Lab10

Sebastián Sánchez Sandí

2025-07-11

Homocedasticidad y normalidad

Primero cargamos las librerias que vamos a utilizar.

```
library(car)
```

```
## Loading required package: carData
```

Para este laboratorio se van a simular datos para poner a prueba los diferentes supuestos que se deben cumplir para que los resultados del a prueba ANOVA sean validos. Iniciamos con una funcion la cual nos devuelve datos de un hipotetico experimento de dos factores y la variable de respuesta provenga de un poblaciones normales que se definen a conveniencia.

```
fnorm = function(mu, var, replicas, trat1, trat2) {
  f1 = factor(rep(1:trat1, each = replicas, times = 2))
  f2 = factor(rep(1:trat2, each = replicas*trat1))
  muj = rep(mu, each = replicas)
  sd = sqrt(var)
  sdj = rep(sd, each = replicas)
  n = replicas * trat1 * trat2
  y = rnorm(n, muj, sdj)
  base = data.frame(f1, f2, y)
  return(base)
}
```

Ahora creamos una funcion la cual realice la prueba de bartlett y la prueba de levene para comparar los resultados de ambas.

```
homocedasticidad = function(base) {
  bartlett = bartlett.test(y~interaction(f1, f2), data = base)
  levene = leveneTest(y~f1*f2, data = base)
  return(list(bartlett, levene))
}
```

Ahora probamos estas funciones con un ejemplo arbitrario donde las varianzas son distintas. En teoria, las dos pruebas deberian indicar que no existe homocedasticidad. Con la siguiente simulacion vamos a aproximar la proporcion de rechazos de cada una de las pruebas, el numero de replicas se puede variar. Mientras mas replicas existan, mayor sera la potenciade la prueba.

```r
mu = c(2, 8, 5, 6, 2, 10)
var = c(1, 5, 3, 8, 2, 1)
replicas = 15
trat1 = 3
trat2 = 2
propBartlett = 0
propLevene = 0
for(i in 1:1000) {
  base = fnorm(mu, var, replicas, trat1, trat2)
  res = homocedasticidad(base)
  propBartlett = propBartlett + (res[[1]]$p.value < 0.05)
  propLevene = propLevene + (res[[2]]$`Pr(>F)`[1] < 0.05)
}

print(propBartlett / 1000)
```

```
## [1] 0.982
```

```r
print(propLevene / 1000)
```

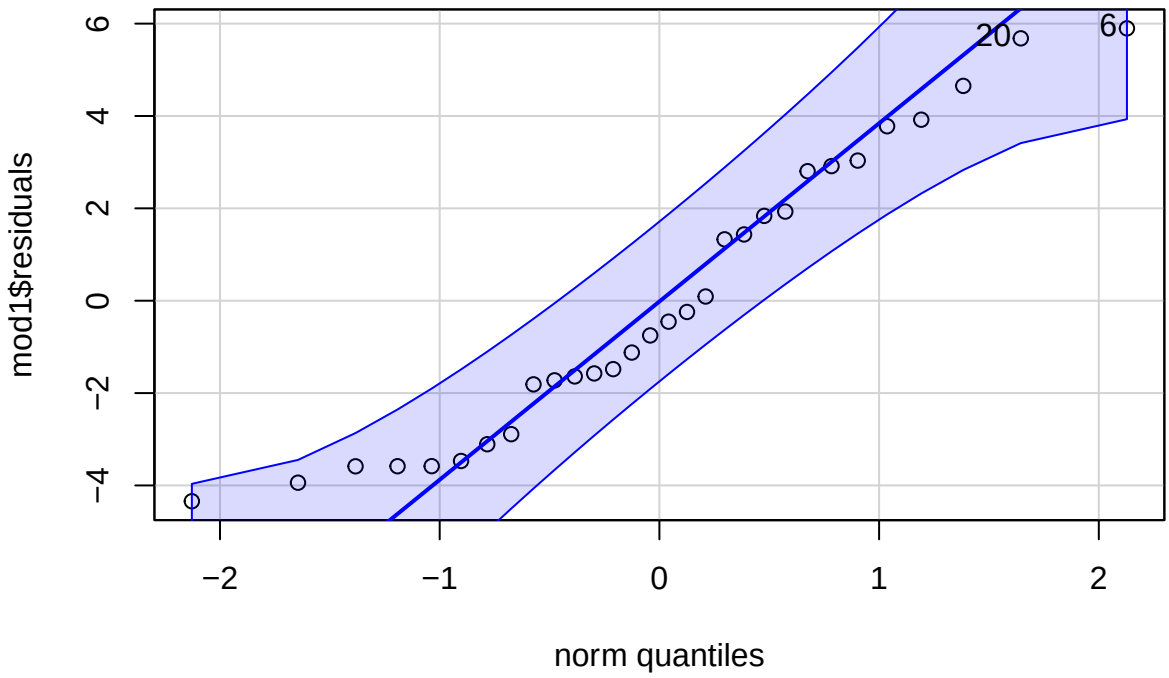
```
## [1] 0.864
```

Ahora para comprobar el supuesto de normalidad tenemos 2 caminos que dependen del tamaño de la muestra. Si tenemos una muestra donde hay muchas replicas por tratamiento, entonces hacemos el analisis dependiendo por separado. Caso contrario y si existe homocedasticidad, entonces juntamos los residuales y verificamos el supuesto. Nunca se juntan las respuestas, solo los residuales. Siempre es buena idea revisar primero el supuesto de homocedasticidad previo al supuesto de normalidad.

Probamos primero cuando no existe homocedasticidad.

```r
mu = c(2, 8, 5, 6, 2, 10)
var = c(1, 5, 3, 8, 2, 1)
replicas = 5
trat1 = 3
trat2 = 2
base = fnorm(mu, var, replicas, trat1, trat2)

mod1 = lm(y~f1*f1, data = base)
qqPlot(mod1$residuals)
```




```
## [1]  6 20
```

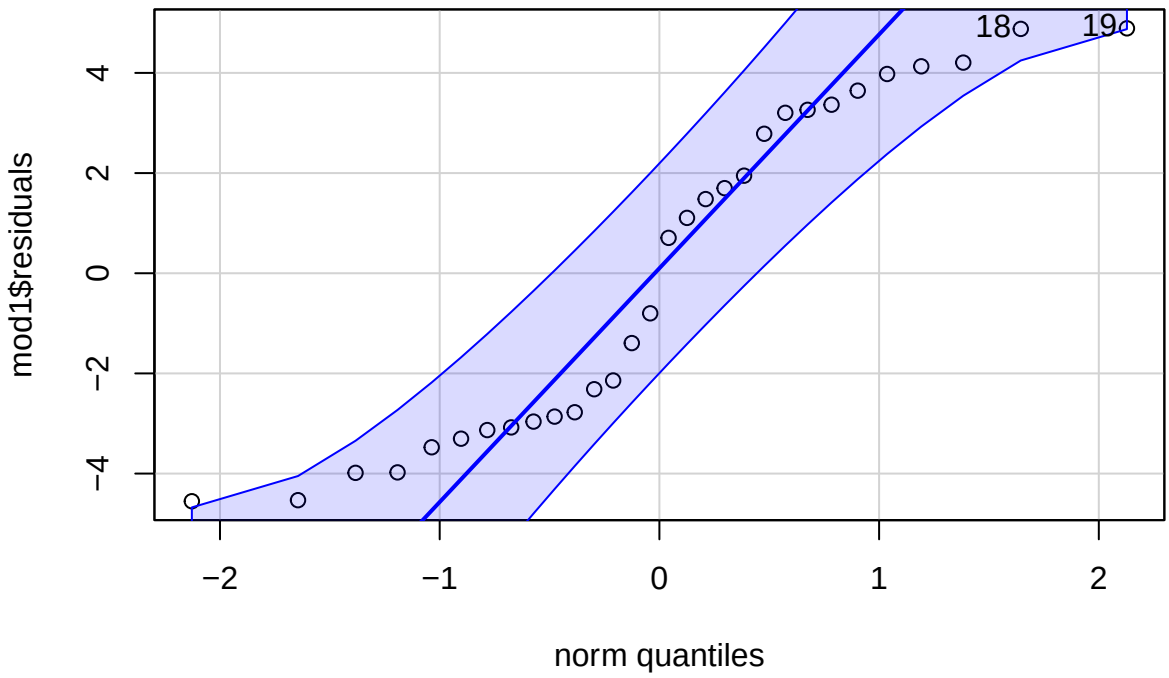
```
print(shapiro.test(mod1$residuals))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  mod1$residuals
## W = 0.93827, p-value = 0.08173
```

Ahora probamos bajo el supuesto de homocedasticidad.

```
mu = c(2, 8, 5, 6, 2, 10)
var = c(2, 2, 2, 2, 2, 2)
replicas = 5
trat1 = 3
trat2 = 2
base = fnorm(mu, var, replicas, trat1, trat2)

mod1 = lm(y~f1*f1, data = base)
qqPlot(mod1$residuals)
```




```
## [1] 19 18
```

```
print(shapiro.test(mod1$residuals))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  mod1$residuals
## W = 0.88832, p-value = 0.004404
```

Como podemos ver el supuesto de homocedasticiadad puede afectar algunas pruebas. Pero son particularmente sensibles al tamaño de la muestra.

Soluciones ante violacion a los supuestos

Transformacion logaritmica

En este curso en particular se trabaja unicamente con la transformacion lograritmica. Esta es una tecnica la cual ayuda a solventar el problema de la normalidad. Supongamos que tenemos una muestra de una poblacion la cual sigue una distribucion exponencial. Esta es claramente no normal. Definimos una funcion que nos devuelve precisamente esta muestra para ponerla a prueba.

```r
fexp = function(mu, replicas, trat1, trat2) {
  f1 = factor(rep(1:trat1, each = replicas, times = 2))
  f2 = factor(rep(1:trat2, each = replicas * trat1))
  muj = rep(mu, each = replicas)
  n = replicas * trat1 * trat2
  y = rexp(n, 1/muj)
  base = data.frame(f1, f2, y)
  return(base)
}
```

Ahora buscamos un ejemplo y ponemos a prueba los supuestos.

```r
mu = c(2, 8, 5, 6, 2, 10)
r = 5
trat1 = 3
trat2 = 2
base = fexp(mu, r, trat1, trat2)
res = homocedasticidad(base)
print(res[[1]])
```

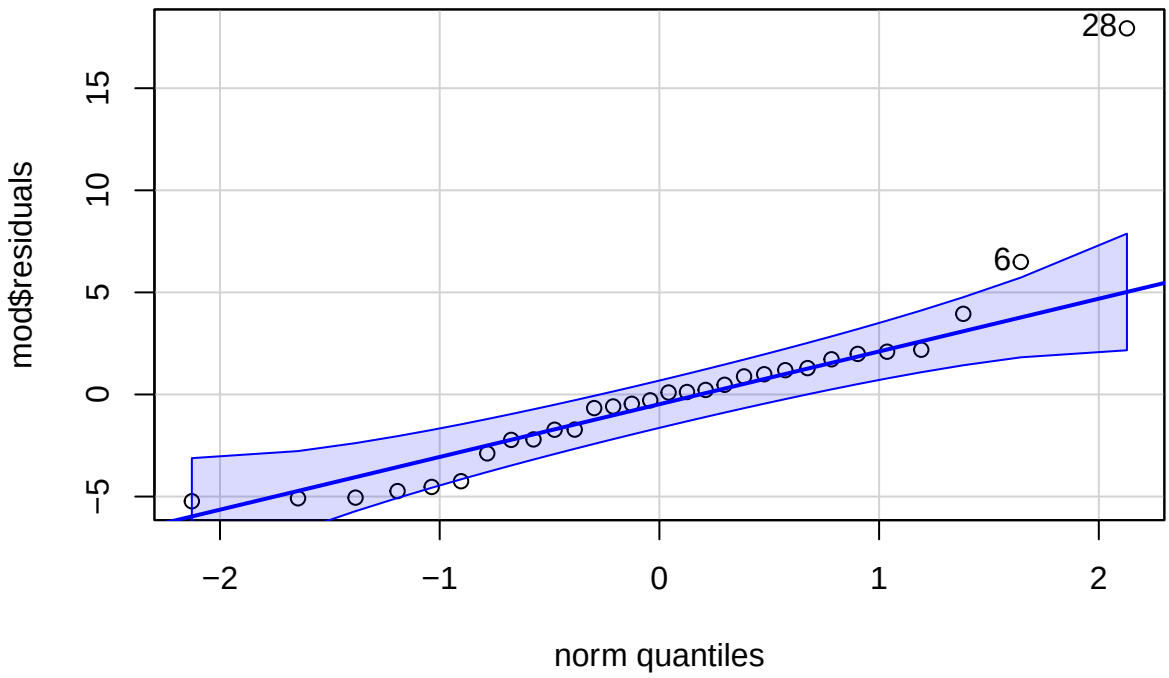
```
##
##  Bartlett test of homogeneity of variances
##
## data:  y by interaction(f1, f2)
## Bartlett's K-squared = 24.719, df = 5, p-value = 0.0001578
```

```r
print(res[[2]])
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  5  0.7451 0.5975
##       24
```

Como podemos ver la prueba de levene si detecta la igualdad de varianzas ya que estra prueba no es sensible al supuesto de normalidad. Ahora probamos el supuesto de normalidad.

```r
mod = lm(y~f1*f2, data = base)
qqPlot(mod$residuals)
```



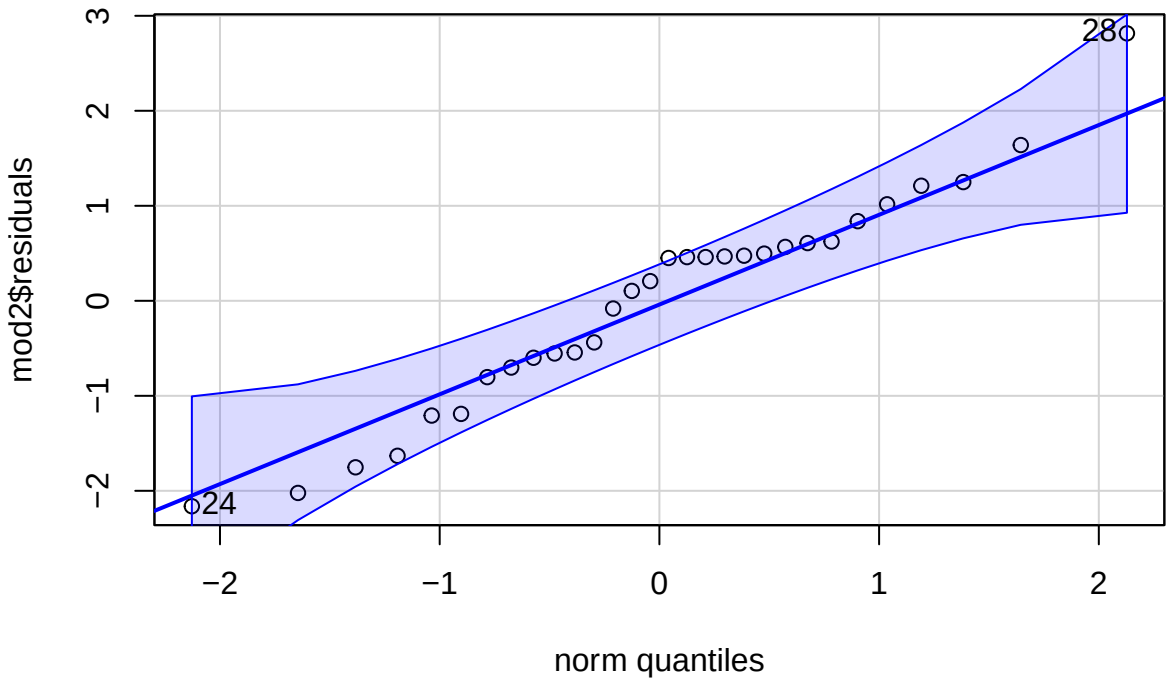

```
## [1] 28  6
```

```
shapiro.test(mod$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  mod$residuals
## W = 0.78966, p-value = 4.273e-05
```

Como esperabamos, se rechaza la hipotesis de normalidad. Pero si le aplicamos la operacion logaritmo a la variable de respuesta.

```
y1 = log(base$y)
mod2 = lm(y1~base$f1*base$f2)
qqPlot(mod2$residuals)
```




```
## [1] 28 24
```

```
shapiro.test(mod2$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  mod2$residuals
## W = 0.96765, p-value = 0.4772
```

Como podemos ver ahora si se cumple el supuesto de normalidad apesar de que la prueba diga lo contrario ya que al ser tan pocos datos esta pierde potencia.