Tema 5 - Cosmología

Cedric Prieels

Mayo 2017

Lo primero que podemos hacer para resolver el ejercicio consiste en usar la herramienta CAMB para generar los espectros de potencias del CMB que vamos a analizar en este ejercicio para los modos T, E y B. Este espectro se genera con unos parámetros generales (con una proporción de materia de 0,68 y un valor de lmax de 3000), y para un valor de razón de amplitudes tensoriales a escalares r igual a 1. Esta herramienta nos devuelve diferentes ficheros *.dat que vamos a poder usar en el ejercicio (en particular vamos a usar 3 de las columnas de estos ficheros, que corresponden a los diferentes modos considerados).

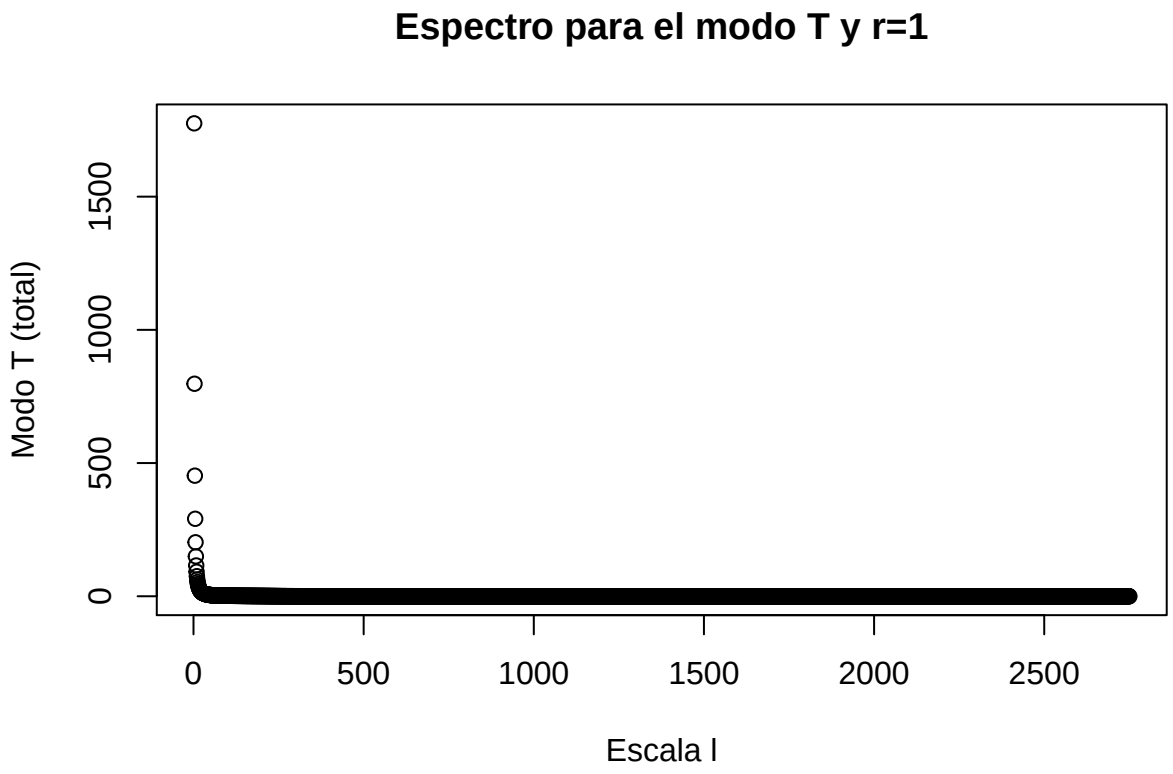
```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Cosmologia/')
scalar <- read.table("camb_24707701_lensedcls.dat", header=FALSE)
total <- read.table("camb_24707701_lensedtotcls.dat", header=FALSE)

l <- total[,1]

T_scal <- scalar[,2]
E_scal <- scalar[,3]
B_scal <- scalar[,4]

T_tot <- total[,2]
E_tot <- total[,3]
B_tot <- total[,4]
```

Solo tenemos que calcular los valores tensoriales, porque no los tenemos de momento. Para calcularlos, hay que quitar los valores scalares a los valores totales que tenemos, porque hemos creado los ficheros de datos precedentes con r=1 y porque sabemos que $f_l^{total} = f_l^{escalar} + r \cdot f_l^{tensorial}$.

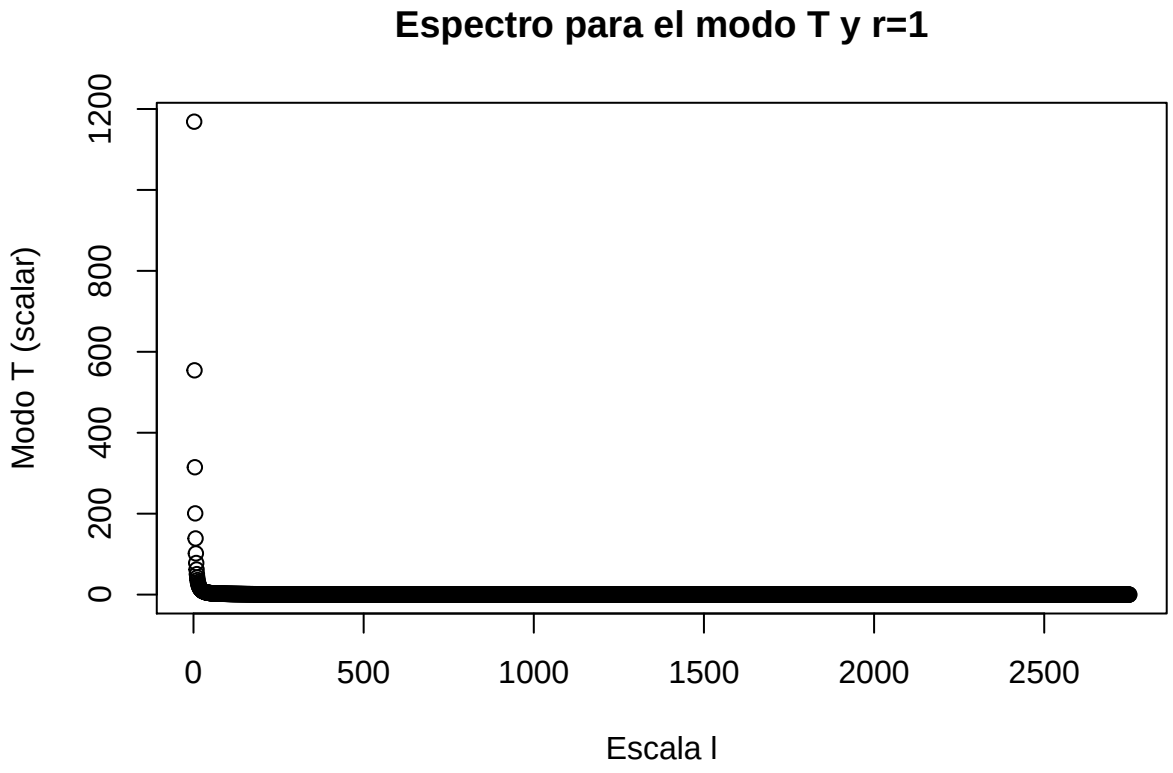
```
T_tens <- (T_tot - T_scal)
E_tens <- (E_tot - E_scal)
B_tens <- (B_tot - B_scal)
```

Podemos ahora representar los diferentes espectros que tenemos. Pero primero, tenemos que aplicar un factor de corrección para poder aplicar Fisher, porque lo que tenemos son los valores de D_l y no $C_l = \frac{2\pi \cdot D_l}{l(l+1)}$.

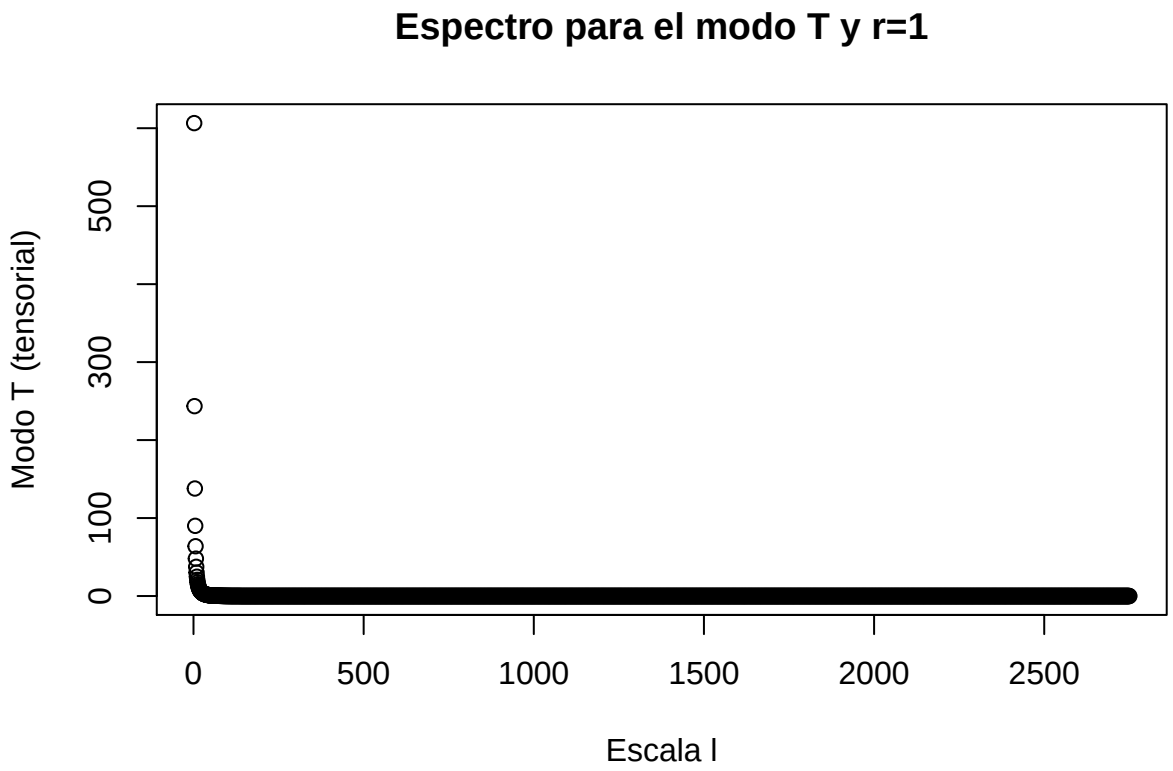
```
T_tot <- (2 * 3.1415 * T_tot)/(l * (l+1))
T_scal <- (2 * 3.1415 * T_scal)/(l * (l+1))
T_tens <- (2 * 3.1415 * T_tens)/(l * (l+1))
plot(l, T_tot, main="Espectro para el modo T y r=1", xlab="Escala l", ylab="Modo T (total)")
```



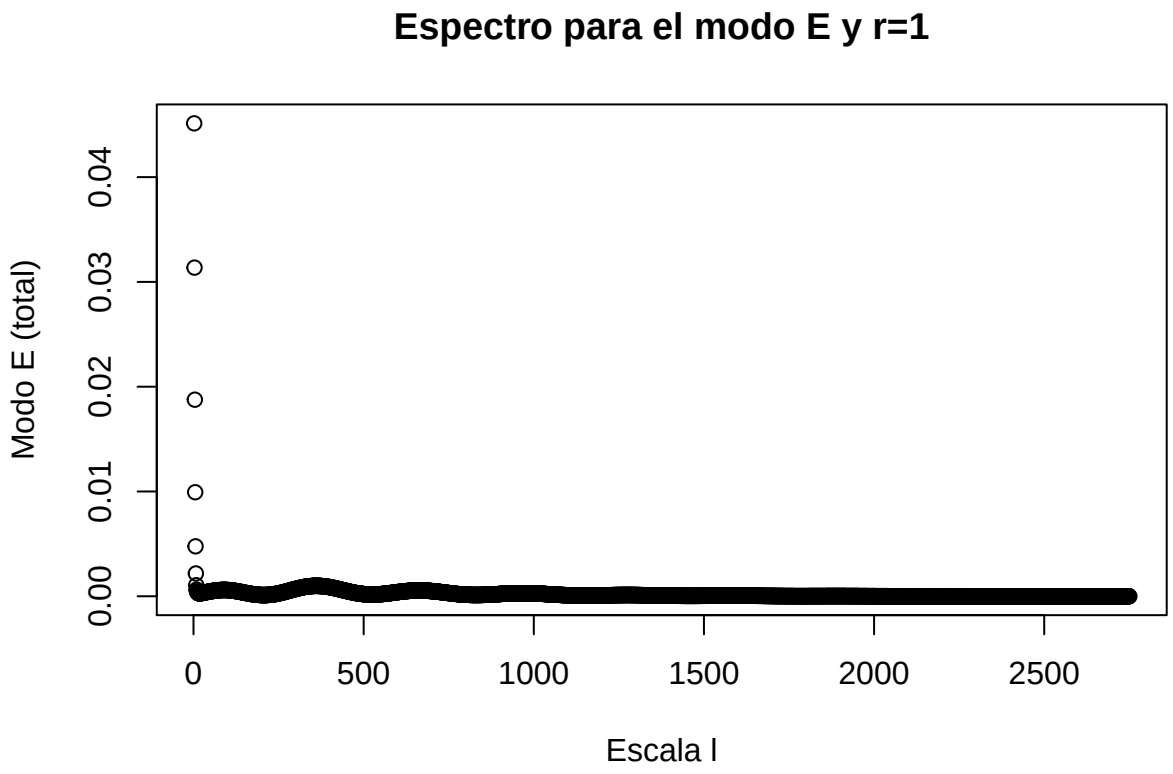

```
plot(l, T_scal, main="Espectro para el modo T y r=1", xlab="Escala l", ylab="Modo T (scalar)")
```



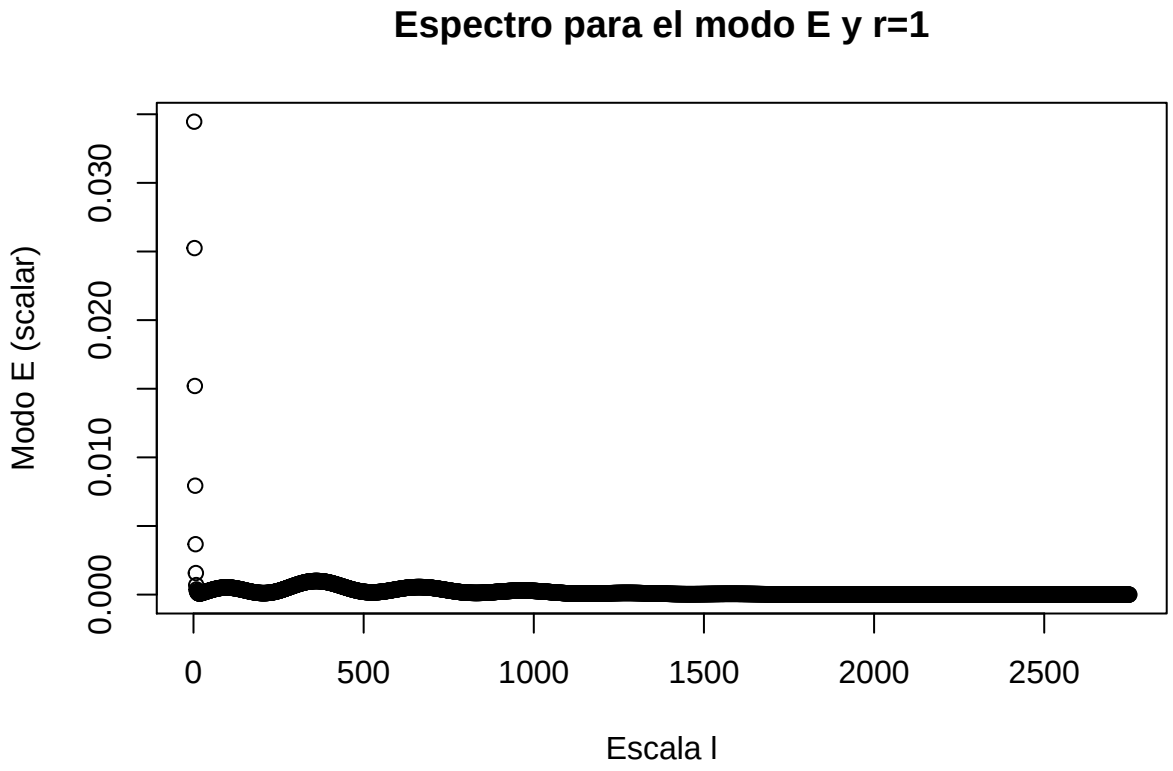
```
plot(l, T_tens, main="Espectro para el modo T y r=1", xlab="Escala l", ylab="Modo T (tensorial)")
```



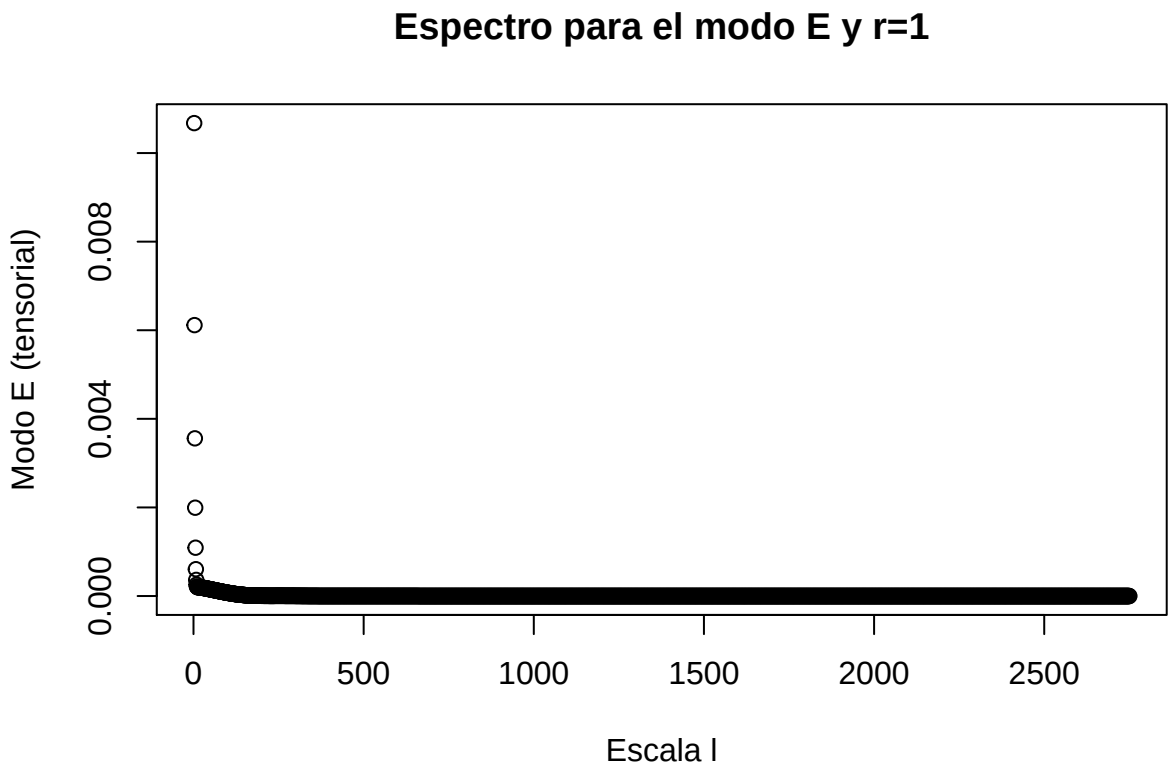

```
E_tot <- (2 * 3.1415 * E_tot)/(l * (l+1))
E_scal <- (2 * 3.1415 * E_scal)/(l * (l+1))
E_tens <- (2 * 3.1415 * E_tens)/(l * (l+1))
plot(l, E_tot, main="Espectro para el modo E y r=1", xlab="Escala l", ylab="Modo E (total)")
```



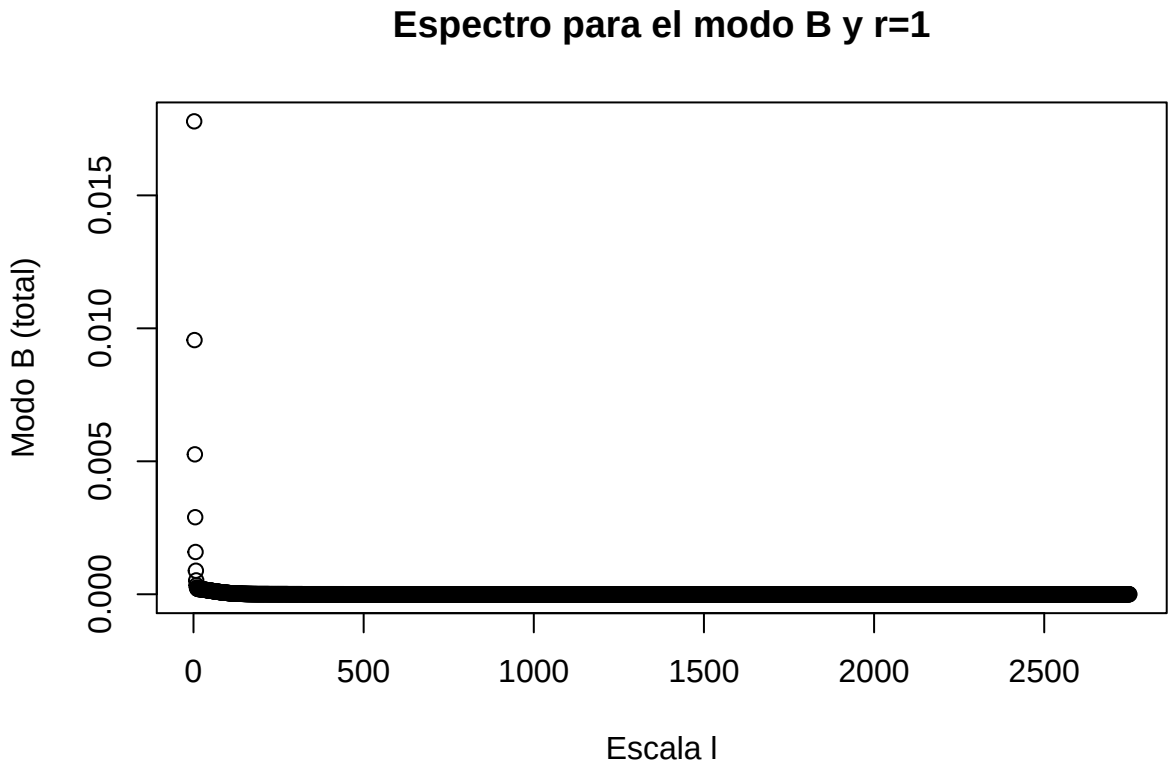

```
plot(l, E_scal, main="Espectro para el modo E y r=1", xlab="Escala l", ylab="Modo E (scalar)")
```



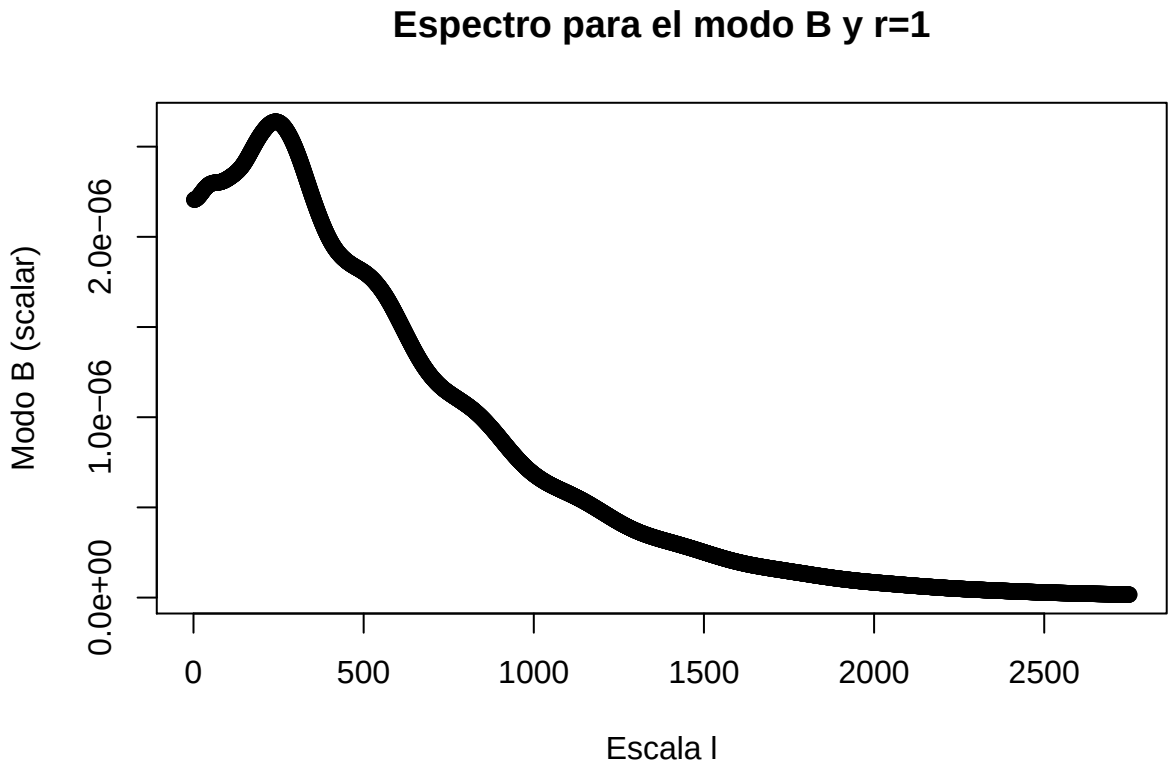
```
plot(l, E_tens, main="Espectro para el modo E y r=1", xlab="Escala l", ylab="Modo E (tensorial)")
```



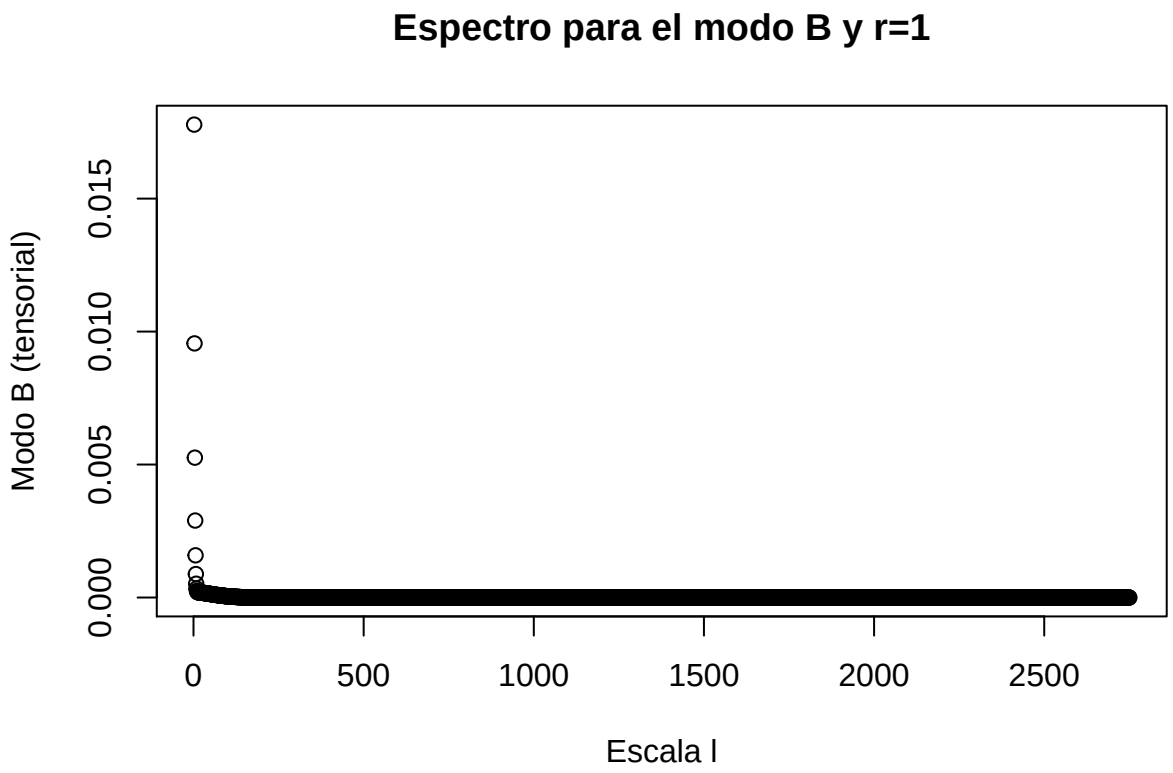

```
B_tot <- (2 * 3.1415 * B_tot)/(l * (l+1))
B_scal <- (2 * 3.1415 * B_scal)/(l * (l+1))
B_tens <- (2 * 3.1415 * B_tens)/(l * (l+1))
plot(l, B_tot, main="Espectro para el modo B y r=1", xlab="Escala l", ylab="Modo B (total)")
```




```
plot(l, B_scal, main="Espectro para el modo B y r=1", xlab="Escala l", ylab="Modo B (scalar)")
```



```r
plot(l, B_tens, main="Espectro para el modo B y r=1", xlab="Escala l", ylab="Modo B (tensorial)")
```



Podemos ahora ver que tener acceso al espectro para r=1 no nos limita porque sabemos que $T_l = T_l^{escalar} + r \cdot T_l^{tensorial}$, que $E_l = E_l^{escalar} + r \cdot E_l^{tensorial}$ y que $B_l = B_l^{escalar} + r \cdot B_l^{tensorial}$. Por lo tanto, podemos calcular los datos esperados para distintos valores de r directamente a partir de los datos que tenemos para r=1.

```r
r <- c(0.1, 0.01, 0.005, 0.001)

for (valor_r in r){

  assign(paste("T_tot", valor_r, sep="_"), (T_scal + valor_r * (T_tens)))
  assign(paste("E_tot", valor_r, sep="_"), (E_scal + valor_r * (E_tens)))
  assign(paste("B_tot", valor_r, sep="_"), (B_scal + valor_r * (B_tens)))

}
```

Ahora tenemos acceso a los valores simulados para los 4 valores de r que queremos considerar en el ejercicio. El paso siguiente consiste en calcular la varianza cósmica de estos datos con la ecuación $var[f(l)] = \sigma^2 = \frac{f(l)^2}{l+0,5}$, donde f corresponde a cada uno de los tres modos precedentes.

```r
for (valor_r in r){

  T_tot_valor = get(paste("T_tot", valor_r, sep="_"))
  E_tot_valor = get(paste("E_tot", valor_r, sep="_"))
  B_tot_valor = get(paste("B_tot", valor_r, sep="_"))

  assign(paste("sigma_T", valor_r, sep="_"), (((T_tot_valor)^2) / (l+0.5)))
```

```
  assign(paste("sigma_E", valor_r, sep="_"), (((E_tot_valor)^2) / (l+0.5)))
  assign(paste("sigma_B", valor_r, sep="_"), (((B_tot_valor)^2) / (l+0.5)))

}
```

Una vez hecho, tenemos que calcular el log-likelihood de las distribuciones que tenemos, porque la "matriz" de Fisher (en este caso, es en realidad solamente un número) que tenemos y que nos permitirá calcular el valor del error sobre el parámetro r en cada caso vale $F = -\frac{\partial^2 log(L)}{\partial(r)^2}$, siendo $log(L) = -0.5 \cdot \chi^2(+C)$. Como conocemos el valor de $\chi^2 = \sum_l (\frac{f(l) - f_{scalar}(l) - r \cdot f_{tensorial}(l)}{\sigma(l)})^2$ podemos derivar dos veces esta expresión para encontrar que $F = \sum_l \frac{f_{tensorial}(l)^2}{\sigma(l)}$, una expresión sencilla que podemos calcular para cada modo.

```
for (valor_r in r){

  sigma_T_valor <- get(paste("sigma_T", valor_r, sep="_"))
  sigma_E_valor <- get(paste("sigma_E", valor_r, sep="_"))
  sigma_B_valor <- get(paste("sigma_B", valor_r, sep="_"))

  fisher_T_valor <- sum((T_tens^2)/sigma_T_valor)
  fisher_E_valor <- sum((E_tens^2)/sigma_E_valor)
  fisher_B_valor <- sum((B_tens^2)/sigma_B_valor)

  assign(paste("fisher_T", valor_r, sep="_"), fisher_T_valor)
  assign(paste("fisher_E", valor_r, sep="_"), fisher_E_valor)
  assign(paste("fisher_B", valor_r, sep="_"), fisher_B_valor)

}
```

Una vez que tenemos acceso al valor de la matriz de Fisher podemos estimar el valor del error sobre el valor de r como siendo igual al inverso de la raíz de esta matriz. Podemos por lo tanto pintar el valor de detección $\frac{r}{\sigma_r}$ en función del valor del parámetro r y del valor de l_max considerado.

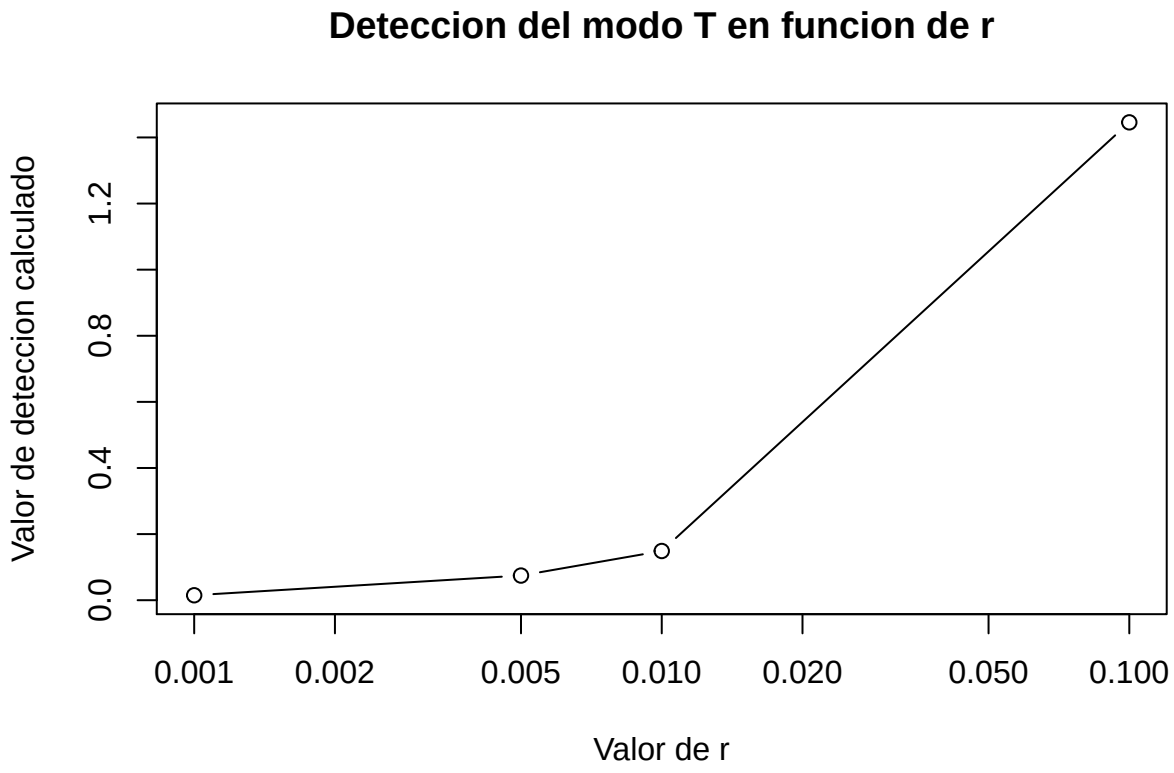
```
for (valor_r in r){

  fisher_T_valor <- get(paste("fisher_T", valor_r, sep="_"))
  fisher_E_valor <- get(paste("fisher_E", valor_r, sep="_"))
  fisher_B_valor <- get(paste("fisher_B", valor_r, sep="_"))

  error_T_valor <- 1/(sqrt(fisher_T_valor))
  error_E_valor <- 1/(sqrt(fisher_E_valor))
  error_B_valor <- 1/(sqrt(fisher_B_valor))

  assign(paste("error_T", valor_r, sep="_"), error_T_valor)
  assign(paste("error_E", valor_r, sep="_"), error_E_valor)
  assign(paste("error_B", valor_r, sep="_"), error_B_valor)

}
```

Por fin, el último etapa antes de pintar los resultados obtenidos consiste en calcular el valor de detección en cada caso, definido como el ratio entre el valor de r y su error.

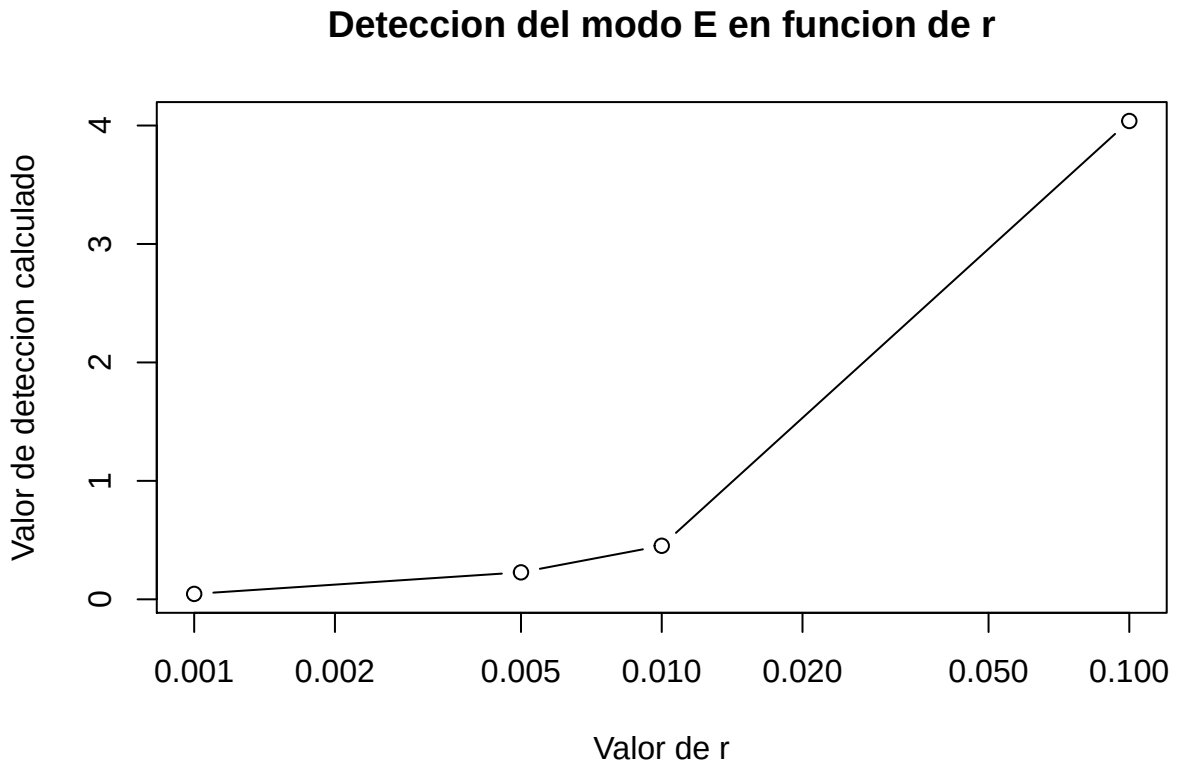
```
for (valor_r in r){

  deteccion_T_valor <- (valor_r / get(paste("error_T", valor_r, sep="_")))
  deteccion_E_valor <- (valor_r / get(paste("error_E", valor_r, sep="_")))
  deteccion_B_valor <- (valor_r / get(paste("error_B", valor_r, sep="_")))

  assign(paste("deteccion_T", valor_r, sep="_"), deteccion_T_valor)
  assign(paste("deteccion_E", valor_r, sep="_"), deteccion_E_valor)
  assign(paste("deteccion_B", valor_r, sep="_"), deteccion_B_valor)

}
```

Ahora solo hace falta representar los resultados obtenidos para el valor de elegido al principio ($l_max = 3000$).

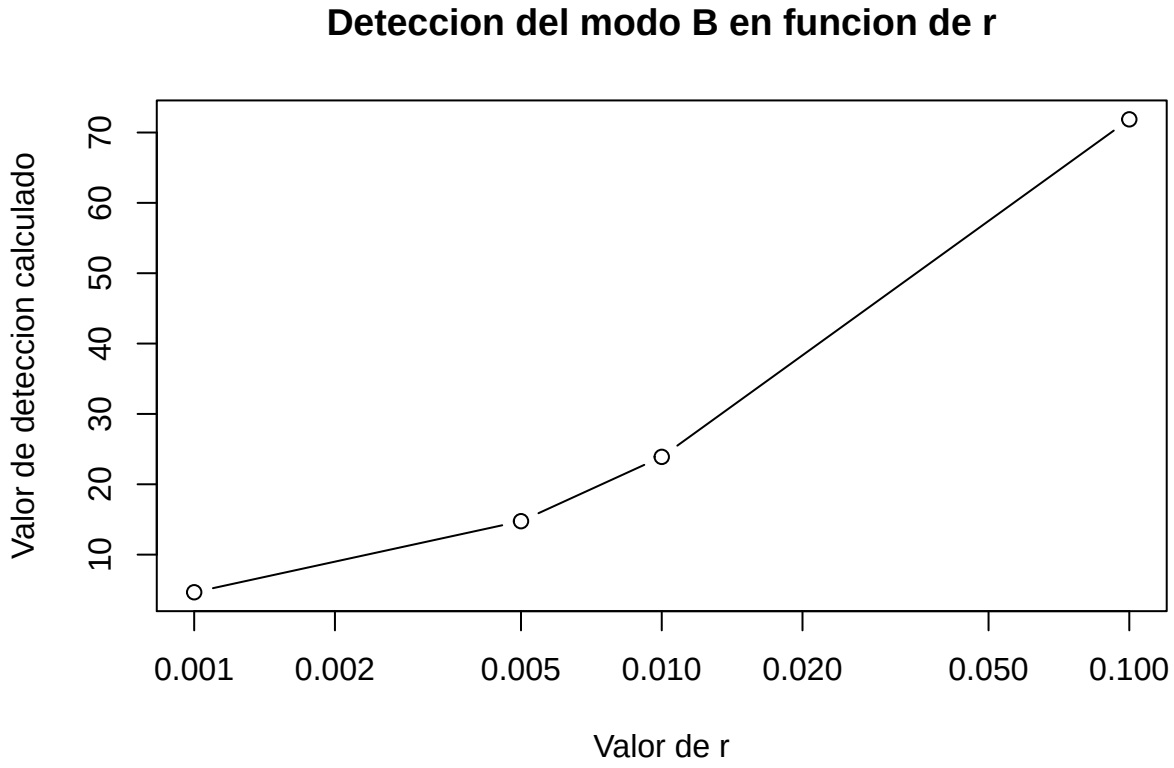
```
deteccion_T <- c()
deteccion_E <- c()
deteccion_B <- c()

for (valor_r in r){

  deteccion_T <- append(deteccion_T, get(paste("deteccion_T", valor_r, sep="_")))
  deteccion_E <- append(deteccion_E, get(paste("deteccion_E", valor_r, sep="_")))
  deteccion_B <- append(deteccion_B, get(paste("deteccion_B", valor_r, sep="_")))

}
plot(r, deteccion_T, main="Deteccion del modo T en funcion de r",
     xlab="Valor de r", ylab="Valor de deteccion calculado", log="x", type="b")
```



```
plot(r, deteccion_E, main="Deteccion del modo E en funcion de r",
     xlab="Valor de r", ylab="Valor de deteccion calculado", log="x", type="b")
```



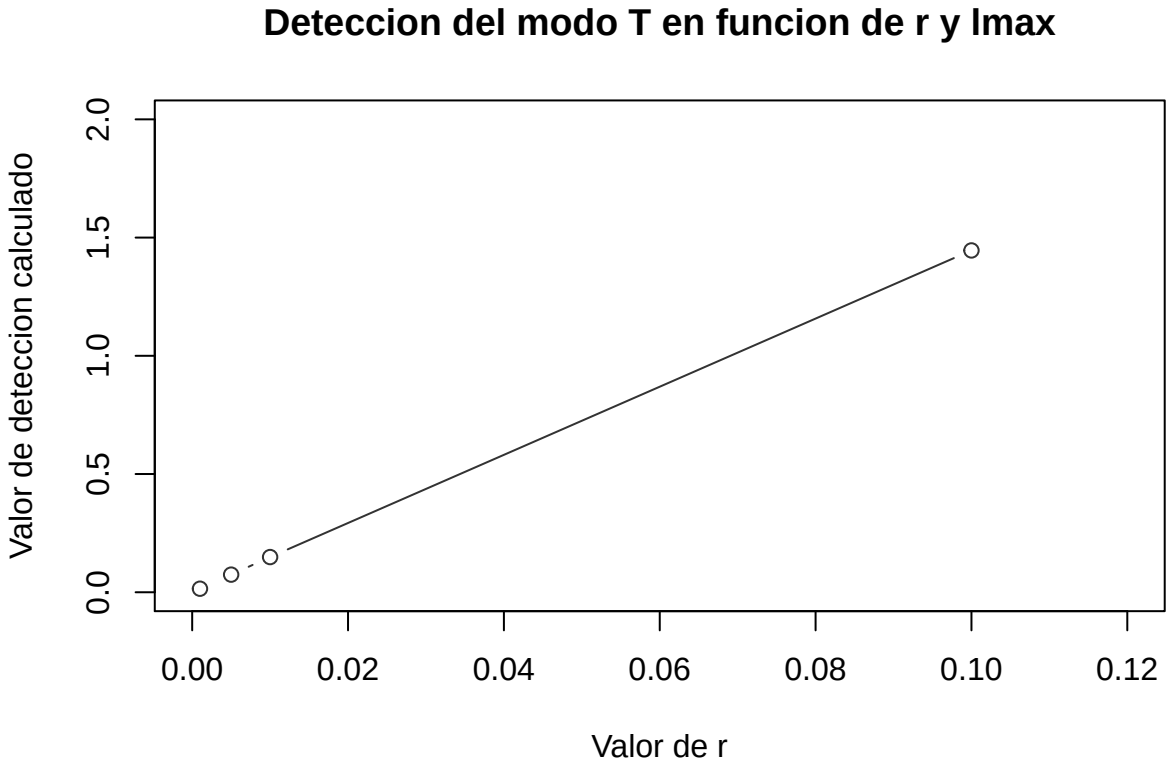

```
plot(r, deteccion_B, main="Deteccion del modo B en funcion de r",
     xlab="Valor de r", ylab="Valor de deteccion calculado", log="x", type="b")
```



Ahora hay que repetir todos los pasos precedentes considerando además valores de l_{max} distintos. Primero, creamos el plot del modo T solamente.

```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Cosmologia/')
scalar <- read.table("camb_24707701_lensedcls.dat", header=FALSE)
total <- read.table("camb_24707701_lensedtotcls.dat", header=FALSE)

l <- total[,1]

T_scal <- scalar[,2]
T_tot <- total[,2]
T_tens <- (T_tot - T_scal)

#Volvemos a pasar de los coeficientes Dl a Cl
T_tot <- (2 * 3.1415 * T_tot)/(l * (l+1))
T_scal <- (2 * 3.1415 * T_scal)/(l * (l+1))
T_tens <- (2 * 3.1415 * T_tens)/(l * (l+1))

l_max <- c(3000)
r <- c(0.1, 0.01, 0.005, 0.001)

counter <- 20

plot(1, type="n", main="Deteccion del modo T en funcion de r y lmax",
     xlab="Valor de r", ylab="Valor de deteccion calculado",
     xlim=c(0,0.12), ylim=c(0,2))
```

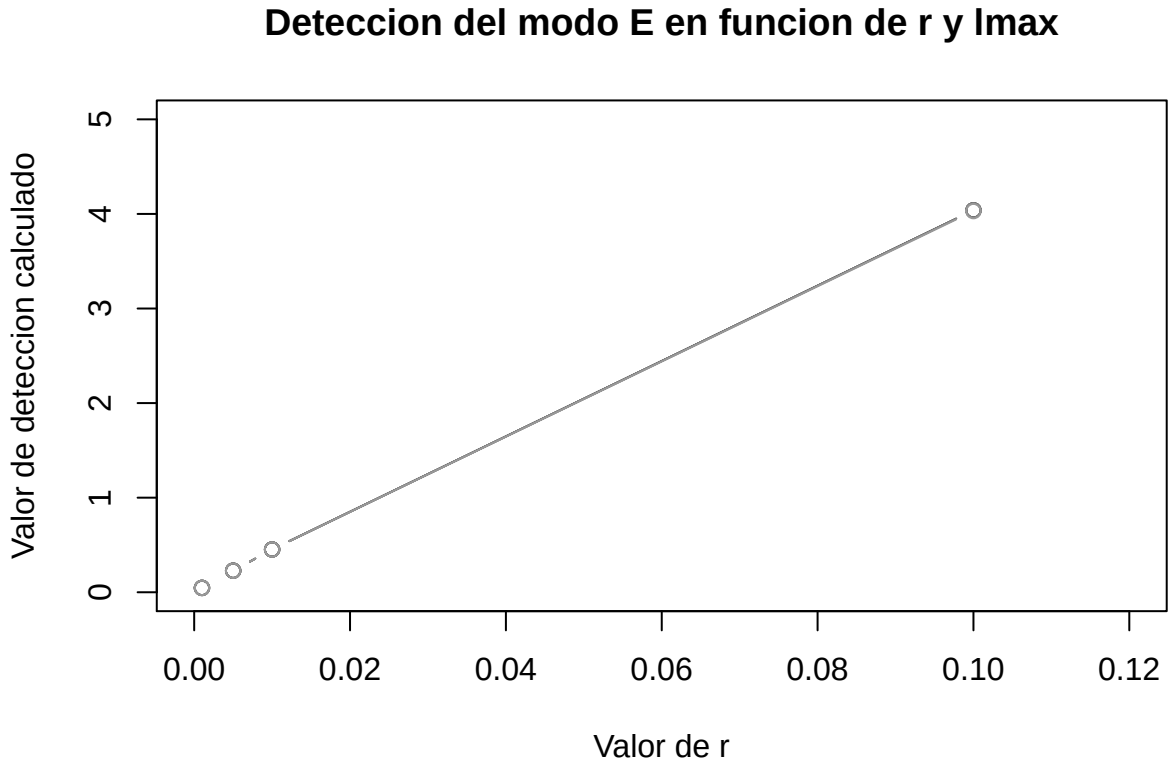
```
for (valor_l in l_max) {

  l <- head(l, valor_l)

  T_scal <- head(T_scal, valor_l)
  T_tens <- head(T_tens, valor_l)
  T_tot <- head(T_tot, valor_l)

  deteccion_T <- c()

  for (valor_r in r){

    assign(paste("T_tot", valor_r, sep="_"), (T_scal + valor_r * (T_tens)))

    T_tot_valor = get(paste("T_tot", valor_r, sep="_"))
    assign(paste("sigma_T", valor_r, sep="_"), (((T_tot_valor)^2) / (l+0.5)))

    sigma_T_valor <- get(paste("sigma_T", valor_r, sep="_"))
    T_tens_valor <- T_tens
    fisher_T_valor <- sum((T_tens_valor^2)/sigma_T_valor)
    assign(paste("fisher_T", valor_r, sep="_"), fisher_T_valor)

    fisher_T_valor <- get(paste("fisher_T", valor_r, sep="_"))
    error_T_valor <- 1/(sqrt(fisher_T_valor))
    assign(paste("error_T", valor_r, sep="_"), error_T_valor)

    deteccion_T_valor <- (valor_r / get(paste("error_T", valor_r, sep="_")))
    assign(paste("deteccion_T", valor_r, sep="_"), deteccion_T_valor)

    deteccion_T <- append(deteccion_T, get(paste("deteccion_T", valor_r, sep="_")))

  }

  myColor = paste("grey", counter, sep="")
  counter <- counter + 10

  lines(r, deteccion_T, main="Deteccion del modo T en funcion de r y lmax",
        xlab="Valor de r", ylab="Valor de deteccion calculado",
        col=myColor, type = "b")

}
```



No detectamos ninguna dependencia con el valor del parámetro l_{max}, porque todas las lineas están superpuestas. Ahora podemos volver a hacer lo mismo, considerando esta vez el modo E.

```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Cosmologia/')
scalar <- read.table("camb_24707701_lensedcls.dat", header=FALSE)
total <- read.table("camb_24707701_lensedtotcls.dat", header=FALSE)

l <- total[,1]

E_scal <- scalar[,3]
E_tot <- total[,3]
E_tens <- (E_tot - E_scal)

#Volvemos a pasar de los coeficientes Dl a Cl
E_tot <- (2 * 3.1415 * E_tot)/(l * (l+1))
E_scal <- (2 * 3.1415 * E_scal)/(l * (l+1))
E_tens <- (2 * 3.1415 * E_tens)/(l * (l+1))

l_max <- c(2500, 2000, 1500, 1000, 200)
r <- c(0.1, 0.01, 0.005, 0.001)

#Consideramos ahora solo el modo E
counter <- 20

plot(1, type="n", main="Deteccion del modo E en funcion de r y lmax",
     xlab="Valor de r", ylab="Valor de deteccion calculado",
     xlim=c(0,0.12), ylim=c(0,5))
```

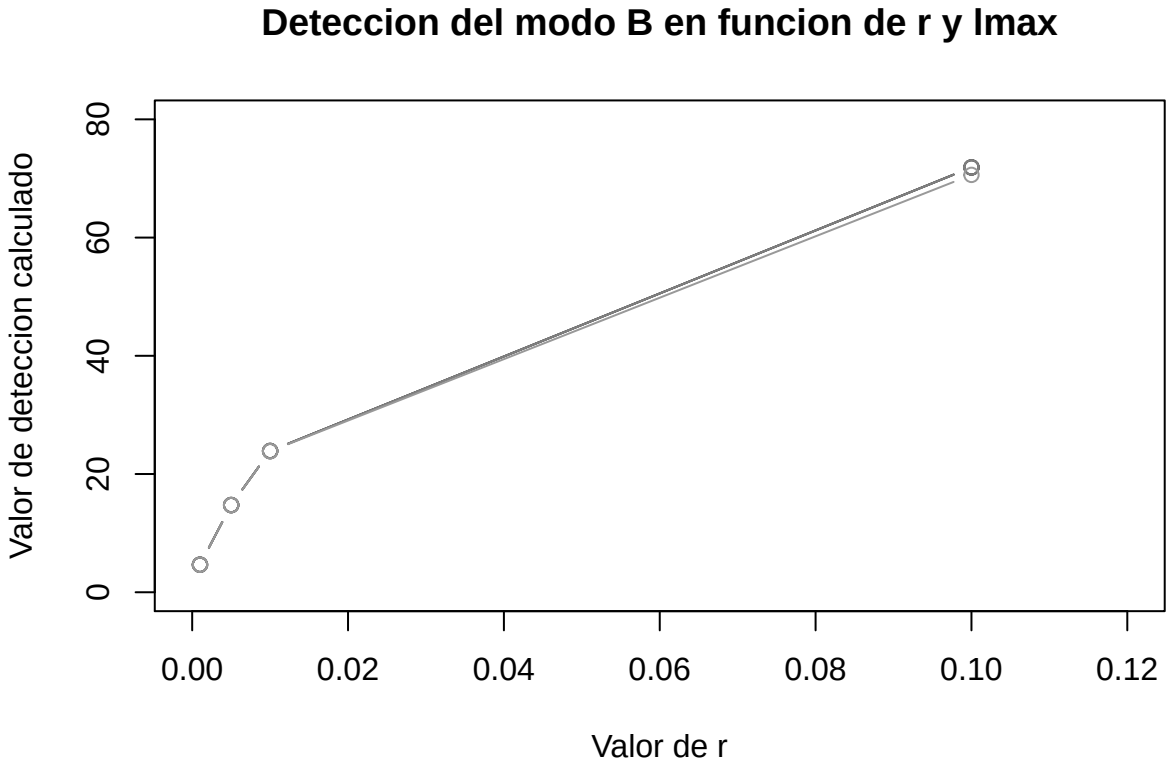
```
for (valor_l in l_max) {

  l <- head(l, valor_l)

  E_scal <- head(E_scal, valor_l)
  E_tens <- head(E_tens, valor_l)
  E_tot <- head(E_tot, valor_l)

  deteccion_E <- c()

  for (valor_r in r){

    assign(paste("E_tot", valor_r, sep="_"), (E_scal + valor_r * (E_tens)))

    E_tot_valor = get(paste("E_tot", valor_r, sep="_"))
    assign(paste("sigma_E", valor_r, sep="_"), (((E_tot_valor)^2) / (l+0.5)))

    sigma_E_valor <- get(paste("sigma_E", valor_r, sep="_"))
    E_tens_valor <- E_tens
    fisher_E_valor <- sum((E_tens_valor^2)/sigma_E_valor)
    assign(paste("fisher_E", valor_r, sep="_"), fisher_E_valor)

    fisher_E_valor <- get(paste("fisher_E", valor_r, sep="_"))
    error_E_valor <- 1/(sqrt(fisher_E_valor))
    assign(paste("error_E", valor_r, sep="_"), error_E_valor)

    deteccion_E_valor <- (valor_r / get(paste("error_E", valor_r, sep="_")))
    assign(paste("deteccion_E", valor_r, sep="_"), deteccion_E_valor)

    deteccion_E <- append(deteccion_E, get(paste("deteccion_E", valor_r, sep="_")))

  }

  myColor = paste("grey", counter, sep="")
  counter <- counter + 10

  lines(r, deteccion_E, main="Deteccion del modo E en funcion de r y lmax",
        xlab="Valor de r", ylab="Valor de deteccion calculado",
        col=myColor, type = "b")

}
```



Tampoco vemos que el valor de l_{max} influye los resultados. Y por fin, sacamos el mismo plot para el modo B.

```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Cosmologia/')
scalar <- read.table("camb_24707701_lensedcls.dat", header=FALSE)
total <- read.table("camb_24707701_lensedtotcls.dat", header=FALSE)

l <- total[,1]

B_scal <- scalar[,4]
B_tot <- total[,4]
B_tens <- (B_tot - B_scal)

#Volvemos a pasar de los coeficientes Dl a Cl
B_tot <- (2 * 3.1415 * B_tot)/(l * (l+1))
B_scal <- (2 * 3.1415 * B_scal)/(l * (l+1))
B_tens <- (2 * 3.1415 * B_tens)/(l * (l+1))

l_max <- c(2500, 2000, 1500, 1000, 200)
r <- c(0.1, 0.01, 0.005, 0.001)

#Consideramos al final solo el modo B
counter <- 20

plot(1, type="n", main="Deteccion del modo B en funcion de r y lmax",
     xlab="Valor de r", ylab="Valor de deteccion calculado",
     xlim=c(0,0.12), ylim=c(0,80))
```

```
for (valor_l in l_max) {

  l <- head(l, valor_l)

  B_scal <- head(B_scal, valor_l)
  B_tens <- head(B_tens, valor_l)
  B_tot <- head(B_tot, valor_l)

  deteccion_B <- c()

  for (valor_r in r){

    assign(paste("B_tot", valor_r, sep="_"), (B_scal + valor_r * (B_tens)))

    B_tot_valor = get(paste("B_tot", valor_r, sep="_"))
    assign(paste("sigma_B", valor_r, sep="_"), (((B_tot_valor)^2) / (l+0.5)))

    sigma_B_valor <- get(paste("sigma_B", valor_r, sep="_"))
    B_tens_valor <- B_tens
    fisher_B_valor <- sum((B_tens_valor^2)/sigma_B_valor)
    assign(paste("fisher_B", valor_r, sep="_"), fisher_B_valor)

    fisher_B_valor <- get(paste("fisher_B", valor_r, sep="_"))
    error_B_valor <- 1/(sqrt(fisher_B_valor))
    assign(paste("error_B", valor_r, sep="_"), error_B_valor)

    deteccion_B_valor <- (valor_r / get(paste("error_B", valor_r, sep="_")))
    assign(paste("deteccion_B", valor_r, sep="_"), deteccion_B_valor)

    deteccion_B <- append(deteccion_B, get(paste("deteccion_B", valor_r, sep="_")))

  }

  myColor = paste("grey", counter, sep="")
  counter <- counter + 10

  lines(r, deteccion_B, main="Deteccion del modo B en funcion de r y lmax",
        xlab="Valor de r", ylab="Valor de deteccion calculado",
        col=myColor, type = "b")

}
```

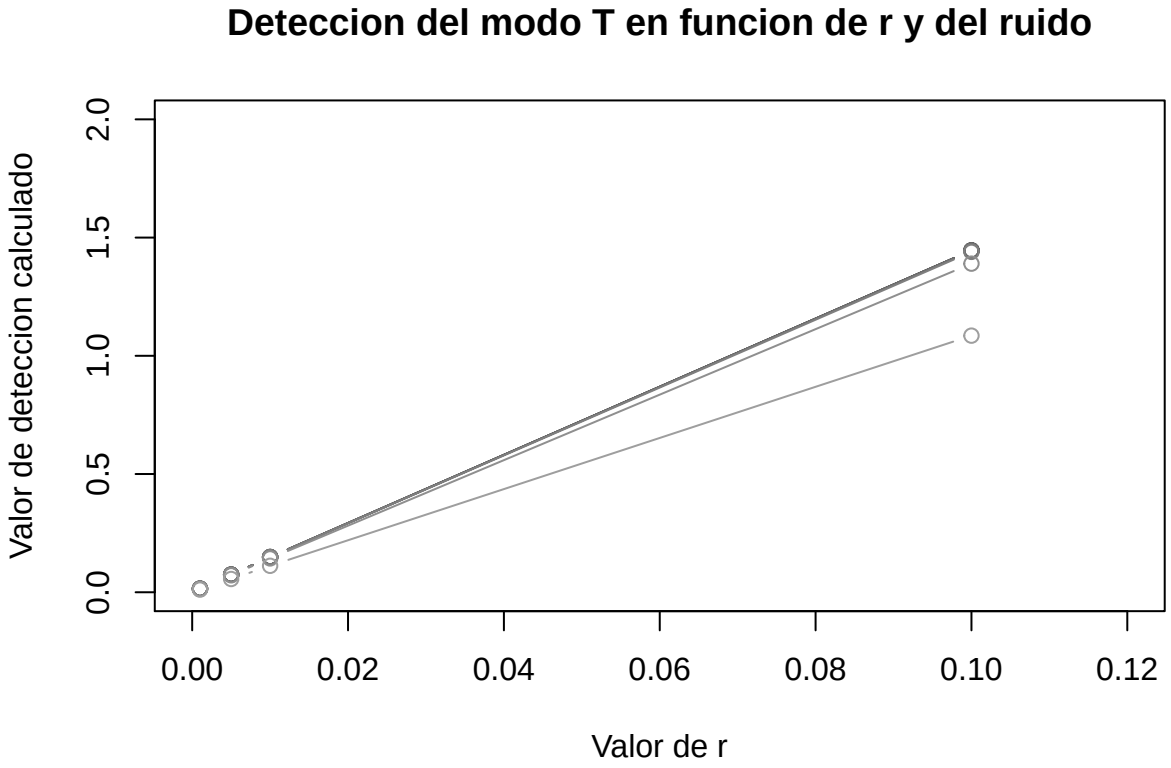


En este último plot, podemos apreciar une pequeña desviación de la detección cuando modificamos el parámetro l_{max}.

Por fin, el último etapa consiste en añadir un ruido blanco (de espectro angular de potencias constante) de varianza 0, 0.00001, 0.001, 0.05 y 0.1 al análisis hecho hasta ahora. Lo que cambia entonces en este caso es el valor de σ, que vale ahora $var[f(l)] = \sigma^2 = \frac{(f(l)+N(l))^2}{l+0,5}$, si N(l) es el ruido añadido. Por supuesto, esperemos volver a obtener exactamente los mismos números que antes en el caso de ruido de varianza 0. Vamos a considerar primero el modo T, y solamente un caso de l_{max}. La distribución de ruido se obtiene calculando primero la varianza del CMB al cuadrado, y multiplicando esta varianza por el valor de varianza de ruido que queremos considerar (de 0 a 100).

```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Cosmologia/')
scalar <- read.table("camb_24707701_lensedcls.dat", header=FALSE)
total <- read.table("camb_24707701_lensedtotcls.dat", header=FALSE)

l <- total[,1]

T_scal <- scalar[,2]
T_tot <- total[,2]
T_tens <- (T_tot - T_scal)

#Volvemos a pasar de los coeficientes Dl a Cl
T_tot <- (2 * 3.1415 * T_tot)/(l * (l+1))
T_scal <- (2 * 3.1415 * T_scal)/(l * (l+1))
T_tens <- (2 * 3.1415 * T_tens)/(l * (l+1))

var <- c(0, 0.00001, 0.001, 0.05, 0.1, 1, 10, 100)
r <- c(0.1, 0.01, 0.005, 0.001)
```

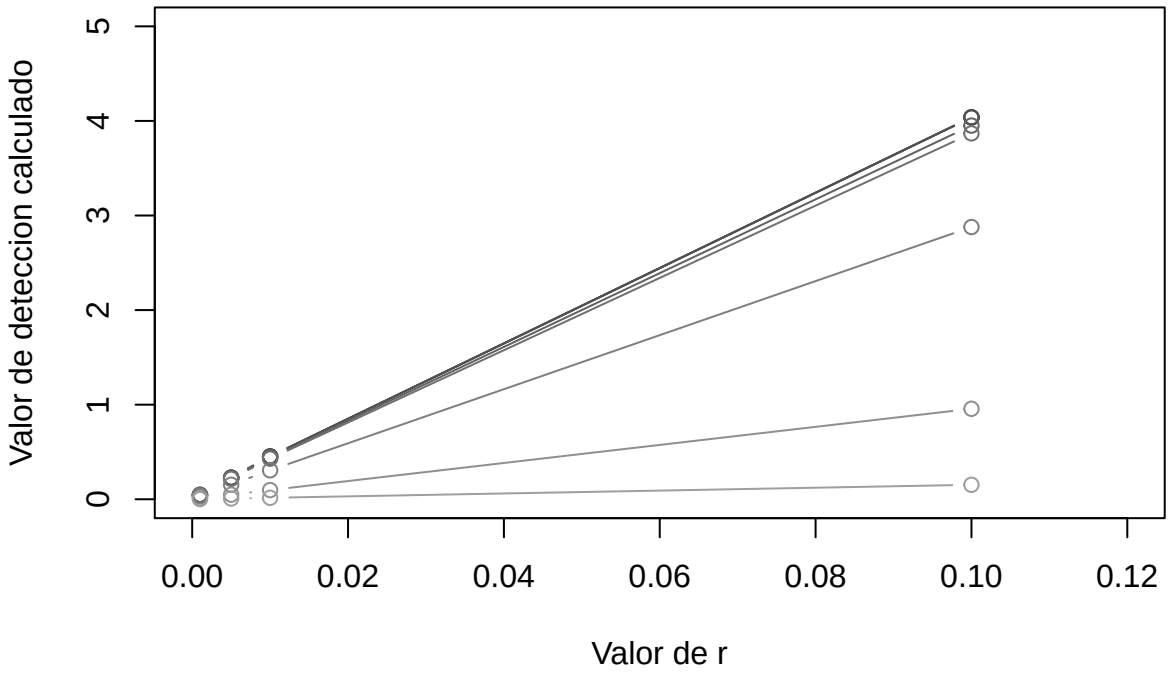
```
counter <- 20

plot(1, type="n", main="Deteccion del modo T en funcion de r y del ruido",
     xlab="Valor de r", ylab="Valor de deteccion calculado",
     xlim=c(0,0.12), ylim=c(0,2))

for (valor_var in var) {

  deteccion_T <- c()

  for (valor_r in r){

    assign(paste("T_tot", valor_r, sep="_"), (T_scal + valor_r * (T_tens)))

    var2_CMB <- sum((2*l+1)*get(paste("T_tot", valor_r, sep="_")))
    ruido <- (var2_CMB*valor_var)/(sum((2*l+1)))

    T_tot_valor = get(paste("T_tot", valor_r, sep="_"))
    assign(paste("sigma_T", valor_r, sep="_"), (((T_tot_valor + ruido)^2) / (l+0.5)))

    sigma_T_valor <- get(paste("sigma_T", valor_r, sep="_"))
    T_tens_valor <- T_tens
    fisher_T_valor <- sum((T_tens_valor^2)/sigma_T_valor)
    assign(paste("fisher_T", valor_r, sep="_"), fisher_T_valor)

    fisher_T_valor <- get(paste("fisher_T", valor_r, sep="_"))
    error_T_valor <- 1/(sqrt(fisher_T_valor))
    assign(paste("error_T", valor_r, sep="_"), error_T_valor)

    deteccion_T_valor <- (valor_r / get(paste("error_T", valor_r, sep="_")))
    assign(paste("deteccion_T", valor_r, sep="_"), deteccion_T_valor)

    deteccion_T <- append(deteccion_T, get(paste("deteccion_T", valor_r, sep="_")))

  }

  myColor = paste("grey", counter, sep="")
  counter <- counter + 6

  lines(r, deteccion_T, main="Deteccion del modo T en funcion de r y del ruido",
        xlab="Valor de r", ylab="Valor de deteccion calculado",
        col=myColor, type = "b")

}
```



No detectemos dependencia de esta parámetro para valores de varianza de ruide pequeños como esperado, solo empezamos a ver desviaciones para los valores de varianza iguales a 10 y 100. Volvemos ahora a calcular lo mismo en el caso del modo E.

```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Cosmologia/')
scalar <- read.table("camb_24707701_lensedcls.dat", header=FALSE)
total <- read.table("camb_24707701_lensedtotcls.dat", header=FALSE)

l <- total[,1]

E_scal <- scalar[,3]
E_tot <- total[,3]
E_tens <- (E_tot - E_scal)

#Volvemos a pasar de los coeficientes Dl a Cl
E_tot <- (2 * 3.1415 * E_tot)/(l * (l+1))
E_scal <- (2 * 3.1415 * E_scal)/(l * (l+1))
E_tens <- (2 * 3.1415 * E_tens)/(l * (l+1))

var <- c(0, 0.00001, 0.001, 0.05, 0.1, 1, 10, 100)
r <- c(0.1, 0.01, 0.005, 0.001)

#Consideramos ahora solo el modo E
counter <- 20

plot(1, type="n", main="Deteccion del modo E en funcion de r y del ruido",
     xlab="Valor de r", ylab="Valor de deteccion calculado",
```

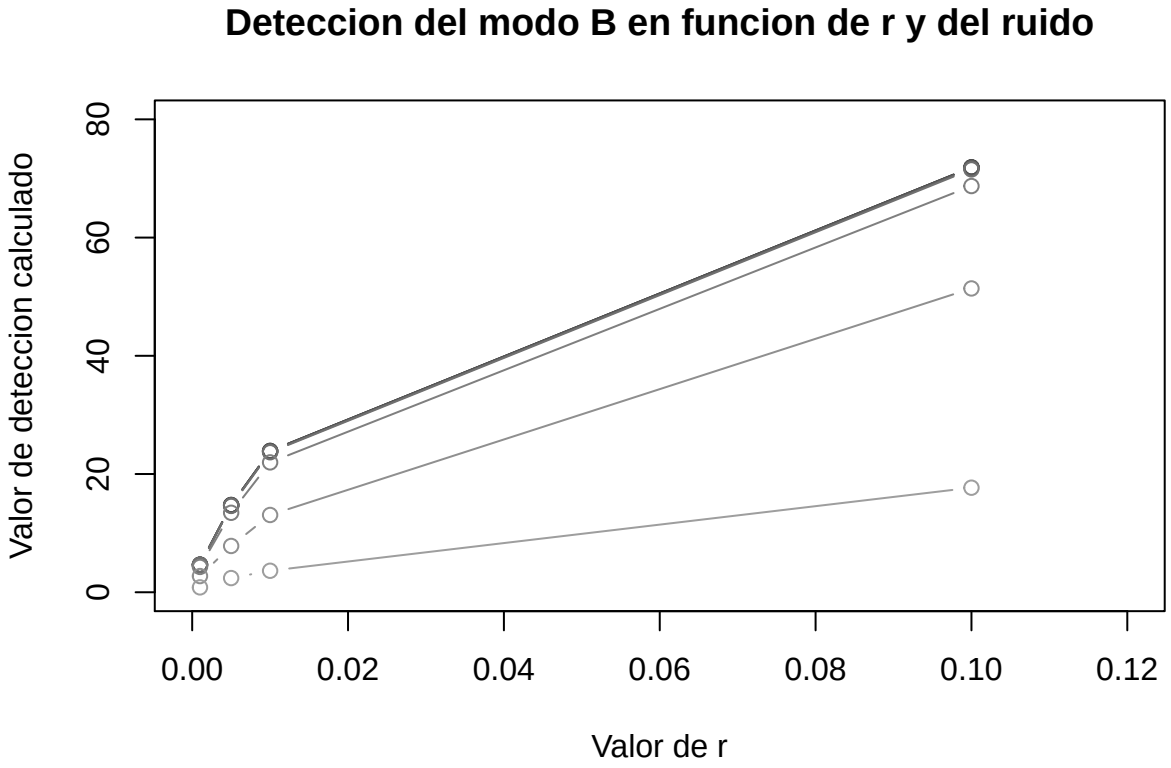
```
     xlim=c(0,0.12), ylim=c(0,5))

for (valor_var in var) {

  deteccion_E <- c()

  for (valor_r in r){

    assign(paste("E_tot", valor_r, sep="_"), (E_scal + valor_r * (E_tens)))

    var2_CMB <- sum((2*l+1)*get(paste("E_tot", valor_r, sep="_")))
    ruido <- (var2_CMB*valor_var)/(sum((2*l+1)))

    E_tot_valor = get(paste("E_tot", valor_r, sep="_"))
    assign(paste("sigma_E", valor_r, sep="_"), (((E_tot_valor+ruido)^2) / (l+0.5)))

    sigma_E_valor <- get(paste("sigma_E", valor_r, sep="_"))
    E_tens_valor <- E_tens
    fisher_E_valor <- sum((E_tens_valor^2)/sigma_E_valor)
    assign(paste("fisher_E", valor_r, sep="_"), fisher_E_valor)

    fisher_E_valor <- get(paste("fisher_E", valor_r, sep="_"))
    error_E_valor <- 1/(sqrt(fisher_E_valor))
    assign(paste("error_E", valor_r, sep="_"), error_E_valor)

    deteccion_E_valor <- (valor_r / get(paste("error_E", valor_r, sep="_")))
    assign(paste("deteccion_E", valor_r, sep="_"), deteccion_E_valor)

    deteccion_E <- append(deteccion_E, get(paste("deteccion_E", valor_r, sep="_")))

  }

  myColor = paste("grey", counter, sep="")
  counter <- counter + 6

  lines(r, deteccion_E, main="Deteccion del modo E en funcion de r y del ruido",
        xlab="Valor de r", ylab="Valor de deteccion calculado",
        col=myColor, type = "b")

}
```



El modo E presenta una variación más grande que el modo T, vemos que añadir este ruido blanco reduce bastante el valor de detección en general. Y acabamos el ejercicio representando al mismo plot, para el modo B.

```
rm(list=ls())

setwd('/Users/ced2718/Documents/Universite/Cosmologia/')
scalar <- read.table("camb_24707701_lensedcls.dat", header=FALSE)
total <- read.table("camb_24707701_lensedtotcls.dat", header=FALSE)

l <- total[,1]

B_scal <- scalar[,4]
B_tot <- total[,4]
B_tens <- (B_tot - B_scal)

#Volvemos a pasar de los coeficientes Dl a Cl
B_tot <- (2 * 3.1415 * B_tot)/(l * (l+1))
B_scal <- (2 * 3.1415 * B_scal)/(l * (l+1))
B_tens <- (2 * 3.1415 * B_tens)/(l * (l+1))

var <- c(0, 0.00001, 0.001, 0.05, 0.1, 1, 10, 100)
r <- c(0.1, 0.01, 0.005, 0.001)

counter <- 20

plot(1, type="n", main="Deteccion del modo B en funcion de r y del ruido",
     xlab="Valor de r", ylab="Valor de deteccion calculado",
     xlim=c(0,0.12), ylim=c(0,80))
```

```
for (valor_var in var) {

  deteccion_B <- c()

  for (valor_r in r){

    assign(paste("B_tot", valor_r, sep="_"), (B_scal + valor_r * (B_tens)))

    var2_CMB <- sum((2*l+1)*get(paste("B_tot", valor_r, sep="_")))
    ruido <- (var2_CMB*valor_var)/(sum((2*l+1)))

    B_tot_valor = get(paste("B_tot", valor_r, sep="_"))
    assign(paste("sigma_B", valor_r, sep="_"), (((B_tot_valor+ruido)^2) / (l+0.5)))

    sigma_B_valor <- get(paste("sigma_B", valor_r, sep="_"))
    B_tens_valor <- B_tens
    fisher_B_valor <- sum((B_tens_valor^2)/sigma_B_valor)
    assign(paste("fisher_B", valor_r, sep="_"), fisher_B_valor)

    fisher_B_valor <- get(paste("fisher_B", valor_r, sep="_"))
    error_B_valor <- 1/(sqrt(fisher_B_valor))
    assign(paste("error_B", valor_r, sep="_"), error_B_valor)

    deteccion_B_valor <- (valor_r / get(paste("error_B", valor_r, sep="_")))
    assign(paste("deteccion_B", valor_r, sep="_"), deteccion_B_valor)

    deteccion_B <- append(deteccion_B, get(paste("deteccion_B", valor_r, sep="_")))

  }

  myColor = paste("grey", counter, sep="")
  counter <- counter + 6

  lines(r, deteccion_B, main="Deteccion del modo B en funcion de r y del ruido",
        xlab="Valor de r", ylab="Valor de deteccion calculado",
        col=myColor, type = "b")

}
```



Vemos que la detección del modo B también presenta una fluctuación bastante grande cuando vamos añadiendo ruido cada vez más importante, como esperado.