TP2-Correcciones-02

July 4, 2019

1 Probabilidad y Estadística (C)

1.1 Trabajo Práctico 2

Alumno: Leandro Carreira

LU: 669/18

Sean $X_1, \ldots, X_n$ una muestra aleatoria con distribución $\mathcal{U}[0, b]$ con b un parámetro desconocido.

1. Calcular analíticamente los estimadores de momentos $\hat{b}_{mom}$ y de máxima verosimilitud $\hat{b}_{mv}$. Implementar estos estimadores en R como funciones.

1.1.1 Estimador de momentos:

Uso el **primer** momento, pues si $b > 0$ con $X_n \sim \mathcal{U}[0, b]$,

$$E[X_n] = \int_0^b x_i * \frac{1}{b} * dx$$

$$= \frac{1}{b} * \left[\frac{x_i^2}{2}\right]_0^b$$

$$= \frac{b^2}{2b}$$

$$E[X_n] = \frac{b}{2}$$

$$\overline{X} = \frac{\hat{b}_{mom}}{2}$$

$$\hat{b}_{mom} = 2 * \overline{X}$$

[262]:
```
# Función estimadora de primeros momentos
bmom1 = function(muestra){
    return(2*mean(muestra))
}
```

[ ]:
```
```

Similarmente se puede calcular el EM con el **segundo** momento, al cual también voy a agregar en los siguientes ejercicios del TP, ya que aporta un tipo de comparación entre estimadores de un mismo tipo, usando dos grados distintos:

$$E[X_n^2] = \int_0^b x_i^2 * \frac{1}{b} * dx$$

$$= \frac{1}{b} * \left[\frac{x_i^3}{3}\right]_0^b$$

$$= \frac{b^3}{3b}$$

$$E[X_n^2] = \frac{b^2}{3}$$

$$b^2 = 3 * E[X_n^2]$$

$$b > 0$$

$$b = \sqrt{3 * E[X_n^2]}$$

$$\hat{b}_{mom2} = \sqrt{3 * \overline{X^2}}$$

[263]:
```
# Función estimadora de segundos momentos
bmom2 = function(muestra){
    n <- length(muestra)
    return(sqrt(3 * mean(muestra^2)))
}
```

1.1.2 Estimador de Máxima Verosimilitud

Estimador de Máxima Verosimilitud (EMV)

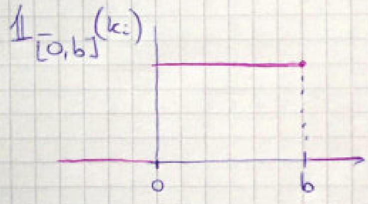
Sean $X_1, \dots, X_n$ una muestra iid $\sim \mathcal{U}[\overset{a}{0},b]$

$$P(X_1=k_1, X_2=k_2, \dots, X_n=k_n) \overset{iid}{=} P(X_1=k_1)\cdot \ldots \cdot P(X_n=k_n)$$

$$\overset{iid}{=} \prod_{i=1}^{n} P(X_i=k_i) \overset{X_i\sim\mathcal{U}}{=} \prod_{i=1}^{n} \frac{1}{b-a}\cdot \mathbb{1}_{[a,b]}(k_i) =$$

$$\overset{a=0}{=} \prod_{i=1}^{n} \frac{1}{b}\cdot \mathbb{1}_{[0,b]}(k_i) \quad (1)$$

• Ahora, quiero expresar $\mathbb{1}_{[0,b]}(k_i)$ en función de b, ya que es b que quiero despejar:



$$\begin{cases} 1 & \text{si } 0 \leq k_i \leq b \\ 0 & \text{si no} \end{cases}$$

$$\Rightarrow \begin{cases} \text{si } b \geq k_i & 1 \\ \text{si } b < k_i & 0 \end{cases}$$

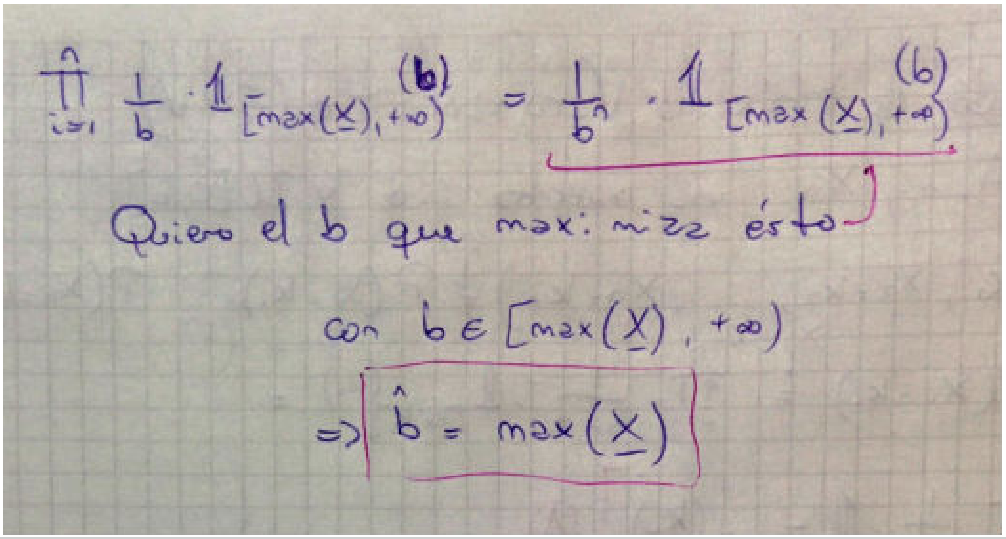
$$\Rightarrow \mathbb{1}_{[k_i,+\infty)}(b) = \mathbb{1}_{[0,b]}(k_i)$$

Reescribo (1):

$\prod_{i=1}^{n} \frac{1}{b}\cdot \mathbb{1}_{[k_i,+\infty)}(b)$ De aquí veo que para todo i, debe darse que $b \geq k_i$, pues si alguno es menor, la indicadora 0, y por ende la productoria será cero:

$$\Rightarrow b \geq \max(k_i) = \max(\underset{\sim}{X})$$

↑ muestra.



[264]:
```
# Estimador de maxima verosimilitud
bmv = function(muestra){
    return(max(muestra))
}
```

2. Implementar el siguiente estimador de b

$$\hat{b}_{med} = 2 \times \text{mediana}\{X_1, \ldots, X_n\}$$

[265]:
```
bmed = function(muestra){
    return(2*median(muestra))
}
```

3. Utilizando $b = 1$, generar una muestra de tamaño $n = 15$. Calcular cada uno de los estimadores con la muestra obtenida y reportar el valor de cada estimador y su error.

[266]:
```
b <- 1
n <- 15
muestra <- runif(n, min=0, max=b)
```

1.1.3 Valores estimados:

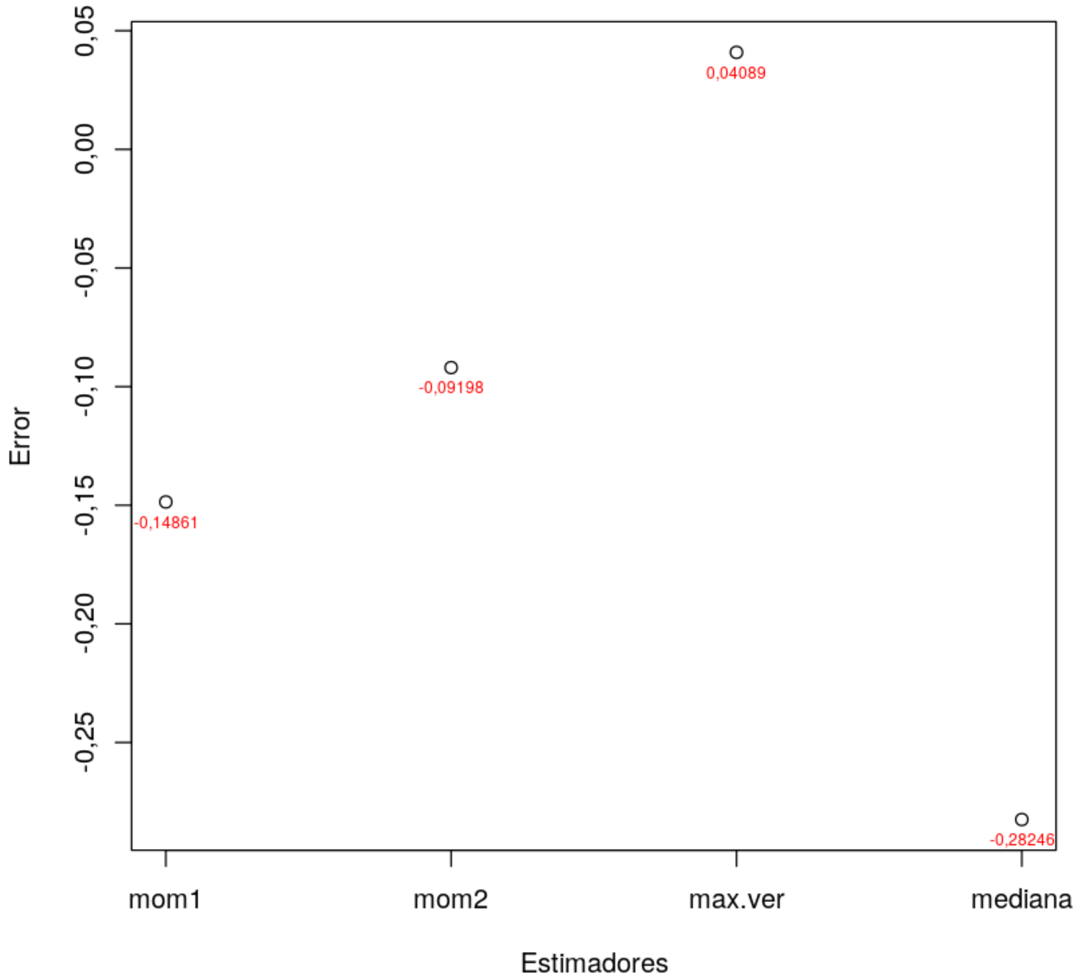
[267]:
```
bmom1(muestra)
bmom2(muestra)
bmv(muestra)
bmed(muestra)
```

1,14860809737196
1,09198123777567
0,959108822746202
1,28245940711349

1.1.4 Errores:

[268]:
```
# Calculo errores
error_momento_1 <- b - bmom1(muestra)
error_momento_2 <- b - bmom2(muestra)
error_max_ver   <- b - bmv(muestra)
error_mediana   <- b - bmed(muestra)
# Agrupo datos para plot
errores <- c(error_momento_1, error_momento_2, error_max_ver,error_mediana)
nombres <- c('mom1','mom2','max.ver','mediana')
# Imprimo y ploteo errores para una mejor comparación
matrix(c(nombres, round(errores,5)), nrow=2, ncol=4, byrow=TRUE)
# Plot
#options(repr.plot.width=7, repr.plot.height=7)
plot(errores, xlab="Estimadores", ylab="Error", xaxt='n')
text(errores, as.character(round(errores,5)), cex=0.6, pos=1, col="red")
axis(1, c(1,2,3,4), nombres)
```

mom1	mom2	max.ver	mediana
-0,14861	-0,09198	0,04089	-0,28246



4. Hacer una simulación para obtener el sesgo, varianza y error cuadratico medio (ECM) de cada uno de los estimadores. Para lograr esto:
   a) Generar una muestra con $b = 1$, $n = 15$.
   b) Para la muestra obtenida, calcular $\hat{b}_{mv}$, $\hat{b}_{mom}$, $\hat{b}_{med}$ y almacenar los resultados.
   c) Repetir $N_{rep} = 1000$ veces los pasos (a) y (b).
   d) Obtener una aproximación del sesgo restando el valor verdadero de b a la media muestral de cada estimador.
   e) Obtener la aproximación de la varianza a partir de la varianza muestral de cada estimador.
   f) Obtener la aproximación del ECM a través de la fórmula que lo relaciona con el sesgo y la varianza.

[269]:
```
experimento = function(){
    # a)
    b <- 1
```

```
    n <- 15
    muestra <- runif(n, min=0, max=b)
    # b)
    b_mom1 <- bmom1(muestra)
    b_mom2 <- bmom2(muestra)
    b_mv  <- bmv(muestra)
    b_med <- bmed(muestra)
    #devuelvo un vector de estimadores
    c(b_mom1, b_mom2, b_mv, b_med)
}
```

[270]:
```
# c)
nrep <- 1000
estimadores <- array(dim=c(nrep,4), dimnames=list(1:nrep, c("b_mom1", "b_mom2",␣
 ↪"b_mv", "b_med")))
for(i in 1:nrep){
    estimadores[i,] <- array(experimento())
}
```

[271]:
```
# Estimaciones guardadas de cada experimento
estimadores[2:4,]
estimadores[997:1000,]
```

	b_mom1	b_mom2	b_mv	b_med
2	1,118143	1,076959	0,9528492	1,1767766
3	1,011031	1,023854	0,9461619	0,9386101
4	1,063968	1,041872	0,9898384	1,0918731

	b_mom1	b_mom2	b_mv	b_med
997	1,026969	1,0208446	0,9965344	1,0133310
998	1,020018	0,9155019	0,7713337	1,0720456
999	1,150225	1,1232409	0,9875382	1,2293783
1000	1,078289	1,0540542	0,9562838	0,9927551

Sesgo:

[272]:
```
# d)
# aplico mean a cada columna (estimador) de mi data
b_muestrales <- apply(estimadores, MARGIN=2, FUN=mean)
print(b_muestrales)
```

```
   b_mom1    b_mom2      b_mv     b_med
1,0058257 0,9985216 0,9400177 1,0109403
```

[273]:
```
#sesgos <- medias_muestrales - b
b <- 1
sesgos <- b_muestrales - b
```

[274]:
```
print(sesgos)
```

```
     b_mom1       b_mom2         b_mv        b_med
0,005825672 -0,001478435 -0,059982265  0,010940324
```

Varianza muestral: Uso estimador insesgado: $S^2 = \frac{\sum(X_i - \hat{\mu})^2}{n-1}$

[275]:
```r
# e)
varianzas_muestrales <- apply(estimadores, MARGIN=2, FUN=var)
```

[276]:
```r
print(varianzas_muestrales)
```

```
     b_mom1      b_mom2        b_mv       b_med
0,022926936 0,014079338 0,003537332 0,062827929
```

Error Cuadrático Medio:

[277]:
```r
# f) Aproximación del Error Cuadratico Medio (ECM)
ECM <- varianzas_muestrales + sesgos^2
print(ECM)
```

```
     b_mom1      b_mom2        b_mv       b_med
0,022960875 0,014081524 0,007135204 0,062947620
```

5. Implementar las funciones $simulacion\_mv(b, n)$, $simulacion\_mom(b, n)$ y $simulacion\_med(b, n)$ que devuelven una aproximación del sesgo y de la varianza de cada uno de los estimadores correspondientes al b y al n.

[278]:
```r
# Funciones simuladoras:
# Devuelven sesgo y varianza aproximados
# promediando 1000 experimentos
# con Estimador de Maxima Verosimilitud
simulacion_mv = function(b, n){
    nE <- 1000
    # Guardo todas las estimaciones
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        b_est <- bmv(muestra)
        # Guardo b estimado para calcular Sesgo/Var luego
        all_b_est[i] <- b_est
    }
    # Calculo Sesgo y Varianza usando todas las muestras
    sesgo_est <- mean(all_b_est) - b
    # Calculo varianza muestral, usando b estimados
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}

# con Estimador de 1er Momento
simulacion_mom = function(b, n){
```

```
    nE <- 1000
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        b_est <- bmom1(muestra)
        all_b_est[i] <- b_est
    }
    sesgo_est <- mean(all_b_est) - b
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}

# Agrego también simulación de 2do momento
simulacion_mom2 = function(b, n){
    nE <- 1000
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        b_est <- bmom2(muestra)
        all_b_est[i] <- b_est
        }
    sesgo_est <- mean(all_b_est) - b
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}

# con Mediana de la muestra
simulacion_med = function(b, n){
    nE <- 1000
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        b_est <- bmed(muestra)
        all_b_est[i] <- b_est
        }
    sesgo_est <- mean(all_b_est) - b
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}
```

6. Comparar mediante gráficos, el sesgo, la varianza y el ECM de cada estimador con $n = 15$ y $0 < b < 2$. ¿Qué observa? ¿Qué estimador elige?

[279]:

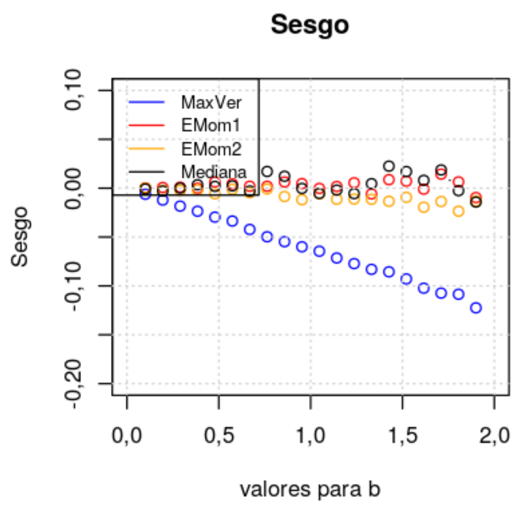
```
# Calculo sesgos, varianzas y ECM para 20 valores
# distintos de b entre 0 y 2 (no inclusives)
```

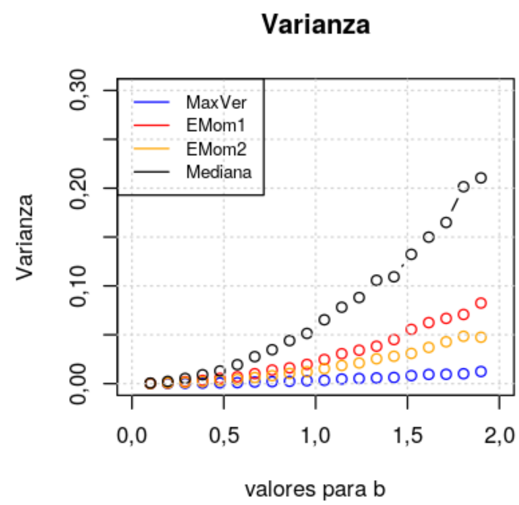
```
nB <- 20
b_values <- seq(0.1, 1.9, by=1.8/(nB-1))
# nB filas, 3 columnas: (bias, var, ECM)
results_mv   <- matrix(nrow=nB, ncol=3)
results_mom  <- matrix(nrow=nB, ncol=3)
results_mom2 <- matrix(nrow=nB, ncol=3)
results_med  <- matrix(nrow=nB, ncol=3)

for (i in 1:nB){
    b <- b_values[i]
    results_mv[i,1:2]   <- simulacion_mv(b, 15)
    results_mom[i,1:2]  <- simulacion_mom(b, 15)
    results_mom2[i,1:2] <- simulacion_mom2(b, 15)
    results_med[i,1:2]  <- simulacion_med(b, 15)
    # ECM = Var + Sesgo^2
    results_mv[i,3]   <- results_mv[i,1]^2   + results_mv[i,2]
    results_mom[i,3]  <- results_mom[i,1]^2  + results_mom[i,2]
    results_mom2[i,3] <- results_mom2[i,1]^2 + results_mom2[i,2]
    results_med[i,3]  <- results_med[i,1]^2  + results_med[i,2]
}
```

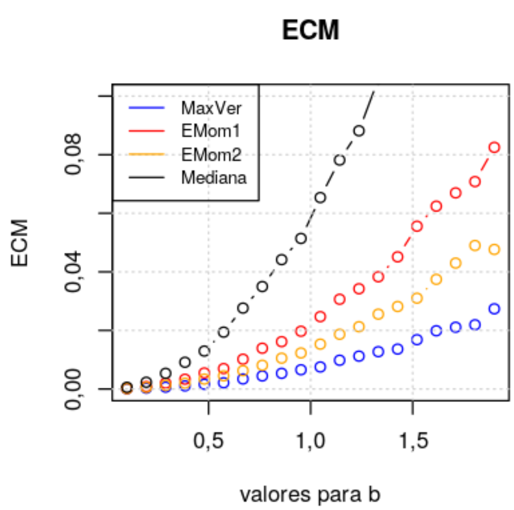
[280]:

```
par(mfrow=c(2,2))
# Sesgos
plot(b_values, results_mv[,1], ylim=c(-0.2,0.1), xlim=c(0,2),
     col="blue", main="Sesgo", xlab="valores para b", ylab="Sesgo", type="b")
points(b_values, results_mom[,1], col="red", type="b")
points(b_values, results_mom2[,1], col="orange", type="b")
points(b_values, results_med[,1], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topleft", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2",␣
↪"Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
# Varianzas
plot(b_values, results_mv[,2], ylim=c(0,0.3), xlim=c(0,2),
     col="blue", main="Varianza", xlab="valores para b", ylab="Varianza",␣
↪type="b")
points(b_values, results_mom[,2], col="red", type="b")
points(b_values, results_mom2[,2], col="orange", type="b")
points(b_values, results_med[,2], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topleft", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2",␣
↪"Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
```

```r
# ECM = Var + Sesgo^2
plot(b_values, results_mv[,3], col="blue", main="ECM", xlab="valores para b", ylab="ECM", type="b", ylim=c(0.0,0.1))
points(b_values, results_mom[,3], col="red", type="b")
points(b_values, results_mom2[,3], col="orange", type="b")
points(b_values, results_med[,3], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topleft", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2", "Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
```







1.1.5 Observaciones:

- A medida que aumento b (manteniendo el tamaño de muestra), los estimadores de momento y mediana aumentan tanto varianza como ECM
- El estimador de máxima verosimilitud muestra una varianza de un orden mucho menor, aunque un sesgo que aumenta de manera negativa.

Esto último es de esperarse ya que para un b cercano a cero, los valores que serán simulados en la muestra estarán muy acotados, mientras que al incrementar b, podrán aparecer valores más grandes en la muestra, y por ende, haber diferencias más grandes en las estimaciones.

1.1.6 Decisiones:

- El criterio para decidir qué estimador elegir será seleccionar el de menor ECM, en este caso, el estimador de máxima verosimilitud.

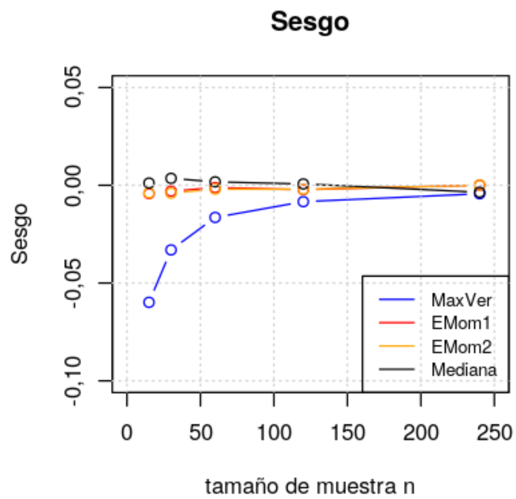
7. Realizar un grafico de los ECM con $b = 1$ y $n = 15, 30, 60, 120, 240$. ¿Qué observa? ¿Qué estimador elige? ¿Que sospecha sobre la consistencia de los estimadores?

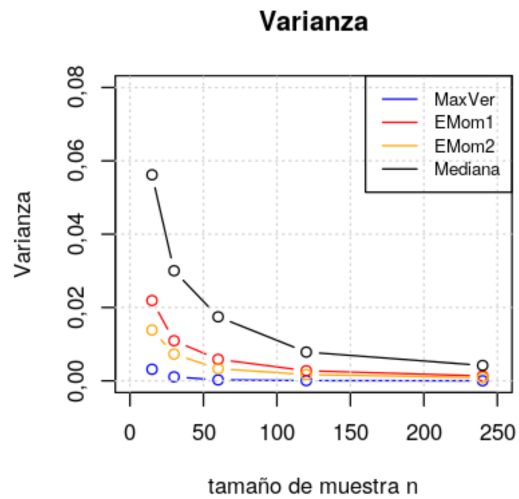
[281]:
```
# Calculo sesgos, varianzas para distintos valores de n
n_values <- c(15, 30, 60, 120, 240)
nN <- length(n_values)
# nN filas, 4 columnas: (n, Sesgo, Var, ECM)
results_mv   <- matrix(nrow=nN, ncol=4)
results_mom  <- matrix(nrow=nN, ncol=4)
results_mom2 <- matrix(nrow=nN, ncol=4)
results_med  <- matrix(nrow=nN, ncol=4)

for (i in 1:nN){
    n <- n_values[i]
    # Guardo n en [1]
    results_mv[i,1]   <- n
    results_mom[i,1]  <- n
    results_mom2[i,1] <- n
    results_med[i,1]  <- n
    # Realizo 1000 simulaciones y guardo
    # Sesgos[2] y Varianzas[3] para graficarlos
    results_mv[i,2:3]   <- simulacion_mv(1, n)
    results_mom[i,2:3]  <- simulacion_mom(1, n)
    results_mom2[i,2:3] <- simulacion_mom2(1, n)
    results_med[i,2:3]  <- simulacion_med(1, n)
    # ECM[4] = Sesgo^2 + Var
    results_mv[i,4]   <- results_mv[i,2]^2   + results_mv[i,3]
    results_mom[i,4]  <- results_mom[i,2]^2  + results_mom[i,3]
    results_mom2[i,4] <- results_mom2[i,2]^2 + results_mom2[i,3]
    results_med[i,4]  <- results_med[i,2]^2  + results_med[i,3]
}
```

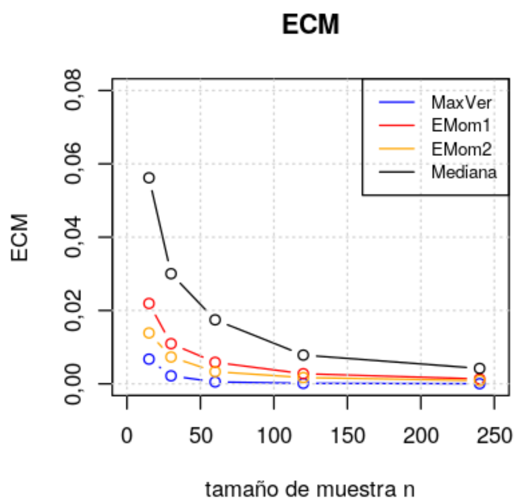
[282]:
```
results_mv
# n   Sesgo     Varianza   ECM
```

15	-0,059862200	3,150367e-03	6,733850e-03
30	-0,033054421	1,081743e-03	2,174338e-03
60	-0,016438013	2,470029e-04	5,172112e-04
120	-0,008375468	6,561827e-05	1,357667e-04
240	-0,004326316	1,930231e-05	3,801932e-05

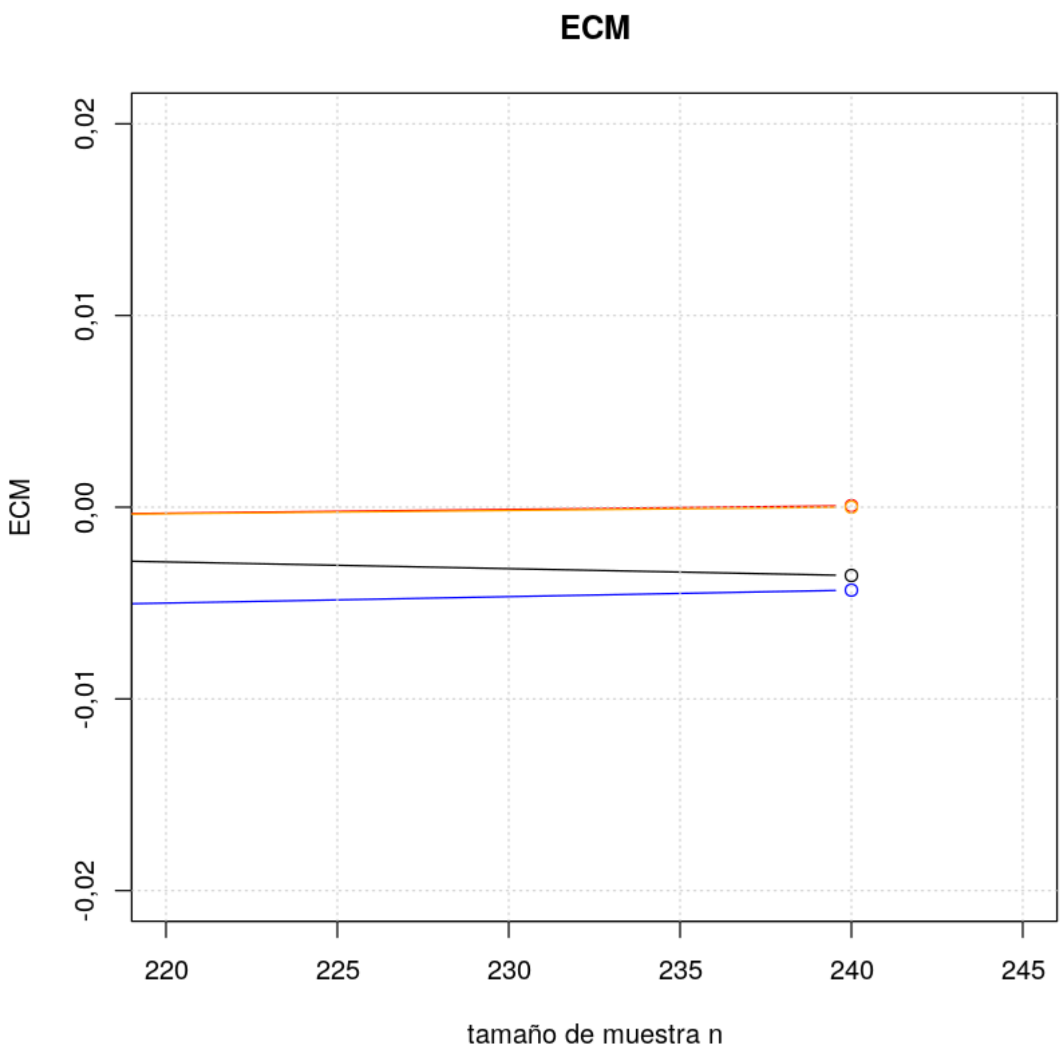
[283]:
```
par(mfrow=c(2,2))
# Sesgos
plot(results_mv[,c(1,2)], ylim=c(-0.10,0.05), xlim=c(0,250),
     col="blue", main="Sesgo", xlab="tamaño de muestra n", ylab="Sesgo",␣
 ↪type="b")
points(results_mom[,c(1,2)], col="red", type="b")
points(results_mom2[,c(1,2)], col="orange", type="b")
points(results_med[,c(1,2)], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("bottomright", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2",␣
 ↪"Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
# Varianzas
plot(results_mv[,c(1,3)], ylim=c(0.0,0.08), xlim=c(0,250),
     col="blue", main="Varianza", xlab="tamaño de muestra n", ylab="Varianza",␣
 ↪type="b")
points(results_mom[,c(1,3)], col="red", type="b")
points(results_mom2[,c(1,3)], col="orange", type="b")
points(results_med[,c(1,3)], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topright", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2",␣
 ↪"Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
# ECM
plot(results_mv[,c(1,4)], col="blue", main="ECM", xlab="tamaño de muestra n",␣
 ↪ylab="ECM", type="b", ylim=c(0.0,0.08), xlim=c(0,250))
points(results_mom[,c(1,4)], col="red", type="b")
points(results_mom2[,c(1,4)], col="orange", type="b")
points(results_med[,c(1,4)], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topright", bg=transpa_color, legend=c("MaxVer", "EMom1", "EMom2",␣
 ↪"Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
```







[284]:
```
# Zoom en mayor n alcanzado
plot(results_mv, ylim=c(-0.02, 0.02), xlim=c(220,245),
     col="blue", main="ECM", xlab="tamaño de muestra n", ylab="ECM", type="b")
points(results_mom, col="red", type="b")
points(results_mom2, col="orange", type="b")
points(results_med, col="black", type="b")
grid()
```



1.1.7 Observaciones:

- Todos los estimadores muestran una tendencia a converger a cero (al menos de manera asintótica) a medida que se aumenta el tamaño de la muestra. Ésto indica que todos los estimadores son **consistentes**.

De tener un tamaño de muestra grande, cualquier estimador devolvería buenas estimaciones, mientras que para tamaños de muestra más pequeños, lo más acertado sería usar el **estimador de máxima verosimilitud**, que mantiene el ECM más bajo de todos los estimadores desde el tamaño de muestra más pequeño hasta el mayor.

8. Calcular los estimadores en la siguiente muestra. ¿Observa algo extraño? ¿A qué cree que se debe?

   0,917 0,247 0,384 0,530 0,798 0,912 0,096 0,684 0,394 20,1 0,769 0,137 0,352 0,332 0,670

[285]:
```
X <- c(0.917, 0.247, 0.384, 0.530, 0.798,
       0.912, 0.096, 0.684, 0.394, 20.1,
       0.769, 0.137, 0.352, 0.332, 0.670)
```

[286]:
```
b_mv <- bmv(X)
b_mom <- bmom1(X)
b_mom2 <- bmom2(X)
b_med <- bmed(X)
```

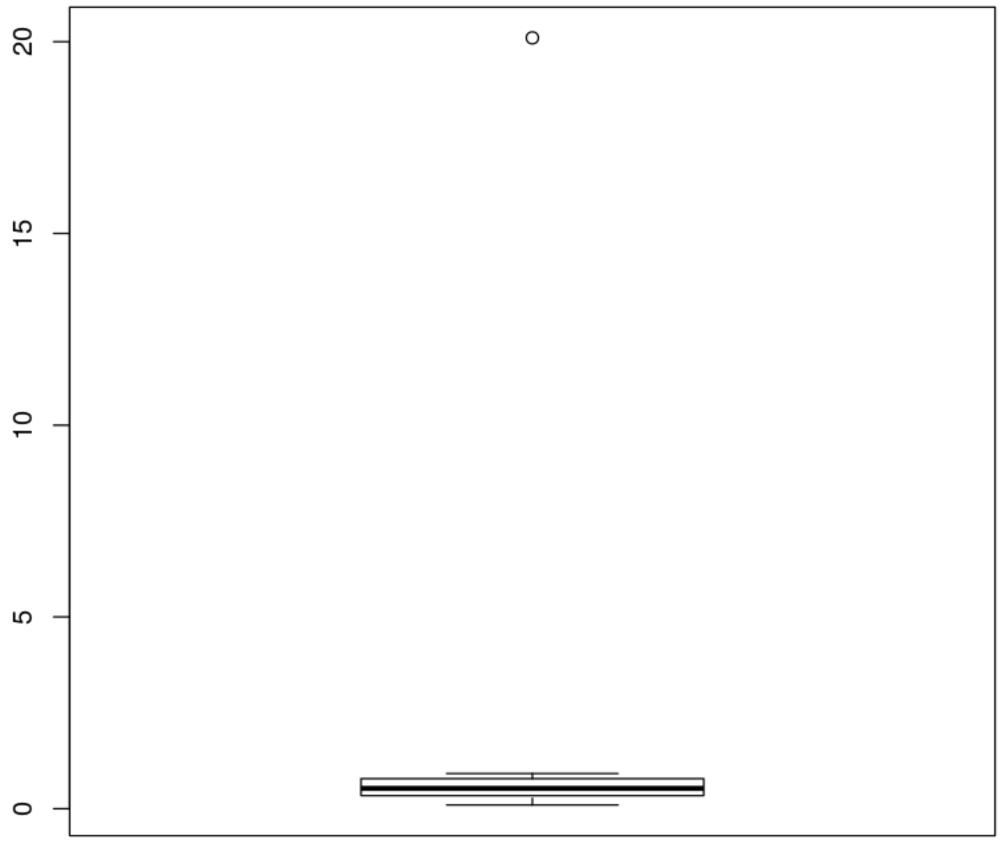
[287]:
```
b_mv
b_mom
b_mom2
b_med
```

20,1
3,64293333333333
9,04139666202075
1,06

1.1.8 Observaciones:

- Tanto revisando la data como gradicando un boxplot, se ve que el error es debido a un outlier de un valor 40 veces mayor al resto de la muestra.

[288]:
```
boxplot(X)
```



- Una vez **comprobado** que sea un outlier, podemos 'reparar' nuestra muestra eliminando el valor fuera de rango, y calcular los estimadores con el resto de la muestra, como se realiza a continuación:

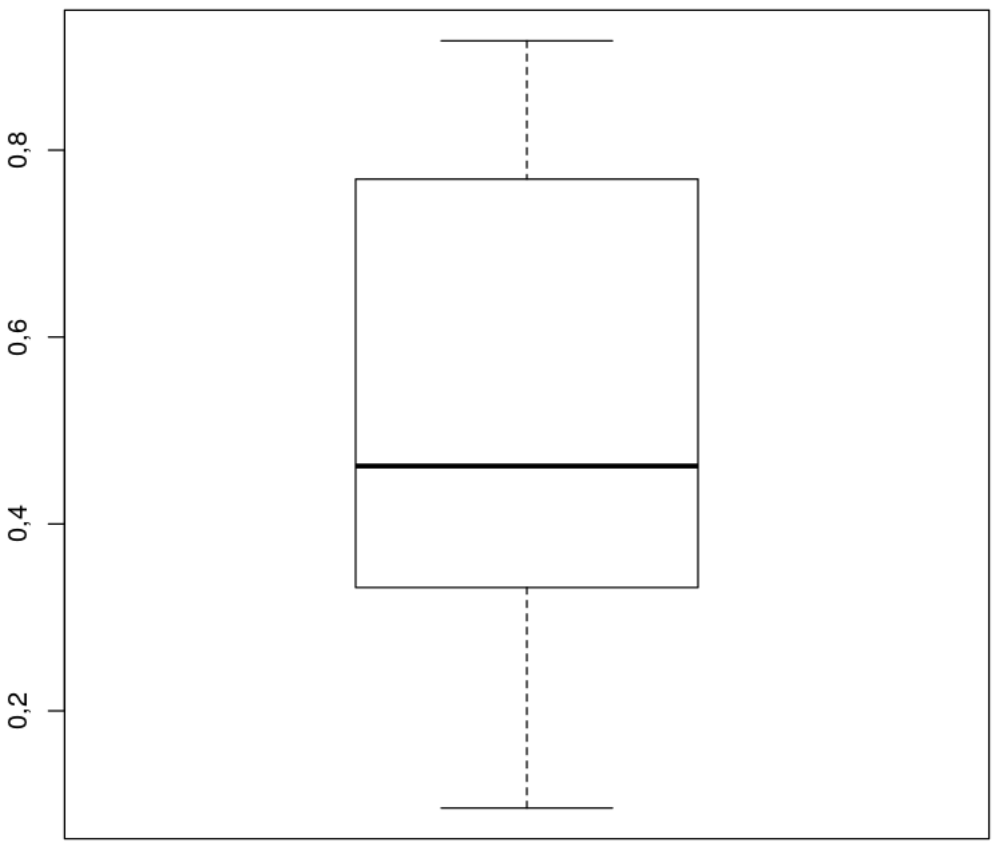
[289]:
```
X_fixed <- c(0.917, 0.247, 0.384, 0.530, 0.798,
       0.912, 0.096, 0.684, 0.394,
       0.769, 0.137, 0.352, 0.332, 0.670)
```

[290]:
```
b_mv <- bmv(X_fixed)
b_mom <- bmom1(X_fixed)
b_mom2 <- bmom2(X_fixed)
b_med <- bmed(X_fixed)
```

[291]:
```
b_mv
b_mom
b_mom2
b_med
```

0,917
1,03171428571429
1,006152643915
0,924

[292]:
```
boxplot(X_fixed)
```



9. Aproximar sesgo, varianza y error cuadrático medio para los estimadores bajo el siguiente escenario con datos contaminados:
Una muestra uniforme con $b = 1$ y $n = 15$ donde de manera independiente, a cada elemento se lo multiplica por 100 con probabilidad 0,005. (Correr la coma dos lugares a la derecha).

   a) Calcular la probabilidad de que una muestra esté contaminada.
   b) Reportar las aproximaciones obtenidas.
   c) ¿Qué estimador prefiere en este escenario?

[293]:
```
n <- 15
b <- 1
X <- runif(n, 0, b)
```

[294]:
```
# Minicódigo a implementar en funciones simuladoras
X_cont <- X
# Contamino cada elemento con proba 0.005
for(i in 1:n){
    pC <- 0.005 # 1/200
    if(runif(1) < pC){

        X_cont[i] <- X_cont[i] * 100
    }
}

# De manera más eficiente (y bonita :)
pC <- 1/200
mask <- runif(n)
X_cont[mask<pC] <- X_cont[mask<pC] * 100
```

a) Probabilidad de que la muestra esté contaminada: Sea C el evento Bernoulli: "La muestra está contaminada"

La probabilidad de que la muestra esté contaminada $P(C)$ será 1 menos la probabilidad de que **no** esté contaminada.

$P(C) = 1 - P($ "muestra no contaminada" $)$

Cada elemento de la muestra tiene probabilidad $p = \frac{1}{200} = 0.005$ de ser contaminado.

La probabilidad de que **cada elemento** no esté contaminado será de $1 - p = \frac{199}{200} = 0.995$.

La probabilidad de que **la muestra** no esté contaminada será el producto de las probabilidades de cada elemento de la muestra:

$P($ "muestra no contaminada" $) = (\frac{199}{200})^n$

siendo n la cantidad de elementos de la muestra, sabemos que $n = 15$

$P($ "muestra no contaminada" $) = (\frac{199}{200})^{15} \approx 0.927569$

por lo que la probabilidad de que la muestra esté contaminada es de:

$P(C) = 1 - (\frac{199}{200})^{15}$

$P(C) \approx 1 - 0.927569$

$P(C) \approx 0.072431$

$P(C) \approx 7.24\%$

b) Aproximaciones obtenidas:

[295]:
```
# Funciones simuladoras CON CONTAMINACIÓN:
# Devuelven sesgo y varianza aproximados
# promediando 1000 experimentos
# con Estimador de Maxima Verosimilitud
simulacion_mv_cont = function(b, n){
    nE <- 1000
    # Guardo todas las estimaciones
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        # --[Contaminación con proba pC]--
        pC <- 1/200
        mask <- runif(n)
        muestra[mask<pC] <- muestra[mask<pC] * 100
        # --[Fin contaminación]--
        b_est <- bmv(muestra)
        # Guardo b estimado para calcular Sesgo luego
        all_b_est[i] <- b_est
    }
    # Calculo Sesgo y Varianza usando todas las muestras
    sesgo_est <- mean(all_b_est) - b
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}

# con Estimador de 1er Momento
simulacion_mom_cont = function(b, n){
    nE <- 1000
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        # --[Contaminación con proba pC]--
        pC <- 1/200
        mask <- runif(n)
        muestra[mask<pC] <- muestra[mask<pC] * 100
        # --[Fin contaminación]--
        b_est <- bmom1(muestra)
        all_b_est[i] <- b_est
    }
    sesgo_est <- mean(all_b_est) - b
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}

# Agrego también simulación de 2do momento
```

```
simulacion_mom2_cont = function(b, n){
    nE <- 1000
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        # --[Contaminación con proba pC]--
        pC <- 1/200
        mask <- runif(n)
        muestra[mask<pC] <- muestra[mask<pC] * 100
        # --[Fin contaminación]--
        b_est <- bmom2(muestra)
        all_b_est[i] <- b_est
        }
    sesgo_est <- mean(all_b_est) - b
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}

# con Mediana de la muestra
simulacion_med_cont = function(b, n){
    nE <- 1000
    all_b_est <- vector(length=nE)
    varianza <- vector(length=nE)
    for (i in 1:nE){
        muestra <- runif(n, min=0, max=b)
        # --[Contaminación con proba pC]--
        pC <- 1/200
        mask <- runif(n)
        muestra[mask<pC] <- muestra[mask<pC] * 100
        # --[Fin contaminación]--
        b_est <- bmed(muestra)
        all_b_est[i] <- b_est
        }
    sesgo_est <- mean(all_b_est) - b
    varianza_est <- var(all_b_est)
    return(c(sesgo_est, varianza_est))
}
```

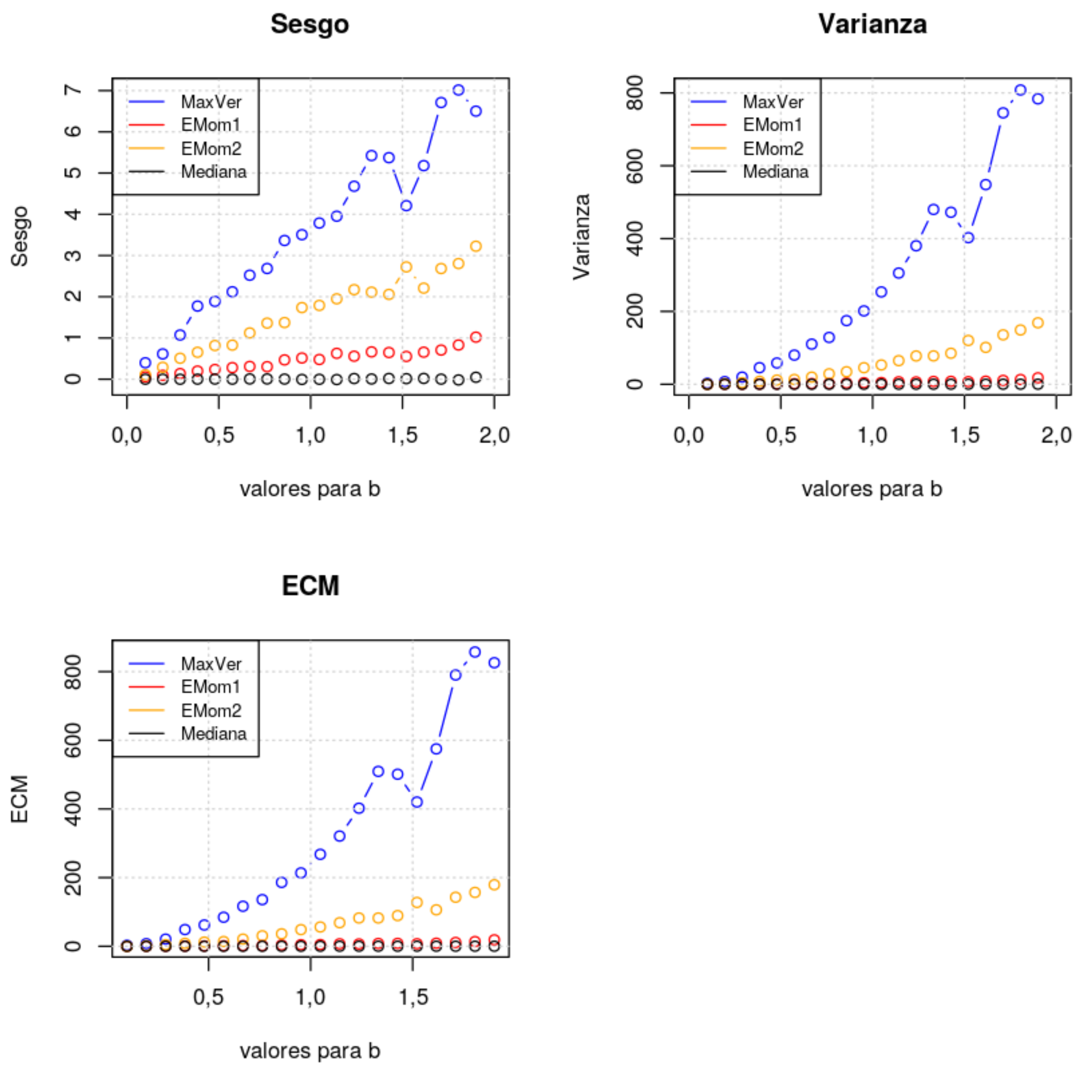
[296]:
```
# Calculo sesgos, varianzas y ECM para 20 valores
# distintos de b entre 0 y 2 (no inclusives)
nB <- 20
b_values <- seq(0.1, 1.9, by=1.8/(nB-1))
# nB filas, 3 columnas: (bias, var, ECM)
results_mv   <- matrix(nrow=nB, ncol=3)
results_mom  <- matrix(nrow=nB, ncol=3)
results_mom2 <- matrix(nrow=nB, ncol=3)
```

```
results_med  <- matrix(nrow=nB, ncol=3)

for (i in 1:nB){
    b <- b_values[i]
    # Guardo Sesgo y Varianza para cada b
    results_mv[i,1:2]   <- simulacion_mv_cont(b, 15)
    results_mom[i,1:2]  <- simulacion_mom_cont(b, 15)
    results_mom2[i,1:2] <- simulacion_mom2_cont(b, 15)
    results_med[i,1:2]  <- simulacion_med_cont(b, 15)
    # Calculo ECM = Sesgo^2 + Var
    results_mv[i,3] <- results_mv[i,1]^2+results_mv[i,2]
    results_mom[i,3] <- results_mom[i,1]^2+results_mom[i,2]
    results_mom2[i,3] <- results_mom2[i,1]^2+results_mom2[i,2]
    results_med[i,3] <- results_med[i,1]^2+results_med[i,2]
}
```

[297]:

```
par(mfrow=c(2,2))
# Plot para Sesgos
ylim <- c(-0.1, max(results_mv[,1]))
plot(b_values, results_mv[,1], xlim=c(0,2), ylim=ylim,
     col="blue", main="Sesgo", xlab="valores para b", ylab="Sesgo", type="b")
points(b_values, results_mom[,1], col="red", type="b")
points(b_values, results_mom2[,1], col="orange", type="b")
points(b_values, results_med[,1], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topleft", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2", "Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
# Plot para Varianzas
plot(b_values, results_mv[,2], xlim=c(0,2),
     col="blue", main="Varianza", xlab="valores para b", ylab="Varianza", type="b")
points(b_values, results_mom[,2], col="red", type="b")
points(b_values, results_mom2[,2], col="orange", type="b")
points(b_values, results_med[,2], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topleft", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2", "Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
# Plot para ECM
plot(b_values, results_mv[,3], col="blue", main="ECM", xlab="valores para b", ylab="ECM", type="b")
points(b_values, results_mom[,3], col="red", type="b")
```

```
points(b_values, results_mom2[,3], col="orange", type="b")
points(b_values, results_med[,3], col="black", type="b")
grid()
transpa_color <- rgb(0, 0, 0, max = 255, alpha = 0, names = "transparent")
legend("topleft", bg=transpa_color,legend=c("MaxVer", "EMom1", "EMom2", "Mediana"),
       col=c("blue", "red", "orange", "black"), lty=1, cex=0.8,
       box.lty=1)
```



c) Qué estimador prefiero?

- En este caso la muestra está contaminada de una forma particular: con esporádicos valores muy por encima de la media.

Se observa en los gráficos que el estimador de Máxima Verosimilitud es **muy** sensible a outliers, dado que utiliza el máximo valor de cada muestra como estimación de b, ignorando todos los otros valores.

Ésto resulta en estimaciones catastróficas, dado que por más que se tenga una muestra con 10 millones de valores cercanos a 1.0, y un único outlier muy por encima de este número, el EMV solo usará la información de este último, errando por un gran márgen su estimación.

Los otros tres estimadores (ambos de Momentos, y Mediana) resultan ser más consistentes con el resto de la data.

El estimador de Mediana muestra gran inmunidad a este tipo de contaminación con respecto al sesgo.

El estimador de 2do momento muestra un peor desempeño que el de primer momento, tanto en sesgo, varianza, como ECM.

Por lo anterior mencionado, en este caso, elegiría el **Estimador de Mediana**, o el de **Momento 1**.

`[Fin del tp]`

[ ]:
```

```