CheatSheet

Models lineals

Prèvia

LLibreries que acostumem a incloure encara que no sabem exactament què fan.

```
library(emmeans)
library(car)
library(RcmdrMisc)
library(tables)
```

Basics

Carregar les dades del fitxer data.csv i plantejar un model.

```
setwd('.')
dades<-read.csv2("./data.csv")
# dades<-read.csv2("~/uni/2n/prob2/exercicis r/comrect.csv")
# Assegurar-se que les columnes que han de ser factors ho son
dades$Any<-as.factor(dades$Any)
dades$M<-as.factor(dades$M)

# per comprovar-ho:
is.factor(dades$Any)
```

```
## [1] TRUE
```

```
# plantejar un model
model = lm(formula=Y~X, data=dades)
```

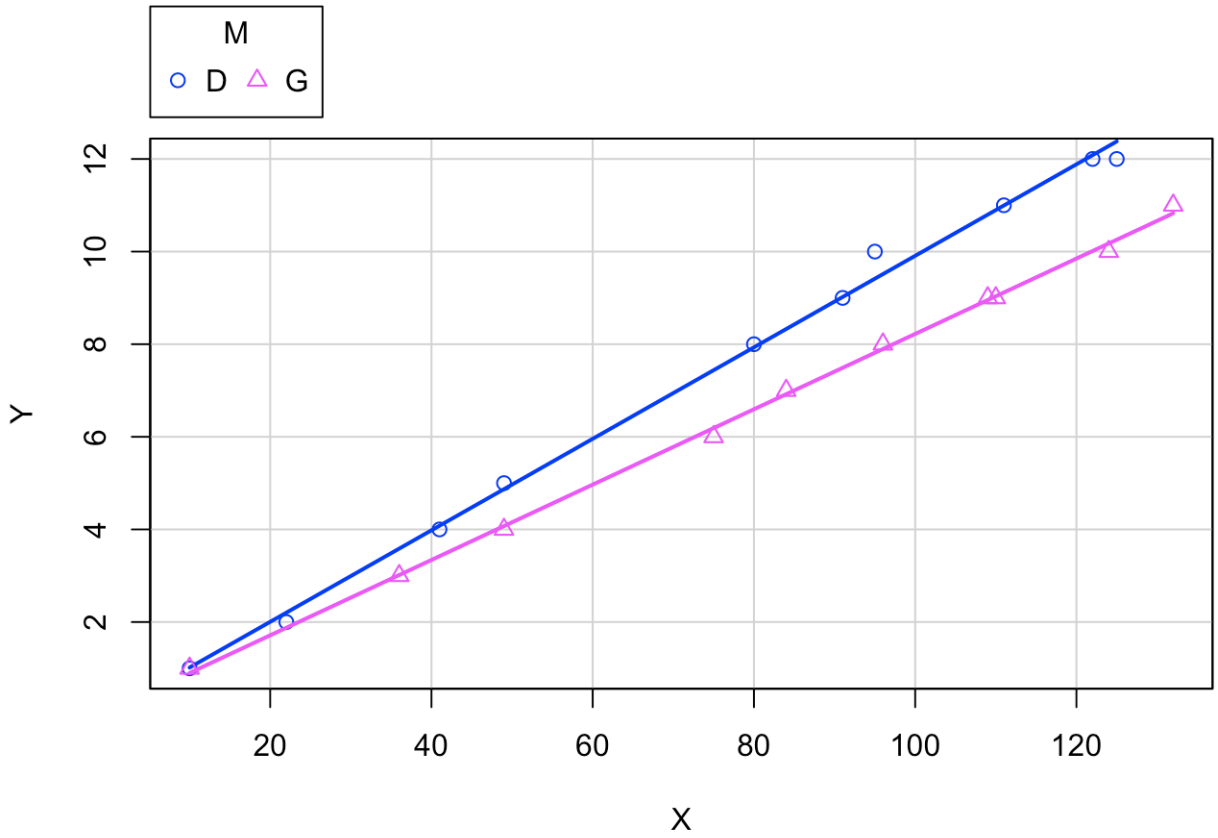
Exemples de fórmules per diferents models:

```
f0 = Y ~ 1 # model null
f1 = Y ~ X # regresió simple
f2 = Y ~ X1 + X2 # model aditiu
f22= Y ~ X + M # model aditiu però una variable és un factor en realitat
f3 = Y ~ X * M # regresió simple amb un factor o model factorial
```

Per veure ràpidament les dades:

Funció scatterplot per les rectes de regressió
scatterplot(formula(factorial), data) ploteja la variable resposta vs la variable en una recta per cada factor. Si la fórmula és d'un model aditiu es veu només una recta. Aquí és on hem d'observar linealitat per complir les hipòtesis dels models lineals.

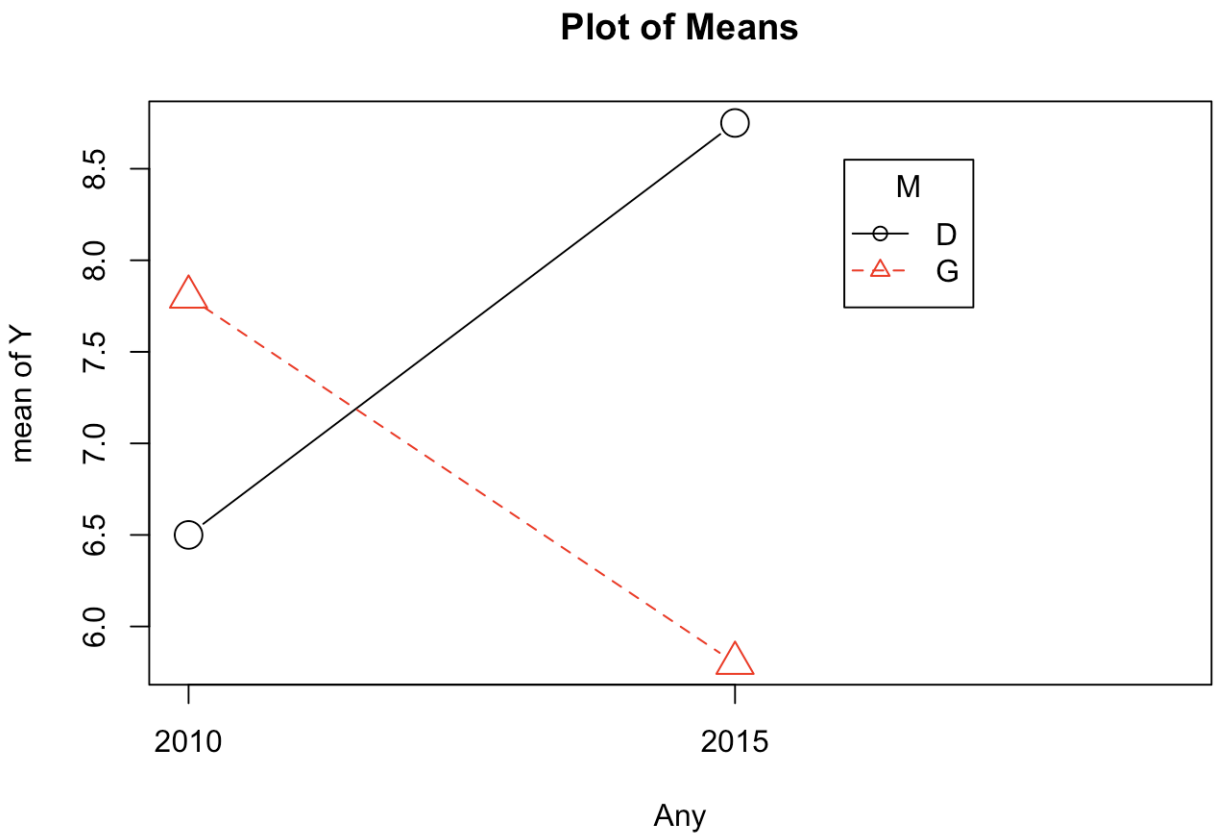
```
scatterplot(Y~X*M,smooth=F, boxplot = F,data=dades)
```



Les mitjanes per diferents factors

with(data, plotMeans(response, factor1, factor2)) fa la mitjana per cada combinació dels dos factors 1 i 2.

```
with(dades,plotMeans(response=Y,factor1=Any,factor2=M,error.bars="none",level=
0.95))
```



EMMeans

EMM (estimated marginal means) calcula les mitjanes marginals per alguns factors especificats a *specs* Podem veure quins factors són estadísticament diferents (amb un nivell de significació donat) visualment amb pwpp(em) si no estan units per la línia. Tenim funcions interessants com pairs(em) que fan un anàlisis de *emm* per parells. Calcula .

```
dades_emm <- read.csv2("./gmd.csv")
m <- lm(GMD~DOSI,dades_emm)

(emm<-emmeans(m,~DOSI))
```

```
##  DOSI emmean   SE df lower.CL upper.CL
##  D00     196 4.01 20      188      205
##  D08     200 4.01 20      192      209
##  D15     224 4.01 20      216      233
##  D20     231 4.01 20      223      239
##  D30     231 4.01 20      223      239
##
## Confidence level used: 0.95
```

```
pairs(emm)
```

```
##  contrast  estimate   SE df t.ratio p.value
##  D00 - D08    -3.80 5.67 20 -0.671  0.9604
##  D00 - D15   -27.85 5.67 20 -4.916  0.0007
##  D00 - D20   -34.71 5.67 20 -6.127  <.0001
##  D00 - D30   -34.68 5.67 20 -6.122  <.0001
##  D08 - D15   -24.05 5.67 20 -4.244  0.0032
##  D08 - D20   -30.91 5.67 20 -5.456  0.0002
##  D08 - D30   -30.88 5.67 20 -5.451  0.0002
##  D15 - D20    -6.87 5.67 20 -1.212  0.7447
##  D15 - D30    -6.84 5.67 20 -1.207  0.7477
##  D20 - D30     0.03 5.67 20  0.005  1.0000
##
## P value adjustment: tukey method for comparing a family of 5 estimates
```

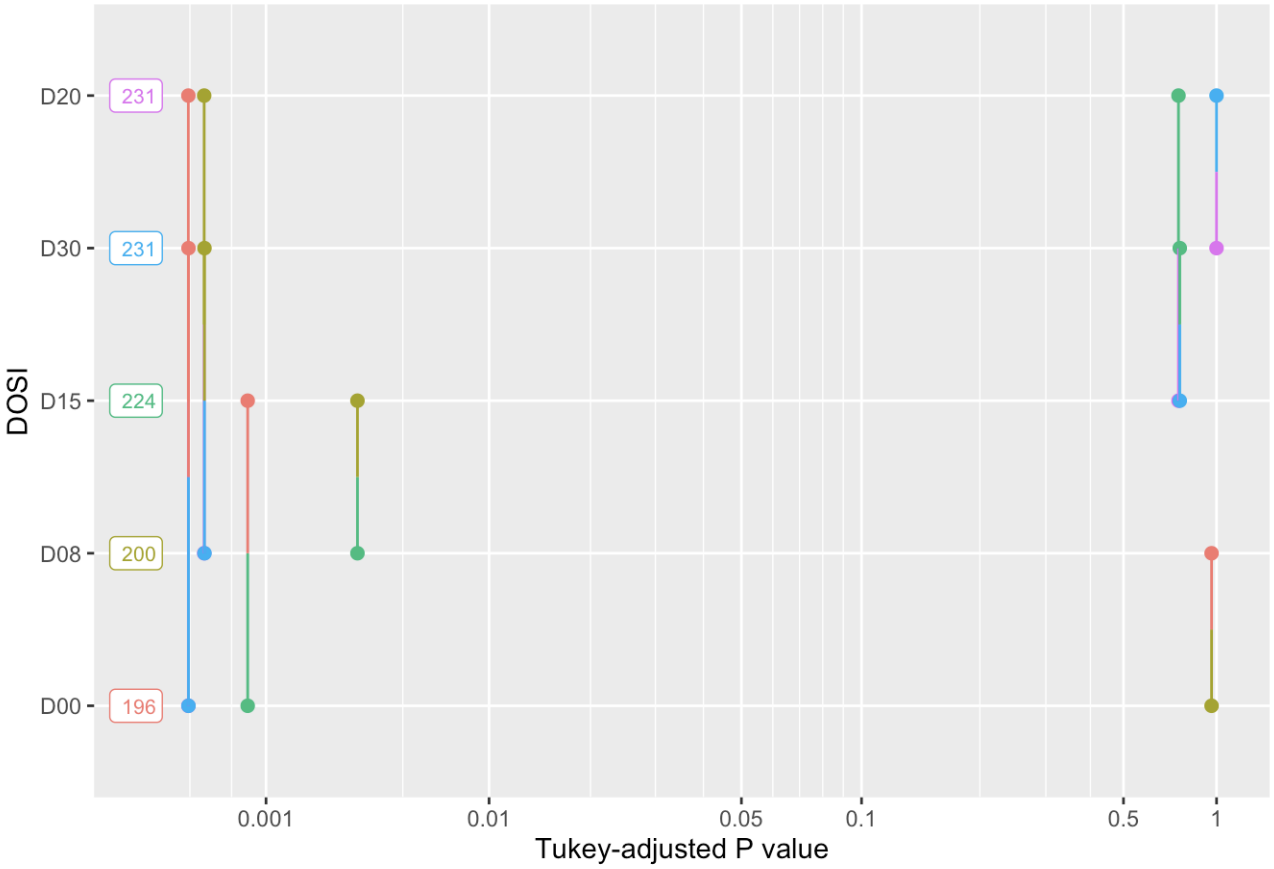
```
CLD(emm)
```

```
## Warning: 'CLD' will be deprecated. Its use is discouraged.
## See '? CLD' for an explanation. Use 'pwpp' or 'multcomp::cld' instead.
```

```
##  DOSI emmean   SE df lower.CL upper.CL .group
##  D00     196 4.01 20      188      205  1
##  D08     200 4.01 20      192      209  1
##  D15     224 4.01 20      216      233   2
##  D30     231 4.01 20      223      239   2
##  D20     231 4.01 20      223      239   2
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 5 estimates
## significance level used: alpha = 0.05
```

La funció *pwpp()* amb el paràmetre *adjust="tukey"* ens fa una gràfica que relaciona cada parell de factors i diu si són estadísticament diferents per un nivell de significació donat en l'eix x.

```
pwpp(emm, adjust="tukey")
```



En aquest cas diríem que donat un nivell de significació del 0.01 són diferents si estan units, per tant D15 i D08, D00 i D15, D00 amb tots menys amb D08…

Comprovacions

Test Anova i Omnibus

Per fer les taules anova utilitzem la funció anova(model). Tenim tipus I, II, III i la més comú i recomenable és el tipus II (Anova()). Ens indica la significació de cada paràmetre (si cada paràmetre sol és igual a zero o no) i l'estadístic de contrast (F value)

```
mod = lm(Y~X*M, dades)
m0d = lm(Y~1, dades)

Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: Y
##             Sum Sq Df  F value    Pr(>F)
## X          245.034  1 5646.111 < 2.2e-16 ***
## M            8.549  1  196.995 2.061e-10 ***
## X:M          2.272  1   52.341 1.993e-06 ***
## Residuals    0.694 16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(mod,m0d)
```

```
## Analysis of Variance Table
##
## Model 1: Y ~ X * M
## Model 2: Y ~ 1
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1     16   0.694
## 2     19 249.800 -3   -249.11 1913.3 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Summary

La funció summary() del model ens dona un resum. Ens dona els paràmetres del model estimats, l'error estàndard, i el t-valor i la significació del paràmetre.

Per la variancia de l'error afegim $sigma^2

```
summary(mod)$sigma^2
```

```
## [1] 0.04339874
```

```
summary(mod)
```

```
##
## Call:
## lm(formula = Y ~ X * M, data = dades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38023 -0.08083 -0.01604  0.08359  0.58419
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.028462   0.140966   0.202    0.843
## X            0.098814   0.001671  59.150  < 2e-16 ***
## MG           0.057054   0.211438   0.270    0.791
## X:MG        -0.017426   0.002409  -7.235 1.99e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2083 on 16 degrees of freedom
## Multiple R-squared:  0.9972, Adjusted R-squared:  0.9967
## F-statistic:  1913 on 3 and 16 DF,  p-value: < 2.2e-16
```

Errors

Errors vs Y

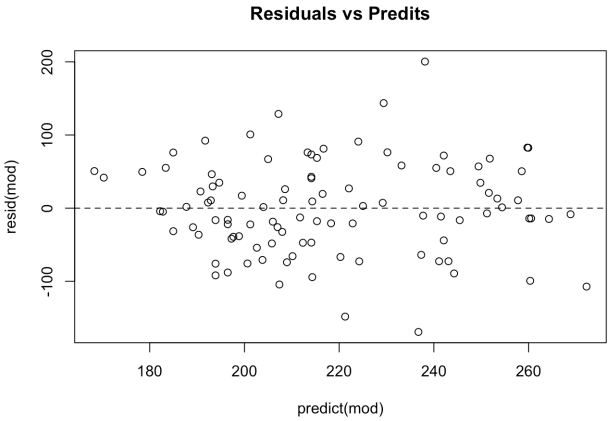
Els errors els podem veure amb la funció plot i el parametre *which* = 1. La funció plot té més

funcionalitats si el primer paràmetre és el model lineal i *which* = [1,6]:

1. A plot of residuals against fitted values
2. A normal Q-Q plot
3. A Scale-Location plot of sqrt(| residuals |) against fitted values
4. A plot of Cook's distances versus row labels
5. A plot of residuals against leverages
6. A plot of Cook's distances against leverage/(1-leverage)

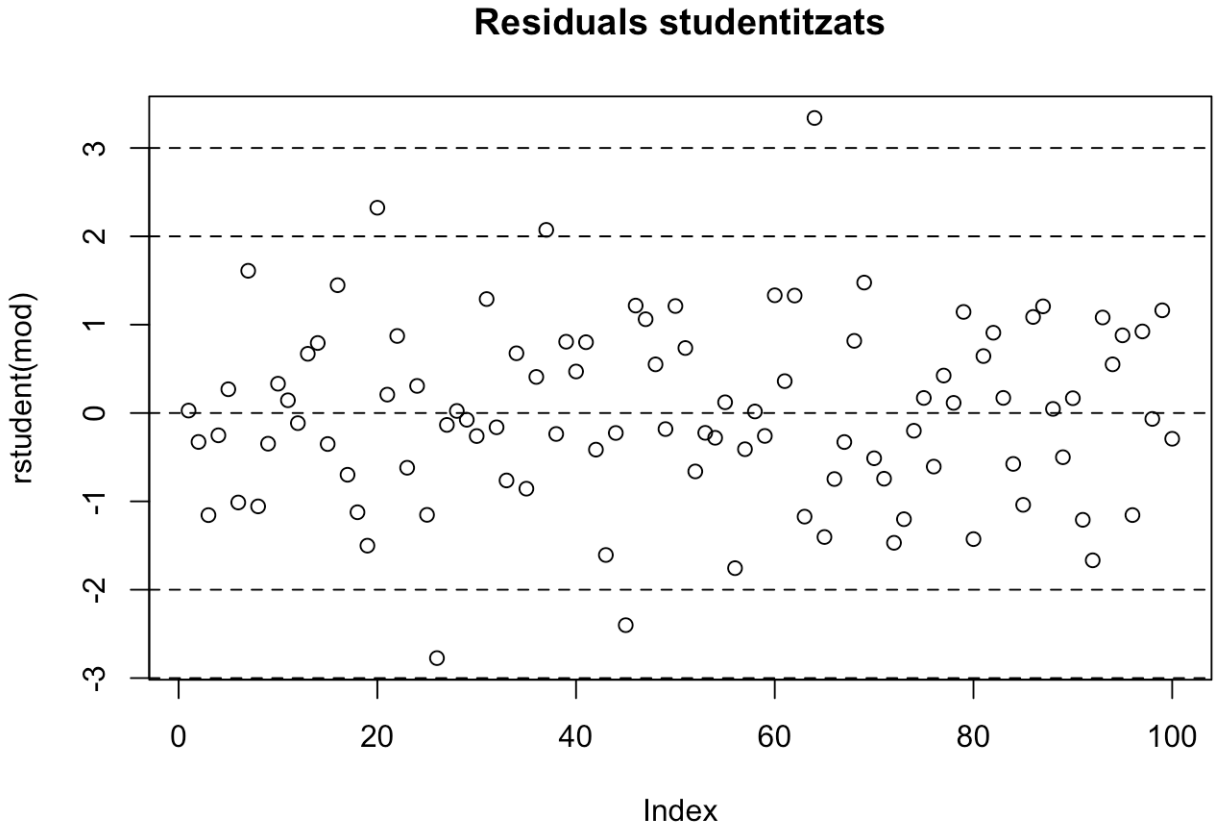
Ara veiem el plot dels residus. Recorda que s'ha de veure una distribució uniforme sobre el pla. Si es veu alguna forma d'embut o paràbòlica cal aplicar transformacions. Aquesta distribució uniforme verifica la hipòtesis de l'independència dels errors.

```
dd <- read.csv2("./col.csv")
mod<-lm(C~P,dd)
# plot(mod,which=1)
plot(predict(mod),resid(mod),main="Residuals vs Predits")
# línia al 0 per veure el centre
abline(h=0,lty=2)
```



Errors studentizats Aqui cal veure que no hi ha valors per sobre del 3 o més avall del -3, y a més que es pot observar normalitat del valors.

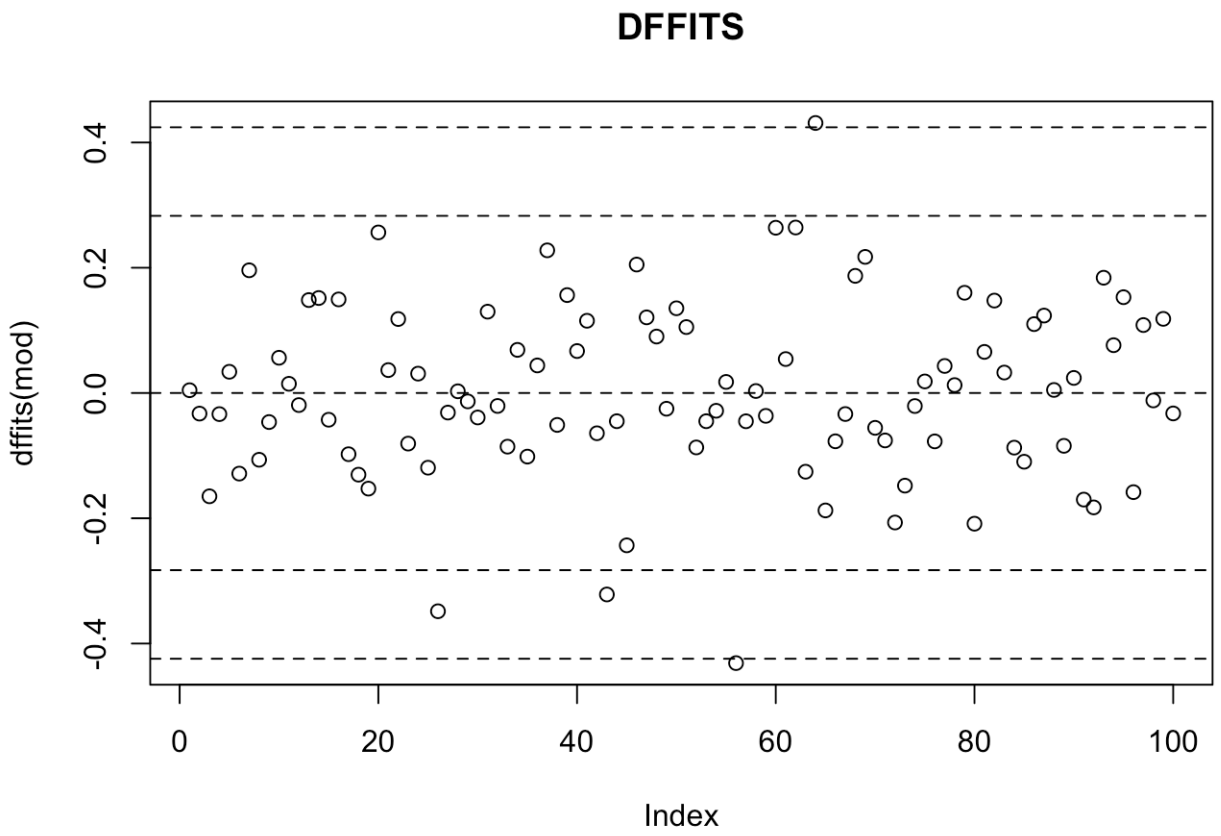
```
plot(rstudent(mod),main="Residuals studentitzats")
abline(h=c(-3,-2,0,2,3),lty=2)
```



Valors influents

Els DFFITS són una mesura similar a la distància de Cook. Ens indica si hi ha algun residu molt lluny.

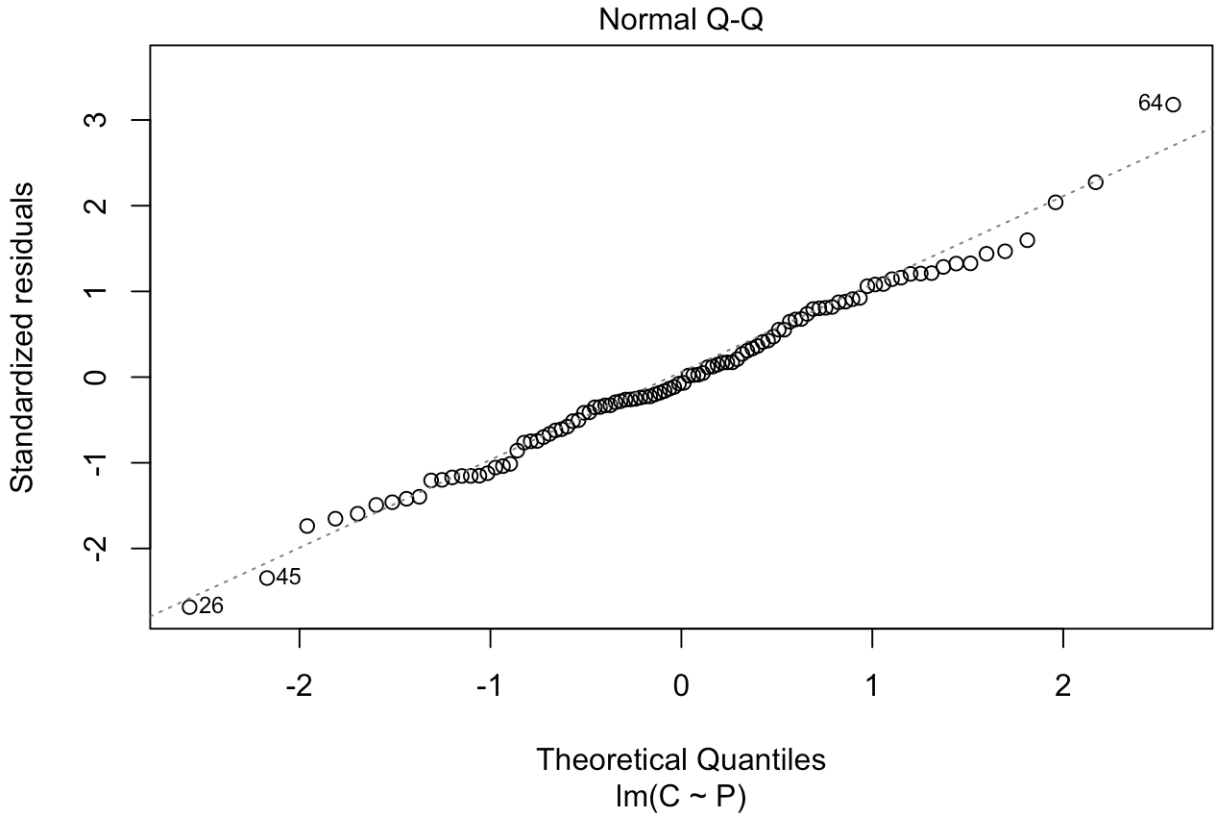
```
p<-2
n<-dim(dd)[1]
plot(dffits(mod),main="DFFITS")
# Les línies que posem son als valors {-3, -2, 0, 2, 3} però multiplicades per
un factor de squrt(p/n)
abline(h=c(-3,-2,0,2,3)*sqrt(p/n),lty=2)
```



QQ-plot dels errors

Per fer la segona comprovació dels errors fem l'anomenat qq-plot dels errors. Aquí hem d'observar linealitat.

```
plot(mod, which=2)
```



Tests d'hipòtesis

per plantejar un test d'hipòtesis de l'estil:
$H_0 : E\left[C|P = 60, E = 15, H = 150\right] = 200$
$H_1 : E\left[C|P = 60, E = 15, H = 150\right] \neq 200.$

```
dd$EP<-dd$P-(dd$H/2-10)
modE2v<-lm(C~EP+E,dd)
P = 60
E = 15
H = 150
lht(modE2v,c(1,P-(H/2-10),E),200)
```

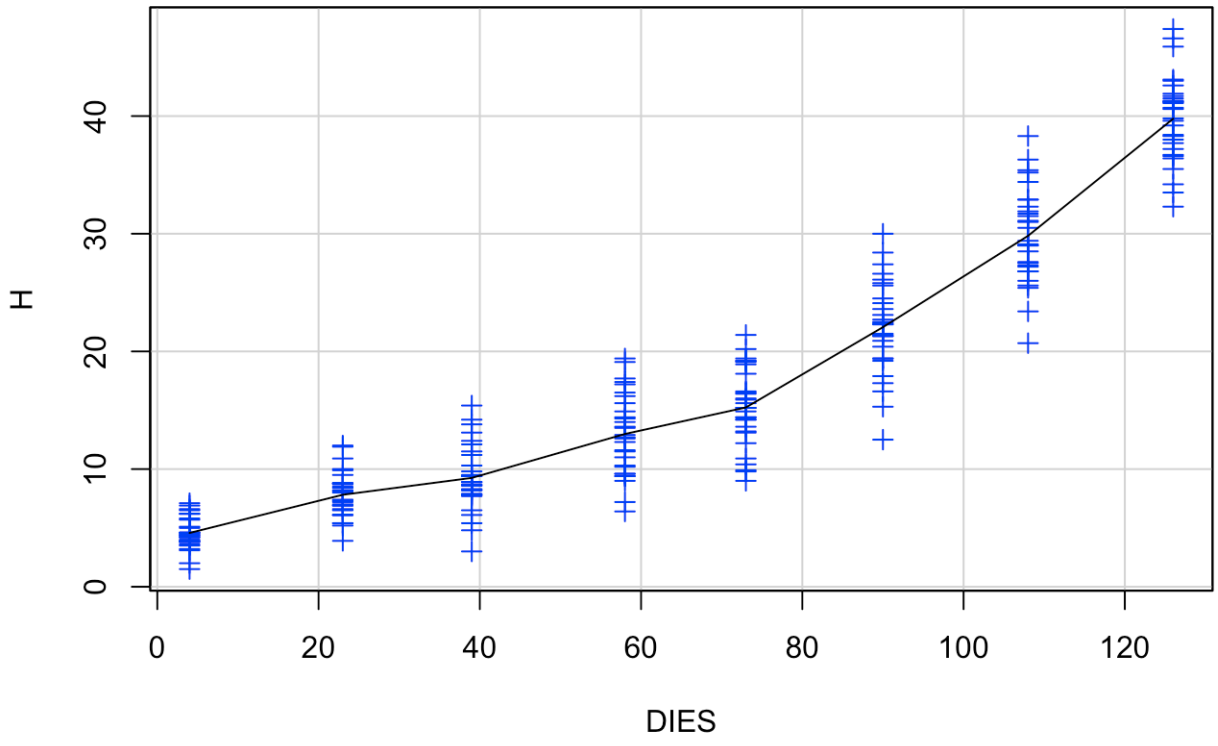
```
## Linear hypothesis test
##
## Hypothesis:
## (Intercept) - 5 EP  + 15 E = 200
##
## Model 1: restricted model
## Model 2: C ~ EP + E
##
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     98 157934
## 2     97  87052  1     70882 78.981 3.384e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Transformacions

Quan els models no compleixen les condicions, cal fer transformacions de diferents tipus per ajustar les prediccions. A vegades cal provar diferents transformacions per saber quina és la bona, per comprovar-ho anem fent les gràfiques dels residus i les de les bandes de predicció. Posarem l'exemple del ficus perquè és més veganfriendly.

Primer carreguem les dades i fem una mica de descriptiva.

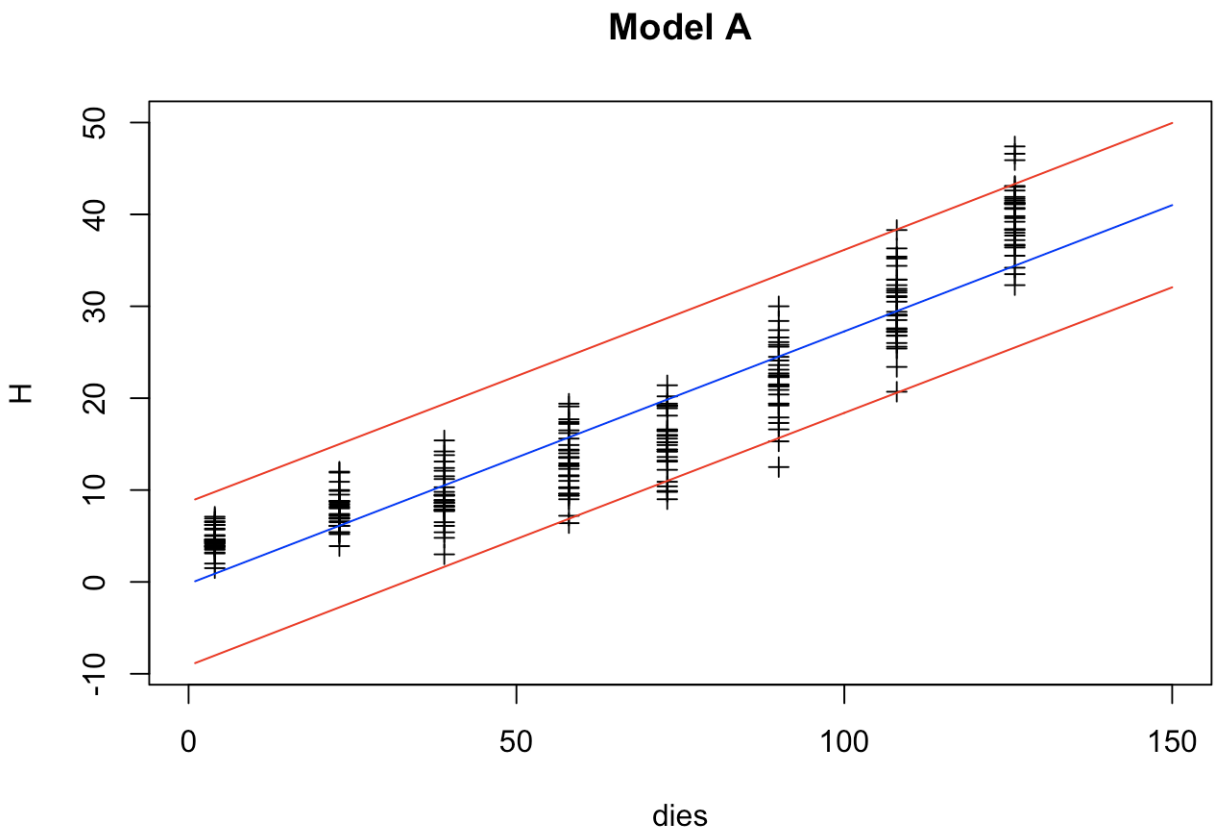
```
ddt <- read.csv2("./ficus.csv")
# una taula que diu coses
cv<-function(x) {sd(x)/mean(x)}
di<-function(x) {var(x)/mean(x)}
t<-tabular((DIES=as.factor(DIES))~H*((n.dades=1)+(mitjana=mean)+(desv.tipus=s
d)+(coef.variació=cv)+(index.disp.=di)),ddt)

scatterplot(H~DIES, regLine=F, smooth=F, boxplots=F,pch=3, data=ddt)
lines(rowLabels(t),t[,2])
```



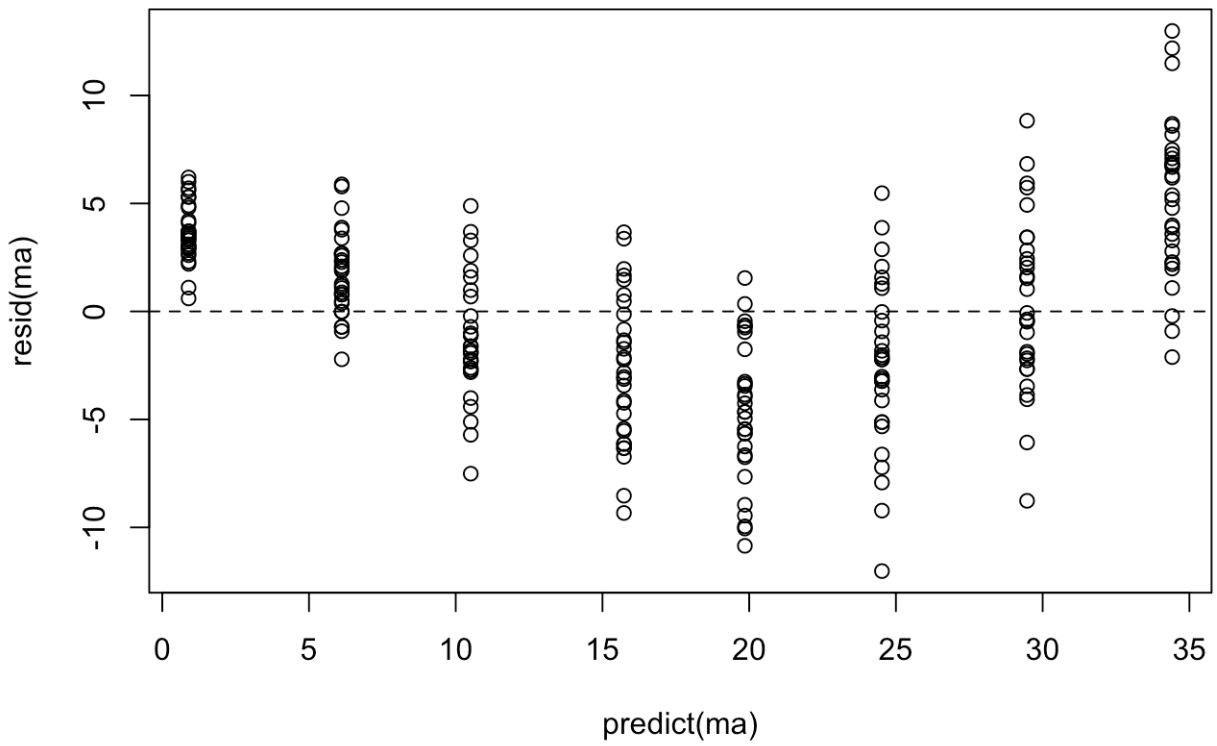
Regressió simple

Bueno aquest model (regressió lineal) és per veure que als residus veiem formes xungues que no hauríem de veure. Posem les tres gràfiques: bandes de predicció, residus vs predits i residus studentitzats.

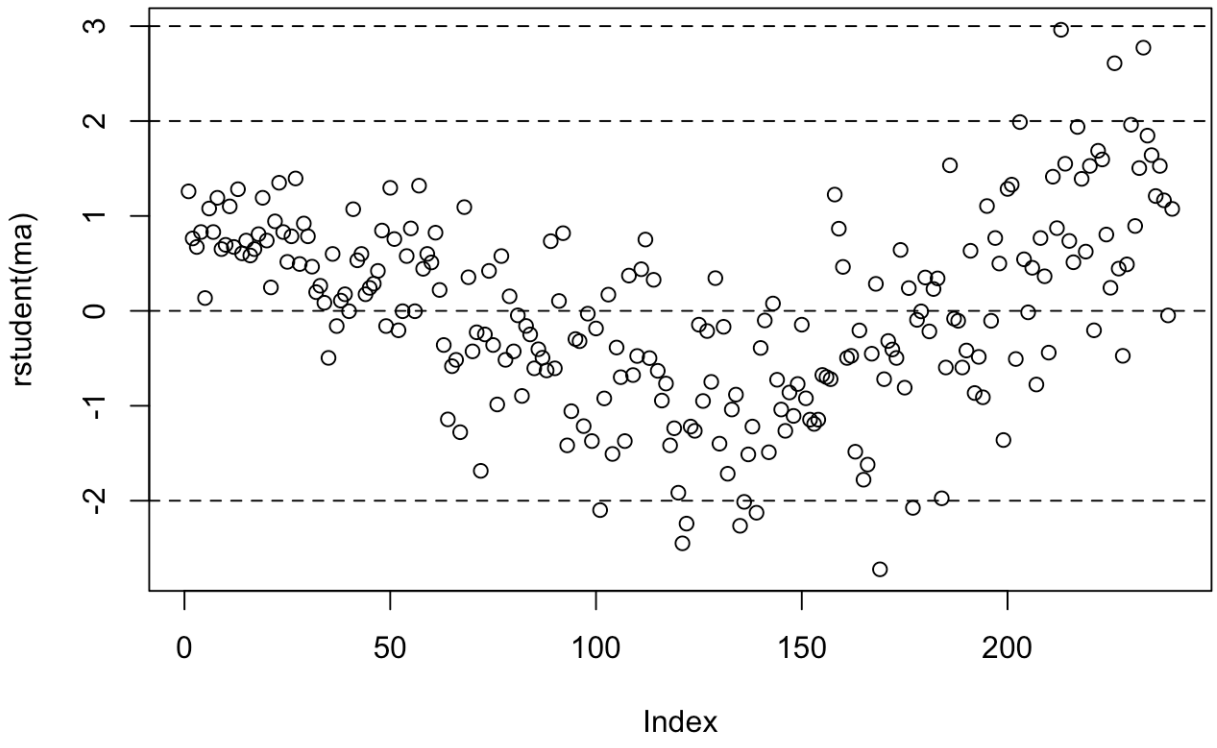
```
ma<-lm(H~DIES,ddt)
dies=1:150


# Bandes de predicció
pra<-predict(ma,data.frame(DIES=dies),interval="prediction")
plot(ddt$DIES,ddt$H,pch=3,xlim=c(0,150),xlab="dies",ylab="H",ylim=c(min(ddt$H,p
ra),max(ddt$H,pra)),main="Model A")
lines(dies,pra[,"fit"],col="blue")
lines(dies,pra[,"lwr"],col="red")
lines(dies,pra[,"upr"],col="red")
```




```
# residus
plot(predict(ma),resid(ma))
abline(h=0,lt=2)
```



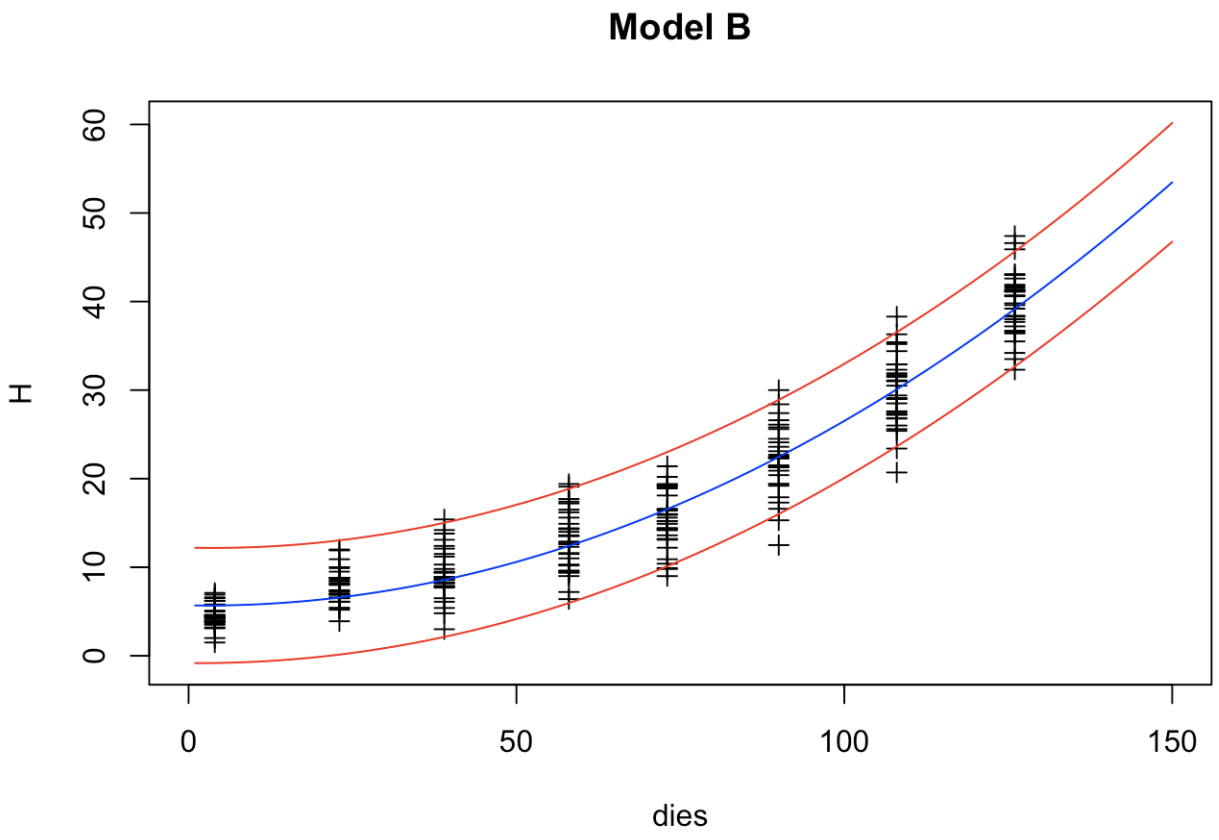

```
plot(rstudent(ma))
abline(h=c(-3,-2,0,2,3),lt=2)
```



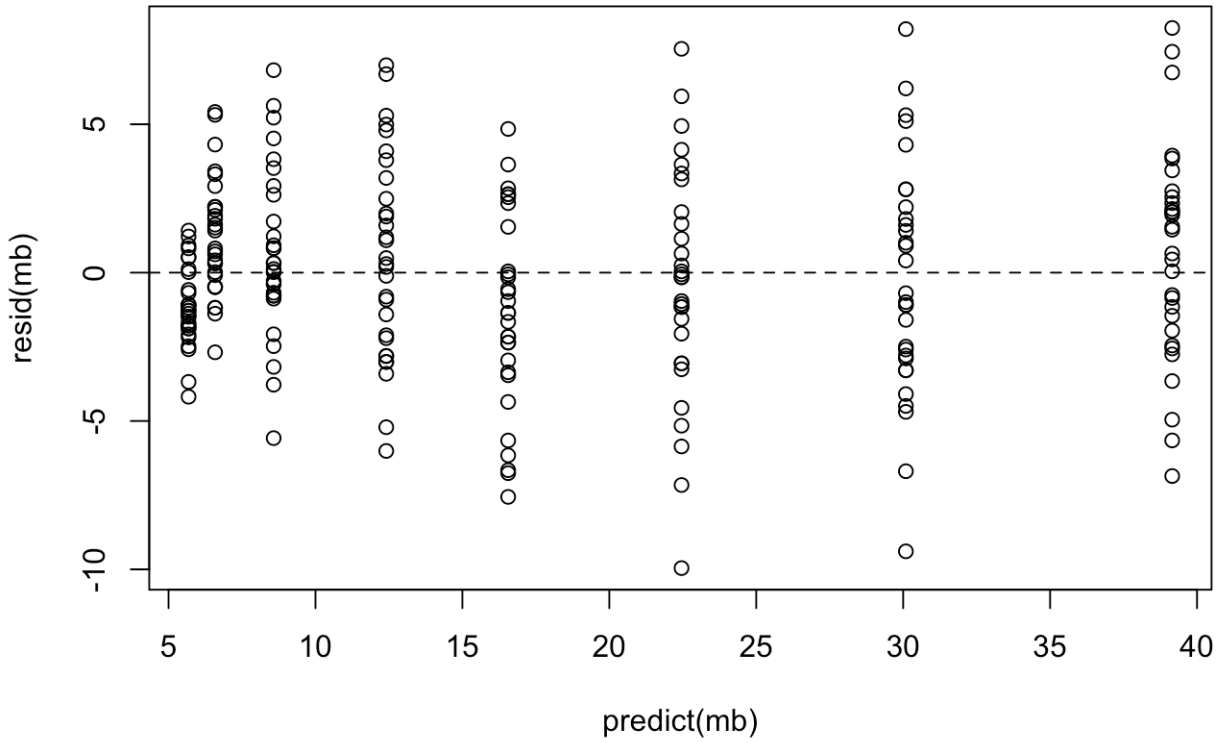
Paràbola

Ara considerarem com a funció de regressió la paràbola de H respecte DIES. Cal afegir una columna que són les observacions al quadrat (en aquest cas DIES). Si es vol augmentar el grau del polinomi cal afegir columnes d'acord amb el grau.

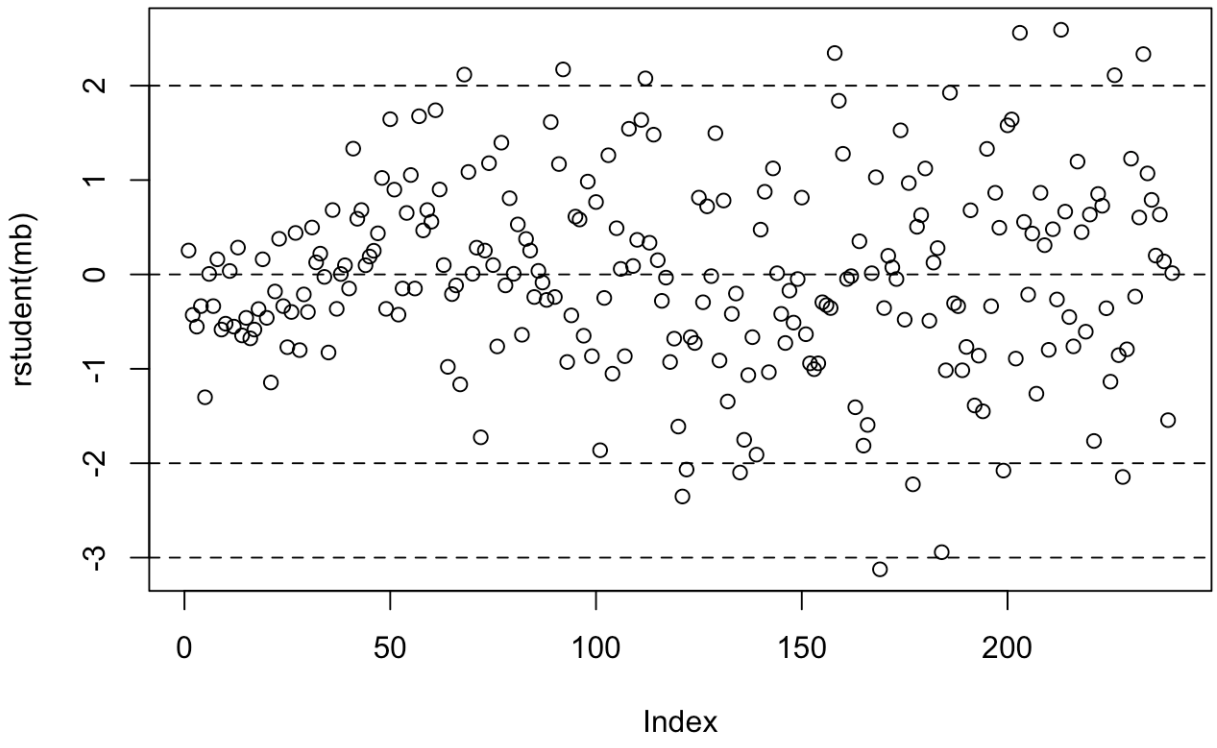
```
# veiem com plantegem el model diferent (creem una columna nova a ddt):
ddt$DIES2 <- ddt$DIES^2
mb<-lm(H~DIES+DIES2,ddt)

# Bandes de predicció
prb<-predict(mb,data.frame(DIES=dies,DIES2=dies^2),interval="prediction")
plot(ddt$DIES,ddt$H,pch=3,xlim=c(0,150),xlab="dies",ylab="H",ylim=c(min(ddt$H,p
rb),max(ddt$H,prb)),main="Model B")
lines(dies,prb[,"fit"],col="blue")
lines(dies,prb[,"lwr"],col="red")
lines(dies,prb[,"upr"],col="red")
```




```
# residus
plot(predict(mb),resid(mb))
abline(h=0,lt=2)
```



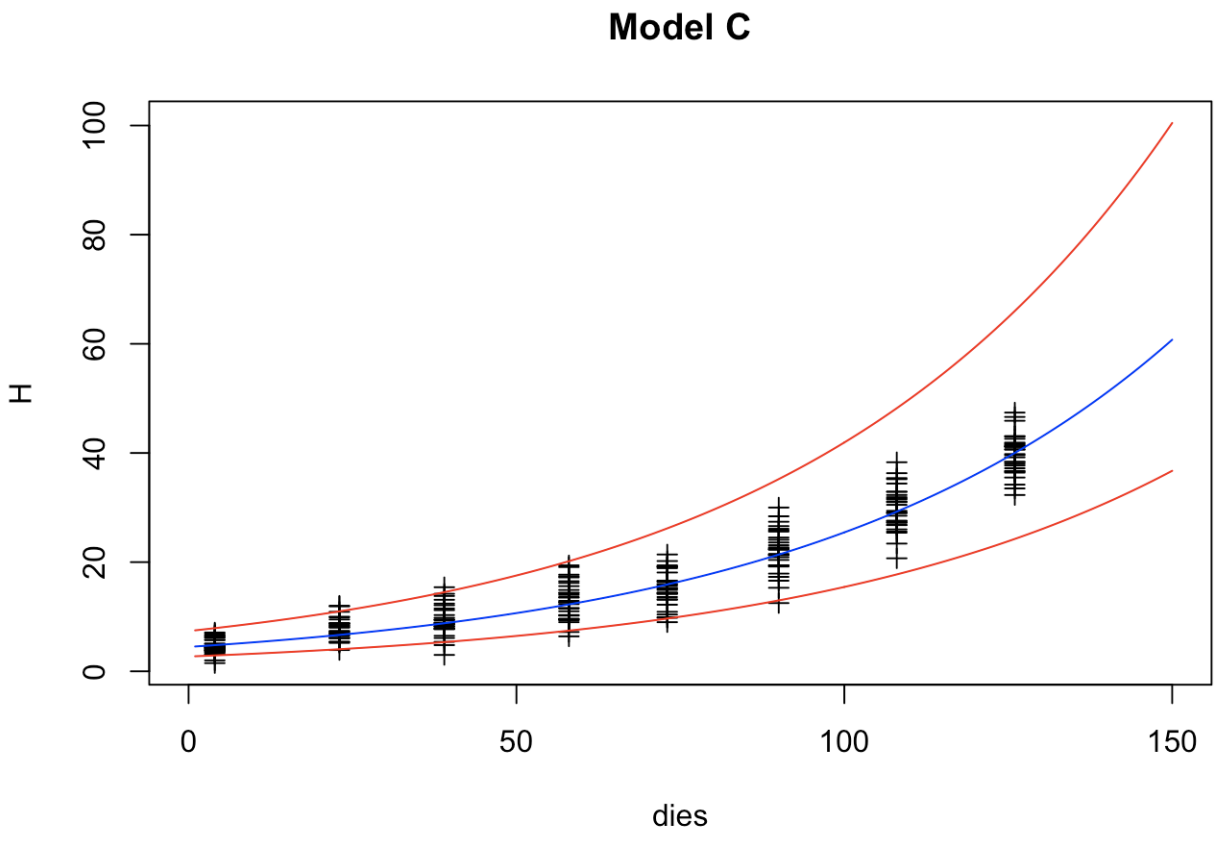

```
plot(rstudent(mb))
abline(h=c(-3,-2,0,2,3),lt=2)
```



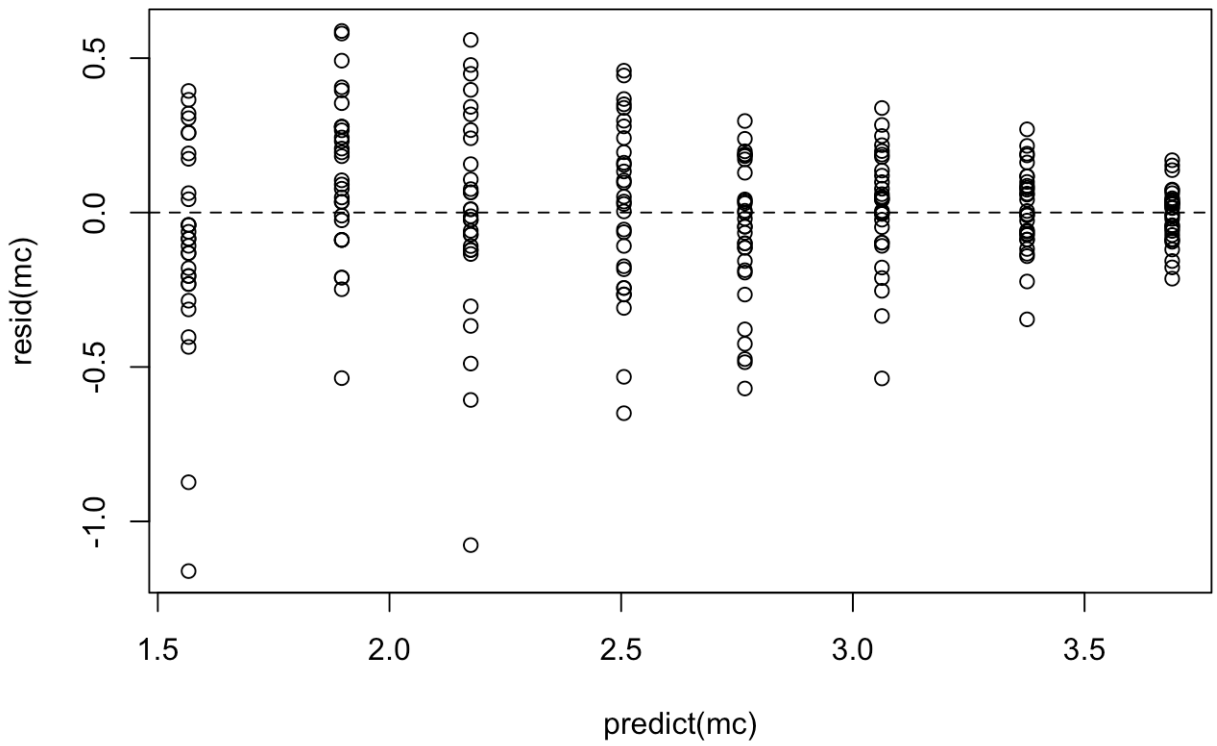
Logaritme

Ara la transformació és amb el logaritme. Tenim que: $E\left[log\left(H\right)|DIES\right] = \alpha + \beta \cdot DIES$ i $Var\left(log\left(H\right)|DIES\right) = \sigma^2$.
L'interessant és que la transformació log homogeneitza les variàncies quan Var∝μ2 encara que l'important en aquest cas és que linealitza la funció eα+β·DIES.

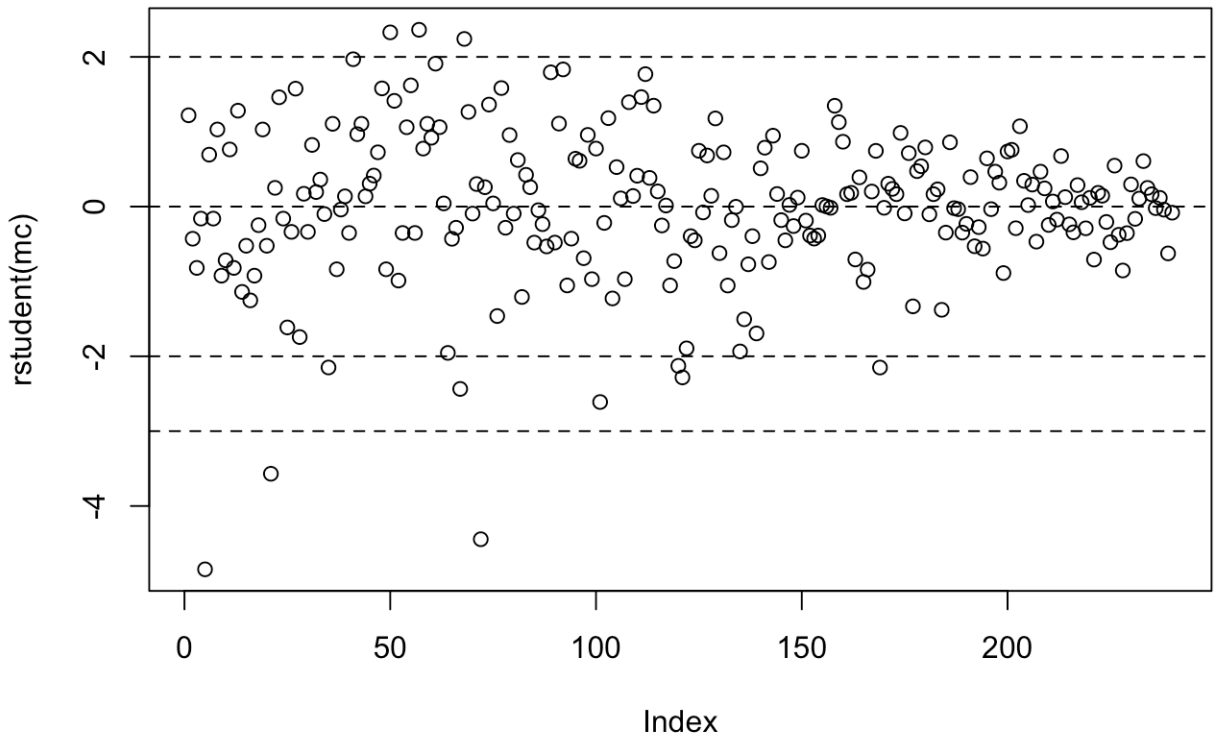
```
# model log(H)
mc<-lm(log(H)~DIES,ddt)
# també tenim transformacions del tipus: (producció de llet)
# ml<-lm(log(PROD)~DIES+log(DIES),dd)

# bandes de predicció
prc<-predict(mc,data.frame(DIES=dies),interval="prediction")
plot(ddt$DIES,ddt$H,pch=3,xlim=c(0,150),xlab="dies",ylab="H",ylim=c(min(ddt$H,e
xp(prc)),max(ddt$H,exp(prc))),main="Model C")
lines(dies,exp(prc[,"fit"]),col="blue")
lines(dies,exp(prc[,"lwr"]),col="red")
lines(dies,exp(prc[,"upr"]),col="red")
```




```
#residus
plot(predict(mc),resid(mc))
abline(h=0,lt=2)
```



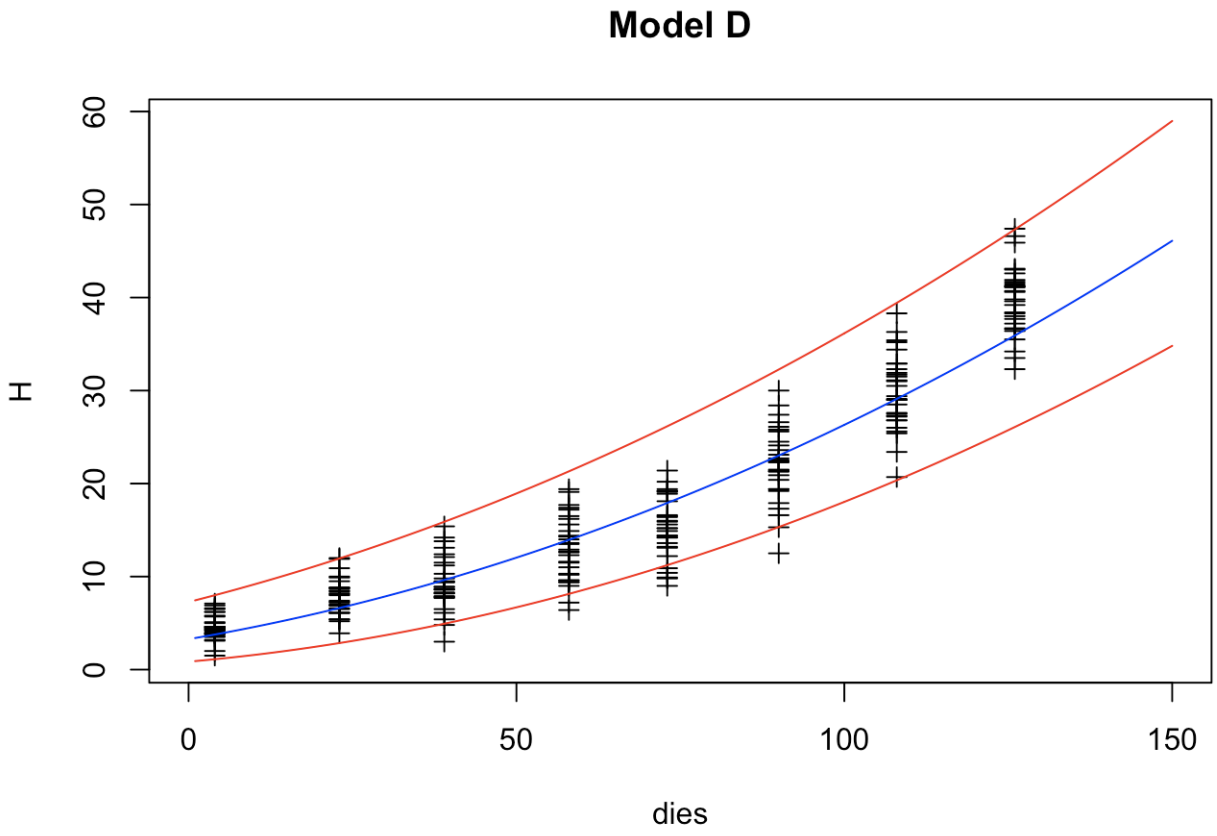

```
plot(rstudent(mc))
abline(h=c(-3,-2,0,2,3),lt=2)
```



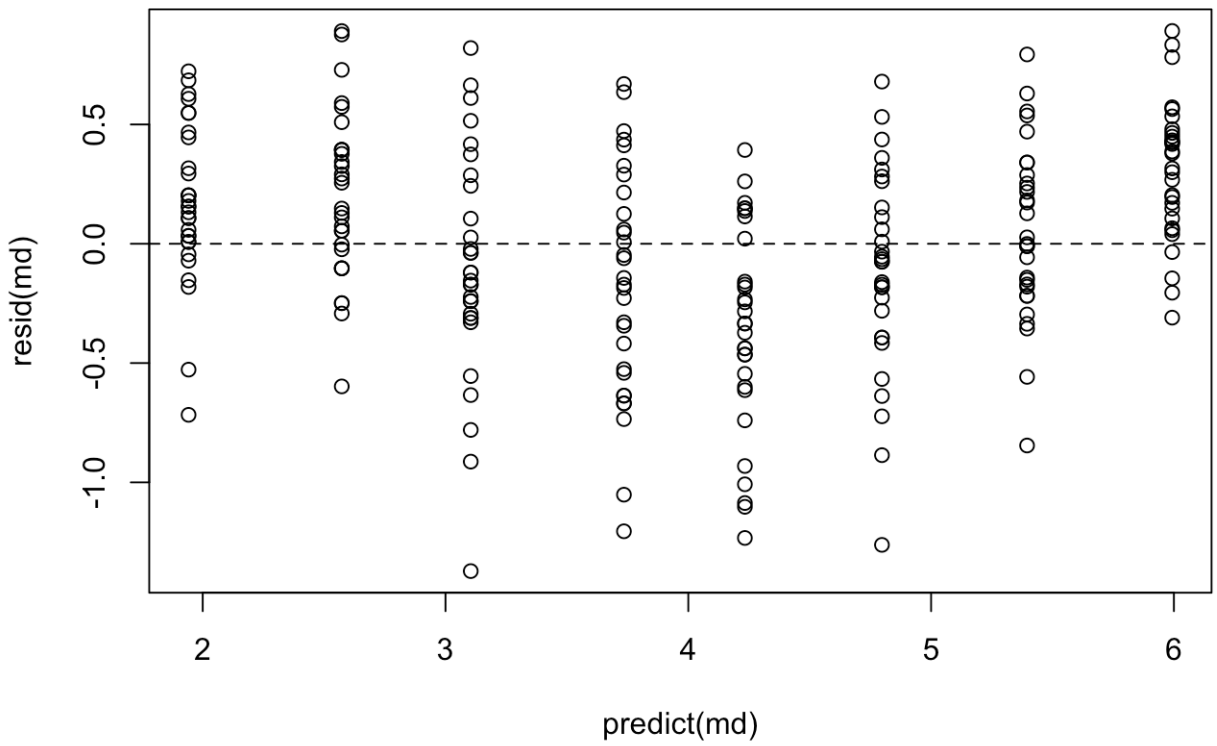
Arrel quadrada

La transformació √ homogeneitza les variàncies quan Var∝μ.

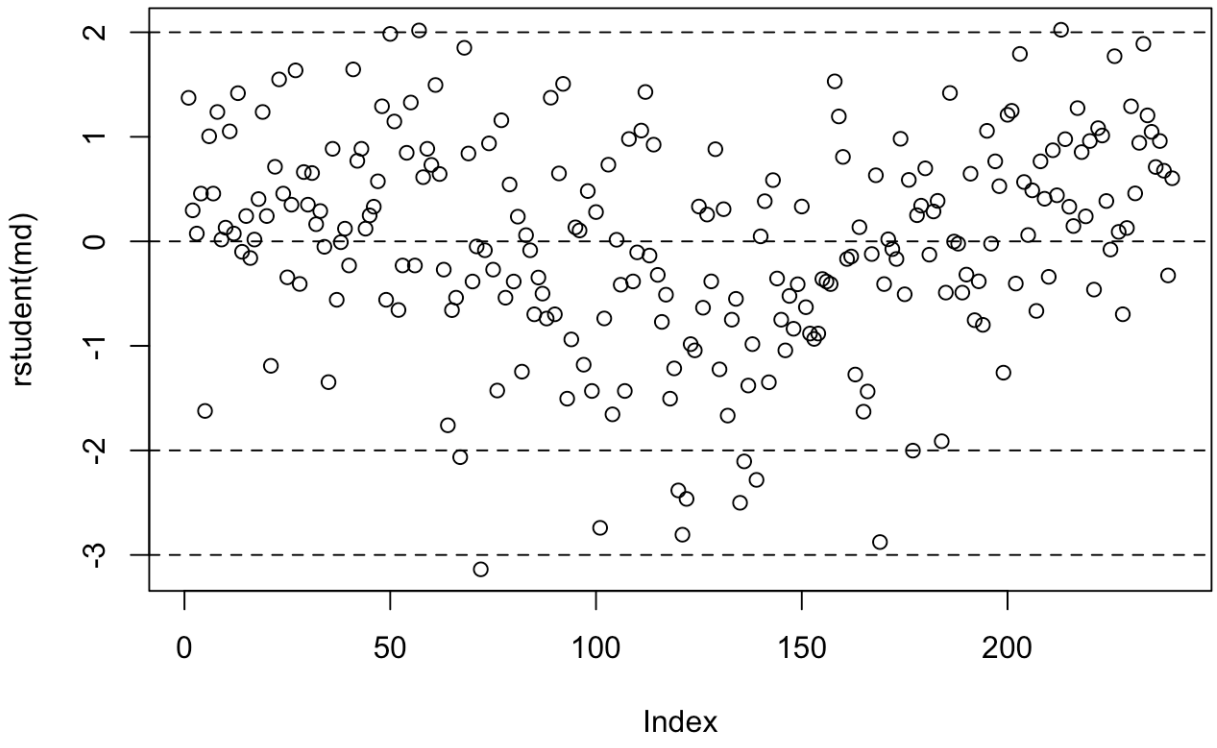
```
# veiem la transformació sqrt(H)
md<-lm(sqrt(H)~DIES,ddt)

# bandes
prd<-predict(md,data.frame(DIES=dies),interval="prediction")
plot(ddt$DIES,ddt$H,pch=3,xlim=c(0,150),xlab="dies",ylab="H",ylim=c(min(ddt$H,p
rd^2),max(ddt$H,prd^2)),main="Model D")
lines(dies,prd[,"fit"]^2,col="blue")
lines(dies,prd[,"lwr"]^2,col="red")
lines(dies,prd[,"upr"]^2,col="red")
```




```
# residus
plot(predict(md),resid(md))
abline(h=0,lt=2)
```




```
plot(rstudent(md))
abline(h=c(-3,-2,0,2,3),lt=2)
```



Test Levene

Serveix per comprovar la hipòtesis d'Homocedasticitat. Per acceptar l'hipòtesis nul·la hem de veure un p-valor menor a 0.05. A més cal posar Factors o una cosa així.

```
ddt$FDIES<-as.factor(ddt$DIES)
leveneTest(resid(md),ddt$FDIES)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   7  1.4751  0.177
##       232
```

Comparació

Podem fer una taula per comparar diferents models; per exemple, dels que han sortit comparem la logLikliehood, AIC, BIC (com més petits millor) i l'R^2 ajustat:

```
rbind(
  "logLik"=c("Model A"=logLik(ma),"Model B"=logLik(mb),"Model C"=logLik(mc),"Mo
del D"=logLik(md)),
  "AIC"=c(AIC(ma),AIC(mb),AIC(mc),AIC(md)),
  "BIC"=c(BIC(ma),BIC(mb),BIC(mc),BIC(md)),
  "R2"=c(summary(ma)$adj.r.squared,summary(mb)$adj.r.squared,summary(mc)$adj.r.
squared,summary(md)$adj.r.squared))
```

```
##             Model A      Model B    Model C      Model D
## logLik -699.8636454 -622.5319643 -9.1404594 -146.1123925
## AIC    1405.7272908 1253.0639286 24.2809189  298.2247850
## BIC    1416.1692076 1266.9864843 34.7228356  308.6667018
## R2        0.8542609    0.9231698  0.8814138    0.8963332
```

GLM

Aqui es donde comienza la fiesta. Primero de todo, las familias:

a. Normal
b. Binomial
c. Poisson
d. Gamma
e. Inversa Gaussiana
f. Otras

Si te olvidas de como tienes que escribirlas solo ve a `family` . Segundo, las funciones link se especifican justo al lado y entre paréntesis de la familia. Ej.: `family=binomial(link=probit)` . Para hacer un ejemplo completo usaré uno de los últimos ejercicios:

```
ddglm <- data.frame(time=c(16,16,16,24,24,24),
                    dose=c(0, 0.45, 0.75, 0, 0.45, 0.75),
                    tumor=c(1,3,7, 20,98,118),
                    total=c(205,304,193, 762,888,587))
ddglm$time <- as.factor(ddglm$time)

m <- glm(tumor/total~time*dose, data=ddglm, family=binomial, weights=total)
anova(m, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: tumor/total
##
## Terms added sequentially (first to last)
##
##
##           Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                          5    198.535
## time       1   75.001         4    123.534   <2e-16 ***
## dose       1  121.498         3      2.035   <2e-16 ***
## time:dose  1    0.067         2      1.968   0.7956
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(m)
```

```
##
## Call:
## glm(formula = tumor/total ~ time * dose, family = binomial, data = ddglm,
##     weights = total)
##
## Deviance Residuals:
##       1        2        3        4        5        6
##  0.4343  -0.4942   0.2214  -0.6904   0.8778  -0.4892
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.7871     0.9398  -6.158 7.38e-10 ***
## time24        2.3262     0.9561   2.433   0.0150 *
## dose          3.2279     1.4869   2.171   0.0299 *
## time24:dose  -0.3864     1.5158  -0.255   0.7988
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 198.5347  on 5  degrees of freedom
## Residual deviance:   1.9683  on 2  degrees of freedom
## AIC: 36.225
##
## Number of Fisher Scoring iterations: 4
```

En general se suele usar el test Chi cuadrado para hacer el anova, aunque también puede aparecer el F de fisher, depende de la familia, por suerte R te suele avisar si lo pones mal. Por lo demás, el análisis es como en los modelos lineales.

Estadístics

Aquí tenim els estadístics de Pearson, paràmetres de dispersió de Pearson i de deviance, i la variància $Var(Y|\{a,b,c\})$.

```
c(Pearson=Pearson<-sum(resid(m,ty="pearson")^2))
```

```
##  Pearson
## 1.979882
```

```
c(dispP=dispP<-Pearson/m$df.residual)
```

```
##     dispP
## 0.9899409
```

```
c(Deviance=Deviance<-sum(resid(m,ty="deviance")^2),m$deviance)
```

```
## Deviance
##  1.96831  1.96831
```

```
c(dispD=dispD<-Deviance/m$df.residual)
```

```
##     dispD
## 0.9841548
```

Residuos

Hay de dos tipos, devianza y pearson. Se puede utilizar para calcular el parámetro de dispersión que en el caso de la Poisson y la binomial debe ser cercano a 1.

```
resid(m, ty="pearson")
```

```
##          1          2          3          4          5          6
##  0.4721751 -0.4735751  0.2243891 -0.6743959  0.8890891 -0.4868463
```

```
resid(m, ty="deviance")
```

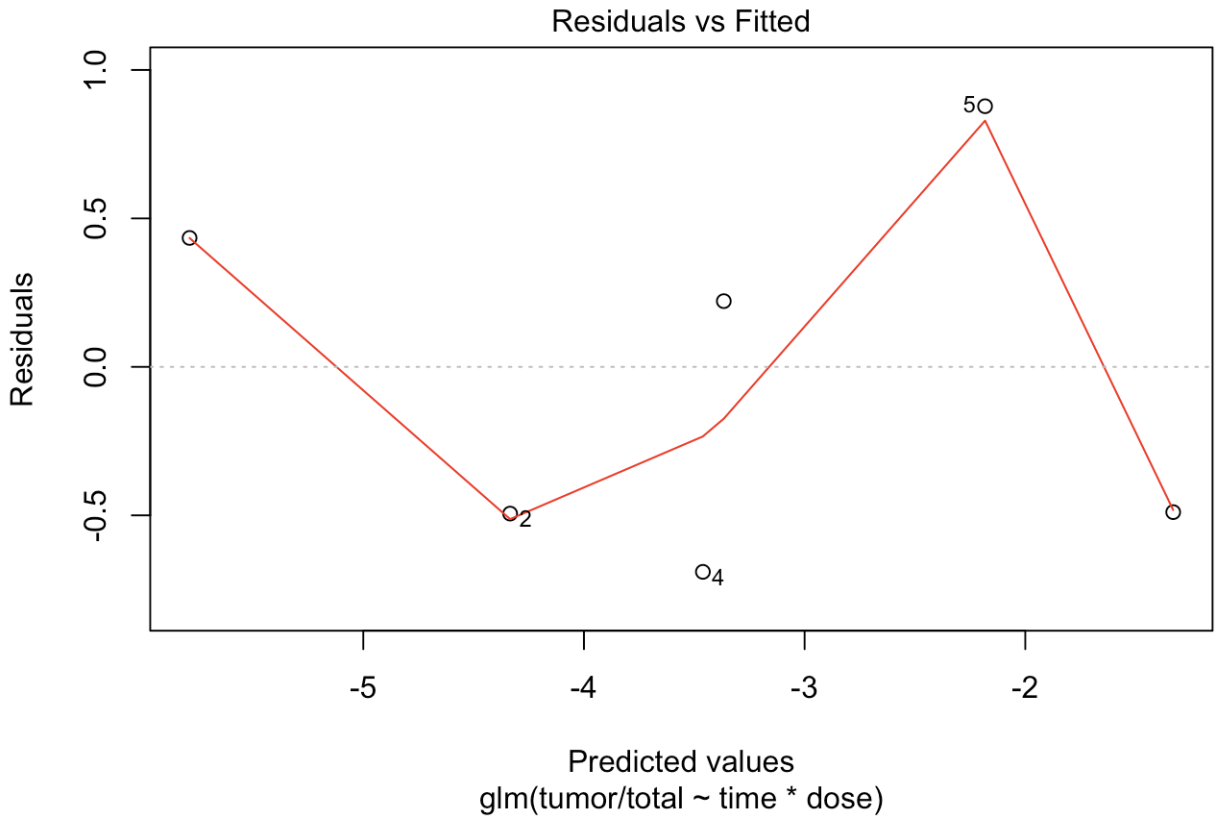
```
##          1          2          3          4          5          6
##  0.4342646 -0.4941714  0.2213597 -0.6903474  0.8778381 -0.4892231
```

```
PRS <- sum(resid(m, ty="pearson"))
c("Par.disp"=PRS/m$df.res, "p-valor"=2*min(pchisq(PRS,m$df.res),1-pchisq(PRS,
m$df.res)))
```

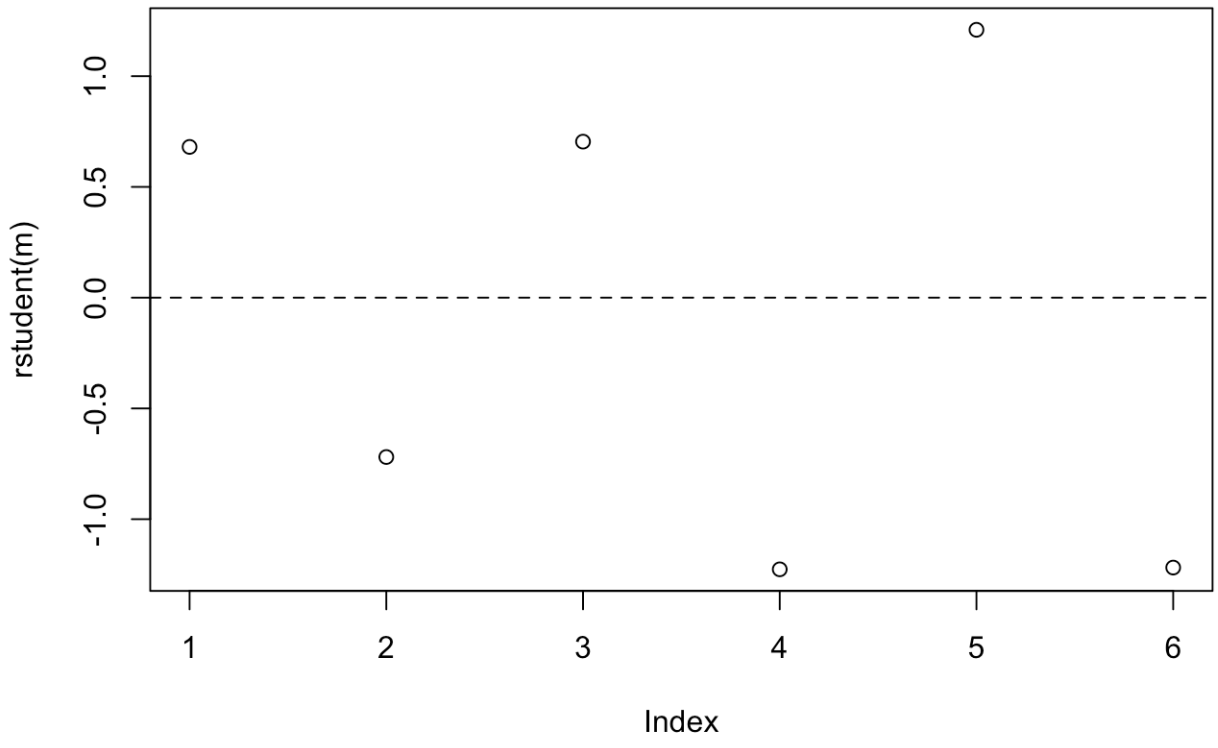
```
##    Par.disp     p-valor
## -0.02458196  0.00000000
```

En este caso sale infradispersión, si sale sobredispersión entonces hay que preocuparse mal y o bien los datos no siguen la distribución esperada o el modelo está mal. También se pueden hacer los análisis de siempre.

```
plot(m, 1)
```




```
plot(rstudent(m))
abline(h=c(-2,0,2), lty=2)
```



Predicciones

```
Dose <- seq(0, 1, .01)
plot(tumor/total~(dose), data=ddglm)
lines(predict(m, data.frame(time=as.factor(rep(16, 101)), dose=Dose), ty="respo
nse")~(Dose), col="red")
lines(predict(m, data.frame(time=as.factor(rep(24, 101)), dose=Dose), ty="respo
nse")~(Dose), col="green")
abline(h=c(0,1), col="grey")
```