INFIRIENDO PATRONES:
TÉCNICAS AVANZADAS DE REGRESIÓN
Y SU APLICACIÓN CON R

Luis M. Carrascal
(www.lmcarrascal.eu)
Museo Nacional de Ciencias Naturales, CSIC

(Madrid, noviembre 2022)

Programa

TIPOS DE DISTRIBUCIONES
- continuas: Normal, Gamma
- discretas: Poisson, Binomial Negativa
- “infladas” de ceros
- "nomiales": Binomiales, Multinomiales ordinales

MODELOS GENERALIZADOS
- generalidades
- construcción de modelos lineales
- modelos de relaciones no lineales (*generalized additive models*, GAM)

VALORACIÓN DE LOS RESIDUOS DEL MODELO
- normalidad
- heterocedasticidad
- puntos influyentes y perdidos

RELACIONES ENTRE LAS VARIABLES PREDICTORAS
- exploraciones visuales
- independencia entre los predictores
- colinearidad y VIF (*variance inflation factor*)

TRANSFORMACIONES DE LA RESPUESTA EN MODELOS GENERALES LINEALES

- logarítmica, raíz cuadrada, arcoseno
- transformación de rangos
- transformación de Box-Cox (automatizada)

RESULTADOS DEL MODELO

- omnibus test
- comparación del modelo nulo con el de interés mediante Akaike AICc
- relación entre lo observado y lo predicho
- variación total explicada (R^2 y pseudo-R^2)
- tabla de coeficientes y su interpretación
- estima de significación de efectos: tipos I, II y III

PARTICIÓN DE LA VARIABILIDAD DE LA RESPUESTA ENTRE LOS DE PREDICTORES

EXPLICACIÓN vs. PREDICCIÓN: ¿CUÁL ES EL PODER PREDICTIVO DEL MODELO?

AFRONTANDO LA VIOLACIÓN DE LA HOMOCEDASTICIDAD DE LOS RESIDUOS

- uso de la matriz de varianzas-covarianzas
- estimadores robustos con correcciones HC0, HC1, HC2, HC3, HC4, HC4m, HC5

DATOS INFLUYENTES - PERDIDOS Y DESVÍOS DE LOS SUPUESTOS

estimas robustas

bootstrapping de los modelos

SOBREDISPERSIÓN: CÓMO TRATARLA Y A QUÉ MODELOS

estima del valor ϕ de sobre dispersión

corrección por ϕ o utilizando pseudo-familias *quasi*

PARTICULARIDADES DE LOS MODELOS GENERALIZADOS CON BINOMIALES

interpretación de los coeficientes (*odds ratios*)

AUC y diagramas ROC

especificidad, sensitividad, npv, ppv, frecuencia umbral óptima de clasificación

caso especial de las variables respuestas multinomiales

REGRESIÓN

qué es
qué proporciona
tipos de modelos

QUÉ ES:

Establece una asociación matemática entre:
una variable (respuesta, "dependiente")
y otras que la predicen – explican (predictoras, "independientes")

respuesta ~ predictoras (P1, P2, P3, …)

El tipo de modelo vendrá determinado por la naturaleza de la variable respuesta (qué tipo de distribución sigue)

Las variables predictoras son "independientes" si, y sólo si, entre ellas tienen correlación "cero"

r(P1, P2) = 0, r(P1, P3) = 0, r(P2, P3) = 0 …

Tanto mejor será el modelo de regresión cuanto más sustento argumental tenga la asociación establecida
(*i.e.*, basado en hipótesis funcionales)

QUÉ PROPORCIONA:

Valores de coeficientes de regresión

establecen cómo cambia la respuesta por unidad de cambio en el predictor

Si las predictoras no son independientes entre si **--> efectos parciales**

control estadístico del efecto de P1 ajustado por P2, P3, …

Significación de efectos (en el esquema **frecuentista**)

dado que la hipótesis nula es cierta,

cuál es la probabilidad de obtener un coeficiente de esa magnitud o superior

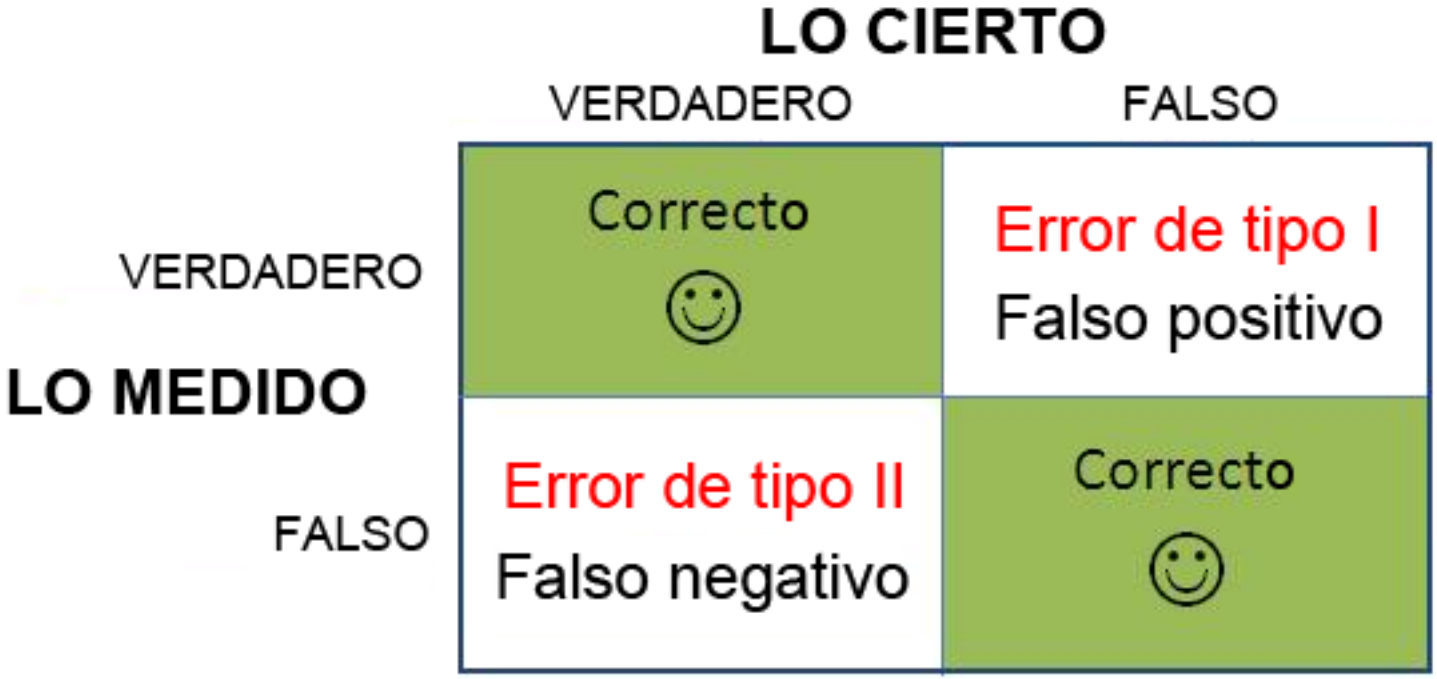
Descomposición de la variación en sus parte (**magnitud de efectos**)

se cuantifica la proporción del contenido informativo en la respuesta

* que es explicado por todas las variables predictoras
* que es atribuible exclusivamente a cada predictor
* y la existencia de concomitancias (efectos conjuntos) entre los predictores

SIGNIFICACIÓN - ERRORES DE TIPO I y II

Esquema gráfico de errores



Significación:

Dado que **la hipótesis nula (Ho) es cierta** (*i.e.*, ausencia de efectos)
cuál es la probabilidad de obtener un resultado de la magnitud observada

GENERALIDADES

El predictor lineal η incluye la suma lineal de los efectos de una o más variables explicativas (x_j).

$$\eta_i = \sum_{j=1}^{p} x_{ij}\beta_j$$

β_j representan los parámetros desconocidos que es necesario estimar.

Estos valores son llevados a una nueva escala mediante una transformación adecuada. Esto es, η_i no representa a y_i, sino a una transformación de los valores y mediante la **función de vínculo**.

La transformación utilizada define la función de vínculo.

La función de vínculo (g) relaciona la media de los valores ***y*** (μ) con el predictor lineal η mediante:

$$\eta = g(\mu)$$

Para volver a la escala original de medida (***y***), el valor ajustado es la función inversa de la transformación definida por la función de vínculo.

Para determinar el ajuste de un modelo,

* el procedimiento evalúa el predictor lineal η para cada valor de la variable dependiente (*y*),
* y luego compara este valor predicho con la transformación de *y*

Mediante el uso de diferentes funciones de vínculo, podemos valorar la adecuación de nuestro modelo a los datos.

Para ello utilizaremos el concepto y parámetro **devianza**.

El modelo más apropiado será aquel que minimice la devianza residual.

En los modelos Generales Lineales operamos con variables dependientes normales, y los modelos proporcionan residuos que siguen la distribución normal. Sin embargo, numerosos datos no presentan errores normales.

* por sesgo y kurtosis
* están acotados (caso de proporciones)
* son conteos que no pueden manifestar valores negativos

Podemos distinguir las siguientes **familias principales de errores**:

* errores Normales
* errores Gamma (datos que muestran un CV constante)
* errores Poisson (conteos de fenómenos raros)
* errores Binomiales negativos (Poisson con mayor dispersión)
* errores Binomiales (datos que miden respuestas si/no o proporciones)
* errores Exponenciales (datos de supervivencia)

Para estos errores se han definido las **funciones de vínculo** más adecuadas (por defecto*; canónicas*):

ERRORES	FUNCIÓN
* Normales	Identidad
* Poisson, Binomiales negativos	Log
* Binomial	Logit
* Gamma	Recíproca, Log

Modelos Generales y Generalizados

Dependiendo de qué naturaleza tenga nuestra variable respuesta, trabajaremos con diferentes tipos de modelos en nuestros análisis de regresión:

Modelos Generales Lineales (LM)

Distribución Gaussiana

Modelos Generalizados Lineales (GLM)

Distribución de Gamma
Distribución de Poisson
Distribución Binomial Negativa
Distribución Binomial o beta-binomial (proporciones)
Distribución Multinomial (ordinal o no ordinal)

Modelos Generalizados No-Lineales (aditivos, GAM)

Se establecen relaciones no-lineales mediante *thin plate splines*
Con distribuciones Gaussiana, Gamma, Poisson, Binomial Negativa y Binomial

Si la variable respuesta no es ni Gamma, ni Poisson, ni Binomial Negativa, pero tampoco presenta características de una Gaussiana, entonces podemos:

transformamos la variable respuesta y aplicamos Modelos Generales Lineales
utilizar **modelos para datos "inflados" de ceros**

Tipos de sumas de cuadrados (o sus equivalentes con devianza)

* Si queremos trabajar con **efectos parciales** (cada predictora controlada por las restantes) utilizaremos el **tipo III** de Suma de Cuadrados (**SS**). Es el más utilizado.

* Si queremos efectuar una **estima secuencial de efectos** (según la ordenación que hemos definido en en la fórmula de nuestro modelo), utilizaremos el **tipo I** de **SS**.

En este esquema el primer efecto no se controla por ningún otro
el segundo efecto es controlado por el primero
el tercer efecto es controlado por los dos previos
el cuatro efecto es controlado por los tres previos …

El orden de efectos afecta a los resultados, a no ser que todos ellos sean perfectamente independientes (*i.e.*, ortogonales).

El resultado de Respuesta ~ P1+P2+P3+P4 es distinto al de
Respuesta ~ P3+P4+P1+P2
Respuesta ~ P2+P3+P4+P1
……

REQUISITOS Y POSIBILIDADES

En estos diseños cada unidad es una réplica independiente de las demás.
Sólo se efectúa una medida por sujeto muestral.

Podemos asumir que existen:

relaciones lineales entre los predictores y la respuesta (modelos **GLM**) o
relaciones no-lineales (modelos **GAM**)

Asumimos que los residuos se ajustan a una **distribución normal**.

Asumimos que los residuos muestran **homogeneidad de la varianza** a lo largo de las predicciones del modelo.

Podemos combinar variables continuas denominadas **covariantes** y
variables predictoras nominales llamadas **factores**.

No existirán combinaciones de niveles de diferentes factores que carezcan de datos (son diseños sin "celdas vacías").

MODO DE PROCEDER

1) Definimos una relación entre una variable respuesta y otras que la quieren explicar.

respuesta ~ Predictores

En la medida de lo posible se sustentará en un esquema de hipótesis funcionales.

2) Valoramos cuál es la naturaleza de la variable **respuesta**.

qué valores toma (ceros, negativos, números enteros o decimales, estados)

cuál es la forma geométrica de su distribución

histogramas (para muchos datos)

qq plots (especialmente para número de valores modesto; e.g. < 50)

estimamos los parámetros media y sd para a partir de ellos estimar

mean y *sd* (Gaussiana)

lambda (Poisson), *mu* y *size* (Binomial Negativa)

shape y *scale* (Gamma)

generamos una distribución de datos para valorar el ajuste a la **respuesta**

usando *rnorm*, *rpois*, *rnbinom*, *rgamma*

comparamos las formas (histogramas y qqplots) de ambas distribuciones

3) En función de la distribución de la **respuesta** elegimos un modelo GLM

si tenemos dudas entre varios tipos de distribuciones obtenemos los modelos que serán comparados usando **Akaike AICc** y las propiedades de sus **residuos**

MODO DE PROCEDER

4) Obtenemos los **residuos del modelo** (¡¡ de devianza !!) y valoramos
   su normalidad de manera
      visual (histogramas y qqplots)
      paramétrica global (test de Shapiro-Wilk)
      paramétrica de sesgo y test de D'Agostino (poco efecto sobre errores I y II)
      paramétrica de kurtosis y test de Anscombe-Glynn (más efecto sobre errores I y II)
   homocedasticidad de los residuos (valorada en un plano X – Y)
      ponemos los residuos de devianza (Y) frente a las predicciones del modelo (X)
      que haya dispersión aleatoria de los residuos de las unidades muestrales
   consideramos los residuos de manera individualizada (por unidad muestral)
      dffits (residuo real – residuo sin considerar ese valor en el modelo)
      distancia de Cook (cómo de distante es esa unidad muestral del modelo)
      residuo estudentizado (pertenencia al modelo mediante la t de Student)
      dfbetas (cómo altera cada unidad muestral los coeficientes del modelo)
   estimamos el *leverage* (cómo de extremos son los valores de las predictoras)

5) **Linealidad** entre respuesta y predictores (biplots para no detectar curvilinealidad)

6) Valoramos el **grado de independencia entre las variables predictoras** (P1, P2, P3, …).
   exploración visual mediante múltiples plots de P1-P2-P3-…
   parametrización mediante la estima de VIF (descartamos predictores VIF > 6)

MODO DE PROCEDER

8) Repasamos nuestros datos para valorar:
   errores en los valores de la matriz de datos
   eliminar unidades muestrales del análisis
   redundancia entre las variables predictoras
   necesidad de usar otro modelo GLM con otra distribución para la respuesta

9) Test global de **significación del modelo** (*omnibus test*)
   likelihood ratio test comparando el modelo de interés con el nulo (sin predictoras)
   valores de AICc comparando el modelo de interés con el nulo (sin predictoras)

10) Test de la **significación de cada predictora**
   test de coeficientes de regresión usando los grados de libertad correctos
   tests Anova y *likelihood ratio*

11) **Interpretación de los coeficientes**
   signo indicativo de relaciones positivas o negativas
   en la escala de la función de vínculo utilizada (*exp*(coef), *odds-ratios*, …)
   parametrización comparable entre diferentes predictoras (*beta coefficients*)

12) En modelos GLM asumiendo distribuciones Poisson o Binomiales
   estima de la **sobredispersión** de los residuos y modelos *quasi*-…
   volvemos al paso 10) para estimar la significación de los efectos

MODO DE PROCEDER

13) **Variación explicada** (partición de la *devianza*):
de todo el modelo
atribuible a cada predictora
poder predictivo utilizando validación cruzada

14) **Visualización del efecto de las predictoras**
partial influence o *partial residual plots*
efectos parciales mediante las funciones *allEffects* y *visreg*

15) Si ha existido **heterocedasticidad en los residuos**
corregimos la heterocedasticidad (HC3, HC4, HC4m, HC5)
nuevas estimas de significación de las predictoras (no cambian los coeficientes)

16) **Si han existido puntos influyentes y/o perdidos** (que hemos decidido no eliminar)
estimas robustas de los modelos GLM

17) Si con el mejor modelo GLM **hemos incumplido bastantes supuestos** necesarios
estima de modelos con ausencia de efectos (*permutation tests***)**
re-parametrización del modelo mediante muestreo con reemplazo (*bootstrapping*)
no usamos estadística frecuentista derivadas de P
valoramos si los intervalos de confianza incluyen el valor "cero"

TIPOS DE DISTRIBUCIONES

continuas: Normal, Gamma

discretas: Poisson, Binomial Negativa

Binomial

"infladas" de ceros

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana)

Se describe por dos parámetros:

- estadístico central o media
- estadístico de dispersión o desviación típica (sd)
  - la varianza = sd^2

Distribuciones geométricas simétricas

… nos inventamos números (generados al azar según unas propiedades predefinidas)

```
## variables normales generadas al azar; 10.000 datos
var.normal <- rnorm(n=10000, mean=4, sd=2)

## conozcamos su media y desviación típica
mean(var.normal)
sd(var.normal)
```

```
> mean(var.normal)
[1] 4.018361
> sd(var.normal)
[1] 2.004826
```

> aparece en la consola como resultado de una instrucción ejecutada

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana)

Representamos el histograma y el Q-Q plot:

abline, curve y qqline bajo un comando plot, hist, qqnorm, … añaden líneas al gráfico
- main añade un título principal
- v para vertical; h para horizontal
- col es color; lwd para el grosor de la línea

```
## el histograma
hist(var.normal, main="variable al azar: media=4 y sd=2")
## con breaks definimos el número de columnas
hist(var.normal, main="variable al azar: media=4 y sd=2", breaks=30)
abline(v=0, col="red", lwd=2)
abline(v=mean(var.normal), col="blue", lwd=2)

## y ahora el Q-Q plot
qqnorm(var.normal, main="variable al azar: media=4 y sd=2")
qqline(var.normal, col="red", lwd=2)
```

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana)

Representamos el histograma y el Q-Q plot en un solo plot con dos paneles, con una versión más elaborada de **`hist`**

`c`(…) es un argumento usado para combinar "cosas" según una secuencia dada

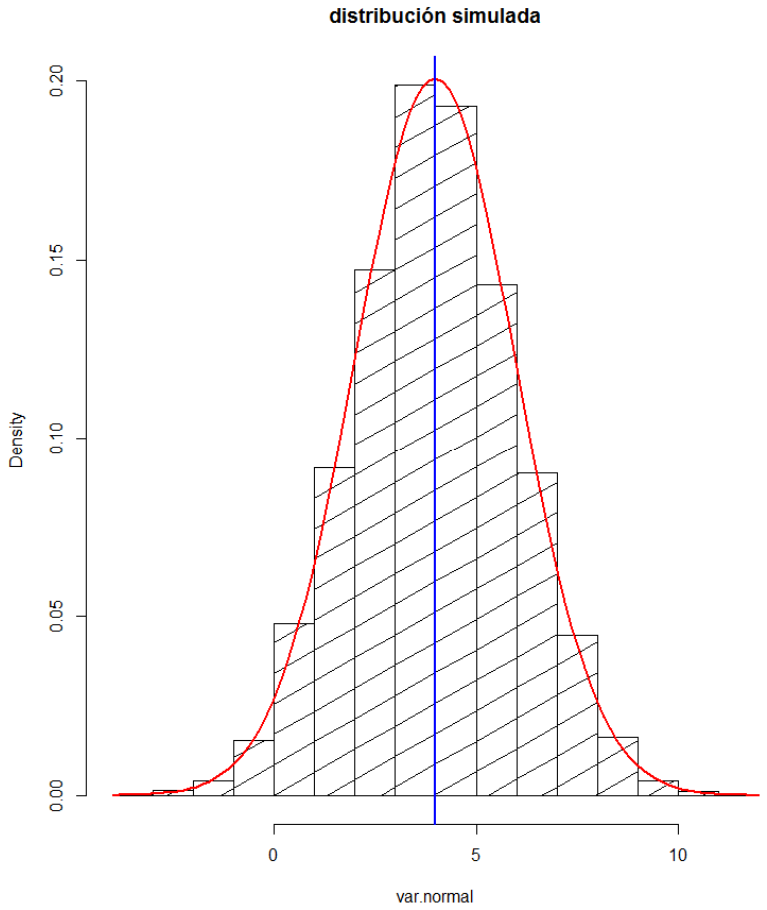
podemos añadir comentarios en una línea de código con `##` …

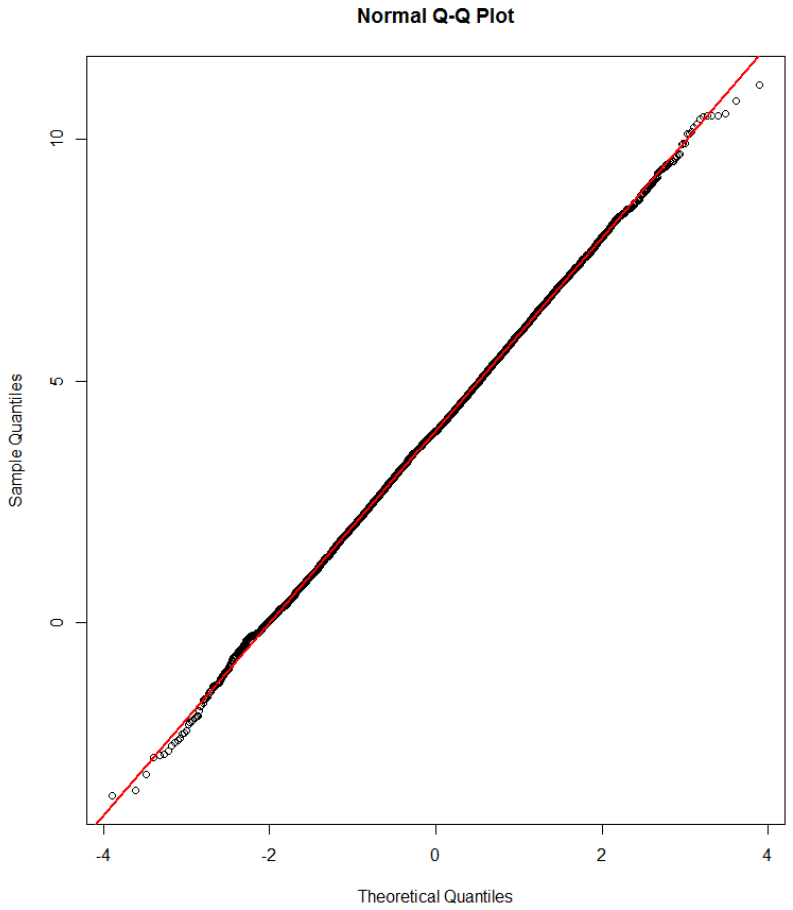
```
par(mfcol=c(1,2))                ## fija los paneles según 1 fila y 2 columnas
hist(var.normal, density=5, freq=FALSE, main="distribución simulada")
curve(dnorm(x, mean=mean(var.normal), sd=sd(var.normal)), col="red", lwd=2,
      add=TRUE, yaxt="n")
abline(v=mean(var.normal), col="blue", lwd=2)
##
qqnorm(var.normal)
qqline(var.normal, col="red", lwd=2)
par(mfcol=c(1,1))                ## volvemos a un gráfico con solo un panel
```

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana)

Representamos el histograma y el Q-Q plot en un solo plot con dos paneles:





TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana) - experimento de variabilidad con tamaño muestral

Una variable "aislada" sólo se define por una dimensión (e.g., `var.normal`).
`var.normal` es un vector de longitud 10.000.

```
length(var.normal)
> length(var.normal)
[1] 10000
```

El dato 253 será: `var.normal[253]`
Los datos 253 y del 401 al 450 serán: `var.normal[c(253, 401:450)]`

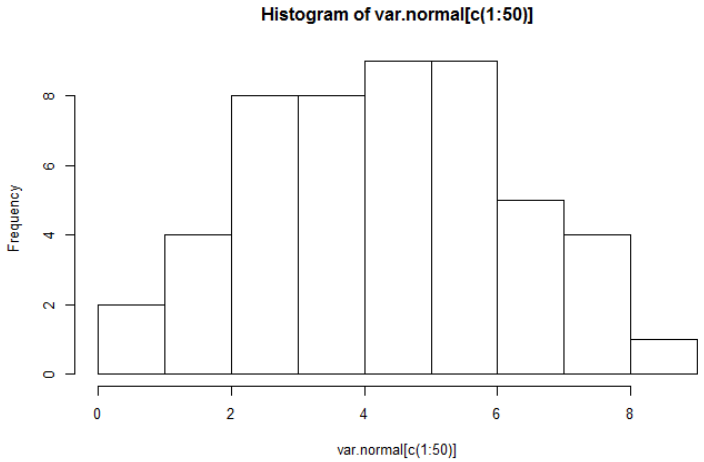
Representamos ahora los histogramas de cuatro subconjuntos de datos seleccionados ciegamente dentro de nuestra variable generada al azar. Y los ponemos en una misma figura con cuatro paneles.

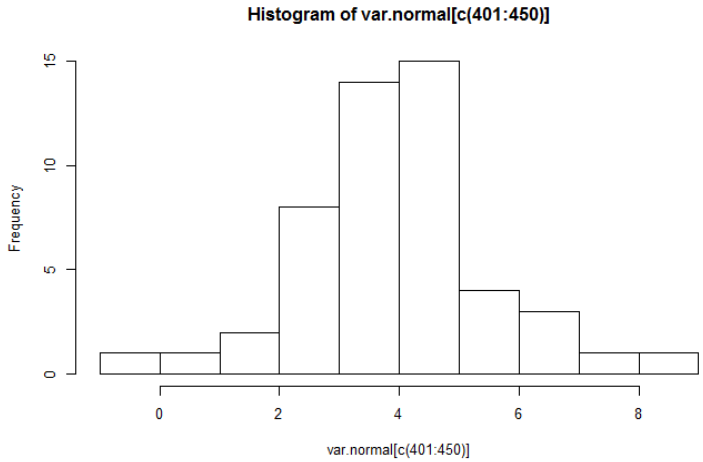
```
par(mfcol=c(2,2))    ## fija los paneles según 2 columnas y 2 filas
hist(var.normal[c(1:50)])
hist(var.normal[c(201:250)])
hist(var.normal[c(401:450)])
hist(var.normal[c(601:650)])
par(mfcol=c(1,1))
```

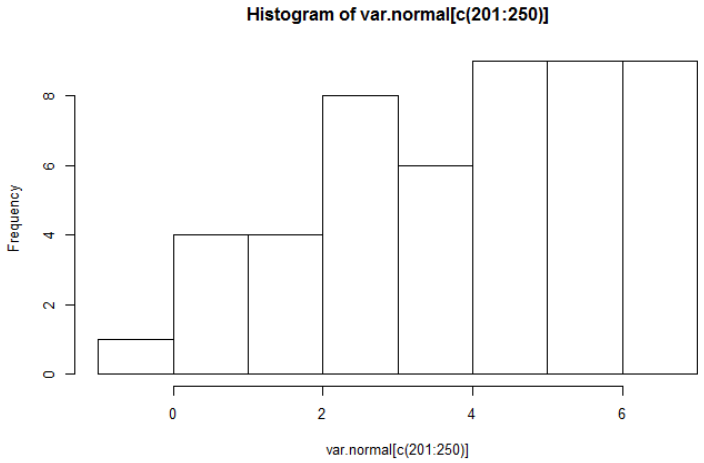
TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

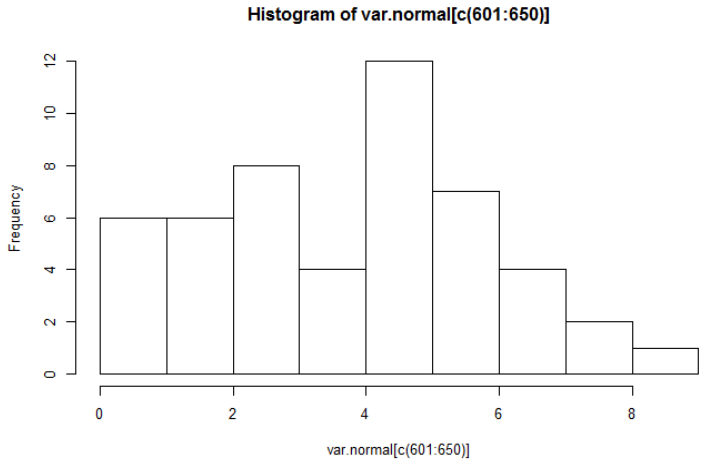
Normal (Gaussiana) - experimento de variabilidad con tamaño muestral

Histogramas de cuatro subconjuntos de datos seleccionados ciegamente









TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana) - experimento de variabilidad con tamaño muestral

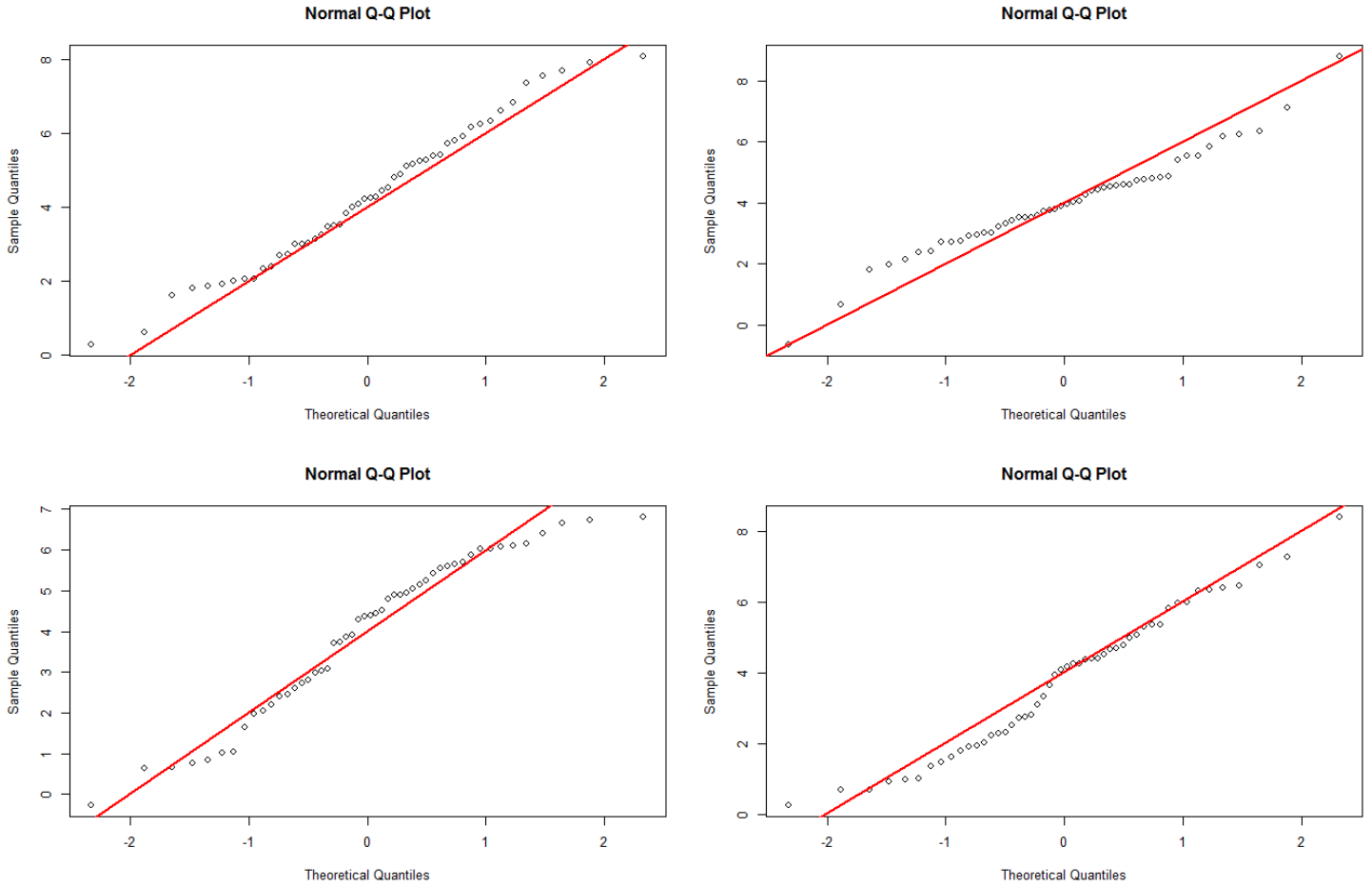
Y ahora los mismos datos pero con Q-Q plots:

```
par(mfcol=c(2,2))    ## fija los paneles según 2 columnas y 2 filas
qqnorm(var.normal[c(1:50)])
  qqline(var.normal, col="red", lwd=2)
qqnorm(var.normal[c(201:250)])
  qqline(var.normal, col="red", lwd=2)
qqnorm(var.normal[c(401:450)])
  qqline(var.normal, col="red", lwd=2)
qqnorm(var.normal[c(601:650)])
  qqline(var.normal, col="red", lwd=2)
par(mfcol=c(1,1))
```

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana) - experimento de variabilidad con tamaño muestral

Y ahora los mismos datos pero con Q-Q plots:



TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana) - experimento de variabilidad con tamaño muestral

Y veamos ahora sus medias y desviaciones para unos datos que vienen de una distribución de media = 4 y varianza = 4 (sd = 2).

```
mean(var.normal[c(1:50)]); sd(var.normal[c(1:50)])
mean(var.normal[c(201:250)]); sd(var.normal[c(201:250)])
mean(var.normal[c(401:450)]); sd(var.normal[c(401:450)])
mean(var.normal[c(601:650)]); sd(var.normal[c(601:650)])
```

```
> mean(var.normal[c(1:50)]); sd(var.normal[c(1:50)])
[1] 4.268634
[1] 1.953344
> mean(var.normal[c(201:250)]); sd(var.normal[c(201:250)])
[1] 3.952677
[1] 1.928345
> mean(var.normal[c(401:450)]); sd(var.normal[c(401:450)])
[1] 3.979421
[1] 1.598691
> mean(var.normal[c(601:650)]); sd(var.normal[c(601:650)])
[1] 3.775069
[1] 2.011229
```

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Normal (Gaussiana) - experimento de variabilidad con tamaño muestral

El mensaje es obvio:

- una cosa es una distribución teórica derivada de un número enorme de datos (con su geometría y parámetros canónicos)
- y otra lo que representa un subconjunto “modesto” de esos datos
- haremos exploraciones visuales “más atinadas” con Q-Q plots que con histogramas

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Gamma

Es una distribución definida por dos parámetros, *shape* y *scale*.

Se puede asociar a una binomial negativa pero para medidas que no son conteos.

Otra forma de definirla es mediante el parámetro tasa (*rate*) = 1/*scale*

Con esta distribución se cumple que:

```
media = shape * scale
varianza = media * scale
varianza = shape * scale^2
```

shape = media^2 / sd^2
scale = sd^2 / media

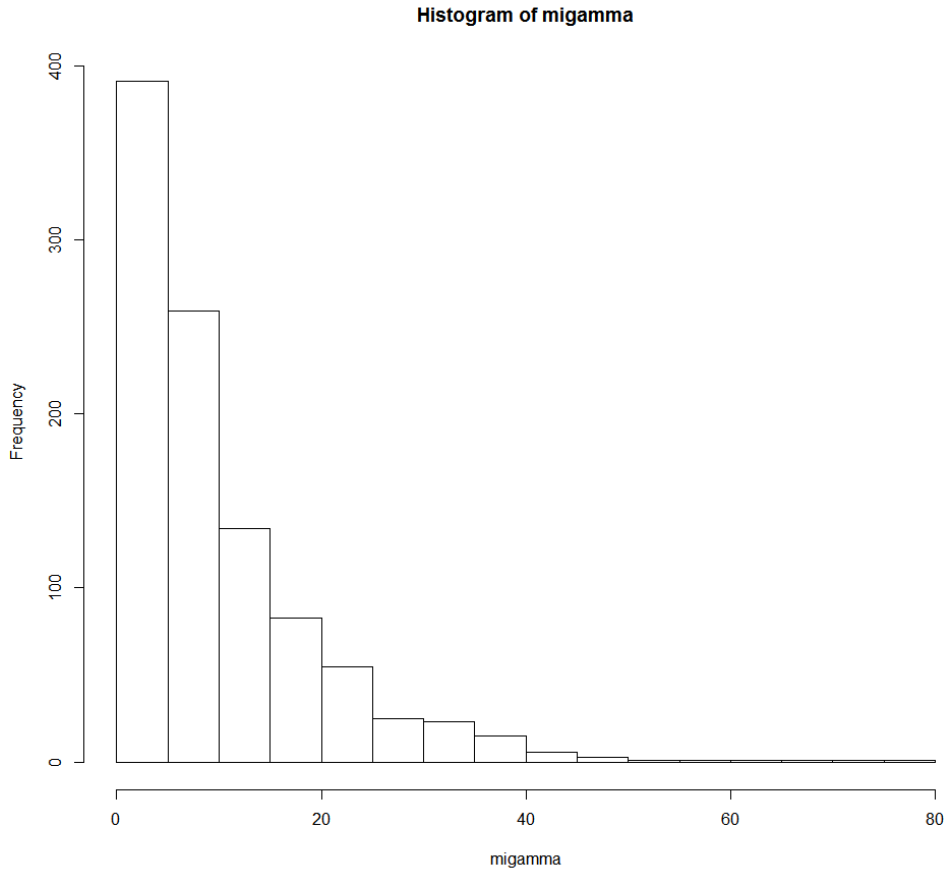
Definamos una distribución aleatoria y veamos su forma:

```
migamma <- rgamma(n=1000, shape=1, scale=10)
hist(migamma)
mean(migamma)
sd(migamma)^2

migamma2 <- rgamma(n=1000, shape=10, scale=1)
hist(migamma2)
mean(migamma2)
sd(migamma2)^2
```

TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Gamma

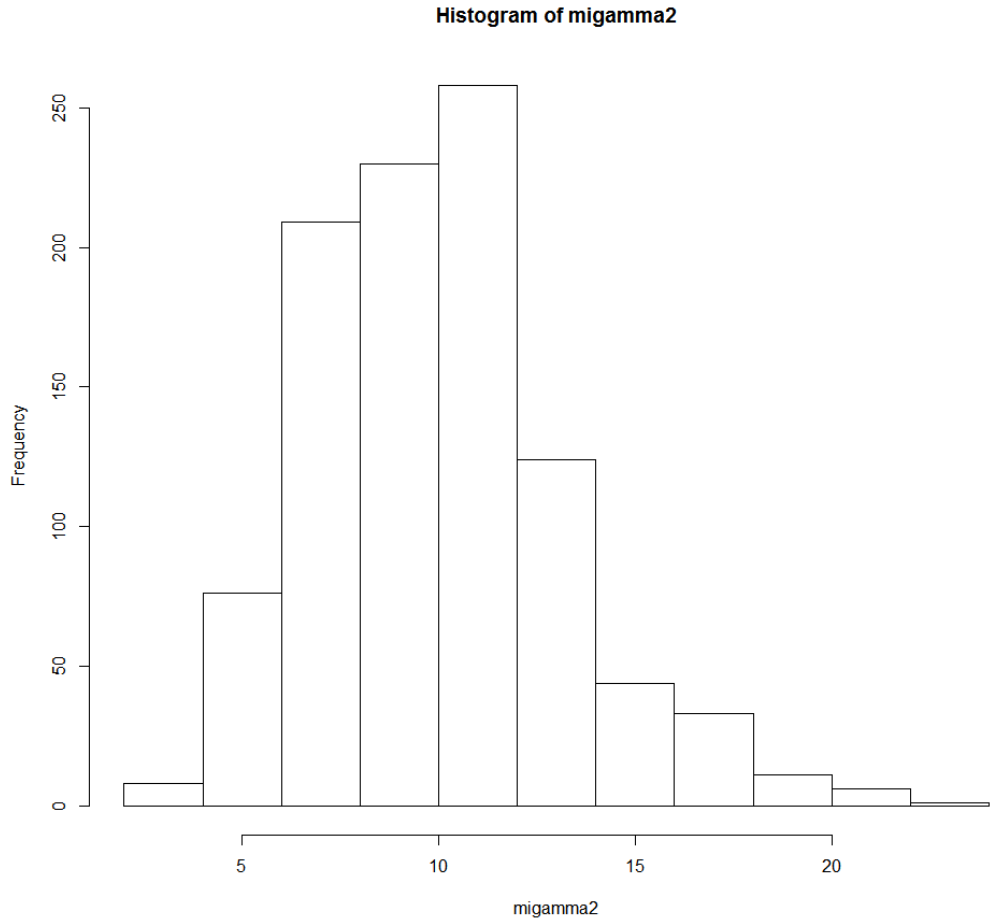
```
> migamma <- rgamma(n=1000, shape=1, scale=10)
> mean(migamma)
[1] 9.858615
> sd(migamma)^2
[1] 100.413
...
```



TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Gamma

```
> migamma2 <- rgamma(n=1000, shape=10, scale=1)
> mean(migamma)
[1] 9.940545
> sd(migamma)^2
[1] 10.02372
...
```



TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Poisson

Es una distribución con las siguientes características:

- son números enteros (generalmente conteos)
- no puede presentar valores negativos
- su distribución se describe por un solo parámetro (lambda), de manera que la media es igual a la varianza de la distribución (lambda → media = sd^2)

```
## generamos valores aleatorios según una Poisson
## 10.000 valores, con media = varianza = 4
## y para que a todos nos de igual fijamos una extracción
##   aleatoria con el comando set.seed(…)
set.seed(699)
var.poisson <- rpois(n=10000, lambda=4)

## veamos qué media y varianza (sd^2) han acabado teniendo
mean(var.poisson)
sd(var.poisson)^2
> mean(var.poisson)
[1] 3.9979
> sd(var.poisson)^2
[1] 3.971093
```

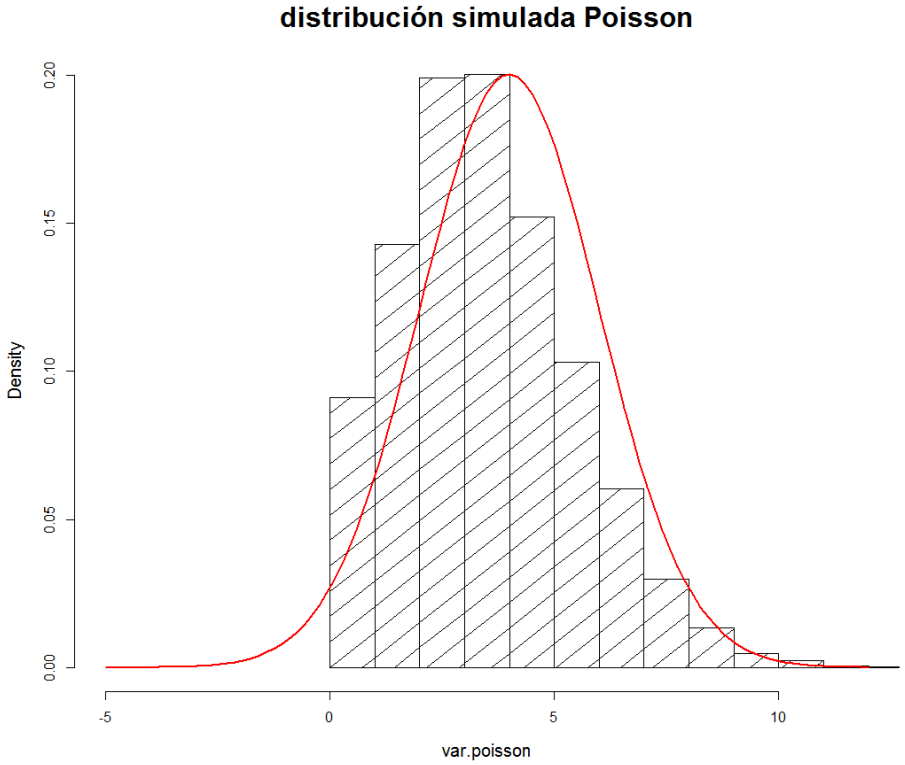
TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Poisson

Hacemos una exploración visual (asimétricas y truncadas en el cero)

No se ajustan a una distribución de campana Gaussiana

```
## primero el histograma de la Poisson (con otros argumentos)
hist(var.poisson, density=5, freq=FALSE, main="distribución simulada Poisson",
     cex.main=2, cex.lab=1.25, xlim=c(-5, 12))
curve(dnorm(x, mean=mean(var.poisson), sd=sd(var.poisson)),
      col="red", lwd=2, add=TRUE, yaxt="n")
```



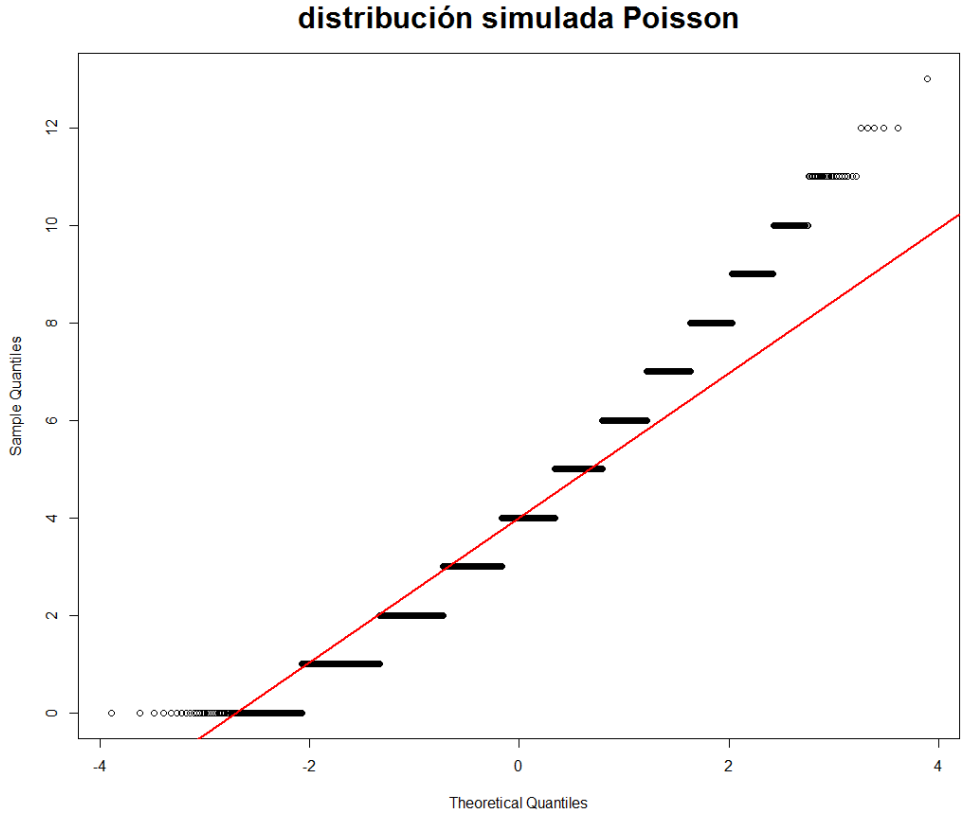
TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Poisson

Hacemos una exploración visual (asimétricas y truncadas en el cero)

No se ajustan a una distribución de campana Gaussiana

```
## luego el Q-Q plot de la Poisson
qqnorm(var.poisson, main="distribución simulada Poisson", cex.main=2)
qqline(var.poisson, col="red", lwd=2)
```

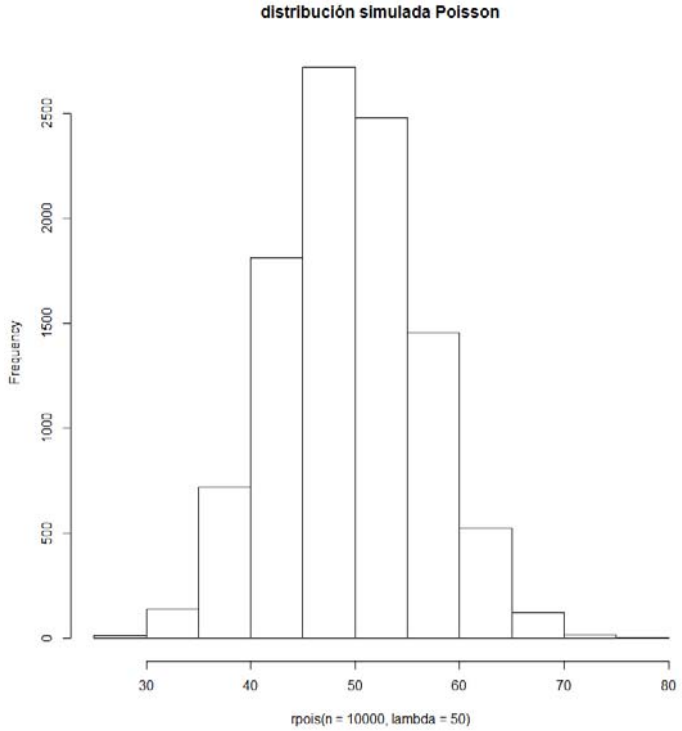


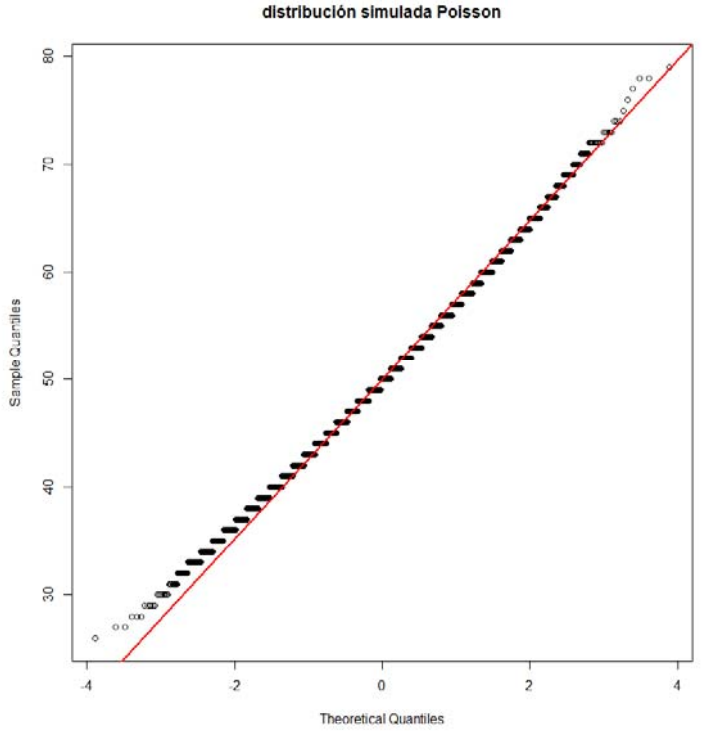
TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Poisson … parecida a una Gaussiana

Distribuciones de Poisson con valor de lambda alto (*i.e.*, distante de cero) pueden asemejarse a una Gaussiana

```
par(mfcol=c(1,2))
hist(rpois(n=10000, lambda=50), main="distribución simulada Poisson")
qqnorm(rpois(n=10000, lambda=50), main="distribución simulada Poisson")
qqline(rpois(n=10000, lambda=50), col="red", lwd=2)
par(mfcol=c(1,1))
```





TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Binomial negativa

Distribuciones más dispersadas que una Poisson (varianza > media).

De nuevo, son distribuciones de números enteros (generalmente conteos),

que carecen de números negativos

y se describen por dos parámetros:

uno es la media ("mu")

y otro es el inflado de la varianza (denominado "size" en R)

La varianza viene definida por: mu + (mu^2)/size

Cuando size es muy grande, 1/size tiende a cero, y la varianza acaba pareciéndose a mu.

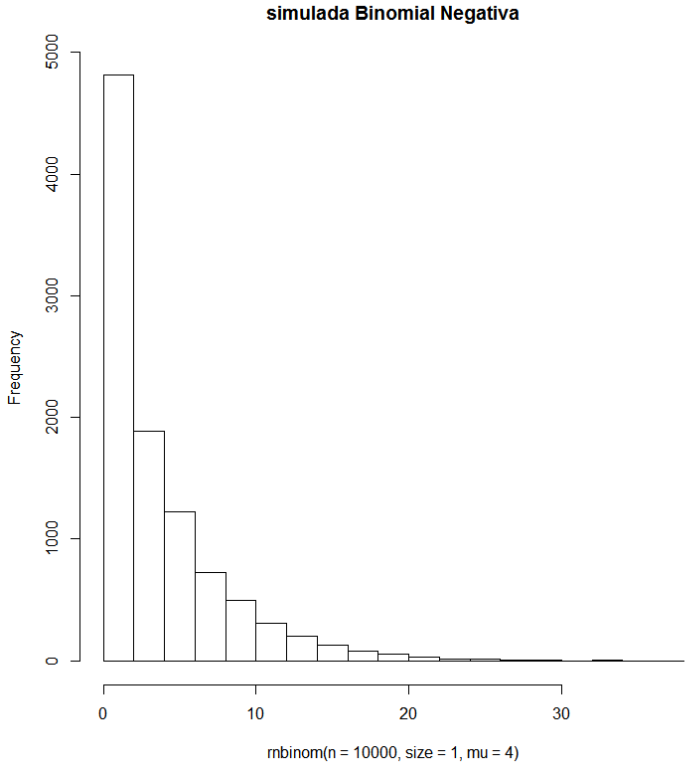
```
## generamos valores aleatorios según una Binomial Negativa
## 10.000 valores, con media = 4 y size = 1
var.negbin <- rnbinom(n=10000, size=1, mu=4)

## veamos qué media y varianza han acabado teniendo
mean(var.negbin)
sd(var.negbin)^2
> mean(var.negbin)
[1] 4.0357
> sd(var.negbin)^2
[1] 20.88551
```

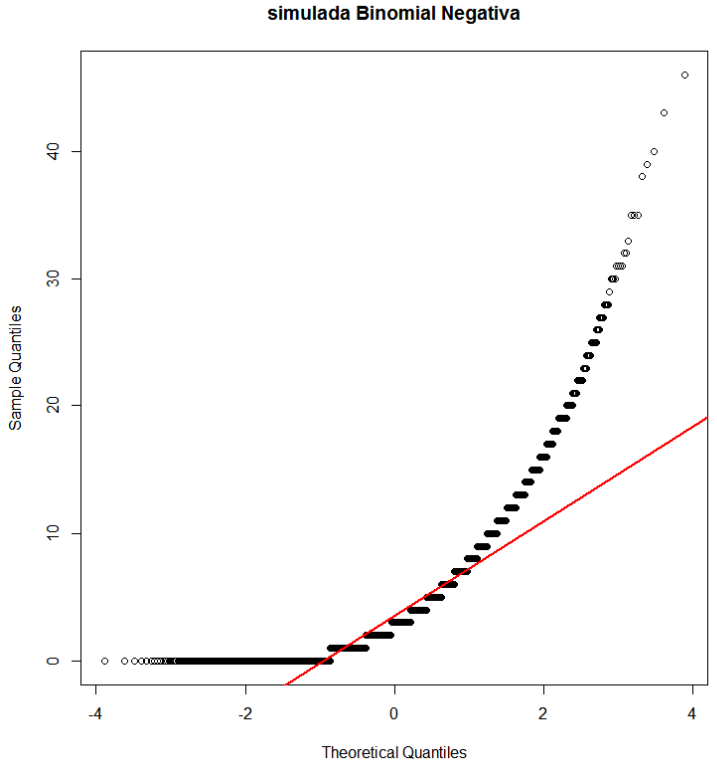
TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

Binomial negativa

Y la representamos:

```
par(mfcol=c(1,2))
hist(rnbinom(n=10000, size=1, mu=4), main="simulada Binomial Negativa")
qqnorm(rnbinom(n=10000, size=1, mu=4), main="simulada Binomial Negativa")
qqline(rnbinom(n=10000, size=1, mu=4), col="red", lwd=2)
par(mfcol=c(1,1))
```





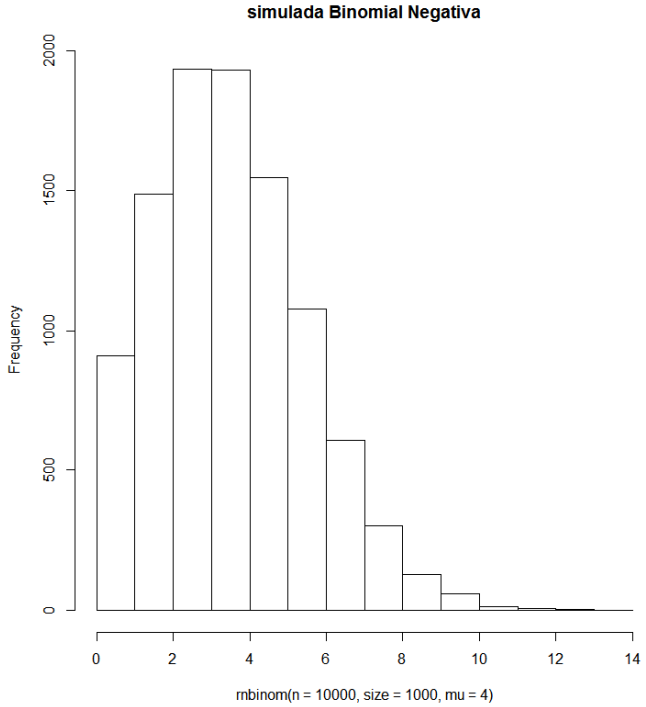
TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

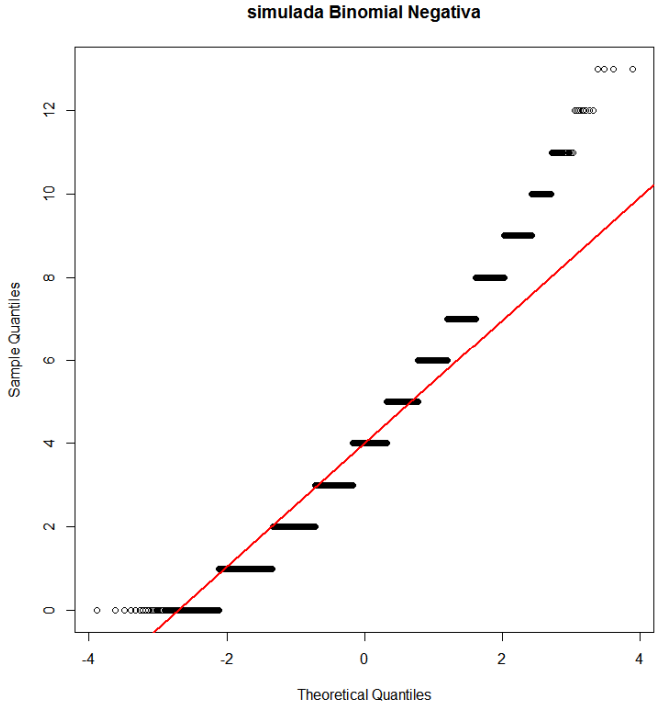
Binomial negativa … aproximada a una Poisson

Cambiamos ahora el parámetro de sobredispersión size a 1000:

```
par(mfcol=c(1,2))
hist(rnbinom(n=10000, size=1000, mu=4), main="simulada Binomial Negativa")
qqnorm(rnbinom(n=10000, size=1000, mu=4), main="simulada Binomial Negativa")
qqline(rnbinom(n=10000, size=1000, mu=4), col="red", lwd=2)
par(mfcol=c(1,1))
```

```
> mean(rnbinom(n=10000, size=1000, mu=4))
[1] 4.0041
> sd(rnbinom(n=10000, size=1000, mu=4))^2
[1] 4.005399
```





TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

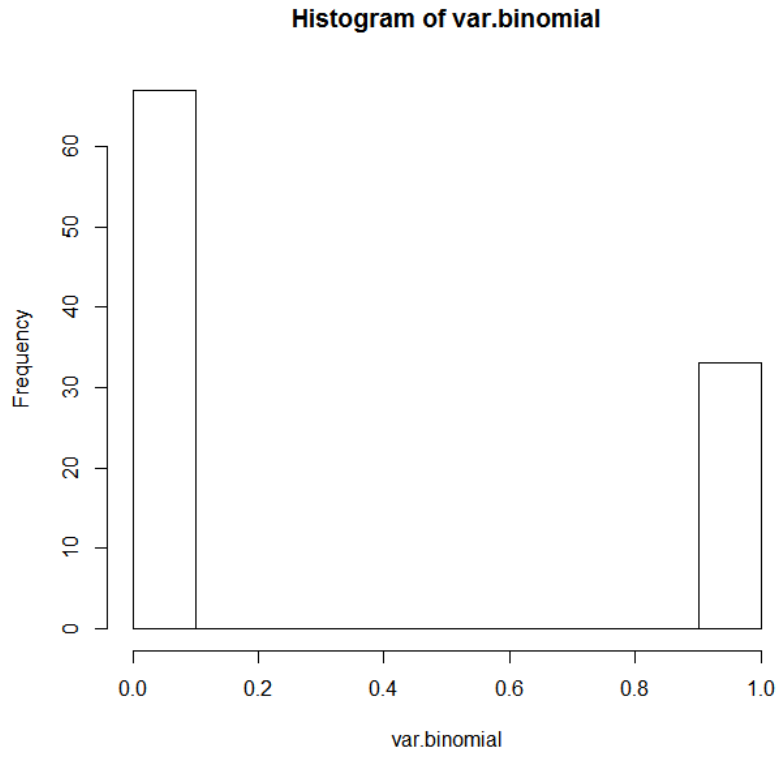
Binomial

Toma dos estados definidos con una probabilidad asociada a sus estados.
Definimos **cuántos estados "no cero"** deseamos con el argumento `size`
Y con el argumento `probs` establecemos su **probabilidad de ocurrencia**

```
var.binomial <- rbinom(n=100, size=1, prob=0.3)
```

```
0 0 0 0 0 1 1 1 1 0 1 0 0 0 1 0 0 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 1 1 0 0 0 0 0 0 1 0 1 0
0 1 1 1 0 1 1 1 1 1 0 0 0 0 1 0 0 1 1 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0
0 0 1 1 0 0 0 1 1 0 0 0
```

```
hist(var.binomial)
```



TIPOS DE DISTRIBUCIONES DE LA VARIABLE RESPUESTA

"infladas" de ceros

Suponen una mezcla de distribuciones binomiales (estados 0 y conteos>0)
y distribuciones de conteos tipo Poisson o Binomiales Negativas (B-N).

Existe una mayor frecuencia de ceros que la que cabría esperar de una Poisson o B-N.

Los **conteos "cero"** pueden proceder de una binomial o de la Poisson o B-N.

```
var.binomial <- rbinom(n=20, size=1, prob=0.5)
estados_0 <- length(var.binomial) - sum(var.binomial)
var.poisson <- rpois(n=100-estados_0, lambda=1)
var.infladadeceros <- c(rep(0, times=estados_0), var.poisson)
```

```
> var.infladadeceros
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 3 2 0 0 1 1 2 4 1 0 0 4 1 0 1 1 1 2 1 1 1 0
0 1 0 0 2 2 2 3 1 1 0 2 0 0 2 1 1 1 2 0 0 4 0 0 0 4 1 0 0 1 2 2 0 1 0 1 0 0
2 2 0 1 0 2 1 2 0 1 0 1 1 1 2 0 1 0 3 2 1 1 0 0
```

construyendo
modelos generalizados

CREACIÓN DEL MODELO GENERALIZADO

En los modelos de regresión lineal clásicos (Gaussianos):
* definimos una función predictora

$$g(x) = \alpha+\beta_1 X_1+ \ldots + \beta_p X_p$$ para p predictores

* establecemos la relación lineal g con la respuesta

$$Y = g(x) + \varepsilon$$ siendo ε la variación residual

En los modelos generalizados de Poisson:
* establecemos el valor esperado de la respuesta Y (representado por su media μ)
* que establece una relación logarítmica con la función predictora $g(x)$

$$\log(\mu) = g(x) + \varepsilon \quad \text{o} \quad \mu = e^{g(x)} + \varepsilon'$$

$$\mu = e^{\alpha+\beta_1 X_1+ \ldots + \beta_p X_p} + \varepsilon'$$

* esta estructura es muy importante para la interpretación de los coeficientes de regresión.

* ε' es la devianza residual

MODELOS GENERALIZADOS LINEALES CON R

Definición de los modelos atendiendo a la distribución de errores

Según qué tipo de variable respuesta tengamos definiremos la familia y la función de vínculo.

eqt es la ecuación que especifica la relación entre la respuesta y las variables predictoras

los clásicos (Gaussianos, Gamma, Poisson, Binomial Negativo, Binomial)

```
modelo <- glm(eqt, data=datos, family=gaussian(link="identity"))
modelo <- lm(eqt, data=datos)

modelo <- glm(eqt, data=datos, family=Gamma(link="log"))
modelo <- glm(eqt, data=datos, family=Gamma(link="inverse"))

modelo <- glm(eqt, data=datos, family=poisson(link="log"))
modelo <- glm(eqt, data=datos, family=quasipoisson(link="log"))

modelo <- glm.nb(eqt, data=datos, link=log)

modelo <- glm(eqt, data=datos, family=binomial(link="logit"))
modelo <- glm(eqt, data=datos, family=binomial(link="cloglog"))
                cloglog trabaja mejor con distribuciones extremadamente sesgadas
modelo <- glm(eqt, data=datos, family=quasibinomial(link="logit"))
```

MODELOS GENERALIZADOS LINEALES CON R

cuando la variable respuesta es una proporción acotada entre 0 y 1 (excluidos)

modelos *beta-binomiales*

```
modelo <- betareg(eqt, data=datos, link="logit")
```

cuando la variable respuesta es una multinomial con estados ordenados

modelos *proportional odds-regression*

```
modelo <- polr(eqt, data=datos, method="logistic", Hess=TRUE)
          method="cloglog" para distribuciones extremadamente sesgadas
```

cuando la variable respuesta está "inflada" de ceros

(más que lo esperable por una Poisson o una Binomial Negativa –BN–)
se establecen relaciones para:

la parte binomial ceros *vs.* conteos no-ceros (cualquier número ≥1 se hace 1)

y para la **parte conteos** no-ceros (para valores ≥1)

ziformula= es para la parte binomial

~. denota lo mismo que lo establecido para la parte de conteo

modelos *zero-inflated*: tienen dos fuentes de ceros

los derivados de la binomial y los que pueden proceder de la Poisson o BN

modelos *hurdle*: con una sola fuente de ceros procedente de la parte binomial

la parte Poisson o Binomial Negativa (BN) "no aportan" ceros

MODELOS GENERALIZADOS LINEALES CON R

cuando la variable respuesta está "inflada" de ceros

son modelos "creativos" en los que se pueden especificar diferentes tipos de relaciones
podemos elaborarlos con dos paquetes: glmmTMB:`glmmTMB`
pscl:`zeroinfl` pscl:`hurdle`

modelos *zero-inflated* (con dos fuentes de ceros) para Poisson o Binomiales Negativas

sencillos: asumen que todos los conteos cero tienen la misma probabilidad de pertenecer al componente cero
no se modeliza la probabilidad de pertenecer a la componente cero con predictores P1 P2 P3 …
se especifican con el intercepto (~1) en la parte Zero-inflation (ziformula)

```
modelo <- glmmTMB(eqt, ziformula=~1, data=datos, family=poisson)
modelo <- glmmTMB(eqt, ziformula=~1, data=datos, family=nbinom2)
modelo <- zeroinfl(respuesta~P1+P2+P3 | 1, data=datos, family=nbinom2)
```

complejos: *se modeliza* la probabilidad de pertenecer a la componente cero con predictores P1 P2 P3 …
~. denota lo mismo que lo establecido para la parte de conteo

```
modelo <- glmmTMB(eqt, ziformula=~., data=datos, family=poisson)
modelo <- glmmTMB(eqt, ziformula=~., data=datos, family=nbinom2)
modelo <- zeroinfl(respuesta~P1+P2+P3 | P1+P2+P3, data=datos, family=nbinom2)
```

modelos *hurdle* (con una sola fuente de ceros)

complejos: *se modeliza* la probabilidad de pertenecer a la componente cero con predictores P1 P2 P3 …
~. denota lo mismo que lo establecido para la parte de conteo

```
modelo <- glmmTMB(eqt, ziformula=~., data=datos, family=truncated_poisson)
modelo <- glmmTMB(eqt, ziformula=~., data=datos, family=truncated_nbinom2)
modelo <- hurdle(respuesta~P1+P2+P3 | P1+P2+P3, data=datos, family=nbinom2)
```

MODELOS GENERALIZADOS NO-LINEALES CON R

modelos GAM (*Generalized Additive Models*)

para distribuciones Gaussiana, Gamma, Poisson, Binomial Negativa, Binomial

existen varios procedimientos matemáticos para abordarlos

la fórmula (eqt) incluye términos no lineales *thin plate splines* con la siguiente expresión:

y ~ x + z + s(w, k=5) + factor + s(v, k=10) … (x, z y factor son lineales)

si no se especifica k (*basis dimension*) por defecto se establece en 10

Método GCV.Cp: "*general cross validation for unknown scale parameter*"

scale=-1 corrige por sobredispersión SÓLO en variables respuestas Binomiales o Poisson

el argumento gamma suaviza las relaciones curvilíneas evitando el sobreajuste

ejemplo para una Poisson

```
modelo <- gam(eqt, data=datos, family=poisson(link="log"),
              method="GCV.Cp", gamma=1.4, scale=0)
```

Método ML o REML: "*Maximum Likelihood*" o "*Restricted Estimation Maximum Likelihood*"

por defecto REML en Binomiales Negativas (family=nb(link="log"))

ejemplo para una Poisson

```
modelo <- gam(eqt, data=datos, family=poisson(link="log"),
              method="ML", scale=0, optimizer=c("outer","newton"))
```

ejemplo para una Binomial Negativa

```
modelo <- gam(eqt, data=datos, family=nb(link="log"),
              method="ML", scale=0, optimizer=c("outer","newton"))
```

MODELOS GENERALIZADOS NO-LINEALES CON R

modelos GAM (*Generalized Additive Models*)

para más detalles sobre modelos GAM y los *splines* consultad:

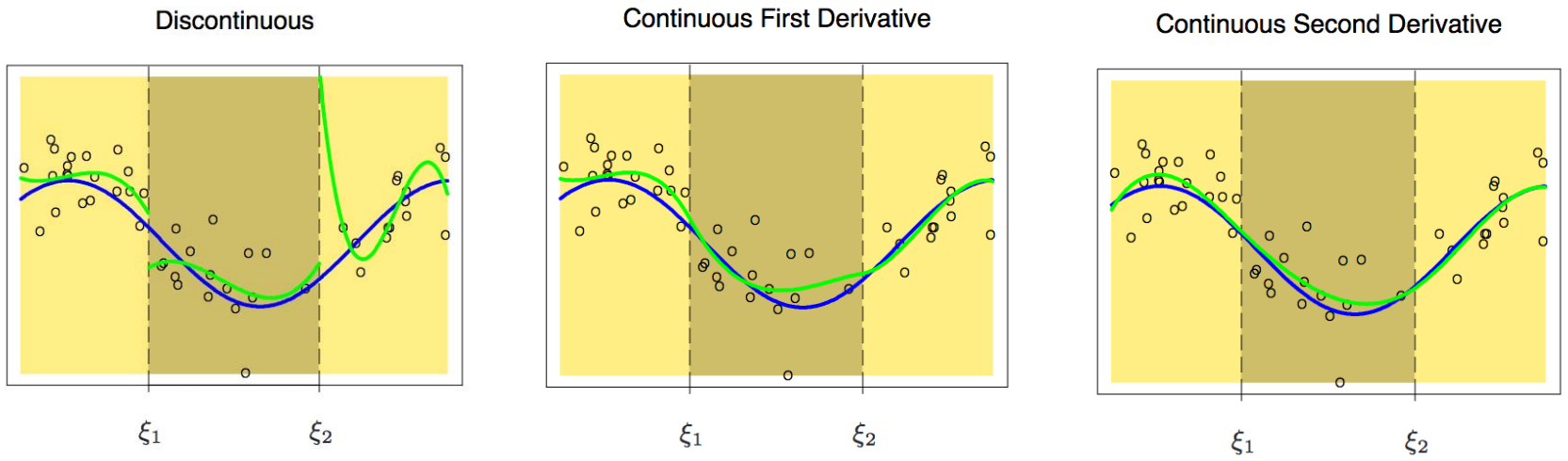
https://reseau-mexico.fr/sites/reseau-mexico.fr/files/igam.pdf (paginas 222, 224)

unas buenas introducciones a los *splines* las tenéis en:

https://www.analyticsvidhya.com/blog/2018/03/introduction-regression-splines-python-codes/

Una *spline* de regresión cúbica es una curva no lineal construida a partir de secciones de polinomios cúbicos unidos para que la curva sea continua.

En la figura, la *spline* de regresión cúbica (línea azul) se compone de tres secciones de polinomios cúbicos (líneas verdes) que se fuerzan para que se unan en los "nodos" del *spline* final (ε1, ε2).



MODELOS GENERALIZADOS NO-LINEALES CON R

modelos GAM (*Generalized Additive Models*)

Esta herramienta también permite definir planos de interacción de efectos
existen diferentes modos de hacerlo: con **s(x, z)**, **te(x, z)**, **ti(x, z)**

s(x, z): plano de interacción cuando **x** y **z** se miden en la misma escala (isotrópicas)
sí se utiliza el mismo grado de suavizado para **x** y **z**
y se controlan los efectos principales s(**x**) y s(**z**)

te(x, z): plano de interacción cuando **x** y **z** no se miden en la misma escala y cuando
no se utiliza el mismo grado de suavizado para **x** y **z**
y se controlan los efectos principales s(**x**) y s(**z**)

ti(x, z): se estima sólo el término interacción (**x, z**)
sin estimar los efectos principales s(**x**) y s(**z**)
si lo quisiésemos hacer tendríamos que añadirlos: **ti(x)+ti(z)+ti(x, z)**

Una covariante con *spline* se puede anidar dentro de los niveles de un FACTOR
produciendo tantos *splines* como niveles tenga el factor
s(covariante, by=FACTOR)

Podemos comparar la adecuación de diferentes modelos a nuestros datos utilizando **Akaike AICc**, pero utilizando el mismo procedimiento de estima.
Compararemos modelos construidos usando `"GCV.Cp"` o `"ML"` sin mezclarlos.

MODELOS GENERALIZADOS

Ahora vamos a **correr nuestro modelo**:

```
## antes cargamos la siguiente línea de código para obtener
## los mismos resultados que STATISTICA y SPSS utilizando type III SS
options(contrasts=c(factor="contr.sum", ordered="contr.poly"))
```

***R* por defecto** usa las tablas de contraste `contr.treatment`. Y esto no es lo correcto.

En la ayuda de R para el comando `contr.treatment` podemos leer:

`contr.treatment contrasts each level with the baseline level (specified by base)…`

`Note that` **`this does not produce 'contrasts' as defined in the standard theory for linear models as they are not orthogonal to the intercept`**.

```
## aquí el modelo:
eqt <- as.formula(nspp ~ altmed+rangoalt+shannon+tempmin+precip +
                         offset(log(ntransectos)))
```

Pero … **¿dónde están mis datos?**

Nuestros datos (que yo aquí los llamo datos) los asociamos al modelo a través del argumento `data` incluido en el comando lm o glm que es el que hace el Modelo General/Generalizado Lineal.

```
modelo <- glm(eqt, data=datos, family=gaussian(link="identity"))
```

MODELOS GENERALIZADOS

Volvamos a nuestro modelo

para ver el resultado de nuestro modelo

```
summary(modelo)
```

nuestro modelo es un objeto de R con muchas cosas dentro; ¿qué tiene?

```
names(modelo)
```

```
> names(modelo)
 [1] "coefficients"      "residuals"         "fitted.values"
 [4] "effects"           "R"                 "rank"
 [7] "qr"                "family"            "linear.predictors"
[10] "deviance"          "aic"               "null.deviance"
[13] "iter"              "weights"           "prior.weights"
[16] "df.residual"       "df.null"           "y"
[19] "converged"         "boundary"          "model"
[22] "call"              "formula"           "terms"
[25] "data"              "offset"            "control"
[28] "method"            "contrasts"         "xlevels"
```

MODELOS GENERALIZADOS

Volvamos a nuestro modelo

`model` contiene todos los datos de las variables respuesta y predictoras usadas, y es de gran utilidad por si queremos trabajar *a posteriori* con los datos ya seleccionados para su uso.
La primera columna es siempre la variable respuesta.

para ver sus datos
```
modelo$model
```
para ver la variable respuesta usada
```
modelo$model[,1]
```
y su nombre
```
names(modelo$model[1])
```

Los **residuos** y las **predicciones del modelo** los podemos sacar, como una variable, de este modo:

para ver los residuos del modelo podemos hacer estas dos cosas
```
modelo$residuals
residuals(modelo)
```
para ver las predicciones
```
modelo$fitted.values
```
para ver las predicciones del modelo en la escala de la *link function*
```
modelo$linear.predictors
```

MODELOS GENERALIZADOS

Y finalmente … ¡el resultado de nuestro modelo!

```
summary(mmodelo)
> summary(modelo)
Call:
glm(formula = eqt, family = gaussian(link = "identity"), data = datos,
    offset = log(ntransectos))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-33.626   -7.448   -0.087    7.327   33.713

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 51.138402   2.029443  25.198  < 2e-16 ***
altmed      -0.009964   0.001749  -5.696 1.61e-08 ***
rangoalt    -0.006862   0.001350  -5.084 4.41e-07 ***
shannon      7.069202   0.768480   9.199  < 2e-16 ***
tempmin     -0.297345   0.247064  -1.204    0.229
precip      -0.010729   0.002562  -4.188 3.07e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 117.9467)

    Null deviance: 148802  on 998  degrees of freedom
Residual deviance: 117121  on 993  degrees of freedom
AIC: 7608.5
```

¡¡ QUIETOS !!
antes de ver esto
…

valoración de los residuos del modelo

MODELOS GENERALIZADOS

Exploración de los supuestos canónicos del "buen" modelo

Realmente esto es lo primero que tendríamos que hacer antes de considerar los resultados de nuestro análisis.

Nos permite conocer cómo nuestro modelo se ajusta a los supuestos canónicos del modelo utilizado (*e.g.,* Generalizado Lineal)

Este examen de los supuestos canónicos lo efectuamos teniendo en cuenta los residuos del modelo.

Normalidad
Heterocedasticidad
Puntos influyentes y perdidos
Independencia entre las variables-factores predictores

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

de devianza, haciendo uso de la transformación implícita en la *link function*

NORMALIDAD: los residuos del modelo se deben ajustar a una distribución normal

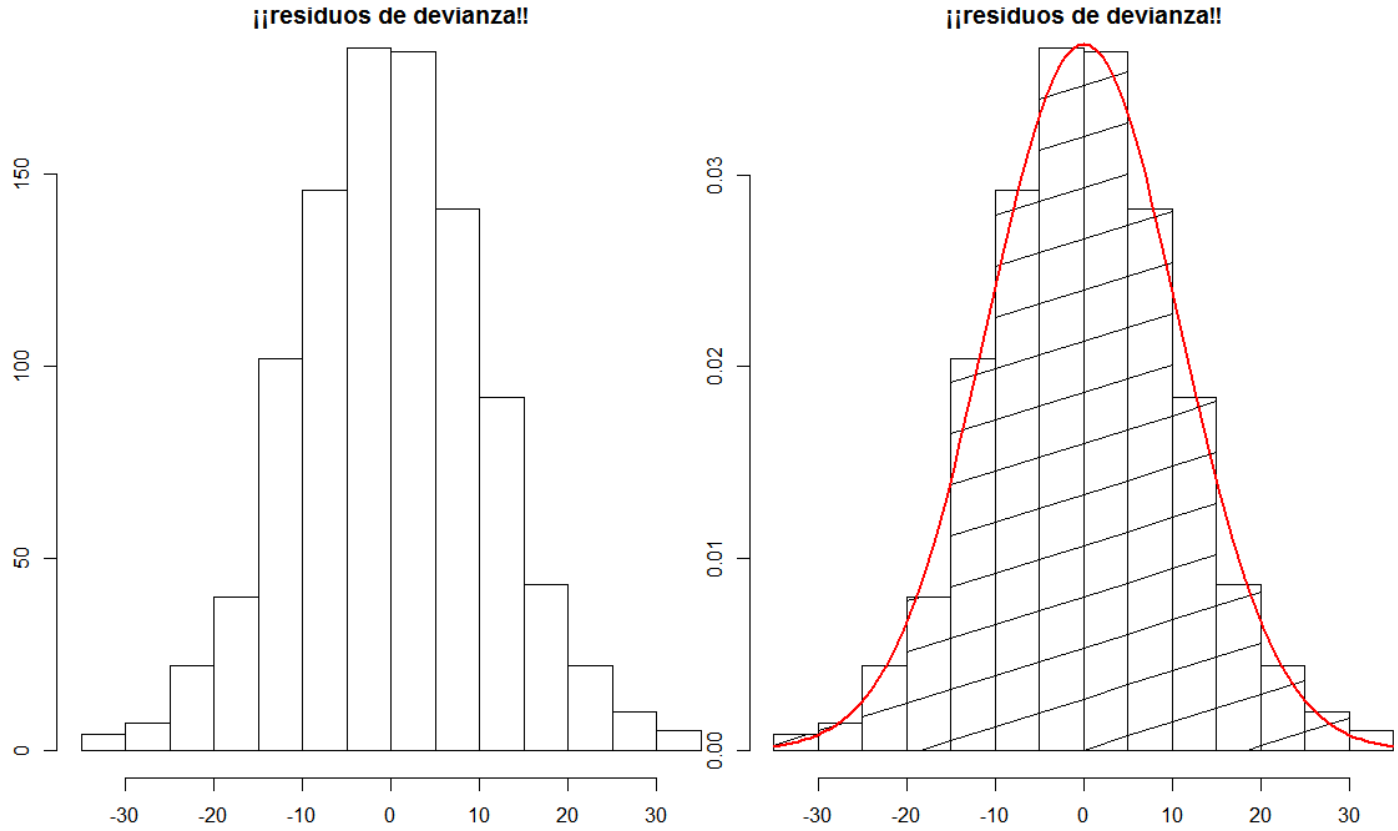
Exploraciones visuales: histograma

```
par(mfcol=c(1,2))
hist(residuals(modelo, type="deviance"), main="¡¡residuos de devianza!!")
hist(residuals(modelo, type="deviance"), density=5, freq=FALSE,
     main="¡¡residuos de devianza!!")
curve(dnorm(x, mean=mean(residuals(modelo, type="deviance")),
      sd=sd(residuals(modelo, type="deviance"))), col="red",
      lwd=2, add=TRUE, yaxt="n")
par(mfcol=c(1,1))
```

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

Exploraciones visuales: histograma



MODELOS GENERALIZADOS

Exploración de los residuos del modelo

Exploraciones visuales: Q-Q plots (normal probability plot)

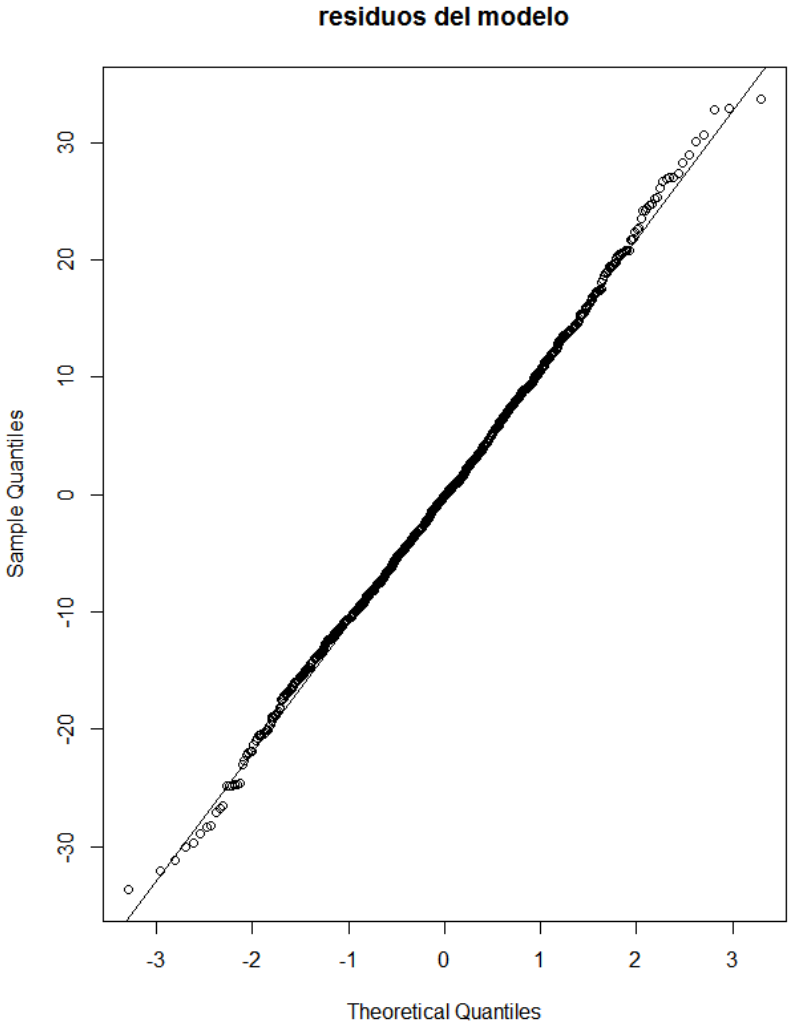
```
par(mfcol=c(1,2))
## el de R por defecto
qqnorm(residuals(modelo, type="deviance"), main="residuos del modelo")
qqline(residuals(modelo, type="deviance"))

## el de otros programas como SPSS, Statistica;    datax=TRUE para cambiar el orden de los ejes
qqnorm(residuals(modelo, type="deviance"), main="residuos del modelo",
       datax=TRUE, xlab="Expected Normal Value", ylab="Residuals")
qqline(residuals(modelo, type="deviance"), datax=TRUE, col="red")
par(mfcol=c(1,1))
```

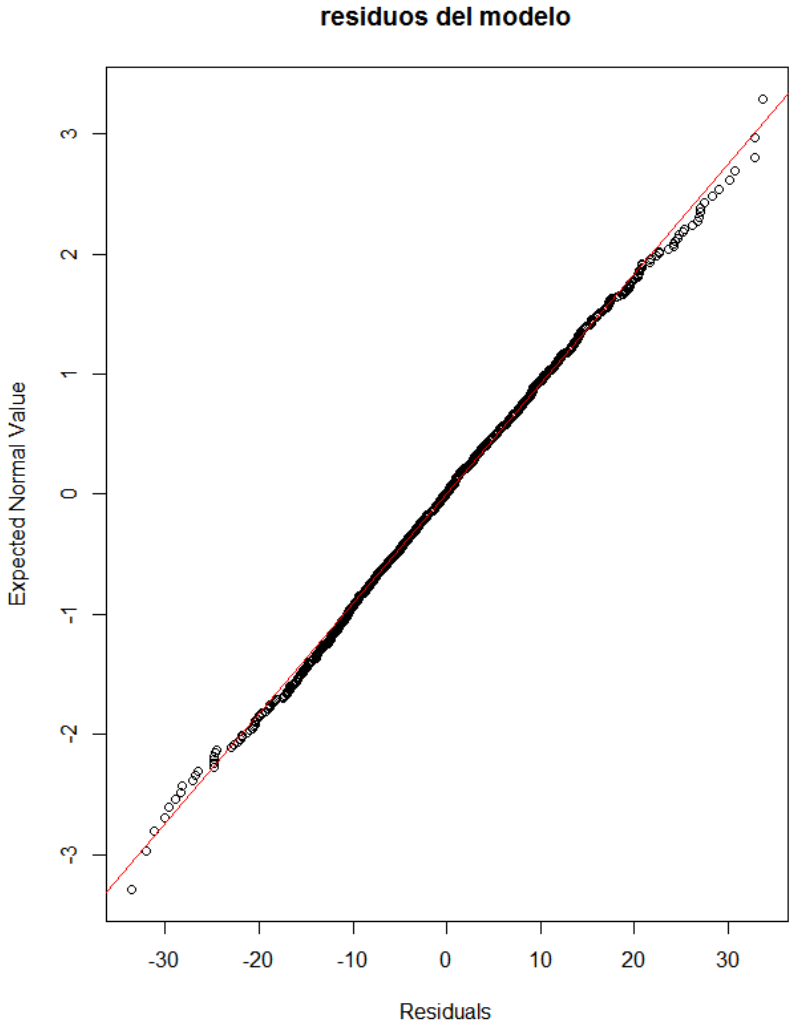
Esta opción es especialmente indicada cuando nuestro tamaño muestral no es grande. Es más atinada "visualmente" que la opción del histograma normal.

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

Exploraciones visuales: **Q-Q plots** (normal probability plot)





MODELOS GENERALIZADOS

Exploración de los residuos del modelo

NORMALIDAD: los residuos de devianza del modelo se deben ajustar a una distribución normal

Globalmente, los tests de la F en modelos Generales y Generalizados son bastante robustos ante las desviaciones de la normalidad de los residuos.

Exploraciones analíticas: test de Shapiro-Wilk

https://en.wikipedia.org/wiki/Shapiro%E2%80%93Wilk_test

En vez del test de Kolmogorov-Smirnov (que asume que muestra y población son coincidentes)
(en la mayoría de los casos tenemos una muestra que no coincide con la población)
https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test

```
shapiro.test(residuals(modelo, type="deviance"))
> shapiro.test(residuals(modelo, type="deviance"))

        Shapiro-Wilk normality test

data:  residuals(modelo, type = "deviance")
W = 0.9983, p-value = 0.4427
```

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

NORMALIDAD: los residuos del modelo se deben ajustar a una distribución normal

Exploraciones analíticas: sesgo y kurtosis

KURTOSIS

puntiagudez de la distribución (Ho= 3 ó 0)
mayor efecto
si K>0 (leptokurtosis) → mayor error tipo II
aceptar la Ho [nula] cuando de hecho es falsa
si K<0 (platikurtosis) → mayor error tipo I

SESGO

simetría de la distribución (Ho= 0)
poco efecto
leve aumento del error de tipo I
rechazar la Ho [nula] cuando es cierta

```
library(moments)
```

```
## kurtosis de Pearson, Ho = 3
kurtosis(residuals(modelo, type="deviance"))
anscombe.test(residuals(modelo, type="deviance"))

> kurtosis(residuals(modelo, type="deviance"))
[1] 3.144636
> anscombe.test(residuals(modelo, type="deviance"))
          Anscombe-Glynn kurtosis test
data:  residuals(modelo, type = "deviance")
kurt = 3.1446, z = 0.9953, p-value = 0.3196
alternative hypothesis: kurtosis is not equal to 3
```

```
## sesgo Ho = 0
skewness(residuals(modelo, type="deviance"))
agostino.test(residuals(modelo, type="deviance"))

> skewness(residuals(modelo, type="deviance"))
[1] 0.06996507
> agostino.test(residuals(modelo, type="deviance"))
          D'Agostino skewness test
data:  residuals(modelo, type = "deviance")
skew = 0.0700, z = 0.9084, p-value = 0.3637
alternative hypothesis: data have a skewness
```

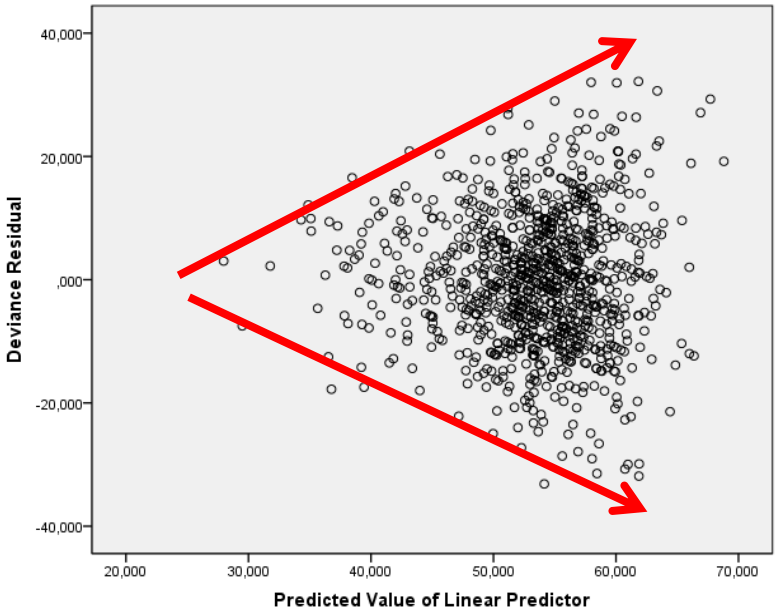
MODELOS GENERALIZADOS

Exploración de los residuos del modelo

HOMOCEDASTICIDAD DE LOS RESIDUOS.

* La varianza de los residuos (de devianza) debe ser similar a lo largo de las predicciones del modelo.
* Debería aparecer un patrón de dispersión aleatoria de puntos, sin dibujar ningún esquema geométrico (e.g., como muchas bolas repartidas al azar en una mesa de billar).

Situación que no deberíamos tener: violación del supuesto de la **homocedasticidad** al haber un patrón triangular indicativo de que hay heterogeneidad en la varianza de los residuos a lo largo de las predicciones del modelo. Hay mayor varianza de los residuos a mayores valores predichos.



¡No podemos asumir las estimas de unos errores estándar (**se**) generalizables!

Tenemos que re-estimar los **se** utilizando procedimientos adecuados y robustos.

Esto modificará la significación de los efectos de las predictoras (las **p**).

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

HOMOCEDASTICIDAD DE LOS RESIDUOS.

Consecuencias de la violación del supuesto de homocedasticidad.

Globalmente, los tests de la F en modelos Generales y Generalizados son bastante robustos ante las desviaciones de la homocedasticidad.

Incluso bajo severas violaciones de este supuesto la *alpha* se modifica poco, tendiendo a incrementarse la probabilidad de cometer el error de tipo I.

Si no se cumple el requisito de homocedasticidad podemos **transformar la respuesta**.

El caso más problemático es aquel en el que la **varianza de los residuos** (diferencia entre valores observados y predichos) se asocia con la **media de las predicciones**.

* si la *relación* es positiva, aumenta el error de tipo I
* si la *relación* es negativa, aumenta el error de tipo II

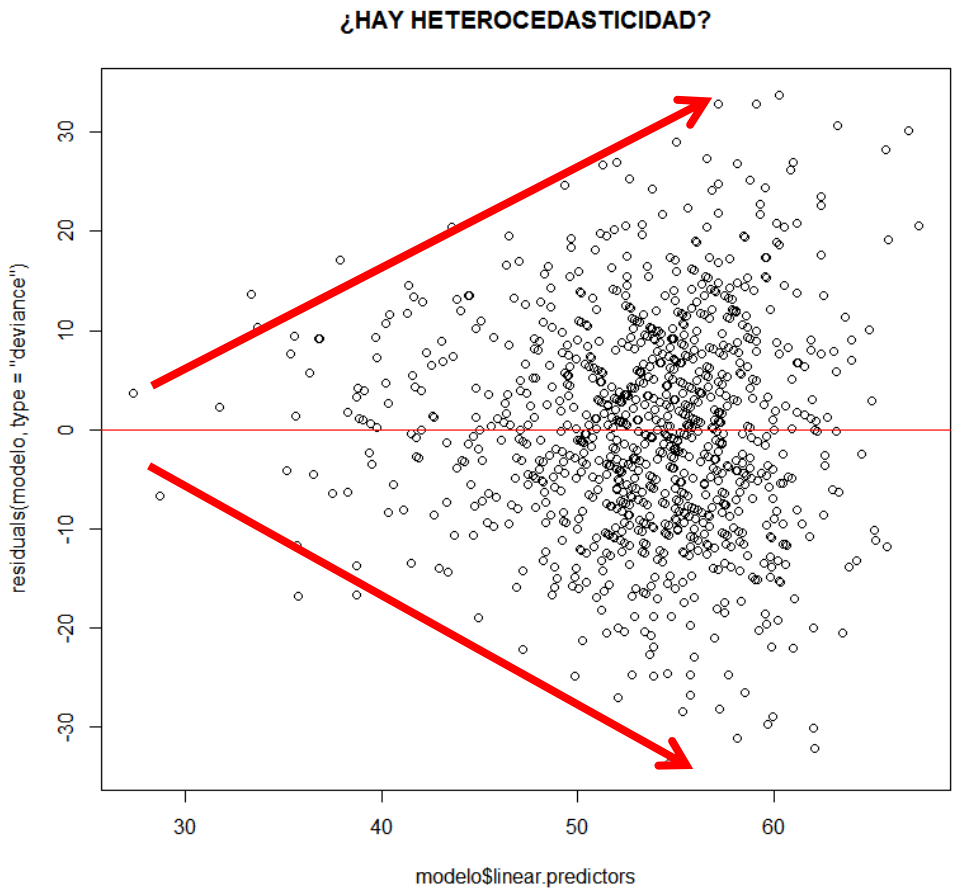
Si hay **heterocedasticidad** en los residuos del modelo, lo volvemos a rehacer utilizando opciones ***robustas*** (*also called the Huber/White/sandwich estimator … a "corrected" model-based estimator that provides a consistent estimate of the covariance*), que utilizan distintas opciones de matrices de varianzas covarianzas (HC0, HC1, HC2, HC3, HC4, HC4m).

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

HOMOCEDASTICIDAD DE LOS RESIDUOS.

```
plot(modelo$linear.predictors, residuals(modelo, type="deviance"),
     main="¿HAY HETEROCEDASTICIDAD?")
abline(h=0, col="red")              ## traza una línea horizontal (h) por el Y=0
```



MODELOS GENERALIZADOS

Exploración de los residuos del modelo

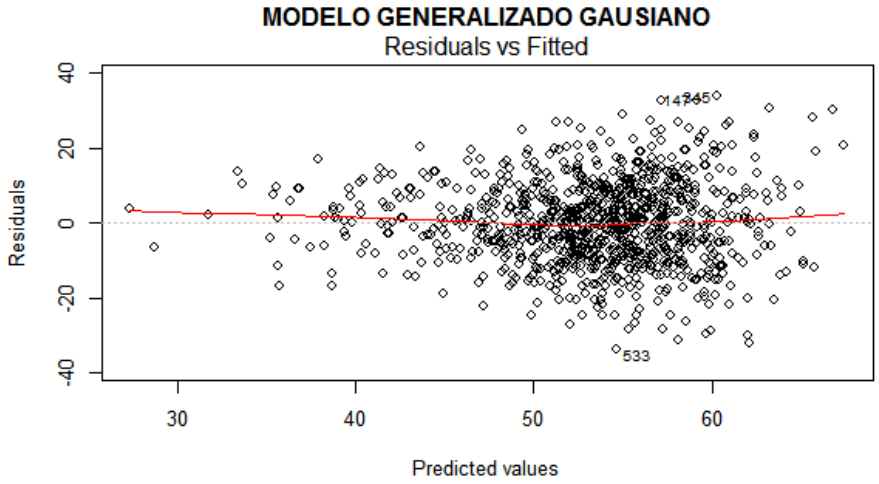
… y finalmente exploración gráfica de una sola vez

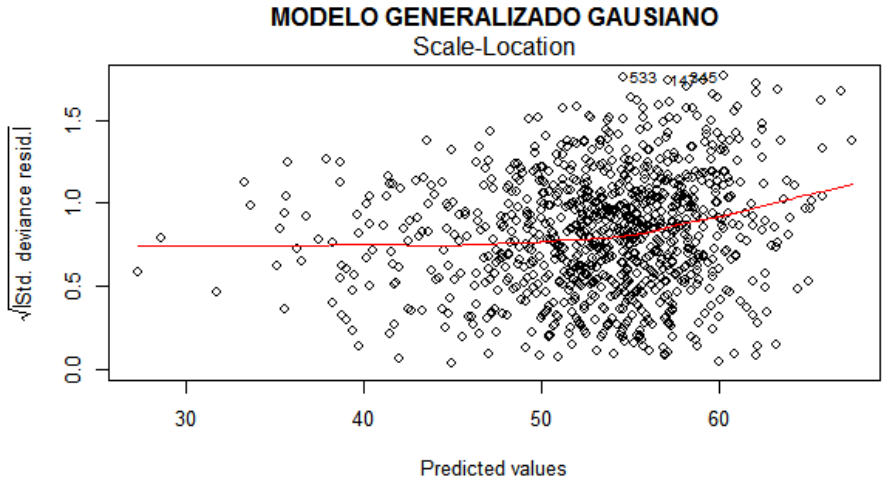
```
## lo mismo de antes pero de una sola vez y ... más
par(mfcol=c(1,1))    ## fija un sólo panel gráfico
par(mfcol=c(2,2))    ## fija cuatro paneles según 2 columnas y 2 filas
plot(modelo, main="MODELO GENERALIZADO GAUSIANO")
par(mfcol=c(1,1))    ## volvemos al modo gráfico de un solo panel

## y otros paneles
par(mfcol=c(1,1))    ## fija un sólo panel gráfico
par(mfcol=c(2,2))    ## fija cuatro paneles según 2 columnas y 2 filas
plot(modelo, c(1:2,4,6), main="MODELO GENERALIZADO GAUSIANO")
par(mfcol=c(1,1))    ## volvemos al modo gráfico de un solo panel
```

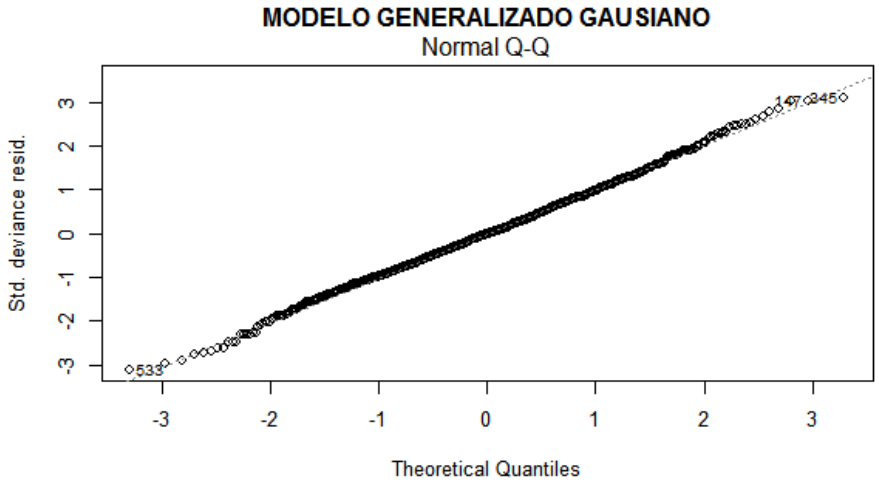
MODELOS GENERALIZADOS

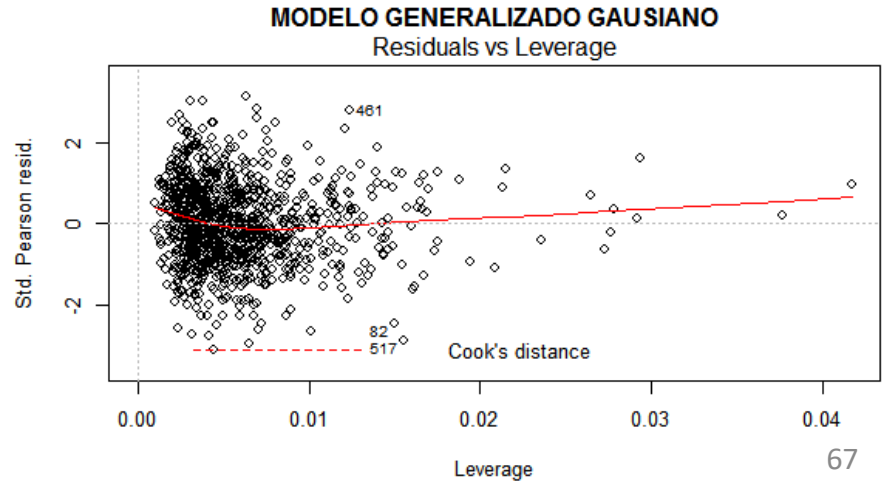
Exploración de los residuos del modelo

… y finalmente exploración gráfica de una sola vez







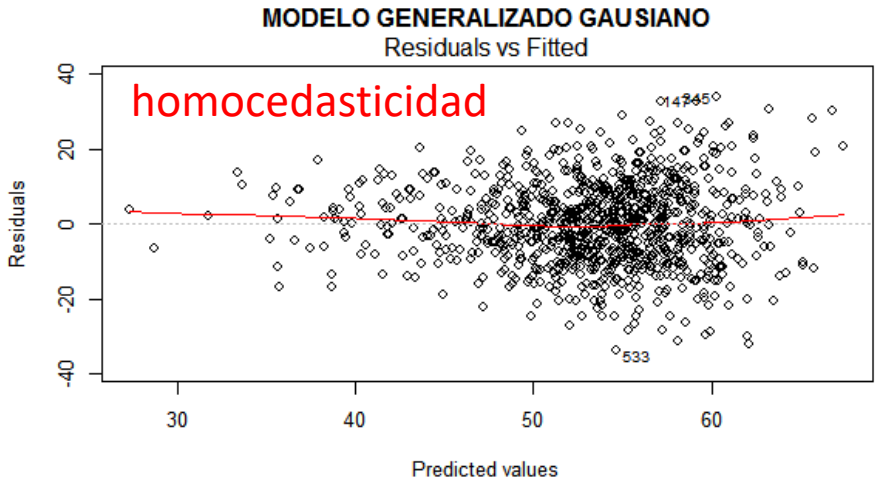


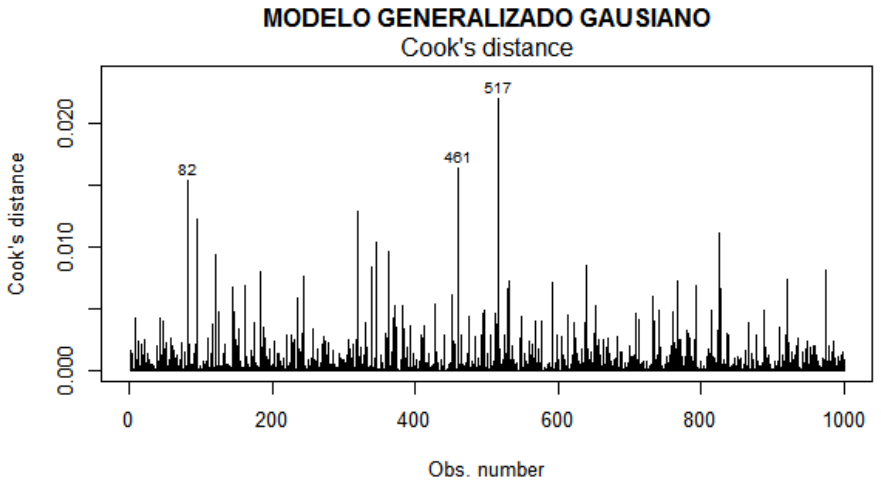
MODELOS GENERALIZADOS

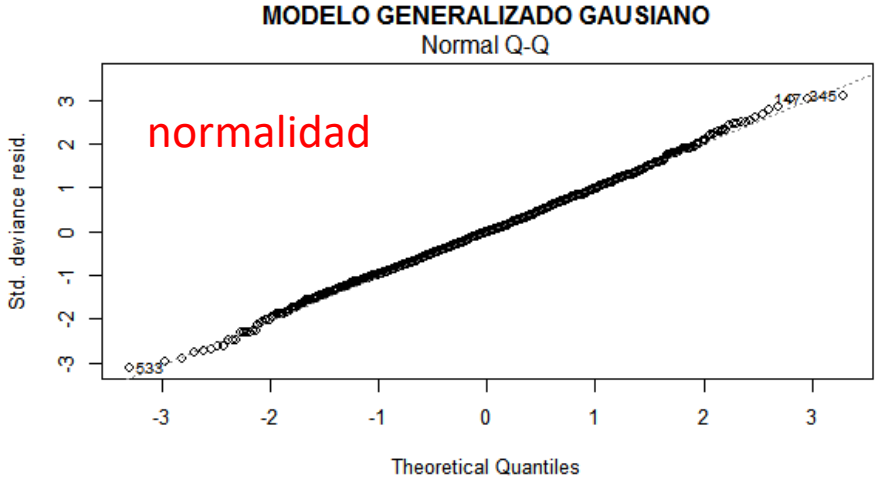
Exploración de los residuos del modelo

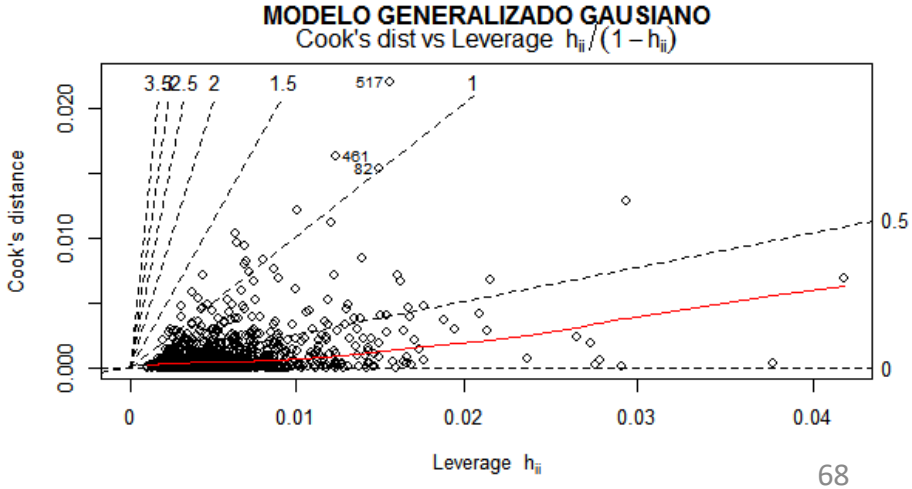
… y finalmente exploración gráfica de una sola vez

puntos influyentes y perdidos









MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

Exploración de **datos residuales de manera individualizada** por cada unidad muestral

Para detectar **puntos influyentes y perdidos** representaremos:

el ***Leverage*** (e.g., eje X) frente a la distancia de Cook (***CooksDistance***; e.g., eje Y).

Leverage: https://en.wikipedia.org/wiki/Leverage_(statistics)

CooksDistance : https://en.wikipedia.org/wiki/Cook%27s_distance

Valores críticos "aproximados":

Distancia de Cook:

posible problema si > **4/n**

problema enorme si >1

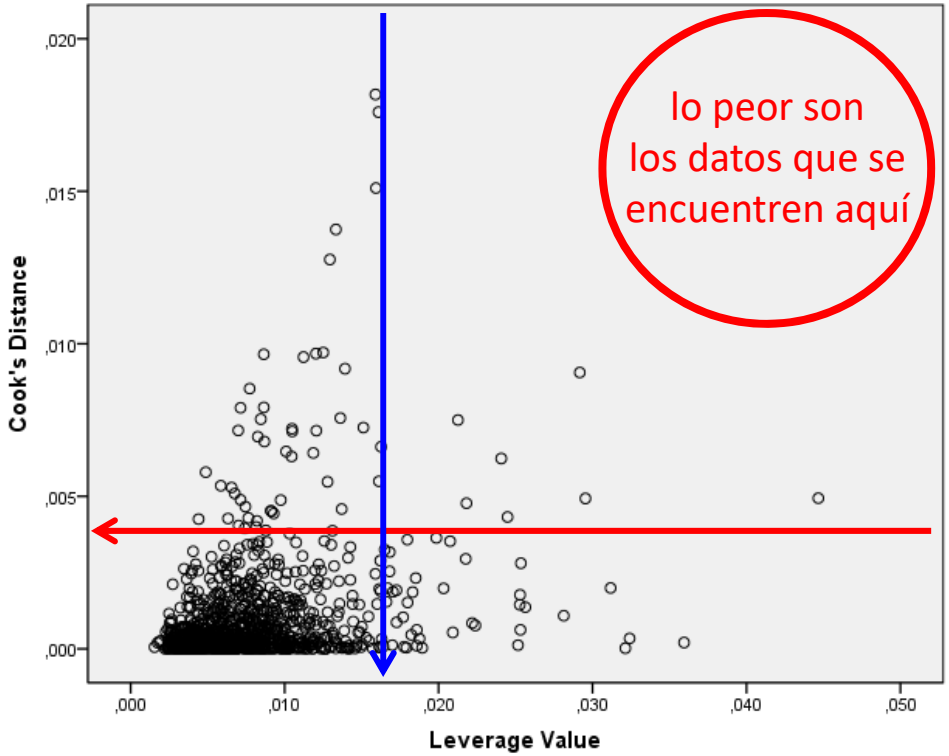
Leverage:

posible problema si > **2*g.l./n**

siendo:

g.l. los grados de libertad del modelo

n el número de casos.



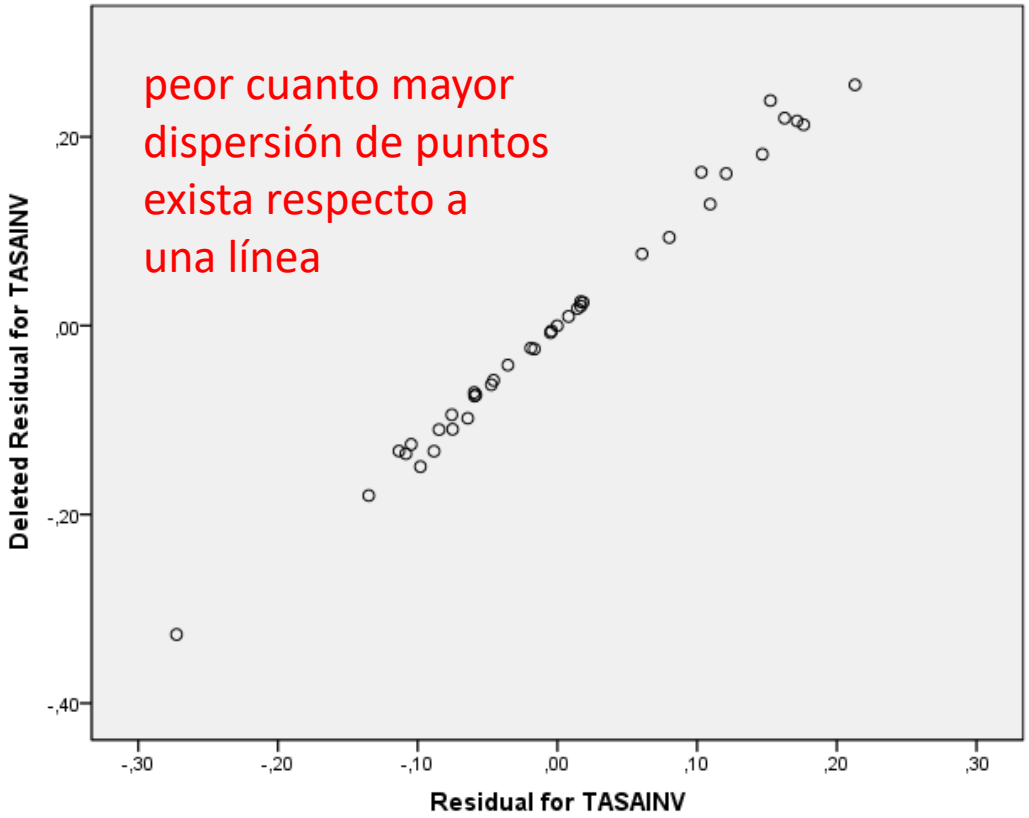
MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

Deleted Residual (qué residuo tendría un dato si no se incluye en el modelo y se predice su valor, restándole su valor realmente observado) **DFFITS** (**https://en.wikipedia.org/wiki/DFFITS**)

Es de gran utilidad representar los Residuos frente a los ***Deleted Residual***



MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

leverage

```
## niveles críticos: 2*(g.l. del modelo)/(Número de datos)
leverage.modelo <- hat(model.matrix(lm(formula(modelo), modelo$model)))
## otra forma más sencilla y general
leverage.modelo <- hatvalues(modelo)
abline(h=2*(length(modelo$residuals)-modelo$df.residual-1)/length(modelo$residuals), col="red")
```

distancia de cook

```
## menor que 4/(número de datos)
plot(cooks.distance(modelo))
abline(h=1, col="red")
abline(h=4/length(modelo$residuals), col="red")
```

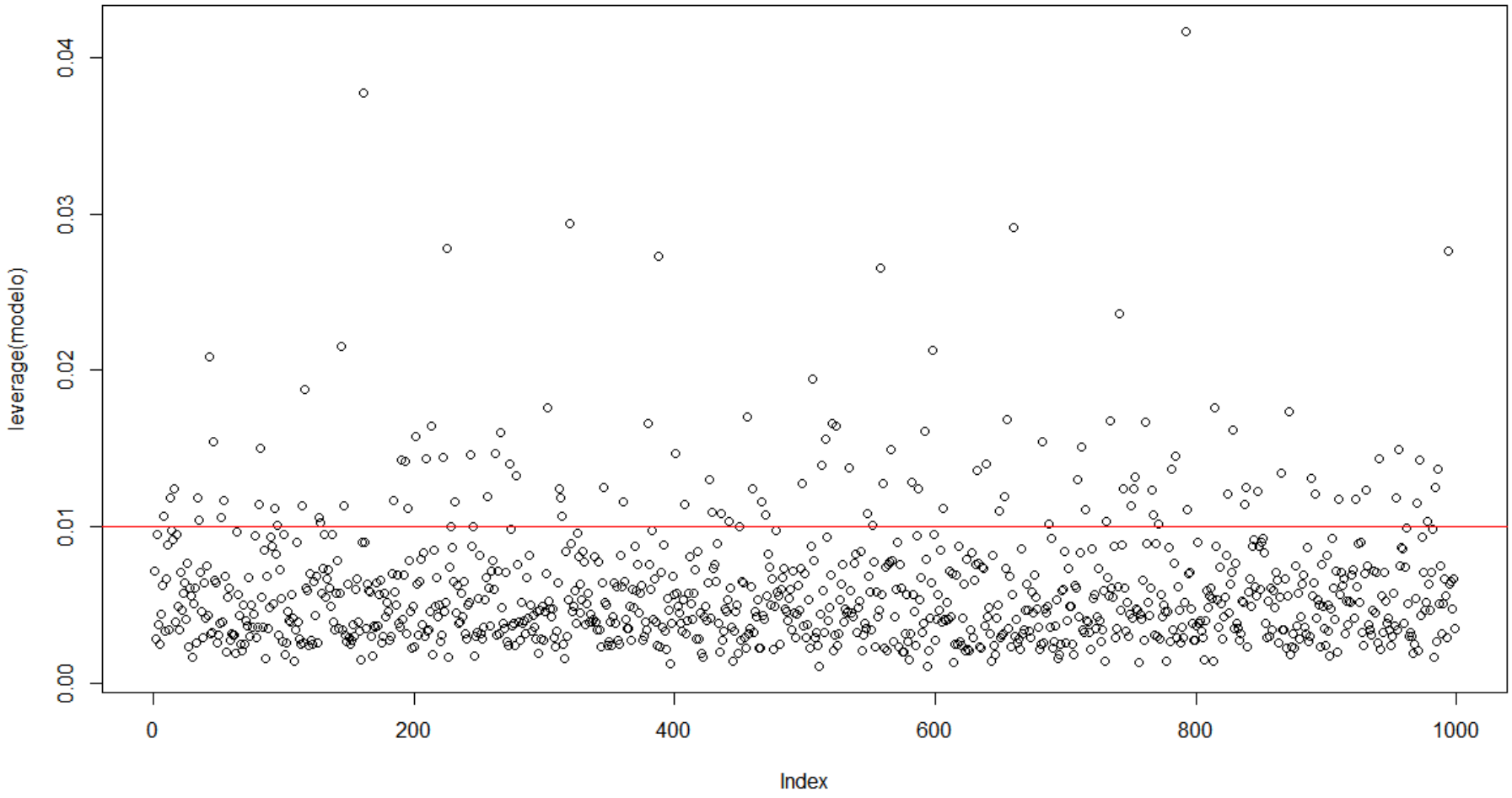
dffits

```
## niveles críticos: 2*raiz((g.l. del modelo)/(Número de datos))
plot(dffits(modelo))
abline(h=2*((length(modelo$residuals)-modelo$df.residual-1)/length(modelo$residuals))^0.5, col="red")
abline(h=-2*((length(modelo$residuals)-modelo$df.residual-1)/length(modelo$residuals))^0.5, col="red")
```

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

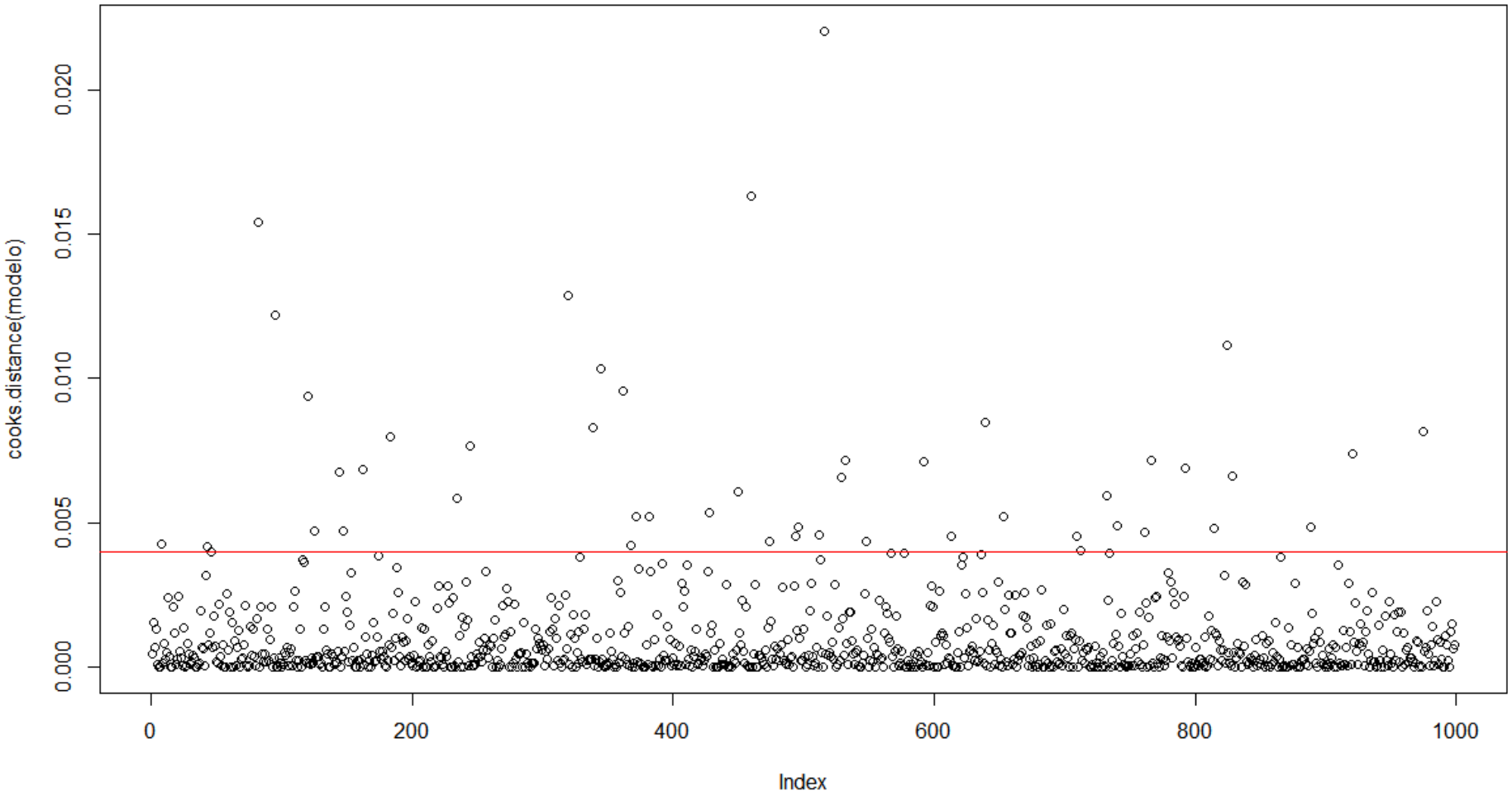


indica el orden del caso-observación en la matriz de datos

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

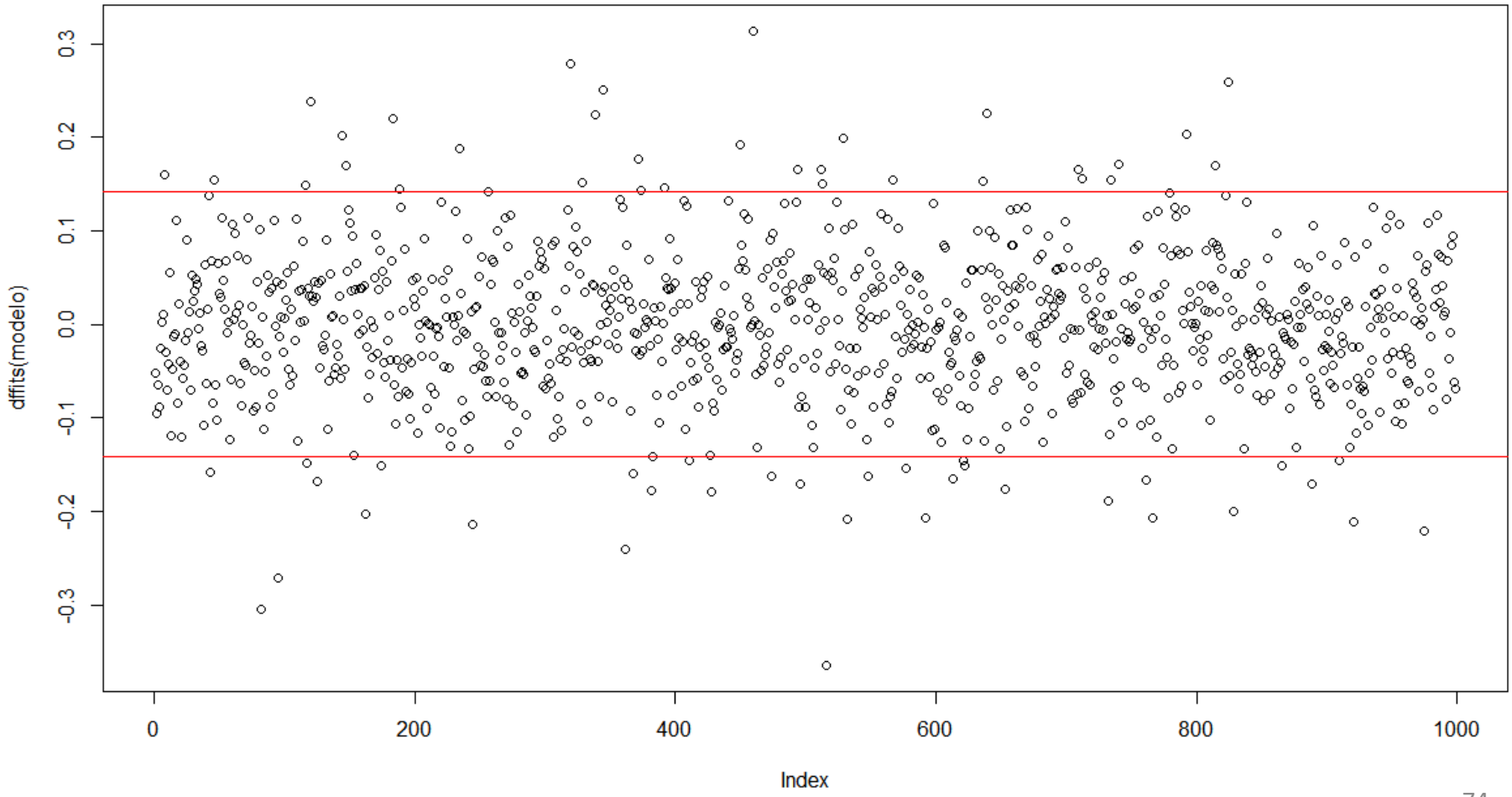
PUNTOS INFLUYENTES Y PUNTOS PERDIDOS



MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

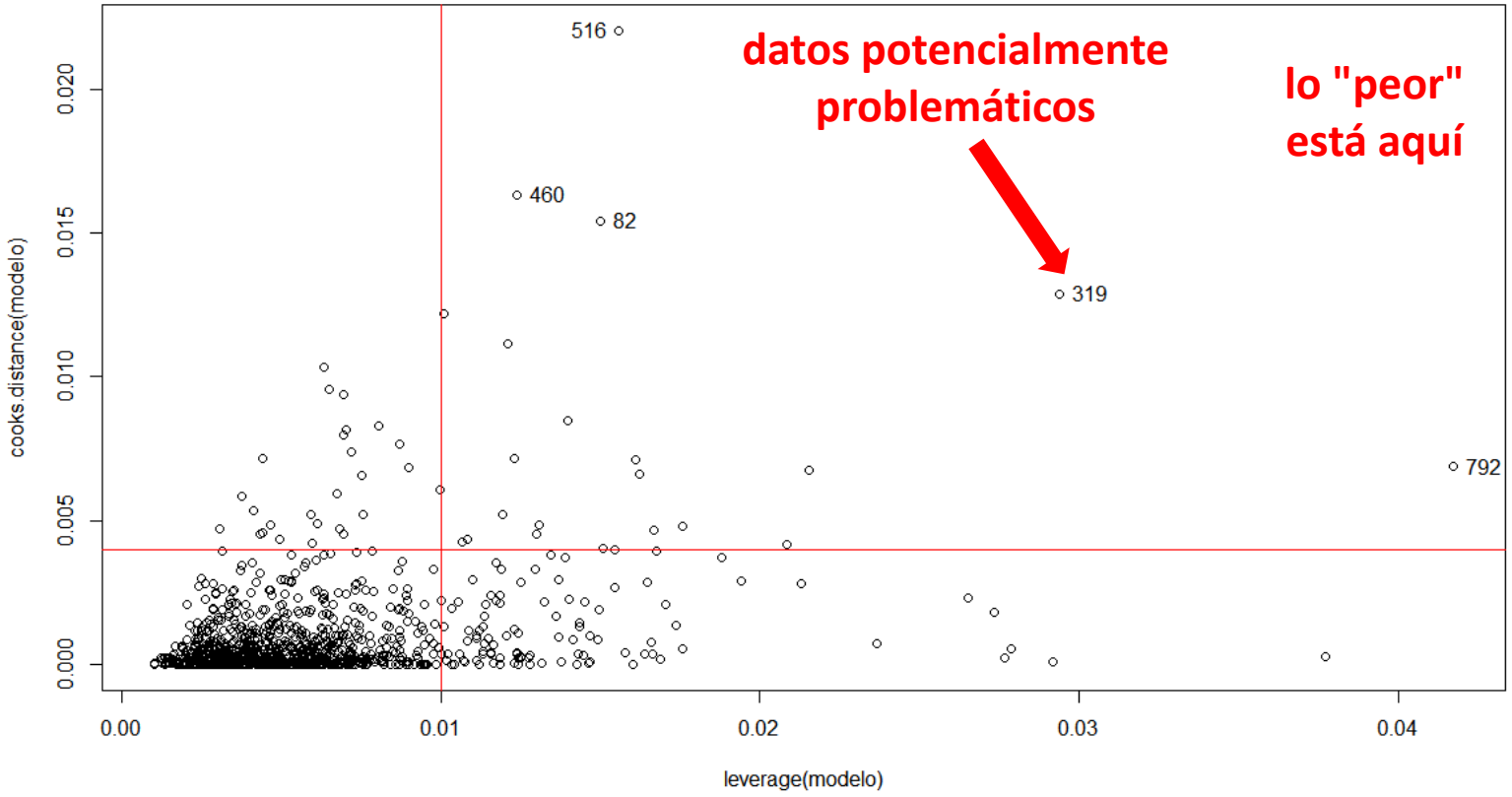


MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

```
plot(cooks.distance(modelo) ~ leverage.modelo, col="darkgreen")
abline(h=4/length(modelo$residuals), col="red")
abline(v=2*(length(modelo$residuals)-modelo$df.residual-1)/length(modelo$residuals), col="red")
identify(cooks.distance(modelo) ~ leverage.modelo)
## terminar dando clik en Finish (boton sup-decho panel de plots)
```



MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

cambio de los coeficientes de regresión (betas, pendientes) según dejemos-quitemos datos

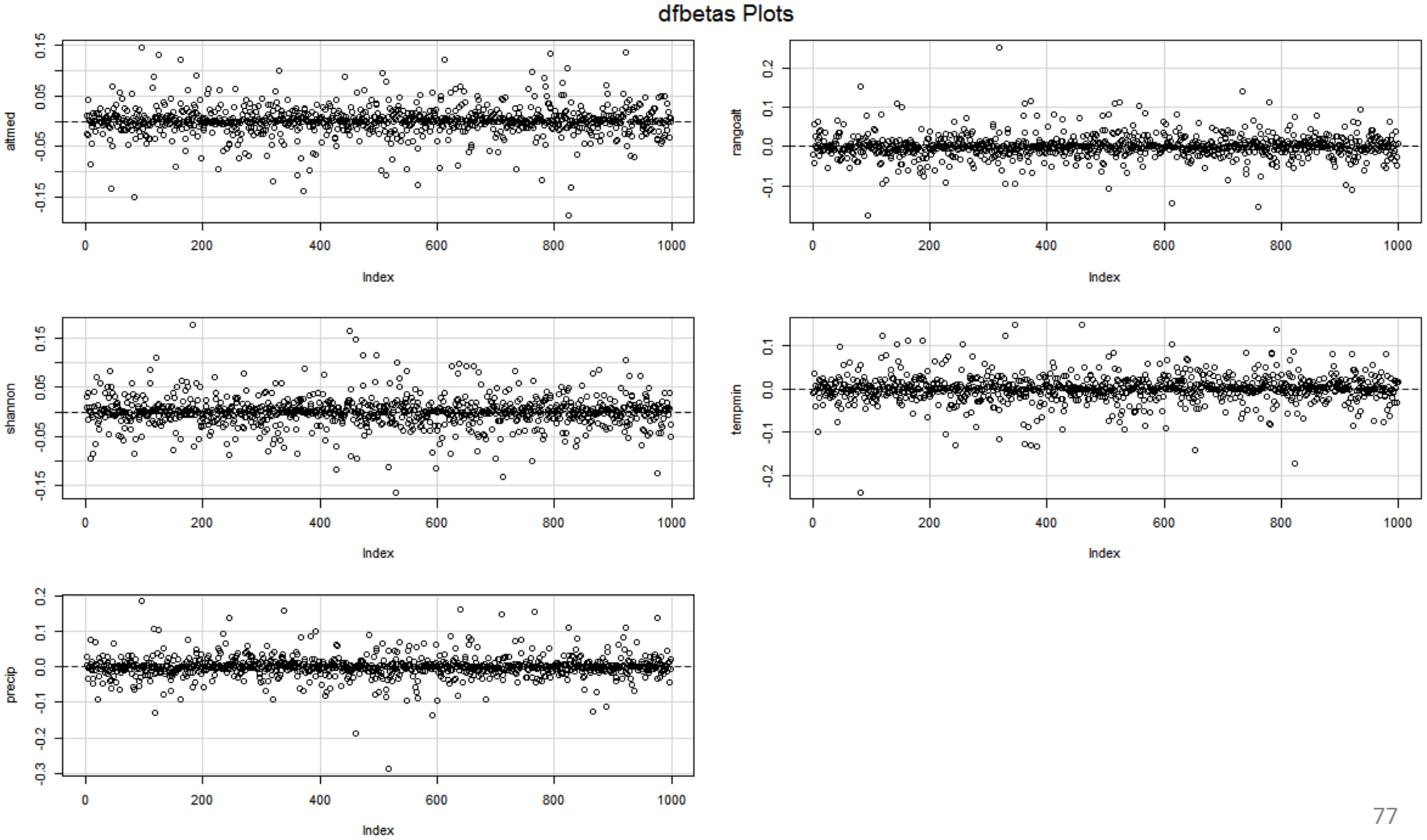
```
dfbetasPlots(modelo)

## convertimos en una matriz de datos los valores de dfbetas del modelo
modelo.dfbetas <- as.data.frame(dfbetas(modelo))
names(modelo.dfbetas)
## para exploración de los datos particulares en una predictora concreta
plot(modelo.dfbetas$shannon)
identify(modelo.dfbetas$shannon)
## terminar dando clik en Finish
```

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

cambio de los coeficientes de regresión (betas, pendientes) según dejemos-quitemos datos

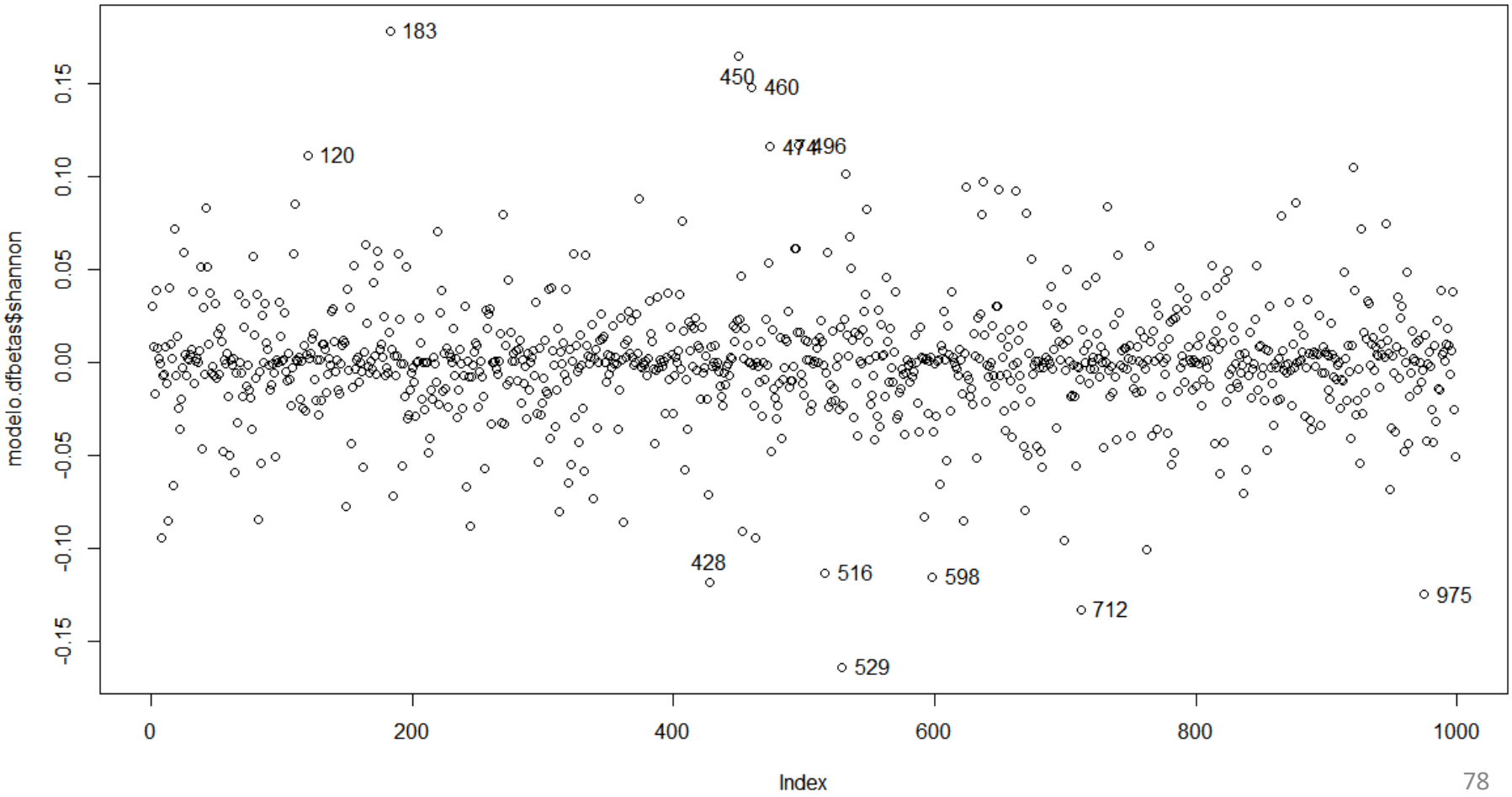


MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

cambio de los coeficientes de regresión (betas, pendientes) según dejemos-quitemos datos



MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS

Los residuos studentizados nos pueden dar una indicación de qué datos es probable que no pertenezcan a la población.

El valor crítico lo define la **t de Student** teniendo en cuenta los g.l. (error) del modelo.

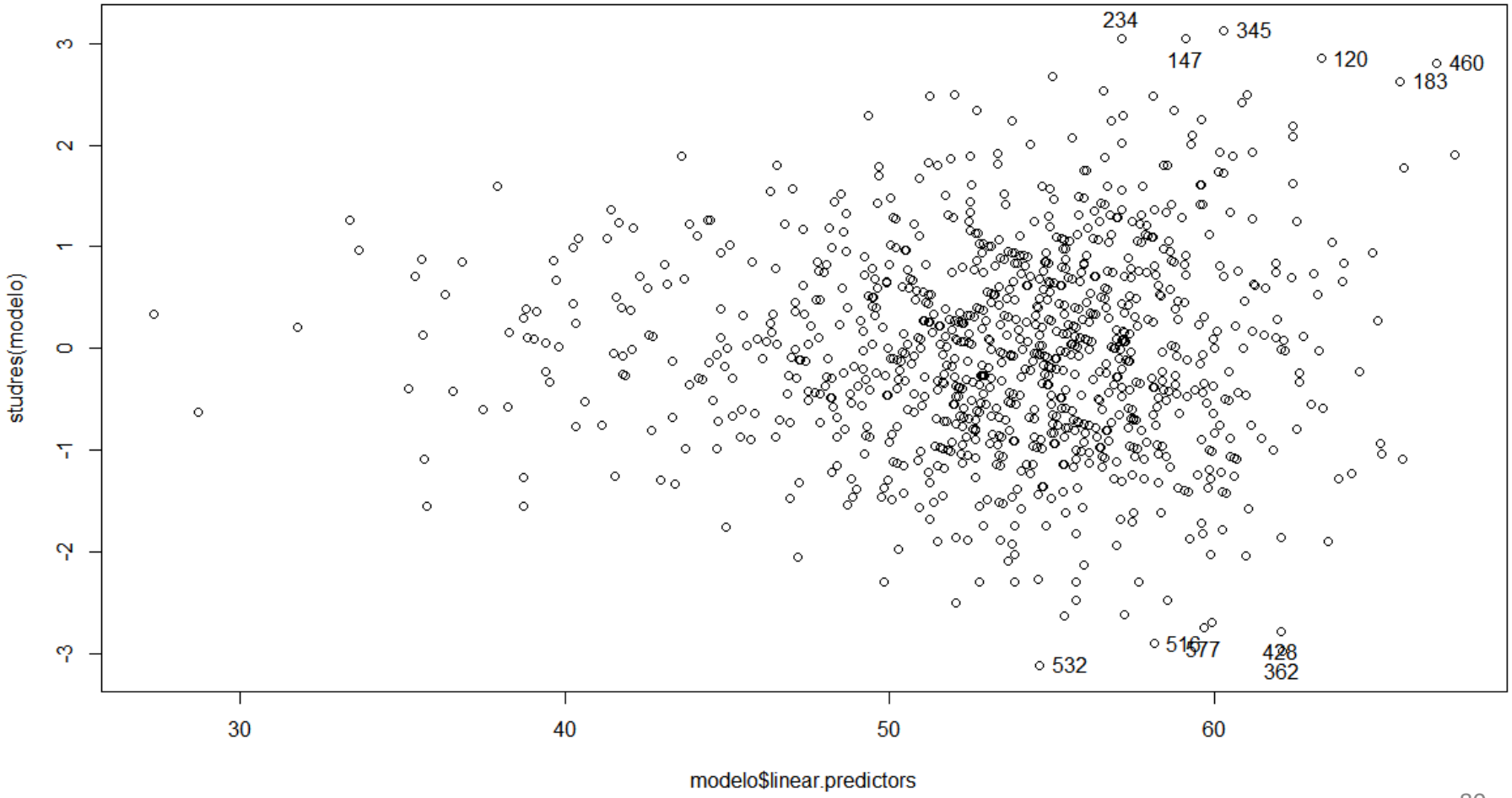
Valores mayores o menores de ca. 2.0 (alfa=0.05) o 2.7 (alfa=0.01) son **peligrosos**

```
## studres es del paquete MASS
library(MASS)
plot(studres(modelo) ~ modelo$fitted.values)
identify(studres(modelo) ~ modelo$fitted.values)
## terminar dando clik en Finish (boton sup-decho panel de plots)
```

MODELOS GENERALIZADOS

Exploración de los residuos del modelo

PUNTOS INFLUYENTES Y PUNTOS PERDIDOS



MODELOS GENERALIZADOS

relaciones entre las variables predictoras

MODELOS GENERALIZADOS

Exploración de la colinealidad entre las variables predictoras

Las variables predictoras durante mucho tiempo fueron llamadas "independientes".
Era el reconocimiento explícito de que debían no estar correlacionadas entre si.

Si hay dependencia entre las variables predictoras, se dice que entre ellas existe **colinealidad**.

La colinealidad puede surgir porque:
- por definición las predictoras están correlacionadas (e.g., temperatura y altitud)
- porque no hay homogeneidad de tamaños muestrales en los diferentes niveles de los factores

El hecho de que las variables predictoras no sean indenpendientes (*i.e.*, ortogonales) conlleva algunos **problemas**:
- las variables se anulan entre si
- las estimas de significación se alteran
- las magnitudes de efecto se ven modificadas
- no hay convergencia entre los resultados de SS (suma de cuadrados o equiv.) de tipo I y tipo III

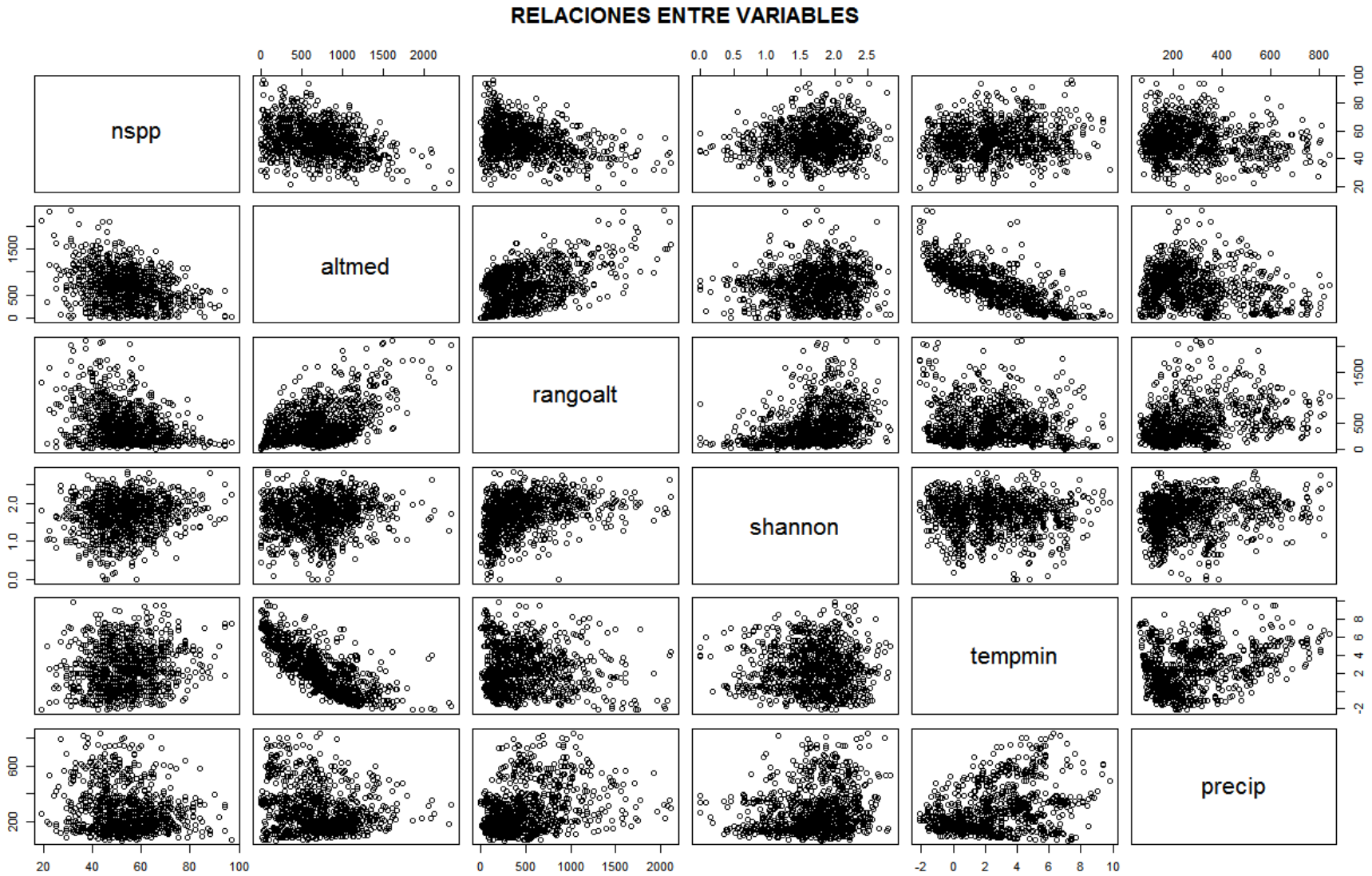
Este asunto lo vamos a abordar mediante exploraciones visuales y cuantitativas de la intensidad de relación entre las variables predictoras

MODELOS GENERALIZADOS

```
## VISUALIZACIÓN DE LAS RELACIONES ENTRE TODAS LAS VARIABLES
##   del modo: ~ respuesta+predictoras
relaciones <- ~ nspp+altmed+rangoalt+shannon+tempmin+precip

## modelos visuales de asociación entre variables
##  en log= ponemos qué ejes queremos transformar logarítmicamente
pairs(relaciones, data=datos, cex.labels=2, log="",
      main="RELACIONES ENTRE VARIABLES")

pairs(relaciones, data=datos, cex.labels=2, log="xy",
      main="RELACIONES ENTRE VARIABLES EN LOGARITMO")

## diagonal es: "density", "boxplot", "histogram", "oned", "qqplot", "none"
## reg.line puede ser lm o rlm (lineal o robusta lineal) o False (F)
scatterplotMatrix(relaciones, data=datos, pch=19, cex=.75, cex.lab=3,
    cex.axis=1.5, diagonal="density", reg.line=F, smoother=loessLine,
    spread=T, ellipse=T, levels=c(0.666, 0.95),
    col=c('green','#2957FF','#FF8000'), col.axis='gray10')
scatterplotMatrix(relaciones, data=datos, pch=19, cex=.75, cex.lab=3,
    cex.axis=1.5, diagonal="density", reg.line=lm, lwd=3,
    smoother=loessLine, spread=T, ellipse=F, levels=c(0.666, 0.95),
    col=c('green','blue','orange'), col.axis='gray10')
```

MODELOS GENERALIZADOS

VISUALIZACIÓN DE LAS RELACIONES ENTRE TODAS LAS VARIABLES

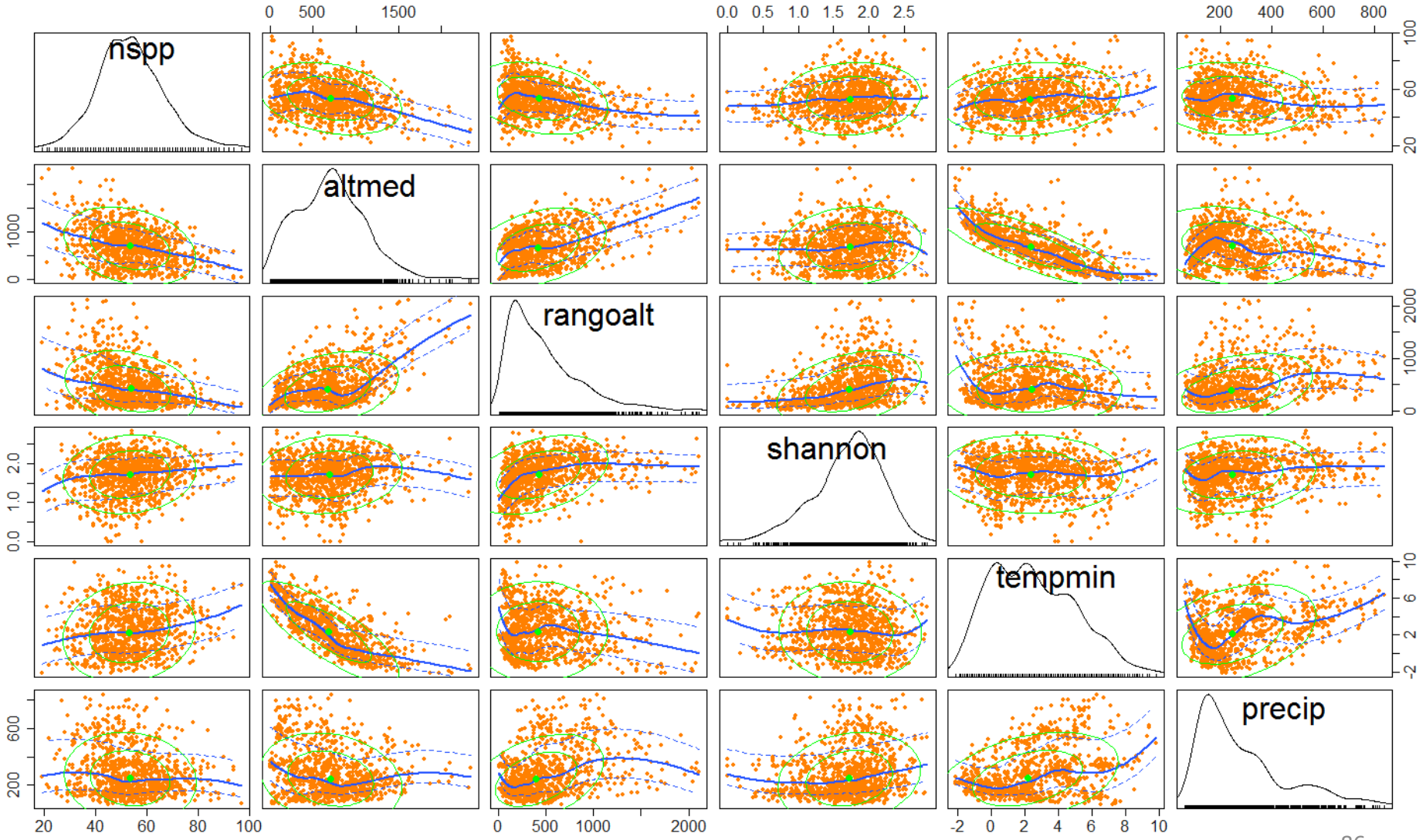


MODELOS GENERALIZADOS

```
## VISUALIZACIÓN DE LAS RELACIONES ENTRE TODAS LAS VARIABLES
##   Otro modo es
relaciones <- ~ nspp+altmed+rangoalt+shannon+tempmin+precip

## diagonal es: "density", "boxplot", "histogram", "oned", "qqplot", "none"
## reg.line puede ser lm o rlm (lineal o robusta lineal)
scatterplotMatrix(relaciones, data=datos, pch=19, cex=.75, cex.lab=3,
    cex.axis=1.5, diagonal="density", reg.line=F, smoother=loessLine,
    spread=T, ellipse=T, levels=c(0.666, 0.95),
    col=c('green','#2957FF','#FF8000'), col.axis='gray10')

scatterplotMatrix(relaciones, data=datos, pch=19, cex=.75, cex.lab=3,
    cex.axis=1.5, diagonal="density", reg.line=lm, lwd=3,
    smoother=loessLine, spread=T, ellipse=F, levels=c(0.666, 0.95),
    col=c('green','blue','orange'), col.axis='gray10')
```

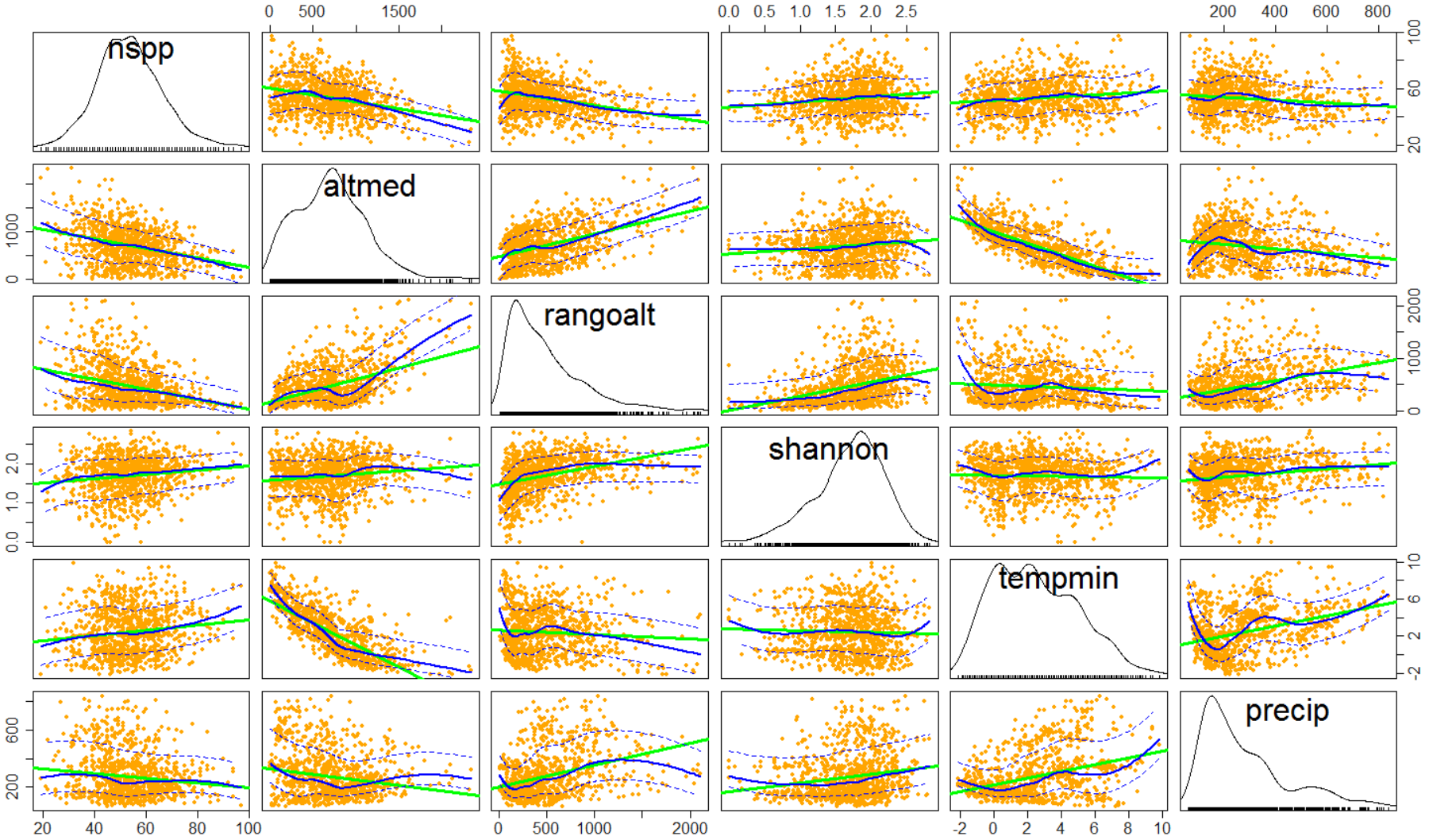
MODELOS GENERALIZADOS

VISUALIZACIÓN DE LAS RELACIONES ENTRE TODAS LAS VARIABLES



MODELOS GENERALIZADOS

VISUALIZACIÓN DE LAS RELACIONES ENTRE TODAS LAS VARIABLES



MODELOS GENERALIZADOS

Exploración de la colinealidad entre las variables predictoras

Esta larga secuencia de líneas de código vale para cualquier situación de análisis.
Sólo tendremos que alterar:

eqt: la cadena de texto que contiene nuestra ecuación usada en el modelo **glm**

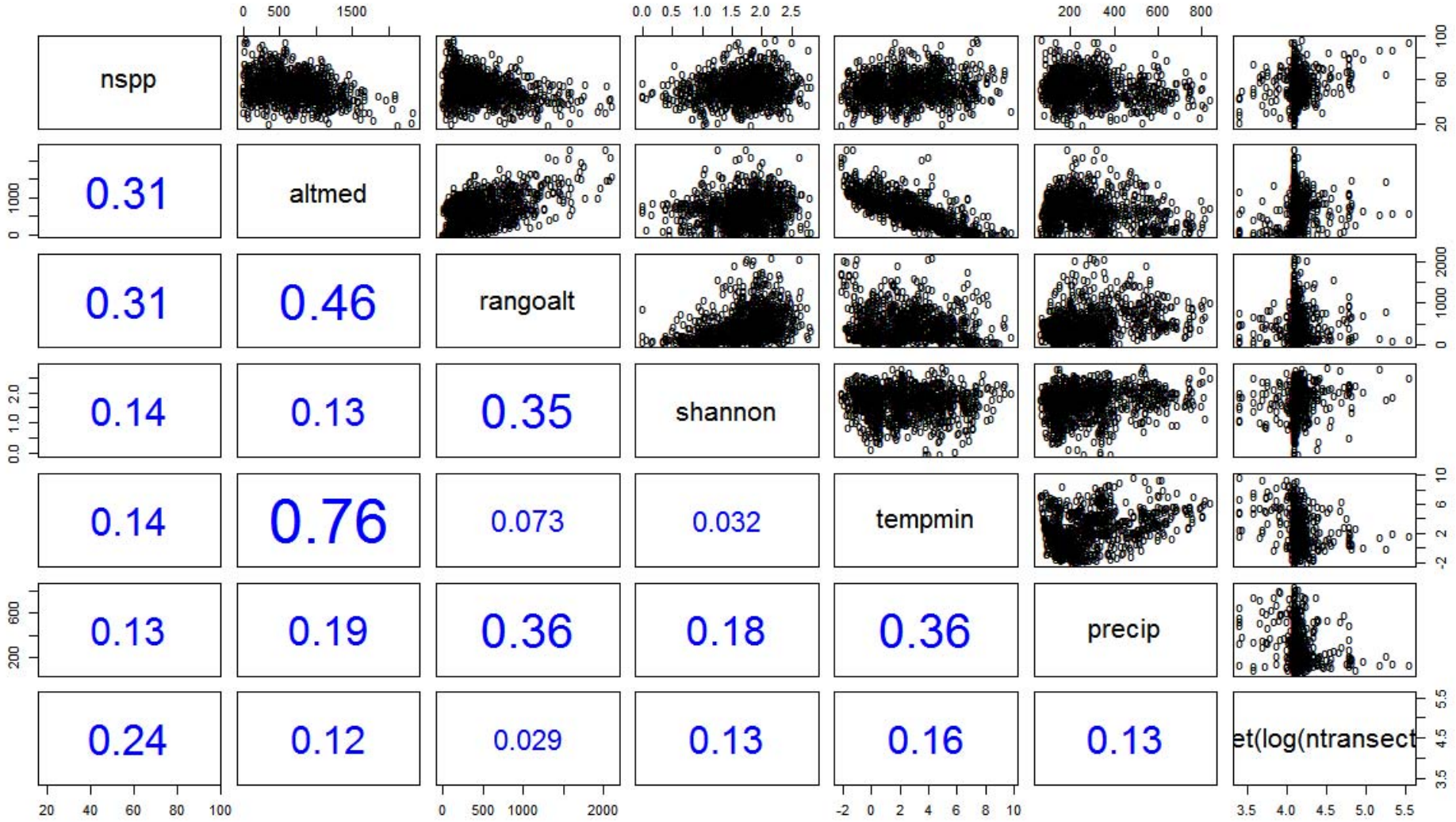
datos: el origen de los datos de análisis

```
## comprobad que no haya "missing cells" en los factores
panel.cor <- function(x, y, digits=2, prefix="", cex.cor, ...) {
    usr <- par("usr")
    on.exit(par(usr))
    par(usr = c(0, 1, 0, 1))
    r <- abs(cor(x, y, use="complete.obs"))
    txt <- format(c(r, 0.123456789), digits=digits)[1]
    txt <- paste(prefix, txt, sep="")
    if(missing(cex.cor)) cex.cor <- 0.8/strwidth(txt)
    text(0.5, 0.5, txt, col="blue", cex = cex.cor * (1 + r) / 1)
}
pairs(eqt, data=datos, cex.labels=2, pch="o", lower.panel = panel.cor)
```

MODELOS GENERALIZADOS

Exploración de la colinealidad entre las variables predictoras

el número azul es el valor absoluto de la correlación entre pares de predictores



MODELOS GENERALIZADOS

Exploración de la colinealidad entre las variables predictoras

Si las variables predictoras no son independientes, existe multicolinealidad.
Este aspecto se puede valorar con el índice **VIF** (*variance inflation factor*)
https://en.wikipedia.org/wiki/Variance_inflation_factor

$$\mathbf{VIF = 1 / (1 - R^2)}$$

donde R^2 se obtiene regresionando cada variable predictora en función de todas las restantes.

$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots + \beta_k X_k$ → este es nuestro modelo de interés
$\mathbf{X_1} = \alpha_2 X_2 + \alpha_3 X_3 + \alpha_4 X_4 + \ldots + \alpha_k X_k$ → este es el modelo para calcular **VIF**

VIF = 1 / (1 – R^2) **Tolerancia** = 1 – R^2

siendo R^2 el coeficiente de determinación de $\mathbf{X_1}$ explicada por las restantes.

Si VIF = 1 entonces cada variable predictora es independiente de las restantes.

La raíz cuadrada del valor VIF es una aproximación a cuántas veces es más grande el error estándar de un coeficiente de regresión respecto a lo que debería ser si no existiese multicolinealidad.

MODELOS GENERALIZADOS

Exploración de la colinealidad entre las variables predictoras

```
library(car)
vif(modelo)
sqrt(vif(modelo))
```

```
> vif(modelo)
 altmed rangoalt  shannon  tempmin   precip
4.013709 2.182928 1.154250 3.123882 1.438275

> sqrt(vif(modelo))
 altmed rangoalt  shannon  tempmin   precip
2.003424 1.477474 1.074360 1.767451 1.199281
```

… si VIF vale más que cuatro … malo

podremos reducir el modelo

o utilizar otros tipos de análisis (e.g., PLS; consultad https://goo.gl/CztkHe)

transformaciones de la respuesta

… para seguir con el modelos *lm* (general lineal)

Intentando salvar el modelo General Lineal con residuos … no "buenos"

Antes de pasar a utilizar otros modelos que no hacen uso de la Normal – Gaussiana (e.g., usando modelos Generalizados) podemos llevar la variable respuesta a una nueva escala de medida.

Esto equivale a **transformar** la variable respuesta.
De hecho, los fenómenos de asociación entre variables no tienen por qué ajustarse a lo lineal.

Es más, por su naturaleza, las variables respuestas no tienen porqué ajustarse ni a una Poisson ni a una binomial ni a una binomial negativa, gamma, etc.

En estos casos, ni siquiera los Modelos Generalizados con otras distribuciones que no sean la normal son adecuados.

Las **transformaciones más clásicas** son:

logarítmica → Y' = log(Y+1) +1 para el caso de que Y contenga "ceros"

raíz cuadrada → Y' = (Y+**a**)^0.5 **a** suele ser 0.5 ó 1 (para el caso de que Y contenga "ceros")

rangos → Y es llevada a una escala ordinal Y' (1 para el valor más pequeño)
equivale a utilizar estadística no paramátrica usando la paramétrica de lm
se pierde, o se puede perder, el significado "métrico" de las interacciones
por tanto, es mejor, para los modelos de efectos principales.

arcoseno → para proporciones **p** acotadas entre 0-y-1: Y' = arcseno(**p**^0.5)
hoy ya está descartada porque podemos contar con modelos logit

Intentando salvar el modelo General Lineal con residuos … no "buenos"

Si estas transformaciones no funcionan, podemos contar con la **transformación de Box-Cox**.
http://onlinestatbook.com/2/transformations/box-cox.html
Para utilizar el modo más sencillo de proceder con modelo generales usando `lm(…)`

Es una transformación potencial:
Y' = (Y^λ - 1)/λ (Y^λ es "Y elevado a λ")

Cuando lambda (λ) es cero, la transformación Box-Cox converge con la logarítmica.

En R podemos contar con procesos automatizados que buscan la **λ** óptima que minimiza los desvíos de los supuestos canónicos de los modelos lm.

Es una transformación muy adecuada para:
- conseguir que los residuos se desvíen menos de la normal
- se reduzca la heterocedasticidad de los residuos
- disminuya la heterogeneidad de la varianza a través de los niveles de los factores

Pero tiene como pega la "interpretación" de los valores de la respuesta, ya que su nueva escala de medida es compleja.

Intentando salvar el modelo General Lineal con residuos … no "buenos"

TRANSFORMACIÓN BOX-COX en R

deberemos añadir +1 a la variable respuesta si tiene ceros

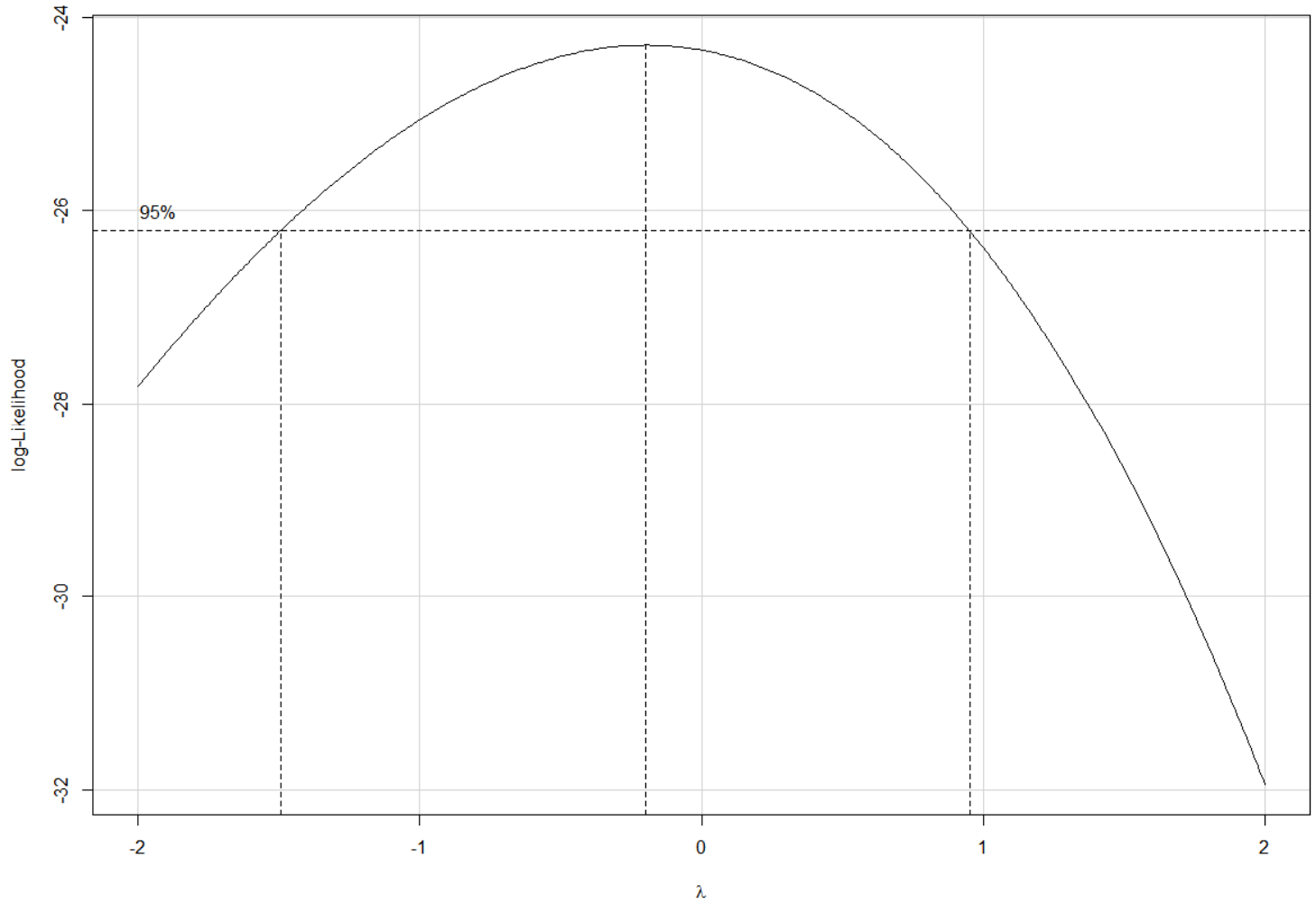
la transformación es: Y' = (Y^lambda - 1)/lambda ,ó,

Y' = ((Y+1)^lambda - 1)/lambda

```
## la siguiente línea de código SOLO TRANSFORMA la respuesta de nuestra ecuación eqt
## podemos afinar y restringir aun más el rango de lambda
## por ejemplo: lambda=seq(0,1, 1/1000))
print(boxCox(modelo, lambda=seq(-2,2, 1/100)))

lambda.bc <- with(boxCox(modelo, lambda=seq(-2,2, 1/1000)), x[which.max(y)])
print(c("parámetro lambda de Box-Cox =",round(lambda.bc, 3)), quote=FALSE)
> print(c("parámetro lambda de Box-Cox =",round(lambda.bc, 3)), quote=FALSE)
[1] parámetro lambda de Box-Cox = -0.196


## transformamos la variable a continuación ... que llamamos respuesta.bc
respuesta.bc <- ((modelo$model[,1]^lambda.bc)-1)/(lambda.bc)
## para explorar la relación entre la respuesta original y su transformación Box-Cox
plot(respuesta.bc, modelo$model[,1], xlab="variable respuesta
transformada", ylab="variable respuesta original")
```

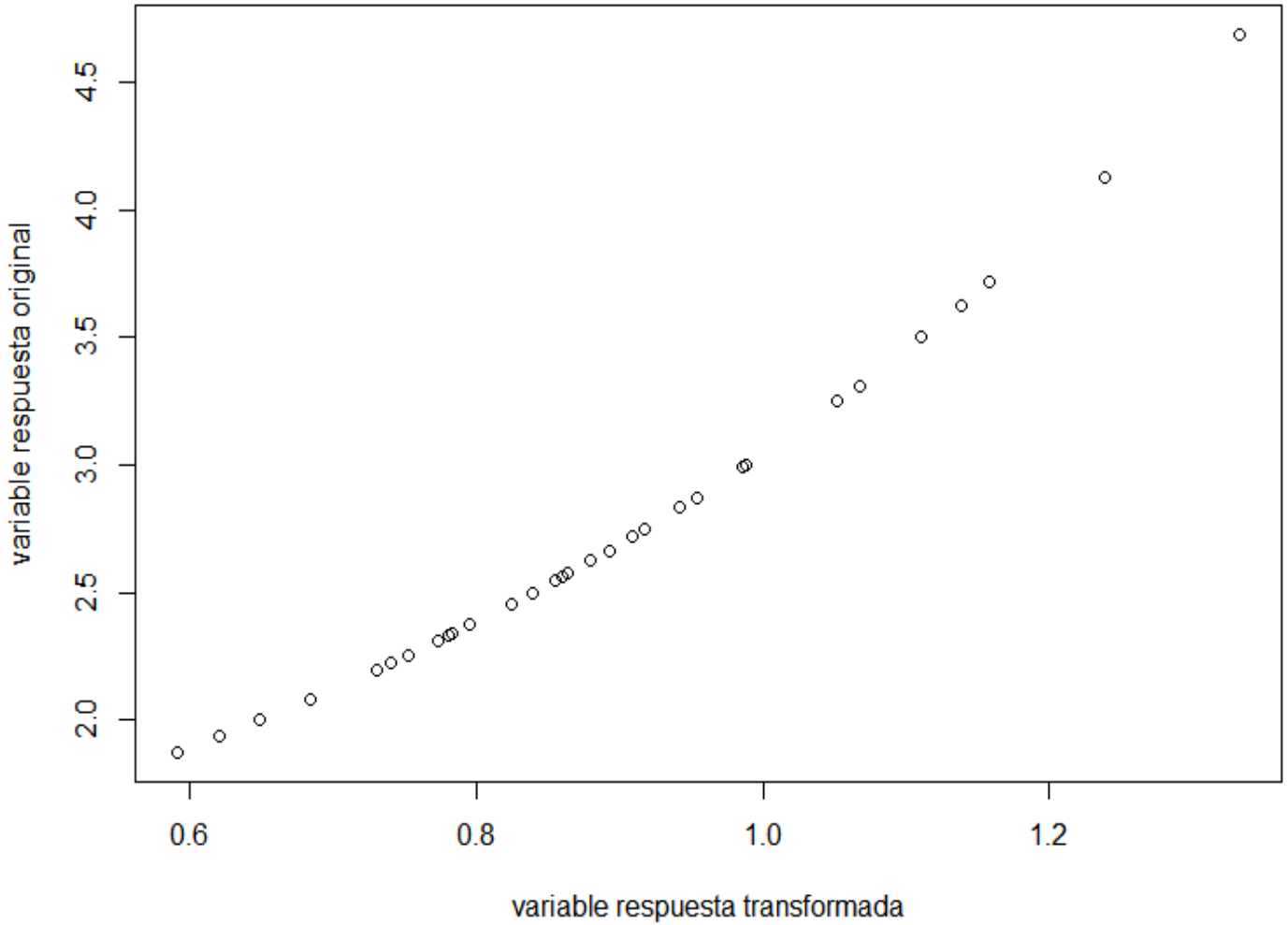
Intentando salvar el modelo General Lineal con residuos … no "buenos"

```
> print(boxCox(modelo, lambda=seq(-2,2, 1/100)))
```



Intentando salvar el modelo General Lineal con residuos … no "buenos"

```
> plot(respuesta.bc, modelo$model[,1], xlab="variable respuesta transformada",
      ylab="variable respuesta original")
```



Intentando salvar el modelo General Lineal con residuos … no "buenos"

Ahora rehacemos el modelo sustituyendo la variable respuesta del modelo por **respuesta.bc**

```
## capturamos en un nuevo data-frame los datos de trabajo de nuestro "modelo"
datos.bc <- modelo$model

## asignamos a la primera variable (¡la respuesta!) el valor de su transformada Box-Cox
datos.bc[,1] <- respuesta.bc

## corremos el mismo modelo pero con la respuesta transformada
modelo.bc <- lm(eqt, data=datos.bc)

## sus resultados son:
summary(modelo.bc)
Anova(modelo.bc, type=3, test="F")
```

A este nuevo modelo le podemos aplicar todo el resto de las cosas vistas hasta este momento.

MODELOS GENERALIZADOS

después de todas estas exploraciones de los supuestos canónicos de los modelos

…

ahora ya sí podemos proceder a valorar sus resultados

Aspectos a considerar:

Resumen del modelo
Significación de efectos
¿Qué proporción de la variabilidad en la respuesta explica el modelo?

resultados del modelo

Resultado global del modelo

Primero valoramos la significación global del modelo, en lo que se conoce como un **omnibus test**.

SI EL RESULTADO ES SIGNIFICATIVO, PODREMOS SEGUIR CON LOS RESULTADOS.
Si no resulta significativo el análisis … **¡se terminó!**

Si en los modelos Generalizados usados con poisson y binomial aplicamos la corrección por **sobredispersión**, el resultado de este omnibus test cambia.

`lrtest` examina el modelo de interés comparado con otro.
si no se especifica nada ese "otro" será el modelo nulo que no incluye ningún efecto

`waldtest` examina cuán probable es que los coeficientes del modelo sean "cero".
produce unos resultados de significación muy parecidos al **`lrtest`**
si el tamaño muestral es enorme

Los dos tests son asintóticamente equivalentes

```
## test de SIGNIFICACIÓN GLOBAL DEL MODELO
lrtest(modelo)
waldtest(modelo)
```

Resultado global del modelo

omnibus test

```
> lrtest(modelo)
Likelihood ratio test

Model 1: nspp ~ altmed + rangoalt + shannon + tempmin + precip
Model 2: nspp ~ 1
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   7 -3797.2
2   2 -3916.8 -5 239.17  < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
> waldtest(modelo)
Wald test

Model 1: nspp ~ altmed + rangoalt + shannon + tempmin + precip
Model 2: nspp ~ 1
  Res.Df Df     F    Pr(>F)
1    993
2    998 -5 53.72 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Resultado global del modelo

Comparamos los valores AICc del modelo de interés y el modelo nulo

`.~1` se refiere a un modelo SÓLO con el intercepto

el intercepto predice a la variable respuesta a través de la media de dicha variable

```
modelo.nulo <- update(modelo, .~1)

AICc(modelo, modelo.nulo)
```

```
> AICc(modelo, modelo.nulo)
             df     AICc
modelo        7 7608.597
modelo.nulo   2 7837.663

veces.mejor <- exp(-0.5*(AICc(modelo)-AICc(modelo.nulo)))
print(c("Veces que MI MODELO es mejor que el modelo NULO =", veces.mejor), quote=FALSE)
```

```
[1] Veces que MI MODELO es mejor que el modelo NULO = 5.50841447944648e+49
```

Este resultado es consistente con el anterior, pero en coordenadas de teoría de la información.
El modelo de interés de $5.5*10^{49}$ veces mejor que el modelo nulo.

Estos dos tipos de "tests" nos proporcionan que nuestro modelo es altamente "relevante", con lo cual podemos continuar valorando sus resultados.

¿Qué proporción de la variabilidad en la respuesta explica el modelo?

Variabilidad (devianza) explicada por todo el modelo

D^2 del modelo:

$$\frac{\text{DEVIANZA RESIDUAL}_{\text{TOTAL}} - \text{DEVIANZA RESIDUAL}_{\text{MODELO}}}{\text{DEVIANZA RESIDUAL}_{\text{TOTAL}}}$$

```
d2 <- round((modelo$null.deviance-modelo$deviance)/modelo$null.deviance, 3)
print(c("D2 de MCFadden =", d2), quote=FALSE)
```

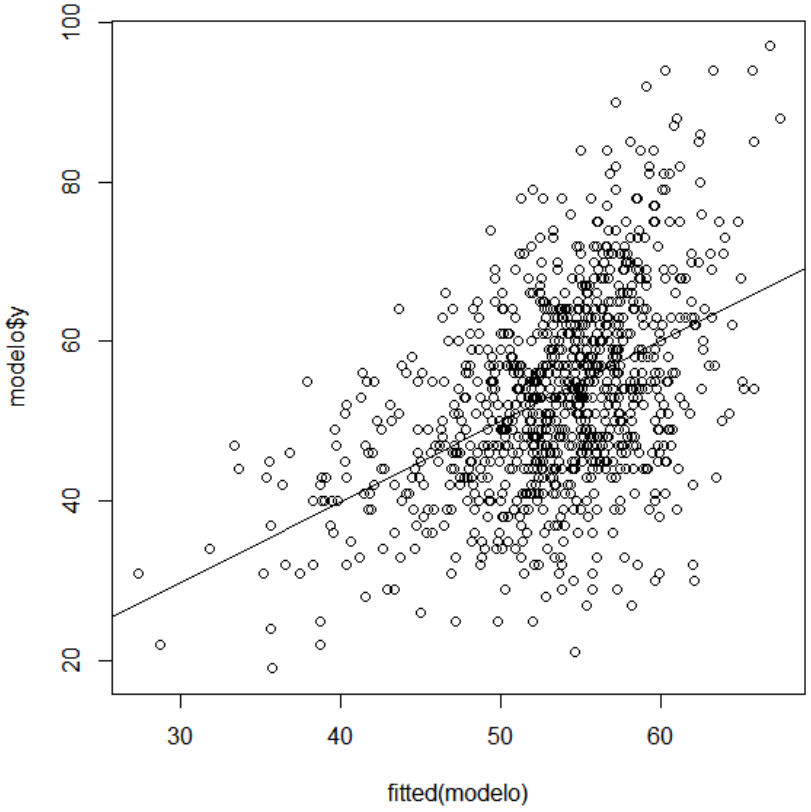
```
> print(c("D2 de MCFadden =", d2), quote=FALSE)
[1] D2 de MCFadden = 0.213
```

¿Qué proporción de la variabilidad en la respuesta explica el modelo?

Variabilidad (devianza) explicada por todo el modelo

RELACIÓN ENTRE VALORES OBSERVADOS Y PREDICHOS

```
plot(modelo$y ~ modelo$fitted.values)
plot(modelo$y ~ fitted(modelo))
## es lo mismo que la línea de código anterior
abline(lm(modelo$y ~ fitted(modelo)))
```



Resultados de los efectos

```
summary(modelo)
> summary(modelo)
Call:
glm(formula = eqt, family = gaussian(link = "identity"), data = datos,
    offset = log(ntransectos))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-33.626   -7.448   -0.087    7.327   33.713

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 51.138402   2.029443  25.198  < 2e-16 ***
altmed      -0.009964   0.001749  -5.696 1.61e-08 ***
rangoalt    -0.006862   0.001350  -5.084 4.41e-07 ***
shannon      7.069202   0.768480   9.199  < 2e-16 ***
tempmin     -0.297345   0.247064  -1.204    0.229
precip      -0.010729   0.002562  -4.188 3.07e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 117.9467)

    Null deviance: 148802  on 998  degrees of freedom
Residual deviance: 117121  on 993  degrees of freedom
AIC: 7608.5

Number of Fisher Scoring iterations: 2
```

INTERPRETACIÓN DE LOS COEFICIENTES DE REGRESIÓN

En los modelos de regresión Gausianos (link=identity) los coeficientes se interpretan normalmente

En los modelos de regresión de Poisson son multiplicativos
… porque la función de vínculo es el logaritmo: familia = poisson vínculo = log

En la regresión de Poisson $\log(Y) = \mathbf{a} + \mathbf{b}\cdot X$ o $Y = \exp(\mathbf{a} + \mathbf{b}\cdot X)$
log(Y) cambia linealmente en función de las variables predictoras
Y cambia linealmente en función del antilogaritmo de la función de las predictoras

El coeficiente **b** en antilogaritmo, **exp(b)**, mide el cambio multiplicativo en la variable respuesta "Y" cuando esa variable predictora cambia en una unidad.

O dicho de otro modo, el coeficiente **b** es el cambio esperado en el log(Y) cuando la variable predictora aumenta una unidad.

En el caso de las predictoras categóricas (definidas por nº categorías del factor – 1) el antilogaritmo del coeficiente, exp(**b**), es el término multiplicativo relativo a la "base" del factor. El antilogaritmo del intercepto, exp(**a**), es el valor basal en relación con el cual se estiman los cambios definidos por los coeficientes.

ejemplo con factores

factor ***edu*** de 4 niveles

factor ***res*** de 3 niveles

Interpretation

$$\log(y) = 1.43 + .21x_{edunone} - 1.0x_{eduse+} - 0.41x_{eduup} - 0.06x_{resSuv} + 0.06x_{resurb}$$

- The predicted number of children for a women with no education living in Suva is given by

$$\log(y) = 1.43 + .21(1) - 1.0(0) - 0.41(0) - 0.06(1) + 0.06(0)$$
$$= 1.58$$
$$\exp(1.58) = 4.85$$

- The predicated number of children for a woman with a secondary education, living in a rural area is:

$$\log(y) = 1.43 + .21(0) - 1.0(1) - 0.41(0) - 0.06(0) + 0.06(0)$$
$$= .43$$
$$\exp(.43) = 1.53$$

Significación de efectos

En `summary(modelo)` lo que hemos visto son los coeficientes de regresión del modelo.
Y esto es de fácil interpretación y lectura para las predictoras continuas (covariantes) pero no para los factores.

Para valorar mejor la magnitud de efectos y su significación, utilizando diferentes tipos de Sumas de Cuadrados (SS), o sus equivalentes de devianza, definidas … y que queramos.

```
library(car)    ## cargamos este paquete de enorme utilidad
```

Hay tres tipos de `test`:
`F`: para modelos Gaussianos y para corregir la sobredispersión (sólo con Poisson y Binomiales)
`Wald`: modelos GLM asumiendo infinitos grados de libertad y comparando vs coeficiente=0
`LR`: para todos los modelos GLM usando *likelihood ratio tests*

TIPO III de SS (el que más vamos a utilizar y el de más fácil interpretación)

```
Anova(modelo, type=3, test="F")
```

TIPO II de SS

```
Anova(modelo, type=2, test="F")
```

TIPO I de SS

```
anova(modelo)
```

Significación de efectos

```
> Anova(modelo, type=3, test="F")
Anova Table (Type III tests)

Response: nspp
              SS  Df       F    Pr(>F)
altmed      3827   1 32.4483 1.612e-08 ***
rangoalt    3049   1 25.8480 4.414e-07 ***
shannon     9981   1 84.6206 < 2.2e-16 ***
tempmin      171   1  1.4484    0.2291
precip      2069   1 17.5381 3.066e-05 ***
Residuals 117121 993
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Otras opciones son likelihood ratio test **"LR"** y **"Wald"**

La opción **"F"** corrige las significaciones por la sobredispersión de los residuos en los modelos Poisson y Binomiales.

Significación de efectos

Otra opción es usando el comando `drop1`

estimas parciales, al estilo SS type-III

nos proporciona la significación de los efectos

la **Deviance** nos vincula a las magnitudes de los efectos

<none> indica la devianza residual sin quitar efectos

los otros valores indican la devianza quitando ese efecto

.~. se podría eliminar (pero es útil para especificar efectos concretos)

estas dos líneas proporcionan idénticos resultados de la significación de los efectos

```
Anova(modelo, type=3, test="LR")
drop1(modelo, .~., test="Chisq", sorted=FALSE)
```

```
Single term deletions
Model: nspp ~ altmed + rangoalt + shannon + tempmin + precip

         Df Deviance    AIC     LRT  Pr(>Chi)
<none>        2199.8 7997.5
altmed    1   2278.2 8073.8  78.324 < 2.2e-16 ***
rangoalt  1   2263.4 8059.1  63.611 1.516e-15 ***
shannon   1   2396.2 8191.8 196.381 < 2.2e-16 ***
tempmin   1   2205.3 8000.9   5.430   0.01979 *
precip    1   2238.5 8034.1  38.652 5.064e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

comparación
de modelos usando
Akaike AIC

COMPARACIÓN DE MODELOS utilizando ***A**kaike **I**nformation **C**riteria*

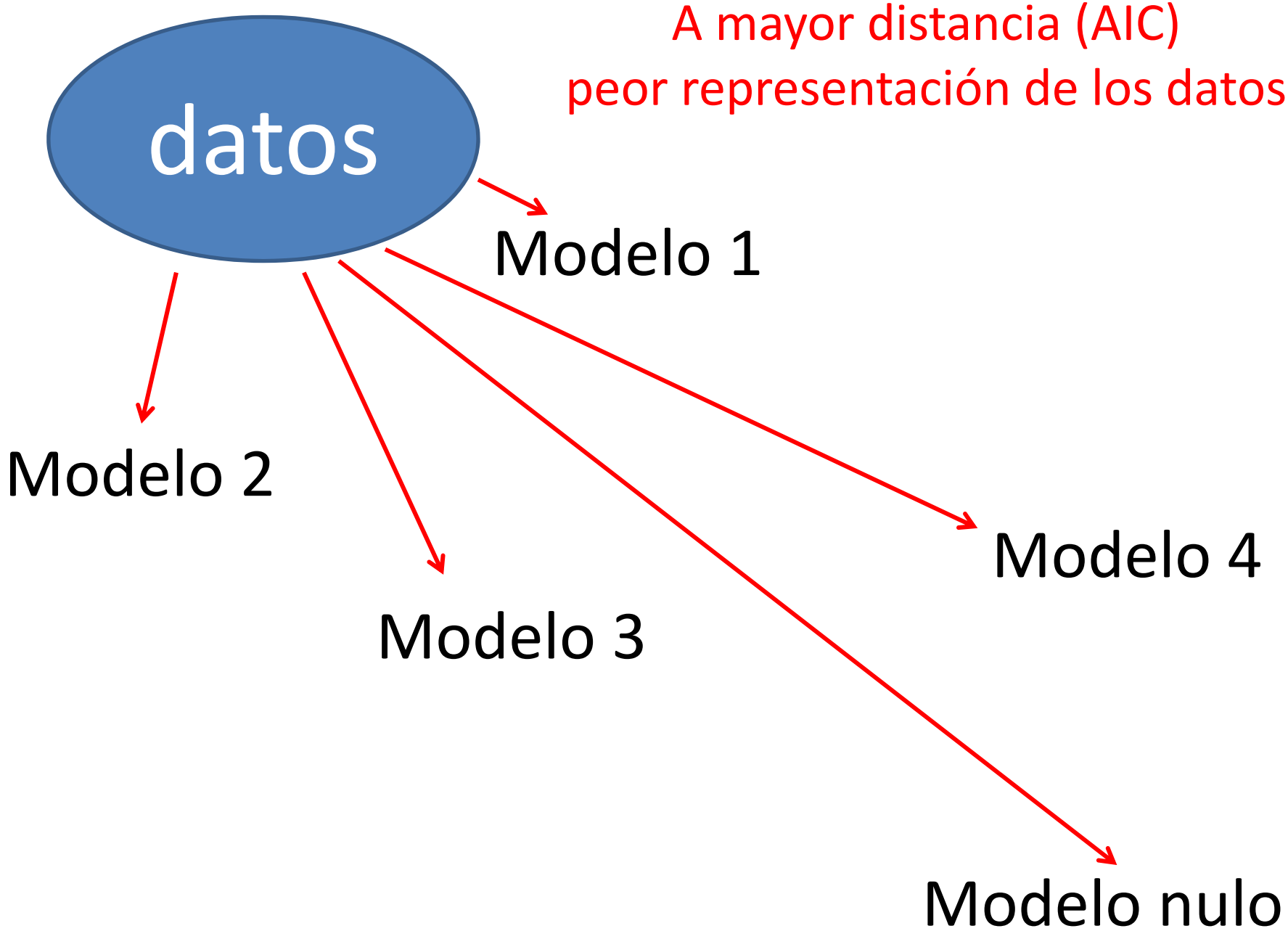
En vez de obtener la diferencia entre dos modelos, se obtiene una estima de la distancia relativa esperada entre cada modelo estimado y los verdaderos mecanismos que realmente han generado los datos observados (posiblemente de una dimensionalidad muy alta).

AIC sirve para seleccionar el mejor modelo dentro de un conjunto de estos obtenidos con los mismos datos. Debemos hacer un esfuerzo por asegurarnos de que el conjunto de modelos de trabajo sea sólido y esté bien apoyado.

AIC sirve para medir la distancia de cada modelo bajo comparación respecto a la "verdad" representada por los datos. Lo único verdadero son los datos; nuestros modelos pretenden representar esa realidad.

Consultad:

- https://en.wikipedia.org/wiki/Akaike_information_criterion
- "PROCEDIMIENTOS DE SIMPLIFICACIÓN DE MODELOS" en:
- http://www.lmcarrascal.eu/regrmult.html


A mayor distancia (AIC)
peor representación de los datos
datos
Modelo 1
Modelo 2
Modelo 3
Modelo 4
Modelo nulo

COMPARACIÓN DE MODELOS utilizando ***A**kaike **I**nformation **C**riteria*

En el caso de modelos GLM, AIC se calcula del siguiente modo:

$$\mathbf{AIC = n\cdot[\ln(2\cdot\pi)+1] + n\cdot\ln(SS_{error}/n) + 2\cdot K}$$

donde **n** es el tamaño muestral,
$\mathbf{SS_{error}/n}$ es la varianza residual ($\mathbf{SS_{error}}$ es la suma de cuadrados error del modelo)
y **K** es el número de parámetros del modelo de regresión (intercepto + predictores + error).

Otra expresión simplificada, a efectos comparativos, es $\mathbf{AIC = n\cdot\ln(SS_{error}/n) + 2k}$

En el caso de modelos Generalizados

$$\mathbf{AIC = 2\cdot K - 2\cdot\ln(L)}$$

donde **L** es la estima de "maximum likelihood"
y **K** el número de parámetros del modelo de regresión.

Lo importante no es el valor absoluto de AIC, **sino las diferencias entre los valores $\mathbf{AIC_i}$ de i modelos** (desde i=1 a i=R, siendo R modelos = comparados)

AIC se recomienda cuando **n/K** es mayor de 40.
siendo **n** el número de observaciones (tamaño muestral)

Si este no es el caso, deberíamos utilizar:
Akaike's second order information criterion ($\mathbf{AIC_c}$):

$$AIC_c = AIC + (2\cdot K\cdot(K+1))/(n-K-1)$$

COMPARACIÓN DE MODELOS utilizando ***A**kaike **I**nformation **C**riteria*

Trabajaremos, por tanto, con **diferencias en una serie de valores AIC.**

Para ello seleccionaremos el menor valor AIC dentro de nuestro subconjunto de modelos (AIC_{min}), para a continuación calcular incrementos de AIC sobre ese valor mínimo.

$$\Delta_i = AIC_i - AIC_{min}$$

No son los valores absolutos de AIC_i lo importante, sino las …
diferencias relativas entre los AIC_i (Δ_i) de diferentes modelos.

Escala relativa de plausibilidad de modelos:

Δ_i	**Plausibilidad**
0 – 2	Similar
4 – 7	Menor
> 10	Mucho menor

La **verosimilitud relativa** de un modelo se calcula mediante **$\exp(-0.5 \cdot \Delta_i)$**.
cuantifica cuántas veces un modelo es mejor que el otro

sobredispersión

cómo y a qué tratarla

SOBREDISPERSIÓN DEL MODELO

Medida para estimar la bondad de ajuste del modelo (**ϕ**).

SÓLO se aplica a las distribuciones definidas por un solo parámetro:

Poisson: lambda (media de la respuesta)

Binomial: prob (probabilidad del estado no-cero)

Mide la existencia de una mayor (o menor) variabilidad que la esperable en la variable respuesta considerando los supuestos acerca de su distribución canónica y la función de vínculo (que liga los valores transformados de la variable a las predicciones del modelo)
http://en.wikipedia.org/wiki/Overdispersion

ϕ debería valer 1.

Si >1 sobredispersión → se "inflan" las significaciones

a sobreparametrización

Goodness of Fit[b]

	Value	df	Value/df
Deviance	2977,482	991	3,005
Scaled Deviance	2977,482	991	
Pearson Chi-Square	2924,210	991	2,951
Scaled Pearson Chi-Square	2924,210	991	
Log Likelihood[a]	-4381,554		
Akaike's Information Criterion (AIC)	8779,109		
Finite Sample Corrected AIC (AICC)	8779,254		
Bayesian Information Criterion (BIC)	8818,363		
Consistent AIC (CAIC)	8826,363		

¡¡¡SOBREDISPERSIÓN!!!

las estimas de significación están "infladas"

Con estos valores (**ϕ**) recalculamos nuevas estimas de significación a través de la **F**.

F = diferencias en Devianza / (dif. en g.l. x **ϕ**)

aparecerán en los resultados en:

Test of Model Effects

MODELOS GENERALIZADOS

SOBREDISPERSIÓN DEL MODELO

Veámoslo prácticamente en R utilizando la distribución de Poisson.

```
## estima de la sobredispersión del modelo
## este valor canónico debería de ser igual a la unidad
##
phi <- sum((residuals(modelo, type="pearson"))^2)/modelo$df.residual
print(c("Pearson overdispersion =", round(phi, 3)), quote=FALSE)
```

```
[1] Pearson overdispersion = 1.176
```

Si este valor hubiese sido muy diferente de uno (e.g., > 2) recalcularíamos el modelo teniendo en cuenta ese valor de sobredispersión, aplicando la **pseudofamilia** `quasipoisson`.
No corregiremos por sobredispersión si **phi <1**

```
modelo2 <- glm(eqt, data=datos, family=quasipoisson(link="log"))
```

Y procederíamos con este nuevo modelo.

afrontando la

heterocedasticidad de los residuos

EXAMEN DEL EFECTO DE LA VIOLACIÓN DEL SUPUESTO DE HOMOCEDASTICIDAD

Hasta ahora hemos efectuado el examen de la significación de los efectos asumendo que ha existido homocedasticidad de los residuos. Si hubiésemos violado este supuesto, deberíamos efectuar nuevas estimas de significación teniendo en cuenta esos desvíos de la homocedasticidad.

Al recalcular las significaciones de los efectos del modelo teniendo en cuenta distintas estructuras de matrices de varianza-covarianza (**vcov**), sólo van a cambiar los errores estándar de los parámetros (coeficientes) del modelo y sus significaciones.

Para ello podemos usar el comando **coeftest**, definiendo diferentes tipos de estructuras **vcov**.

`const`: constante u homogénea; no se ha violado el supuesto de homocedasticidad
`HC0`: sandwich
`HC3`: la mejor corrección para pequeñas muestras, dando menos peso a los datos influeyente
`HC4`: mejora HC3 , especialmente en el caso de datos muy influyentes (alto leverage)
`HC4m`: mejora HC4, tanto con residuos del modelo normales, como "menos" normales (gausianos)
`HC5`: los errores estándar obtenidos proporcionan mejores inferencias para muestras "finitas"

La limitación que tiene es que estima la significación de coeficientes de las columnas de contraste para los factores, y no el efecto en sí.

El uso del comando es (en "modelo" incluimos el nombre de nuestro modelo `lm` o `glm`:

```
coeftest(modelo, vcovHC(modelo, type="HC4m"))
```

EXAMEN DEL EFECTO DE LA VIOLACIÓN DEL SUPUESTO DE HOMOCEDASTICIDAD

```
> coeftest(modelo, vcovHC(modelo, type="const"))

 E           stimate Std. Error z value  Pr(>|z|)
(Intercept) 51.1384017  2.0294429 25.1982 < 2.2e-16 ***
altmed      -0.0099640  0.0017492 -5.6963 1.224e-08 ***
rangoalt    -0.0068619  0.0013497 -5.0841 3.694e-07 ***
shannon      7.0692018  0.7684797  9.1989 < 2.2e-16 ***
tempmin     -0.2973450  0.2470637 -1.2035   0.2288
precip      -0.0107294  0.0025620 -4.1879 2.816e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

los coeficientes no cambian

```
> coeftest(modelo, vcovHC(modelo, type="HC4m"))

             Estimate Std. Error z value  Pr(>|z|)
(Intercept) 51.1384017  1.9129487 26.7328 < 2.2e-16 ***
altmed      -0.0099640  0.0016623 -5.9942 2.045e-09 ***
rangoalt    -0.0068619  0.0012949 -5.2993 1.162e-07 ***
shannon      7.0692018  0.7538525  9.3774 < 2.2e-16 ***
tempmin     -0.2973450  0.2506976 -1.1861   0.2356
precip      -0.0107294  0.0025905 -4.1418 3.445e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

las significaciones sí cambian

(y sus valores asociados de se y t)

EXAMEN DEL EFECTO DE LA VIOLACIÓN DEL SUPUESTO DE HOMOCEDASTICIDAD

Otra posibilidad de corregir las estimas de significación es mediante el argumento
white.adjust dentro del comando **Anova**(…).

white.adjust puede tomar los valores:
FALSE (en cuyo caso no se efectúa ningún ajuste por heterocedastcidad),
"hc0", "hc1", "hc2", "hc3", "hc4"
(consultad http://ftp.auckland.ac.nz/software/CRAN/doc/vignettes/sandwich/sandwich.pdf)

Mediante las siguientes líneas de código podemos crear una tabla ANOVA de resultados, mostrando los valores de F y P sin corregir y corregidos:

```
tmp1 <- Anova(modelo, type=3, test="F", white.adjust=FALSE)
tmp2 <- Anova(modelo, type=3, test="F", white.adjust="hc4")
tmp1$F_hc4 <- tmp2$F;   tmp1$P_hc4 <- tmp2[,4]
tabla.efecto.heterocedst <- tmp1
round(tabla.efecto.heterocedst, 5)
```

EXAMEN DEL EFECTO DE LA VIOLACIÓN DEL SUPUESTO DE HOMOCEDASTICIDAD

En **F_hc4** y **P_hc4** podemos ver los resultados de F y P corregidos teniendo en cuenta el desvío de la homocedasticidad en los residuos, respecto a los valores originales F y P sin corregir.

A más diferencia entre esos valores, más efecto tiene la violación del supuesto de la homocedasticidad de los residuos, y más influencia tienen observaciones con alto leverage.

```
> round(tabla.efecto.heterocedst, 5)
Analysis of Deviance Table (Type III tests)

Response: nspp
             SS  Df       F  Pr(>F)  F_hc4   P_hc4
altmed     3827   1 32.4483 0.00000 32.448 0.00000
rangoalt   3049   1 25.8480 0.00000 25.848 0.00000
shannon    9981   1 84.6206 0.00000 84.621 0.00000
tempmin     171   1  1.4484 0.22906  1.448 0.22906
precip     2069   1 17.5381 0.00003 17.538 0.00003
Residuals 117121 993
```

magnitud de efectos

explicación de la variable respuesta por cada una de las predictoras

¿Qué proporción de la variabilidad en la respuesta explica el modelo?

PARTICION DE LA VARIABILIDAD ENTRE LOS PREDICTORES

magnitudes de efectos usando las devianzas

```
tabla.devianza <- drop1(modelo, .~., test="LRT")[,c(1:3)]
tabla.devianza[1,1] <- sum(tabla.devianza[-1,1])
tabla.devianza$Dev.retenida <- tabla.devianza$Deviance - tabla.devianza[1,2]
concomitancias <- ((modelo$null.deviance-modelo$deviance) -
                   sum(tabla.devianza[,4])) / modelo$null.deviance
tabla.devianza[1,4] <- modelo$null.deviance - modelo$deviance
rownames(tabla.devianza)[1] <- "modelo"
tabla.devianza$proporcion.explicada <-
                tabla.devianza$Dev.retenida / modelo$null.deviance
```

La diferencia entre la "proporcion.explicada" para el modelo y la suma de esos valores para los efectos mide las concomitancias entre los términos predictores

```
round(tabla.devianza, 3)
print(c("Concomitancias (proporción) =",round(concomitancias, 3)), quote=FALSE)
```

¿Qué proporción de la variabilidad en la respuesta explica el modelo?

```
> round(tabla.devianza, 3)
          Df Deviance    AIC Dev.retenida proporcion.explicada
modelo     5   117121 7608.5        31681                 0.213
altmed     1   120948 7638.6         3827                 0.026
rangoalt   1   120170 7632.2         3049                 0.020
shannon    1   127102 7688.2         9981                 0.067
tempmin    1   117292 7607.9          171                 0.001
precip     1   119190 7624.0         2069                 0.014

> print(c("Concomitancias (proporción) =",round(concomitancias, 3)), quote=FALSE)
[1] Concomitancias (proporción) = 0.085
```

explicación *vs.* predicción

predecibilidad de la variable respuesta

¿Cuál es el poder predictivo del modelo?

Bueno … es un asunto engorroso. ¡Reconozcámoslo!

Una cosa es cómo explico mis datos con un modelo

Y otra es cómo un modelo explica los datos … que no sólo son los míos

(esto es "tabú", por incómodo, en algunas disciplinas de investigación)

Asumir este reto, implica dar el salto de **(auto)explicar** a **predecir**.

Podríamos repetir de nuevo nuestro experimento, toma de datos, etc.
Esto es muy costoso (en tiempo, dinero, personal) y no es manejable en la práctica de investigación.

Un "atajo" es cros-validar nuestros datos (***cross-validation***).

Este procedimiento consiste en dividir nuestros datos, al azar, en grupos de datos (*v-fold*)

Hacemos **v** número de modelos, cada uno con **v-1** grupos

Y con ese modelo predecimos el grupo de datos no considerado en la construcción del modelo

estimamos las predicciones de ese modelo para el conjunto de datos no utilizado en función de los valores que toman sus unidades muestrales en los factores y covariantes

Juntamos todas las predicciones de los **v** grupos … cuando no se han considerado sus datos

Y ahora … el drama: correlacionamos los valores observados con los predichos

cuando los datos no han sido utilizados en los **v** modelos.

¿Cuál es el poder predictivo del modelo?

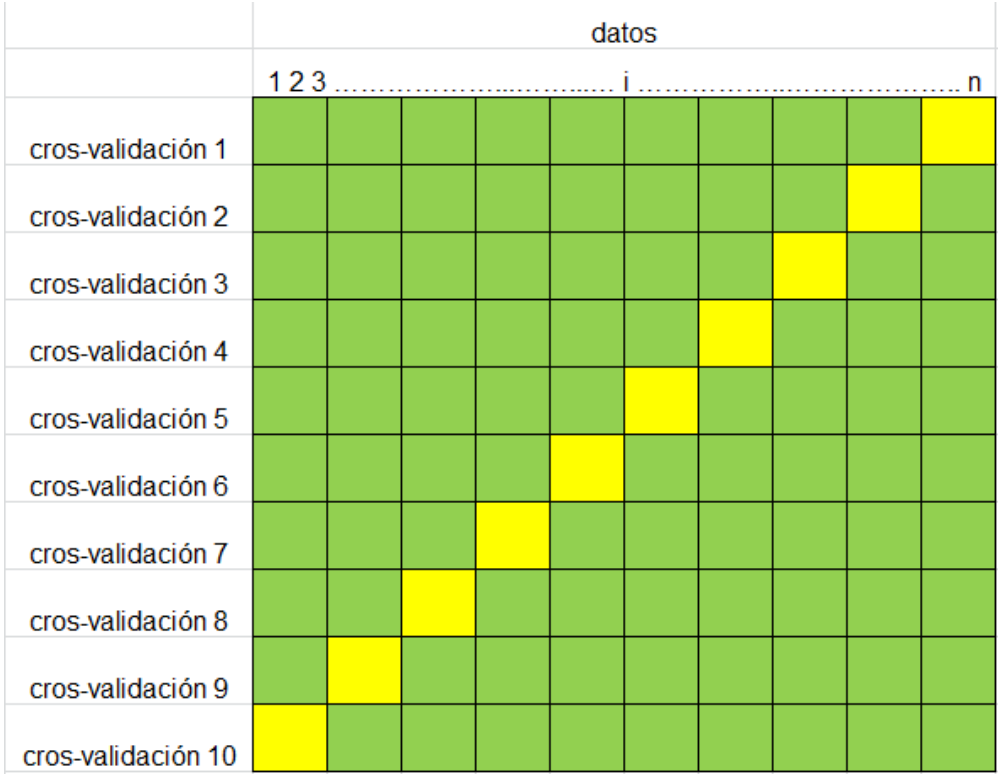
Veámoslo gráficamente:

hacemos los modelos con los grupos de datos **verdes**

y predecimos los datos **amarillos** que no hemos utilizado

hay tantos modelos como grupos de datos **v** hemos creado

cada modelo se realiza con los datos contenidos en los **v-1** grupos



¿CUÁL ES EL PODER PREDICTIVO DEL MODELO?

Para realizar estos cálculos intesivos utilizando **glm**

```
## número de particiones de datos
N_fold <- 10

## preparando los datos
##   obtenemos los datos de trabajo usados en el modelo
datos2 <- modelo$model
##   generamos números al azar para efectuar particiones aleatorias
datos2$azar <- runif(n=dim(datos2)[1], min=1, max=1000)
datos2 <- datos2[order(datos2$azar),]
datos2$azar <- NULL
datos2$id <- rep(c(1:N_fold), length.out=dim(datos2)[1])
##   valores y matriz donde se almacenarán los datos
predichos <- 99999
observados <- 99999
id_cv <- 0
matriz_cv <- data.frame(predichos, observados, id_cv)
```

¿CUÁL ES EL PODER PREDICTIVO DEL MODELO?

```
## bucle de validación cruzada
for (i in 1:N_fold) {
  datos3 <- subset(datos2, id !=i)
  respuesta <- datos3[,1]
  datos3 <- datos3[, -1]
  mod.predict <- glm(respuesta~.-id, family=modelo$family, data=datos3)
  datos4 <- subset(datos2, id==i)
  predichos <- predict(mod.predict, newdata=datos4, type="response")
  observados <- datos4[,1]
  id_cv <- datos4$id
  datos_cv <- data.frame(predichos, observados, id_cv)
  matriz_cv <- rbind(matriz_cv, datos_cv)
}
matriz_cv <- matriz_cv[-1,]
```

¿CUÁL ES EL PODER PREDICTIVO DEL MODELO?

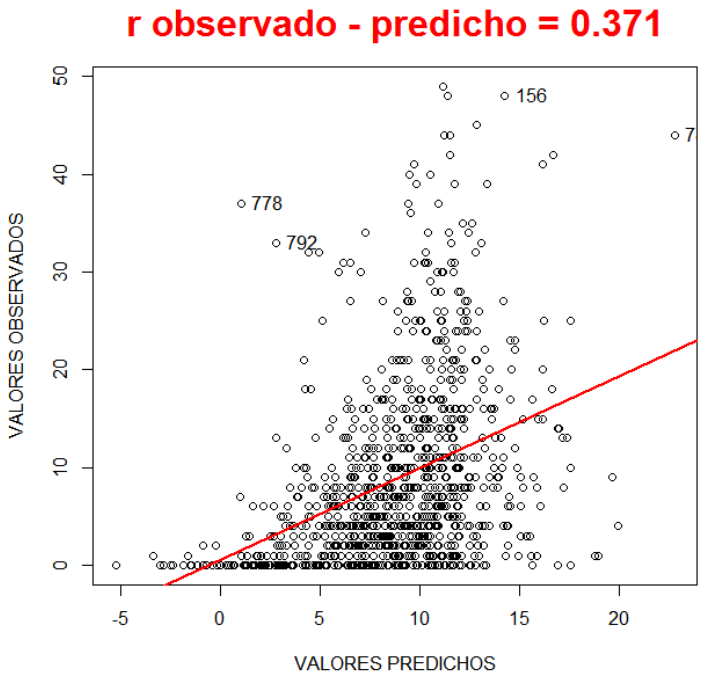
resultado de la validación cruzada para **RESPUESTAS CONTINUAS**

```
r.obs_pred <- round(cor(matriz_cv$predichos, matriz_cv$observados), 3)
resultado <- paste("r observado - predicho =", r.obs_pred)

plot(matriz_cv$predichos, matriz_cv$observados, xlab="VALORES PREDICHOS",
ylab="VALORES OBSERVADOS", cex.lab=1)
abline(lm(matriz_cv$observados ~ matriz_cv$predichos), col="red", lwd=2)
title(main=resultado, cex.main=2, col.main="red")
```

dad clik a los datos-puntos que deseéis saber qué unidades son

```
identify(matriz_cv$predichos, matriz_cv$observados)
```

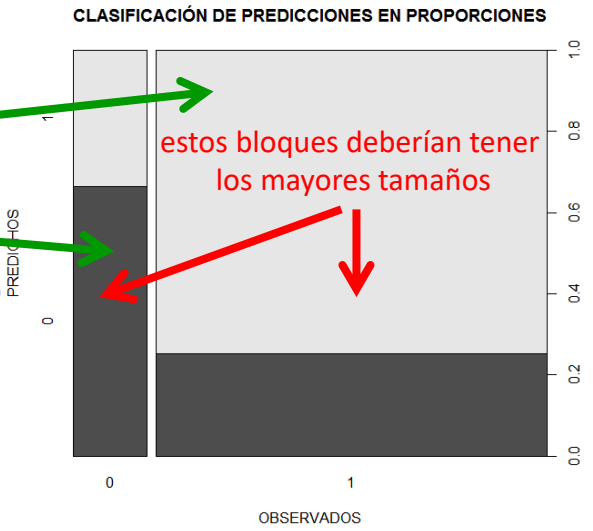
terminad dando clik al botón de Finish



¿CUÁL ES EL PODER PREDICTIVO DEL MODELO?

```
## resultado de la validación cruzada para RESPUESTAS BINOMIALES
respuesta <- factor(matriz_cv$observados)
predichos <- factor(ifelse(matriz_cv$predichos>mean(modelo$y), 1, 0))
## asumiendo que el umbral de corte es la prevalencia de (0-1)
plot(respuesta, predichos, xlab="OBSERVADOS", ylab="PREDICHOS",
     main="CLASIFICACIÓN DE PREDICCIONES EN PROPORCIONES")

## tabla 2x2 de frecuencias observadas y predichas (las columnas son las categorías predichas)
tb.obspre <- table(respuesta, predichos)
## tabla con valores observados y predichos
print("PORCENTAJES DE MUESTRAS CORRECTAMENTE CLASIFICADAS (predichas en columnas)")
round(100*tb.obspre/margin.table(tb.obspre),1)
print("TABLA DE NÚMERO DE MUESTRAS CORRECTAMENTE CLASIFICADAS (predichas en columnas)", quote=FALSE)
tb.obspre
```

```
[1] "PORCENTAJES DE MUESTRAS CORRECTAMENTE CLASIFICADAS (predichas en columnas)"
          predichos
respuesta    0    1
        0 10.5  5.3
        1 21.3 62.9

[1] TABLA DE NÚMERO DE MUESTRAS CORRECTAMENTE CLASIFICADAS (predichas en columnas)
          predichos
respuesta   0   1
        0 105  53
        1 213 628
```



particularidades con la Binomial Negativa

Modelos de regresión utilizando una Binomial Negativa

Repetiremos todos los pasos previos, sólo que en esta ocasión nuestro modelos será:

```
modelo <- glm.nb(eqt, data=datos, link=log)
```

```
> summary(modelo)

Call:
glm.nb(formula = eqt, data = datos, link = log, init.theta = 10.08705502)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1470  -1.0874  -0.4259   0.5242   2.0050

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)              -0.591067   0.244692  -2.416 0.015711 *
covariante                0.009298   0.002735   3.399 0.000676 ***
insolacion1               0.896394   0.134216   6.679 2.41e-11 ***
tratamiento1              0.698006   0.133797   5.217 1.82e-07 ***
insolacion1:tratamiento1 -0.149458   0.134011  -1.115 0.264736
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for Negative Binomial(10.0032) family taken to be 1)

    Null deviance: 305.36  on 111  degrees of freedom
Residual deviance: 114.48  on 107  degrees of freedom
AIC: 335.01

Number of Fisher Scoring iterations: 1

              Theta:  10.00
          Std. Err.:  7.52

 2 x log-likelihood:  -3223.012
```

En esta ocasión el modelo estima un parámetro más Theta
que mide la sobredispersión de un modelo Binomial Negativo
este parámetro se relaciona con el "size" de la distribución NegBin
size = 1/Theta
no deberíamos corregir por sobredispersión

particularidades con la Binomial

Modelos de regresión utilizando una Binomial

En esta ocasión nuestro modelos será:

```
modelo <- glm(eqt, data=datos, family=binomial(link="logit"))
```

Si hay sobredispersión utilizaremos la pseudofamilia:

```
family=quasibinomial(link="logit")
```

Si no hay buenos ajustes o alta sobredispersión utilizaremos la función de vínculo:

```
family = binomial(link ="cloglog")
```

cloglog trabaja mejor con distribuciones extremadamente sesgadas
por ejemplo: porporciones de un estado <0.1 o >0.9

Modelos de regresión utilizando una Binomial

Si nuestra variable respuesta no es una binomial con estados [0-1] ó [SI-NO] entonces podremos construir un modelo definiendo esa **variable respuesta "frecuencia"**.

Hay **dos modos**:

- la respuesta es un valor "proporción" (acotado entre cero y uno)
- la respuesta es un valor combinado de dos vectores: **valores SI, valores NO**

Para **proporciones**, tenemos que definir el denominador que genera la frecuencia en **weights**

```
modelo <- glm(eqt, data=datos, family=binomial(link="logit"), weights=denominador)
```

Para **respuestas combinadas**, tenemos que definir los dos vectores **conteo-SI**, **conteo-NO** en una nueva variable respuesta con el comando **cbind**

```
cbind(valoresSI, valoresNO)
```

```
## ejemplo con los datos de trabajo
eqt <- as.formula(cbind(presen8, ausen8) ~ covariante + insolacion * tratamiento)

modelo <- glm(eqt, data=datos, family=binomial(link="logit")
```

Modelos de regresión utilizando una Binomial

Nuestro modelo ahora tendrá la **forma**:

p: proporción de un "estado" respecto a toda la muestra
(80 "ceros" y 20 "unos", N=100: p = 20/100 = 0.20)

X: k variables predictoras

$$\text{logit}(p) = \log[p/(1-p)] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots + \beta_k X_k$$

$$p/(1-p) = \exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots + \beta_k X_k)$$

exp: antilogaritmo

$$p = \frac{[\exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots + \beta_k X_k)]}{1 + [\exp(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \ldots + \beta_k X_k)]}$$

El modelo **Generalizado Lineal** **Logit** predice valores de probabilidad continua (**p**):
entre 0 y 1.

Interpretación de los coeficiente utilizando una Binomial

Considerando las relaciones previas, vamos a interpretar el significado de los coeficientes:

si logit (**p**) = log [**p** / (1 – **p**)] = $\beta_0 + \beta_1 X_1 + \ldots$

entonces **p** define la probabilidad de éxito y (1 – **p**) la probabilidad de fracaso

o **p** la probabilidad de un estado y (1 – **p**) la probabilidad de ocurrencia del contrario

el cociente **p** / (1 – **p**) establece lo que se conoce como *odds ratio*

el *odds ratio* cuantifica cuántas veces es más probable un estado que otro

el antilogaritmo (*exp*) de los coeficientes β_0, β_1, … cuantifican esos *odds*

Sea una variable respuesta con dos estados: 1-sí-éxito y 0-no-fracaso:

para el intercepto, $\exp(\beta_0)$ mide cuántas veces es más probable el estado 1 que el 0 cuando todas las predictoras X_1 X_2 … toman el valor cero.

para las predictoras continuas, $\exp(\beta_i)$ mide cuántas veces es más probable 1 que 0 cuando esa predictora aumenta en una unidad de medida (e.g., por 1 ºC).

para las predictoras nominales, $\exp(\beta_i)$ mide cuántas veces es más probable 1 que 0 cuando ese estado del factor se compara con otro (e.g., al ser macho respecto a hembra).

Interpretación de los coeficiente utilizando una Binomial

Veamos una tabla para interpretar el significado de los coeficientes:

		exp(β)	
p (estado 1)	1 - p (estado 0)	odds ratio p / (1-p)	coeficientes β (log_odds)
0.001	0.999	0.0010	-6.9068
0.010	0.990	0.0101	-4.5951
0.100	0.900	0.1111	-2.1972
0.200	0.800	0.2500	-1.3863
0.250	0.750	0.3333	-1.0986
0.333	0.667	0.4993	-0.6946
0.500	0.500	1.0000	0.0000
0.666	0.334	1.9940	0.6901
0.750	0.250	3.0000	1.0986
0.800	0.200	4.0000	1.3863
0.900	0.100	9.0000	2.1972
0.990	0.010	99.0000	4.5951
0.999	0.001	999.0000	6.9068

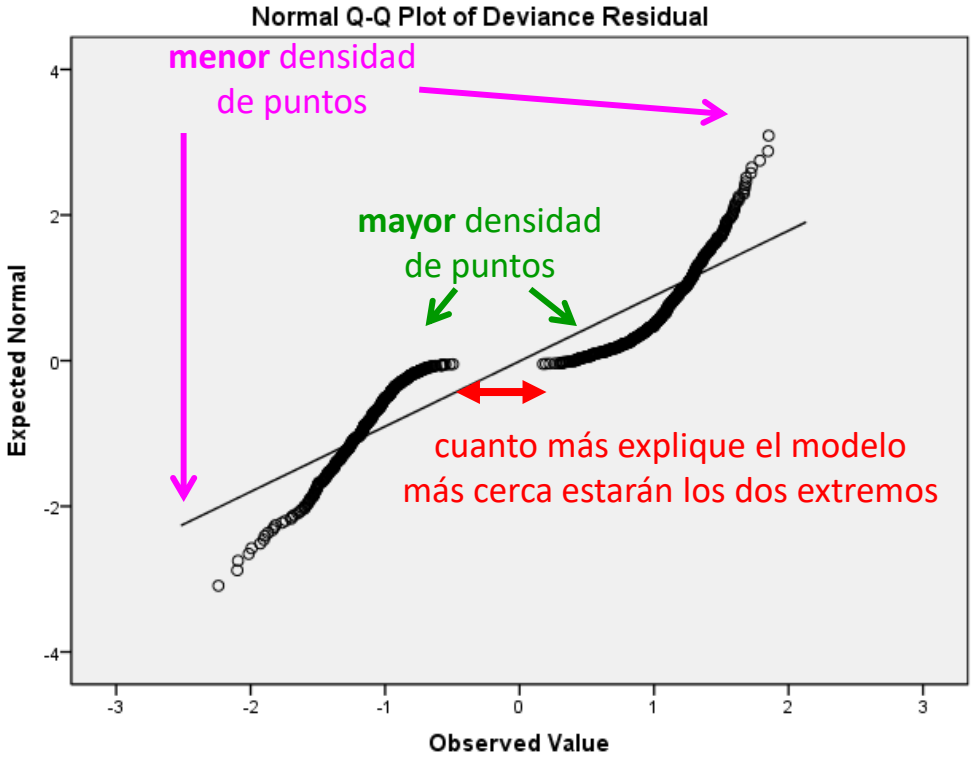
Lecturas:

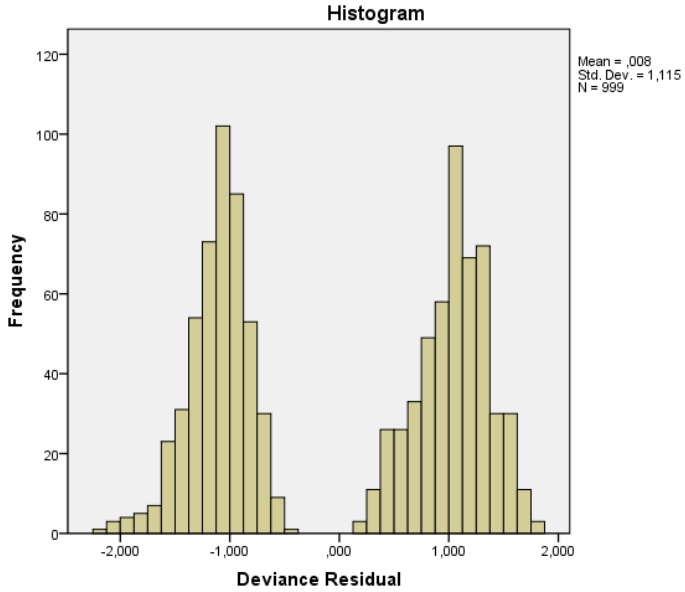
https://stats.idre.ucla.edu/other/mult-pkg/faq/general/faq-how-do-i-interpret-odds-ratios-in-logistic-regression/
http://freerangestats.info/blog/2018/08/17/risk-ratios
https://bookdown.org/roback/bookdown-bysh/ch-logreg.html

Residuos utilizando una Binomial

La exploración de los residuos en esta ocasión es un tanto diferente, debido al estado binomial de la respuesta con dos valores discretos (e.g., 0-1, sí-no).

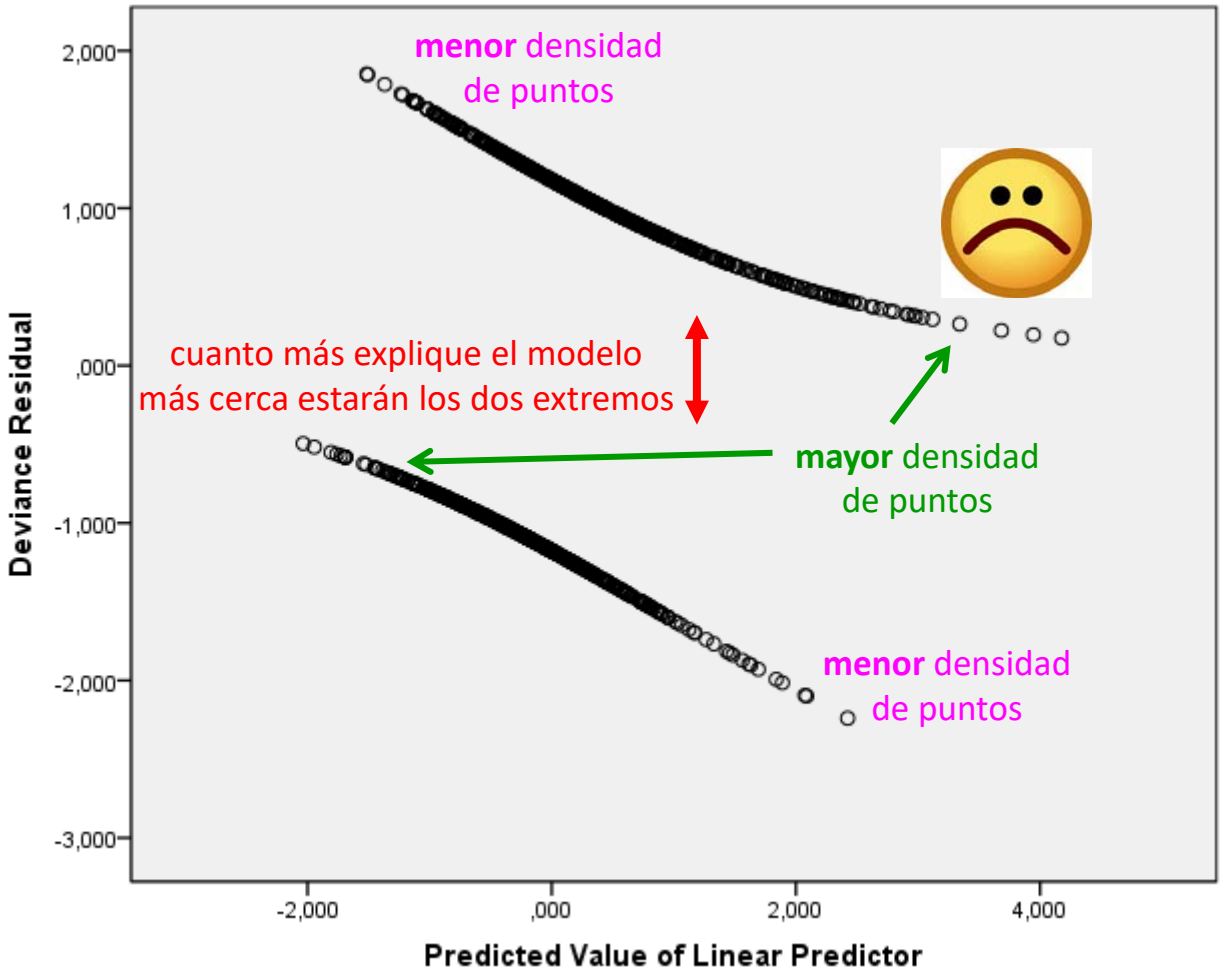
Con la "**normalidad de los residuos**" **de devianza**, en el mejor de los casos, tendríamos algo parecido a lo siguiente (con antisimetría en los dos lados del "bigote"):





Residuos utilizando una Binomial

En el caso de la **relación entre los residuos de devianza** y las **predicciones del modelo** (predictor lineal al que se le aplica la transformación ***logit***), esperaríamos encontrar algo como esto:



Residuos utilizando una Binomial

El **Modelo Generalizado Binomial** produce **probabilidades de ocurrencia p** de uno de los estados de la variable respuesta (e.g., el valor 1 en 0-1, o sí en sí-no).

Estos valores de **p**, continuos entre 0 y 1, hay que convertirlos a "estados" 0 o 1, utilizando **umbrales de corte**.

Estos valores umbrales nos permitirán convertir "probabilidades" en "estados".

por ejemplo, si el umbral es p=0.5
- si p≤0.5 entonces es "cero"
- si p>0.5 entonces es "uno"

Podemos utilizar como **umbral de corte** (*cut-off point*) la proporción real observada. No obstante, en muchas ocasiones este es un valor incierto, y es conveniente preguntarse:
¿cómo de bueno es nuestro modelo "clasificando las observaciones" independientemente de los valores umbral de corte?

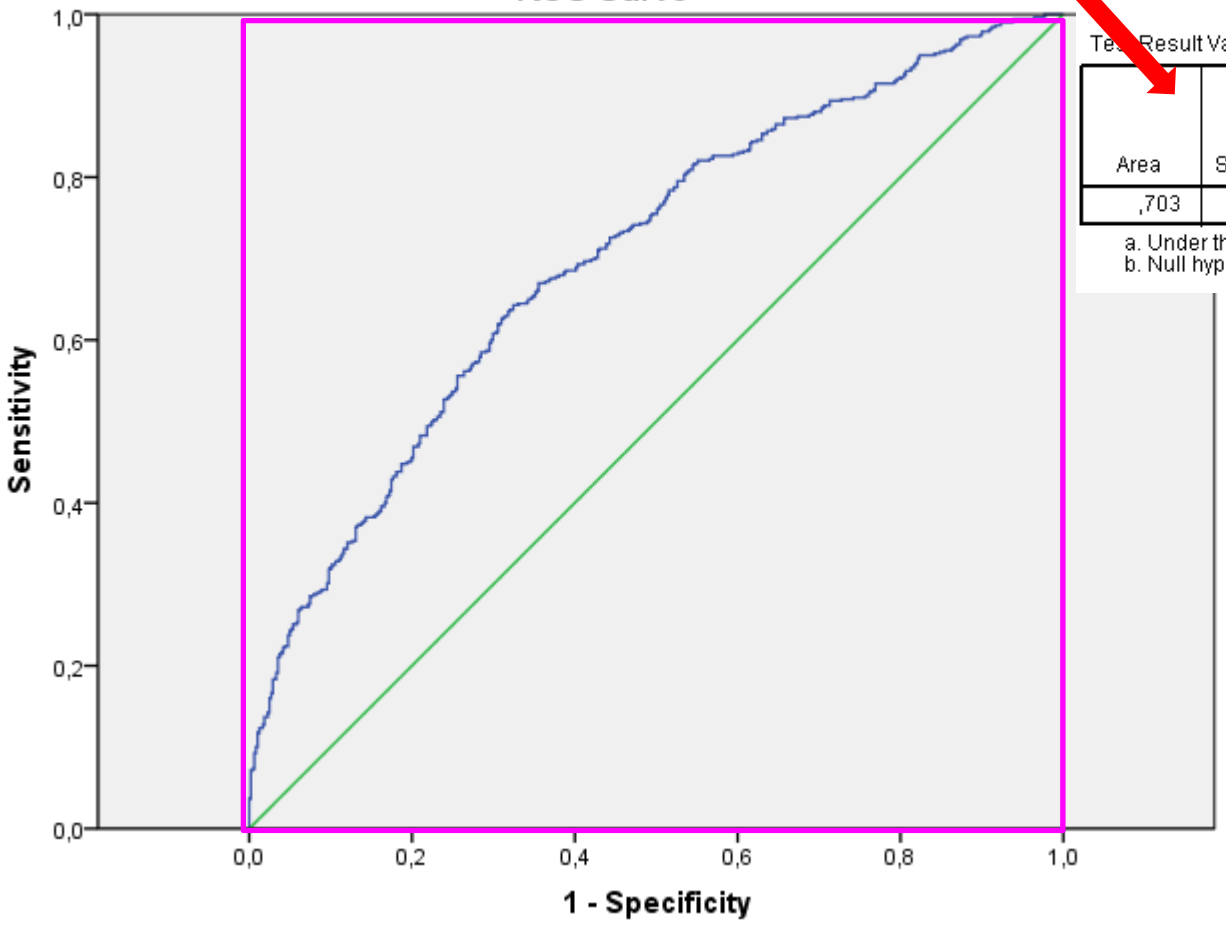
Para ello podemos contar con los **diagramas ROC** (*Receiver operating characteristic*):
https://en.wikipedia.org/wiki/Receiver_operating_characteristic
http://www.anaesthetist.com/mnm/stats/roc/Findex.htm (excelente página)

DIAGRAMAS ROC EN MODELOS GENERALIZADOS BINOMIALES

El área en el **cuadrado morado** suma "uno". De esa área,

¿cuánto ocupa la superficie bajo la curva azul? (la proporción es el valor **AUC**)

AUC



Area Under the Curve

Test Result Variable(s):Predicted Value of Linear Predictor

Area	Std. Error[a]	Asymptotic Sig.[b]	Asymptotic 95% Confidence Interval	
			Lower Bound	Upper Bound
,703	,016	,000	,671	,735

a. Under the nonparametric assumption
b. Null hypothesis: true area = 0.5

DIAGRAMAS ROC EN MODELOS GENERALIZADOS BINOMIALES

Las líneas de código que podemos usar son:

```
library(ROCR)        ## para obtener AUC en modelos GLM Binomiales

## CÁLCULO DE AUC
##    https://en.wikipedia.org/wiki/Receiver_operating_characteristic
## solo para modelos con una respuesta binomial con dos niveles (no válido para frecuencias)


pred.modelo <- prediction(fitted(modelo), modelo$y)
perf.modelo <- performance(pred.modelo,"tpr","fpr", measure="auc")
print(c("AUC =", round(perf.modelo@y.values[[1]], 3)), quote=FALSE)
plot(performance(pred.modelo,"tpr","fpr"),colorize=TRUE)


## cálculo de parámetros de precisión para el MODELO
sensit <- performance(pred.modelo, measure="sens")@y.values[[1]]
especi <- performance(pred.modelo, measure="spec")@y.values[[1]]
cutoff <- performance(pred.modelo, measure="sens")@x.values[[1]]
acc <- performance(pred.modelo, measure="acc")@y.values[[1]]
ppv <- performance(pred.modelo, measure="ppv")@y.values[[1]]
npv <- performance(pred.modelo, measure="npv")@y.values[[1]]
```

DIAGRAMAS ROC EN MODELOS GENERALIZADOS BINOMIALES

Las líneas de código que podemos usar son:

```
## algunas representaciones para entenderlo
plot(cutoff, sensit, ylab="SENSITIVIDAD", xlab="PROBABILIDAD DE CORTE",
     main="true positive rate (TPR)")
plot(cutoff, I(1-especi), ylab="1 - ESPECIFICIDAD", xlab="PROBABILIDAD DE CORTE",
     main="false positive rate (FPR)")
plot(cutoff, acc, ylab="ACCURACY", xlab="PROBABILIDAD DE CORTE",
     main="PROPORCIÓN DE ACIERTOS")
plot(I(1-especi), sensit, ylab="SENSITIVIDAD", xlab="1 - ESPECIFICIDAD", main="CURVA ROC")
abline(a=0, b=1, col="red")

## resultados numéricos
difroc <- abs(sensit-especi)
pcutoff.total <- as.numeric(cutoff[which.min(difroc)])
senesp.total <- (sensit[which.min(difroc)]+especi[which.min(difroc)])/2
acc.total <- acc[which.min(difroc)]
ppv.total <- ppv[which.min(difroc)]
npv.total <- npv[which.min(difroc)]
print(c("PROBABILIDAD DE CORTE OPTIMA =", round(pcutoff.total, 3)), quote=FALSE)
print(c("SENSITIVIDAD (TPR) = ESPECIFICIDAD (FPR) =", round(senesp.total, 3)), quote=FALSE)
print(c("PROPORCIÓN TOTAL PREDICHOS COMO 1 QUE SON 1 =", round(ppv.total, 3)), quote=FALSE)
print(c("PROPORCIÓN TOTAL PREDICHOS COMO 0 QUE SON 0 =", round(npv.total, 3)), quote=FALSE)
print(c("PROPORCIÓN TOTAL DE ACIERTOS =", round(acc.total, 3)), quote=FALSE)
```

estimas robustas

cómo tratar los datos influyentes y/o perdidos

ESTIMAS ROBUSTAS

¿Qué hacer cuando …? nuestros datos presentan observaciones – casos con valores
perdidos (*outliers*) → valorado con dffits, residuos studentizados, distancias de Cook
influyentes → valorado con el Leverage

Bien, esto es incómodo … ¡quitarlos! de nuestro análisis
Esta opción es válida si tenemos criterios objetivos para hacerlo
(confusiones, el dato deriva de una unidad muestral que no pertenece a la población, etc)
Si no tenemos estos criterios objetivos y de "honestidad profesional" esto es una ¡¡ chapuza !!

Alternativamente:
Podemos dar unos pesos a esas observaciones perdidas o influyentes proporcionalmente inversos a lo "aberrantes" o sesgados que son en nuestro modelo de análisis.
Un dato con un residuo "anormalmente" positivo o negativo o con una distancia de Cook grande …
se lo deja en el análisis, pero dándole menos peso
que será tanto menor cuanto mayor es su distancia de Cook
Un dato con un con un valor de influencia muy grande (gran valor de Leverage)
se lo deja en el análisis, pero dándole menos peso
que será tanto menor cuanto mayor es su Leverage

Esto lo podemos abordar con el recálculo de nuestro modelo utilizando **estimas robustas**
… frente a la influencia distorsionadora de esas observaciones muy influyentes o aberrantes

Estimas robustas

Se ajusta el modelo de interés, no por el procedimiento de OLS clásico (ordinary least squares), sino con otro denominado "*iterated re-weighted least squares*" (**IWLS** or **IRwLS**).

Os recomiendo la consulta – estudio de:
https://en.wikipedia.org/wiki/Iteratively_reweighted_least_squares
http://www.ats.ucla.edu/stat/r/dae/rreg.htm
http://www.dst.unive.it/rsr/BelVenTutorial.pdf

La matemática subyacente es compleja, pero en esencia consiste en la búsqueda iterativa de unos "pesos" que, teniendo en cuenta el leverage y la distancia de Cook de las observaciones – casos, consiga que esas observaciones tengan menos influencia distorsionadora de los resultados del modelo pero … ¡sin quitarlos!

Existen varios paquetes en R para poder abordar estimas robustas.
Y cada uno de ellos llevan implícitos diferentes algoritmos de cálculo que podemos elegir.
Nos vamos a centrar en dos paquetes:

```
library(MASS)                ## para  rlm
library(robustbase)          ## para lmrob
library(robustbase)          ## para glmrob
```

Estimas robustas

Podemos contar con **varios procedimientos para encontrar esos "pesos"** de cada observación.

En OLS (panel superior izquierdo) todas las observaciones pesan igual: 1

En los otros, procedimientos se decide pesarlos de manera diferente (e.g., M-estimation, Huber).

El eje X (denominado E en la figura) representa los residuos de las observaciones

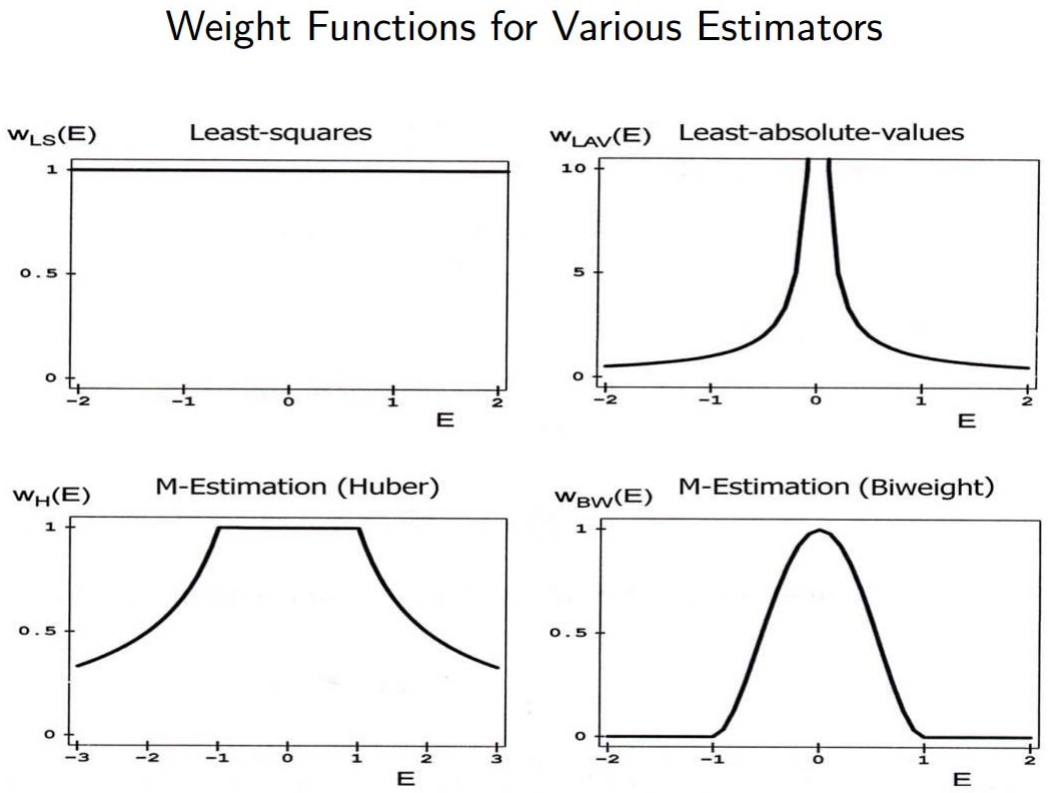


Figure 14.13 from Fox (1997)

Estimas robustas

Para modelos generalizados **glm** podemos contar con **glmrob**.

`weights.on.x="hat"` infrapesa a las observaciones en función de su leverage
`maxit=10` podemos incrementarlo si hay problemas de convergencia

```
eqt <- as.formula(nspp ~ altmed+rangoalt+shannon+tempmin+precip+log(ntransectos))

modelo.original <- glm(eqt, data=datos, family=poisson(link="log"))
modelo.robusto <- glmrob(eqt, data=datos, family=poisson(link="log"),
                         weights.on.x="hat", method="Mqle",
                         control=glmrobMqle.control(tcc=1.2, maxit=100))

summary(modelo.robusto)
coeftest(modelo.robusto, df=modelo.original$df.residual)
Anova(modelo.robusto, type=3, test="Wald")
```

`...` y todo lo demás que hemos visto previamente

Las **medidas de la robustez de los datos individuales** están en

`$w.r`: teniendo en cuenta lo outliers que son las observaciones
`$w.x`: teniendo en cuenta lo extremas que son las observaciones en los valores de las predictoras
el producto de `w.r * w.x` cuantifica el peso global de cada unidad muestral

```
robustez <- modelo.robusto$w.r * modelo.robusto$w.x
plot(robustez, ylab="robustez de las observaciones")
identify(robustez)
```

bootstrapping del modelo

bootstrapping del modelo

Para valorar el efecto de las observaciones influyentes y/o perdidas podemos utilizar *bootstraps*.

En este proceso se extrae, con reemplazo, una muestra de observaciones igual en número al tamaño muestral de nuestros datos.

En cada proceso *bootstrap* habrá unidades muestrales que se repitan y otras que no aparezcan.

Este proceso se repite muchas veces con el objeto de obtener miles de coeficientes obtenidos con "muestras diferentes".

Para modelos **lm**, **glm**, **glm.nb**, **betareg**, **gls**, **hurdle**
Poner el número de simulaciones en R=...
Con system.time(...) podemos saber cuánto dura el proceso

```
system.time( mi.bootstrap <- Boot(modelo, f=coef, R=1000, method="case") )
boot.modelo <- as.data.frame(mi.bootstrap$t)
```

Resumen de los resultados

original son los coeficientes originales del modelo; **bootMed** es la media de los bootstraps
bootBias es la diferencia entre el coeficiente medio remuestreado y el original
bootSE es la SD de los bootstraps, que en este caso equivale al error estandard (SE)
bootSkew y **bootKurtosis** son el sesgo (Ho=0) y kurtosis (Ho=0) de los bootstraps
si el sesgo y kurtosis tienen a cero, se puede ajustar una distribución normal a los valores del bootstrap

```
summary(mi.bootstrap, high.moments=TRUE)
```

bootstrapping del modelo

INTERVALOS DE CONFIANZA NUMERICOS

Que los intervalos siguientes no incluyan el valor "cero"

Intervalos ETI (Equal-Tailed Interval)

intervalos de confianza por el método que asume que los coeficientes siguen una distribución normal a dos niveles de probabilidad (95% y 99%)

```
confint(mi.bootstrap, level=c(0.95, 0.99), type="norm")
```

intervalos de confianza por el método de los percentiles que no asume distribución normal

```
confint(mi.bootstrap, level=c(0.95, 0.99), type="perc")
```

intervalos de confianza por el método Bias Corrected Accelerated
(tiende a proporcionar intervalos más ajustados y estrechos)
la ventaja de este método sobre el de los percentiles es que corrige el bias y el sesgo de la distribución de los coeficientes del bootstrap; usados para ajustar los límites de confianza de los precentiles
https://blogs.sas.com/content/iml/2017/07/12/bootstrap-bca-interval.html

```
confint(mi.bootstrap, level=c(0.95, 0.99), type="bca")
```

Intervalos HDI (Highest Density Interval)

https://www.sciencedirect.com/topics/mathematics/highest-density-interval

```
library(bayestestR)
options(scipen = 999)
print(data.frame(ci(boot.modelo, method="HDI", ci=0.95))[,1:4], digits=3)
options(scipen = 0)
summary(mi.bootstrap, high.moments=TRUE)
```