

Aplicaciones del análisis multivariante con R

Estadística Multivariante - Universidad de Granada

Laura Gómez Garrido

Miguel Lentisco Ballesteros

Antonio Martín Ruiz

Daniel Pozo Escalona

Francisco Javier Sáez Maldonado

22 de enero de 2020

Contenido

- 1 Introducción
- 2 Operaciones básicas
- 3 Análisis multivariante
- 4 Implementaciones
 - Distribución normal multivariante
 - Inferencia sobre DNM
- 5 Bibliografía

- 1 Introducción
- 2 Operaciones básicas
- 3 Análisis multivariante
- 4 Implementaciones
 - Distribución normal multivariante
 - Inferencia sobre DNM
- 5 Bibliografía

¿Qué es R?

R es un entorno y lenguaje de programación enfocados a la computación estadística y de gráficos. Surge como una reimplementación libre del lenguaje y entorno S. Proporciona una amplia variedad de funcionalidades estadísticas y gráficas y es altamente extensible.



¿Qué es R?

R está disponible como software libre bajo los términos de la GNU General Public License de la Free Software Foundation en forma de código fuente. Puede ser compilado y ejecutado en una gran cantidad de plataformas UNIX, Windows y MacOS.



- 1 Introducción
- 2 Operaciones básicas
- 3 Análisis multivariante
- 4 Implementaciones
 - Distribución normal multivariante
 - Inferencia sobre DNM
- 5 Bibliografía

Hola mundo

```
# Esto es un comentario  
miString <- "Hola_mundo"  
print(miString)
```

```
[1] "Hola_mundo"
```

Tipos

```
v <- TRUE
print(class(v))
v <- 42
print(class(v))
v <- 2L
print(class(v))
v <- 4+2i
print(class(v))
v <- "multivariante"
print(class(v))
v <- charToRaw("multivariante")
print(class(v))
```


Tipos

```
[1] "logical"
```

```
[1] "numeric"
```

```
[1] "integer"
```

```
[1] "complex"
```

```
[1] "character"
```

```
[1] "raw"
```

Estructuras de datos

```
vector <- c('v1', 'v2', 'v3')
lista <- list(c(1,2,3), 2+3i, "multivariante")
matriz <- matrix(c(1,2,3,4,5,6), nrow = 2, ncol =
  3)
array <- array(c('hola', 'adios'), dim=c(3,2,3))
dataframe <- data.frame(
  nombre = c("A", "B", "C"),
  peso = c(1,2,3),
  cantidad = c(4,5,6)
)
```

Aritméticos:

- + Suma
- - Resta
- * Producto
- / División
- %% Módulo
- %/ % División entera
- ^ Potencia

Relacionales:

- $<$ Menor que
- $>$ Mayor que
- $==$ Igualdad
- $<=$ Menor o igual
- $>=$ Mayor o igual

Lógicos:

- & AND
- | OR
- ! NOT
- && AND entre los primeros elementos de los vectores
- || OR entre los primeros elementos de los vectores

Misceláneos:

- `- >, < -, =` Asignación
- `:` Crea un vector con todos los valores entre los dados
- `%in %` Identifica si un elemento está en un vector
- `% %* % %` Multiplica una matriz con su traspuesta

Estructuras de decisión

```
if (3 > 2) {  
    print("3>2")  
}  
  
if (2 > 3){  
    print("2>3")  
}  
  
i = 4  
if(i > 5){  
    print("i>5")  
} else {  
    print("i<5")  
}  
  
x <- switch(2, "a", "b", "c")  
print(x)
```

Bucles

```
i = 0
repeat{
    i = i+1
    if(i > 3){
        break
    }
}
print(i)
```

```
i = 0
while(i < 5){
    i = i+1
}
print(i)
```


Bucles

```
v <- 1:4
for (i in v){
  v[i] = i+1
}
print(v)
```

Funciones

```
media <- function(a,b){  
  m <- (a+b)/2  
  return(m)  
}
```

```
print(media(1,2))
```

Cómo instalar dependencias

R incluye un gestor de paquetes integrado en el lenguaje, que permite instalar dependencias con la orden `install.packages()`.

```
install.packages("MVA")  
install.packages("HSAUR2")  
install.packages("car")  
install.packages("MASS")
```

Cómo instalar dependencias

Para importar los paquetes para usarlos en un programa:

```
library("MVA")  
library("HSAUR2")  
library("car")  
library("MASS")
```

R forma parte de un proyecto colaborativo y abierto donde sus propios usuarios pueden publicar paquetes.

Utiliza la forja de desarrollo R-Forge.

En Diciembre de 2019, el repositorio oficial tenía disponibles 15.315 paquetes organizados en vistas o temas.





- Bayesian
- ChemPhys
- ClinicalTrials
- Cluster
- Databases
- DiferentialEquations
- Distribution
- Genetics
- MachineLearning
- Multivariate



- 1 Introducción
- 2 Operaciones básicas
- 3 Análisis multivariante**
- 4 Implementaciones
 - Distribución normal multivariante
 - Inferencia sobre DNM
- 5 Bibliografía

Nuestro dataset

Característica del data set	Multivariante	Nº de Instancias	178
Características de los atributos	Enteros, Reales	Nº de Atributos	13
Área	Física	Donado	01/07/1991

Fuente: Machine Learning Repository

Propietarios Originales:

Forina, M. et al, PARVUS -

*An Extendible Package for Data Exploration, Classification
and Correlation.*

*Institute of Pharmaceutical and Food Analysis and Technologies,
Via Brigata Salerno, 16147 Genoa, Italy.*




```
wine <- read.table("http://archive.ics.uci.edu/ml  
/machine-learning-databases/wine/wine.data",  
sep=" ",")
```

In [4]: wine

A data.frame: 178 x 14

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<int>
1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065	
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050	
1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185	
1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480	

- *read.csv* o *read.csv2*
- *read.delim*

Media y desviación típica. Normalización.

```
sapply(wine[2:14], mean)      sapply(wine[2:14], sd)
V2  13.0006179775281         V2  0.811826538005857
V3  2.33634831460674         V3  1.11714609761446
...                          ...
V13 2.61168539325843         V13 0.70999042876505
V14 746.893258426966         V14 314.907474276849

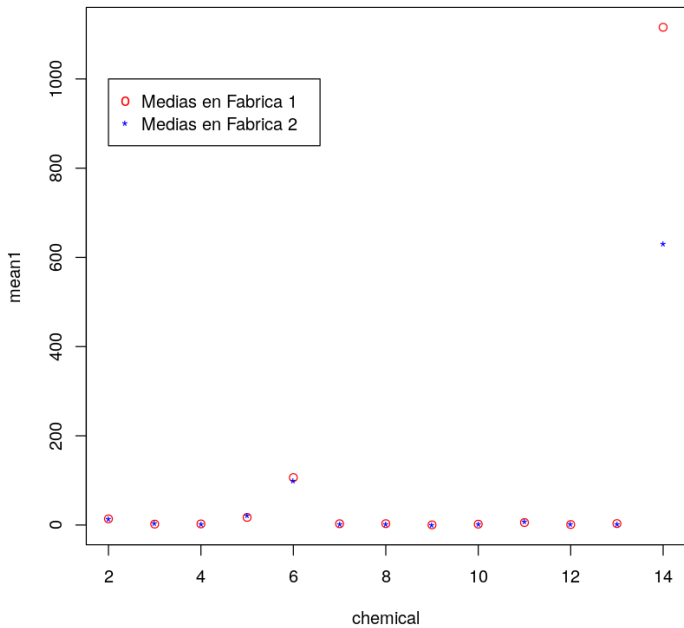
standardised_wine <- as.data.frame(scale(wine
  [2:14]))
```

Ejemplo de selección de datos

```
selection1 <- wine[wine$V1 == "1",]
selection3 <- wine[wine$V1 == "3",]

mean1 <- sapply(selection1[2:14], mean)
mean2 <- sapply(selection3[2:14], mean)

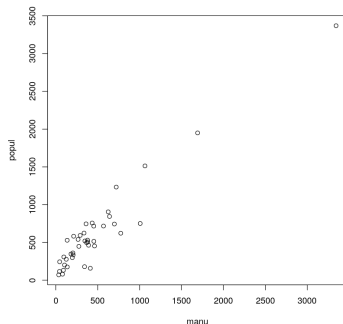
chemical <- c(2,3,4,5,6,7,8,9,10,11,12,13,14)
plot(chemical, mean1, col = "red")
points(chemical, mean2, col="blue", pch="*")
legend(2,1000, legend=c("Medias en Fabrica 1", "Medias en Fabrica 2"),
      col=c("red", "blue"),
      pch=c("o", "*"))
```



Scatterplots

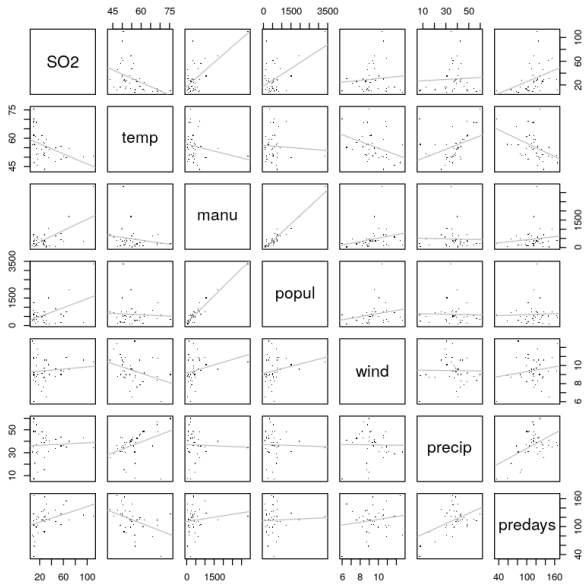
Podemos realizar scatterplots de la siguiente manera

```
plot(popul ~ manu, data = USairpollution)
```



O, dibujar todas las parejas posibles a la vez, y añadir las rectas de regresión

```
pairs(USairpollution,
      panel = function(x, y, ...) {
        points(x, y, ...)
        abline(lm(y ~ x), col = "grey")
      }, pch = ".", cex = 1.5)
```



- 1 Introducción
- 2 Operaciones básicas
- 3 Análisis multivariante
- 4 Implementaciones**
 - Distribución normal multivariante
 - Inferencia sobre DNM
- 5 Bibliografía

Implementación Teórica de una DNM

```
DNM <- setRefClass("DNM",  
                  fields = list(p = "numeric",  
                               media = "matrix"  
                               ,  
                               cov = "matrix"))
```

Ejemplo de creación de una DNM

$$\mathbf{X} = (X_1, X_2, X_3)^T \sim N_3 \left(\begin{pmatrix} 2 \\ 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 \\ 0 & 5 & -2 \\ 1 & -2 & 2 \end{pmatrix} \right)$$

```
means <- matrix(c(2,3,-1), nrow = 3, ncol = 1)
```

```
cov <- matrix(c(1, 0, 1, 0, 5, -2, 1, -2, 2),  
              nrow = 3, ncol = 3)
```

```
X <- DNM$new(p = 3, media = means, cov = cov); X
```

Función de densidad

```
# Pre: X es DNM, x matriz
funcion_densidad <- function(X, x) {
  if (dim(x)[1] != X$p)
    stop("Filas de x no coinciden con dimension de X")
  if (dim(x)[2] != 1)
    stop("x no es vector columna")
  if (det(X$cov) <= 0)
    stop("Matriz de covarianzas no es definida positiva")
  # f_X(x)
  exp(-0.5 * as.numeric(t(x - X$media) %*% solve(X$cov) %*% (x - X$media))) / ((2 * pi)^(X$p / 2)
    * sqrt(det(X$cov)))
}
```

Función característica

```
# Pre: X es DNM, t es matriz
funcion_caracteristica <- function(X, t) {
  # Comprobacion de rango
  if (dim(t)[1] != X$p)
    stop("Filas de t no coinciden con dimension de X")
  if (dim(t)[2] != 1)
    stop("t no es vector columna")
  # psi_X(t)
  exp(as.complex(1i * t(t) %*% X$media - 0.5 *
    t(t) %*% X$cov %*% t))
}
```

Teorema de la transformación lineal

Necesita como argumentos una DNM $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \Sigma)$, una matriz $B \in \mathcal{M}_{q \times p}$ ($q \leq p$) y un vector $\mathbf{b} \in \mathbb{R}^q$. Entonces devuelve la DNM definida como $\mathbf{Y} = B\mathbf{X} + \mathbf{b}$, entonces $\mathbf{Y} \sim N_q(B\boldsymbol{\mu} + \mathbf{b}, B\Sigma B^T)$, con $B\Sigma B^T > 0$.

```
transformacion_lineal <- function(X, B, b) {  
  # Comprobaciones de rango  
  # Nueva DNM  
  media_y = matrix(B %*% X$media + b, ncol = 1)  
  cov_y = matrix(B %*% X$cov %*% t(B), nrow =  
    dim(B)[1])  
  DNM$new(p = dim(B)[1], media = media_y, cov  
    = cov_y)  
}
```

Ejemplo

Ejemplo con la \mathbf{X} anterior, $B = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix}$, $b = (0)$:

```
B = matrix(c(1, 1, 0), nrow = 1, ncol = 3)
```

```
b = matrix(0, nrow = 1, ncol = 1)
```

```
Y <- transformacion_lineal(X, B, b); Y
```

```
---
```

```
Reference class object of class "DNM"
```

```
Field "p":
```

```
[1] 1
```

```
Field "media":
```

```
[,1]
```

```
[1,] 5
```

```
Field "cov":
```

```
[,1]
```

```
[1,] 6
```

Marginalización

Dada una DNM $\mathbf{X} = (X_1, \dots, X_p)^T \sim N_p(\boldsymbol{\mu}, \Sigma)$, que consiste en tomar el subvector $\mathbf{X}_{\mathbf{r}} = (X_{r_1}, \dots, X_{r_q})^T$ con $\mathbf{r} = (r_1, \dots, r_q)^T$, $r_1, \dots, r_q \in \{1, \dots, p\}$, $q \leq p$; obteniendo $\mathbf{X}_{\mathbf{r}} \sim N_q(\boldsymbol{\mu}_{\mathbf{r}}, \Sigma_{\mathbf{r}})$ donde: - $\boldsymbol{\mu}_{\mathbf{r}}$ es el subvector de $\boldsymbol{\mu}$ correspondiente a \mathbf{r} . - $\Sigma_{\mathbf{r}}$ es la submatriz de Σ definida por las filas y columnas correspondientes a \mathbf{r} .

```
# Pre: X es DNM, r es matriz
marginalizar <- function(X, r) {
  # Comprobar rangos
  # mu_r
  media_r <- matrix(X$media[r, ], ncol = 1)
  # sigma_r
  cov_r <- matrix(X$cov[r, r], nrow = dim(r)
    [1])
  # X_r
  DNM$new(p = dim(r)[1], media = media_r, cov =
    cov_r)
}
```


Particionamiento independiente de una DNM

Dada una DNM $\mathbf{X} = (X_1, \dots, X_p)^T \sim N_p(\boldsymbol{\mu}, \Sigma)$ con $p > 1$ y $\Sigma > 0$ podemos realizar una partición de $\mathbf{X} = (\mathbf{X}_{(1)}^T, \mathbf{X}_{(2)}^T)^T$ con $\boldsymbol{\mu} = (\boldsymbol{\mu}_{(1)}^T, \boldsymbol{\mu}_{(2)}^T)^T$ y $\Sigma = \begin{pmatrix} \Sigma_{(11)} & \Sigma_{(12)} \\ \Sigma_{(21)} & \Sigma_{(22)} \end{pmatrix}$ de manera que $\mathbf{X}_{(1)} = (X_1, \dots, X_q)^T$, y $\mathbf{X}_{(2)} = (X_{q+1}, \dots, X_p)^T$ ($1 \leq q < p$).

```
particiones_independientes <- function(X, q) {
  media_1 = matrix(X$media[1:q], ncol = 1)
  cov_11 = matrix(X$cov[1:q,1:q], nrow = q)
  X_1 <- DNM$new(p = q, media = media_1, cov = cov_11)
  # X_2 - cov_21 * cov_11^-1 * cov_12
  media_2 = matrix(X$media[(q+1):X$p], ncol=1)
  cov_22 = matrix(X$cov[(q+1):X$p, (q+1):X$p], nrow = X$p - q)
  cov_12 = matrix(X$cov[1:q, (q+1):X$p], nrow=q)
  cov_21 = matrix(X$cov[(q+1):X$p, 1:q], ncol=q)
  X_2 <- DNM$new(p = X$p - q,
    media = media_2 - cov_21 %*% solve(cov_11) %*%
      media_1,
    cov = cov_22 - cov_21 %*% solve(cov_11) %*% cov_
      12)

  # Devolvemos
  c(X_1, X_2)
}
```

Distribución condicionada

En las condiciones del apartado anterior, tenemos que la distribución condicionada de $\mathbf{X}_{(2)}$ dado $\mathbf{X}_{(1)} = \mathbf{x}_{(1)}$ es una DNM con

$$\mathbf{X}_{(2)} \sim N_{p-q}(\boldsymbol{\mu}_{(2)} + \boldsymbol{\Sigma}_{(21)}\boldsymbol{\Sigma}_{(11)}^{-1}(\mathbf{x}_{(1)} - \boldsymbol{\mu}_{(1)}), \boldsymbol{\Sigma}_{(22)} - \boldsymbol{\Sigma}_{(21)}\boldsymbol{\Sigma}_{(11)}^{-1}\boldsymbol{\Sigma}_{(12)}).$$

```

particion_condicionada <- function(X, q, x, dado_x2) {
  media_1 = matrix(X$media[1:q], ncol = 1)
  media_2 = matrix(X$media[(q+1):X$p], ncol = 1)
  cov_11 = matrix(X$cov[1:q,1:q], nrow = q)
  cov_22 = matrix(X$cov[(q+1):X$p, (q+1):X$p], nrow = X$p - q)
  cov_12 = matrix(X$cov[1:q, (q+1):X$p], nrow = q)
  cov_21 = matrix(X$cov[(q+1):X$p, 1:q], ncol = q)
  # DNM condicionada
  if (dado_x2) {
    p_cond = q
    media_cond = media_1 + cov_12 %*% solve(cov_22) %*% (x -
      media_2)
    cov_cond = cov_11 - cov_12 %*% solve(cov_22) %*% cov_21
  } else {
    p_cond = X$p - q
    media_cond = media_2 + cov_21 %*% solve(cov_11) %*% (x -
      media_1)
    cov_cond = cov_22 - cov_21 %*% solve(cov_11) %*% cov_12
  }
  DNM$new(p = p_cond, media = media_cond, cov = cov_cond)
}

```

Con $\mathbf{X} = (X_1, \dots, X_p)^T \sim N_p(\boldsymbol{\mu}, \Sigma)$, $\Sigma > 0$, consideramos una muestra aleatoria simple dada por $\mathbf{X} = (\mathbf{X}_1^T, \dots, \mathbf{X}_N^T)^T$, donde N es el tamaño muestral; de manera que cada observación se representa con $\mathbf{X}_\alpha, \alpha \in \{1, \dots, N\}$.

Veremos como hacer inferencia sobre la DNM que determina \mathbf{X} .

Muestra aleatoria simple

Dada $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \Sigma)$, $\Sigma > 0$, podemos generar una muestra aleatoria simple de \mathbf{X} de tamaño N , mediante la siguiente función:

```
muestra_aleatoria <- function(N, media, cov) {  
  mvrnorm(N, media, cov)  
}
```

```
muestra_aleatoria_EMV <- function(N, X) {  
  mvrnorm(N, X$media, X$cov)  
}
```

Muestra aleatoria simple

Ejemplo de la m.a.s. de tamaño 14 de la DNM

$$\mathbf{X} = (X_1, X_2, X_3)^T \sim N_3 \left(\begin{pmatrix} 2 \\ 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & -0,5 \\ 1 & -0,5 & 2 \end{pmatrix} \right):$$

```
X_DNM <- DNM$new(p = 3,  
  media = matrix(c(2,3,-1), ncol = 1),  
  cov = matrix(c(1,0,1,0,1,-0.5,1,-0.5,  
    2), ncol = 3, nrow = 3))  
X <- muestra_aleatoria_EMV(20, X_DNM); X[1:10,]
```

Muestra aleatoria simple

A matrix: 10 × 3 of type dbl

2.4044045	3.758882	0.4791484
1.9707922	2.475678	0.7559109
1.1323421	1.754754	-0.3500211
0.9028611	4.558904	-2.6476227
3.1241580	3.700116	-0.3400015
2.2546504	3.728964	-2.3389588
2.1142132	2.730817	-1.1191378
0.9457807	3.874198	-1.8523743
2.8161923	2.932069	0.2482438
2.8143862	1.730912	1.2577925

Media muestral

Implementamos la media muestral $\bar{\mathbf{X}} = \frac{1}{N} \sum_{\alpha=1}^N \mathbf{X}_{\alpha} = \begin{pmatrix} \bar{X}_1 \\ \vdots \\ \bar{X}_p \end{pmatrix}$

Además este es el estimador máximo verosímil ($\hat{\boldsymbol{\mu}} = \bar{\mathbf{X}}$).

```
media_muestral <- function(X) {  
  matrix(colMeans(X), ncol = 1)  
}
```

Media muestral

Ejemplo con X anterior:

```
media_muestral(X)
```

A matrix: 3 ×
1 of type dbl

2.2408009

3.1898752

-0.7302118

Matrices muestrales

Implementamos las siguientes matrices:

- Matriz de dispersiones muestral: $A = \sum_{\alpha=1}^N (\mathbf{X}_{\alpha} - \bar{\mathbf{X}})(\mathbf{X}_{\alpha} - \bar{\mathbf{X}})^T$
- Matriz de covarianzas muestral: $S_N = \frac{1}{N}A$
- Matriz de cuasi-covarianzas muestral: $S_{N+1} = \frac{1}{N-1}A$ (también llamada matriz de covarianzas muestral)
- Matriz de correlaciones muestral: $R = D^{1/2}S_N D^{-1/2}$, donde D es la diagonal de S_N

Sabemos que $\hat{\Sigma} = S_N$, $T = S_{N+1}$ es estimador eficiente de Σ , y $\hat{\rho} = R$ (coeficientes de correlacion lineal de Pearson).

Matrices muestrales

```
disp_muestral <- function(X) {  
  cov(X) * (nrow(X) - 1)  
}  
  
cov_muestral <- function(X) {  
  disp_muestral(X) / nrow(X)  
}  
  
cuasicov_muestral <- function(X) {  
  disp_muestral(X) / (nrow(X)-1)  
}  
  
cor_muestral <- function(X) {  
  cor(X)  
}
```

Matrices muestrales

Ejemplo con la X anterior:

```
print(dispmuestral(X))  
print(covmuestral(X))  
print(cuasicovmuestral(X))  
print(cormuestral(X))
```

	[,1]	[,2]	[,3]
[1,]	21.479470	-1.012076	22.441216
[2,]	-1.012076	18.552064	-9.202547
[3,]	22.441216	-9.202547	39.923667

	[,1]	[,2]	[,3]
[1,]	1.0739735	-0.0506038	1.1220608
[2,]	-0.0506038	0.9276032	-0.4601274
[3,]	1.1220608	-0.4601274	1.9961834

	[,1]	[,2]	[,3]
[1,]	1.13049840	-0.05326715	1.1811166
[2,]	-0.05326715	0.97642444	-0.4843446
[3,]	1.18111663	-0.48434460	2.1012456

	[,1]	[,2]	[,3]
[1,]	1.00000000	-0.05069968	0.7663363
[2,]	-0.05069968	1.00000000	-0.3381401
[3,]	0.76633630	-0.33814014	1.0000000

Contraste sobre μ

Sobre $\mathbf{X} \sim N_p(\mu, \Sigma)$, $\Sigma > 0$ y $\{\mathbf{X}_\alpha : \alpha = 1, \dots, N\}$, con $N > p$ una m.a.s de \mathbf{X} nos planteamos el problema de contraste

$$\begin{cases} H_0 : \mu = \mu_0 \\ H_1 : \mu \neq \mu_0 \end{cases}, \mu_0 \in \mathbb{R}^p \text{ dado.}$$

Usamos estadístico de Wishart $W = N(\bar{\mathbf{X}} - \boldsymbol{\mu}_0)^T \Sigma^{-1}(\bar{\mathbf{X}} - \boldsymbol{\mu}_0)$ sigue una $\chi_p^2(\delta)$ con $\delta = N(\boldsymbol{\mu} - \boldsymbol{\mu}_0)^T \Sigma^{-1}(\boldsymbol{\mu} - \boldsymbol{\mu}_0)$, y la función test para el problema es:

$$\Phi(X) = \begin{cases} 1 & \text{si } W > \chi_{p;\alpha}^2 \\ 0 & \text{si } W \leq \chi_{p;\alpha}^2 \end{cases},$$

donde $\chi_{p;\alpha}^2$ representa el valor de una distribución χ_p^2 que deja a su derecha una probabilidad α .

En nuestro caso si no proporcionamos la probabilidad α nos devolverá el p-value, que es la probabilidad que deja W a su derecha.

```
media_test_sigma <- function(X, media_0, cov,
  alpha = NA) {
  N <- nrow(X)
  p <- ncol(X)
  media <- media_muestral(X)
  W <- N * (t(media - media_0) %*% solve(cov) %
    %*% (media - media_0))
  p_value <- 1 - pchisq(W, p)
  if (!is.na(alpha))
    cat("\nResultado del test:", as.logical(
      p_value < alpha))
  else
    cat("\np-value:", p_value)
}
```


Σ conocida

Probamos el test con el X anterior:

```
print(X_DNM$media)
print(media_muestral(X))
media_test_sigma(X, X_DNM$media, X_DNM$cov)
media_test_sigma(X, media_muestral(X), X_DNM$cov)
media_test_sigma(X, c(2.5, 3.5, -1.2), X_DNM$cov)
media_test_sigma(X, X_DNM$media, X_DNM$cov, alpha = 0.05)
```

```
      [,1]
[1,]      2
[2,]      3
[3,]     -1

      [,1]
[1,] 2.2408009
[2,] 3.1898752
[3,] -0.7302118
```

```
p-value: 0.5143849
p-value: 1
p-value: 0.007210622
Resultado del test: FALSE
```

Usamos el estadístico de T^2 de Hotelling, con

$T^2 = N(\bar{\mathbf{X}} - \boldsymbol{\mu}_0)^T S_{N-1}^{-1}(\bar{\mathbf{X}} - \boldsymbol{\mu}_0)$, donde $\frac{T^2}{N-1} \frac{N-p}{p} \sim F_{p;N-p}(\delta)$ con $\delta = N(\boldsymbol{\mu} - \boldsymbol{\mu}_0)^T \Sigma^{-1}(\boldsymbol{\mu} - \boldsymbol{\mu}_0)$, y entonces la función test para el problema es:

$$\Phi(X) = \begin{cases} 1 & \text{si } (N-p)T^2 > (N-1)pF_{p;N-p;\alpha} \\ 0 & \text{si } (N-p)T^2 \leq (N-1)pF_{p;N-p;\alpha} \end{cases},$$

donde $F_{p;N-p;\alpha}$ representa el valor de una distribución $F_{p;N-p}$ que deja a su derecha una probabilidad α .

Σ desconocida

```
media_test <- function(X, media_0, alpha = NA) {  
  N <- nrow(X)  
  p <- ncol(X)  
  media <- media_muestral(X)  
  S <- cuasicov_muestral(X)  
  T <- N * (t(media - media_0) %*% solve(S) %*%  
    (media - media_0))  
  T_val <- T * (N - p) / (p * (N - 1))  
  p_value <- 1 - pf(T_val, p, N - p)  
  if (!is.na(alpha))  
    cat("\nResultado del test:", as.logical(  
      p_value < alpha))  
  else  
    cat("\np-value:", p_value)  
}
```

Σ desconocida

Probamos con X :

```
print(X_DNM$media)
print(media_muestral(X))
media_test(X, X_DNM$media)
media_test(X, media_muestral(X))
media_test(X, c(2.1, 3.1, -1.2))
media_test(X, c(3, 0, 0), alpha = 0.05)
```

```
      [,1]
[1,]     2
[2,]     3
[3,]    -1
```

```
      [,1]
[1,] 2.2408009
[2,] 3.1898752
[3,] -0.7302118
```

```
p-value: 0.5912787
p-value: 1
p-value: 0.2963594
Resultado del test: TRUE
```

Superficies de confianza

Ahora nos planteamos como representar las superficies de confianza de μ , es decir, dada X_i, X_j de \mathbf{X} la región de \mathbb{R}^2 donde se pueda encontrar μ con cierta probabilidad $1 - \alpha$.

Para formar las regiones de confianza para el vector de medias μ consideramos que:

$$P[W \leq \chi_{p;\alpha}^2] = 1 - \alpha,$$

donde W era el estadístico de Wishart definido en el test de contraste. Tenemos entonces que la región de confianza al $100(1 - \alpha)\%$ del vector de medias μ está definida por todos los $\mu_0 \in \mathbb{R}^p$ tales que cumplen:

$$N(\bar{\mathbf{X}} - \mu_0)^T \Sigma^{-1} (\bar{\mathbf{X}} - \mu_0) \leq \chi_{p;\alpha}^2$$

Se representa como elipse, con Σ y $\sqrt{\frac{1}{N} \chi_{p;\alpha}^2}$.

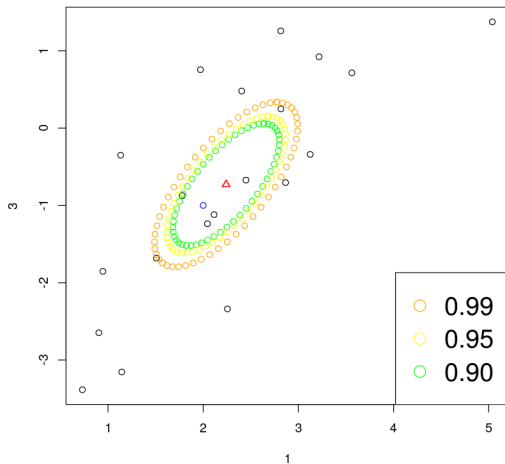
Σ conocida

```
ellipse_medias_sigma <- function(X, id, cov, alpha
, col = "black", pch = 1, draw = FALSE) {
  p <- ncol(X)
  N <- nrow(X)
  media <- media_muestral(X)
  radio <- sqrt(qchisq(alpha,p) / N)
  ellipse <- ellipse(center = media[id], shape =
    cov[id,id], radius = radio, draw = FALSE,
    pch = pch, col = col)
  if (!draw)
    plot(ellipse, pch = pch, col = col)
  else
    points(ellipse, pch = pch, col = col)
  points(matrix(media[id], ncol = 2), pch = 2,
    col = "red")
}
```

Ejemplo con X con $\alpha = 0,99, 0,95, 0,90$:

```
plot(X[, c(1,3)], xlab = "1", ylab = "3")
ellipse_medias_sigma (X, c(1,3), X_DNM$cov, 0.99,
  col = "orange", draw = TRUE)
points(matrix(X_DNM$media[c(1,3)]), ncol=2), pch =
  1, col = "blue")
ellipse_medias_sigma (X, c(1,3), X_DNM$cov, 0.90,
  col = "green", pch = 1, draw = TRUE)
ellipse_medias_sigma (X, c(1,3), X_DNM$cov, 0.95,
  col = "yellow", pch = 1, draw = TRUE)
```


Σ desconocida



Para formar las regiones de confianza para el vector de medias μ consideramos que:

$$P \left[T^2 \leq \frac{p(N-1)}{N-p} F_{p;N-p;\alpha} \right] = 1 - \alpha,$$

donde T^2 era el estadístico de Hotelling definido en el test de contraste. Tenemos entonces que la región de confianza al $100(1 - \alpha)\%$ del vector de medias μ está definida por todos los $\mu_0 \in \mathbb{R}^p$ tales que cumplen:

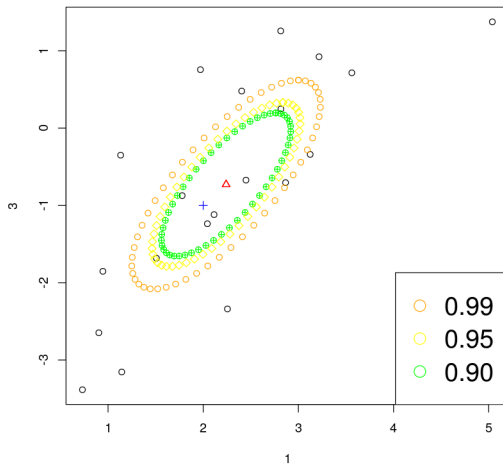
$$N(\bar{\mathbf{X}} - \mu_0)^T S_{N-1}^{-1} (\bar{\mathbf{X}} - \mu_0) \leq \frac{p(N-1)}{N-p} F_{p;N-p;\alpha}$$

Se representa como elipse, usando S_{N-1} y $\sqrt{\frac{p(N-1)}{N(N-p)} F_{p;N-p;\alpha}}$.

```
ellipse_medias <- function(X, id, alpha, col, pch,
  draw) {
  p <- ncol(X)
  N <- nrow(X)
  media <- media_muestral(X)
  S <- cuasicov_muestral(X)
  radio <- sqrt(p*(N-1)*qf(alpha,p,N-p)/(N*(N-p
    )))
  ellipse <- ellipse(center = media[id],
    shape = S[id,id], radius = radio, draw =
      FALSE, pch = pch, col = col)
  plot(ellipse)
  points(matrix(media[id], ncol = 2), pch = 2,
    col = "red")
}
```

```
plot(X[, c(1,3)], xlab = "1", ylab = "3")
ellipse_medias(X, c(1,3), 0.99, col = "orange",
  draw = TRUE)
points(matrix(X_DNM$media[c(1,3)]), ncol=2), pch =
  3, col = "blue")
ellipse_medias(X, c(1,3), 0.90, col = "green", pch
  = 10, draw = TRUE)
ellipse_medias(X, c(1,3), 0.95, col = "yellow",
  pch = 5, draw = TRUE)
```

Σ desconocida



- 1 Introducción
- 2 Operaciones básicas
- 3 Análisis multivariante
- 4 Implementaciones
 - Distribución normal multivariante
 - Inferencia sobre DNM
- 5 Bibliografía

Bibliografía

Using R for Data Analysis and Graphics. Introduction, Code and Commentary J.H. Maindonald (2004)

Applied Predictive Modeling Max Kuhn & Kjell Johnson (2013)

An Introduction to Statistical Learning with Applications in R Gareth James, Daniela Witten, Trevor Hastie & Robert Tibshirani (6th printing 2015)

The Elements of Statistical Learning. Data Mining, Inference, and Prediction. Trevor Hastie, Robert Tibshirani & Jerome Friedman (2nd edition 2001)

An Introduction to Applied Multivariate Analysis with R. Brian Everitt & Torsten Hothorn (2011)

R-Forge (Consultado 21/01/2020)

<http://r-forge.r-project.org/>

Cran R Contributed Packages (Consultado 21/01/2020)

<https://cran.r-project.org/web/packages/>

Cran R Contributed Packages Captura Diciembre 2019

<https://web.archive.org/web/20191210081209/http://cran.r-project.org/web/packages>

Cran R Task Views (Consultado 21/01/2020)

<https://cran.r-project.org/web/views/>

Tutorialspoint (Consultado 21/01/2020)

<https://www.tutorialspoint.com/r/index.htm>