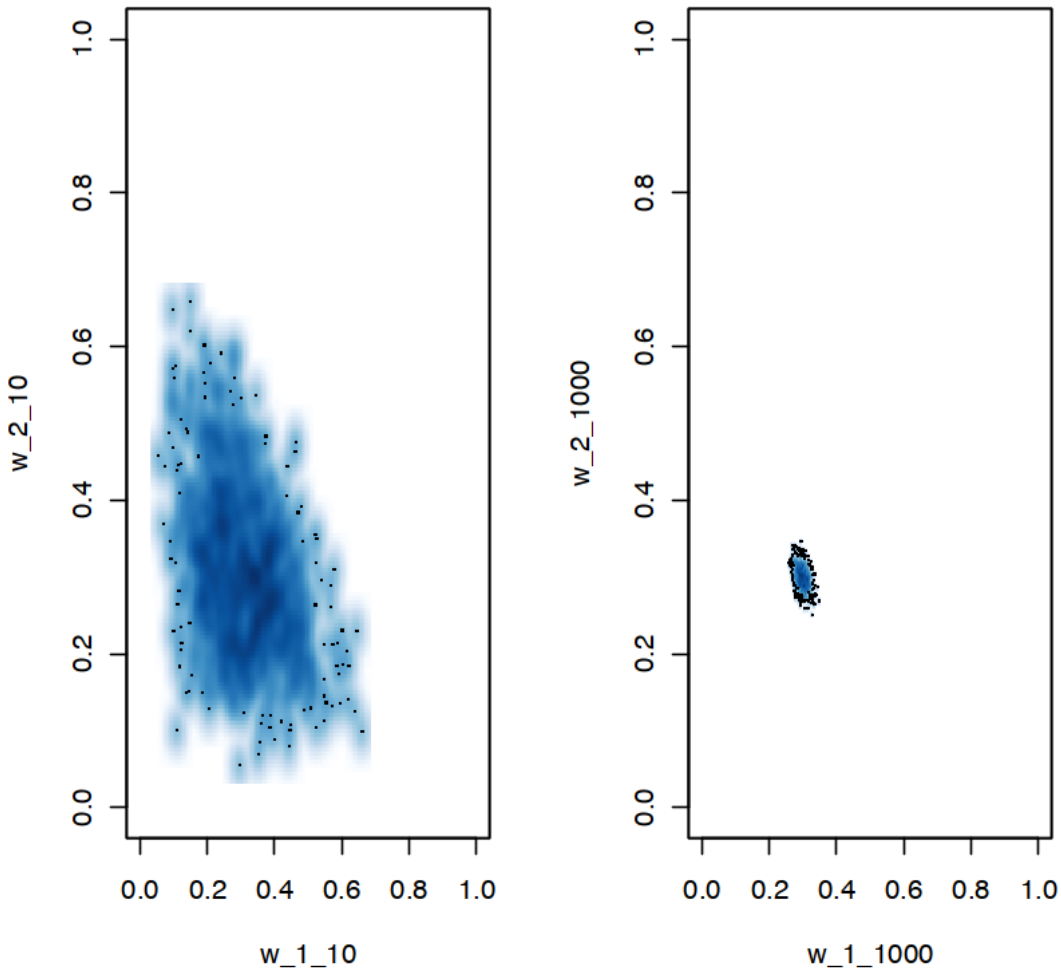
```
eta_1_1000 = apply(pi_sim_1000[,c(1,3,5)],1,sum)
eta_2_1000 = apply(pi_sim_1000[,c(1,3,5) +1],1,sum)
ci95_1000 = quantile(eta_1_1000,prob = c(0.025, 1 - 0.025))
plot(density(eta_1_1000), xlim=c(0,1))
abline(v=ci95_1000, col=2)
abline(v = 0.5, col=3)
```



La a posteriori la possiamo valutare in un piano 2D visto che su K componenti di un vettore di probabilità, solo $K-1$ sono variabili aleatorie: il vetttore vive nel simplesso

[77]:
```
# punto 3
par(mfrow=c(1,2))
w_1_10 = apply(pi_sim_10[,c(1,2)],1,sum)
```

```
w_2_10 = apply(pi_sim_10[,c(1,2) +2],1,sum)
smoothScatter(w_1_10, w_2_10, xlim=c(0,1), ylim=c(0,1))

w_1_1000 = apply(pi_sim_1000[,c(1,2)],1,sum)
w_2_1000 = apply(pi_sim_1000[,c(1,2) +2],1,sum)
smoothScatter(w_1_1000, w_2_1000, xlim=c(0,1), ylim=c(0,1))
par(mfrow=c(1,1))
```

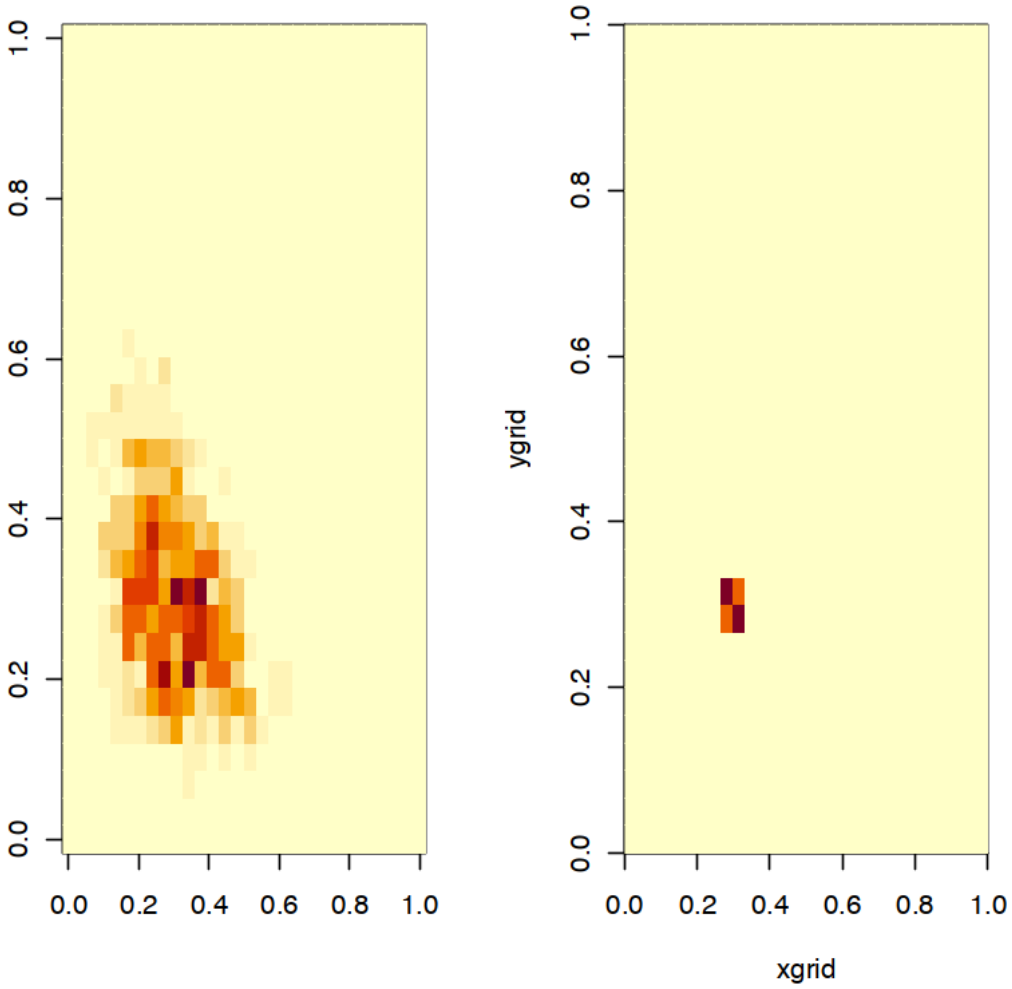


Un'altra soluzione è di creare una griglia nel piano $[0,1]^2$ e stimare la probabilità di cadere in un punto della griglia con Monte Carlo, ricordando che

$$P(w_1 \in \mathcal{A}_1, w_2 \in \mathcal{A}_2) \approx \frac{\sum_{b=1}^{B} 1_{w_1}(\mathcal{A}_1) 1_{w_2}(\mathcal{A}_2)}{B}$$

Poi possimao usare image()

[137]:
```
ngrid = 30
xgrid = seq(0,1, length.out = ngrid+1)
ygrid = seq(0,1, length.out = ngrid+1)

z_mat_10 = matrix(0, nrow = ngrid, ncol=ngrid)
z_mat_1000 = matrix(0, nrow = ngrid, ncol=ngrid)

## calcolo le distanze per
for(irow in 1:ngrid)
{
    set_1_10 = (w_1_10>xgrid[irow])& (w_1_10<xgrid[irow+1] )
    set_1_1000 = (w_1_1000>xgrid[irow])& (w_1_1000<xgrid[irow+1] )
    for(icol in 1:ngrid)
    {
        set_2_10 = (w_2_10>ygrid[icol])& (w_2_10<ygrid[icol+1] )
        set_2_1000 = (w_2_1000>ygrid[icol])& (w_2_1000<ygrid[icol+1] )

        z_mat_10[irow, icol] = sum(set_1_10*set_2_10)
        z_mat_1000[irow, icol] = sum(set_1_1000*set_2_1000)

    }

}

par(mfrow=c(1,2))
image((z_mat_10/nsim))
image((z_mat_1000/nsim), x=xgrid, y = ygrid )
par(mfrow=c(1,1))
```



1.3 Non identificabilità

Abbiamo una regressione

$$y_i = \beta_0 x_i + \beta_1 z_i + \epsilon_i$$

con $i = 1, \dots, n$ e

$$\epsilon_i \sim N(0, 1)$$

e $z_i = 1$ e $x_i = 2$ con $n = 100$ Assumete che le prior siano

$$\beta_0, \beta_1 \sim TN(0, 1000, -10, 10)$$

In questo caso il modello si dice **non identificabile**, perchè per diversi valori dei parametri la distribuzione dei dati rimane la stessa:

$$y_i \sim N(\mu, \sigma^2)$$

con

$$\mu = \beta_0 + 2\beta_2$$

Per esempio, se prendiamo

$$\beta_0' = \beta_0 + 2c$$
$$\beta_1' = \beta_1 - c$$

per ogni valore di c tale per cui $\beta_0', \beta_1' \in [-10, 10]$, allora

$$y_i \sim N(\beta_0 + 2\beta_2, \sigma^2) \equiv N(\beta_0' + 2\beta_2', \sigma^2)$$

Fate attenzione che la non identificabilità si valuta sulla verosimiglianza, perchè una volta che mettiamo la prior, questa cosa non è più vera, ma se la prior sono poco informative, continua a esserlo, almeno approssimativamento.

1. Determinate le a posteriori dei beta e ottenete $B = 1000$ campioni da (β_0, β_1).
2. Valutate la distribuzione marginale delle a posteriori e quella congiunta. Valutate poi la distribuzione di $\beta_0 + 2\beta_1$
3. ripetete il punto 1, ma moltiplicate per 100 i limiti della troncata. Confrontate le marginali a posteriori con la prior

Soluzioni

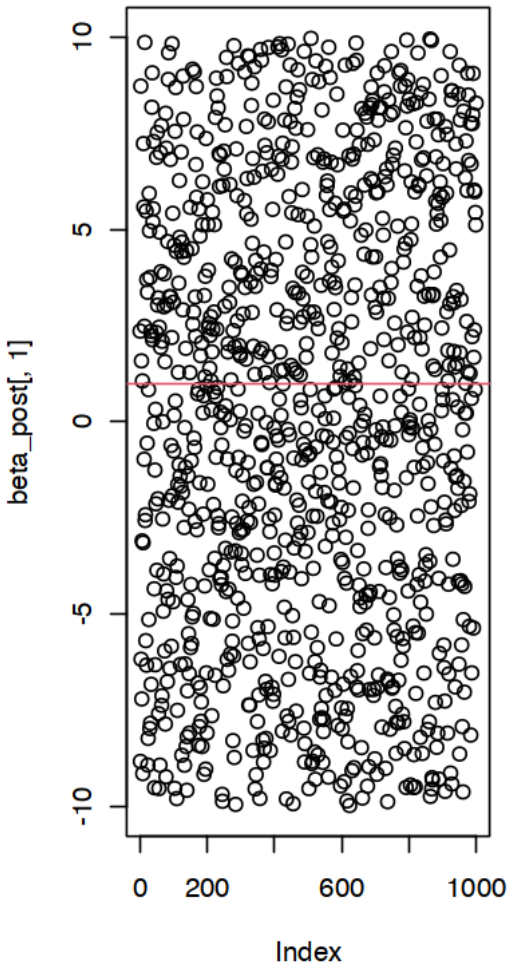
Nel calcolo del primo punto possiamo vedere che il kernel dalla a psoteriori è esattamente lo stesso, come forma funzionale, di quello del primo esercizio. Quindi, la a posteriori è uan distribuzioni troncata in $[-10, 10]^2$. Possiamo quindi ottenere dei campioni simulando dalla distribuzione non troncata e poi tenendo solo quelli nell'iuntervallo $[-10, 10]^2$ (quesot deriva dalla simulazione con accept-reject con kernel, e lo potete vedere nell'esercitazione 3)

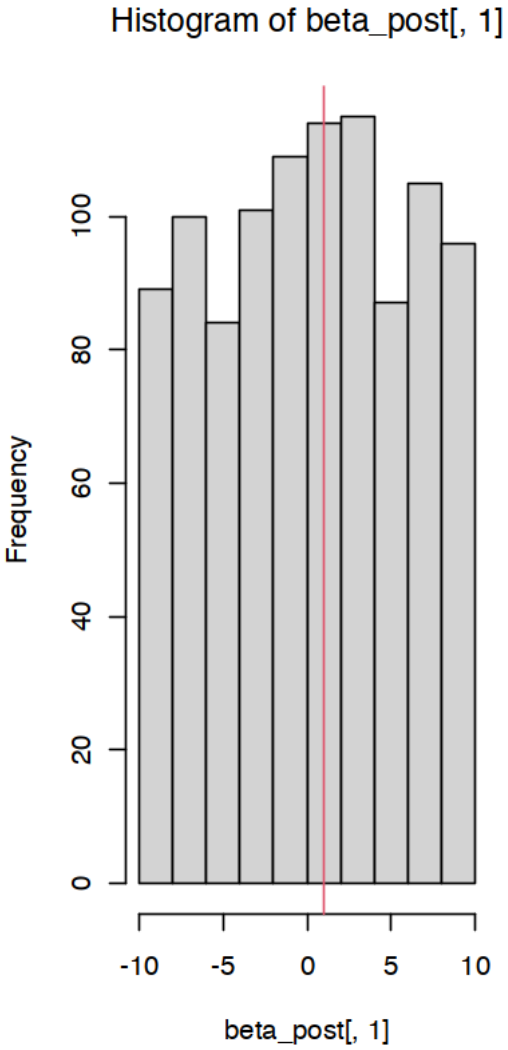
[4]:
```
set.seed(1)
n = 1000
beta0 = 1
beta1 = 0.5
sigma2 = 1
prior_mean = 0
prior_var = 1000
x = rep(2,n)
z = rep(1,n)
epsilon = rnorm(n,0, sigma2^0.5)
y = beta0*z+beta1*x +epsilon

B = 1000
### simulo dalla a posteriori
X = matrix(1, nrow = n, ncol=2)
X[,2] = x
X[,1] = z
# varianza a posteriori
v_p = solve( t(X)%*%solve(diag(sigma2,n))%*%X + solve(diag(prior_var,2)))
# media a posteriori
m_p = v_p%*%(  t(X)%*%solve(diag(sigma2,n))%*%matrix(y, ncol=1) +␣
 ↪solve(diag(prior_var,2))%*%matrix(prior_mean, ncol=1, nrow=2))
```

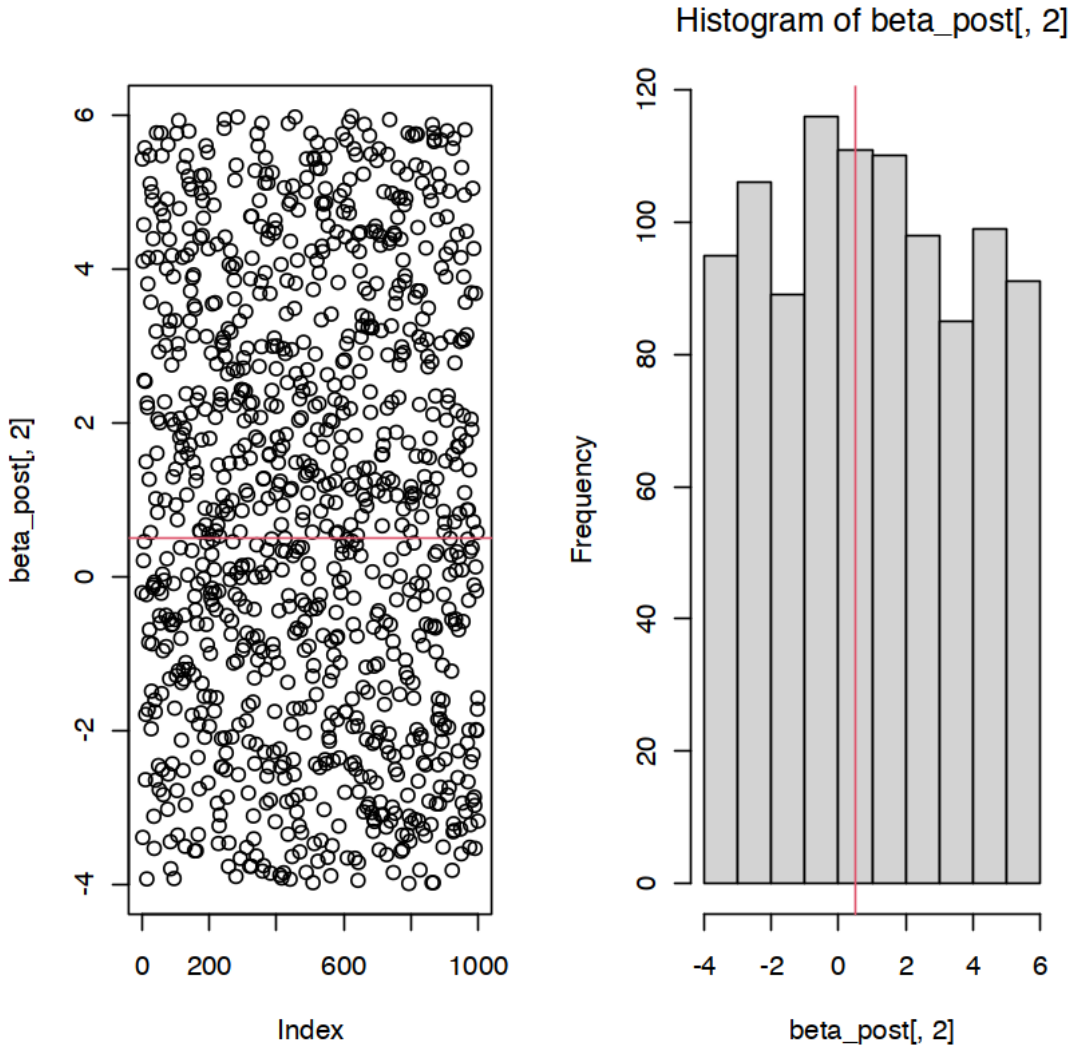
```
v_p_chol = t(chol(v_p))

beta_post = matrix(NA, nrow = B, ncol=2)

for(b in 1:B)
{
    repeat{
        sim = m_p + v_p_chol%*%matrix(rnorm(2, 0,1), ncol=1)
        if((sim[1]> -10) & (sim[1]< 10) & (sim[2]> -10) & (sim[2]< 10))
        {
            break
        }
    }
    beta_post[b,] = sim
}
```

Il punto 2 è relativamente facile. Quello che si vede è che c'è molto incertezza nel valore die parametri, perchè non sono identificabili, mentre siamo molto sicuri sul valore di $E(y_i)$

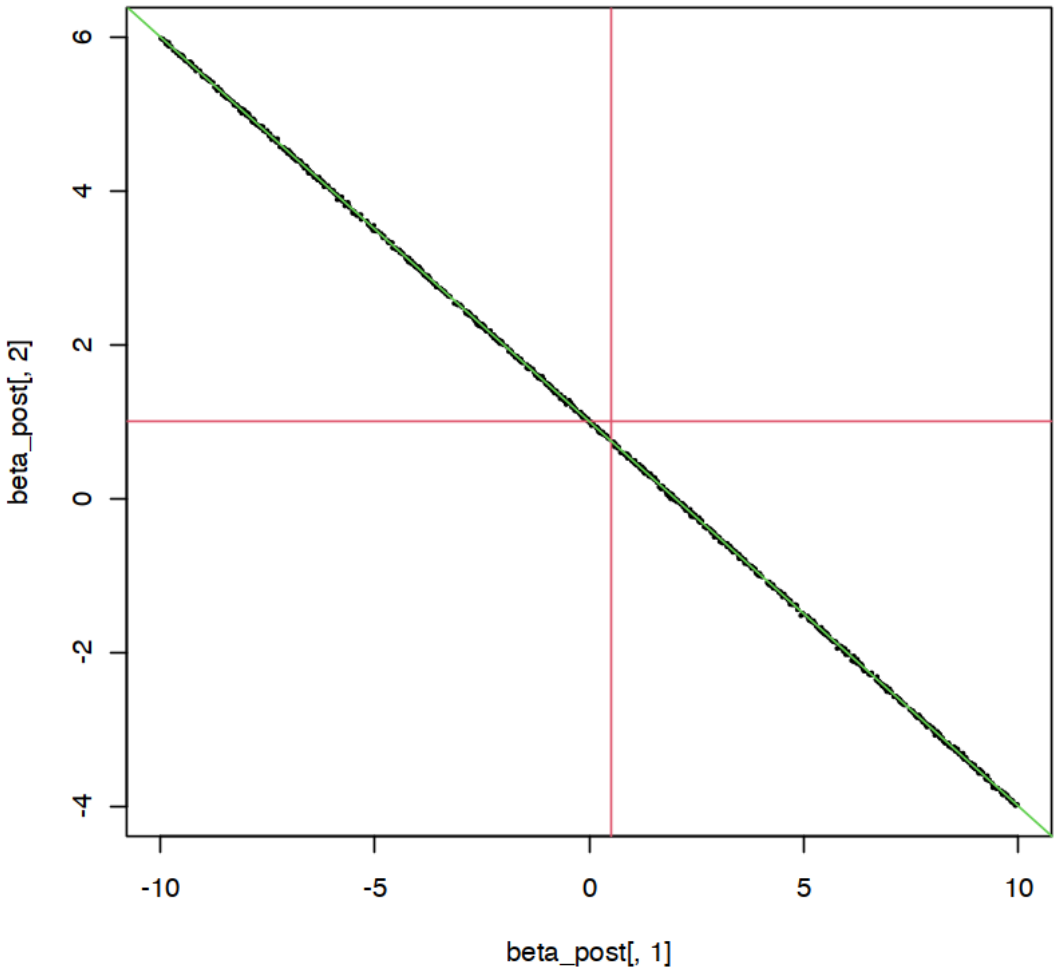
[5]:
```
par(mfrow=c(1,2))
plot(beta_post[,1])
abline(h = beta0, col=2)
hist(beta_post[,1])
abline(v = beta0, col=2)
plot(beta_post[,2])
abline(h = beta1, col=2)
hist(beta_post[,2])
abline(v = beta1, col=2)
```


beta_post[, 1]
10
5
0
-5
-10
0
200
600
1000
Index

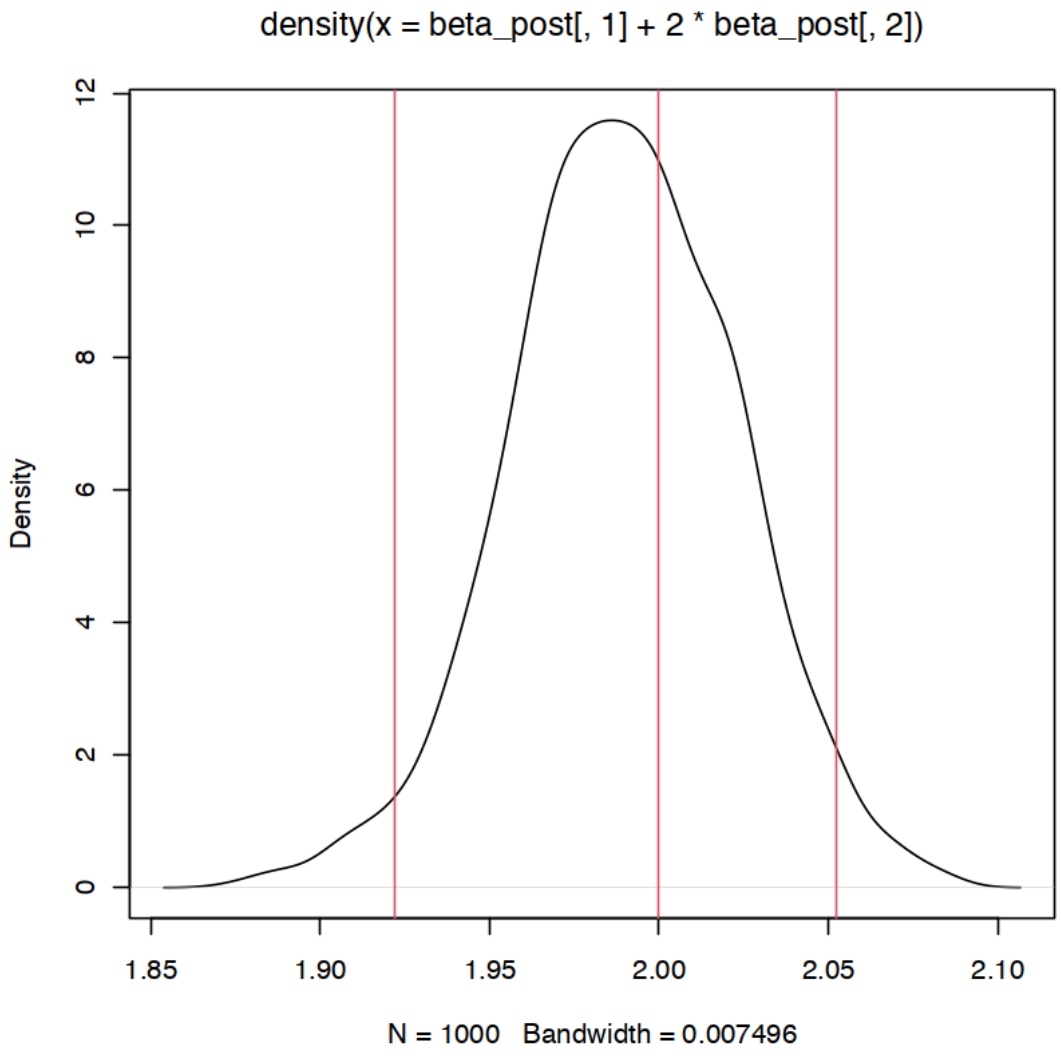

Histogram of beta_post[, 1]
Frequency
100
80
60
40
20
0
-10
-5
0
5
10
beta_post[, 1]



[130]:
```
plot(beta_post[,1], beta_post[,2], pch = 20, cex = 0.2)
abline(h = beta0, col=2)
abline(v = beta1, col=2)
# disegno anche la linea in cui i parametri producono lo stesso valore di mu
c = seq(-10,10, by = 0.01)
lines( beta0 +2*c,  beta1 -c, col=3)
par(mfrow=c(1,1))
```



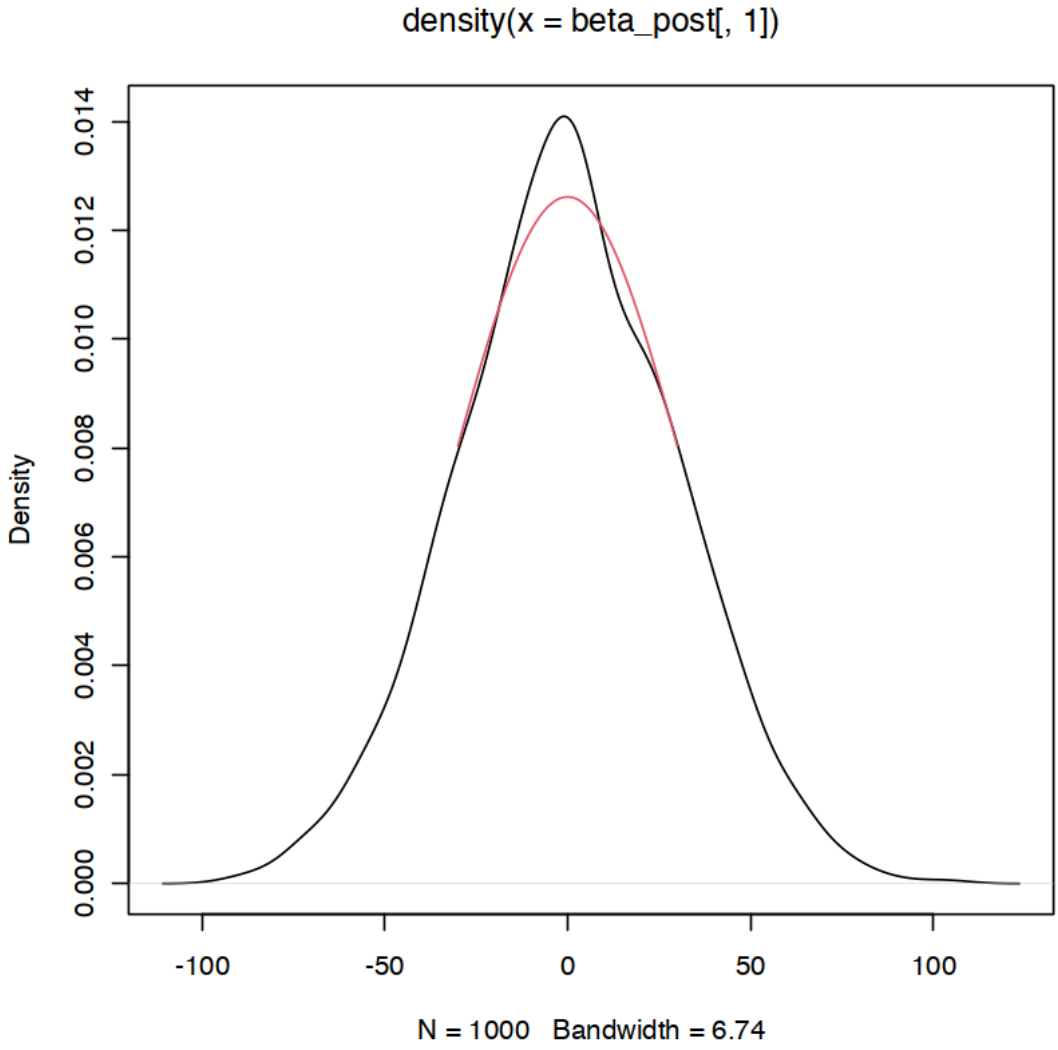
[131]:
```
plot(density(beta_post[,1]+2*beta_post[,2]))
abline(v = beta0 +  2*beta1, col=2)
abline(v = quantile(beta_post[,1]+2*beta_post[,2], prob = c(0.025, 1-0.025)),␣
↪col=2)
```



[132]:
```
# punto 3
set.seed(1)
n = 1000
beta0 = 1
beta1 = 0.5
sigma2 = 1
prior_mean = 0
prior_var = 1000
x = rep(2,n)
z = rep(1,n)
epsilon = rnorm(n,0, sigma2^0.5)
y = beta0*z+beta1*x +epsilon
```

```
B = 1000
### simulo dalla a posteriori
X = matrix(1, nrow = n, ncol=2)
X[,2] = x
X[,1] = z
# varianza a posteriori
v_p = solve( t(X)%*%solve(diag(sigma2,n))%*%X + solve(diag(prior_var,2)))

# media a posteriori
m_p = v_p%*%(  t(X)%*%solve(diag(sigma2,n))%*%matrix(y, ncol=1) +␣
 ↪solve(diag(prior_var,2))%*%matrix(prior_mean, ncol=1, nrow=2))

v_p_chol = t(chol(v_p))

beta_post = matrix(NA, nrow = B, ncol=2)

for(b in 1:B)
{
    repeat{
        sim = m_p + v_p_chol%*%matrix(rnorm(2, 0,1), ncol=1)
        if((sim[1]> -1000) & (sim[1]< 1000) & (sim[2]> -1000) & (sim[2]< 1000))
        {
            break
        }
    }
    beta_post[b,] = sim
}
```

[136]:
```
library(truncnorm)
xseq = seq(-30,30,by = 0.01)
plot(density(beta_post[,1]))
lines(xseq, dtruncnorm(xseq, -1000,1000, 0,1000^0.5), col=2)
```


density(x = beta_post[, 1])
Density
0.000
0.002
0.004
0.006
0.008
0.010
0.012
0.014
-100
-50
0
50
100
N = 1000 Bandwidth = 6.74