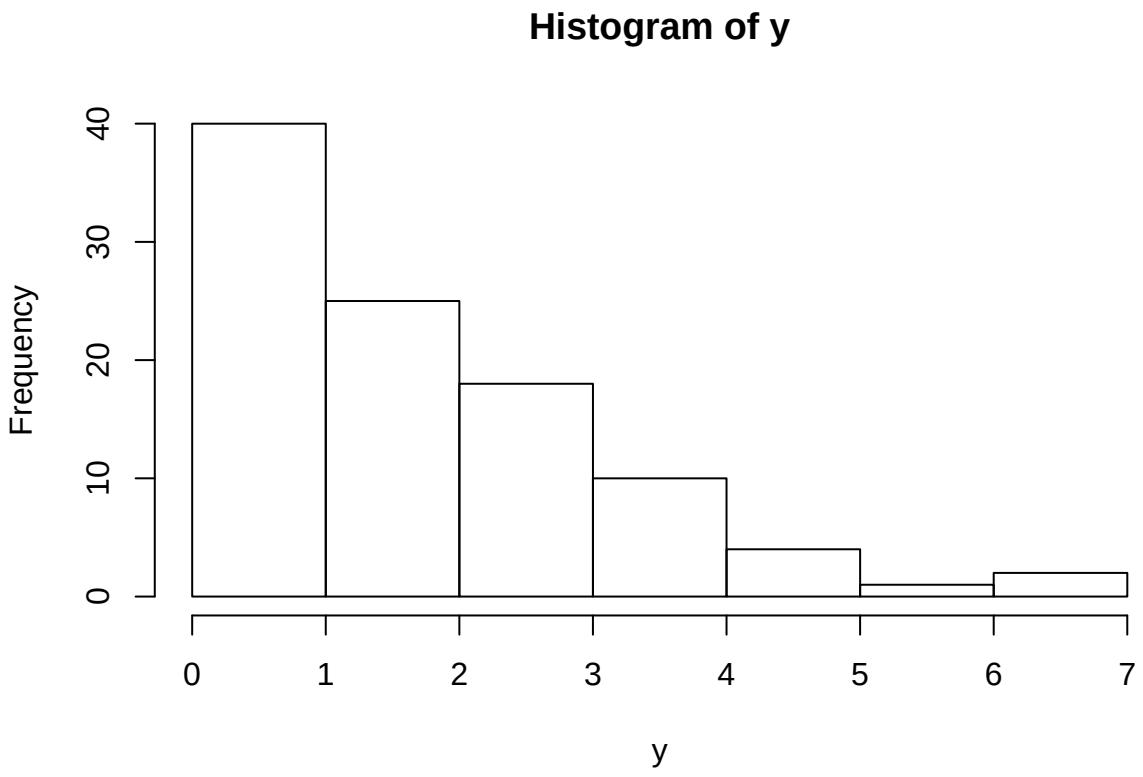
Aplicaciones de modelos lineales generalizados

Regresión Gamma

```
set.seed(999)
N <- 100
x <- runif(N, -1, 1)
a <- 0.5
b <- 1.2
mu_true <- exp(a + b * x)
shape <- 10
log(shape)
```

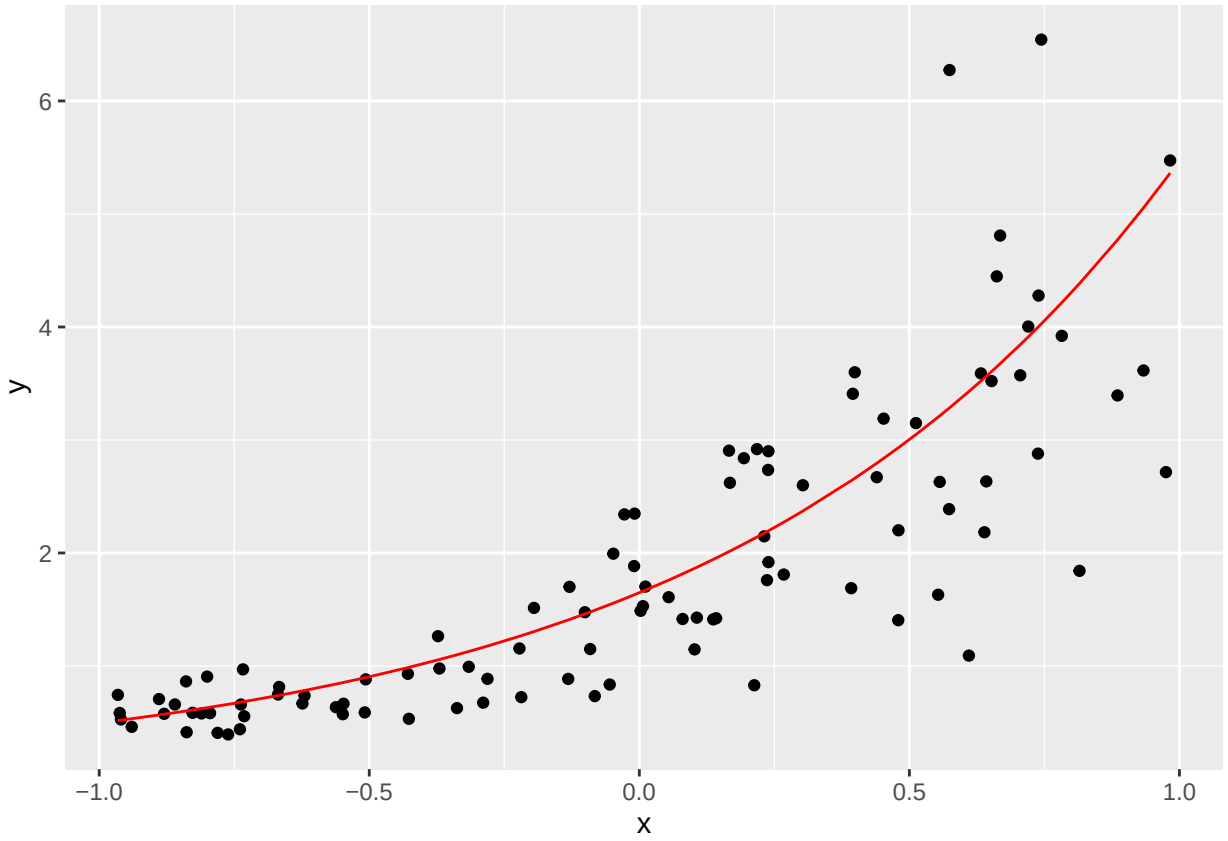
```
## [1] 2.302585
```

```
y <- rgamma(N, rate = shape / mu_true, shape = shape)

hist(y)

library(tibble)
library(ggplot2)
```




```
newdf <- tibble(
  y = mu_true[order(x)],
  x = sort(x)
)
```

```r
data <- data.frame(y, x)
ggplot(data, aes(x=x, y=y)) +
  geom_jitter(height=0.0) +
  geom_line(aes(x=x, y=y), col='red', data=newdf)
```




```r
m_glm <- glm(y ~ x, family = Gamma(link = "log"))
summary(m_glm)
```

```
##
## Call:
## glm(formula = y ~ x, family = Gamma(link = "log"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9007  -0.2307  -0.0210   0.1797   0.8326
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.43559    0.03256   13.38   <2e-16 ***
## x            1.16522    0.05789   20.13   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.1057873)
##
##     Null deviance: 50.968  on 99  degrees of freedom
## Residual deviance: 10.768  on 98  degrees of freedom
```

```
## AIC: 139.12
##
## Number of Fisher Scoring iterations: 4
```

```
m_glm_ci <- confint(m_glm)
```

```
## Waiting for profiling to be done...
```

```
coef(m_glm)
```

```
## (Intercept)           x
##   0.4355899   1.1652181
```

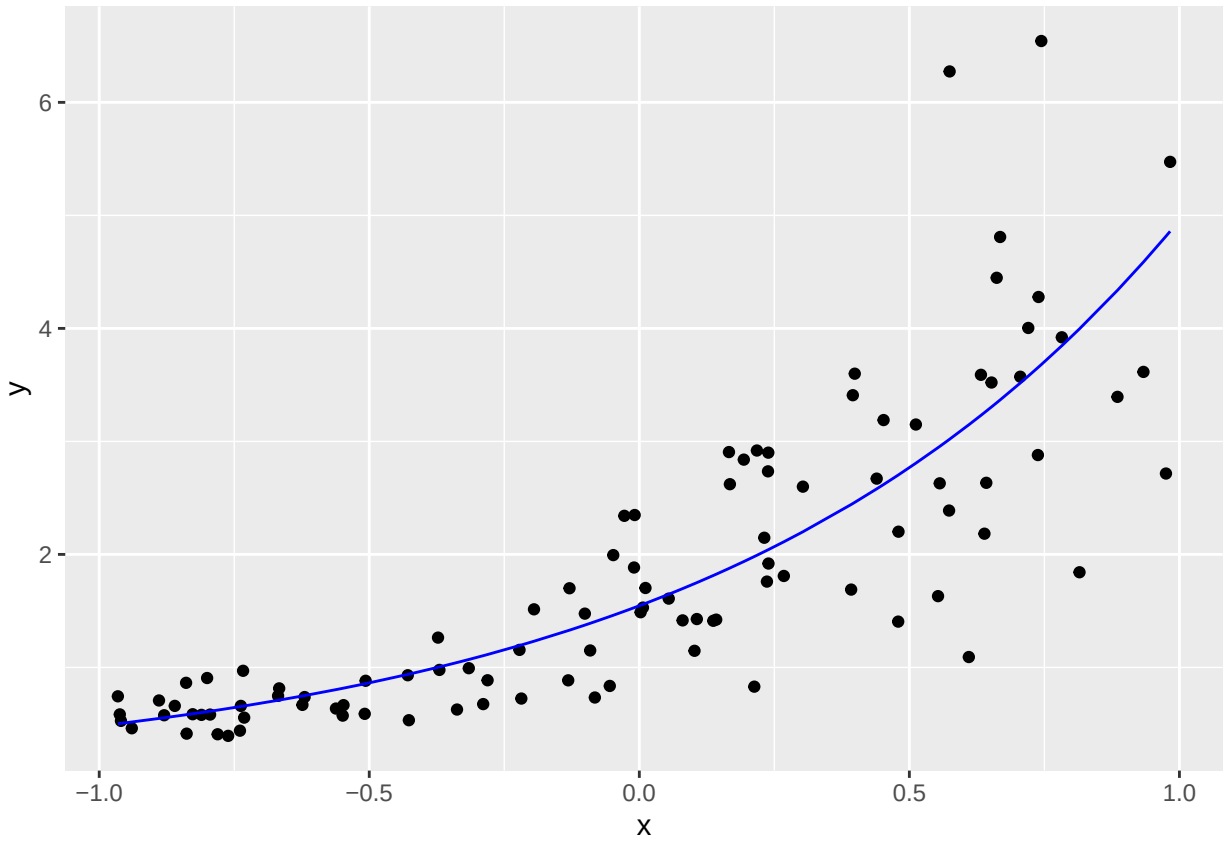
```
# dispersion parameter
shape
```

```
## [1] 10
```

```
shape_est = 1/0.1057
shape_est
```

```
## [1] 9.460738
```

```
xpred = x
ypred = predict(m_glm, newdata=data.frame(x=xpred), type='response')



newdf4 <- tibble(
  y = ypred[order(xpred)],
  x = sort(xpred)
)


ggplot(data, aes(x=x, y=y)) +
  geom_jitter(height=0.0) +
  geom_line(aes(x=x, y=y), col='blue', data=newdf4)
```




```
library("bbmle")
```

```
## Loading required package: stats4
```

```
nll_gamma <- function(a, b, logshape) {
  linear_predictor <- a + b * x
  # rate = shape / mean:
  rate <- exp(logshape) / exp(linear_predictor)
  # sum of negative log likelihoods:
  -sum(dgamma(y, rate = rate, shape = exp(logshape),
      log = TRUE))
}

m_mle2 <- bbmle::mle2(nll_gamma,
  start = list(a = rnorm(1), b = rnorm(1),
    logshape = rlnorm(1)))

m_mle2
```

```
##
## Call:
## bbmle::mle2(minuslogl = nll_gamma, start = list(a = rnorm(1),
##     b = rnorm(1), logshape = rlnorm(1)))
##
## Coefficients:
##         a         b  logshape
## 0.4355901 1.1652181 2.2460913
##
```

```
## Log-likelihood: -66.55
```

```
log(shape)
```

```
## [1] 2.302585
```

```
m_mle2_ci <- confint(m_mle2)
m_mle2_ci
```

```
##              2.5 %    97.5 %
## a        0.3718551 0.5007119
## b        1.0506909 1.2794049
## logshape 1.9612370 2.5069892
```

Aplicacion a Datos:

Experimento realizado para evaluar el desempeño de cinco tipos de turbinas de alta velocidad para motores de avión. Fueron considerados 10 motores para cada turbina y registrado el tiempo (en unidades de millones de ciclos) hasta la perdida de velocidad.

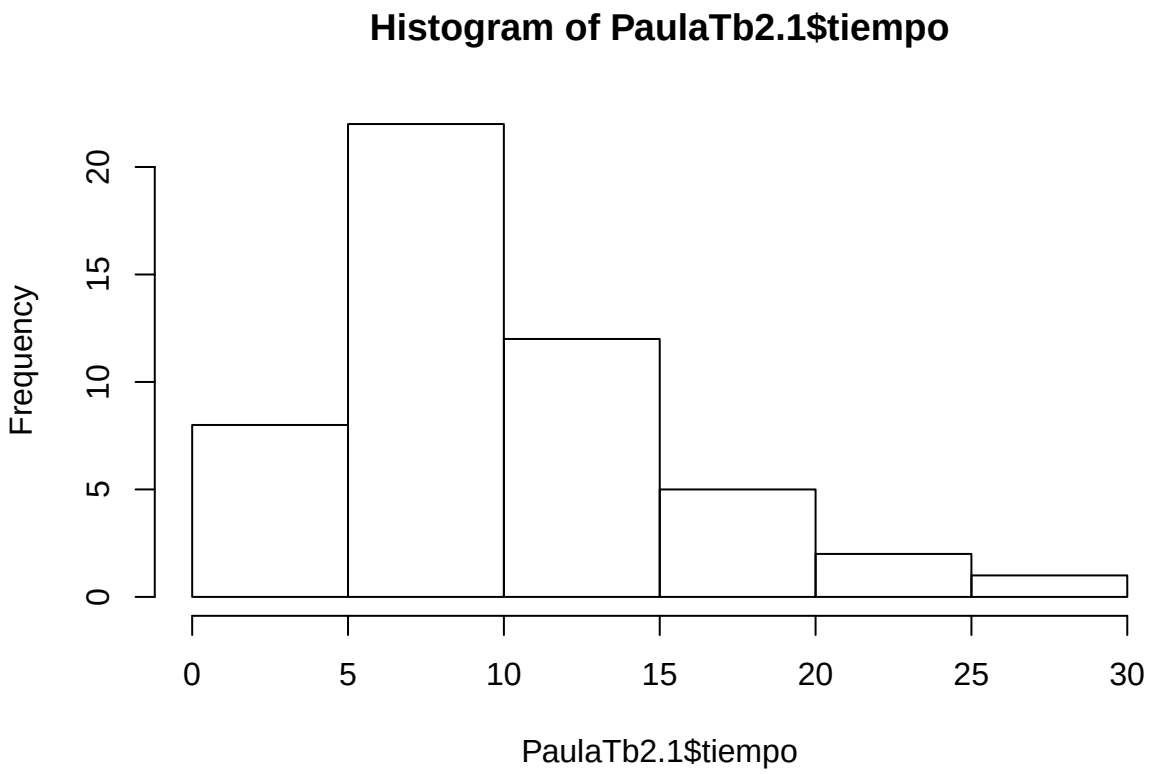
```
require(MASS)
```

```
## Loading required package: MASS
```

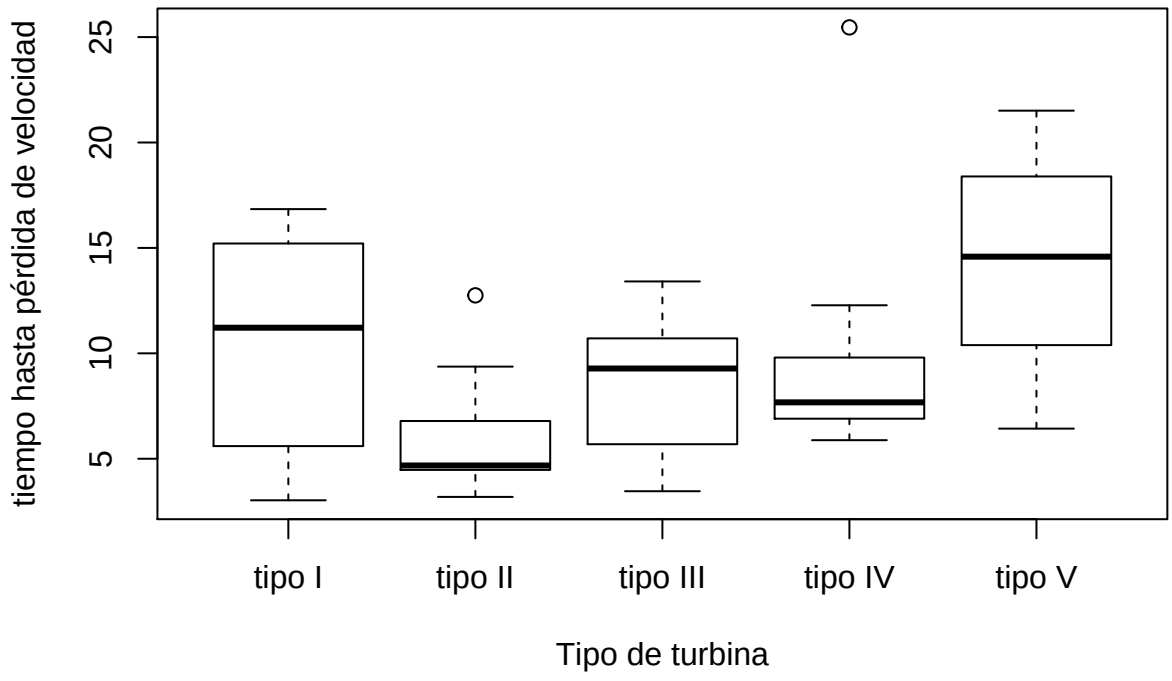
```
#require(labestData)
#data(PaulaTb2.1)

PaulaTb2.1 <- read.csv('/media/zaidajqc/f21961f4-c511-41f4-984a-1c0c989fe427/zaidajqc/Documents/2019-2/r

head(PaulaTb2.1)
```

```
##     turb tiempo
## 1 tipo I   3.03
## 2 tipo I   5.53
## 3 tipo I   5.60
## 4 tipo I   9.30
## 5 tipo I   9.92
## 6 tipo I  12.51
```

```
### Análisis descriptivo

hist(PaulaTb2.1$tiempo)
```




```
with(PaulaTb2.1, boxplot(tiempo ~ turb, xlab='Tipo de turbina',
                         ylab='tiempo hasta pérdida de velocidad'))
```



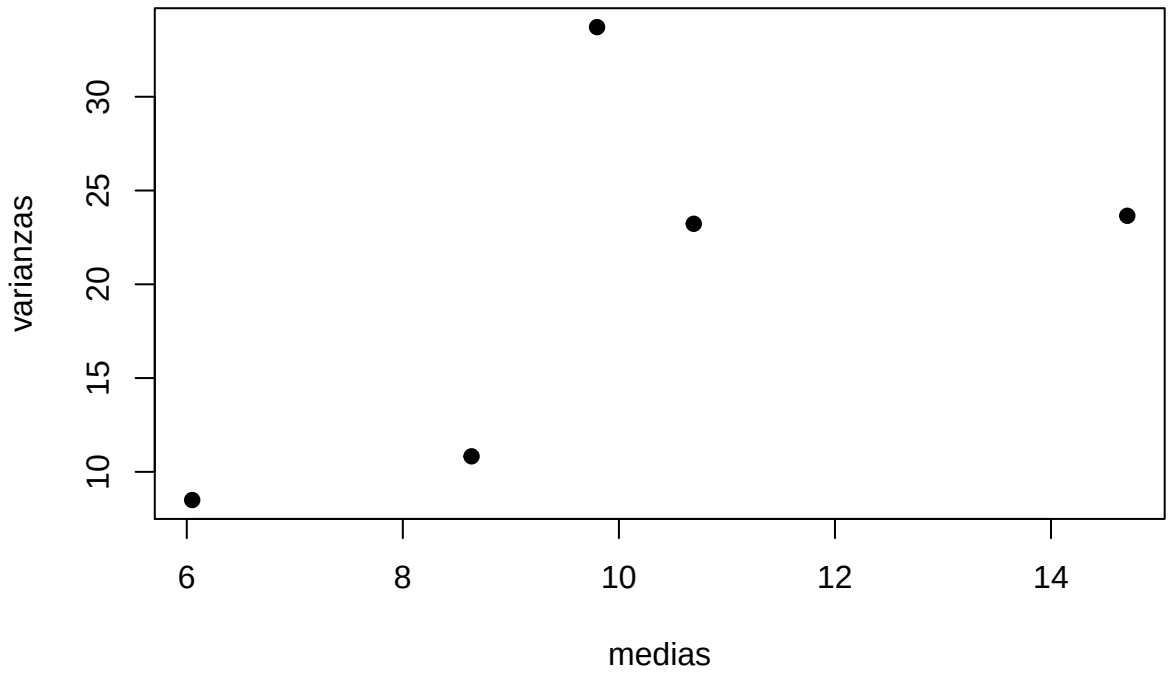

```
medias <- with(PaulaTb2.1, tapply(tiempo,turb,mean)); medias
```

```
##   tipo I  tipo II tipo III  tipo IV   tipo V
##   10.693    6.050    8.636    9.798   14.706
```

```r
varianzas <- with(PaulaTb2.1, tapply(tiempo,turb,var)); varianzas
```

```
##    tipo I   tipo II  tipo III   tipo IV    tipo V
## 23.225512  8.497489 10.828116 33.711796 23.652316
```

```r
plot(medias, varianzas, pch=20, cex=1.5)
```




```r
cvs <- sqrt(varianzas)/medias; cvs # Coeficientes de variacion.
```

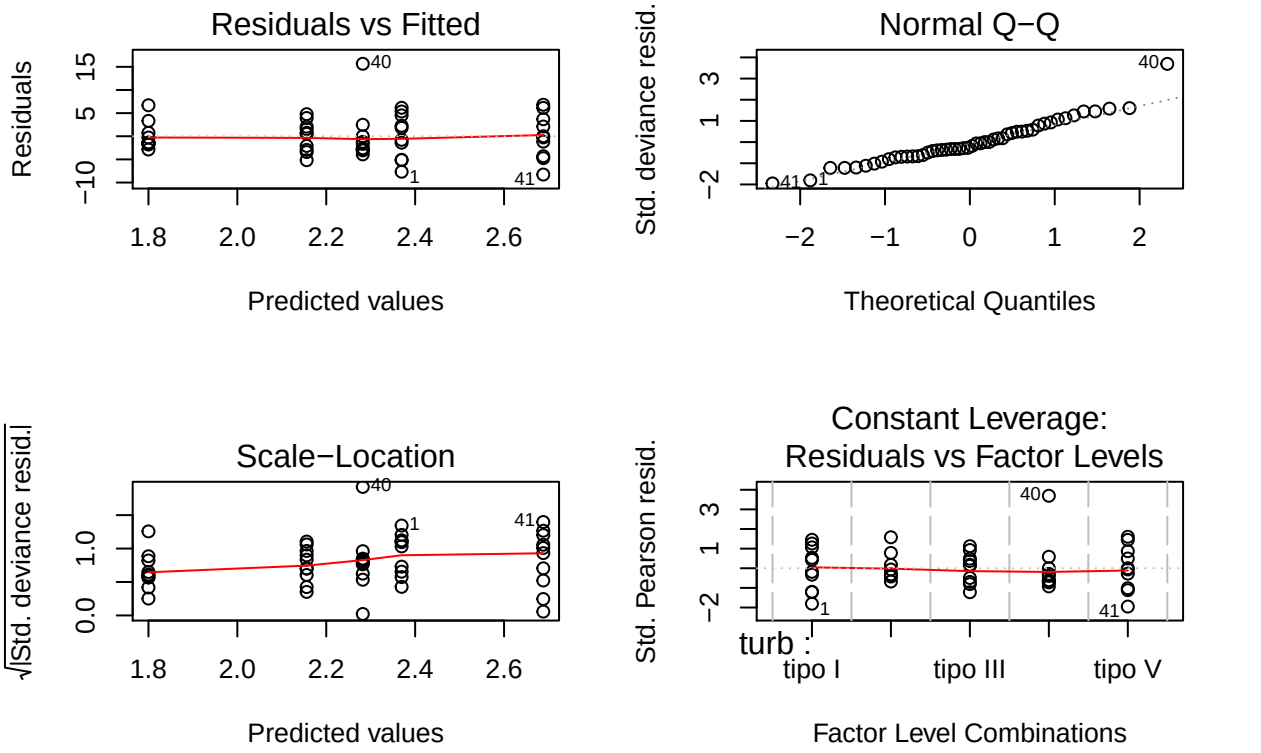
```
##    tipo I   tipo II  tipo III   tipo IV    tipo V
## 0.4506954 0.4818257 0.3810341 0.5925889 0.3307061
```

```r
### Ajuste 0: modelo normal (con función de enlace logaritmica):
ajuste0 <- glm(tiempo ~ turb, data = PaulaTb2.1,family=gaussian(link="log"))
summary(ajuste0)
```

```
##
## Call:
## glm(formula = tiempo ~ turb, family = gaussian(link = "log"),
##     data = PaulaTb2.1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -8.2760  -2.8495  -0.9645   2.2187  15.6620
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.36959    0.13220  17.924   <2e-16 ***
## turbtipo II  -0.56953    0.26846  -2.121   0.0394 *
## turbtipo III -0.21365    0.21041  -1.015   0.3153
## turbtipo IV  -0.08741    0.19568  -0.447   0.6572
## turbtipo V    0.31867    0.16345   1.950   0.0575 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for gaussian family taken to be 19.98312)
##
##     Null deviance: 1300.51  on 49  degrees of freedom
## Residual deviance:  899.24  on 45  degrees of freedom
## AIC: 298.37
##
## Number of Fisher Scoring iterations: 5
```

```
par(mfrow = c(2,2))
plot(ajuste0)
```




```
AIC(ajuste0)
```

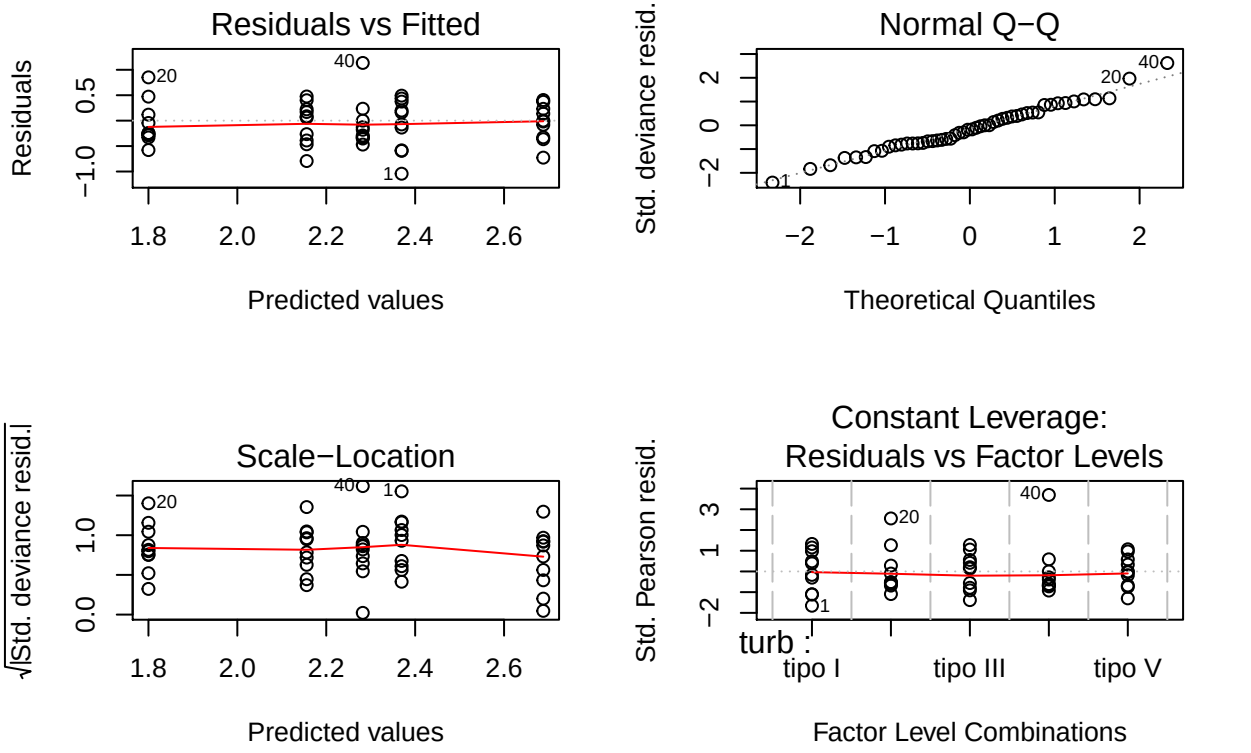
```
## [1] 298.37
```

```
### Ajuste 1: modelo gamma (con función de enlace logaritmica, para comparación):

ajuste1 <- glm(tiempo ~ turb, family = 'Gamma'(link = 'log'), data = PaulaTb2.1)
summary(ajuste1)
```

```
##
## Call:
## glm(formula = tiempo ~ turb, family = Gamma(link = "log"), data = PaulaTb2.1)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.04345  -0.33058  -0.07744   0.21689   1.13451
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.36959    0.14429  16.422  < 2e-16 ***
```

```
## turbtipo II  -0.56953    0.20406  -2.791  0.00768 **
## turbtipo III -0.21365    0.20406  -1.047  0.30069
## turbtipo IV  -0.08741    0.20406  -0.428  0.67043
## turbtipo V    0.31867    0.20406   1.562  0.12538
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.2081995)
##
##     Null deviance: 12.9654  on 49  degrees of freedom
## Residual deviance:  8.8616  on 45  degrees of freedom
## AIC: 285.91
##
## Number of Fisher Scoring iterations: 4
```

```
x11()
par(mfrow = c(2,2))
plot(ajuste1)
```




```
AIC(ajuste1)
```

```
## [1] 285.9131
```

Estimación del parámetro de dispersión.

En base a la devianza.

$$\hat{\phi} = \frac{D_p}{n-p}$$

$D = (\phi)^{-1} D_p$

En base a la estadistica X2 de Pearson.

$$\hat{\phi} = \frac{1}{n-p}\sum_{i}^{n}\frac{(y_i - \hat{\mu}_i)^2}{V(\hat{\mu}_i)} = \frac{1}{n-p}\sum_{i}^{n} e_i^{P2}$$

Por máxima verosimilitud.

Sea $\theta = (\beta, \phi)$,

$$\begin{pmatrix} \hat{\beta} \\ \hat{\phi} \end{pmatrix} \sim N\left[\begin{pmatrix} \beta \\ \phi \end{pmatrix}, I^{-1}(\theta)\right]$$

donde

$$I(\theta) = \begin{pmatrix} I_{\beta,\beta} & I_{\beta,\phi} \\ I_{\phi,\beta} & I_{\phi,\phi} \end{pmatrix}$$

$$I_{\beta,\beta} = E(-\frac{d^2(l)}{d\beta d\beta^t})$$

$$I_{\beta,\phi} = E(-\frac{d^2(l)}{d\beta d\phi})$$

```
### Vamos a estimar el parámetro de dispersión.
### En base a la  devianza.
estimat1 <- ajuste1$deviance/ajuste1$df.residual
estimat1
```

```
## [1] 0.1969237
```

```
### En base a la  estadistica X2 de Pearson.
estimat2 <- sum(residuals(ajuste1,type='pearson')**2)/ajuste1$df.residual
estimat2
```

```
## [1] 0.2081995
```

```
### Por máxima verosimilitud.
estimat3 <- gamma.dispersion(ajuste1) #  función de MASS
estimat3
```

```
## [1] 0.1722981
```

```
#shape = alpha = 1/phi
1/estimat3
```

```
## [1] 5.803896
```

```
# la función gamma.shape calcula directamente alpha
estimat4 <- gamma.shape(ajuste1) #  función de MASS
estimat4
```

```
##
## Alpha: 5.803896
## SE:    1.128995
```

```
summary(ajuste1)
```

```
##
## Call:
## glm(formula = tiempo ~ turb, family = Gamma(link = "log"), data = PaulaTb2.1)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.04345  -0.33058  -0.07744   0.21689   1.13451
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.36959    0.14429  16.422  < 2e-16 ***
## turbtipo II  -0.56953    0.20406  -2.791  0.00768 **
## turbtipo III -0.21365    0.20406  -1.047  0.30069
## turbtipo IV  -0.08741    0.20406  -0.428  0.67043
## turbtipo V    0.31867    0.20406   1.562  0.12538
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 0.2081995)
##
##     Null deviance: 12.9654  on 49  degrees of freedom
## Residual deviance:  8.8616  on 45  degrees of freedom
## AIC: 285.91
##
## Number of Fisher Scoring iterations: 4
```

Interpretación de variable explicativa:

La turbina II tiene tiempo medio de vida inferior que la turbina I

La media esperada del tiempo en que pierde velocidad una turbina del tipo II es $\exp(-0.56953) = 0.5657913$ veces la media esperada del tiempo en que pierde velocidad una turbina del tipo I. Es decir la media esperada del tiempo en que pierde velocidad la turbina II es (1-0.57)*100% = 43% menor que de la turbina de tipo I.

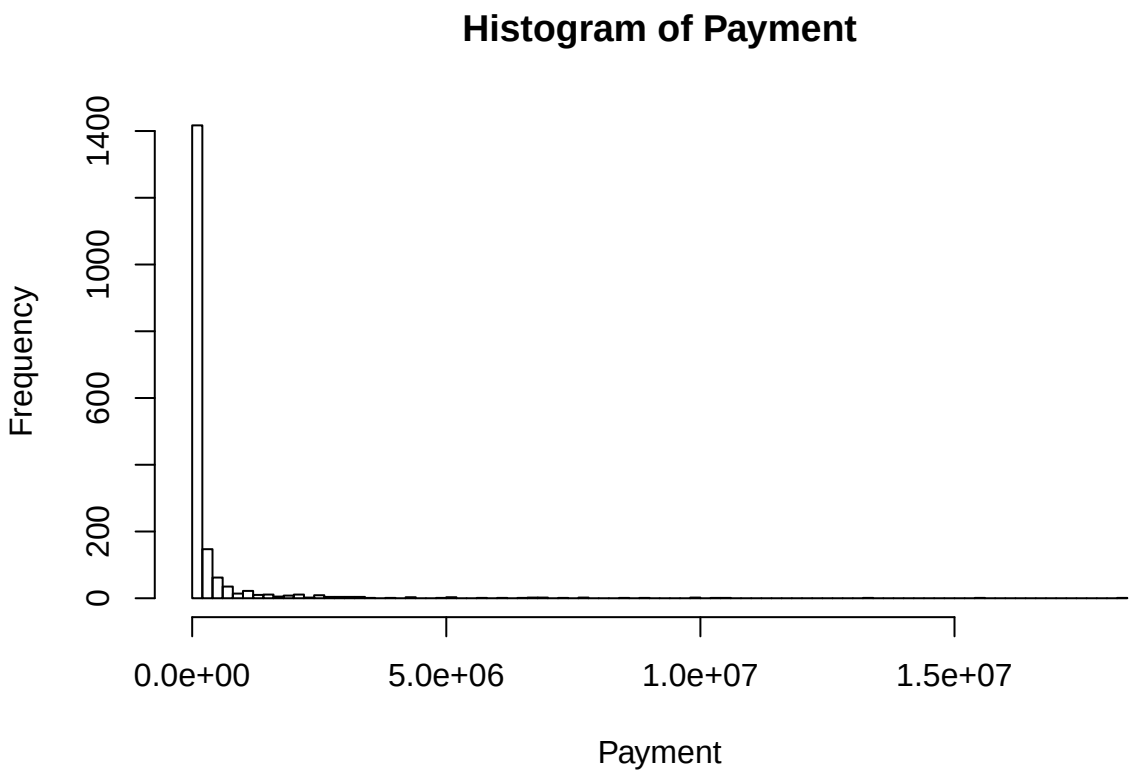
Ejemplo 2:

```
library(faraway)
# Third party motor insurance claims in Sweden in 1977
#En Suecia, todas las compañías de seguros de automóviles aplican
#argumentos de riesgo idénticos para clasificar a los clientes y,
#por lo tanto, sus carteras y sus estadísticas de reclamos se pueden
#combinar. Los datos fueron compilados por un Comité sueco sobre el
#análisis de la prima de riesgo en el seguro de automóviles. Se le pidió
#al comité que analizara el problema de analizar la influencia real en
#los reclamos de los argumentos de riesgo y que comparara esta
#estructura con la tarifa real.

#Payment
# valor total de pagos en u.m. Skr

#Kilometres
#factor ordenado representando los kilometros recorridos por año
```

```
#con niveles 1: < 1000, 2: 1000-15000, 3: 15000-20000, 4: 20000-25000,
#5: > 25000

# Zone
# factor representando la zona geografica
# con nivel 1: Stockholm, Goteborg, Malmo y alrededores
#... 7: Gotland

#Bonus
# Sin bonificación de reclamos. Igual al número de años,
#más uno, desde el último reclamo

#Make
#Un factor que representa ocho modelos diferentes de automóviles
#comunes. Todos los demás modelos se combinan en la clase 9.


#Insured
#Número de asegurados en años póliza (policy-years)

#Claims
#número de reclamos


#perd
#Pago por reclamo


#leyendo los datos
data(motorins)

head(motorins)
```

```
##   Kilometres Zone Bonus Make Insured Claims Payment     perd
## 1          1    1     1    1  455.13    108  392491 3634.176
## 2          1    1     1    2   69.17     19   46221 2432.684
## 3          1    1     1    3   72.88     13   15694 1207.231
## 4          1    1     1    4 1292.39    124  422201 3404.847
## 5          1    1     1    5  191.01     40  119373 2984.325
## 6          1    1     1    6  477.66     57  170913 2998.474
```

```
# adjuntando los datos 'motorins':

attach(motorins)

# Configuracion inicial:
# solo escogemos los datos de la zona 1 para el análisis.

motori <- motorins[motorins$Zone == 1,]


hist(Payment,breaks=100)
```




```
# Ajustando el modelo gamma con funcion de enlace logaritmica

gamma.motor2 <- glm(Payment ~ + as.numeric(Kilometres) +
                      Make + Bonus, family = Gamma(link=log), motori)
# "link=log" debe ser especificado, porque la función de
#enlace inversa es la que glm tiene por defecto para una familia gamma
summary(gamma.motor2)
```

```
##
## Call:
## glm(formula = Payment ~ +as.numeric(Kilometres) + Make + Bonus,
##     family = Gamma(link = log), data = motori)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3248  -1.0242  -0.3845   0.3380   2.4312
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             12.90091    0.24144  53.434  < 2e-16 ***
## as.numeric(Kilometres) -0.43270    0.04231 -10.227  < 2e-16 ***
## Make2                  -0.85886    0.24211  -3.547 0.000455 ***
## Make3                  -1.06274    0.24389  -4.357 1.84e-05 ***
## Make4                  -1.42012    0.26012  -5.460 1.04e-07 ***
## Make5                  -1.44897    0.24582  -5.894 1.06e-08 ***
## Make6                  -1.22931    0.24211  -5.078 6.93e-07 ***
## Make7                  -2.00854    0.24782  -8.105 1.58e-14 ***
## Make8                  -1.84759    0.25436  -7.264 3.64e-12 ***
## Make9                   2.16575    0.24211   8.945  < 2e-16 ***
## Bonus                   0.16555    0.02920   5.668 3.54e-08 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 1.025779)
##
##     Null deviance: 1080.29  on 294  degrees of freedom
## Residual deviance:  312.61  on 284  degrees of freedom
## AIC: 7399.3
##
## Number of Fisher Scoring iterations: 13
```

```
gamma.motor0 <- glm(Payment ~  as.numeric(Kilometres)  ,
                    family = Gamma(link=log), motori)
summary(gamma.motor0)
```

```
##
## Call:
## glm(formula = Payment ~ as.numeric(Kilometres), family = Gamma(link = log),
##     data = motori)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2275  -1.9053  -1.3200  -0.6492   6.0468
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              13.8919     0.3726  37.286  < 2e-16 ***
## as.numeric(Kilometres)  -0.4235     0.1147  -3.694 0.000263 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 7.606834)
##
##     Null deviance: 1080.29  on 294  degrees of freedom
## Residual deviance:  993.01  on 293  degrees of freedom
## AIC: 7812.1
##
## Number of Fisher Scoring iterations: 9
```

```
anova(gamma.motor2,gamma.motor0,test='F')
```

```
## Analysis of Deviance Table
##
## Model 1: Payment ~ +as.numeric(Kilometres) + Make + Bonus
## Model 2: Payment ~ as.numeric(Kilometres)
##   Resid. Df Resid. Dev Df Deviance      F    Pr(>F)
## 1       284     312.61
## 2       293     993.01 -9  -680.41 73.701 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#Interpretación:
exp(0.12007)
```

```
## [1] 1.127576
```

```r
#La media esperada de pago  se incrementa en
#(1.127576-1)*100% = 12% por cada km adicional.

#### Predicciones:
# Prediccion para el pago de un individuo con
# Make="1", Kilometres=1, Bonus=1, Insured=100 :

# Bajo el modelo gamma model:

x0 <- data.frame(Make="1", Kilometres=1, Bonus=1, Insured=100)
pred <- predict(gamma.motor2, new=x0, se=T, type="response")

pred$fit
```

```
##        1
## 306740.7
```