Taller de problemas resueltos inferencia

Ricardo Alberich

Contenidos

1 Taller Problemas resueltos: Estadística Inferencial

Se trata de resolver los siguientes problemas y cuestiones en un fichero Rmd y su salida en un informe en html, word o pdf.

1.1 Problema 1: Contraste de parámetros de dos muestras.

Queremos comparar los tiempos de realización de un test entre estudiantes de dos grados G1 y G2, y determinar si es verdad que los estudiantes de G1 emplean menos tiempo que los de G2. No conocemos σ_1 y σ_2. Disponemos de dos muestras independientes de cuestionarios realizados por estudiantes de cada grado, $n_1 = n_2 = 50$.

Los datos están en https://github.com/joanby/estadistica-inferencial/, en la carpeta `datasets` en dos ficheros `grado1.txt` y `grado2.txt`.

Para bajarlos utilizad la dirección del los ficheros `raw` que se muestran en el siguiente código

```
G1=read.csv(
  "https://raw.githubusercontent.com/joanby/estadistica-inferencial/master/datasets/grado1.txt",
            header=TRUE)$x
G2=read.csv(
  "https://raw.githubusercontent.com/joanby/estadistica-inferencial/master/datasets/grado2.txt",
  header=TRUE)$x

n1=length(na.omit(G1))
n2=length(na.omit(G2))
media.muestra1=mean(G1,na.rm=TRUE)
media.muestra2=mean(G2,na.rm=TRUE)
```

```
desv.tip.muestra1=sd(G1,na.rm=TRUE)
desv.tip.muestra2=sd(G2,na.rm=TRUE)
```

Calculamos las medias y las desviaciones típicas muestrales de los tiempos empleados para cada muestra. Los datos obtenidos se resumen en la siguiente tabla:

$$\begin{array}{llll} n_1 & = 50, & n_2 & = 50 \\ \overline{x}_1 & = 9.7592926, & \overline{x}_2 & = 11.4660825 \\ \tilde{s}_1 & = 1.1501225, & \tilde{s}_2 & = 1.5642932 \end{array}$$

Se pide:

1. Comentad brevemente el código de R explicando que hace cada instrucción.
2. Contrastad si hay evidencia de que las notas medias son distintas entre los dos grupos. En dos casos considerando las varianzas desconocidas pero iguales o desconocidas pero distintas. Tenéis que hacer el contraste de forma manual y con funciones de `R` y resolver el contrate con el p-valor.
3. Calculad e interpretar los intervalos de confianza para la diferencia de medias asociados a los dos test anteriores.
4. Comprobad con el test de Fisher y el de Levene si las varianza de las dos muestras son iguales contra que son distintas. Tenéis que resolver el test de Fisher con `R` y de forma manual y el test de Levene con `R` y decidir utilizando el p-valor.

1.1.1 Solución

Apartado 1. El cogido R carga en las variables G1 y G2 la variables `x` de dos data frames de un servidor en github y por lo tanto hemos tenido que pasar la url del fichero original o *raw*.

Luego calcula los estadísticos básicos para realizar las siguientes preguntas. Para los tamaños muestrales n_1 y n_2 se omiten los valores `NA` antes de asignar la `length` de los arrays. También se calculan las medias y las desviaciones típicas muestrales omitiendo (si es que hay) los valores no disponibles.

Apartado 2. Denotemos por μ_1 y μ_2 las medias de las notas de los grupos 1 y 2 respectivamente. El contraste que se pide es

$$\begin{cases} H_0 : \mu_1 = \mu_2 \\ H_1 : \mu_1 \neq \mu_2 \end{cases}$$

Estamos en un diseño de comparación de medias de dos grupos con dos muestras independientes de tamaño 50 que es grande. Tenemos dos casos varianzas desconocidas pero iguales y varianzas desconocidas pero distintas. Las funciones de R del contraste para estos casos son:

Varianzas iguales

```
# test para varianzas iguales
t.test(G1,G2,var.equal = TRUE,alternative = "two.sided")
```

```
##
##  Two Sample t-test
##
## data:  G1 and G2
## t = -6.2159, df = 98, p-value = 0.00000001248
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.251691 -1.161889
## sample estimates:
## mean of x mean of y
##  9.759293 11.466083
```

Varianzas distintas

```
# test para varianzas distintas
t.test(G1,G2,var.equal = FALSE,alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  G1 and G2
## t = -6.2159, df = 89.996, p-value = 0.00000001562
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.252298 -1.161282
## sample estimates:
## mean of x mean of y
##  9.759293 11.466083
```

El p-valor en ambos casos es muy pequeño así que la muestra no aporta evidencias rechazar la hipótesis nula las medias son iguales contra que son distintas.

Veamos el cálculo manual.

Varianzas desconocidas pero iguales, n_1 y n_2 grande

Si suponemos que $\sigma_1 = \sigma_2$, el estadístico de contraste es

$$t0 = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{(\frac{1}{n_1} + \frac{1}{n_2}) \cdot \frac{((n_1-1)\widetilde{S}_1^2 + (n_2-1)\widetilde{S}_2^2)}{(n_1+n_2-2)}}} = \frac{9.7592926 - 11.4660825}{\sqrt{(\frac{1}{50} + \frac{1}{50}) \cdot \frac{((50-1)1.1501225^2 + (50-1)1.5642932^2)}{(50+50-2)}}}$$

```
t0=(media.muestra1-media.muestra2)/
  sqrt((1/n1+1/n2)*
((n1-1) *desv.tip.muestra1^2+(n2-1)*desv.tip.muestra2^2)/(n1+n2-2))
t0
```

```
## [1] -6.215931
```

operando obtenemos que $t0 = -6.215931$. y sabemos que sigue una distribución t-Student $t_{n_1+n_2-2} = t_{98}$. Para este hipótesis alternativa el p-valor es

$2 \cdot P(t_{98} > |-6.2159314|)$, lo calculamos con R

```
t0=(media.muestra1-media.muestra2)/
  sqrt((1/n1+1/n2)*
((n1-1)*desv.tip.muestra1^2+(n2-1)*desv.tip.muestra2^2)/(n1+n2-2))
t0
```

```
## [1] -6.215931
```

```
n1
```

```
## [1] 50
```

```
n2
```

```
## [1] 50
```

```
2*(1-pt(abs(t0),df=n1+n2-2)) # calculo la probabilidad del complementario
```

```
## [1] 0.00000001247958
```

```r
2*pt(abs(t0),df=n1+n2-2,lower.tail = FALSE)# calcula el área la cola  superior
```

```
## [1] 0.00000001247958
```

Varianzas desconocidas pero distintas, n_1 y n_2 grande

Si suponemos que $\sigma_1 \neq \sigma_2$, el estadístico de contraste es $t0 = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{\widetilde{S}_1^2}{n_1} + \frac{\widetilde{S}_2^2}{n_2}}} \sim t_f$, que, cuando $\mu_1 = \mu_2$, tiene distribución (aproximadamente, en caso de muestras grandes) t_f con

$$f = \frac{\left(\frac{\widetilde{S}_1^2}{n_1} + \frac{\widetilde{S}_2^2}{n_2}\right)^2}{\frac{1}{n_1 - 1}\left(\frac{\widetilde{S}_1^2}{n_1}\right)^2 + \frac{1}{n_2 - 1}\left(\frac{\widetilde{S}_2^2}{n_2}\right)^2}$$

Calculamos el estadístico y los grados de libertad con R

```r
t0=(media.muestra1-media.muestra2)/
  sqrt(desv.tip.muestra1^2/n1+desv.tip.muestra2^2/n2)
#calculo el valor dentro del floor que es el que utiliza la función t.test de R para este caso.
t0
```

```
## [1] -6.215931
```

```r
f=(desv.tip.muestra1^2/n1+desv.tip.muestra2^2/n2)^2/
  ((1/(n1-1))*(desv.tip.muestra1^2/n1)^2+
     (1/(n2-1))*(desv.tip.muestra2^2/n2)^2)
f
```

```
## [1] 89.99613
```

El p valor es

```r
# el p-valor de la función t.test de  R
2*(pt(abs(t0),f,lower.tail = FALSE))
```

```
## [1] 0.00000001562353
```

Apartado 3

Los intervalos de confianza al nivel del 95% los podemos obtener así

```r
t.test(G1,G2,var.equal=TRUE,
       alternative ="two.sided",
       conf.level=0.95)$conf.int
```

```
## [1] -2.251691 -1.161889
## attr(,"conf.level")
## [1] 0.95
```

```r
t.test(G1,G2,var.equal=FALSE,
       alternative="two.sided",
       conf.level = 0.95)$conf.int
```

```
## [1] -2.252298 -1.161282
## attr(,"conf.level")
## [1] 0.95
```

Son similares, podemos asegurar que la diferencia de medias se encuentra $-2.25 < \mu_1 - \mu_2 < -1.16$ al nivel del 95 el grupo 2 tiene una media entre 2.25 y 1.16 puntos mayor que el grupo 2 aproximadamente.

Apartado 4 El test que nos piden es el de igualdad de varianzas

$$\begin{cases} H_0: & \sigma_1^2 = \sigma_2^2 \\ H_1: & \sigma_1^2 \neq \sigma_2^2 \end{cases}.$$

El test de Fisher de igualdad de varianzas

```
var.test(G1,G2,alternative ="two.sided" )
```

```
##
##  F test to compare two variances
##
## data:  G1 and G2
## F = 0.54057, num df = 49, denom df = 49, p-value = 0.03354
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.3067606 0.9525862
## sample estimates:
## ratio of variances
##            0.54057
```

Obtenemos un p-valor alto no podemos rechazar la igualdad de varianzas.

De forma manual el estadístico de este test sabemos que es

$$f_0 = \frac{\tilde{S_1}^2}{\tilde{S_2}^2} = \frac{1.3227818}{2.4470131} = 0.54057.$$

Que sigue una ley de distribución de Fisher y el $p_$valor es $\min\{2 \cdot P(F_{n_1-1,+n_2-1} \leq f_0), 2 \cdot P(F_{n_1-1,+n_2-1} \geq f_0)$.

que con R es

```
n1
```

```
## [1] 50
```

```
n2
```

```
## [1] 50
```

```
f0=desv.tip.muestra1^2/desv.tip.muestra2^2
f0
```

```
## [1] 0.54057
```

```
pvalor=min(2*pf(f0,n1-1,n2-2),2*pf(f0,n1-1,n2-2,lower.tail = FALSE))
pvalor
```

```
## [1] 0.03420609
```

Obtenemos los mismos resultados que con la función `var.test`.

El test de Levene con R tiene las mismas hipótesis que el anterior

```
library(car,quietly = TRUE)# pongo quietly para que quite avisos
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
notas=c(G1,G2)
grupo=as.factor(c(rep(1,length(G1)),rep(2,length(G1))))
leveneTest(notas~grupo)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  1  1.8029 0.1825
##       98
```

El p-valor obtenido es alto así que el test de Levene no aporta evidencias contra la igualdad de varianzas entre las notas de los dos grupos.

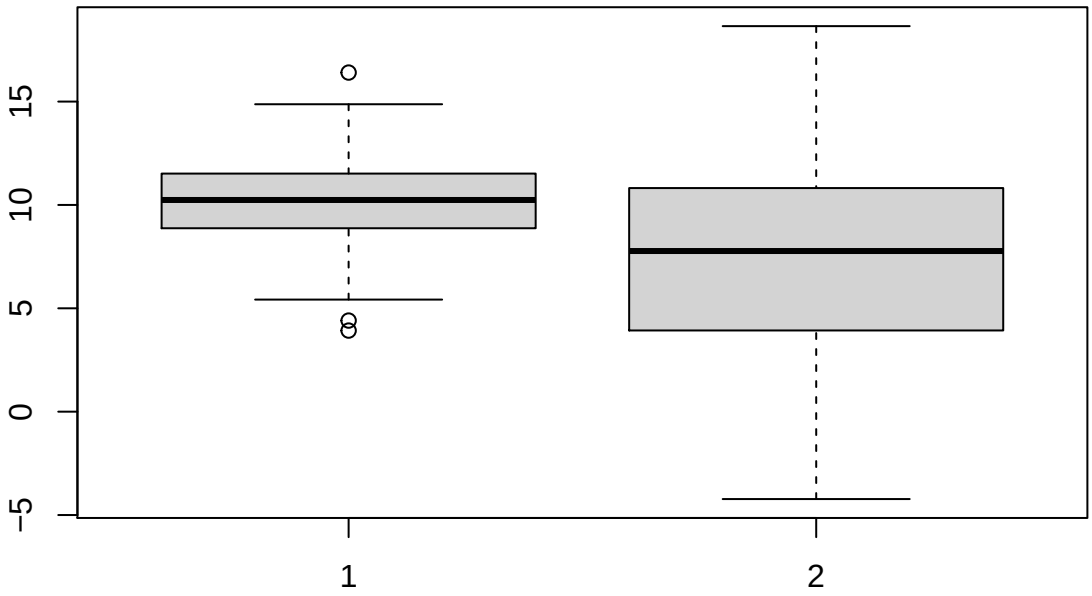
1.2 Problema 2 : Contraste dos muestras

Simulamos dos muestras con las funciones siguientes

```
set.seed(2020)
x1=rnorm(100,mean = 10,sd=2)
x2=rnorm(100,mean = 8,sd=4)
```

Dibujamos estos gráficos

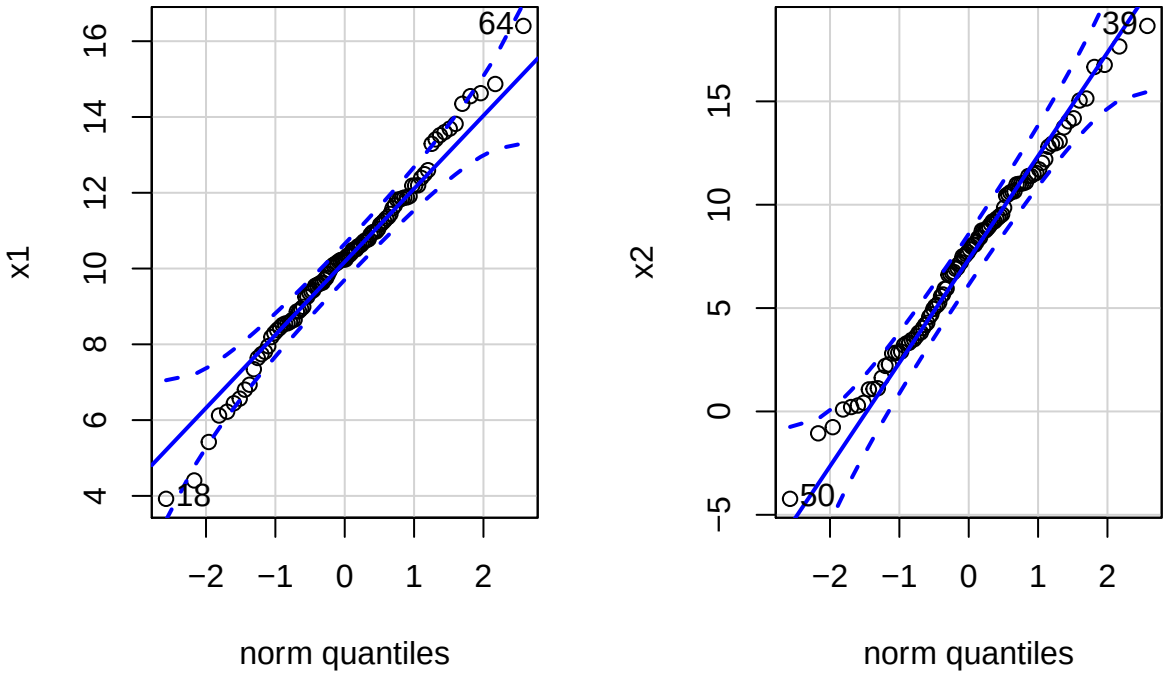
```
boxplot(x1,x2)
```




```
library(car)
par(mfrow=c(1,2))
qqPlot(x1)
```

```
## [1] 18 64
```

```
qqPlot(x2)
```




```
## [1] 50 39
```

```
par(mfrow=c(1,1))
```

Realizamos algunos contrastes de hipótesis de igual de medias entre ambas muestras

```
t.test(x1,x2,var.equal = TRUE,alternative = "greater")
```

```
##
##  Two Sample t-test
##
## data:  x1 and x2
## t = 5.3009, df = 198, p-value = 0.0000001531
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  1.844757      Inf
## sample estimates:
## mean of x mean of y
## 10.217784  7.537402
```

```
t.test(x1,x2,var.equal = FALSE,alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  x1 and x2
## t = 5.3009, df = 144.56, p-value = 0.0000004221
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.680966 3.679797
## sample estimates:
```

```
## mean of x mean of y
## 10.217784  7.537402
```

```
t.test(x1,x2,var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  x1 and x2
## t = 5.3009, df = 198, p-value = 0.0000003061
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  1.683238 3.677526
## sample estimates:
## mean of x mean of y
## 10.217784  7.537402
```

Se pide

1. ¿Cuál es la distribución y los parámetros de las muestras generadas?
2. ¿Qué muestran y cuál es la interpretación de los gráficos?
3. ¿Qué test contrasta si hay evidencia a favor de que las medias poblacionales de las notas en cada grupo sean distintas? Di qué código de los anteriores resuelve este test.
4. Para el test del apartado anterior dad las hipótesis nula y alternativa y redactar la conclusión del contraste.

1.2.1 Solución

Apartado 1

Se generan dos muestras de poblaciones normales de medias 10 y 8 y desviaciones típicas 2 y 4.

Apartado 2 El primer gráficos es un diagrama de caja (*boxplot*) que compara las distribuciones de los datos. Vemos que efectivamente la muestra 1 tiene una caja y unos bigotes más comprimidos que la muestra 2 así que la primera tiene menos varianza. Vemos que los valores medianos de la muestra 1 son más grandes que los de la muestra 2. Recordemos que la distribución normal es simétrica por lo que la media y la mediana coinciden. La muestra 1 tiene valores atípicos en la parte superior 1 y en la inferior parece que 2.

El segundo gráfico es un gráfico cuantil-cuantil o qqplot de normalidad. Compara los cuantiles muestrales con los teóricos de una normal y nos da un intervalo de confianza para esas observaciones.

Vemos que los cuantiles teóricos no difieren excesivamente de los muestrales en cada una de las muestras y que muy pocos valores se escapan de los intervalos de confianza esperados en el caso de normalidad. Así que no hay motivo para pensar que las distribuciones de ambas muestras proceden de poblaciones normales.

Apartado 3

El código es

```
t.test(x1,x2,var.equal = TRUE,alternative = "greater")
t.test(x1,x2,var.equal = FALSE,alternative = "two.sided")
t.test(x1,x2,var.equal = TRUE)
```

El primer test contrasta para muestras independientes supuestas varianzas desconocidas pero iguales $H_0; \mu_1 = \mu_2$ contra $H_1 : \mu_1 > \mu_2$. **Así que este TEST NO ES**

El segundo test contrasta para muestras independientes supuestas varianzas desconocidas pero iguales $H_0; \mu_1 = \mu_2$ contra $H_1 : \mu_1 \neq \mu_2$. **Así que este TEST SÍ PUEDE SER** contrasta contra medias distintas para el caso de varianzas distintas.

El tercer test contrasta para muestras independientes supuestas varianzas desconocidas pero iguales $H_0; \mu_1 = \mu_2$ contra $H_1 : \mu_1 > \mu_2$ pues la opción por defecto de la función. **Así que este TEST SÍ PUEDE SER** contra medias distintas para el caso de varianzas iguales.

Apartado 4 El contrastes es

$$\begin{cases} H_0 : & \mu_1 = \mu_2 \\ H_1 : & \mu_1 \neq \mu_2 \end{cases}$$

En los dos últimos test los p-valores son muy muy pequeños así que hay evidencias en contra de la igualdad de medias entre las dos muestras. Además claramente los intervalos de confianza no contienen al cero.

1.3 Problema 3 : Bondad de ajuste. La ley de Benford

La ley de Benford es una distribución discreta que siguen las frecuencias de los primero dígitos significativos (de 1 a 9) de algunas series de datos curiosas.

Sea una v.a. X con dominio $D_X = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ diremos que sigue una ley de Benford si

$$P(X = x) = \log_{10}\left(1 + \frac{1}{x}\right) \text{ para } x \in \{1, 2, 3, 4, 5, 6, 7, 8, 9\}\,.$$

Concretamente lo podemos hacer así

```
prob=log10(1+1/c(1:9))
prob
```

```
## [1] 0.30103000 0.17609126 0.12493874 0.09691001 0.07918125 0.06694679 0.05799195
## [8] 0.05115252 0.04575749
```

```
df=data.frame(rbind(prob))
# Y hacemos una bonita tabla
colnames(df)=paste("Díg.",c(1:9),sep =" ")
knitr::kable(df,format ='markdown')
```

	Díg. 1	Díg. 2	Díg. 3	Díg. 4	Díg. 5	Díg. 6	Díg. 7	Díg. 8	Díg. 9
prob	0.30103	0.1760913	0.1249387	0.09691	0.0791812	0.0669468	0.0579919	0.0511525	0.0457575

En general esta distribución se suele encontrar en tablas de datos de resultados de observaciones de funciones científicas, contabilidades, cocientes de algunas distribuciones ...

Por ejemplo se dice que las potencias de números enteros siguen esa distribución. Probemos con las potencias de 2. El siguiente código calcula las potencias de 2 de 1 a 1000 y extrae los tres primeros dígitos.

```
# R puede según qué versión pasar los enteros
# muy grande a reales. Para nuestros propósitos
# es suficiente para extraer
# los tres primeros dígitos.
muestra_pot_2_3digitos=str_sub(as.character(2^c(1:1000)),1,3)
head(muestra_pot_2_3digitos)
```

```
## [1] "2"  "4"  "8"  "16" "32" "64"
```

```
tail(muestra_pot_2_3digitos)
```

```
## [1] "334" "669" "133" "267" "535" "107"
```

```r
#Construimos un data frame con tres columnas que nos dan el primer,
#segundo y tercer dígito respectivamente.
df_digitos=data.frame(muestra_pot_2_3digitos,
                      primer_digito=as.integer(
                        substring(muestra_pot_2_3digitos, 1, 1)),
                      segundo_digito=as.integer(
                        substring(muestra_pot_2_3digitos, 2, 2)),
                      tercer_digito=as.integer(
                        substring(muestra_pot_2_3digitos, 3, 3)))
head(df_digitos)
```

```
##   muestra_pot_2_3digitos primer_digito segundo_digito tercer_digito
## 1                      2             2             NA            NA
## 2                      4             4             NA            NA
## 3                      8             8             NA            NA
## 4                     16             1              6            NA
## 5                     32             3              2            NA
## 6                     64             6              4            NA
```

Notad que los NA en el segundo y el tercer dígito corresponden a número con uno o dos dígitos.

Se pide:

1. Contrastad con un test χ^2 que el primer dígito sigue una ley de Benford. Notad que el primer dígito no puede ser 0. Resolved manualmente y con una función de `R`.

2. Contrastad con un test χ^2 que el segundo dígito sigue una ley de uniforme discreta. Notad que ahora si puede ser 0. Resolved con funciones de `R`.

3. Contrastad con un test χ^2 que el tercer dígito sigue una ley de uniforme discreta. Notad que ahora si puede ser 0. Resolved con manualmente calculado las frecuencias esperadas y observadas, el estadístico de contraste y el p-valor utilizando `R`. Comprobad que vuestros resultados coinciden con los de la función de `R` que calcula este contraste.

4. Dibujad con `R` para los apartados 1 y 2 los diagramas de frecuencias esperados y observados. Comentad estos gráficos

1.3.1 Solución

Apartado 1 El contraste que nos piden es

$$\begin{cases} H_0: & \text{El primer dígito de las 1000 primeras potencias de 2 sigue una ley Benford,} \\ H_1: & \text{sigue cualquier otra distribución.} \end{cases}$$

El siguiente código resuelve el tes de forma manual calculando el estadístico y el p-valor

```r
prob=log10(1+1/(1:9))
prob_benford=prob
n=1000
frec_esp_benford=n*prob_benford
frec_esp_benford
```

```
## [1] 301.03000 176.09126 124.93874  96.91001  79.18125  66.94679  57.99195
## [8]  51.15252  45.75749
```

```r
frec_obs_primer=table(df_digitos$primer_digito)
frec_obs_primer
```

```
##
##   1   2   3   4   5   6   7   8   9
## 301 176 125  97  79  69  56  52  45
```

```
chi2_est=sum((frec_obs_primer-frec_esp_benford)^2/frec_esp_benford)
chi2_est
```

```
## [1] 0.1585506
```

```
pchisq(chi2_est,9-1,lower.tail = FALSE)
```

```
## [1] 0.9999985
```

La función de R que resuelve el test es

```
chisq.test(frec_obs_primer,p=prob_benford)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  frec_obs_primer
## X-squared = 0.15855, df = 8, p-value = 1
```

Obtenemos los mismos resultados. El p valor es muy alto. No podemos rechazar que el primer dígito de las 1000 primeras potencias enteras de 2 siga una ley de distribución de Benford.

Apartado 2

El contraste que nos piden es

$$\begin{cases} H_0: & \text{El segundo dígito de las 1000 primeras potencias de 2 sigue una ley uniforme,} \\ H_1: & \text{sigue cualquier otra distribución.} \end{cases}$$

Procedemos de forma similar al caso anterior. De forma manual el cálculo es

```
prob_unif=rep(1/10,10)
prob_unif
```

```
##  [1] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
```

```
segundo_digito=na.omit(df_digitos$segundo_digito)
n=length(segundo_digito)
n
```

```
## [1] 997
```

```
frec_esp_uniforme=n*prob_unif
frec_esp_uniforme
```

```
##  [1] 99.7 99.7 99.7 99.7 99.7 99.7 99.7 99.7 99.7 99.7
```

```
frec_obs_segundo=table(segundo_digito)
frec_obs_segundo
```

```
## segundo_digito
##   0   1   2   3   4   5   6   7   8   9
## 121 112 109 108  98  95  94  91  83  86
```

```
chi2_est=sum((frec_obs_segundo-frec_esp_uniforme)^2/frec_esp_uniforme)
chi2_est
```

```
## [1] 13.64193
```

```
1-pchisq(chi2_est,10-1)
```

```
## [1] 0.1356449
```

```
pchisq(chi2_est,10-1,lower.tail = FALSE)
```

```
## [1] 0.1356449
```

Con la función `chisq.test` obtenemos los mismo resultados

```
chisq.test(frec_obs_segundo,p=prob_unif)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  frec_obs_segundo
## X-squared = 13.642, df = 9, p-value = 0.1356
```

Para el segundo dígito con un p-valor de 0.1356 no podemos rechazar que el segundo dígito siga una ley uniforme.

Apartado 3

El contraste que nos piden es

$$\begin{cases} H_0: & \text{El tercer dígito de las 1000 primeras potencias de 2 sigue una ley uniforme,} \\ H_1: & \text{sigue cualquier otra distribución.} \end{cases}$$

Procedemos de forma similar al caso anterior. De forma manual el cálculo es

```
prob_unif=rep(1/10,10)
prob_unif
```

```
##  [1] 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1
```

```
tercer_digito=na.omit(df_digitos$tercer_digito)
n=length(tercer_digito)
n
```

```
## [1] 994
```

```
frec_esp_uniforme=n*prob_unif
frec_esp_uniforme
```

```
##  [1] 99.4 99.4 99.4 99.4 99.4 99.4 99.4 99.4 99.4 99.4
```

```
frec_obs_tercer=table(tercer_digito)
frec_obs_tercer
```

```
## tercer_digito
##   0   1   2   3   4   5   6   7   8   9
##  96 101  88 116  97  90 110  92  95 109
```

```
chi2_est=sum((frec_obs_tercer-frec_esp_uniforme)^2/frec_esp_uniforme)
chi2_est
```

```
## [1] 7.971831
```

```
1-pchisq(chi2_est,10-1)
```

```
## [1] 0.5369875
```

```r
pchisq(chi2_est,10-1,lower.tail = FALSE)
```

```
## [1] 0.5369875
```

Con la función `chisq.test` obtenemos los mismo resultados

```r
chisq.test(frec_obs_tercer,p=prob_unif)
```

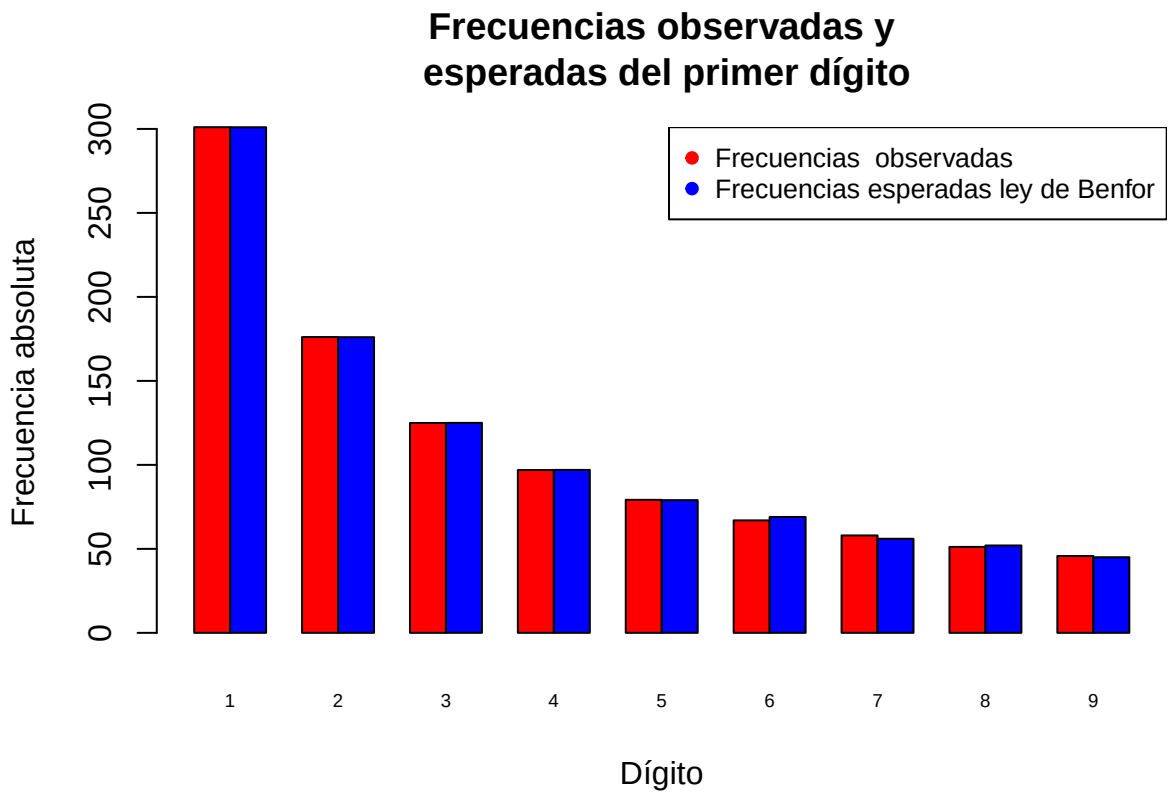
```
##
##  Chi-squared test for given probabilities
##
## data:  frec_obs_tercer
## X-squared = 7.9718, df = 9, p-value = 0.537
```

Para el segundo dígito con un p-valor alto 0.537 no podemos rechazar que el tercer dígito siga una ley uniforme.

Apartado 4

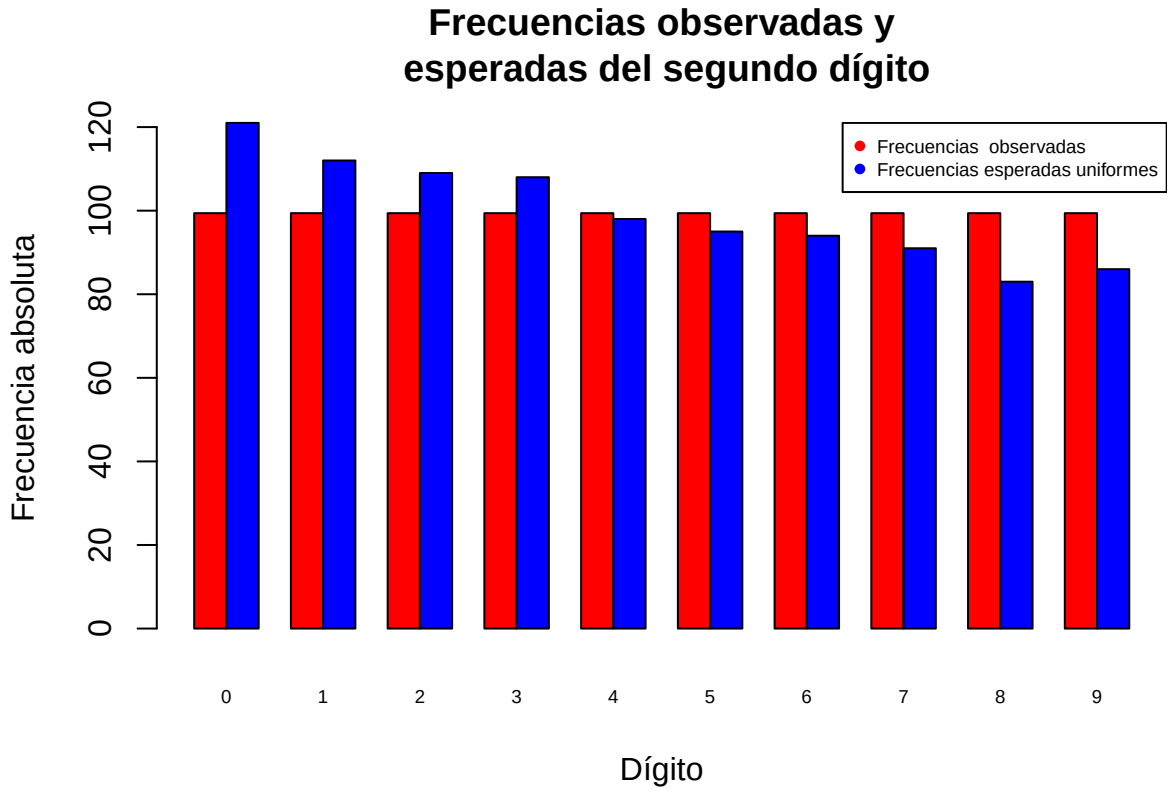
Una gráfica comparando las frecuencias

```r
barplot(rbind(frec_esp_benford,frec_obs_primer),
        beside=TRUE,col=c("red","blue"),
        main="Frecuencias observadas y\n esperadas del primer dígito",
        cex.names =0.6,xlab="Dígito",ylab="Frecuencia absoluta")
legend("topright",legen=c("Frecuencias  observadas",
                          "Frecuencias esperadas ley de Benfor"),pch=19,col=c("red","blue"),
       cex=0.8)
```




```r
barplot(rbind(frec_esp_uniforme,frec_obs_segundo),
        beside=TRUE,col=c("red","blue"),
        main="Frecuencias observadas y\n esperadas del segundo dígito",
```

```
       cex.names =0.6,xlab="Dígito",ylab="Frecuencia absoluta")
legend("topright",legen=c("Frecuencias  observadas",
                          "Frecuencias esperadas uniformes"),pch=19,col=c("red","blue"),
       cex=0.6)
```



1.4 Problema 4 : Homegeneidad e independencia

Queremos analiza los resultados de aprendizaje con tres tecnologías. Para ello se seleccionan 3 muestras de 50 estudiantes y se les somete a evaluación después de un curso.

```
set.seed(2020)
nota=factor(sample(c(1,2,3,4),p=c(0.1,0.4,0.3,0.2),replace=TRUE,size=150),
            labels=c("S","A","N","E"))
tecnologia=rep(c("Mathematica","R","Python"),each=50)
frec=table(nota,tecnologia)
frec
```

```
##     tecnologia
## nota Mathematica Python  R
##    S           7      6  2
##    A          18     15 22
##    N          15     20 18
##    E          10      9  8
```

```
col_frec=colSums(frec)
col_frec
```

```
## Mathematica      Python           R
##          50          50          50
```

```
row_frec=rowSums(frec)
row_frec
```

```
##  S  A  N  E
## 15 55 53 27
```

```
N=sum(frec)
teoricas=row_frec%*%t(col_frec)/N
teoricas
```

```
##      Mathematica   Python        R
## [1,]     5.00000  5.00000  5.00000
## [2,]    18.33333 18.33333 18.33333
## [3,]    17.66667 17.66667 17.66667
## [4,]     9.00000  9.00000  9.00000
```

```
dim(frec)
```

```
## [1] 4 3
```

```
dim(teoricas)
```

```
## [1] 4 3
```

```
sum((frec-teoricas)^2/teoricas)
```

```
## [1] 5.084658
```

```
chisq.test(table(nota,tecnologia))
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(nota, tecnologia)
## X-squared = 5.0847, df = 6, p-value = 0.533
```

Se pide

1. Discutid si hacemos un contraste de independencia o de homogeneidad de las distribuciones de las notas por tecnología. Escribid las hipótesis del contraste.
2. Interpretad la función `chisq.test` y resolved el contraste.
3. Interpretad `teoricas=row_frec%*%t(col_frec)/N` reproducid manualmente el segundo resultado de la primera fila.

1.4.1 Solución

Apartado 1

Podemos plantear dos hipótesis distintas. La primera es que la calificación obtenidas es independiente de la tecnología, la segunda es las distribuciones en las 4 clases de notas son las mismas para cada tecnología. En el primer caso es un contraste de independencia y en el segundo es de homogeneidad.

Podemos optar en este caso por cualquiera de los dos. Yo he optado por el de independencia

así que la hipótesis H_0: la nota obtenida es independientes ce la tecnología del curso, contra que no lo es.

Apartado 2 Para el contraste de independencia (o el de homogeneidad) hay que pasar una tabla de contingencia a la función `chisq.test` eso es lo que hacemos

```
table(nota,tecnologia)
```

```
##     tecnologia
## nota Mathematica Python  R
##    S           7      6  2
##    A          18     15 22
##    N          15     20 18
##    E          10      9  8
```

```
chisq.test(table(nota,tecnologia))
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(nota, tecnologia)
## X-squared = 5.0847, df = 6, p-value = 0.533
```

El p-valor obtenido es grande así que no podemos rechazar la hipótesis nula las calificaciones obtenidas no dependen del programa utilizado en el curso.

Apartado 3

La expresión `teoricas=row_frec%*%t(col_frec)/N`

```
row_frec
```

```
##  S  A  N  E
## 15 55 53 27
```

```
col_frec
```

```
## Mathematica      Python           R
##          50          50          50
```

```
row_frec%*%t(col_frec)
```

```
##      Mathematica Python    R
## [1,]         750    750  750
## [2,]        2750   2750 2750
## [3,]        2650   2650 2650
## [4,]        1350   1350 1350
```

```
row_frec%*%t(col_frec)/N
```

```
##      Mathematica   Python        R
## [1,]     5.00000  5.00000  5.00000
## [2,]    18.33333 18.33333 18.33333
## [3,]    17.66667 17.66667 17.66667
## [4,]     9.00000  9.00000  9.00000
```

Así que `row_frec%*%t(col_frec)` calcula el producto de las frecuencia marginal por filas y columnas.

Por ejemplo la segunda fila es $2750 = 50 \cdot 55$ en los tres casos

Por último al dividir por $N = 150$ el tamaño total de las muestra obtenemos las frecuencias esperadas en el caso de independencia $2750/150 = 18.33333$.

1.5 Problema 5 : ANOVA notas numéricas de tres grupos.

El siguiente código nos da las notas numéricas (variable `nota`) de los mismos ejercicios para tres tecnologías en tres muestra independientes de estudiantes de estas tres tecnologías diferentes

```r
head(nota)
```

```
## [1] 79.424303 77.538709 42.549421 41.739852  0.086642 88.014337
```

```r
library(nortest)
lillie.test(nota[tecnologia=="Mathematica"])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nota[tecnologia == "Mathematica"]
## D = 0.08739, p-value = 0.4436
```

```r
lillie.test(nota[tecnologia=="R"])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nota[tecnologia == "R"]
## D = 0.082139, p-value = 0.5449
```

```r
lillie.test(nota[tecnologia=="Python"])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nota[tecnologia == "Python"]
## D = 0.089681, p-value = 0.4019
```

```r
lillie.test(nota)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nota
## D = 0.056381, p-value = 0.2885
```

```r
bartlett.test(nota~tecnologia)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  nota by tecnologia
## Bartlett's K-squared = 0.50309, df = 2, p-value = 0.7776
```

```r
library(car)
leveneTest(nota~as.factor(tecnologia))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   2  0.3881  0.679
##       147
```

```r
sol_aov=aov(nota~as.factor(tecnologia))
sol_aov
```

```
## Call:
##    aov(formula = nota ~ as.factor(tecnologia))
##
```

```
## Terms:
##                 as.factor(tecnologia) Residuals
## Sum of Squares                 837.39 123445.06
## Deg. of Freedom                     2       147
##
## Residual standard error: 28.97865
## Estimated effects may be unbalanced
```

Del `summary(sol_aov)` os damos la salida a falta de algunos de los valores

```
> summary(sol_aov)
                     Df Sum Sq Mean Sq F value Pr(>F)
as.factor(tecnologia) ---    837   418.7   ---  ---
Residuals             --- 123445   839.8
```

```r
pairwise.t.test(nota,as.factor(tecnologia),
                p.adjust.method = "none")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  nota and as.factor(tecnologia)
##
##        Mathematica Python
## Python 0.35        -
## R      0.89        0.43
##
## P value adjustment method: none
```

```r
pairwise.t.test(nota,as.factor(tecnologia),
                p.adjust.method = "bonferroni")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  nota and as.factor(tecnologia)
##
##        Mathematica Python
## Python 1           -
## R      1           1
##
## P value adjustment method: bonferroni
```

Se pide

1. ¿Podemos asegurar que la muestras son normales en cada grupo? ¿y son homocedásticas? Sea cual sea la respuesta justificad que parte del código la confirma.
2. La función `aov` que test calcula. Escribid formalmente la hipótesis nula y la alternativa.
3. Calcula la tabla de ANOVA y resuelve el test.
4. ¿Qué contrates realiza la función `pairwise.t.test`? Utilizando los resultados anteriores aplicad e interpretad los contrates del apartado anterior utilizando el ajuste de Holm.

1.5.1 Solución

Apartado 1 Este código es el que pasa un test de normalidad con la función `lillie.test` para la distribución de notas en cada una de las tecnologías

```r
head(nota)
```

```
## [1] 79.424303 77.538709 42.549421 41.739852  0.086642 88.014337
```

```r
library(nortest)
lillie.test(nota[tecnologia=="Mathematica"])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nota[tecnologia == "Mathematica"]
## D = 0.08739, p-value = 0.4436
```

```r
lillie.test(nota[tecnologia=="R"])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nota[tecnologia == "R"]
## D = 0.082139, p-value = 0.5449
```

```r
lillie.test(nota[tecnologia=="Python"])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  nota[tecnologia == "Python"]
## D = 0.089681, p-value = 0.4019
```

Como se ve los p-valores son altos en el pero de los caso es mayor que 0.4 no podemos rechazar de las notas numéricas en cada uno de los tres casos.

Para la homogeneidad de las varianzas en las tres poblaciones hacemos un `bartlett.testo` un LeveneTest.

```r
bartlett.test(nota~tecnologia)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  nota by tecnologia
## Bartlett's K-squared = 0.50309, df = 2, p-value = 0.7776
```

```r
library(car)
leveneTest(nota~as.factor(tecnologia))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   2  0.3881  0.679
##       147
```

Nos salen p-valores del orden de 0.7 o 0.6 por lo que no podemos rechazar la homocedasticidad de las tres varianzas.

Apartado 2

La función `aov` calcula un test de igualdad de medias de las notas numéricas en los tres cursos.

$$\begin{cases} H_0: & \mu_{Mathematica} = \mu_{Python} = \mu_R \\ H_1: & \text{no todas las medias son iguales.} \end{cases}$$

Apartado 3

```
sol_aov=aov(nota~as.factor(tecnologia))
sol_aov
```

```
## Call:
##    aov(formula = nota ~ as.factor(tecnologia))
##
## Terms:
##                 as.factor(tecnologia) Residuals
## Sum of Squares                 837.39 123445.06
## Deg. of Freedom                     2       147
##
## Residual standard error: 28.97865
## Estimated effects may be unbalanced
```

```
summary(sol_aov)
```

```
##                         Df Sum Sq Mean Sq F value Pr(>F)
## as.factor(tecnologia)    2    837   418.7   0.499  0.608
## Residuals              147 123445   839.8
```

EL p-valor es muy grade 0.608 no podemos rechazar la igualdad de medias.

Apartado 4

La función `pairwise.t.test` compara las medias dos a dos en este caso hay 4 comparaciones por pares. Para ajustar el nivel de significación tenemos que recurrir a alguno de los métodos de ajuste del p-valor que se pasan con el parámetro `p.adjust.method` en este caso nos piden que utilicemos el método de Holm que tenemos que calcular pues no figura en el código del enunciado.

```
pairwise.t.test(nota,as.factor(tecnologia),p.adjust.method = "holm")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  nota and as.factor(tecnologia)
##
##        Mathematica Python
## Python 1           -
## R      1           1
##
## P value adjustment method: holm
```

Los tres p-valores son prácticamente son 1 con el ajuste del método de Holm; o podemos rechazar las igual de medias dos a dos.

1.6 Problema 6 : ANOVA Comparación de las tasas de interés para la compra de coches entre seis ciudades.

Consideremos el `data set newcar` accesible desde https://www.itl.nist.gov/div898/education/anova/newcar.dat de *Hoaglin, D., Mosteller, F., and Tukey, J. (1991). Fundamentals of Exploratory Analysis of Variance. Wiley, New York, page 71.*

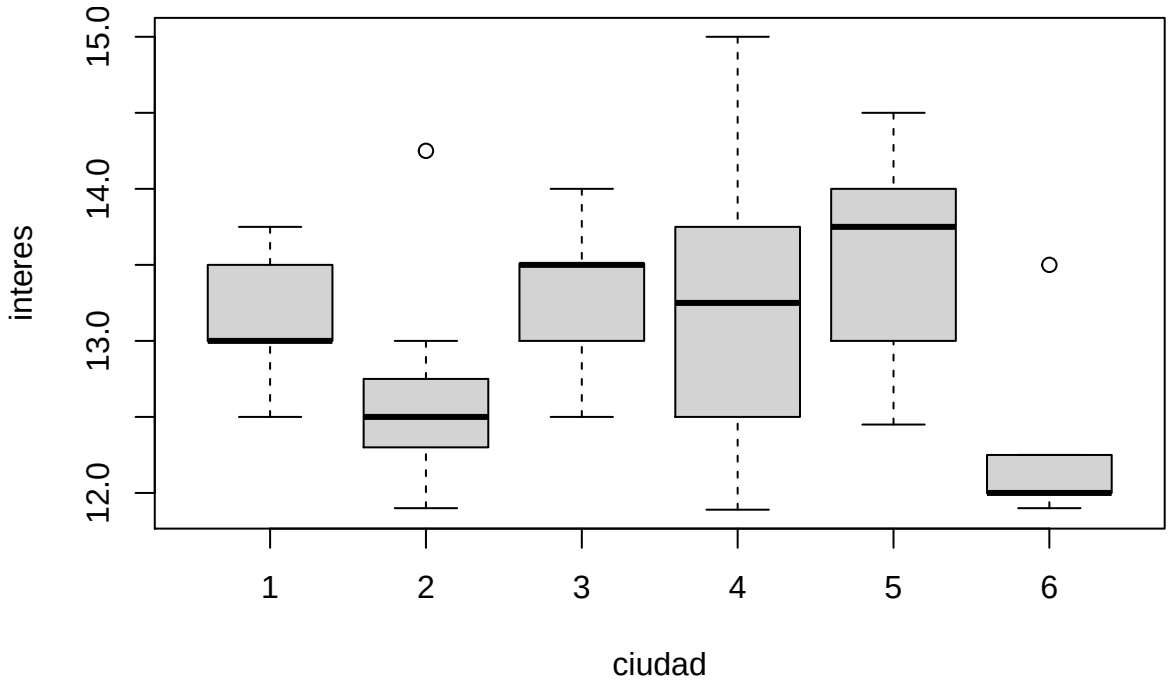
Este data set contiene dos columnas:

- Rate (interés): tasa de interés en la compra de coches a crédito
- City (ciudad) : la ciudad en la que se observó la tasa de interés para distintos concesionarios (codificada a enteros). Tenemos observaciones de 6 ciudades.

```r
datos_interes=read.table(
  "https://www.itl.nist.gov/div898/education/anova/newcar.dat",
  skip=25)
# salto las 25 primeras líneas del fichero,son un preámbulo que explica los datos.
names(datos_interes)=c("interes","ciudad")
str(datos_interes)
```

```
## 'data.frame':    54 obs. of  2 variables:
##  $ interes: num  13.8 13.8 13.5 13.5 13 ...
##  $ ciudad : int  1 1 1 1 1 1 1 1 1 2 ...
```

```r
boxplot(interes~ciudad,data=datos_interes)
```



Se pide:

1. Comentad el código y el diagrama de caja.
2. Se trata de contrastar si hay evidencia de que la tasas medias de interés por ciudades son distintas. Definid el ANOVA que contrasta esta hipótesis y especificar qué condiciones deben cumplir las muestras para poder aplicar el ANOVA.

3. Comprobad las condiciones del ANOVA con un test KS y un test de Levene (con código de `R`). Justificad las conclusiones.

4. Realizad el contraste de ANOVA (se cumplan las condiciones o no) y redactar adecuadamente la conclusión. Tenéis que hacedlo con funciones de `R`.

5. Se acepte o no la igualdad de medias realizar las comparaciones dos a dos con ajustando los p-valor tanto por Bonferroni como por Holm al nivel de significación $\alpha = 0.1$. Redactad las conclusiones que se obtienen de las mismas.

1.6.1 Solución

Apartado 1

El código del enunciado nos carga los datos de una web, tenemos que pasar el parámetro skip=25 para que se salte las 25 primeras lineas del fichero de texto que son un preámbulo que explica los datos.

En el diagrama de caja vemos que las medias las distribuciones de la `Rate` por ciudad son muy distintas, no parecen tener ni medias ni varianzas iguales

Apartado 2

Las condiciones para realizar un ANOVA son:

- Muestras independientes y aleatorias
- Distribución normal de la Rate $N(\mu_i, \sigma_i)$ para las seis ciudades $i = 1, 2, 3, 4, 5, 6$.
- homocedasticidad; igualdad de varianzas $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = \sigma_5 = \sigma_6$.

El ANOVA que se pide es

$$\begin{cases} H_0: & \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 = \mu_6 \\ H_1: & \text{no todas las medias son iguales.} \end{cases}$$

Apartado 3

El siguiente código realiza un test KS con corrección de Lillie para la normalidad de la variable Rate en cada una de la seis ciudades

```
library(nortest)
lillie.test(datos_interes$interes[datos_interes$ciudad==1])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  datos_interes$interes[datos_interes$ciudad == 1]
## D = 0.22384, p-value = 0.2163
```

```
lillie.test(datos_interes$interes[datos_interes$ciudad==2])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  datos_interes$interes[datos_interes$ciudad == 2]
## D = 0.22884, p-value = 0.1903
```

```
lillie.test(datos_interes$interes[datos_interes$ciudad==3])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  datos_interes$interes[datos_interes$ciudad == 3]
## D = 0.19145, p-value = 0.4459
```

```
lillie.test(datos_interes$interes[datos_interes$ciudad==4])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  datos_interes$interes[datos_interes$ciudad == 4]
## D = 0.11264, p-value = 0.9852
```

```
lillie.test(datos_interes$interes[datos_interes$ciudad==5])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
```

```
##
## data:  datos_interes$interes[datos_interes$ciudad == 5]
## D = 0.20021, p-value = 0.3743
```

```
lillie.test(datos_interes$interes[datos_interes$ciudad==6])
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  datos_interes$interes[datos_interes$ciudad == 6]
## D = 0.3494, p-value = 0.002236
```

No podemos rechazar la normalidad con el lillie.test en las 5 primeras ciudades, pero parece que la última está lejos de ser normal.

Ahora comprobemos que

$$\begin{cases} H_0: & \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \sigma_4^2 = \sigma_5^2 = \sigma_6^2 \\ H_1: & \text{no todas las varianzas son iguales.} \end{cases}$$

con el test de Levene (o el de Bartlett)

```
library(car)
print(leveneTest(datos_interes$interes~as.factor(datos_interes$ciudad)))
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  5  1.2797 0.2882
##       48
```

El test de levene nos da un p-valor superior a 0.28 aceptamos la igualdad de varianzas

Apartado 4

Resolvemos el ANOVA con el código siguiente

```
summary(aov(datos_interes$interes~as.factor(datos_interes$ciudad)))
```

```
##                               Df Sum Sq Mean Sq F value  Pr(>F)
## as.factor(datos_interes$ciudad)  5  10.95  2.1891   4.829 0.00117 **
## Residuals                       48  21.76  0.4533
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

El p-valor es muy bajo 0.00117 rechazamos la igualdad de las seis medias, al menos hay dos distintas.

Comprobamos por gusto el p-valor a partir de los datos del summary

```
Fest=2.1891/0.4533
Fest
```

```
## [1] 4.829252
```

```
1-pf(Fest,5,48)
```

```
## [1] 0.001174782
```

```
pf(Fest,5,48,lower.tail = FALSE)
```

```
## [1] 0.001174782
```

Apartado 5

Comparemos las medias dos a dos son 15 comparaciones

```
pairwise.t.test(datos_interes$interes,as.factor(datos_interes$ciudad),p.adjust.method = "holm")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  datos_interes$interes and as.factor(datos_interes$ciudad)
##
##   1      2      3      4      5
## 2 0.5781 -      -      -      -
## 3 1.0000 0.3330 -      -      -
## 4 1.0000 0.4651 1.0000 -      -
## 5 1.0000 0.0926 1.0000 1.0000 -
## 6 0.0353 1.0000 0.0148 0.0244 0.0028
##
## P value adjustment method: holm
```

Nos piden que decidamos con $\alpha = 0.1$, así que rechazaremos la igualdad de medias de todas las comparaciones con p-valor inferior a 0.1.

Tenemos que rechazar la igualdad de medias entre la ciudad 2 con la 5 y la de la ciudad 6 con las ciudades 1, 3, 4 y 5.

1.7 Problema 7: Cuestiones cortas

- Cuestión 1: Supongamos que conocemos el p-valor de un contraste. Para que valores de nivel de significación α RECHAZAMOS la hipótesis nula.
- Cuestión 2: Hemos realizado un ANOVA de un factor con 3 niveles, y hemos obtenido un p-valor de 0.001. Suponiendo que las poblaciones satisfacen las condiciones para que el ANOVA tenga sentido, ¿podemos afirmar con un nivel de significación $\alpha = 0.05$ que las medias de los tres niveles son diferentes dos a dos? Justificad la respuesta.
- Cuestión 3: Lanzamos 300 veces un dado de 6 caras de parchís, queremos contrastar que los resultados son equiprobables. ¿Cuáles serian las frecuencias esperadas o teóricas del contraste?
- Cuestión 4: En un ANOVA de una vía, queremos contrastar si los 6 niveles de un factor definen poblaciones con la misma media. Sabemos que estas seis poblaciones son normales con la misma varianza $\sigma = 2$. Estudiamos a 11 individuos de cada nivel y obtenemos que $SS_{Total} = 256.6$ y $SS_{Tr} = 60.3$. ¿Qué vale SS_E. ¿Qué valor estimamos que tiene σ^2?
- Cuestión 5: Calculad la correlación entre los vectores de datos $x = (1, 3, 4, 4)$, $y = (2, 4, 12, 6)$.
- Cuestión 6: De estas cuatro matrices, indicad cuáles pueden ser matrices de correlaciones, y explicad por qué.

$$A = \begin{pmatrix} 1 & 0.8 \\ -0.8 & 1 \end{pmatrix}, B = \begin{pmatrix} 0.8 & 0.6 \\ 0.6 & 0.8 \end{pmatrix}, C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, D = \begin{pmatrix} 1 & 1.2 \\ 1.2 & 1 \end{pmatrix}.$$

1.7.1 Solución

Cuestion 1: Rechazamos la hipótesis nula para todos los niveles de significación α menores que el p-valor.

Cuestion 2: No, no podemos afirmar eso. Lo que sabemos después de rechazar un ANOVA es que hay al menos dos medias distintas.

Cuestion 3: pues son

```
probs=c(1/6,1/6,1/6,1/6,1/6,1/6)
frec.esp=300*probs
frec.esp
```

```
## [1] 50 50 50 50 50 50
```

Cuestion 4 $SS_E=SS_{Total}-SS-{Tr}=256.6-60.3=196.3.

El número de observaciones totales es $N = 11 \cdot 6 = 66$ y el número de niveles del factor es $k = 6$.

El estimador de la varianza conjunto σ^2 es $MS_E = \frac{SS_e}{N-k} = \frac{196.3}{66-6} = 17.8454545$.

***Cuestión 5**

```
x=c(1,3,4,4)
y=c(2,4,12,6)
cor(x,y)
```

```
## [1] 0.7637626
```

Cuestión 6

Son matrices de correlaciones 2x2 sabemos que la diagonal tiene que ser 1 pues es la correlación de una variable consigo misma. También sabemos que $\rho_{xy} = \rho_{yx}$ así que la matriz debe ser simétrica. Además $-1 \leq \rho_{xy} \leq .1$.

A no es simétrica, B no tiene la diagonal de unos y D tienen valores mayores que 1 luego estas tres matrices no son de correlaciones. La única matriz que cumple todas las condiciones es la C.

1.8 Problema 8: Contraste de proporciones de dos muestras independientes.

Queremos comparar las proporciones de aciertos de dos redes neuronales que detectan si una foto con un móvil de una avispa es una avispa velutina o asiática o si es una avispa común. Esta avispa en una especie invasora y peligrosa por el veneno de su picadura. Para ello disponemos de una muestra de 1000 imágenes de insectos etiquetadas como avispa velutina y no velutina.

Aquí tenéis el acceso a los datos. Cada uno está en fichero selecciona 500 fotos de de forma independiente para el algoritmo 1 y el 2. Los aciertos están codificados con 1 y los fallos con 0.

Se pide:

1. Cargad los datos desde el servidos y calcular el tamaño de las muestras y la proporción de aciertos de cada muestra.
2. Contrastad si hay evidencia de que las las proporciones de aciertos del algoritmo 1 son mayores que las del algoritmo 2. Definid bien las hipótesis y las condiciones del contraste. Tenéis que hacer el contraste con funciones de `R` y resolver el contrate con el p-valor.
3. Calculad e interpretar los intervalos de confianza para la diferencia de proporciones asociados al test anterior, con funciones de R.

1.8.1 Solución

```
algoritmo1=read.table("http://bioinfo.uib.es/~recerca/MATIIIGINF/velutina/algoritmo1.csv")
algoritmo2=read.table("http://bioinfo.uib.es/~recerca/MATIIIGINF/velutina/algoritmo2.csv")
```

Proporción aciertos de cada algoritmo

```
n1=dim(algoritmo1)[1]
n1
```

```
## [1] 500
```

```
n1=length(algoritmo1$V1)
n1
```

```
## [1] 500
```

```
n2=length(algoritmo2$V1)
n2
```

```
## [1] 500
```

```
aciertos_absolutos_algoritmo1=table(algoritmo1)["1"]
aciertos_absolutos_algoritmo1
```

```
##   1
## 396
```

```
p1=prop.table(table(algoritmo1))["1"]
p1
```

```
##     1
## 0.792
```

```
aciertos_absolutos_algoritmo2=table(algoritmo2)["1"]
aciertos_absolutos_algoritmo2
```

```
##   1
## 437
```

```
p2=prop.table(table(algoritmo2))["1"]
p2
```

```
##     1
## 0.874
```

Después de los cálculos preliminares si denotamos las proporciones poblacionales de aciertos de cada algoritmo por p_1 y p_2 respectivamente, el contraste que nos piden es

$$\begin{cases} H_0: & p_1 = p_2 \\ H_1: & p_1 > p_2 \end{cases}$$

estamos ante un diseño de comparación de proporciones con muestras independientes. Con R lo podemos resolver con el `fisher.test` o con el `prop.test`

```
x=matrix(c(aciertos_absolutos_algoritmo1,
           n1-aciertos_absolutos_algoritmo1,
           aciertos_absolutos_algoritmo2,
           n2-aciertos_absolutos_algoritmo2),
         ncol=2,byrow=FALSE)
x
```

```
##      [,1] [,2]
## [1,]  396  437
## [2,]  104   63
```

```
fisher.test(x,alternative="greater",conf.level=0.95)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  x
## p-value = 0.9998
## alternative hypothesis: true odds ratio is greater than 1
## 95 percent confidence interval:
```

```
##  0.4056457       Inf
## sample estimates:
## odds ratio
##  0.5492712
```

```
c(aciertos_absolutos_algoritmo1,
  aciertos_absolutos_algoritmo2)
```

```
##   1   1
## 396 437
```

```
c(n1,n2)
```

```
## [1] 500 500
```

```
prop.test(c(aciertos_absolutos_algoritmo1,
            aciertos_absolutos_algoritmo2), c(n1,n2),
          alternative="greater",conf.level=0.95)
```

```
##
##  2-sample test for equality of proportions with continuity correction
##
## data:  c(aciertos_absolutos_algoritmo1, aciertos_absolutos_algoritmo2) out of c(n1, n2)
## X-squared = 11.502, df = 1, p-value = 0.9997
## alternative hypothesis: greater
## 95 percent confidence interval:
##  -0.1225654  1.0000000
## sample estimates:
## prop 1 prop 2
##  0.792  0.874
```

Con ambos test obtenemos p valores altos (el más pequeño es el de Fisher y es mayor que 0.4, así que no podemos rechazar que las proporciones de aciertos de los dos algoritmos sean iguales contra que la proporción de aciertos del algoritmo 1 es mejor que la del 2.

El intervalo de confianza asociado a este test es

```
prop.test(c(aciertos_absolutos_algoritmo1,
            aciertos_absolutos_algoritmo2),
          c(n1,n2),
          alternative="greater",
          conf.level=0.95)$conf.int
```

```
## [1] -0.1225654  1.0000000
## attr(,"conf.level")
## [1] 0.95
```

luego con una probabilidad del 95% la $p_1 - p_2 > -1$ contiene el 0 y no podemos despreciar que sean iguales contra que $p_1 > p_2$.

1.9 Problema 9 : Contraste de proporciones de dos muestras emparejadas.

En el problema anterior hemos decidido quedarnos con el mejor de los algoritmos y mejorarlo. Pasamos las mismas 1000 imágenes a la version_beta del algoritmo y a la version_alpha. Aquí tenéis el acceso a los datos en el mismo orden para las 1000 imágenes. Cada uno está en fichero los aciertos están codificados con 1 y los fallos con 0.

1. Cargad los datos desde el servidos y calcular el tamaño de las muestras y la proporción de aciertos de cada muestra.

2. Contrastad si hay evidencia de que las las proporciones de aciertos del algoritmo alfa son iguales que las del algoritmo beta. Definid bien las hipótesis y las condiciones del contraste. Tenéis que hacer el contraste con funciones de `R` y resolver el contrate con el p-valor.

1.9.1 Solución

Cargamos los datos y hacemos los cálculos preliminares

```
algoritmoalfa=read.table("http://bioinfo.uib.es/~recerca/MATIIIGINF/velutina2/algoritmo_alpha.csv")
algoritmobeta=read.table("http://bioinfo.uib.es/~recerca/MATIIIGINF/velutina2/algoritmo_beta.csv")
```

El test que nos piden es

$$\begin{cases} H_0: & p_\alpha = p_\beta \\ H_1: & p_\alpha \neq p_\beta \end{cases}$$

Es un diseño de muestras emparejadas y tenemos que utilizar el `mcnear.test`:

```
X=table(algoritmoalfa$V1,algoritmobeta$V1)
X
```

```
##
##       0   1
##   0  15 110
##   1  88 787
```

```
mcnemar.test(X)
```

```
##
##  McNemar's Chi-squared test with continuity correction
##
## data:  X
## McNemar's chi-squared = 2.2273, df = 1, p-value = 0.1356
```

El p-valor es 0.1356 no podemos rechazar la igualdad de la proporción de aciertos.

1.10 Problema 10 : ANOVA comparación media puntuaciones según fabricante.

Una vez mejorado nuestro algoritmo queremos saber su comportamiento bajo distintos tipos de móviles.

Seleccionamos 6 móviles de la misma gama de calidad de 6 fabricantes distintos. A los fabricantes los denotamos por F1, F2, F3, F4, F5 y F6.

Vamos a jugar no con la clasificación sino con el score que produce el algoritmo. Para ello seleccionamos 4 muestra aleatorias de fotos de insectos enviadas por los usuarios y la puntuación (*score*) que nos da el algoritmo que es una variable aleatoria continua de con rango de 0 a 100.

La idea es comprobar si la media de las puntuaciones del algoritmo es la misma para cada uno de los fabricantes.

Los datos los podéis descargar de esta dirección del servidor bioinfo.uib.es.

Antes de descargarlo, visualizar el fichero desde el navegador, para saber cómo descargarlo.

1. ¿Podemos asegurar que la muestras son normales en cada grupo? ¿y son homocedásticas? Justificar la respuesta con el correspondiente código en R comentado.
2. Escribid formalmente la hipótesis nula y la alternativa. Calcular la tabla de ANOVA y resuelve el test de forma manual.
3. Calcular la tabla de ANOVA y resuelve el test con la función `aov` de R.

4. Haced una comparación de pares con la función adecuada de R para la corrección del holm al nivel de significación $\alpha = 0.1$. Interpreta el resultado.
5. Comparar por grupos con el test de Duncan del paquete `agricolae`. Interpreta el resultado.

1.10.1 Solución

```
df=read.table("http://bioinfo.uib.es/~recerca/MATIIIGINF/anova_score/score_manufacturer.csv")
head(df)
```

```
##      score manufacturer
## 1 69.32030           F1
## 2 66.93433           F1
## 3 67.70541           F1
## 4 63.47195           F1
## 5 65.58738           F1
## 6 65.47437           F1
```

```
df$manufacturer=as.factor(df$manufacturer)
```

```
table(df$manufacturer)
```

```
##
##  F1  F2  F3  F4  F5  F6
## 100 100 100 100 100 100
```

Tenemos que comprobar la normalidad de la distribución de la muestra para cada nivel del factor $i = 1, 2, 3, 4, 5, 6$; el test es

$$\begin{cases} H_0: & \text{la distribución de los datos en el nivel } F_i \text{ es normal,} \\ H_1: & \text{la distribución de los datos en el nivel } F_i \text{ no es normal,} \end{cases}$$

```
library(nortest)
# El test KS_Lillie para en nivel "F1"
#lillie.test(df$score[df$manufacturer=="F1"])
sapply(levels(df$manufacturer), FUN=function(x) {print(lillie.test(df$score[df$manufacturer==x]))})
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  df$score[df$manufacturer == x]
## D = 0.091505, p-value = 0.03825
##
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  df$score[df$manufacturer == x]
## D = 0.067758, p-value = 0.3121
##
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  df$score[df$manufacturer == x]
## D = 0.069567, p-value = 0.2744
##
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
```

```
##
## data:  df$score[df$manufacturer == x]
## D = 0.069567, p-value = 0.2744
##
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  df$score[df$manufacturer == x]
## D = 0.069567, p-value = 0.2744
##
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  df$score[df$manufacturer == x]
## D = 0.10632, p-value = 0.007255
```

```
##           F1
## statistic 0.09150477
## p.value   0.03825059
## method    "Lilliefors (Kolmogorov-Smirnov) normality test"
## data.name "df$score[df$manufacturer == x]"
##           F2
## statistic 0.06775771
## p.value   0.3121052
## method    "Lilliefors (Kolmogorov-Smirnov) normality test"
## data.name "df$score[df$manufacturer == x]"
##           F3
## statistic 0.06956719
## p.value   0.2743622
## method    "Lilliefors (Kolmogorov-Smirnov) normality test"
## data.name "df$score[df$manufacturer == x]"
##           F4
## statistic 0.06956719
## p.value   0.2743622
## method    "Lilliefors (Kolmogorov-Smirnov) normality test"
## data.name "df$score[df$manufacturer == x]"
##           F5
## statistic 0.06956719
## p.value   0.2743622
## method    "Lilliefors (Kolmogorov-Smirnov) normality test"
## data.name "df$score[df$manufacturer == x]"
##           F6
## statistic 0.1063201
## p.value   0.007255259
## method    "Lilliefors (Kolmogorov-Smirnov) normality test"
## data.name "df$score[df$manufacturer == x]"
```

```
# También podemos hacer un bucle clásico
#for(Fabricante in levels(df$manufacturer)){
#print(lillie.test(df$score[df$manufacturer==Fabricante]))
#}
```

El nivel “F1” y “F6” dan valores pequeños no podemos asegurar la normalidad en estos casos.

Nos aseguramos con el ómnibus test de D’Agostino

```r
library(fBasics)
```

```
## Loading required package: timeDate
```

```
## Loading required package: timeSeries
```

```
##
## Attaching package: 'fBasics'
```

```
## The following object is masked from 'package:car':
##
##     densityPlot
```

```r
dagoTest(df$score[df$manufacturer=="F1"])
```

```
##
## Title:
##  D'Agostino Normality Test
##
## Test Results:
##   STATISTIC:
##     Chi2 | Omnibus: 5.8827
##     Z3   | Skewness: 2.1113
##     Z4   | Kurtosis: 1.1939
##   P VALUE:
##     Omnibus  Test: 0.05279
##     Skewness Test: 0.03475
##     Kurtosis Test: 0.2325
##
## Description:
##  Wed May 26 10:23:09 2021 by user: t169
```

```r
dagoTest(df$score[df$manufacturer=="F6"])
```

```
##
## Title:
##  D'Agostino Normality Test
##
## Test Results:
##   STATISTIC:
##     Chi2 | Omnibus: 1.7625
##     Z3   | Skewness: -1.2831
##     Z4   | Kurtosis: 0.3408
##   P VALUE:
##     Omnibus  Test: 0.4143
##     Skewness Test: 0.1995
##     Kurtosis Test: 0.7333
##
## Description:
##  Wed May 26 10:23:09 2021 by user: t169
```

Parece que no podemos rechazar la normalidad para los casos dudosos.

Ahora realizamos el test de comparación de medias μ_i para $i = 1, 2, 3, 4, 5, 6$ son las medias para cada nivel del factor.

$$\begin{cases} H_0: & \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 = \mu_6 \\ H_1: & \text{no todas la medias son iguales,} \end{cases}$$

```
summary(aov(df$score~df$manufacturer))
```

```
##                   Df Sum Sq Mean Sq F value               Pr(>F)
## df$manufacturer    5   9143  1828.5   17.54 0.000000000000000317 ***
## Residuals        594  61910   104.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como el p-valor es muy pequeño NO podemos aceptar que las 6 medias sean iguales

Ahora tenemos que contrastar que pares de medias dos a dos son iguales y ajustar los p-valores con el ajuste del p-valor por el método de Holm

```
pairwise.t.test(df$score,df$manufacturer,p.adjust.method = "holm")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$score and df$manufacturer
##
##    F1      F2            F3            F4      F5
## F2 1.00000 -             -             -       -
## F3 0.07729 0.44250       -             -       -
## F4 0.00011 0.00000297115 0.00000000017 -       -
## F5 0.00011 0.00000297115 0.00000000017 1.00000 -
## F6 0.00724 0.00040       0.00000014687 1.00000 1.00000
##
## P value adjustment method: holm
```

Comparamos los p-valores con 0.05 y aceptamos que las medias de los niveles F_4, F_6 y F_5 son iguales dos a dos, también son iguales la media del F_1 con el F_2, y la media del nivel F_2 con el F_3. EL resto de comparaciones tienen p-valores bajos así que no podemos aceptar la igualdad de medias.

Ahora comparamos las medias por grupos de igualdades con el test de Duncan

```
library(agricolae)
```

```
##
## Attaching package: 'agricolae'
```

```
## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness
```

```
resultado.anova=aov(df$score~df$manufacturer)
duncan.test(resultado.anova,"df$manufacturer",group=TRUE)$group
```

```
##    df$score groups
## F3 74.07499      a
## F2 71.61166     ab
## F1 70.47337      b
## F6 65.71299      c
## F4 64.07499      c
## F5 64.07499      c
```

Obtenemos tres grupos el **a** dice la media de $\mu_3 = mu_2$ el **b** dice que $\mu_2 = \mu_1$ y el **c** dice que $\mu_6 = \mu_4 = \mu_5$. Obtenemos conclusiones similares al test de comparación de medias.

1.11 Problema 11: Regresión lineal simple.

Consideremos los siguientes datos

```
x=c(-2,-1,2,0,1,2)
y=c(-7, -5,  5, -3,  3.0,  4)
summary(lm(y~x))
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      1      2      3      4      5      6
##  0.675 -0.400  0.375 -1.475  1.450 -0.625
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5250     0.4872  -3.130 0.035176 *
## x             3.0750     0.3189   9.642 0.000647 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.165 on 4 degrees of freedom
## Multiple R-squared:  0.9587, Adjusted R-squared:  0.9484
## F-statistic: 92.96 on 1 and 4 DF,  p-value: 0.0006472
```

1. Calcular manualmente los coeficiente de la regresión lineal de y sobre x
2. Calcular los valores $\hat{y}_i = b_0 + b_1 \cdot x_1$ para los valores de la muestra y el error cometido.
3. Calcular la estimación de la varianza del error.
4. Resolver manualmente el contraste $\begin{cases} H_0: & \beta_1 = 0 \\ H_1: & \beta_1 \neq 0 \end{cases}$, calculando el p-valor.
5. Calcular SST, SSR y SSE.
6. Calcular el coeficiente de regresión lineal r_{xy} y el coeficiente de determinación R^2. Interpretad el resultado en términos de la cantidad de varianza explicada por el modelo
7. Comprobar que los resultados son los mismos que los obtenidos con la función `summary(lm(y~x))`.

1.11.1 Solución

Faltan añadir los NECESARIOS COMENTARIOS.

```
x=c(-2,-1,2,0,1,2)
y=c(-7, -5,  5, -3,  3.0,  4)
sol_lm=lm(y~x)
summary(sol_lm)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      1      2      3      4      5      6
##  0.675 -0.400  0.375 -1.475  1.450 -0.625
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.5250     0.4872  -3.130 0.035176 *
## x             3.0750     0.3189   9.642 0.000647 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.165 on 4 degrees of freedom
## Multiple R-squared:  0.9587, Adjusted R-squared:  0.9484
## F-statistic: 92.96 on 1 and 4 DF,  p-value: 0.0006472
```

```
mediay=mean(y)
mediax=mean(x)
sdx=sd(x)
sdy=sd(y)
sxy=cov(x,y)
b1=sxy/sdx^2
b1
```

```
## [1] 3.075
```

```
b0=mediay-b1*mediax
b0
```

```
## [1] -1.525
```

```
sol_lm$coefficients
```

```
## (Intercept)           x
##      -1.525       3.075
```

```
c(b0,b1)==sol_lm$coefficients# dan distintos errores de redondeo
```

```
## (Intercept)           x
##       FALSE        TRUE
```

```
near(c(b0,b1),sol_lm$coefficients)# opcional
```

```
## (Intercept)           x
##        TRUE        TRUE
```

```
sol_lm$fitted.values
```

```
##      1      2      3      4      5      6
## -7.675 -4.600  4.625 -1.525  1.550  4.625
```

```
recta=function(x) b0+b1*x
y_est=recta(x)
y_est
```

```
## [1] -7.675 -4.600  4.625 -1.525  1.550  4.625
```

```
predict(sol_lm,newdata = data.frame(x=x))
```

```
##      1      2      3      4      5      6
## -7.675 -4.600  4.625 -1.525  1.550  4.625
```

```
y
```

```
## [1] -7 -5  5 -3  3  4
```

```r
y_est
```

```
## [1] -7.675 -4.600  4.625 -1.525  1.550  4.625
```

```r
e=y-y_est
e
```

```
## [1]  0.675 -0.400  0.375 -1.475  1.450 -0.625
```

```r
sol_lm$residuals
```

```
##      1      2      3      4      5      6
##  0.675 -0.400  0.375 -1.475  1.450 -0.625
```

```r
mean(e) # es cero, pero por error de redondeo no da exacto.
```

```
## [1] -0.00000000000000002220446
```

```r
SSE=sum(e^2)
SSE
```

```
## [1] 5.425
```

```r
n=length(x)
n
```

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

```r
S2=SSE/(n-2)#estimacion_var_error
S2
```

```
## [1] 1.35625
```

```r
S=sqrt(S2)# Residual estándar error: 1.165
S
```

```
## [1] 1.164581
```

```r
round(S,3) # con los mismos decimales da lo mismo
```

```
## [1] 1.165
```

```r
 # contraste beta1=0
t0=b1/(S/(sdx*sqrt(n-1)))
t0
```

```
## [1] 9.6415
```

```r
2*pt(abs(t0),n-2,lower.tail = FALSE)
```

```
## [1] 0.0006472191
```

```r
2*(1-pt(abs(t0),n-2,lower.tail = TRUE))
```

```
## [1] 0.0006472191
```

```r
2*(1-pt(abs(t0),n-2))
```

```
## [1] 0.0006472191
```

comparar con

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.5250     0.4872  -3.130 0.035176 *
```

```
x             3.0750     0.3189   9.642 0.000647 ***
```

```
knitr::include_graphics("formulas_regre.PNG")
```

- **Variabilidad total** o suma total de cuadrados: $SS_T = \sum_{i=1}^{n}(y_i - \overline{y})^2 = (n-1)\cdot \tilde{s}_y^2$.
- **Variabilidad de la regresión** o suma de cuadrados de la regresión: $SS_R = \sum_{i=1}^{n}(\hat{y}_i - \overline{y})^2 = (n-1)\cdot \tilde{s}_{\hat{y}}^2$.
- **Variabilidad del error** o suma de cuadrados del error: $SS_E = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = (n-1)\cdot \tilde{s}_e^2$.

```
SST=sum((y-mean(y))^2)
SST
```

```
## [1] 131.5
```

```
mean(y_est)
```

```
## [1] -0.5
```

```
mean(y)#media estimados regresión igual a media variable y
```

```
## [1] -0.5
```

```
SSR=sum((y_est-mean(y))^2)
SSR
```

```
## [1] 126.075
```

```
SSE# ya lo había a calculado
```

```
## [1] 5.425
```

```
SST-SSR# da lo mismo pues SST=SSR+SSE
```

```
## [1] 5.425
```

```
R2=SSR/SST
R2
```

```
## [1] 0.9587452
```

```
cor(x,y)
```

```
## [1] 0.9791554
```

```
cor(x,y)^2# en el caso regresión simple  R2=cor(xy)^2
```

```
## [1] 0.9587452
```

1.12 Problema 12: Distribución de los grados de un grafo de contactos.

En el artículo de A. Broder et al., Graph structure in the Web. Computer Networks 33, 309 (2000).

Se recopiló el número de enlaces a sitios web encontrados en un rastreo web de 1997 de aproximadamente 200 millones de páginas web,

Con el se construyó una tabla con la frecuencia de sitios por número de enlaces. El código siguiente carga del enlace que han puesto los autores del artículo

```
data_links=read.table("http://tuvalu.santafe.edu/~aaronc/powerlaws/data/weblinks.hist",header=TRUE)
head(data_links)
```

```
##   degree frequency
## 1      0  35159835
## 2      1 106649769
## 3      2  40711748
## 4      3  22648832
## 5      4  12617832
## 6      5   8188854
```

```
str(data_links)
```

```
## 'data.frame':    14480 obs. of  2 variables:
##  $ degree   : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ frequency: int  35159835 106649769 40711748 22648832 12617832 8188854 6438634 4690068 4954649 373
```

```
# eliminamos la páginas con menos de 8 enlaces  enlaces y las de más de 1000 enlaces
data_links_central=data_links[data_links$degree>8&data_links$degree<10^3,]
head(data_links_central)
```

```
##    degree frequency
## 10      9   3731928
## 11     10   3036333
## 12     11   2496648
## 13     12   2119312
## 14     13   1790068
## 15     14   1546579
```

```
tail(data_links_central)
```

```
##      degree frequency
## 995     994       213
## 996     995       193
## 997     996       157
## 998     997       137
## 999     998       178
## 1000    999       153
```

El siguiente código calcula las regresiones exponencial, potencial y lineal (en algún orden) de las frecuencias (`frequency`) contra los enlaces (`degree`).

```
sol1=lm(frequency~ degree,data=data_links_central)
summary(sol1)
```

```
##
## Call:
## lm(formula = frequency ~ degree, data = data_links_central)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -96861  -69548  -25033   22374 3598744
##
## Coefficients:
##              Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 134974.49   13778.17   9.796 <0.0000000000000002 ***
## degree        -198.98      23.77  -8.369 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 214100 on 989 degrees of freedom
```

```
## Multiple R-squared:  0.06614,    Adjusted R-squared:  0.06519
## F-statistic: 70.04 on 1 and 989 DF,  p-value: < 0.00000000000000022
```

```
sol2=lm(log10(frequency)~ degree,data=data_links_central)
summary(sol2)
```

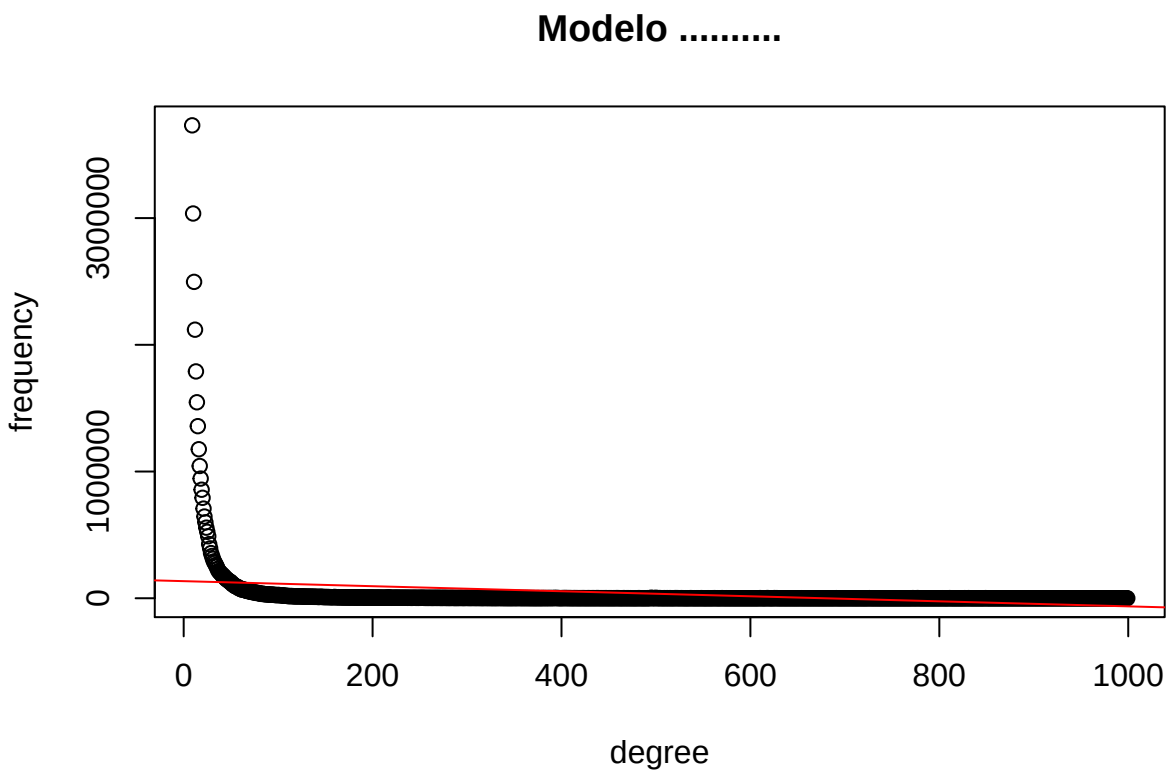
```
##
## Call:
## lm(formula = log10(frequency) ~ degree, data = data_links_central)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43758 -0.26558 -0.07671  0.16681  2.13097
##
## Coefficients:
##               Estimate  Std. Error t value            Pr(>|t|)
## (Intercept)  4.46504979  0.02381018  187.53 <0.0000000000000002 ***
## degree      -0.00267658  0.00004109  -65.15 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.37 on 989 degrees of freedom
## Multiple R-squared:  0.811,  Adjusted R-squared:  0.8108
## F-statistic:  4244 on 1 and 989 DF,  p-value: < 0.00000000000000022
```

```
sol3=lm(log10(frequency)~ log10(degree),data=data_links_central)
summary(sol3)
```

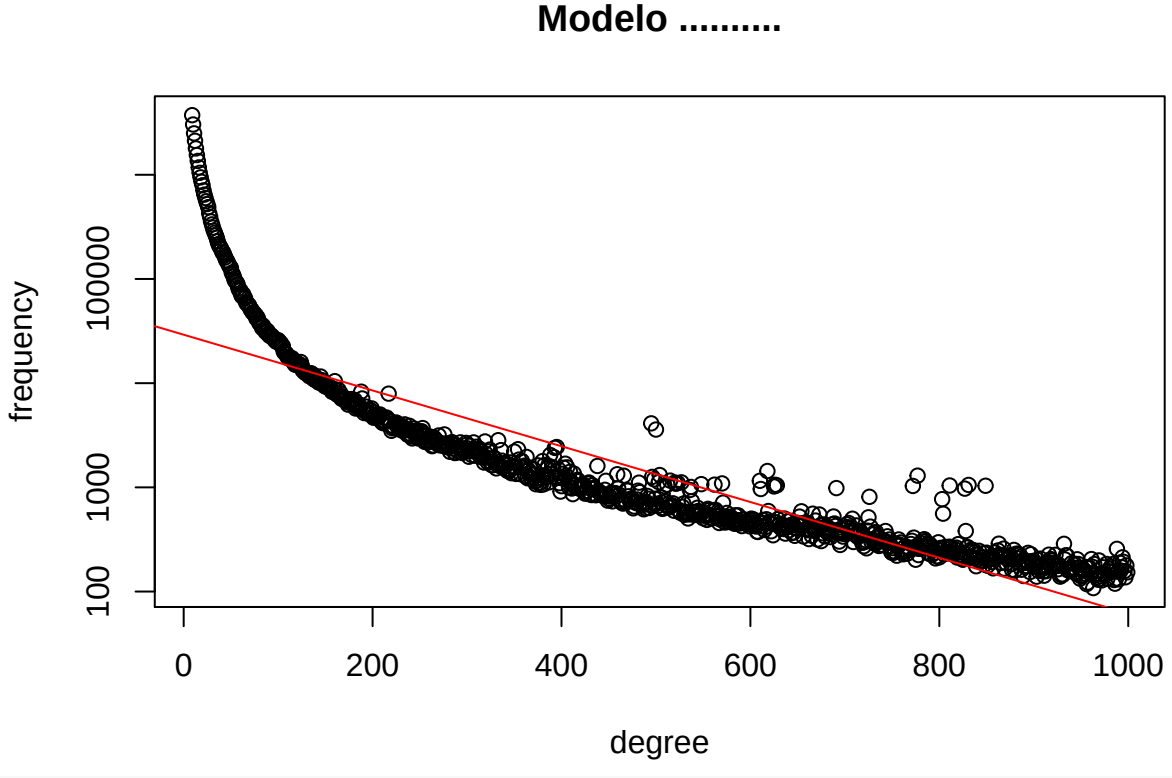
```
##
## Call:
## lm(formula = log10(frequency) ~ log10(degree), data = data_links_central)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.21376 -0.04747 -0.01555  0.01958  0.73976
##
## Coefficients:
##               Estimate Std. Error t value            Pr(>|t|)
## (Intercept)    8.722036   0.020623   422.9 <0.0000000000000002 ***
## log10(degree) -2.170129   0.007894  -274.9 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09674 on 989 degrees of freedom
## Multiple R-squared:  0.9871, Adjusted R-squared:  0.9871
## F-statistic: 7.557e+04 on 1 and 989 DF,  p-value: < 0.00000000000000022
```

Ahora dibujamos los gráficos adecuados a cada modelo

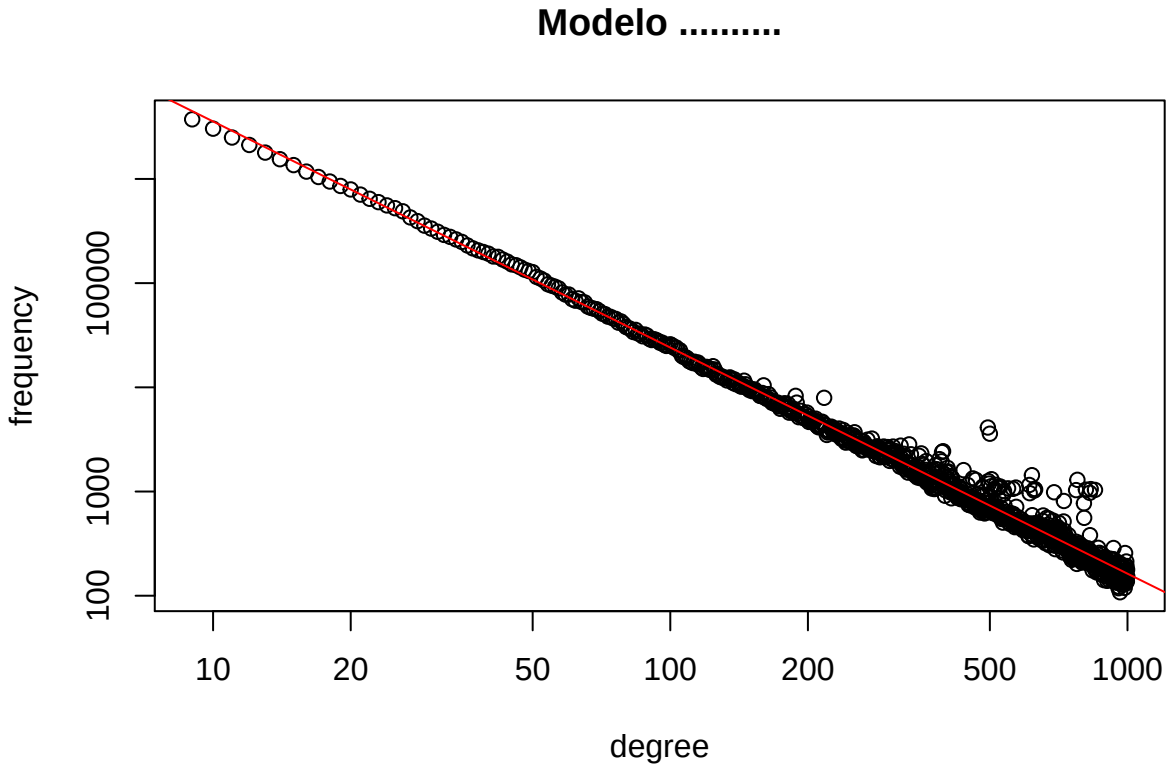
```
plot(data_links_central,main="Modelo ..........")
abline(sol1,col="red")
```




```
plot(data_links_central,main="Modelo .....",log="y")
abline(sol2,col="red")
```




```
plot(data_links_central,main="Modelo .....",log="xy")
abline(sol3,col="red")
```



Se pide:

1. Explicad el modelo de regresión que calcula cada función `lm`
2. ¿Qué modelo y en función de qué parámetros es el mejor?
3. Para el mejor modelo calcular los coeficientes en las unidades originales y escribir la ecuación del modelos.

1.12.1 Solución

Apartado 1

Los modelos son

```
sol1=lm(frequency~ degree,data=data_links_central)
summary(sol1)
```

```
##
## Call:
## lm(formula = frequency ~ degree, data = data_links_central)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -96861  -69548  -25033   22374 3598744
##
## Coefficients:
##              Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 134974.49   13778.17   9.796 <0.0000000000000002 ***
## degree        -198.98      23.77  -8.369 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 214100 on 989 degrees of freedom
```

```
## Multiple R-squared:  0.06614,    Adjusted R-squared:  0.06519
## F-statistic: 70.04 on 1 and 989 DF,  p-value: < 0.00000000000000022
```

```
sol2=lm(log10(frequency)~ degree,data=data_links_central)
summary(sol2)
```

```
##
## Call:
## lm(formula = log10(frequency) ~ degree, data = data_links_central)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43758 -0.26558 -0.07671  0.16681  2.13097
##
## Coefficients:
##               Estimate  Std. Error t value            Pr(>|t|)
## (Intercept)  4.46504979  0.02381018  187.53 <0.0000000000000002 ***
## degree      -0.00267658  0.00004109  -65.15 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.37 on 989 degrees of freedom
## Multiple R-squared:  0.811,  Adjusted R-squared:  0.8108
## F-statistic:  4244 on 1 and 989 DF,  p-value: < 0.00000000000000022
```

```
sol3=lm(log10(frequency)~ log10(degree),data=data_links_central)
summary(sol3)
```

```
##
## Call:
## lm(formula = log10(frequency) ~ log10(degree), data = data_links_central)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.21376 -0.04747 -0.01555  0.01958  0.73976
##
## Coefficients:
##               Estimate Std. Error t value            Pr(>|t|)
## (Intercept)    8.722036   0.020623   422.9 <0.0000000000000002 ***
## log10(degree) -2.170129   0.007894  -274.9 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09674 on 989 degrees of freedom
## Multiple R-squared:  0.9871, Adjusted R-squared:  0.9871
## F-statistic: 7.557e+04 on 1 and 989 DF,  p-value: < 0.00000000000000022
```

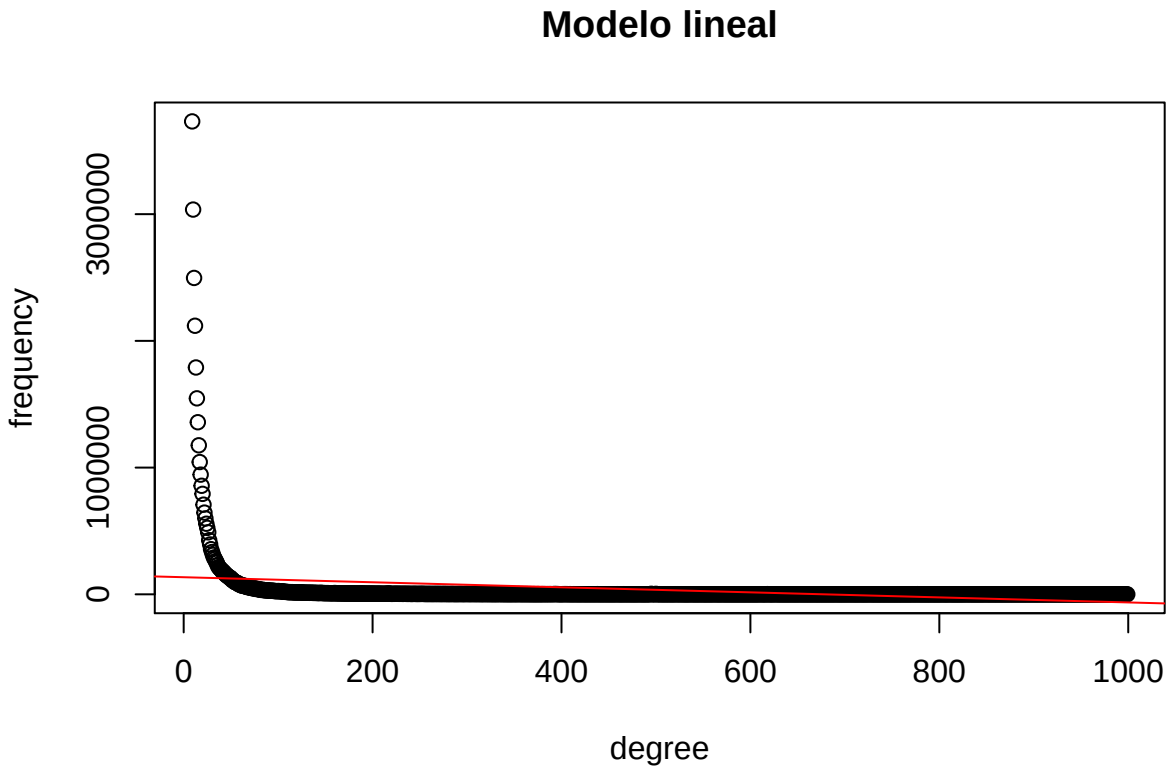
Así que

- el primero es $frequency = b_0 + b_1 \cdot degree$ que es el modelo LINEAL que resuelve la función `lm`.
- el segundo es $\log_{10}(frequency) = b_0 + b_1 \cdot degree$ que operando adecuadamente es

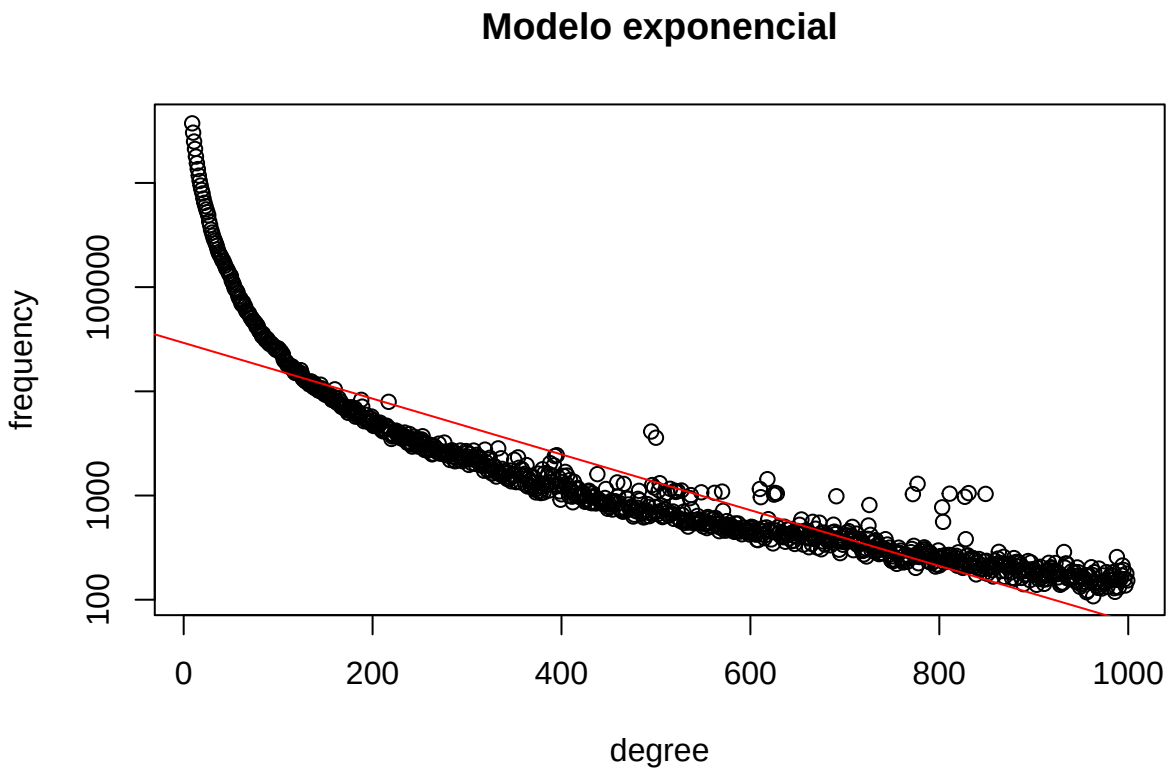
$10^{\log_{10}(frequency)} = 10^{b_0} \cdot 10^{b_1 \cdot degree}$ operando obtenemos que el modelo final es un modelo EXPONENCIAL $frequency = 10^{b_0} \cdot \left(10^{b_1}\right)^{degree}$. * el tercer modelo es $\log_{10}(frequency) = b_0 + b_1 \cdot \log_{10}(degree)$ que despejando es $10^{\log_{10}(frequency)} = 10^{b_0} \cdot \left(10^{(\log_{10}(degree))}\right)^{b_1}$ operando obtenemos que el modelo final es un modelo POTENCIAL $frequency = 10^{b_0} \cdot degree^{b_1}$.

Los dibujos con el título adecuado son

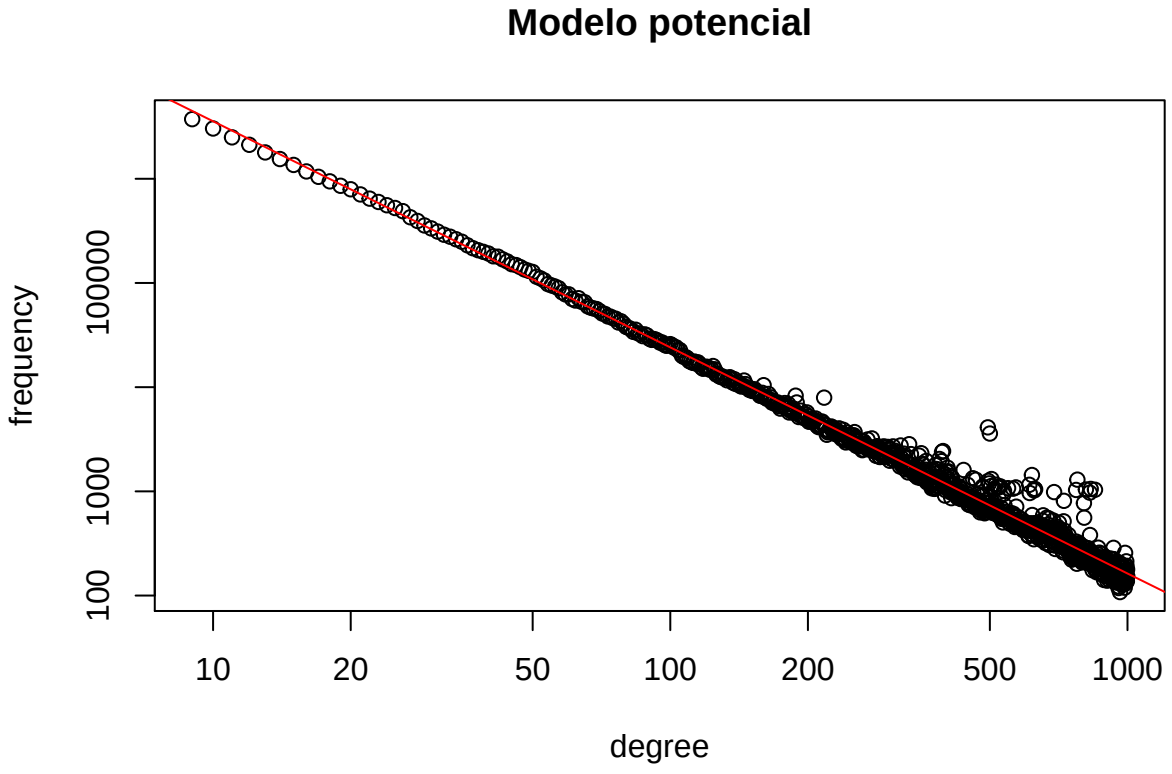
```
plot(data_links_central,main="Modelo lineal")
abline(sol1,col="red")
```




```
plot(data_links_central,main="Modelo exponencial",log="y")
abline(sol2,col="red")
```




```
plot(data_links_central,main="Modelo potencial",log="xy")
abline(sol3,col="red")
```



Con lo que hemos visto el modelo con mejor R^2 es el mejor

```r
summary(sol1)$r.squared
```

```
## [1] 0.06613879
```

```r
summary(sol2)$r.squared
```

```
## [1] 0.8110118
```

```r
summary(sol3)$r.squared
```

```
## [1] 0.9870815
```

Así que el mejor modelo es el tercero el potencial pues su R^2 es muy alto

Apartado 3

Calcularemos la ecuación del modelo para las unidades originales sin transformaciones logarítmicas.

Recordemos el resultado de la `sol3`

```r
summary(sol3)
```

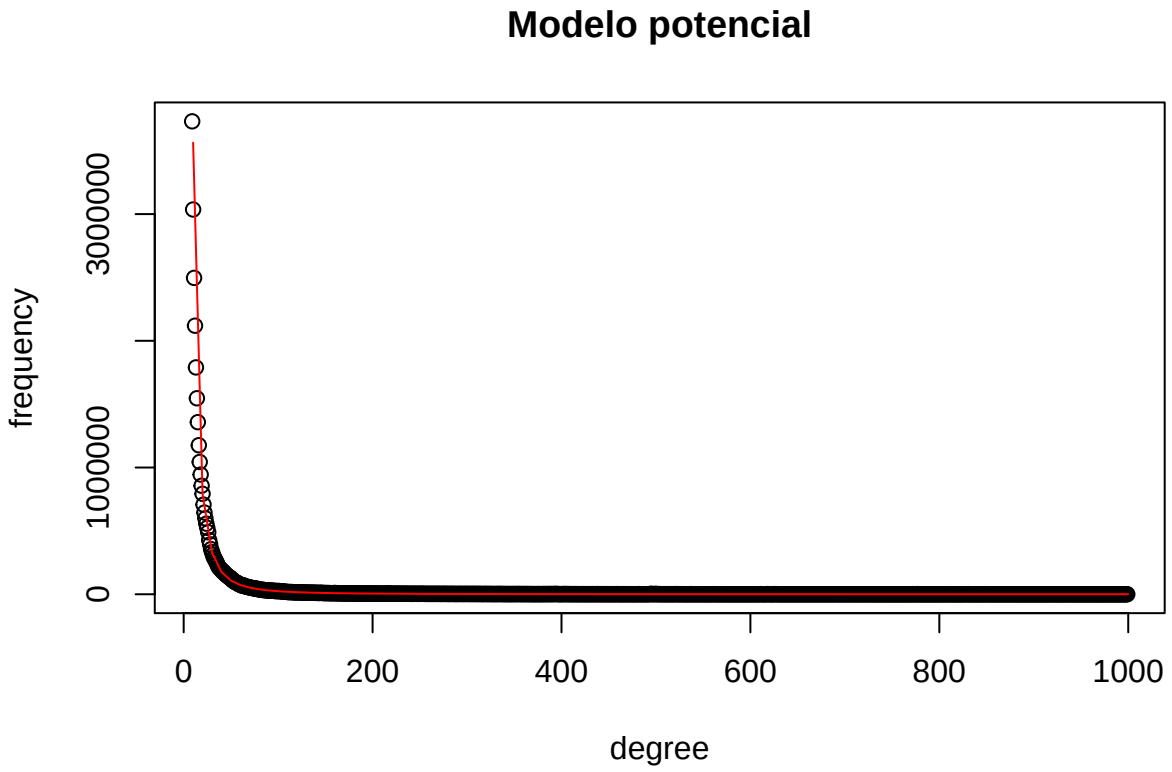
```
##
## Call:
## lm(formula = log10(frequency) ~ log10(degree), data = data_links_central)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.21376 -0.04747 -0.01555  0.01958  0.73976
##
## Coefficients:
##                Estimate Std. Error t value            Pr(>|t|)
## (Intercept)    8.722036   0.020623   422.9 <0.0000000000000002 ***
## log10(degree) -2.170129   0.007894  -274.9 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09674 on 989 degrees of freedom
## Multiple R-squared:  0.9871, Adjusted R-squared:  0.9871
## F-statistic: 7.557e+04 on 1 and 989 DF,  p-value: < 0.00000000000000022
```

Los estimadores son $b_0 = 8.722036$ y $b_1 = -2.170129$ el modelo es $frequency = 10^{b_0} \cdot degree^{b_1}$. Sustituyendo los valores el modelo obtenemos $frequency = 10^{8.722036} \cdot degree^{-2.170129}$, finalmente operando

$$frequency = 527273566.93254 \cdot degree^{-2.170129}.$$

No se pedía en el ejercicio pero hacemos el dibujo de los datos y la ecuación en las unidades originales

```r
frequency=data_links_central$frequency
degree=data_links_central$degree
potencial=function(x) 10^(8.722036)*x^(-2.170129)
plot(degree,frequency,main="Modelo potencial",xlab="degree",ylab="frequency")
curve(potencial,xlim=c(0,1000),ylim=c(0,3731928),col="red",add=TRUE)
```


Modelo potencial
frequency
0
1000000
3000000
0
200
400
600
800
1000
degree