Προπτυχιακος Φοιτητης
1-7-2021

# Εθνικο Μετσοβιο Πολυτεχνειο


# Τελική Εργασία Εξαμήνου

*Υπολογιστικές Μέθοδοι στη Στατιστική*

ΝΙΚΟΛΑΟΣ ΤΣΙΠΡΟΣ
n.tsipros@gmail.com
A.M. : $ge15066$
Διδάσκων: Δημητριος Φουσκακης

# 1 Άσκηση 1

Δίνεται το ολοκλήρωμα

$$J = \int_{-\infty}^{+\infty} (x+a)^2 \phi(x) dx = 1 + a^2 \tag{1}$$

όπου, $\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$, η σ.π.π της τυποποιημένης κανονικής κατανομής.

***α.*** Το παραπάνω ολοκλήρωμα θα εκτιμηθεί με τη βοήθεια της Monte Carlo ολοκλήρωσης. Πιο συγκεκριμένα, παρατηρούμε οτι το ολοκλήρωμα της σχέσης (1) πρόκειται για τη μέση τιμή $\mathbb{E}_\phi[(x+a)^2]$, εφόσον η $\phi(x)$ είναι σ.π.π. Προσομοιώνουμε ένα τυχαίο δείγμα $X_1, X_2, ..., X_n$ από τη σ.π.π

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

με τη βοήθεια της εντολής *rnorm* στην R και εκτιμούμε το ολοκλήρωμα (1) ως:

$$\hat{J} = \frac{1}{n} \sum_{i=1}^{n} (x_i + a)^2 \tag{2}$$

Παρακάτω βρίσκεται η συνάρτηση στην $R$, η οποία δέχεται ένα διάνυσμα με τιμές απο τη τυποποιημένη κανονική κατανομή, καθώς και τη μεταβλητή $a$.

```
   MC <- function(SAMPLE,a){
 n <- length(SAMPLE)
 i <- 1:n
 result <- sum((SAMPLE[i] + a)^2)
 return(result/n)
}
```

Αποθηκεύουμε τα αποτέλεσματα για $a = 0, 1, 2, 3, 4$ και $n = 100, 1000$ σε ενα dataframe και τα τυπώνουμε:

```
   res = data.frame(n = c(100,1000), a0 = c(MC(rnorm(100), 0),
   MC(rnorm(1000), 0)), a1 = c(MC(rnorm(100),
   1),MC(rnorm(1000), 1)), a2 = c(MC(rnorm(100), 2),
   MC(rnorm(1000), 2)), a3 = c(MC(rnorm(100),
   3),MC(rnorm(1000), 3)), a4 = c(MC(rnorm(100), 4),
   MC(rnorm(1000), 4)))

  knitr :: kable(res)
```

| n | a0 | a1 | a2 | a3 | a4 |
|---|---|---|---|---|---|
| 100 | 1.1686846 | 1.558177 | 4.642401 | 9.629946 | 16.76762 |
| 1000 | 0.9442027 | 2.034971 | 5.114469 | 9.890107 | 16.85328 |

Παρατηρούμε οτι για $n = 1000$ η εκτίμησή μας πλησιάζει περισσότερο τη θεωρητική, ενώ όσο το $a$ αυξάνεται, οι προσεγγίσεις μας δεν φαίνονται να είναι τοσο ακριβείς.

***β.***

$$\mathbb{E}[\hat{J}] = \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n}(x_i+a)^2]$$

$$= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[(x_i+a)^2] = \frac{1}{n}n\,\mathbb{E}[(x+a)^2] = \int_{-\infty}^{\infty}(x+a)^2\phi(x)dx \tag{3}$$

Άρα απο τη σχέση (3) αποδείχθηκε οτι ο Monte Carlo εκτιμητής είναι αμερόληπτος, αφού $\mathbb{E}[\hat{J}] = J$.

Για τον υπολογισμό της διασποράς:

$$Var[\hat{J}] = Var[\frac{1}{n}\sum_{i=1}^{n}(x_i+a)^2] = \frac{1}{n^2}\sum_{i=1}^{n}Var[(x_i+a)^2]$$

$$= \frac{1}{n}Var[(x+a)^2] = \frac{1}{n}\int_{-\infty}^{+\infty}[(x+a)^2 - \mathbb{E}[(x+a)^2]]^2\phi(x)dx$$

$$= \frac{1}{n}\int_{-\infty}^{+\infty}[(x+a)^2 - (1+a^2)]^2\phi(x)dx = \frac{4a^2+2}{n} \tag{4}$$

Οπότε η τυπική απόκλιση της Monte Carlo ολοκλήρωσης είναι
$SD(\hat{J}) = \sqrt{\frac{4a^2+2}{n}}$ .

***γ.*** Σκοπός μας είναι και πάλι να εκτιμήσουμε το ολοκλήρωμα
$J = \int_{-\infty}^{+\infty}(x+a)^2\phi(x)dx$, αυτή τη φορά όμως με δειγματοληψία σπουδαιότητας. Σύμφωνα με τη μέθοδο αυτή, αρκεί να ορίσουμε μια νέα σ.π.π., έστω $g(x)$ και να θεωρήσουμε το ολοκλήρωμα:

$$\int_{-\infty}^{+\infty}\frac{(x+a)^2\phi(x)}{g(x)}g(x)dx$$

Παράγουμε δείγμα $X_1, X_2, ..., X_n$ απο τη σ.π.π $g(x)$, και θέτωντας $Y(x) = \frac{(x+a)^2\phi(x)}{g(x)}$, έχουμε ότι:

$$\int_{-\infty}^{+\infty}Y(x)g(x)dx = \mathop{\mathbb{E}}_{g}[Y(x)] \tag{5}$$

Άρα μπορούμε να εκτιμήσουμε το (5) ως:

$$\hat{J} = \frac{1}{n}\sum_{i=1}^{n}Y(x_i) \tag{6}$$

Δίνεται $g(x) = \phi(x-a) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-a)^2}{2}}$. Είναι φανερό πως πρόκειται για τη σ.π.π της κανονικής κατανομής $N(a,1)$.

Ορίζουμε στην $R$ τη συνάρτηση $ImpSamp$ η οποία δέχεται το μέγεθος του δείγματος που θα προσομοιωθεί απο την $N(a,1)$, καθώς και η τιμή της $a$.

```
  ImpSamp <- function(n, a){
SAMPLE <- rnorm(n,mean = a, sd = 1)
i <- 1:n
result <- sum(((SAMPLE[i]+a)^2)*exp((-2*a*SAMPLE[i] + a^2)/2))
return(result/n)
}
```

Τυπώνουμε τα αποτελέσματα για $n = 100$, $a = 0, 1, 2, 3, 4$ με τη βοήθεια ενός $dataframe$:

```
resImpSamp = data.frame(n = c(100,1000), a0 = c(ImpSamp(100,0)
, ImpSamp(1000,0)), a1 = c(ImpSamp(100,1), ImpSamp(1000,1)),
a2 = c(ImpSamp(100,2), ImpSamp(1000,2)), a3 = c(ImpSamp(100,3),
ImpSamp(1000,3)), a4 = c(ImpSamp(100,4), ImpSamp(1000,4)))

knitr :: kable(resImpSamp)
```

| n | a0 | a1 | a2 | a3 | a4 |
|---|---|---|---|---|---|
| 100 | 0.9269706 | 1.919765 | 5.155897 | 82.793946 | 3.383315 |
| 1000 | 1.0359660 | 1.999674 | 4.778845 | 7.661859 | 12.013046 |

Παρατηρούμε καλές εκτιμήσεις όταν το $a$ είναι μικρό, και το $n = 1000$, αλλά για $a = 3, 4$ φαίνεται να υπάρχει μεγάλη απόκλιση της εκτίμησης απο τη θεωρητική τιμή, αλλά και μεταξύ των εκτιμήσεων για διαφορετικά $n$. Αυτό περιμένουμε να επιβεβαιωθεί απο τη διασπορά.
Για να αποδειχθεί οτι ο εκτιμητής είναι αμερόληπτος θα πρέπει $\mathbb{E}[\hat{J}] = J$.

$$\mathbb{E}[\frac{1}{n}\sum_{i=1}^{n} Y(x_i)] = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[Y(x_i)] = \mathop{\mathbb{E}}_{g}[Y(x)]$$

$$\int_{-\infty}^{+\infty} \frac{\phi(x)f(x)}{g(x)} g(x)dx = \int_{-\infty}^{+\infty} \phi(x)f(x)dx = J \tag{7}$$

Για τη διασπορά:

$$Var[\frac{1}{n}\sum_{i=1}^{n} Y(x_i)] = \frac{1}{n^2}\sum_{i=1}^{n} Var[Y(x_i)] = \frac{1}{n} Var[Y(x)] = \frac{1}{n} Var[\frac{(x+a)^2 * (2\pi)^{-\frac{1}{2}} e^{\frac{-x^2}{2}}}{(2\pi)^{-\frac{1}{2}} e^{-\frac{(x-a)^2}{2}}}]$$

$$= \frac{1}{n} e^{a^2} Var[(x+a)^2 e^{-ax}] = \frac{1}{n} e^{a^2} (\mathop{\mathbb{E}}_{g}[(x+a)^4 e^{-2ax}] - \mathop{\mathbb{E}}_{g}^{2}[(x+a)^2 e^{-ax}])$$

$$= \frac{1}{n}(3e^{a^2} - (a^2-1)e^{\frac{a^2}{2}}) \tag{8}$$

Άρα η διασπορά της δειγματοληψίας σπουδαιότητας γίνεται πολύ μεγαλύτερη καθώς το $a$ μεγαλώνει, σε σχέση με την απλή Monte Carlo εκτίμηση.

**δ.** Για $a = 4$ το ολοκλήρωμα της σχέσης (1) γίνεται:

$$J = \int_{-\infty}^{+\infty} (x+4)^2 \phi(x) dx = 17$$

Για να εκτιμήσουμε το τυπικό σφάλμα του εκτιμητή Monte Carlo με τη τεχνική Bootstrap, θα χρειαστεί να δημιουργήσουμε ενα διάνυσμα '$Jstar$' στην $R$ μεγέθους $B$ το οποίο θα αποτελείται απο εκτιμήσεις του ολοκληρώματος (1). Για τις εκτιμήσεις του ολοκληρώματος θα προσομοιώσουμε, αρχικά, 1000 τιμές απο την τυποποιημένη κανονική κατανομή, τις οποίες θα αποθηκεύσουμε σε ένα διάνυσμα $SAMPLE = [X_1, X_2, ..., X_{1000}]$. Στη συνέχεια για κάθε μια απο τις $B$ επαναλήψεις, θα επιλέγουμε ένα τυχαίο δείγμα $BSAMPLE$ μεγέθους 1000, με επανάθεση, τιμές απο το αρχικό μας δείγμα $SAMPLE$. Η κάθε θέση του $Jstar$ υπολογίζεται με τη βοήθεια του εκτιμητή Monte Carlo

$$\hat{J}_i^* = \frac{1}{1000} \sum_{i=j}^{1000} (BSAMPLE_j + 4)^2$$

Η τυπική απόκλιση υπολογίζεται απο τον τύπο:

$$se(\hat{J}) = \sqrt{\frac{1}{B-1} \sum_{i=1}^{B} [\hat{J}_i^* - \tilde{\hat{J}}^*]} \tag{9}$$

όπου, $\tilde{\hat{J}}^* = \frac{1}{B} \sum_{i=1}^{B} \hat{J}_i^*$.

Ακολουθεί η συνάρτηση στην $R$, η οποία δέχεται ενα διάνυσμα απο τιμές της τυποποιημένης κανονικής κατανομής και τον αριθμό των ζητούμενων δειγμάτων Bootstrap.

```
  BTS_SE <- function(SAMPLE,B){
n <- length(SAMPLE)
Jstar <- c(rep(0,B))
for (i in 1:B){
  BSAMPLE <- SAMPLE[sample(n, replace = T, prob = c(rep(1/n,n)))]
  Jstar[i] <- sum((BSAMPLE + 4)^2)/n
}
Jbar <- (1/B)*sum(Jstar)
i <- 1:B
```

```
  se <- ((1/(B-1))*sum((Jstar[i] - Jbar)^2))^(1/2)
  print(se)
}
```

Για $B = 100$ και 1000 επαναλήψεις Bootstrap, η συνάρτηση επιστρέφει:

```
    BTS_SE(rnorm(1000),100)

## [1] 0.2665895

    BTS_SE(rnorm(1000),1000)

## [1] 0.248538
```

Παρατηρούμε ότι η εκτίμηση για το σφάλμα είναι πολύ κοντά στη θεωρητική τίμη που υπολογίστηκε το ερώτημα β: $SD(\hat{J}) = \sqrt{\frac{4*4^2+2}{1000}} = 0.256904$.

# 2 Άσκηση 2

Θεωρούμε τη συνάρτηση:

$$f(x) = \frac{1}{e^3 - 1} e^x, x \, \epsilon \, [0, 3] \tag{10}$$

***α.*** Θα χρειαστούμε τη συνάρτηση κατανομής της σ.π.π (10):

$$F(x) = \int_0^x \frac{1}{e^3 - 1} e^x dx = \frac{e^x - 1}{e^3 - 1} \tag{11}$$

Σύμφωνα με τη μέθοδο αντιστροφής, αν η συνάρτηση κατανομής είναι αντιστρέψιμη, τότε μπορούμε να πάρουμε τυχαίο δείγμα απο την $F$ ως εξής: Αν $U_i$ τυχαίο δείγμα απο την $U[0, 1]$, τότε τα $X_i = F^{-1}(U_i)$ αποτελούν τυχαίο δείγμα απο την $F$. Παρατηρούμε οτι η συνάρτηση κατανομής (5) είναι γνησίως αύξουσα, αφού $F^{'}(x) > 0$, σε όλο το πεδίο ορισμού της, άρα είναι και αντιστρέψιμη. Ορίζουμε λοιπόν την αντίστροφή της:

$$F^{-1}(u) = ln[u(e^3 - 1) + 1]. \tag{12}$$

Ορίζουμε στην $R$ την αντίστροφη συνάρτηση και παίρνουμε τυχαίο δείγμα απο την ομοιόμορφη κατανομή στο διάστημα $[0, 1]$ με τη βοήθεια της εντολής $runif$:

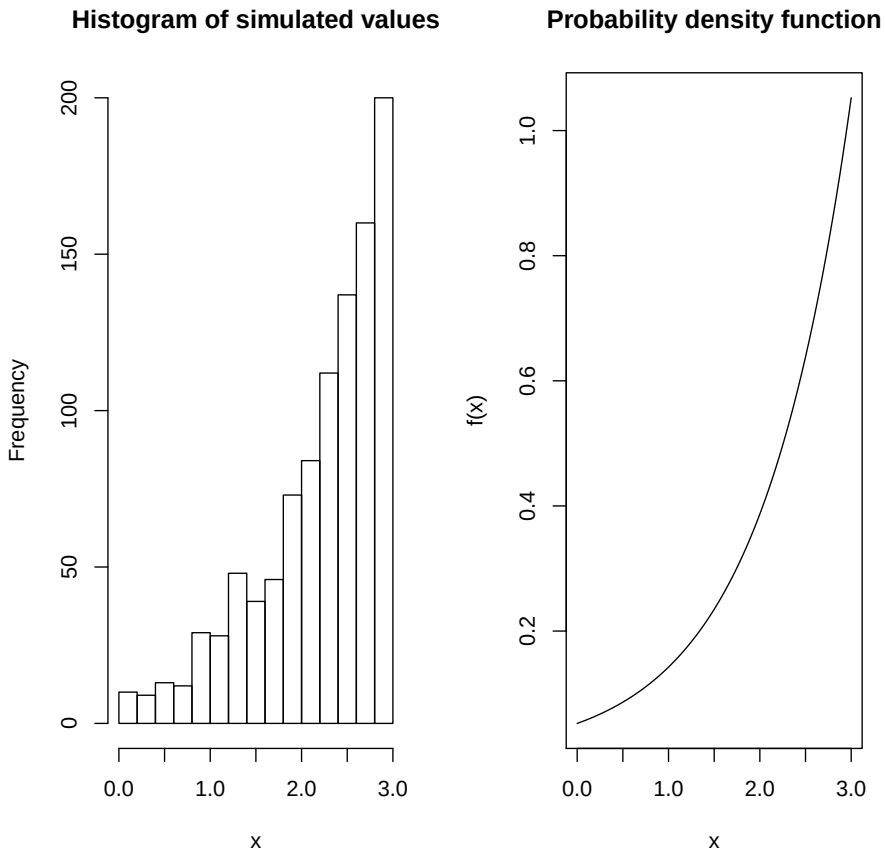
```
invF <- function(u) {log(u*(exp(3)-1)+1)}
u <- runif(1000)
```

Τυπώνουμε ενδεικτικά τις πρώτες και τις τελευταίες 5 τιμές, και φτιάχνουμε το ιστόγραμμα των προσομοιωμένων τιμών, καθώς και το διάγραμμα της $f$:

```r
x <- invF(u)
head(x, 5)

## [1] 2.237033 2.458246 1.272946 2.868708 2.107912

tail(x,5)

## [1] 2.356705 2.068797 2.734270 2.644829 2.521478
```

Ακολουθούν το διάγραμμα της $f$ και το ιστόγραμμα των προσομοιωμένων τιμών:

```r
par(mfrow=c(1,2))
hist(x, main = "Histogram of simulated values")
x <- seq(0,3,length = 1000)
f <- function(x) {exp(x)/(exp(3)-1)}
 plot(x, f(x), type = 'l', main = "Probability density function")
```



Το ιστόγραμμα των προσομοιωμένων τιμών βλέπουμε να συμπεριφέρεται όπως θα περιμέναμε, με βάση το διάγραμμα της σ.π.π. που μας δώθηκε.

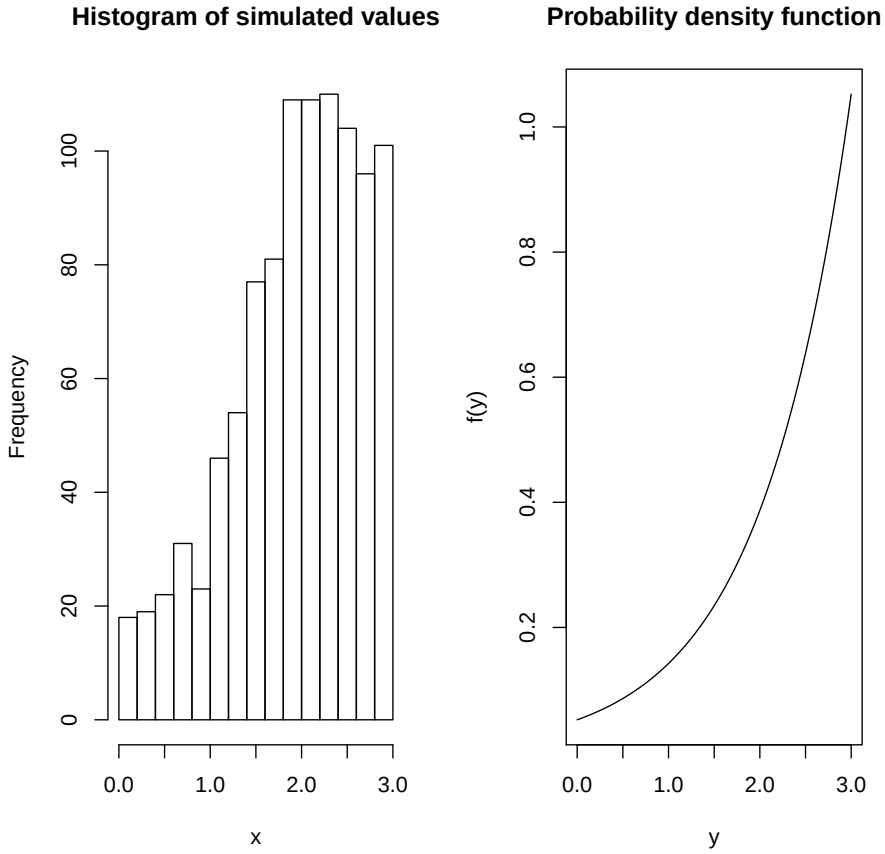
***β.*** Για να παράξουμε τιμές απο τη σ.π.π (10) με τη μέθοδο απόρριψης, θα χρειαστεί να ορίσουμε ένα αριθμό $M$ και μια νέα σ.π.π, $g(x)$, ώστε να δημιουργήσουμε ένα ¨φάκελο' γύρω απο την $f$ : $f \leq Mg = G$.

Παρατηρώντας οτι η $f$ είναι γνησίως αύξουσα, μπορούμε να βρούμε εύκολα το μέγιστο της: $f(3) = \frac{e^3}{e^3-1}$. Ορίζουμε τη $g(x) = \frac{1}{3}, x \epsilon [0,3]$ ως την ομοιόμορφη στο διάστημα $[0,3]$, και $M = \frac{e^3}{e^3-1}$.
Στην $R$ δημιουργούμε έναν αλγόριθμο ο οποίος σε κάθε επανάληψη θα παράγει ενα $y \sim U[0,3]$, και ένα $u \sim U[0,1]$. Στη συνέχεια θα ελέγχει άν $u \leq \frac{f(y)}{Mg(y)}$, οπότε και θα αποθηκεύει τη τιμή αυτή του $y$ σε ένα διάνυσμα $X$ με τις ζητούμενες τιμές απο τη κατανομή (10). Αν ο παραπάνω έλεγχος δεν ισχύει η μέθοδος επαναλαμβάνεται, μέχρι να συμπληρωθούν οι απαιτούμενες θέσεις στο διάνυσμα $X$.
Ακολουθεί η συνάρτηση στην $R$ που τυπώνει $n$ προσομοιωμένες τιμές απο τη σ.π.π (10):

```
   RejS <- function(n){
 M <- exp(3)/(exp(3)-1)
 X <- c(rep(0,n))
 i <- 1
 while (i < n) {
   y <- runif(1, min = 0, max = 3)
   u <- runif(1)
   if (u <= (3*exp(y))/exp(3)){
     X[i] <- y
     i <- i + 1
   }
 }
 return(X)
}
```

Για $n = 1000$:

```
x = RejS(1000)
head(x, 5)

## [1] 2.4922051 2.9214028 0.7769112 2.1495573 2.1326606

tail(x, 5)

## [1] 1.850475 2.657633 1.063884 2.113010 0.000000
```




Αυτή τη φορά το ιστόγραμμα των προσομοιωμένων τιμών, δεν φαίνεται να περιγράφει όσο καλά όσο η μέθοδος αντιστροφής, τη σ.π.π. που μας δώθηκε.

***γ.*** Σκοπός μας είναι να εκτιμήσουμε την σ.π.π $f(x)$ γνωρίζοντας ένα δείγμα μεγέθους 100 που προήλθε απο αυτή. Θα χρησιμοποιηθεί ο πυρήνας Epanechnikov, οπότε η $f$ θα εκτιμηθεί απο τον τύπο:

$$\hat{f} = \frac{1}{nh}\sum_{i=1}^{n}\frac{3}{4}(1-(\frac{x-x_i}{h})^2). \tag{13}$$

Για τον υπολογισμό του βέλτιστου $h_{optim}$, θα χρησιμοποιήσουμε τη μέθοδο Leave-one-out Cross Validation. Σε κάθε επανάληψη αφήνουμε μια παρατήρηση απ΄έξω, και αποθηκεύουμε τη τιμή του νέου μοντέλου

$$\hat{f}_{h,-i}(x_i) = \frac{1}{h(n-1)}\sum_{j\neq i}^{n}\frac{3}{4}(1-(\frac{x-x_i}{h})^2).$$

σε ένα διάνυσμα. Ειδικότερα, θα προσπαθήσουμε να βρούμε το $h$ που μεγιστοποιεί την πιθανοφάνεια:

$$L(h,i) = \prod_{i=1}^{n} \hat{f}_{h,-i}(x_i) \tag{14}$$

Για υπολογιστική ευκολία μπορούμε να λογαριθμήσουμε την εξίσωση (14), μιας και η συνάρτηση $ln(x)$ είναι γνησίως αύξουσα.
Ακολουθεί η συνάρτηση στην $R$ που επιστρέφει για δοσμένο $h$ τη λογαριθμισμένη σχέση (14):

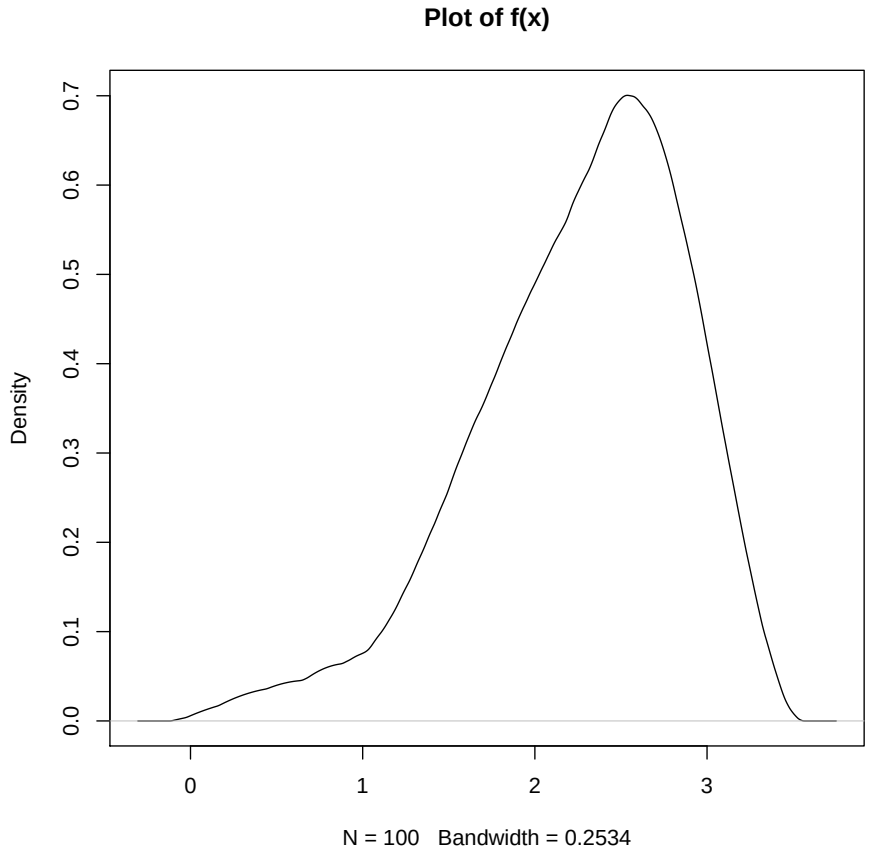
```
Cross <- function(h){
  invF <- function(u) {log(u*(exp(3)-1)+1)}
  set.seed(100)
  x <- invF(runif(100))
  f <- c(rep(0,100))
  sum <- 0
  for (i in 1:100){
    for (j in 1:100){
      if ( j != i){
        if (abs((x[i]-x[j])/h) < 1){
          sum <- sum + (1 - ((x[i] - x[j])/h)^2)
        }
      }
    }
    f[i] <- (3/(396*h))*sum
  }
  return(log(prod(f)))
}
```

Στη συνέχεια, θα μεγιστοποιήσουμε την συνάρτηση $Cross$ με χρήση της εντολής $optim$, η οποία δέχεται μια αρχική εκτίμηση για το $h$, τη συνάρτηση προς μεγιστοποίηση και τη προτιμούμενη μέθοδο μεγιστοποίησης. Θα δώσουμε στην $optim$ την εντολή $control = list(fnscale = -1)$, έτσι ώστε να μεγιστοποιήσει, και όχι να ελαχιστοποιήσει.

```
hoptim <- optim(par = 0.1, fn = Cross, method = "BFGS",
control = list(fnscale = -1))
hoptim[1]

## $par
## [1] 0.2534821
```

Ακολουθεί το διάγραμμα της $f(x)$ με $h_{optim} = 0.2534$, απο 100 προσομοιωμένες τιμές, χρησιμοποιώντας τον πυρήνα Epanechnikov.

```r
invF <- function(u) {log(u*(exp(3)-1)+1)}
set.seed(100)
val <- invF(runif(100))
plot(density(val, kernel="epanechnikov", bw = 0.2534),
main = "Plot of f(x)")
```




Απο το παραπάνω σχήμα φαίνεται να έχει γίνει μια σχετικά καλή και λεία προσέγγιση για την $f(x)$.

**δ.** Για τον έλεγχο της μηδενικής υπόθεσης $H_0 : \mu = 2$ έναντι της εναλλακτικής $H_1 : \mu \neq 2$, θα ορίσουμε την ελεγχοσυνάρτηση

$$T = |\bar{x} - 2|. \tag{15}$$

Ζητείται ο ελέγχος να γίνει με τη μέθοδο Bootstrap, μιας και το μέγεθος του δείγματος είναι πολυ μικρό για να εφαρμοστεί το Κ.Ο.Θ. Στο αρχικό μας δείγμα θα προσθέσουμε την τιμή $T$, ώστε σε κάθε ένα από τα $B$ δείγματα Bootstrap που θα δημιουργήσουμε, θα ισχύει η υπόθεση $H_0$. Για κάθε Bootstrap δείγμα θα

υπολογίζουμε το $T_i^* = |\bar{x}_i - 2|$ και θα το αποθηκεύουμε. Είναι προφανές πως ο έλεγχος ορίζεται ως:

$$p_{value} = \frac{\sum_{i=1}^{n} \mathbb{1}(T_i^* > T) + 1}{B + 1} \tag{16}$$

Ακολουθεί ο κώδικας στην $R$:

```
HypTestB <- function(B){
  invF <- function(x){log(u*(exp(3)-1)+1)}
  u <- runif(10)
  SAMPLE <- invF(u)
  Sbar <- mean(SAMPLE)
  Tstar <- c(rep(0,B))
  SAMPLE <- SAMPLE + abs(Sbar - 2)
  for (i in 1:B){
    y <- sample(10, replace = TRUE, prob = c(rep(1/10,10)))
    BSAMPLE <- SAMPLE[y]
    Tstar[i] <- abs(mean(BSAMPLE) - 2)
  }
  flag <- 0
  for (j in 1:B){
    if (Tstar[j] > abs(Sbar - 2)){flag <- flag + 1}
  }
  p <- (flag + 1)/(B+1)
  print(p)
}
```

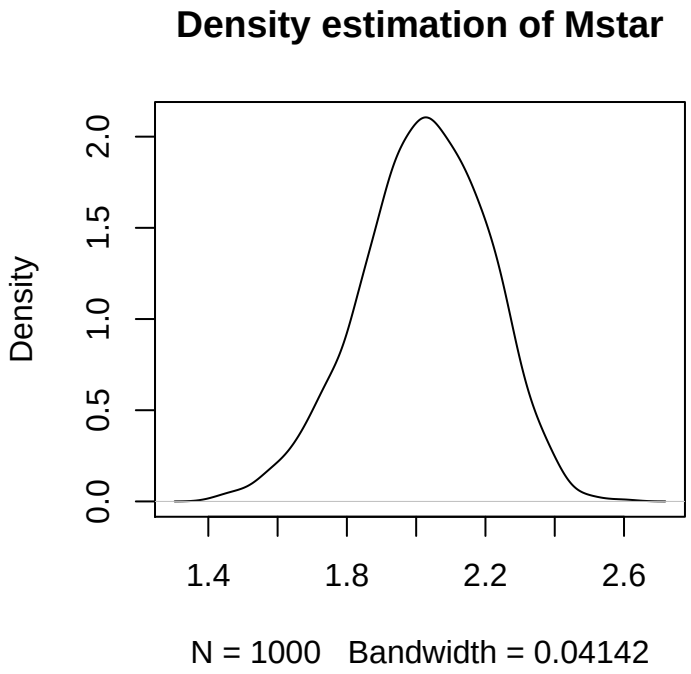
Για $B = 100$ δείγματα Bootstrap:

```
set.seed(1)
HypTestB(100)

## [1] 0.9009901
```

Παρατηρούμε οτι ο έλεγχος παίρνει μεγάλη τιμή, οπότε αποδεχόμαστε την μηδενική υπόθεση $H_0 : \mu = 2$

Επιπρόσθετα, θα κάνουμε τον έλεγχο υποθέσεων, δημιουργώντας και ένα 95% Bootstrap διάστημα εμπιστοσύνης. Ειδικότερα, με την μέθοδο της αντιστροφής θα προσομοιώσουμε ενα δείγμα μεγέθους 10 απο την $f(x)$, θα δημιουργήσουμε 1000 Bootstrap δείγματα με βάση το αρχικό, και απο το κάθε ένα θα αποθηκεύουμε σε ένα διάνυσμα $Mstar$ το δειγματικό μέσο του κάθε Bootstrap δείγματος. Με χρήση της εντολής $plot(density(Mstar))$ παρατηρούμε οτι η κατανομή των Bootstrap εκτιμητών είναι συμμετρική. Άρα θα διατάξουμε το διάνυσμα σε αύξουσα σειρά και παίρνοντας τα $a/2$ και $1 - a/2$ ποσοστιαία σημεία του διανύσματος, βρίσκουμε ενα 95% Δ.Ε.

```r
invF <- function(x){log(u*(exp(3)-1)+1)}
set.seed(100)
u <- runif(10)
x <- invF(u)
Mstar <- c(rep(0,1000))
for (i in 1:1000){
  s <- sample(10, replace = TRUE, prob = c(rep(1/10,10)))
  boot <- x[s]
  Mstar[i] <- mean(boot)
}
```




```r
Mstar <- sort(Mstar, decreasing = FALSE)
 c1 <- Mstar[50]
 c2 <- Mstar[950]
 print(c(c1, c2))

## [1] 1.702131 2.304871
```

Άρα το 95% Δ.Ε. μας διαμορφώνεται απο τις [Mstar(50), Mstar(950)]: [1.7021, 2.3049], το οποίο περιέχει το 2, οπότε και αποδεχόμαστε την $H_0$.

# 3 Άσκηση 3

***α.*** Θεωρούμε την κατανομή Γάμμα με σ.π.π

$$f(x) = \frac{\beta^a}{\Gamma(a)} x^{a-1} e^{-\beta x}, \quad x \in (0, +\infty) \quad a, \beta > 0 \tag{17}$$

Το στήριγμα της Γάμμα $S : x \in (0, +\infty)$, είναι ανεξάρτητο απο τις παραμέτρους $a, \beta$ και η (17) μπορεί να έρθει στη μορφή:

$$f(x|\vec{\theta}) = p(x)g(\vec{\theta})e^{\sum_{i=1}^{k} c_i(\vec{\theta})d_i(x)}$$

ως:

$$f(x|(a,\beta)) = \frac{1}{x}\frac{\beta^a}{\Gamma(a)} e^{a ln x - \beta x}$$

οπότε η Γάμμα ανήκει στην εκθετική οικογένεια κατανομών. Απο θεώρημα Pitman-Koopam υπάρχει επαρκής δειγματοσυνάρτηση της παραμέτρου $\vec{\theta} = (a, \beta)$,
$t_j = \sum_{i=1}^{n} d_j(x_i)$.
Για τυχαίο δείγμα $x = (x_1, ..., x_n)^T$:

$$\prod_{i=1}^{n} f(x_i, (a,\beta)) = (\frac{\beta^a}{\Gamma(a)})^n e^{a\sum ln(x_i) - \beta \sum x_i} \prod_{i=1}^{n} \frac{1}{x_i}$$

Άρα η επαρκής στατιστική συνάρτηση για τις παραμέτρους $a, \beta$ θα έχει διάσταση 2 και είναι η:

$$t = (\sum_{i=1}^{n} ln(x_i), \sum_{i=1}^{n} x_i). \tag{18}$$

Η λογαριθμική πιθανοφάνεια:

$$l(a,\beta) = ln(\prod_{i=1}^{n} f(x_i, (a,\beta)))$$

$$= anln(\beta) - nln[\Gamma(a)] + (a-1)\sum_{i=1}^{n} ln(x_i) - \beta \sum_{i=1}^{n} x_i \tag{19}$$

Για να την μεγιστοποιήσουμε βρίσκουμε που μηδενίζονται οι μερικές παράγωγοι της (19).

$$\frac{\partial l}{\partial \beta} = 0 \Rightarrow \beta = \frac{a}{\bar{x}}$$

$$\frac{\partial l}{\partial a} = 0 \Rightarrow nln\beta - n(ln\Gamma(a))' + \sum_{i=1}^{n}(x_i) = 0 \tag{20}$$

Για την επίλυση της (20) θα χρησιμοποιήσουμε τη μέθοδο Newton-Raphson. Θέτουμε $g(a) = nln\beta - n(ln\Gamma(a))' + \sum_{i=1}^{n}(x_i)$, και $\Psi(a) = (ln\Gamma(a))'$.
Ο αλγόριθμος Newton-Raphson, δεδομένου κάποιας αρχικής προσέγγισης $a_0$ διαμορφώνεται ως:

$$a_{n+1} = a_n - \frac{g(a_n)}{g'(a_n)}.$$

***β.*** Έστω η τ.μ. $X|\theta \sim Poisson(\theta)$, και $\theta \sim Gamma(a,\beta)$, άρα πρόκειται για μια κατανομή Poisson με τον ρυθμό της να ακολουθεί την Γαμμα(α,β).

$$f(x) = \int_0^{+\infty} \mathbb{P}[X = x|\Theta = \theta] * Gamma(a,\beta)d\theta = \int_0^{+\infty} \frac{\theta^x e^{-\theta}}{x!}\frac{\beta^a}{\Gamma(a)}\theta^{(a-1)}e^{-\beta\theta}d\theta$$

$$= \frac{\beta^a}{x!\Gamma(a)}\frac{\Gamma(x+a)}{(\beta+1)^{x+a}}\int_0^{+\infty}\frac{(\beta+1)^{x+a}}{\Gamma(x+a)}\theta^{x+a-1}e^{-\theta(\beta+1)}d\theta$$

Όμως, το ολοκλήρωμα:

$$\int_0^{+\infty}\frac{(\beta+1)^{x+a}}{\Gamma(x+a)}\theta^{x+a-1}e^{-\theta(\beta+1)}d\theta = 1$$

αφού πρόκειται για τη σ.π.π της $Gamma(x+a,\beta+1)$.
Οπότε αποδείχθηκε το ζητούμενο:

$$f(x) = \frac{\beta^a}{x!\Gamma(a)}\frac{\Gamma(x+a)}{(\beta+1)^{x+a}} = \frac{\Gamma(x+a)}{x!\Gamma(a)}(\frac{\beta}{\beta+1})^a(\frac{1}{\beta+1})^x.$$

# 4 Άσκηση 4

Θα προσαρμόσουμε το μοντέλο πολλαπλής γραμμικής παλινδρόμησης
$Y = \beta_0 + \beta_1 X_1 + ... + \beta_{15}X_{15} + \epsilon$ με $n = 50$ παρατηρήσεις. Οι ανεξάρτητες μεταβλητές $\vec{X}_{1...10} \sim MVN(\vec{0}, I_{10\times 10}) \Rightarrow X_i \sim N(0,1),\ i = 1,...,10.$
Οι υπόλοιπες ανεξάρτητες μεταβλητές βρίσκονται ως:

$$X_{ij} \sim N(0.2X_{i1}+0.4X_{i2}+0.6X_{i3}+0.8X_{i4}+1.1X_{i5},1),\ i = 1,...50\ \ j = 11,...,15.$$

Οι τιμές της μεταβλητής απόκρισης δίνονται απο τη σχέση:

$$Y_i \sim N(4 + 2X_{i1} - X_{i5} + 2.5X_{i7} + 1.5X_{i11} + 0.5X_{i13}, 1.5^2),\ i = 1,...,50.$$

```
v1 <- c(rep(0,50))
v2 <- c(rep(0,15))
x <- array(c(v1,v2), dim = c(50,15))
for (i in 1:10){
  x[,i] = rnorm(50)
}
for (k in 11:15){
  for (i in 1:50){
  x[i,k] = rnorm(1,mean = 0.2*x[i,1] + 0.4*x[i,2] + 0.6*x[i,3] +
  0.8*x[i,4] + 1.1*x[i,5], sd = 1)
  }
```

```
  }
  Y <- c(rep(0,50))
  for (i in 1:50){
    Y[i] <- rnorm(1, mean = 4 + 2*x[i,1] - x[i,5] + 2.5*x[i,11] +
    0.5*x[i,13], sd = 1.5)
  }
  xx <- as.data.frame(x)
  mod <- lm(Y~., data = xx)
```

Άρα το μοντέλο:

```
##
## Call:
## lm(formula = Y ~ ., data = xx)
##
## Coefficients:
## (Intercept)           V1           V2           V3           V4           V5
##      4.2162       1.5435      -0.2023      -0.2344      -0.2609      -1.0903
##          V6           V7           V8           V9          V10          V11
##     -0.2199       0.2777      -0.3202       0.1873       0.1090       2.7783
##         V12          V13          V14          V15
##     -0.0273       0.4065       0.5053      -0.1850
```

***α.*** Σκοπός μας είναι να εξερευνήσουμε όλα τα πιθανά $2^{15} = 32.768$ μοντέλα και να βρούμε το μοντέλο το οποίο επιστρέφει τη μικρότερη τιμή του κριτηρίου BIC.
Θα δημιουργήσουμε στην $R$ μια λίστα '$enum$' μεγέθους $32.768 \times 15$ η οποία θα περιέχει ολους τους συνδιασμούς $\{0,1\}^{15}$. Στη συνέχεια θα γίνεται κατάλληλος πολλαπλασιασμός με τη λίστα που στις στείλες της περιέχει τις τιμές των ανεξάρτητων μεταβλητών και θα ελέγχουμε το κριτήριο BIC για κάθε ενα μοντέλο. Ο αλγόριθμος δέχεται το dataframe των ανεξάρτητων μεταβλητών, τη λίστα των $Y$ και επιστρέφει το μοντέλο με το ελάχιστο BIC. Χρησιμοποιήθηκε το πακέτο *lgcp* για το πολλαπλασιασμό λίστας.

```
enumerate <- function(xx, Y){
  enum <- expand.grid(0:1, 0:1, 0:1,0:1, 0:1, 0:1,0:1, 0:1,
  0:1,0:1, 0:1, 0:1,0:1, 0:1, 0:1)
  xlist <- list(xx[1], xx[2], xx[3], xx[4], xx[5], xx[6], xx[7],
  xx[8], xx[9], xx[10], xx[11], xx[12], xx[13], xx[14], xx[15])
  Bmin <- 10000
  for (i in 1:32768){
   z <- as.data.frame(multiply.list(enum[i,],xlist))
   mod <- lm(Y~., data = z)
    b <- BIC(mod)
   if (Bmin > b){
     result <- mod
```

```
      Bmin <- b
    }
   }
  }
  return(result)
}
```

Άρα το μοντέλο με το ελάχιστο BIC:

```
print(enumerate(xx, Y))
>Call:
lm(formula = Y ~ ., data = z)

Coefficients:
(Intercept)           V1           V2           V3           V4
     3.7544       1.8797           NA           NA           NA
         V5           V6           V7           V8           V9
    -0.8711           NA           NA      -0.4028           NA
        V10          V11          V12          V13          V14
         NA       2.3488           NA       0.5594           NA
         V15
         NA
```

Παρατηρούμε οτι στο μοντέλο μας έμειναν μόνο οι συντελεστές των μεταβλητών $X_1, X_5, X_8, X_{11}, X_{13}$ ενώ οι άλλοι μηδενίστηκαν.

***β.*** Η Least Absolute Shrinkage and Selection Operator (Lasso) αποτελεί μια μέθοδο για την εκτίμηση των συντελεστών ενός μοντέλου παλινδρόμησης, αλλά και για την επιλογή των μεταβλητών. Η ιδέα της συγκεκριμένης μεθόδου βασίζεται στην ελαχιστοποίηση της παράστασης $(y - X\beta)^T(y - X\beta)$ για το προσδιορισμό των συντελεστών του μοντέλου, αλλά η βασική διαφορά με την μέθοδο ελαχίστων τετραγώνων βρίσκεται στην προυπόθεση για την $L_1$ νόρμα: $\sum_{j=1}^{p} |\beta_j| \leq t$. Αυτό επιτυγχάνει τη μείωση του τυπικού σφάλματος, αλλά κοστίζει αύξηση στην μεροληψία. Έχουμε πλέον το πρόβλημα ελαχιστοποίησης:

$$minimize[(y - X\beta)^T(y - X\beta) + \lambda \sum_{j=1}^{p} |\beta_j|]$$

όπου $\lambda \in [0, 1]$. Όταν το $\lambda = 0$, δηλαδή απαιτούμε απο τον περιορισμό $t = t_0 = max|\beta|_1$, τότε ή μέθοδος ταυτίζεται με αυτή των ελαχίστων τετραγώνων, ενώ όσο το $\lambda$ μεγαλώνει, αυξάνεται και ο βαθμός συρρίκνωσης των συντελεστών του μοντέλου. Ορίζεται, τέλος, ο συντελεστής συρρίκνωσης $s = \frac{|\beta|_1}{max|\beta|_1}$. Με τη βοήθεια της βιβλιοθήκης *glmnet* θα ορίσουμε στην $R$ το μοντέλο Lasso. Έστω τα διάνυσματα $x$ και $Y$ των παρατηρήσεων των ανεξάρτητων και εξαρτημένων μεταβλητών αντίστοιχα, που έχουν δημιουργηθεί απο τα παραπάνω ερωτήματα.

```
lasso <- glmnet(x, Y)
plot(lasso, label = T)
```

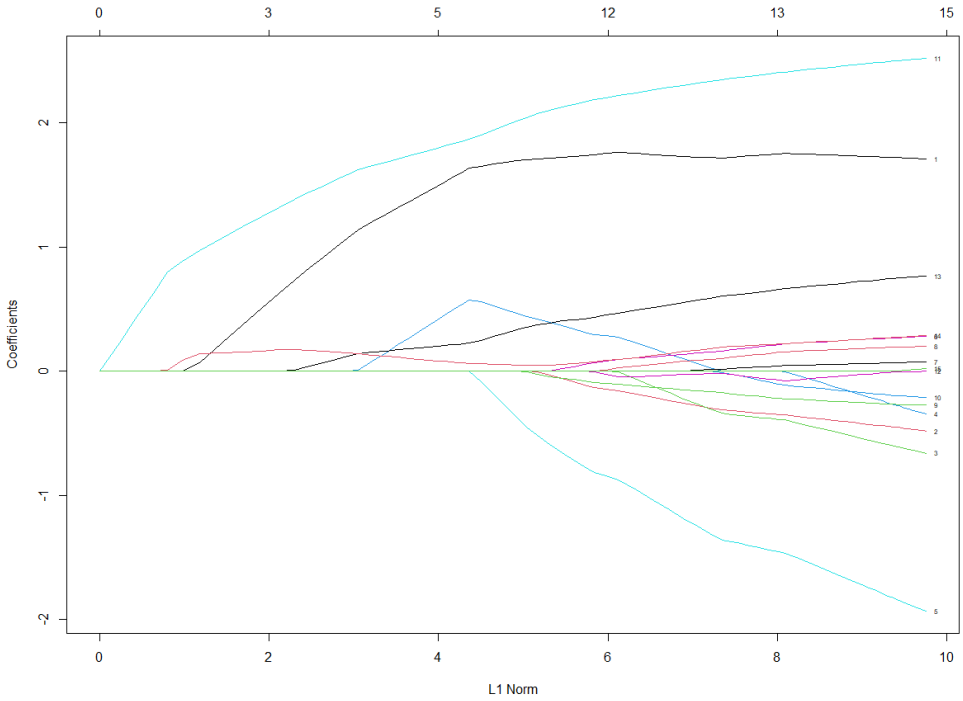



Figure 1: Plot of Coefficients and their L1 norm

Απο το παραπάνω σχήμα μπορούμε να παρατηρήσουμε πόσο ¨απότομα' τείνουν να μηδενιστούν οι συντελεστές της κάθε μεταβλητής, όσο κινούμαστε απο μεγαλύτερες τιμές της $L_1$ νόρμας (άρα το $OLS$ μοντέλο), πρός μικρότερες.

```
> lasso2 <- cv.glmnet(x, Y)
#Minimum Lambda
> lasso2$lambda.min
 0.2062794
 #1se Lambda
lasso2$lambda.1se
 0.4341984
```

Άρα με Cross Validation πήραμε τις τιμές του ελάχιστου $\lambda_{min} = 0.2063$ και αυτού που βρίσκεται ένα τυπικό σφάλμα πιο δεξιά απο το ελάχιστο $\lambda_{1se} = 0.4342$, το οποίο μπορεί να χρησιμοποιηθεί για τη δημιουργία πιο φειδωλών μοντέλων. Τέλος ο συντελεστής συρρίκνωσης $s$ μπορεί να υπολογιστεί ως εξής:

```
zlasso_min <- lasso_min[-1] * apply(x,2,sd)
z <- coef(mod)[-1] * apply(x,2,sd)
s <- sum(abs(zlasso_min))/sum(abs(z))
s
0.5696729
```