

# Análisis de regresión

Capítulo 5: Valores atípicos e influenciales

Mg. Sc. J. Eduardo Gamboa U.



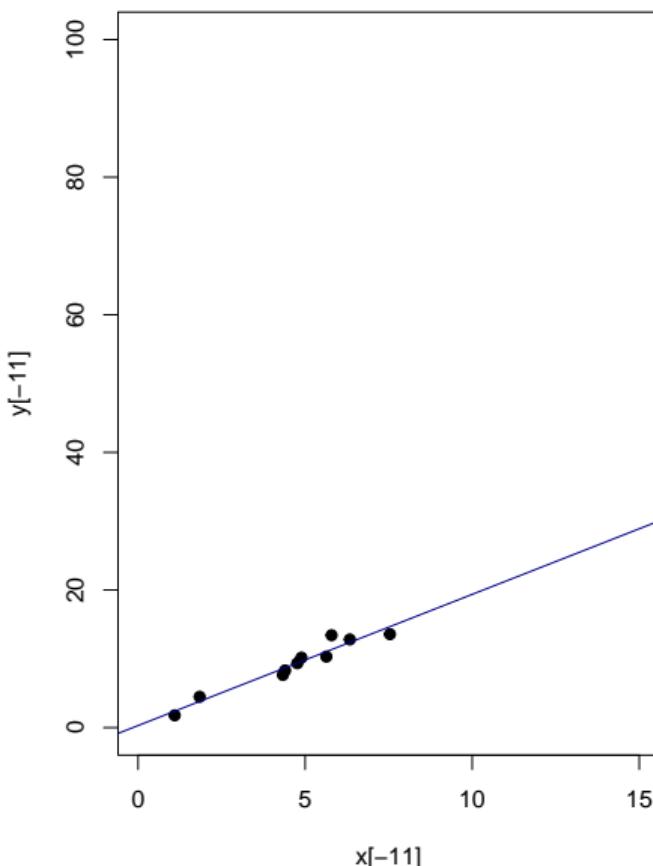
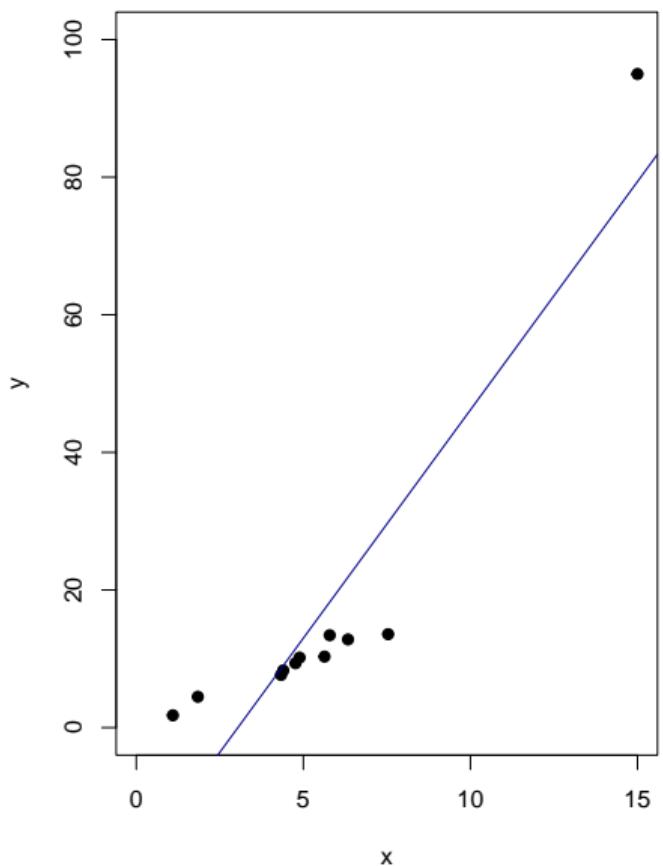
## Introducción

- ▶ Un outlier es una observación que no se ajusta a un determinado modelo. En regresión lineal, diríamos que “se aleja de la línea de regresión estimada”.
- ▶ Un leverage es un valor inusual que podría controlar o modificar ciertas propiedades del modelo. Para la regresión lineal simple correspondería a un valor inusual de la variable predictora.
- ▶ Un valor influencial es aquel cuya presencia altera las estimaciones u otras propiedades del modelo, por ejemplo: signos de los coeficientes de regresión, no significancia de variables importantes, intervalos de predicción amplios; en general, el proceso de inferencia

## Ejemplo 1

Outlier, leverage y valor influencial

```
set.seed(111)
x = c(runif(10,1,10),15)
y = c(rnorm(10,2*x,1),95)
par(mfrow=c(1,2))
plot(x,y, ylim= c(0,100), xlim = c(0,15), pch=19)
abline(lm(y~x), col="darkblue")
plot(x[-11],y[-11], ylim= c(0,100), xlim = c(0,15), pch=19)
abline(lm(y[-11]~x[-11]), col="darkblue")
```



Note el cambio en los coeficientes de regresión

```
library(broom)
lm(y~x) |> tidy()
```

```
# A tibble: 2 x 5
  term      estimate std.error statistic p.value
  <chr>      <dbl>     <dbl>      <dbl>    <dbl>
1 (Intercept) -20.2      6.18     -3.27 0.00975
2 x             6.64      0.939     7.07 0.0000588
```

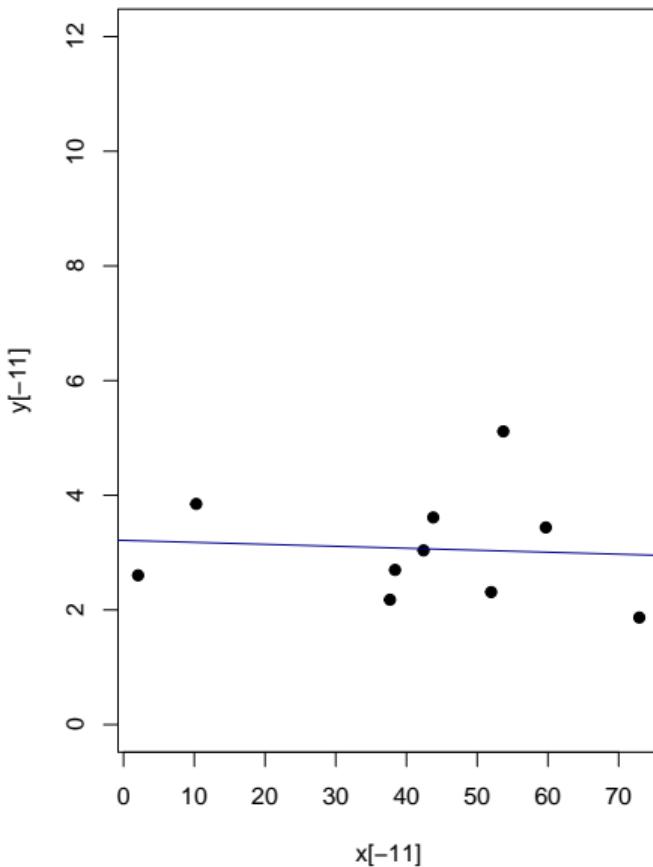
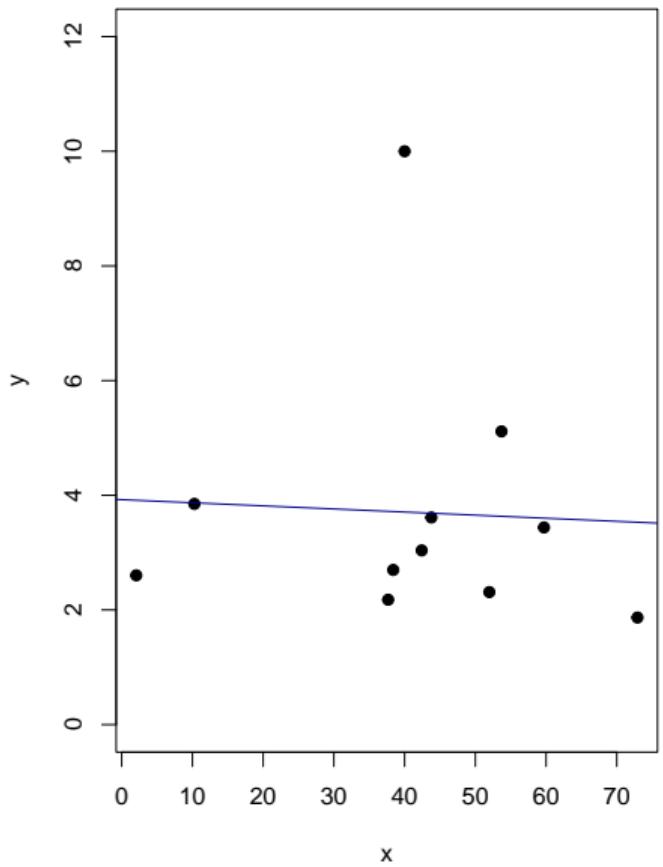
```
lm(y[-11]~x[-11]) |> tidy()
```

```
# A tibble: 2 x 5
  term      estimate std.error statistic p.value
  <chr>      <dbl>     <dbl>      <dbl>    <dbl>
1 (Intercept)  0.298     0.876     0.340 0.743
2 x[-11]        1.91      0.175     10.9  0.00000438
```

## Ejemplo 2

Outlier, no leverage y valor influencial

```
set.seed(111)
x = c(runif(10,1,100),40)
y = c(rnorm(10,3+0.005*x,1),10)
par(mfrow=c(1,2))
plot(x,y, ylim=c(0,12), pch=19);abline(lm(y~x), col="darkblue")
plot(x[-11],y[-11], ylim=c(0,12), pch=19)
abline(lm(y[-11]~x[-11]), col="darkblue")
```



Note el nulo cambio (“muy pequeño”) en los coeficientes de regresión, pero sí cambian los errores estándar

```
lm(y~x) |> tidy()
```

```
# A tibble: 2 x 5
  term      estimate std.error statistic p.value
  <chr>      <dbl>     <dbl>      <dbl>    <dbl>
1 (Intercept)  3.92      1.70      2.31    0.0463
2 x           -0.00540   0.0373    -0.145   0.888
```

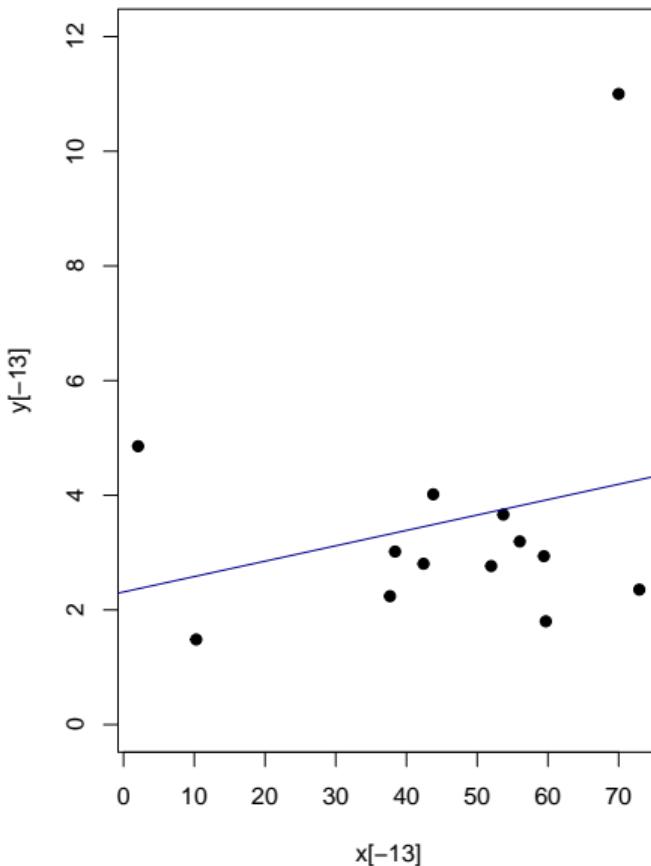
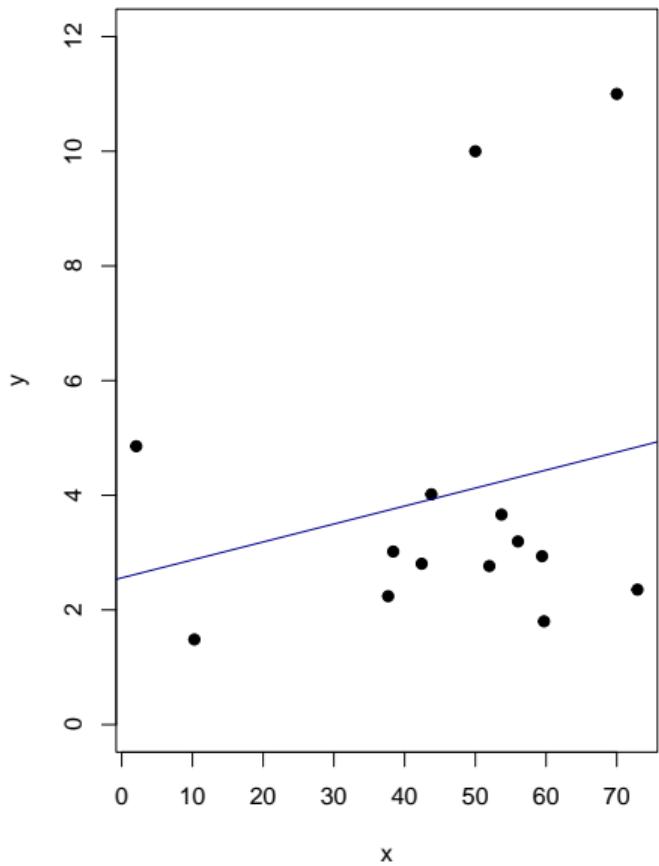
```
lm(y[-11]~x[-11]) |> tidy()
```

```
# A tibble: 2 x 5
  term      estimate std.error statistic p.value
  <chr>      <dbl>     <dbl>      <dbl>    <dbl>
1 (Intercept)  3.21      0.731      4.40    0.00229
2 x[-11]       -0.00344   0.0159    -0.217   0.834
```

## Ejemplo 3

Identifique gráficamente si en este caso se trata de outlier(s), leverage(s) y/o punto(s) influencial(es)

```
set.seed(111)
x = c(runif(12,1,100),50,70)
y = c(rnorm(12,3+0.005*x,1),10,11)
par(mfrow=c(1,2))
plot(x,y, ylim=c(0,12), pch=19);abline(lm(y~x), col="darkblue")
plot(x[-13],y[-13], ylim=c(0,12), pch=19)
abline(lm(y[-13]~x[-13]), col="darkblue")
```



```
lm(y~x) |> tidy()
```

# A tibble: 2 x 5

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
1 (Intercept)	2.56	2.04	1.26	0.233
2 x	0.0313	0.0406	0.772	0.455

```
lm(y[-13]~x[-13]) |> tidy()
```

# A tibble: 2 x 5

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
1 (Intercept)	2.31	1.70	1.36	0.202
2 x[-13]	0.0268	0.0339	0.791	0.446

## Residuales

- ▶ Recordemos que nuestro modelo está definido como:

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon$$

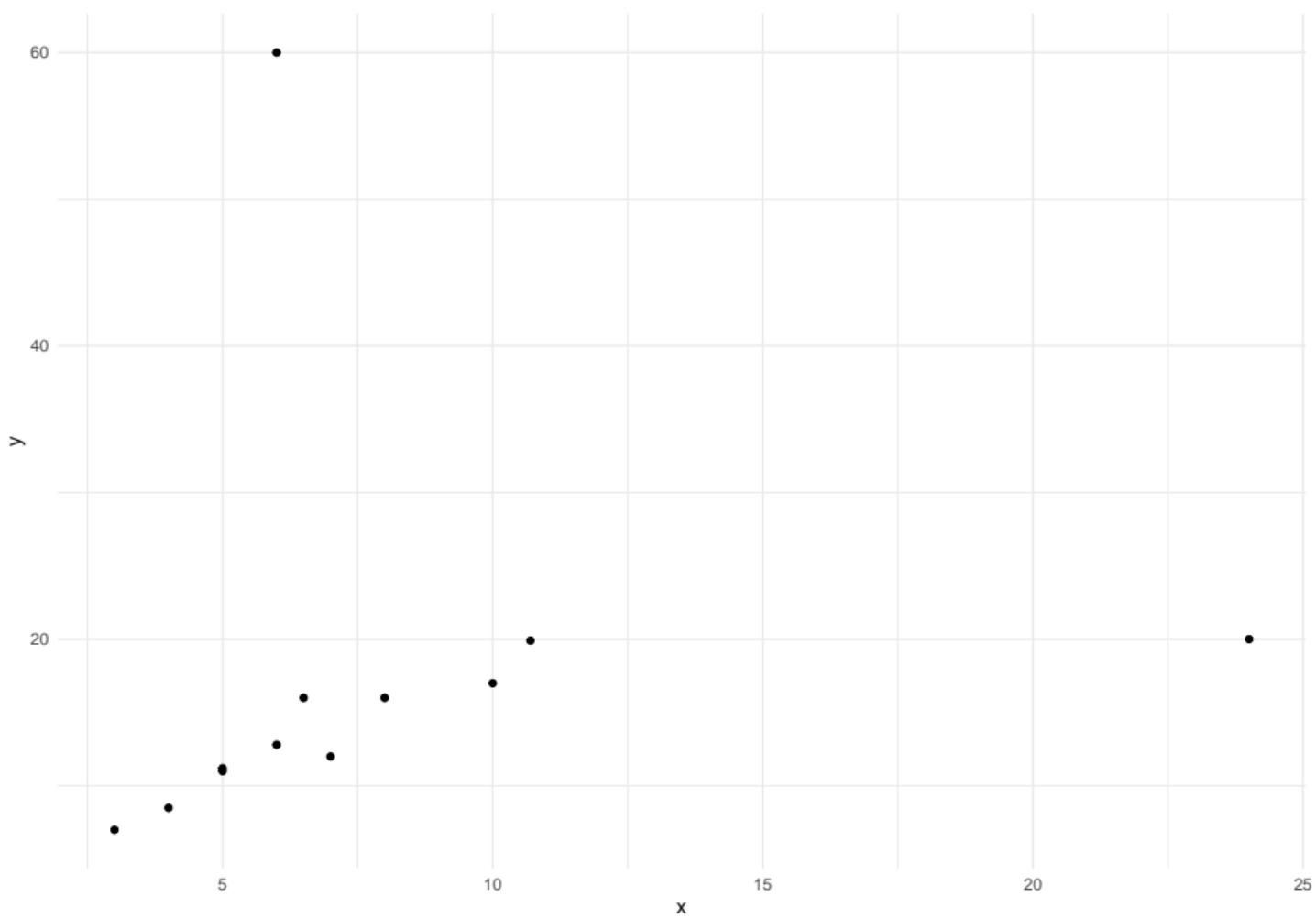
- ▶ El vector  $\epsilon$  contiene los errores del modelo, que son representados en la muestra por los residuales
- ▶ Los residuales han sido importantes en la verificación de supuestos y ahora lo serán para determinar outliers y valores influenciales

```
x = c(3,5,10,4,7,6,8,6,5,24,6.5,10.7)
y = c(7,11,17,8.5,12,12.8,16,60,11.2,20,16,19.9)
data.frame(x,y)
```

	x	y
1	3.0	7.0
2	5.0	11.0
3	10.0	17.0
4	4.0	8.5
5	7.0	12.0
6	6.0	12.8
7	8.0	16.0
8	6.0	60.0
9	5.0	11.2
10	24.0	20.0
11	6.5	16.0
12	10.7	19.9

```
library(dplyr)
library(ggplot2)
data.frame(x,y) |>
  ggplot(aes(x=x,y=y))+
  geom_point()+
  theme_minimal()

lm(y~x) -> modelo
```



## Residual estandarizado

También conocidos como residuales estudentizados de manera interna, se define como:

$$d_i = \frac{e_i}{\hat{\sigma} \sqrt{1 - h_{ii}}}$$

donde: -  $e_i$  es el i-ésimo residual

- ▶  $h_{ii}$  es el i-ésimo leverage (i-ésimo término de la diagonal de la matriz hat)
- ▶  $\hat{\sigma}$  es la estimación de la desviación estándar, recordando que para un modelo con  $n$  observaciones y  $k$  coeficientes de regresión.

$$\hat{\sigma}^2 = \frac{1}{n - k} \sum_{j=1}^n e_j^2 = CME$$

La media de un residual estandarizado es 0 y su varianza es 1. Además, asumiendo normalidad, valores por encima de 2, o por debajo de -2 son considerados outliers.

```
modelo |> resid() -> r
summary(modelo)$sigma -> s
modelo |> model.matrix() -> X
X %*% solve(t(X) %*% X) %*% t(X) -> H
(r/(s*sqrt(1-diag(H)))) -> res_stand
res_stand |> round(2)
```

1	2	3	4	5	6	7	8	9	10	11	12
-0.68	-0.42	-0.09	-0.58	-0.38	-0.30	-0.12	3.11	-0.40	-0.46	-0.08	0.10

Nota: se redondea solo con la finalidad de mostrar los residuales en la diapositiva.

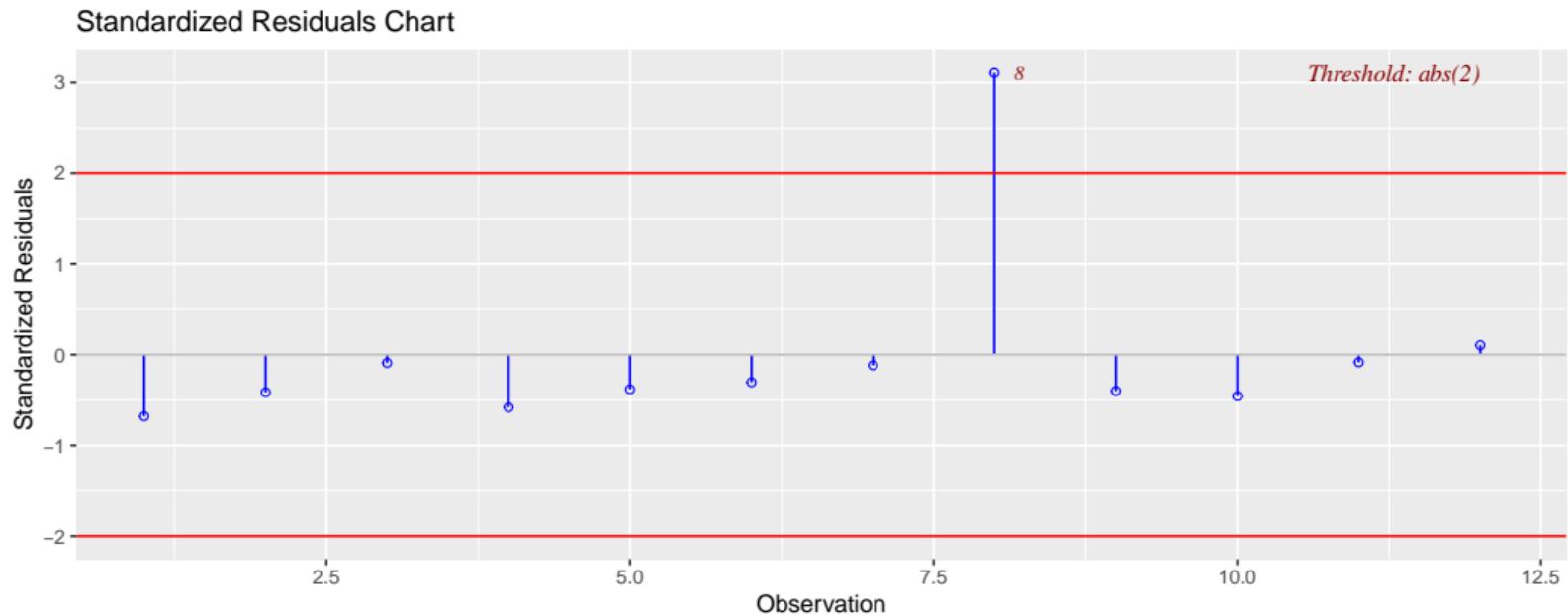
```
modelo |> rstandard() |> round(2) |> abs()
```

1	2	3	4	5	6	7	8	9	10	11	12
0.68	0.42	0.09	0.58	0.38	0.30	0.12	3.11	0.40	0.46	0.08	0.10

```
(modelo |> rstandard() |> round(2) |> abs()) > 2
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE	TRUE	FALSE	FALSE	FALSE	FALSE						

```
library(olsrr)
modelo |> ols_plot_resid_stand()
```



## Residual estudentizado

Sin embargo, de haber outliers, los residuales estandarizados podrían no ser normales. Ante ello, se propone redefinir la expresión para la estimación del CME, retirando la  $i$ -ésima observación. Así, el residual estudentizado (externamente) sería:

$$t_i = \frac{e_i}{\hat{\sigma}_{(i)} \sqrt{1 - h_{ii}}}$$

donde:

$$\hat{\sigma}_{(i)} = \frac{1}{n - k - 1} \sum_{j=1, j \neq i}^n e_j^2 = CME_{(i)}$$

Rigen las mismas características que los residuales estandarizados: media cero, varianza 1 y valores por encima de 2 o por debajo de -2 son considerados outliers.

```
# residual estudentizado 1  
lm(y[-1]~x[-1]) -> modelo_1  
summary(modelo_1)$sigma -> s_1  
r[1]/(s_1*sqrt(1-H[1,1]))
```

1  
-0.6603354

```
# residual estudentizado 2  
lm(y[-2]~x[-2]) -> modelo_2  
summary(modelo_2)$sigma -> s_2  
r[2]/(s_2*sqrt(1-H[2,2]))
```

2  
-0.3977445

```
length(x) -> n; 2 -> k  
(res_stand*sqrt((n-k-1)/(n-k-res_stand**2))) |> round(2)
```

1	2	3	4	5	6	7	8	9	10	11	12
-0.66	-0.40	-0.09	-0.56	-0.37	-0.29	-0.11	15.56	-0.38	-0.44	-0.08	0.10

```
modelo |> rstudent() |> round(2)
```

1	2	3	4	5	6	7	8	9	10	11	12
-0.66	-0.40	-0.09	-0.56	-0.37	-0.29	-0.11	15.56	-0.38	-0.44	-0.08	0.10

```
library(MASS)
```

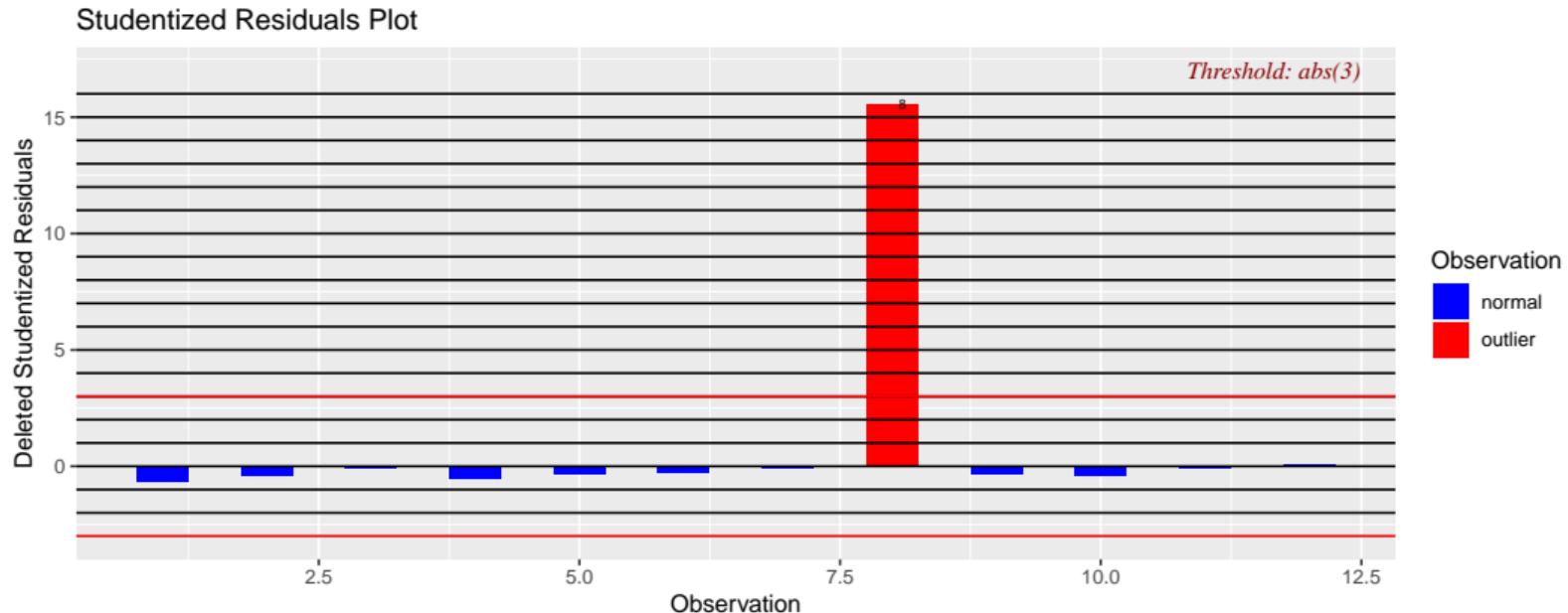
```
modelo |> studres() |> round(2) |> abs()
```

1	2	3	4	5	6	7	8	9	10	11	12
0.66	0.40	0.09	0.56	0.37	0.29	0.11	15.56	0.38	0.44	0.08	0.10

```
(modelo |> studres() |> round(3) |> abs()) > 2
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE	TRUE	FALSE	FALSE	FALSE	FALSE						

```
modelo |> ols_plot_resid_stud()
```



## Leverages

Sabemos que:

$$\hat{y} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} = \mathbf{H}\mathbf{y}$$

Es decir:

$$\hat{y}_i = h_{i1}y_1 + h_{i2}y_2 + \dots + h_{ii}y_i + \dots + h_{in}y_n$$

El valor  $h_{ii}$  es conocido como el leverage de la i-ésima observación y mide la influencia de  $y_i$  en la estimación  $\hat{y}_i$ . Geométricamente es la distancia estandarizada de la i-ésima observación al centroide de espacio X.

Regla sugerida:

- ▶ Se define:

$$\bar{h} = \frac{\sum_i h_{ii}}{n} = \frac{\text{rango}(\mathbf{H})}{n} = \frac{\text{rango}(\mathbf{X})}{n} = \frac{k}{n}$$

- ▶ Entonces, una observación podrá ser considerada como un leverage si  $h_{ii} > 2\frac{k}{n}$ , donde  $k$  es el número de coeficientes de regresión y  $n$  es el tamaño de muestra, siempre y cuando  $2\frac{k}{n} < 1$
- ▶ ¿En qué situación no se cumpliría que  $2\frac{k}{n} < 1$ ?
- ▶ Vea la matriz hat en la siguiente diapositiva

```
modelo |> model.matrix() -> X  
X %*% solve(t(X) %*% X) %*% t(X) -> H  
H
```

	1	2	3	4	5	6
1	0.15544812	0.12621239	0.05312309	0.14083026	0.09697667	0.111594532
2	0.12621239	0.10882899	0.06537048	0.11752069	0.09144559	0.100137289
3	0.05312309	0.06537048	0.09598898	0.05924679	0.07761788	0.071494182
4	0.14083026	0.11752069	0.05924679	0.12917547	0.09421113	0.105865911
5	0.09697667	0.09144559	0.07761788	0.09421113	0.08591451	0.088680047
6	0.11159453	0.10013729	0.07149418	0.10586591	0.08868005	0.094408668
7	0.08235881	0.08275389	0.08374158	0.08255635	0.08314896	0.082951425
8	0.11159453	0.10013729	0.07149418	0.10586591	0.08868005	0.094408668
9	0.12621239	0.10882899	0.06537048	0.11752069	0.09144559	0.100137289
10	-0.15152697	-0.05631334	0.18172076	-0.10392015	0.03890030	-0.008706517
11	0.10428560	0.09579144	0.07455603	0.10003852	0.08729728	0.091544357
12	0.04289058	0.05928629	0.10027557	0.05108844	0.07568200	0.067484148
	7	8	9	10	11	12
1	0.08235881	0.111594532	0.12621239	-0.151526974	0.10428560	0.04289058
2	0.08275389	0.100137289	0.10882899	-0.056313336	0.09579144	0.05928629
3	0.08374158	0.071494182	0.06537048	0.181720759	0.07455603	0.10027557
4	0.08255635	0.105865911	0.11752069	-0.103920155	0.10003852	0.05108844
5	0.08314896	0.088680047	0.09144559	0.038900302	0.08729728	0.07568200
6	0.08295143	0.094408668	0.10013729	-0.008706517	0.09154436	0.06748415
7	0.08334650	0.082951425	0.08275389	0.086507121	0.08305019	0.08387986
8	0.08295143	0.094408668	0.10013729	-0.008706517	0.09154436	0.06748415
9	0.08275389	0.100137289	0.10882899	-0.056313336	0.09579144	0.05928629
10	0.08650712	-0.008706517	-0.05631334	0.848216226	0.01509689	0.21504553
11	0.08305019	0.091544357	0.09579144	0.015096893	0.08942082	0.07158307
12	0.08387986	0.067484148	0.05928629	0.215045533	0.07158307	0.10601406

Evaluando:

```
length(x) -> n; 2 -> k  
diag(H) |> as.vector()
```

```
[1] 0.15544812 0.10882899 0.09598898 0.12917547 0.08591451 0.09440867  
[7] 0.08334650 0.09440867 0.10882899 0.84821623 0.08942082 0.10601406
```

```
2*k/n
```

```
[1] 0.3333333
```

```
diag(H) > 2*k/n
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE	TRUE	FALSE	FALSE								

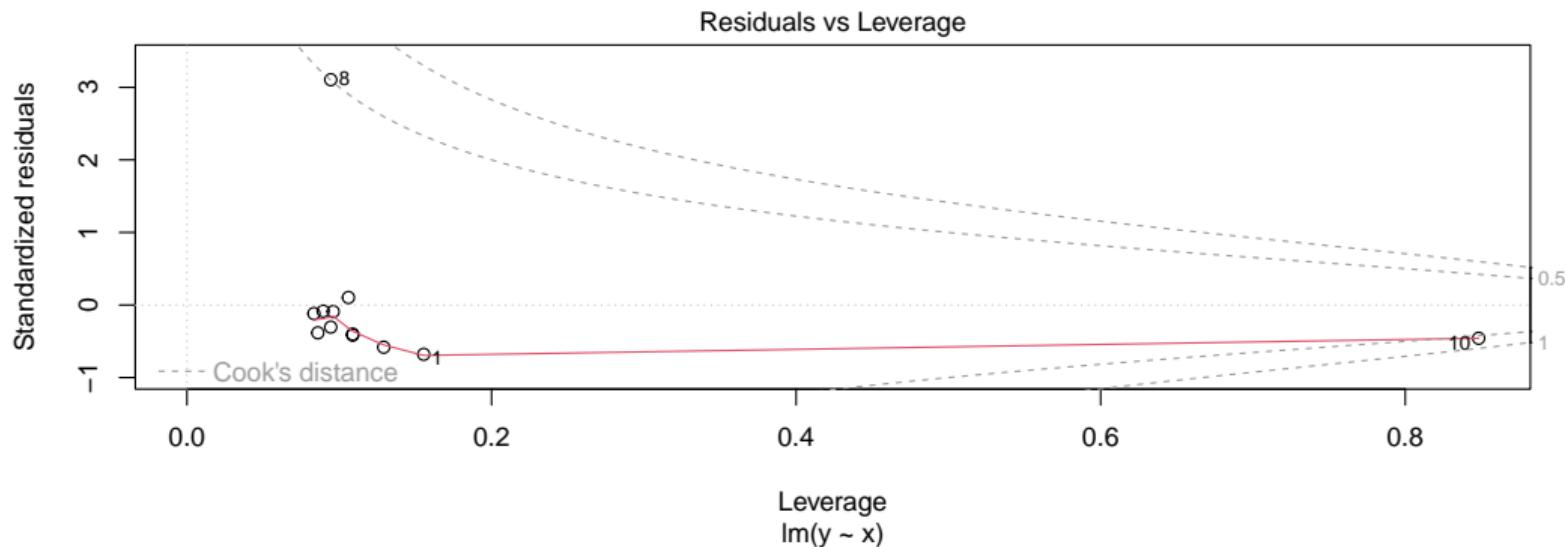
¿Todo leverage es outlier?

```
data.frame(leverage = diag(H)>2*k/n,
           outlier = abs(modelo |> rstudent())>2)
```

	leverage	outlier
1	FALSE	FALSE
2	FALSE	FALSE
3	FALSE	FALSE
4	FALSE	FALSE
5	FALSE	FALSE
6	FALSE	FALSE
7	FALSE	FALSE
8	FALSE	TRUE
9	FALSE	FALSE
10	TRUE	FALSE
11	FALSE	FALSE
12	FALSE	FALSE

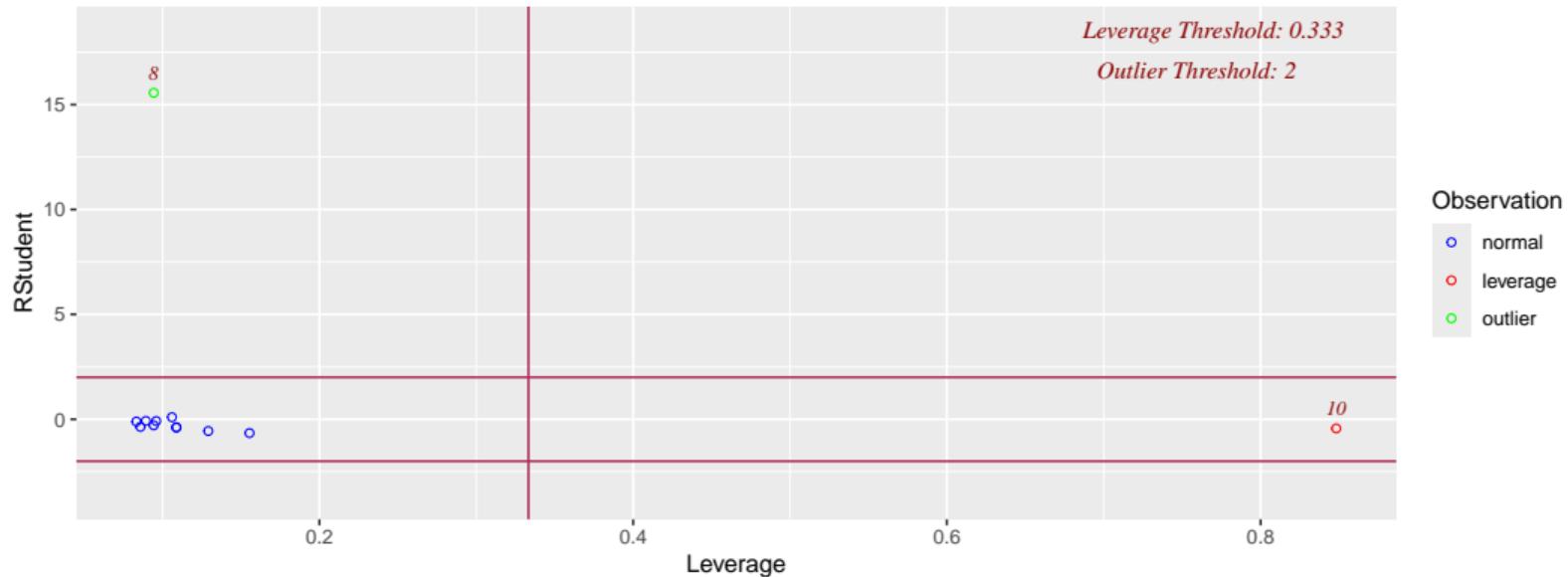
Gráficamente

```
modelo |> plot(which=5)
```



```
modelo |> ols_plot_resid_lev()
```

Outlier and Leverage Diagnostics for y



## Distancia de Cook

Mide la influencia de cada observación en los coeficientes de regresión estimados

$$D_i = \frac{(\hat{\beta}_{(i)} - \hat{\beta})' \mathbf{X}' \mathbf{X} (\hat{\beta}_{(i)} - \hat{\beta})}{k \hat{\sigma}^2} = \frac{d_i^2 h_{ii}}{k(1 - h_{ii})}$$

donde:

- ▶  $\hat{\beta}_{(i)}$  es el valor estimado del vector de coeficientes  $\beta$  al no ser considerada la i-ésima observación.
- ▶  $d_i$  es el i-ésimo residual estandarizado.
- ▶  $h_{ii}$  es el i-ésimo leverage.
- ▶  $k$  es el número de coeficientes estimados de regresión.

## Regla sugerida

- ▶ El valor de  $D_i$  puede compararse con  $F_{k,n-k}$  de tal modo que:
  - ▶ Si  $D_i = F_{0.5,k,n-k}$ , el estimador  $\hat{\beta}_{(i)}$  se moverá hacia el borde de la región de confianza conjunta de nivel  $1 - \alpha$  para  $\beta$ .
  - ▶ Si  $D_i > F_{0.5,k,n-k}$ , la observación es considerada influyente.
- ▶ En la práctica rara vez se usa la comparación con  $F$ , sino que se aplica la regla empírica: es una observación influyente si  $D_i > \frac{4}{n}$  (muestras pequeñas) o  $D_i > 1$  (muestras grandes).

```
# ¿observación 1 es influyente?  
modelo = lm(y~x)  
modelo_sin = lm(y[-1]~x[-1])  
modelo |> coef() -> beta  
modelo_sin |> coef() -> beta_sin  
modelo |> model.matrix() -> X  
(modelo |> sigma())**2 -> cme  
2 -> k  
(t(beta-beta_sin) %*% t(X) %*% X %*% (beta-beta_sin))/(k*cme)
```

```
[,1]  
[1,] 0.04252736
```

```
modelo |> rstandard() -> rsta  
modelo |> hatvalues() -> h  
(rsta[1]**2*h[1])/((k*(1-h[1])))
```

```
1  
0.04252736
```

```
((rsta**2*h)/(k*(1-h))) |> round(2) |> as.vector()
```

```
[1] 0.04 0.01 0.00 0.03 0.01 0.00 0.00 0.50 0.01 0.59 0.00 0.00
```

```
modelo |> cooks.distance() |> round(2) |> as.vector()
```

```
[1] 0.04 0.01 0.00 0.03 0.01 0.00 0.00 0.50 0.01 0.59 0.00 0.00
```

```
(modelo |> cooks.distance() > 1
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE											

```
qf(0.5,k,n-k)
```

```
[1] 0.7434918
```

```
(modelo |> cooks.distance() > qf(0.5,k,n-k)
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE											

```
4/n
```

```
[1] 0.3333333
```

```
(modelo |> cooks.distance() > 4/n
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE	TRUE	FALSE	TRUE	FALSE	FALSE						

¿Entre qué valores fluctúa F? Interprete los valores de la siguiente tabla:

```
alfas = c(0.01,0.05,0.1,0.25,0.5,0.66,0.75,0.90,0.95,0.99)
data.frame(alfas,valorF=qf(alfas,k,n-k))
```

	alfas	valorF
1	0.01	0.01006044
2	0.05	0.05155730
3	0.10	0.10647844
4	0.25	0.29611921
5	0.50	0.74349177
6	0.66	1.20403474
7	0.75	1.59753955
8	0.90	2.92446596
9	0.95	4.10282102
10	0.99	7.55943216

Supongamos  $k = 5$  y  $n = 700$ . Interprete los valores de la siguiente tabla:

```
alfas = c(0.01,0.05,0.1,0.25,0.5,0.66,0.75,0.90,0.95,0.99)
data.frame(alfas,valorF=qf(alfas,5,700-5))
```

	alfas	valorF
1	0.01	0.1106651
2	0.05	0.2287904
3	0.10	0.3217405
4	0.25	0.5347960
5	0.50	0.8711387
6	0.66	1.1355399
7	0.75	1.3285961
8	0.90	1.8555771
9	0.95	2.2269931
10	0.99	3.0436173

Haciendo el proceso inverso: Sabemos que  $D_8 = 0.5025745143$ , ¿qué porcentaje mínimo de variación tendría  $\hat{\beta}$  en la región de confianza si retiramos la observación 8?

```
pf(0.5025745143,k,n-k) # pf(D,k,n-k)
```

[1] 0.3805299

Sabemos que  $D_{10} = 0.5869709889$ , ¿qué porcentaje mínimo de variación tendría  $\hat{\beta}$  en la región de confianza si retiramos la observación 10?

```
pf(0.5869709889,k,n-k) # pf(D,k,n-k)
```

[1] 0.4259259

```
library(car)
```

```
modelo = lm(y~x); modelo |> coef()
```

```
(Intercept)           x  
15.1572753   0.3100073
```

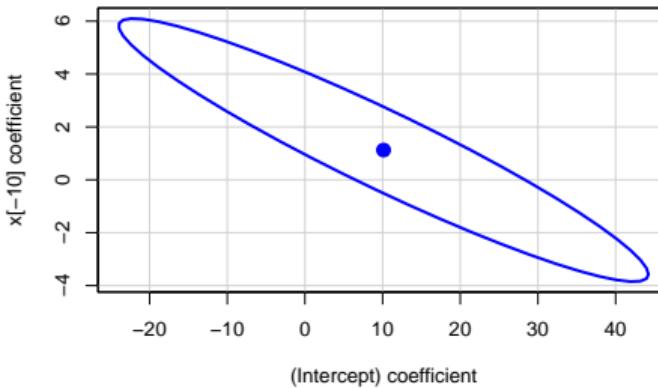
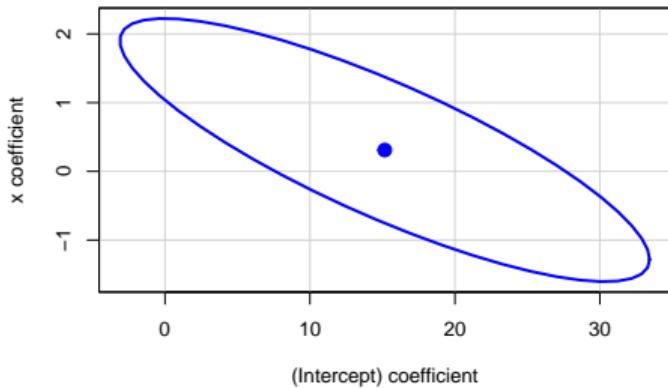
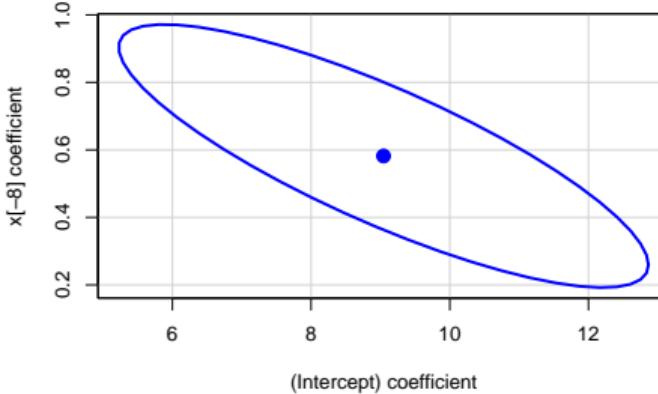
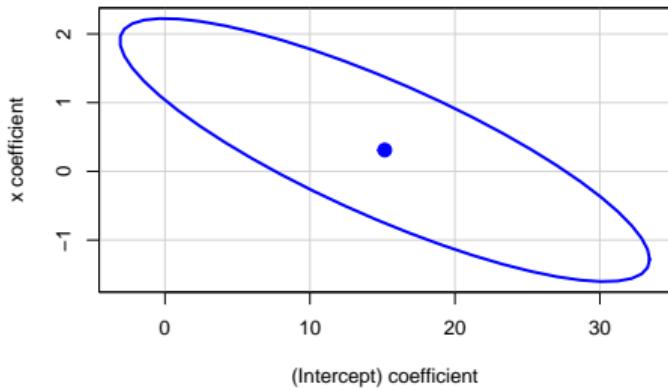
```
modelo_sin_obs8 = lm(y[-8]~x[-8]); modelo_sin_obs8 |> coef()
```

```
(Intercept)       x[-8]  
9.0448862    0.5819087
```

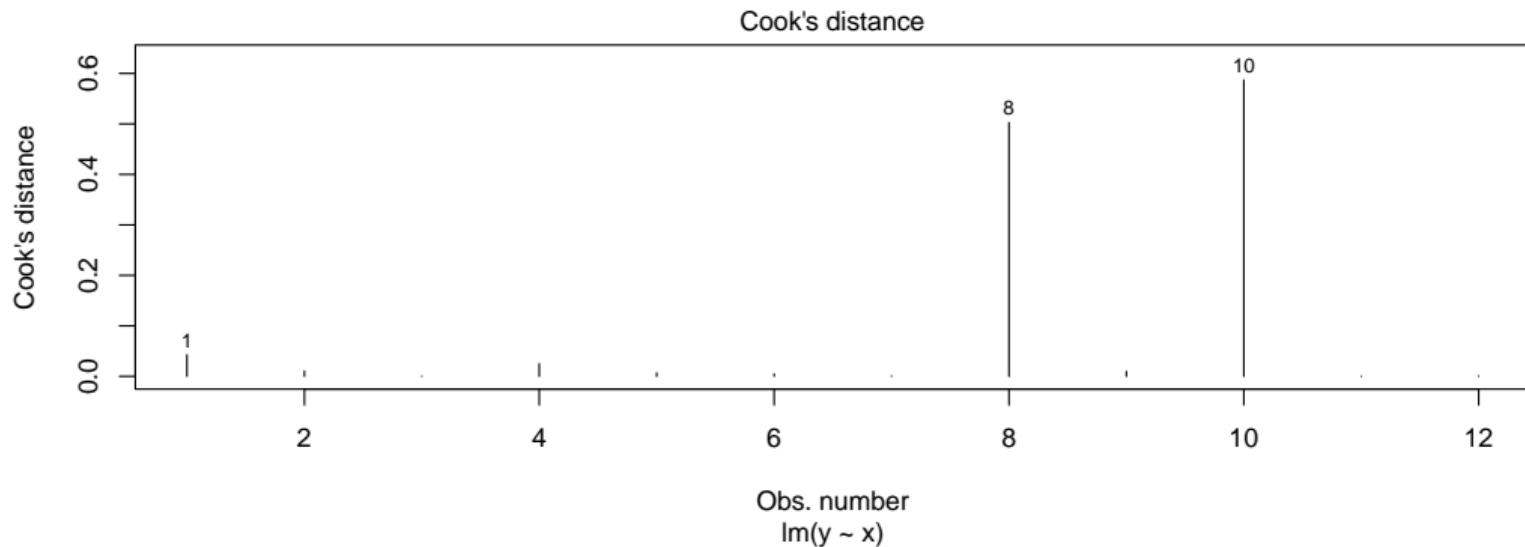
```
modelo_sin_obs10 = lm(y[-10]~x[-10]); modelo_sin_obs10 |> coef()
```

```
(Intercept)      x[-10]  
10.120156     1.124695
```

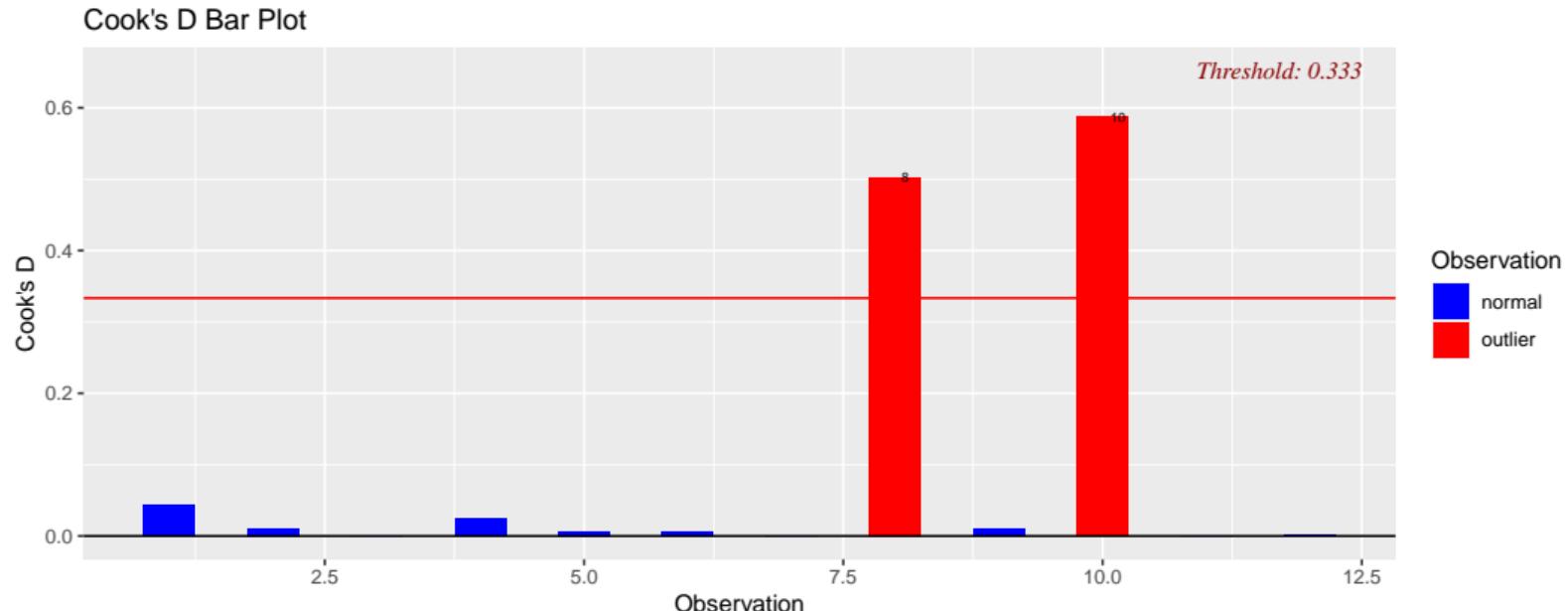
```
par(mfrow=c(2,2))
modelo |> confidenceEllipse(levels = 0.9)
modelo_sin_obs8 |> confidenceEllipse(levels = 0.9)
modelo |> confidenceEllipse(levels = 0.9)
modelo_sin_obs10 |> confidenceEllipse(levels = 0.9)
```



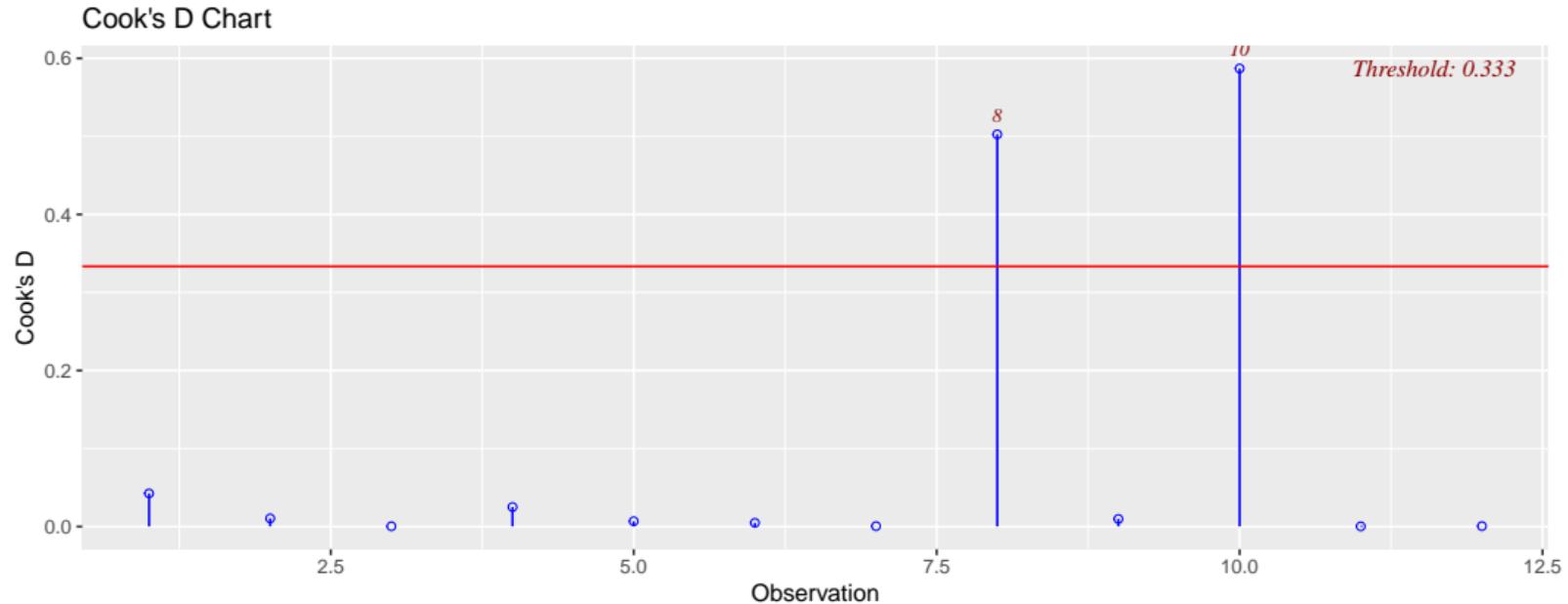
```
modelo |> plot(which=4)
```



```
modelo |> ols_plot_cooksd_bar()
```



```
modelo |> ols_plot_cooksd_chart()
```



## DFBETAS

De manera similar a la distancia de Cook, permite evaluar cuánto cambia el j-ésimo coeficiente de regresión si se elimina la i-ésima observación. Se tiene que:

$$\hat{\beta} - \hat{\beta}_{(i)} = \frac{(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_i\epsilon_i}{1 - h_{ii}}$$

donde:

- ▶  $\hat{\beta}$  es la estimación de  $\beta$  en el conjunto de datos completo
- ▶  $\hat{\beta}_{(i)}$  cuando se elimina la i-ésima observación

¿Cuál es el efecto de los leverages y los residuales en la variación de  $\hat{\beta}$  al eliminar la i-ésima observación? Ahora, escalamos:

$$DFBETAS_{i,j} = \frac{\hat{\beta}_j - \hat{\beta}_{j(i)}}{\sqrt{\hat{\sigma}_{(i)}^2 C_{jj}}}$$

donde:

- ▶  $\hat{\beta}_j$  es la estimación de  $\beta_j$  en el conjunto de datos completo
- ▶  $\hat{\beta}_{j(i)}$  es la estimación de  $\beta_j$  cuando se elimina la i-ésima observación
- ▶  $\hat{\sigma}_{(i)}^2$  es el CME al eliminar la i-ésima observación
- ▶  $C_{jj}$  es el j-ésimo elemento de la diagonal de  $(\mathbf{X}'\mathbf{X})^{-1}$

Así, DFBETAS es una matriz de dimensión  $n \times k$

Regla sugerida: Si un punto es valor influencial, entonces  $|DBETA_{i,j}| > \frac{2}{\sqrt{n}}$

```
modelo = lm(y~x)
modelo_sin = lm(y[-1]~x[-1])
modelo |> coef() -> beta
modelo_sin |> coef() -> beta_sin
beta - beta_sin
```

(Intercept)	x
-2.1444672	0.1572868

```
modelo |> model.matrix() -> X  
modelo |> residuals() -> residuales  
modelo |> hatvalues() -> h  
(solve(t(X) %*% X) %*% X[1,] * residuales[1]) / (1 - h[1])
```

```
[,1]  
(Intercept) -2.1444672  
x 0.1572868
```

```
(modelo_sin |> sigma())^2 -> cme_sin  
solve(t(X) %*% X) -> C  
(beta - beta_sin) / sqrt(cme_sin * diag(C))
```

```
(Intercept) x  
-0.2756912 0.1929583
```

```
modelo |> dfbetas()
```

	(Intercept)	x
1	-0.275691211	0.1929583403
2	-0.123523176	0.0672754784
3	-0.006071150	-0.0102087577
4	-0.203638872	0.1288329898
5	-0.077603339	0.0194540553
6	-0.075649967	0.0321127060
7	-0.018349996	-0.0004230407
8	4.053966371	-1.7208709421
9	-0.119122513	0.0648787078
10	0.639178659	-0.9865042898
11	-0.018923427	0.0065534903
12	0.003665428	0.0156715670

Verificando las observaciones que tienen DFBETA(s) alto:

```
length(x) -> n  
2/sqrt(n)
```

```
[1] 0.5773503
```

```
(beta - beta_sin)/sqrt(cme_sin*C[1,1]) >= 2/sqrt(n)
```

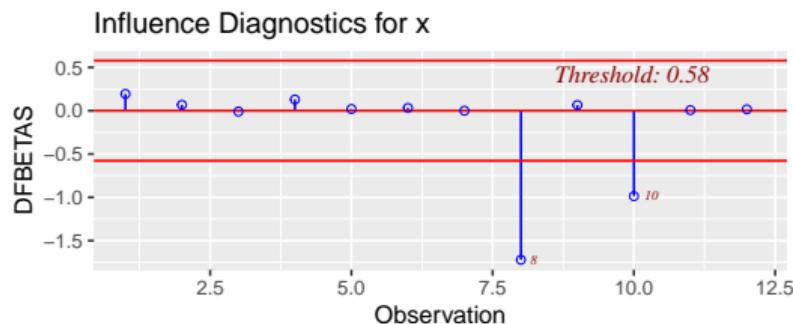
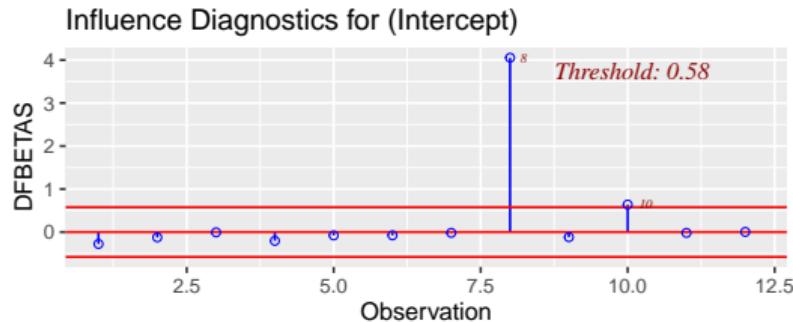
(Intercept)	x
FALSE	FALSE

```
(abs(modelo > dfbetas())) >= 2/sqrt(n)
```

	(Intercept)	x
1	FALSE	FALSE
2	FALSE	FALSE
3	FALSE	FALSE
4	FALSE	FALSE
5	FALSE	FALSE
6	FALSE	FALSE
7	FALSE	FALSE
8	TRUE	TRUE
9	FALSE	FALSE
10	TRUE	TRUE
11	FALSE	FALSE
12	FALSE	FALSE

```
modelo |> ols_plot_dfbetas()
```

page 1 of 1



## DFFITS

¿Qué sucede con el valor de  $\hat{y}$  si se elimina la i-ésima observación?

$$DFFITS_i = \frac{\hat{y}_i - \hat{y}_{(i)}}{\sqrt{\hat{\sigma}_{(i)}^2 h_{ii}}} = t_i \times \sqrt{\frac{h_{ii}}{1 - h_{ii}}}$$

donde:

- ▶  $\hat{y}_i$  es el valor ajustado para la i-ésima observación en el modelo con datos completos.
- ▶  $\hat{y}$  es el valor ajustado para la i-ésima observación en el modelo cuando se omite la i-ésima observación.
- ▶  $\hat{\sigma}_{(i)}^2$  es el CME en el modelo sin la i-ésima observación
- ▶  $h_{ii}$  es el i-ésimo valor de la diagonal de la matriz hat de los datos completos.

Puede interpretarse como el número de desviaciones estándar que  $\hat{y}_i$  cambia si la i-ésima observación es removida.

```
modelo |> predict() -> y_pred
x_sin1 = x[-1]
y_sin1 = y[-1]
modelo_sin1 = lm(y_sin1 ~ x_sin1, data.frame(x_sin1,y_sin1))
modelo_sin1 |> predict(data.frame(x_sin1 = x[1])) -> y_pred_sin
(modelo_sin1 |> sigma())**2 -> cme_sin
modelo |> hatvalues() -> h
(y_pred[1]-y_pred_sin[1])/sqrt(cme_sin*h[1])
```

```
1
-0.2832984
```

```
modelo |> rstudent() -> t
t[1]*sqrt(h[1]/(1-h[1]))
```

```
1
-0.2832984
```

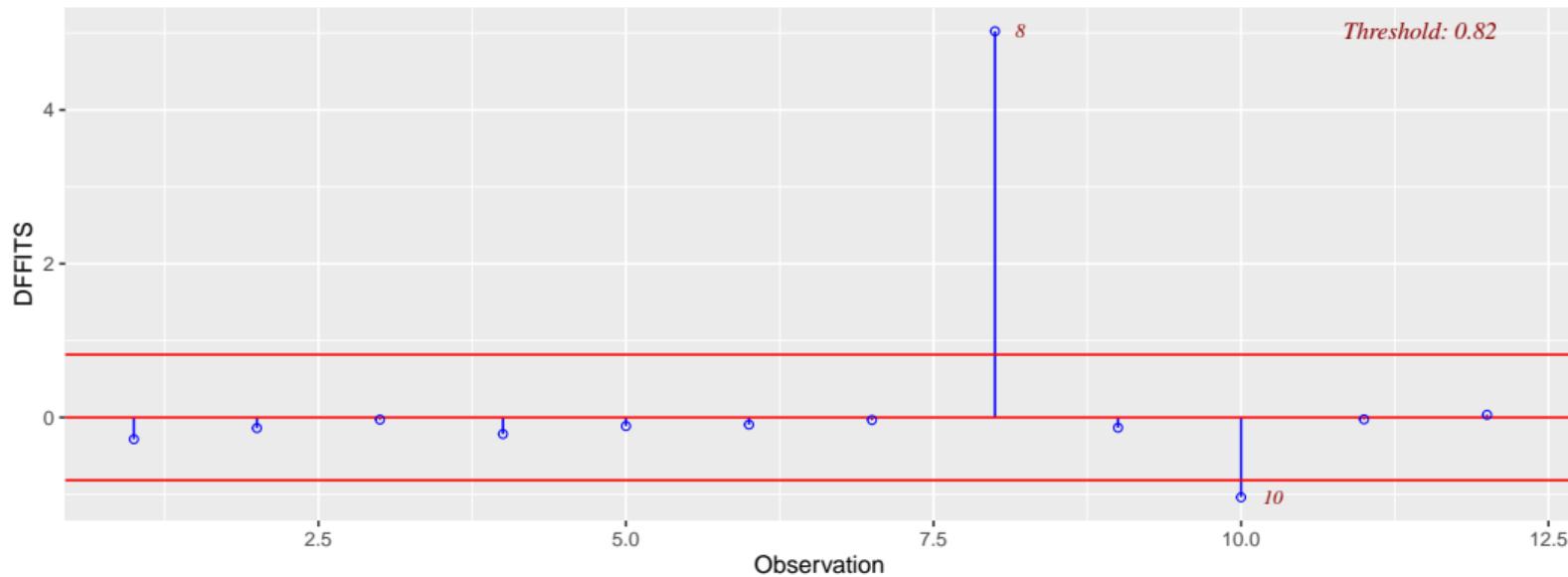
```
modelo |> dffits() |> round(3)
```

```
 1      2      3      4      5      6      7      8      9      10  
-0.283 -0.139 -0.028 -0.216 -0.112 -0.094 -0.034  5.024 -0.134 -  
1.039 -0.025  
 12  
0.034
```

```
2 -> k  
length(x) -> n  
(abs(modelo |> dffits())) >= 2*sqrt(k/n)
```

```
 1      2      3      4      5      6      7      8      9      10     11     12  
FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE
```

## Influence Diagnostics for y



## COVRATIO

El indicador COVRATIO (covariance ratio) mide el efecto de la i-ésima observación en la precisión de los estimadores. La varianza generalizada es una medida de dispersión de un vector, así, la VG de  $\hat{\beta}$  es  $|\hat{\sigma}^2(\mathbf{X}'\mathbf{X})^{-1}|$ . Esta medida se utiliza para definir el COVRATIO:

$$COVRATIO_i = \frac{|\hat{\sigma}_{(i)}^2(\mathbf{X}'_{(i)}\mathbf{X}_{(i)})^{-1}|}{|\hat{\sigma}^2(\mathbf{X}'\mathbf{X})^{-1}|} = \left(\frac{\hat{\sigma}_{(i)}^2}{\hat{\sigma}^2}\right)^k \left(\frac{1}{1-h_{ii}}\right)$$

¿Qué sucede si  $COVRATIO > 1$ ? ¿Y si es menor a 1? ¿Qué papel cumplen los leverages en los COVRATIOS?

Regla sugerida: Una observación presentará COVRATIO alto si:

$|COVRATIO_i| > 1 + 3\frac{k}{n}$  o  $|COVRATIO_i| < 1 - 3\frac{k}{n}$ . La segunda condición está sujeta a que  $n > 3k$ .

```
x_sin1 = x[-1]
y_sin1 = y[-1]
modelo_sin1 = lm(y_sin1 ~ x_sin1, data.frame(x_sin1,y_sin1))
(modelo_sin1 |> sigma()**2 -> cme_sin
(modelo |> sigma()**2 -> cme
modelo |> model.matrix() -> X
det(cme_sin*solve(t(X[-1,])%*%X[-1,]))/det(cme*solve(t(X)%*%X))
```

```
[1] 1.329823
```

```
2 -> k
modelo |> hatvalues() -> h
(cme_sin/cme)**k*(1/(1-h[1]))
```

1

1.329823

```
2 -> k  
length(x) -> n  
abs((cme_sin/cme)**k*(1/(1-h[1]))) > 1 + 3*k/n
```

```
1  
FALSE
```

```
n > 3*k
```

```
[1] TRUE
```

```
abs((cme_sin/cme)**k*(1/(1-h[1]))) < 1 - 3*k/n
```

```
1  
FALSE
```

```
modelo |> covratio() |> round(3)
```

1	2	3	4	5	6	7	8	9	10	11	12
1.330	1.338	1.363	1.323	1.311	1.338	1.343	0.002	1.341	7.796	1.354	1.378

```
abs(modelo |> covratio()) > 1 + 3*k/n
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE	TRUE	FALSE	FALSE								

```
abs(modelo |> covratio()) < 1 - 3*k/n
```

1	2	3	4	5	6	7	8	9	10	11	12
FALSE	TRUE	FALSE	FALSE	FALSE	FALSE						

Resumen proporcionado por R:

```
modelo |> influence.measures()
```

Influence measures of

```
lm(formula = y ~ x) :
```

	dfb.1_	dfb.x	dffit	cov.r	cook.d	hat	inf
1	-0.27569	0.192958	-0.2833	1.32982	0.042527	0.1554	
2	-0.12352	0.067275	-0.1390	1.33788	0.010548	0.1088	
3	-0.00607	-0.010209	-0.0281	1.36340	0.000439	0.0960	
4	-0.20364	0.128833	-0.2163	1.32335	0.025104	0.1292	
5	-0.07760	0.019454	-0.1122	1.31126	0.006896	0.0859	
6	-0.07565	0.032113	-0.0938	1.33808	0.004838	0.0944	
7	-0.01835	-0.000423	-0.0337	1.34310	0.000628	0.0833	
8	4.05397	-1.720871	5.0243	0.00175	0.502575	0.0944	*
9	-0.11912	0.064879	-0.1340	1.34113	0.009821	0.1088	
10	0.63918	-0.986504	-1.0389	7.79559	0.586971	0.8482	*
11	-0.01892	0.006553	-0.0251	1.35387	0.000350	0.0894	
12	0.00367	0.015672	0.0339	1.37800	0.000637	0.1060	

Leverage → Residual → DFFITS → DFBETAS → Distancia de Cook → COVRATIO

## Bibliografía

- ▶ Mendenhall, W. (2012). A Second Course in Statistics Regression Analysis. Pearson.
- ▶ Montgomery, D., Peck, E., Vining, G. (2012). Introduction to Linear Regression Analysis. Wiley.
- ▶ Rawlings, J. (1998). Applied Regression Analysis: A Research Tool. Springer.
- ▶ Weisberg, S. (2014) Applied Linear Regression. Wiley