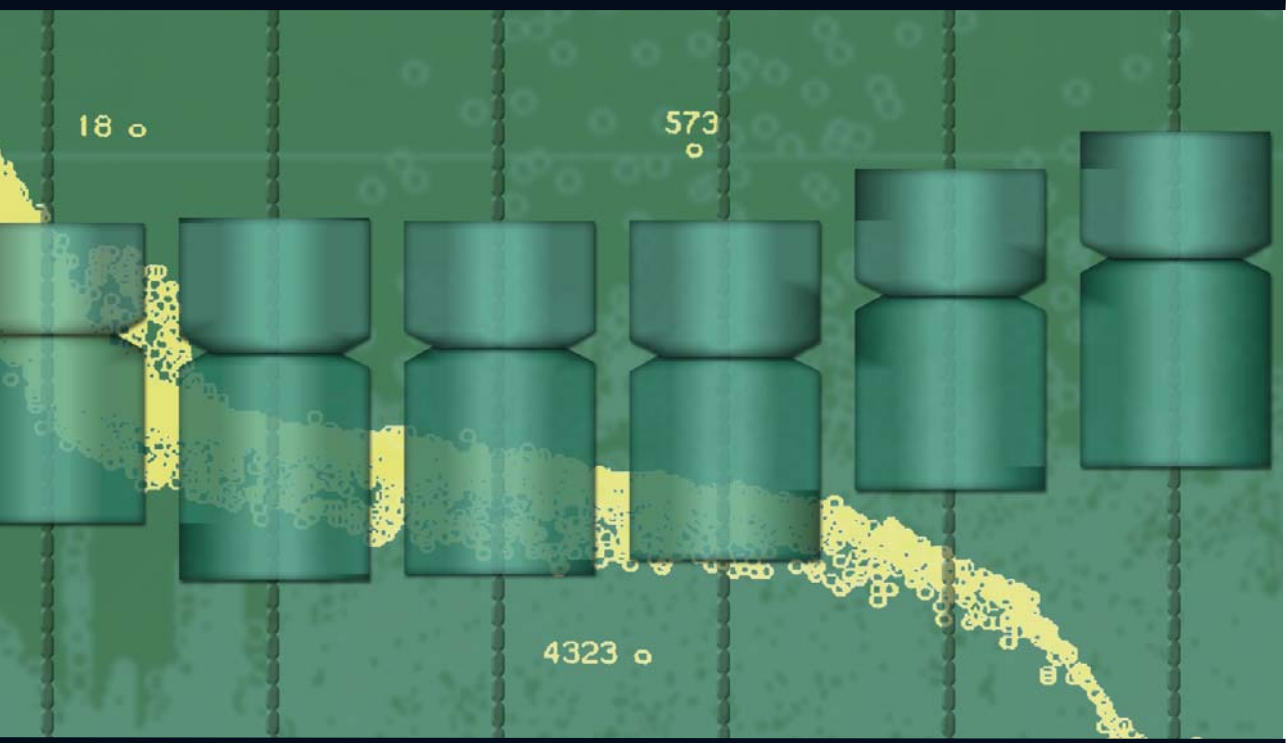
Control de Calidad

Metodología para el análisis previo a la modelización de datos en procesos industriales.

Fundamentos teóricos y aplicaciones prácticas con R



Manuel Castejón Limas,
Joaquín Ordieres Meré,
Francisco Javier de Cos Juez,
Francisco Javier Martínez de Pisón

UNIVERSIDAD DE LA RIOJA

CONTROL DE CALIDAD

MONOGRAFÍAS DE I+D

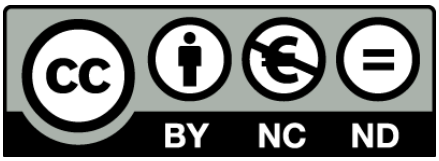
nº 1

Manuel Castejón Limas, Joaquín Ordieres Meré,
Francisco Javier de Cos Juez, Francisco Javier Martínez de Pisón

CONTROL DE CALIDAD

Metodología para el análisis previo a la modelización de datos en procesos industriales. Fundamentos teóricos y aplicaciones prácticas con R.

UNIVERSIDAD DE LA RIOJA
SERVICIO DE PUBLICACIONES
2014



publicaciones.unirioja.es

E-mail: publicaciones@unirioja.es

ISBN: 978-84-697-0032-7

Índice de contenidos

Índice de figuras

Capítulo 1
Introducción

1.1. Presentación

Las nuevas tecnologías emergentes han facilitado la aparición de nuevas técnicas de modelización matemática que dejan obsoletas otras vías clásicas más difundidas entre la comunidad científica.

La modelización de procesos industriales se caracteriza por la necesidad, por parte del ingeniero encargado del proyecto, de una serie de conocimientos que generalmente se escapan del alcance de su formación puramente académica. Las herramientas y conceptos necesarios para la realización de semejante tarea proceden de múltiples campos como la inteligencia artificial, la inferencia estadística, las tecnologías de la información o la matemática aplicada.

Este trabajo viene a rellenar un hueco en este área de conocimiento, con el animo de aunar en este documento las nociones básicas que, un investigador dedicado al campo de la modelización de datos en procesos industriales, debe reunir al enfrentarse a la masiva cantidad de información muestral que caracteriza este tipo de proyectos. Es un texto, pues, dirigido a nuevos investigadores y por tanto pretende tener un claro carácter pedagógico.

1.2. Objetivos

El contenido de esta obra repasa aspectos básicos del análisis multivariante. Pretende presentar no sólo los fundamentos teóricos sino además su aplicación práctica en el software que a nuestro juicio ha resultado más adecuado.

Se detallan las herramientas básicas de análisis estructural de los datos procedentes de un proceso productivo real y del análisis descriptivo de las muestras observadas. También se recogen los métodos de clasificación de las muestras en distintos grupos funcionales o áreas del espacio de estado, así como los tests necesarios para validar la conformidad de las distintas clases obtenidas con los algoritmos descritos.

Se ha dedicado especial atención a los capítulos concernientes al análisis de normalidad multivariante y a los métodos de detección de casos atípicos. Especialmente es este último apartado un campo donde aún puede realizarse una investigación intensa a fin de mejorar los algoritmos existentes y extenderlos al caso de distribuciones no normales multivariantes.

A continuación se presenta el estudio de la dimensión fractal de la estructura de las observaciones, herramienta que revela la dimensión del modelo más adecuado para la explicación de las variables de especial interés.

El ajuste de modelos se realiza por dos vías bien distintas, aplicando métodos clásicos de modelización lineal y mediante el entrenamiento de redes neuronales, introduciendo al nuevo investigador también en este campo de las tecnologías de la información.

1.3. Alcance

Este trabajo pretende centrarse en obtener una metodología de trabajo en la etapa de análisis previa a la modelización de los datos procedentes de procesos industriales. Por ello se dedica mayor atención al tratamiento previo de la información observada y no se profundiza en la obtención de modelos no lineales por técnicas clásicas. Únicamente se muestran ejemplos de implementación de modelos lineales y lineales generalizados, dejando quizás para una próxima obra la profundización en los modelos no lineales según técnicas clásicas. Como aproximación no lineal al problema de la modelización se presenta la utilización de redes neuronales, en virtud de su capacidad de aproximación a una función no lineal multivariante con tolerancia controlable por el investigador.

1.4. Metodología de presentación

A lo largo del documento se presentan los distintos capítulos intentando seguir, en la medida de los posible, una estructura según la cual primero se presentan los fundamentos básicos de las técnicas aplicables en la resolución de problemas concretos, tras lo cual se desarrolla la implementación de tales técnicas en el software elegido. Finalmente se procura exponer las conclusiones a las que la reflexión sobre los resultados conduzca. Se procura, asimismo, utilizar la siguiente tipografía a efectos de facilitar al nuevo investigador la ubicación concreta del punto de desarrollo, dentro de los distintos apartados, en el que se encuentra:

- Arial 12 para el texto descriptivo.
- `Courier 11 para los comandos internos de` **`R`**`. Estos serán precedidos del carácter ">" como PROMPT principal y "+" como PROMPT secundario si la orden se prolonga a varias líneas.`
- `Courier 9 y 10 para los resultados devueltos por R.`
- `Courier 11 para los comandos del sistema operativo UNIX. Irán precedidos por el carácter "$"`

Asimismo se procura facilitar siempre que es posible una representación gráfica del problema, lo cual queda demostrado a lo largo de las 73 figuras que se recogen en el texto.

1.5. Introducción al proceso productivo considerado

Este trabajo persigue plantear y analizar una metodología novedosa de trabajo para la modelización de procesos industriales, basándose en los propios datos suministrados por los sensores del proceso. Esto tiene especial significado a la hora de optimizar los modelos de control existentes, especialmente en el caso de los sistemas de control en bucle abierto o aquellos otros gobernados por métodos de cálculo dependientes de parámetros “ajustables”. El caso real que tomaremos como ejemplo para ilustrar el procedimiento corresponde a un tren de laminación de chapa que una empresa siderúrgica de la Unión Europea tiene actualmente en funcionamiento en sus instalaciones.

En este tren de chapa ya se han modelizado previamente algunas variables, como la fuerza de laminación o el ajuste de forma. Este estudio **se centrará en el análisis de los datos reales** de funcionamiento obtenidos durante la operación normal de la factoría, poniendo especial atención en el par aplicado en los rodillos durante la laminación pues ésta es la variable a explicar en los modelos que de este estudio se deriven.

Del modelo físico del problema se cree que las variables más relevantes para explicar el par son:

- TRRE: Par real aplicado a los rodillos de laminación.
- F: Fuerza de apriete de los rodillos de laminación.
- DR: Diferencia de espesores entre pasadas.
- PLT: Espesor de la chapa a la entrada a la pasada.
- R: Radio de los rodillos de laminación.
- TRCEQ: Contenido en carbono del acero a laminar.

donde se ha utilizado la nomenclatura interna de la empresa interesada.

El objetivo final de la empresa es mejorar la estimación de par que los modelos que posee la empresa predicen. En el proceso productivo actualmente se distinguen distintas clases de producto (y por tanto de parámetros de modelado) en función del contenido en carbono del acero de la chapa. Las clases se definen de acuerdo a los intervalos de valores (0,20], (20,24], (24,28], (28,32], (32,36], (36,40], (40,44], (44,48], (52,...], expresándose estos valores en centésimas de %, a las que nos referiremos habitualmente como clase nº 1, nº 2, …, nº 9 por comodidad. Un aspecto significativo de este estudio es analizar si esta hipótesis de existencia de distintas clases se sostiene o si por el contrario no existen evidencias intrínsecas a la estructura de los datos de tal existencia. El objetivo del presente estudio será, además del ya mencionado de la verificación de la existencia de clases, el llevar a cabo un análisis previo al proceso de modelización que asegure un control de calidad, lo que, al final, redundará en la obtención de mejores modelos.

Para estudiar los datos recogidos se utilizarán herramientas matemáticas de análisis multivariado y factorial, como puede ser el análisis de componentes principales, herramientas visuales como la técnica de reducción dimensional de Sammon, y otras herramientas estadísticas que se detallaran a lo largo del texto.

Es preciso advertir que lo que se desea presentar principalmente es la metodología de pre-tratamiento de la información, independientemente de que se haga sobre un ejemplo industrial concreto. Esto hace que, en ocasiones, se presenten pasos metodológicos que deberían descartarse en el caso concreto del ejemplo, por no corresponderse con su naturaleza, pero que en un caso general deberían de considerase y valorar la pertinencia de su aplicación.

La implementación práctica de los cálculos necesarios se realizará utilizando el paquete de análisis matemático **R**, de libre distribución[1].

R puede ser considerado como una reimplementación del lenguaje **S** desarrollado en AT&T por Rick Becker, John Chambers y Allan Wilks. **R** facilita la manipulación intensiva de datos y la generación de nuevos algoritmos de cálculo gracias a un lenguaje de programación simple, efectivo y bien desarrollado. Constituye un entorno dentro del cual gran número de técnicas estadísticas clásicas y modernas han sido implementadas. Algunas pertenecen al núcleo de la aplicación mientras que otras forman paquetes donados por sus creadores y son parte del fondo de contribuciones públicas.

Este trabajo tiene como objetivo adicional la presentación de **R** como paquete matemático de gran utilidad en este tipo de análisis, por lo que se han pormenorizado profusamente los pasos necesarios para realizar los cálculos a fin de ilustrar a otros investigadores en el uso de esta herramienta. Si bien no es indispensable que el lector disponga de conocimientos previos de `UNIX`, sí le facilitará la comprensión de los pasos llevados a cabo pues es en el marco de este sistema operativo donde se desarrollan los ejemplos prácticos presentados.

Se parte de un fichero de datos, procedente del proceso mencionado en la introducción., al que denominamos *datos.dat* y cuyas 6 columnas están separadas por tabuladores. De ellas la primera es la variable a explicar. Las primeras líneas de este fichero tienen el siguiente aspecto:

```
150 1956 15.08 233.82 463.295 14
203 1922 15.35 218.74 463.295 14
195 1835 12.54 203.39 463.295 14
152 1567 24.73 190.85 463.295 14
```

Para disponer en el entorno de **R** de los datos del fichero *datos.dat*, ejecutamos dentro de **R** el comando:

[1] Disponible en http://www.r-project.org

```
> datos ← read.table("datos.dat")
> names(datos) <- c(”TRRE”,”F”,”DR”,”PLT”,”R”,”TRCEQ”)
> dim(datos)
[1] 15019 6
```

Con lo que se dispone de un *data.frame*, llamado *datos*, de 15019 filas y 6 columnas, en el entorno de trabajo de **R**, para comenzar con los análisis oportunos.

Capítulo 2

Metodología de Trabajo

Las principales fases metodológicas son:

1. Representar sobre un plano 2-D los datos con criterios de interdistancias para "comprender" su estructura.

2. Estimar las direcciones de mayor "relevancia" y evaluar esta relevancia relativa, al objeto de decidir si alguna puede ser eliminada y a que coste.

3. Estimar la dimensión intrínseca de la estructura de datos con objeto de contrastar la estimación.

4. Caracterizar estadísticamente la distribución multivariada de los datos.

5. Eliminar las muestras que, mediante criterios de distancias de Mahalanobis, se consideren espurias.

6. Contrastar la veracidad de la hipótesis relativa a la existencia de distintas clases de comportamiento.

7. En caso de resultar veraz la anterior hipótesis, clasificar los datos en grupos de comportamiento distinto.

8. Revisar la dimensión intrínseca, después de suprimir los datos irrelevantes y generar las clases.

9. Proponer modelos matemáticos mediante técnicas clásicas que se ajusten a la estructura de los datos y expliquen suficientemente las variaciones en el valor de la función a modelar.

10. Aplicar modelos basados en redes neuronales y compararlos con los obtenidos con técnicas clásicas.

11. Validar los diferentes modelos y estimar su “calidad”.

Cada uno de estos apartados será desarrollado, sobre el ejemplo de base ya mencionado, en un capítulo independiente.

Capítulo 3

Análisis estructural de los datos del proceso

3.1. Proyección Sammon de los datos.

3.1.1. *Fundamentos*.

La interpretación geométrica de las N muestras multivariadas n-dimensionales puede llevarse a cabo de dos formas distintas.

- La interpretación más usual relaciona cada una de las N observaciones con un punto en el espacio n-dimensional generado. Este es el convenio utilizado generalmente a lo largo del trabajo. Se considera una muestra $x = \{\bar{x}_1, \bar{x}_2, \mathrm{K}, \bar{x}_N\}$, de N observaciones con cada $\bar{x}_i \in \Re^n$. Según el fichero *datos.dat* de partida esta sería una interpretación por filas.

- Puede resultar en ocasiones interesante realizar una interpretación geométrica por columnas, esto es, representar n puntos en un espacio N dimensional, esto es, asociando a cada variable un punto. Si las n variables tienen media nula, el ángulo entre cada pareja de variables está relacionado con la correlación existente entre esas dos variables. El coseno del ángulo que forman dos variables cualesquiera mide el grado de colinealidad de las variables en cuestión.

Por ejemplo para una muestra de 3 observaciones de dos componentes A y B, tales como:

	A	B
a	1	1
b	1	1.5
c	2	1.5

Tabla 1

La interpretación por filas llevaría a la representación bidimensional plana de tres puntos a, b y c (ver Fig. 1), mientras que la interpretación por columnas lleva a la ubicación de dos puntos A y B en el espacio tridimensional generado por la base de vectores $\{\vec{a}, \vec{b}, \vec{c}\}$ (Fig. 2).

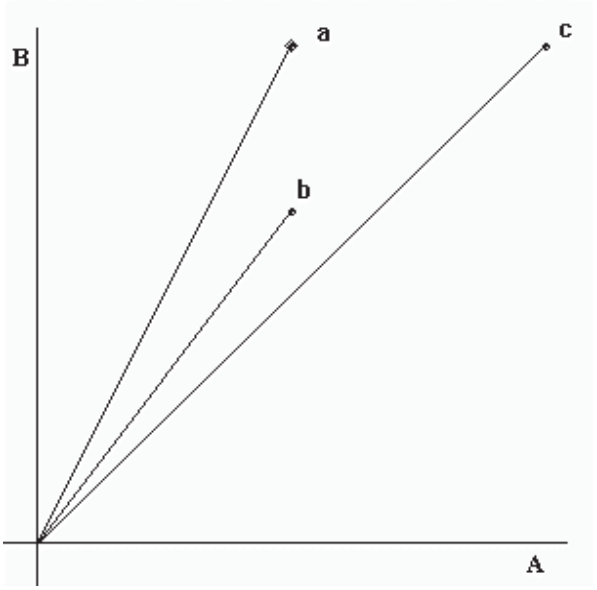


Fig. 1: Interpretación Bidimensional.

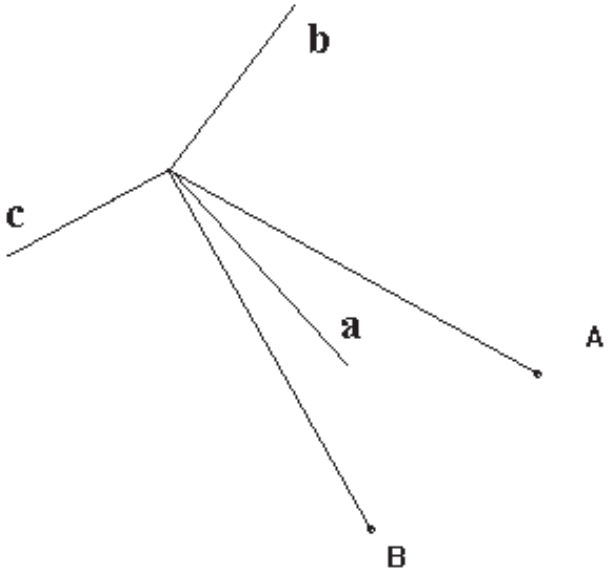


Fig. 2: Interpretación Tridimensional.

En la medida en que las direcciones de los radiovectores de A y B sigan direcciones parecidas, significará que sus componentes son similares y que por tanto la información contenida en una y otra es similar. En términos prácticos, en el caso de datos procedentes de un proceso industrial, el número de observaciones es considerablemente superior al número de componentes involucradas, por lo que se suele adoptar la interpretación por filas.

Existen distintos algoritmos matemáticos **[2]** que permiten examinar, de forma visual clara, la disposición espacial de los N puntos correspondientes a espacios vectoriales de dimensión n, $(n>2)$, a través de su proyección sobre un espacio vectorial de dimensión 2. De estos algoritmos se muestran como más útiles aquellos que intentan conservar las distancias entre los distintos puntos a la hora de proyectar, de forma que la distancia entre dos puntos cualesquiera pueda ser medible, si así se desea, directamente sobre su proyección. Entre estos algoritmos destaca el método de Sammon **[1]** que será el utilizado en este estudio.

La capacidad de ver sobre un plano la posición de cada punto permite distinguir visualmente las distintas clases de comportamiento que pudieran presentarse, por presentarse agrupados los puntos en la proyección, así como también desestimar (o estudiar separadamente) vectores demasiado alejados a cualquier grupo reconocible. Las diferentes clases estarán formadas por aquellas observaciones que presenten un comportamiento uniforme entre sí y distinto al de otros grupos existentes.

El criterio utilizado por este método a la hora de proyectar los vectores a un espacio de dimensión 2 es el de minimizar la discrepancia en las distancias resultantes en el plano frente a las distancias calculadas en el espacio de dimensión completa.

Si llamamos δ_{ij} a la distancia entre las observaciones $\overline{x}_i$ y $\overline{x}_j$, y d_{ij} a la distancia entre sus proyecciones, puede considerarse una función de discrepancia en la distancia E

$$E = \frac{1}{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\delta_{ij}} \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\frac{\left(\delta_{ij}-d_{ij}\right)^2}{\delta_{ij}}$$

El método de Sammon persigue obtener la distribución espacial de las proyecciones de los puntos dentro del plano que proporcione un valor mínimo optimizado para esta discrepancia, variando el método numérico aplicado para esta optimización de unas implementaciones a otras.

Uno de los posibles métodos de optimización **[7]** es el denominado *steepest descent algorithm*. En cada iteración la solución se mueve en la dirección del gradiente de la función de discrepancia E. En cada iteración el valor de las soluciones se modifica de acuerdo a las siguiente expresiones:

$$x_i^{m+1} = x_i^m - c\frac{\left|\frac{\partial E^{(m)}}{\partial x}\right|_{x=x_i^{(m)}}}{\left|\frac{\partial^2 E^{(m)}}{\partial x^2}\right|_{x=x_i^{(m)}}}$$

$$y_i^{m+1} = y_i^m - c\frac{\left|\frac{\partial E^{(m)}}{\partial y}\right|_{y=y_i^{(m)}}}{\left|\frac{\partial^2 E^{(m)}}{\partial y^2}\right|_{y=y_i^{(m)}}}$$

donde c es una constante que modifica la velocidad de convergencia del método y $proyección_{2D}(\bar{x}_i) = (x_i, y_i)$.

Pueden citarse dos aspectos negativos de este método, como son:

1. Su complejidad es de orden $O(N^2)$, siendo N el número de observaciones que se pretenden proyectar.

2. Si se desea ubicar en la proyección datos de nueva adquisición debe recalcularse nuevamente todo el algoritmo.

Para aumentar la velocidad del algoritmo, limitada debido al primero de estos puntos, existen distintas soluciones cuya idea básica radica en tomar un subconjunto de los datos a proyectar y aplicar sobre ellos el algoritmo, añadiendo el resto de puntos con posterioridad. Esto contradice en principio el segundo de los inconvenientes, por lo que esta adición debe realizarse independientemente del método de Sammon con algún algoritmo de limitado coste computacional, como puede ser el uso de triangularización **[1] [5]**, redes neuronales **[3] [4] [6]** y transformaciones algebraicas.

1. Triangularización.

   Este método se fundamenta en el hecho de que cuando un nuevo punto es proyectado, sus distancias a dos puntos previamente proyectados sobre el plano se conservan. Si existen n puntos en un espacio m dimensional, entonces pueden ser proyectados secuencialmente de forma que $(2n-3)$ de las $\frac{n(n-1)}{2}$ distancias son preservadas. Este es un algoritmo rápido ya que no es iterativo y sólo preserva algunas distancias. El método híbrido primero crea una base procedente de los datos proyectando p de todos los N puntos $(p<n)$ del plano, utilizando el método iterativo de Sammon. Los p puntos proyectados son entonces optimizados y el resto de puntos $(N-p)$ se proyectan secuencialmente mediante triangularización.

2. Redes Neuronales

Las redes neuronales permiten aproximar la proyección de Sammon. El patrón de entrenamiento considera como variables de entrada las coordenadas de los puntos en el espacio m-dimensional, y como salida las coordenadas (x,y) de sus proyecciones. Tras el entrenamiento de la correspondiente red se puede aproximar la ubicación de nuevos puntos por los valores que la red estime en sus salidas tras presentar en la entrada las coordenadas de estos nuevos puntos.

3. Transformación Algebraica

Otra posible solución radica en una transformación lineal de la matriz de distancias en el espacio m-dimensional de forma que su transformada sea la matriz de distancias de las proyecciones obtenidas con el Sammon. Tras obtener el operador se puede aplicar a nuevos puntos de forma inmediata.

3.1.2. *Implementación en **R***

Por limitaciones en la gestión de memoria se han dividido los datos en tres bloques, cada uno de ellos con aproximadamente 5000 filas. Cada una de estas tres proyecciones se ha dividido en dos gráficos por idénticas razones, puesto que el programa *xgobi*[2] se ha mostrado incapaz de mostrar con comodidad más observaciones.

xgobi es un software de representación visual de datos multivariantes, desarrollado por Deborah F. Swayne, Di Cook y Andreas Buja. Se encuentra implementado en prácticamente todas las plataformas y soporta comunicación entre procesos lo que le permite acoplarse modularmente a otros paquetes software como **R** o Arcview.

```
> prim ← datos[1:5000 , ]

> segu ← datos[5001:10000 , ]

> terc ← datos[10001:nrow(d) ,  ]

> prim.sam <- sammon(prim, 2, maxit=10000, tol=0.001)

> segu.sam <- sammon(segu, 2, maxit=10000, tol=0.001)

> terc.sam <- sammon(terc, 2, maxit=10000, tol=0.001)
```

En los gráficos de las proyecciones de Sammon se han utilizado colores distintos para cada una de las presuntas clases (atendiendo al Carbono equivalente).

Previo al cálculo de la dimensión fractal es pertinente eliminar las observaciones correspondientes a puntos que en los gráficos de Sammon

[2] Disponible en http://www.research.att.com/areas/stat/xgobi/

residían en la lejanía. Estos puntos pueden identificarse y eliminarse bien utilizando *xgobi* o con el mismo **R**.

```
> prim.all ← cbind(prim, prim.sam$rproj)
> segu.all ← cbind(segu, segu.sam$rproj)
> terc.all ← cbind(terc, terc.sam$rproj)
> plot(prim.all[,7], prim.all[,8])
> identify(prim.all[,7], prim.all[,8])
[1]   18  573 4323
```

Resulta comodo utilizar la función *dev2bitmap* para volcar el contenido de la gráfica a un fichero.

```
> dev2bitmap(file="sam1.bmp",width=6, height=6, res=300,
+ type="pngmono")
```

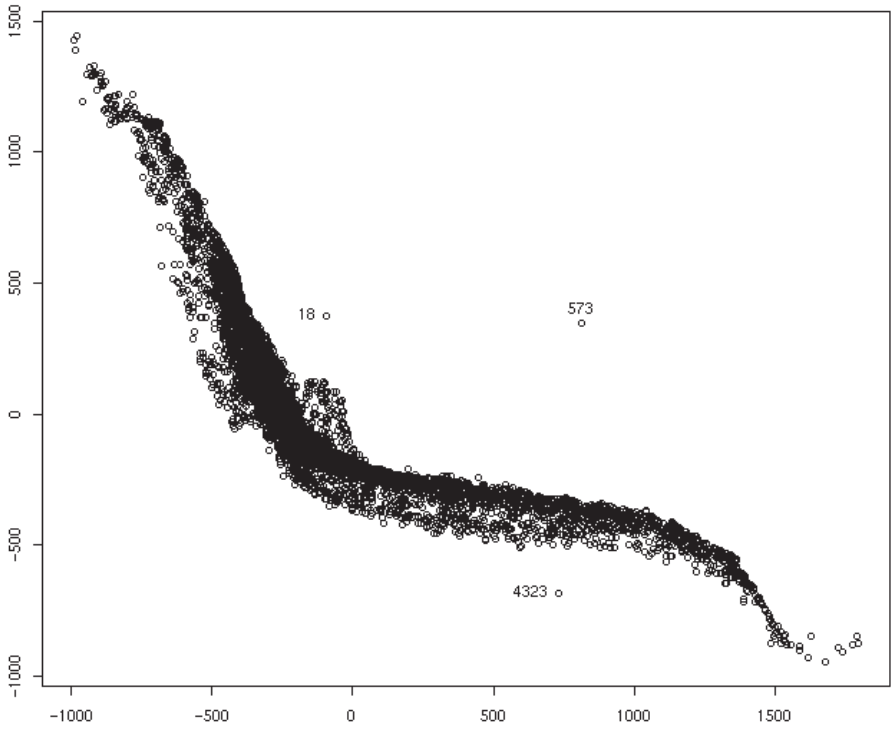


Fig. 3: Proyección Sammon del 1er tercio de los datos.

```
> plot(segu.all[,7], segu.all[,8])
> identify(segu.all[,7], segu.all[,8])
[1] 3354
```

Resultaría el gráfico de la Fig. 4 y con

```
> plot(terc.all[,7], terc.all[,8])
> identify(terc.all[,7], terc.all[,8])
[1]  544 1460 3504 3698
```

se obtendría la Fig. 5.

Para eliminarlos

```
> malos ← c(18,573,4323)
> prim.all ← prim.all[-malos , ]
> malos ← 3354
> segu.all ← segu.all[-malos , ]
> malos ← c(544,1460,3504,3698)
> terc.all ← terc.all[-malos , ]
> datos.nuevos ← rbind(prim.all, segu.all, terc.all)
> datos.nuevos ← datos.nuevos[ ,c(-7,-8)]
```

En *datos.nuevos* existe una copia de los datos originales excepto aquellas observaciones eliminadas.

```
> cortes ← c(13,20,24,28,32,36,40,44,48,63)
> clase.datos ← cut(datos.nuevos[,6], cortes)
> clase.prim.all ← cut(prim.all[,6],cortes)
> clase.segu.all ← cut(segu.all[,6],cortes)
> clase.terc.all ← cut(segu.all[,6],cortes)
> prim.colors ← as.numeric(clase.prim.all)
> segu.colors ← as.numeric(clase.segu.all)
> terc.colors ← as.numeric(clase.terc.all)
```

En este caso se prefiere el uso del comando *cut* frente a *factor* pues el primero permite generar factores cuyos niveles vengan dados por puntos de corte, permitiendo especificar entonces los intervalos de valores que forman cada una de las clases. En el caso de haber utilizado *factor* cada valor numérico habría generado un nivel o clase.

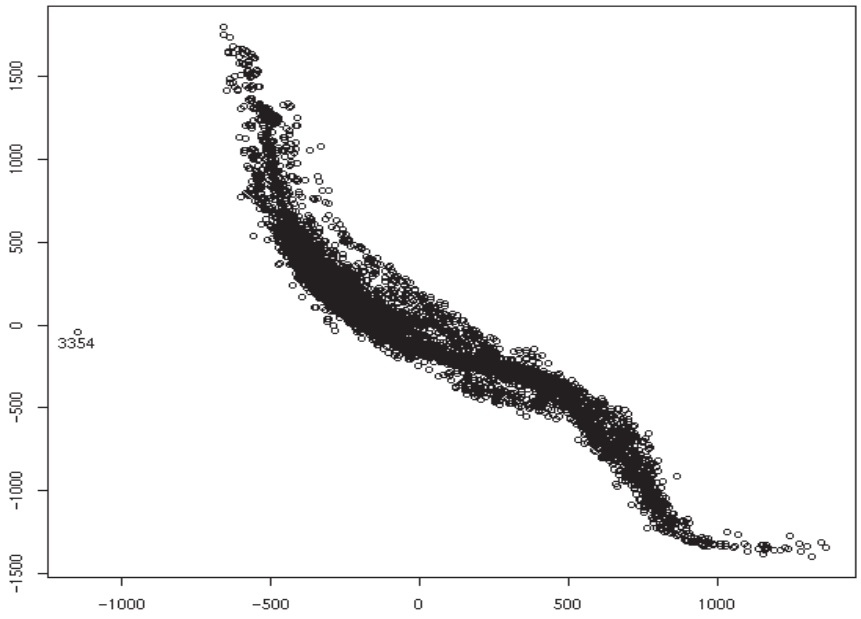


Fig. 4: Proyección Sammon del 2º tercio de los datos

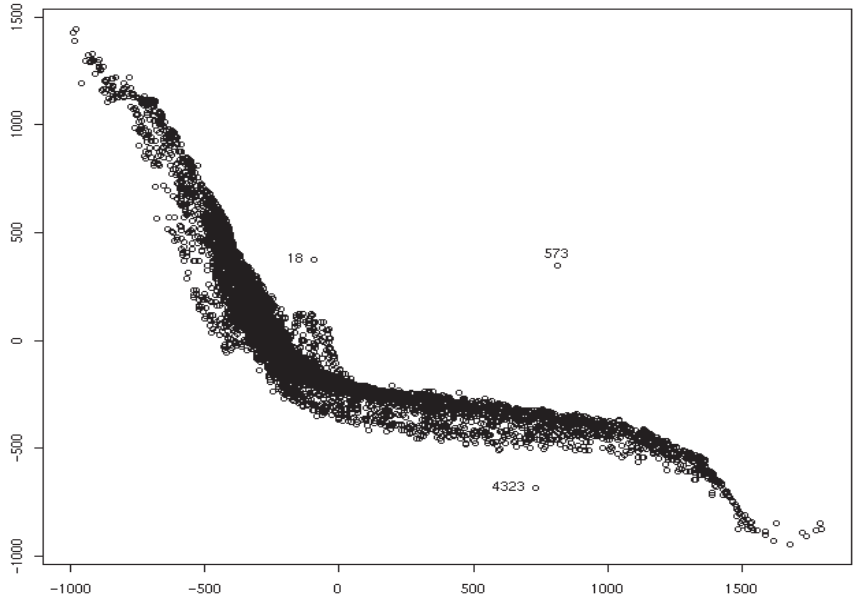


Fig. 5: Proyección Sammon del 3er tercio de los datos

Si se desea visualizar los datos en *xgobi,* ha de pasársele un vector con cadenas que expresen los códigos de colores. Es necesario convertir cada valor numérico en una secuencia de caracteres indicadora del color asignado.

Si nos basta con una paleta básica de 7 colores, la instrucción palette() proporciona la cadena correspondiente a cada color.

```
> palette()[1:14]
 [1] "black" "red" "green3" "blue" "cyan" "magenta" "yellow"
 [8] "white" "NA"  "NA" "NA" "NA" "NA" "NA"
```

Si se desea mayor flexibilidad en la variedad de colores asignados, puede resolverse fácilmente gracias al particular estilo de programación de **R**.

```
> cadena.colores <- c("Red","Yellow","Gray","Brown",
+  "Pink","Green","White","Orchid")
> prim.colors.cad <- cadena.colores[prim.colors]
```

Es importante tener presente las posibilidades que ofrece **R** si se utiliza plenamente su orientación matricial. Este código contrasta con el que se podría haber generado utilizando, digamos, un bucle y una serie de condicionales que asignasen cadenas numéricas a una matriz de cadenas de caracteres dependiendo en cada caso del código numérico a convertir.

```
> num2color <- function(numeros)
+ {
+ salida <- matrix(nrow=nrow(numeros), ncol=1)
+ for ( i in 1:nrow(numeros))
+    if (numeros[i]==1) {salida[i] <- "Red"}
+    else if (numeros[i]==2) {salida[i] <- "Yellow"}
+    else if (numeros[i]==3) {salida[i] <- "Gray"}
+    else if (numeros[i]==4) {salida[i] <- "Brown"}
+    else if (numeros[i]==5) {salida[i] <- "Pink"}
+    else if (numeros[i]==6) {salida[i] <- "Green"}
+    else if (numeros[i]==7) {salida[i] <- "White"}
+    else if (numeros[i]==8) {salida[i] <- "Orchid"}
+ return(salida)
+ }

> prim.colors.cad <- num2color(prim.colors)
> library(xgobi)
> xgobi(prim.all , colors=prim.colors.cad)
```

En las figuras siguientes se muestra la representación Sammon. Se ha separado por colores varias de las presuntas clases. La distribución espacial de estas clases es indiferente entre distintas clases. La información que de estas gráficas visualmente se desprende es que no existe un comportamiento factorial evidente, al menos atendiendo al carbono equivalente, puesto que en todos los gráficos existe un completo solapamiento de cada una de las clases.

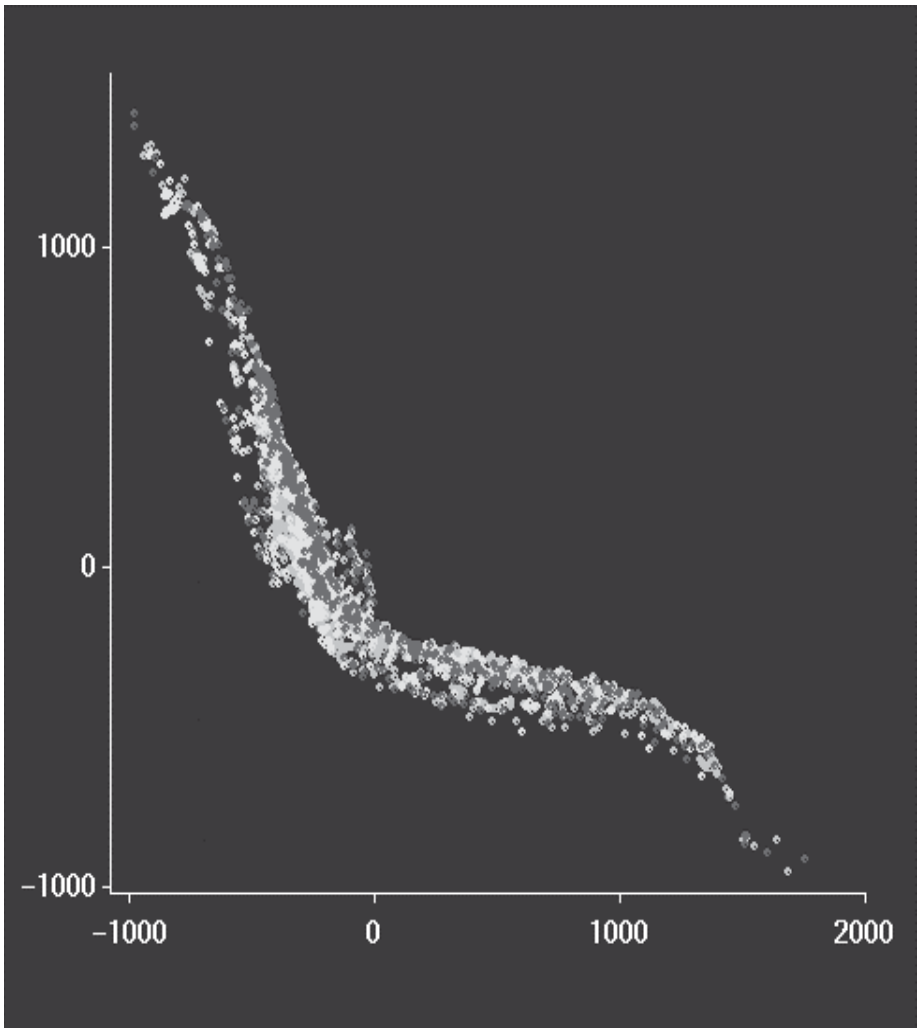


Fig. 6: ***Representación conjunta de los puntos correspondientes a todas las clases***

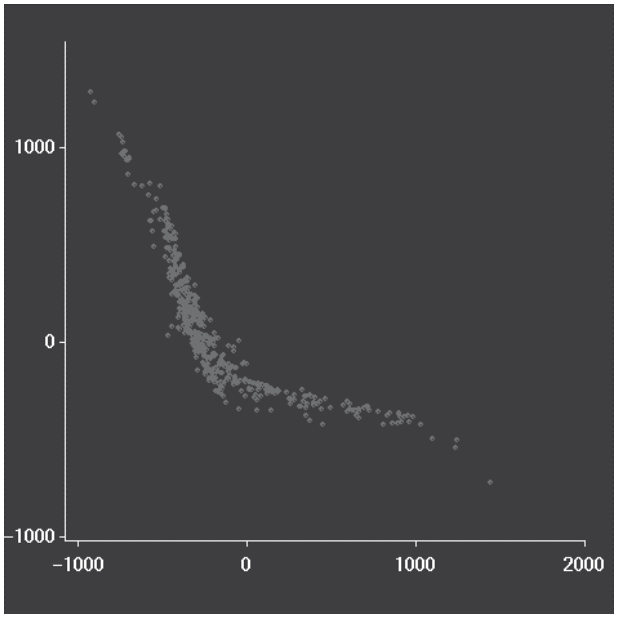


Fig. 7: ***Observaciones correspondientes a la clase (13,20]***

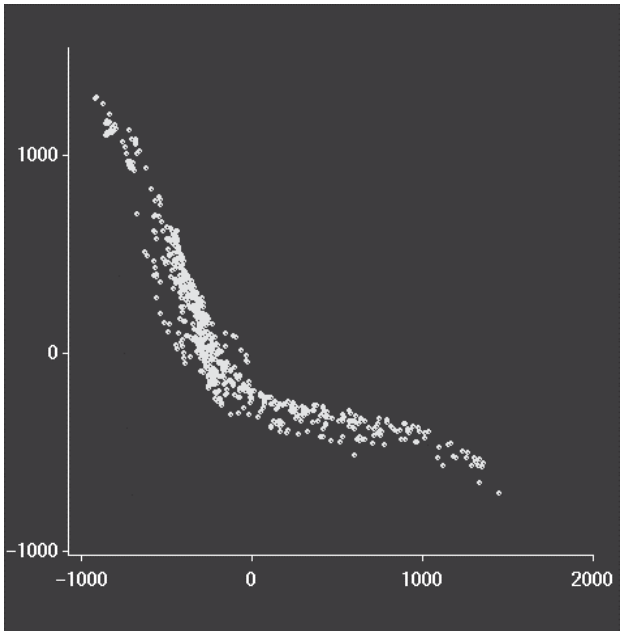


Fig. 8: ***Observaciones correspondientes a la clase (20,24]***

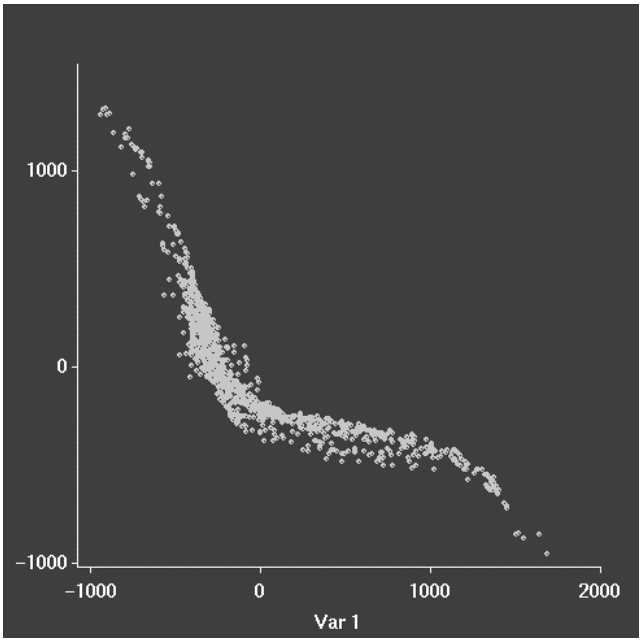


Fig. 9: ***Observaciones correspondientes a la clase (24,28]***

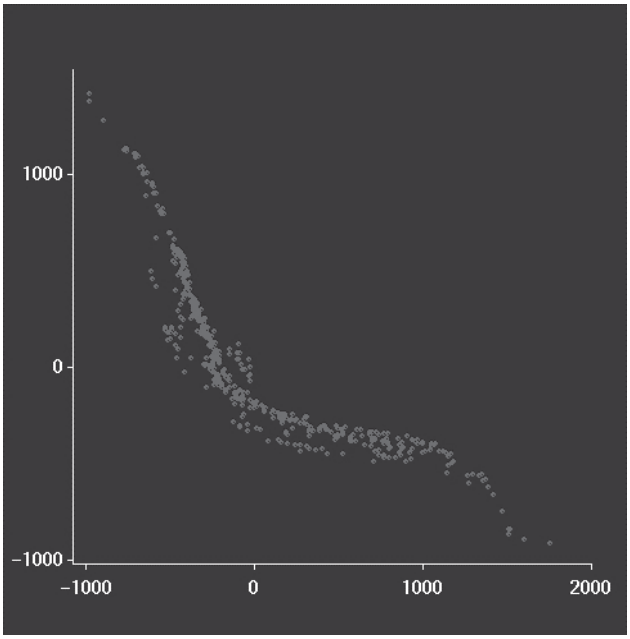


Fig. 10: ***Observaciones correspondientes a la clase (28,32]***

3.1.3. *Conclusiones*

La proyección Sammon pone en tela de juicio la existencia de una clasificación basada en el valor del contenido en carbono de la chapa. Es necesario pues aplicar otras técnicas que contrasten la validez de tal hipótesis. Asimismo se comprobará si los algoritmos de clasificación, que se aplicarán en este trabajo y que ordenan los patrones atendiendo a su similitud, proporcionan una clasificación de mayor calidad que la hasta ahora aplicada.

3.2. Análisis de Componentes Principales

3.2.1. *Fundamentos*

Este análisis proporciona las direcciones que deben adoptar los vectores de la base del espacio vectorial formado por las distintas variables muestreadas, a fin de estar orientados óptimamente, de acuerdo con las múltiples observaciones registradas. Esta orientación óptima debe ser entendida en el sentido de mínimos cuadrados de las distancias de los puntos a los ejes. El criterio que optimiza el ajuste de los ejes a los puntos es así mismo un criterio que optimiza la varianza de sus proyecciones sobre los ejes.

El análisis de componentes principales es frecuentemente utilizado como una técnica de reducción dimensional, puesto que permite estimar las orientaciones preferentes de los datos y despreciar aquellas dimensiones que no resulten especialmente relevantes en el estudio por tener mínima influencia en la variable a explicar.

En el planteamiento del problema se considera $x = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_N\}$ una población de N observaciones, con cada $\bar{x}_i \in \Re^n$. Se trata de buscar un sistema ortogonal de nuevas coordenadas, en cuyo sistema de referencia los puntos anteriores pueden expresarse como $w = \{\bar{w}_1, \bar{w}_2, \ldots, \bar{w}_n\}$, que denominaremos componentes principales. Sea $L = \{l_{ij}\}$ la matriz de cambio de base. Los coeficientes que nos relacionan la base antigua con la base nueva se pueden escribir como

$$\bar{w}_i = \sum_{j=1}^{n} l_{ij} \bar{x}_j \ , \ i = 1, \ldots, n$$

La condición de ortogonalidad (no-correlación) nos lleva a

$$E(\bar{w}_i \bar{w}_j) = E\left(\sum_{k=1}^{n} l_{ik} \bar{x}_k \sum_{m=1}^{n} l_{jm} \bar{x}_m \right) = 0 \ , \ (i \neq j)$$

o, lo que es lo mismo,

$$\sum_{k,m=1}^{n} l_{ik} l_{jm} E(\bar{x}_k \bar{x}_m) = \sum_{k,m=1}^{n} l_{ik} l_{jm} c_{km} = 0$$

donde por comodidad hemos substituido $E(\bar{x}_k \bar{x}_m)$ por c_{km}.

La nueva base debe ser ortogonal, lo que se consigue imponiendo

$$\sum_{i=1} l_{ij} l_{ik} = \delta_{ik}$$

siendo δ_{ik} la delta de Kronecker.

Con estas condiciones el problema esta suficientemente condicionado. Para concretar aun más su solución se puede escribir de forma matricial

$$\bar{w} = L\bar{x}$$

La condición de ortogonalidad implica

$$LL^T = I$$

y la de no correlación

$$E\left[\bar{w}\bar{w}^T\right] = E\left[L\bar{x}\left(L\bar{x}\right)^T\right] = E\left[L\bar{x}\bar{x}^T L^T\right] = LcL^T = \Lambda$$

Λ es la matriz de varianzas en la nueva base, y por tanto es diagonal. El problema se reduce entonces a diagonalizar la matriz C de covarianzas.

$$\left|C - \lambda I\right| = 0$$

Esta diagonalización nos proporciona las componentes principales $\bar{w}$ y sus correspondientes autovalores λ.

Se persigue escoger un subconjunto de r, $r < n$, componentes obtenidas de forma que se retenga la mayor parte de la variación respecto a la matriz de covarianzas. Para ello se pueden aplicar diversos criterios, como el basado en el porcentaje de perdida de variación, el de la media aritmética y el de la media geométrica:

Criterio del % de pérdida de variación:

La suma de las desviaciones al cuadrado de la matriz de covarianzas inicial C y la estimada basada en el subconjunto de r componentes viene dada por

$$tr\left(C^T C\right) - \sum_{j=1}^{r} \lambda_j = \sum_{j=r+1}^{n} \lambda_j$$

de ahí que la proporción de la suma total de desviaciones al cuadrado debida a las primeras r componentes, ip (información preservada), se expresa como

$$ip = 100 \cdot \frac{\sum_{j=1}^{r} \lambda_j}{\sum_{j=1}^{n} \lambda_j}$$

Para determinar el número r de componentes a preservar se debe establecer una referencia de corte, como por ejemplo plantearse preservar el 95% de variación original. Será el utilizado en este trabajo.

Criterio de la media aritmética:

Dado que la variación total viene dada por $\sum_{j=1}^{n} \lambda_j$, donde λ_j es la varianza de w_j, un posible criterio es retener aquellas componentes cuya varianza supere la media $\overline{\lambda} = \frac{\sum_{j=1}^{n} \lambda_j}{n}$. O lo que es lo mismo, retener aquellas $\overline{w}_j$ cuyas $\lambda_j > \overline{\lambda}$.

La matriz de correlación de una muestra multivariante es la matriz de varianzas y covarianzas que se obtiene al normalizar las variables respecto a su varianza, de forma que se obtengan variables de varianza unidad. Si se trabaja con las matrices de correlación, $\sum_{j=1}^{n} \lambda_j = n$ y por tanto $\overline{\lambda} = 1$.

Criterio de la media geométrica:

Como $\left|C^T C\right| = \prod_{j=1}^{n} \lambda_j$, se tiene que $\left|C^T C\right|^{\frac{1}{n}} = \left(\prod_{j=1}^{n} \lambda_j\right)^{\frac{1}{n}} = \overline{\lambda}_G$. La varianza media generalizada viene dada por la media geométrica de los autovalores, $\overline{\lambda}_G$, y este criterio selecciona como componentes a preservar aquellas que tengan $\lambda_j > \overline{\lambda}_G$.

3.2.2. *Implementación en **R**.*

Para poder ejecutar la función correspondiente al análisis de componentes principales, *pca,* es necesario cargar previamente la librería *multiv*:

```
> library(multiv)
```

Existe la posibilidad de aplicar la función *pca* al menos en tres formas distintas:

1. Utilizando directamente los datos de partida sin centrarlos en el origen ni alterar sus dimensiones, como se muestra en el ejemplo de la Fig. 11. Se trata de una distribución binormal de puntos descentrada respecto a la base $\{x, y\}$ y con una matriz de covarianzas distinta a la identidad. Los ejes $\overline{w}_1$ y $\overline{w}_2$ siguen la dirección de las componentes principales que le corresponden a la muestra al no haber centrado los datos. Nótese que las direcciones principales no siguen la dirección de $\overline{v}_1$ y $\overline{v}_2$ como correspondería en el caso de haber centrado la muestra.

2. Centrando las variables de forma que cada una de ellas presente media nula como en el ejemplo de la Fig. 12.

3. Normalizando las variables respecto a su varianza de forma que tengan no sólo media nula sino también desviación típica unidad (Fig. 13). Con ello se pierde la información contenida en la estructura espacial de los datos, y por tanto no se aplicará en este trabajo.

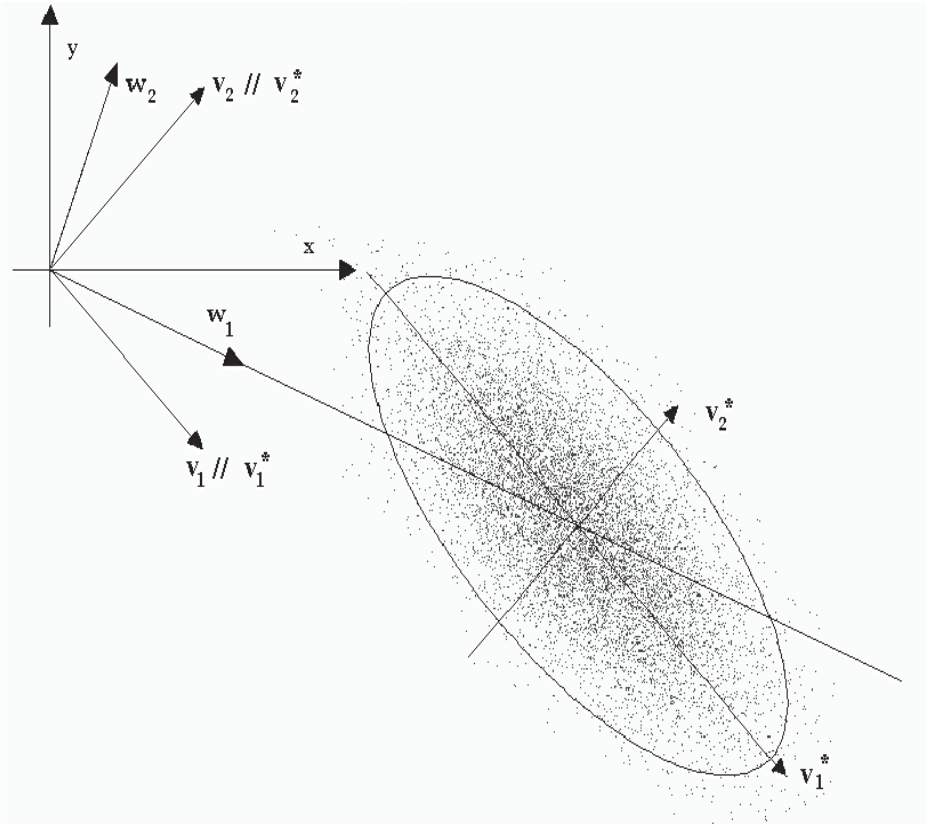


Fig. 11: P.C.A. obtenido con el método nº 1

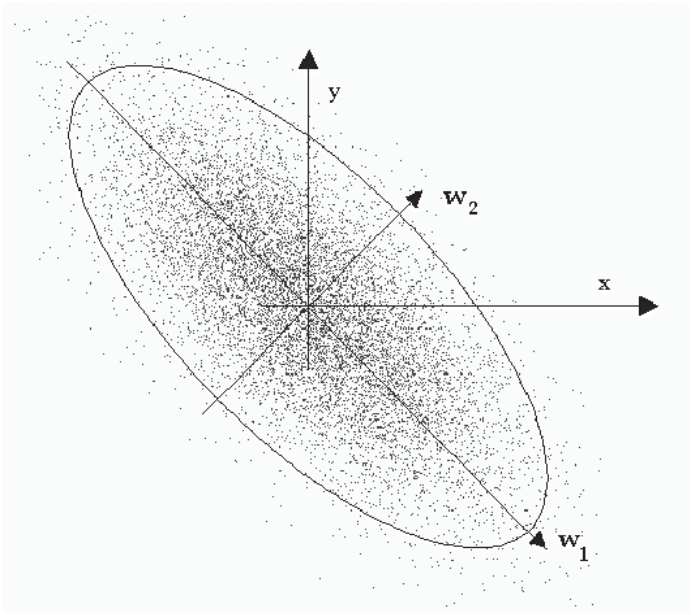


Fig. 12: P.C.A. obtenido con el método 2

En la Fig. 13 la distribución se ha convertido en esférica y ha perdido gran parte de la información que podría aportar su estructura original.

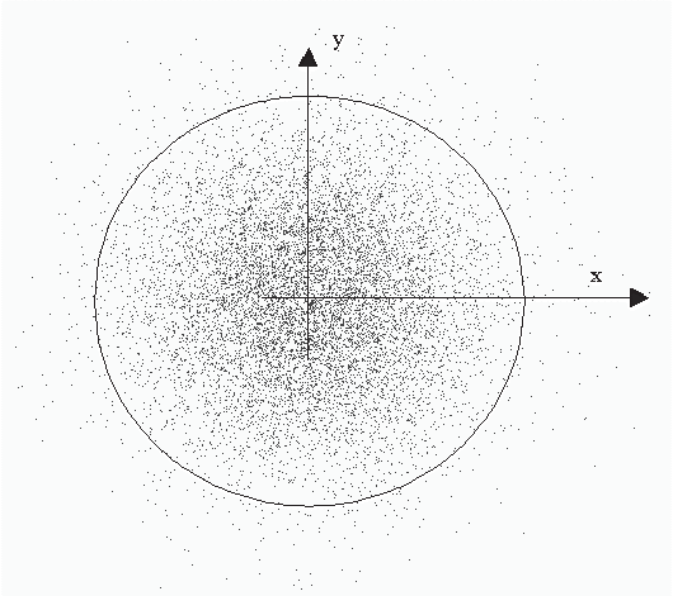


Fig. 13: P.C.A. obtenido con el método 3

El método preferido en este tipo de análisis es el mencionado en segundo lugar, en tanto en cuanto centra los ejes principales respecto a los datos y no deforma la estructura intrínseca de los datos como haría el tercer método.

```
> datos.pca ← pca(as.matrix(datos),2)
```

Se centran, pues, las variables de forma que tengan vector de medias nulo (ver pág. 33.). La función *pca* devuelve su resultado en una lista. Para observar la magnitud de los autovalores:

```
> datos.pca$evals
[1] 6490353213.7   69672752.8   24462726.4    3539318.6    1447904.4
[6] 131272.0
```

El valor porcentual relativo de cada uno de ellos respecto a los demás indica su importancia relativa:

```
> trazapca ← sum(datos.pca$evals)
> trazapca
[1] 6589607188
> importancia ← 0
> for (i in 1:length(datos.pca$evals))
+       importancia[i] ← datos.pca$evals[i]/ trazapca*100
> importancia
[1] 98.493780110  1.057312687  0.371231937  0.053710616  0.021972544
[6] 0.001992106
```

Según se ha comentado, el P.C.A. es frecuentemente utilizado como herramienta de reducción dimensional. A la vista de estos resultados, en función de cuantas componentes principales decidamos considerar significativas e incorporemos en nuestro modelo, habremos explicado en diferente cuantía la estructura de los datos (Tabla 2). Asimismo, cuantas más componentes se desprecien, mayor será la cantidad de información no considerada (Tabla 3).

Componentes Principales Involucradas	Porcentaje de variación Explicada
CP1	98.494
CP1 & CP2	99.551
CP1 & CP2 & CP3	99.922
CP1 & CP2 & CP3 & CP4	99.976
CP1 & CP2 & CP3 & CP4 & CP5	99.998
CP1 & CP2 & CP3 & CP4 & CP5 & CP6	100.000

Tabla 2

Componentes Principales Desestimadas	Porcentaje de variación despreciada
CP6	0.002
CP6 & CP5	0.024
CP6 & CP5 & CP4	0.088
CP6 & CP5 & CP4 & CP3	0.449
CP6 & CP5 & CP4 & CP3 & CP2	1.506
CP6 & CP5 & CP4 & CP3 & CP2 & CP1	100.000

Tabla 3

Parece razonable despreciar las últimas cuatro componentes principales. Con las restantes columnas de datos se procederá posteriormente a un análisis de dimensión fractal.

También podemos observar el valor de las direcciones principales.

```
> datos.pca$evecs
            Comp1        Comp2        Comp3         Comp4
TRRE  -0.0442672803 -0.832626652 -0.5487852006  5.479015e-03
F     -0.9969855514  0.001600388  0.0772694483 -5.191717e-04
DR     0.0042391165 -0.055323121 -0.0231745778 -2.110909e-02
PLT    0.0635761915 -0.551008647  0.8314296383  4.936682e-05
R     -0.0001373743  0.004064477 -0.0005803025  9.951984e-01
TRCEQ  0.0005450366  0.006526916 -0.0324283007 -9.541651e-02

            Comp5      Comp6
TRRE  -0.010727057  0.058847095
F      0.002884377 -0.006171889
DR    -0.021929297 -0.997726332
PLT    0.030902827  0.010830901
R      0.094962366 -0.023355285
TRCEQ  0.994697382 -0.019450349
```

Porcentualmente, la orientación de estas componentes principales sigue la distribución de la **Tabla 4**.

La componente principal nº 1, con el 98.493% de importancia relativa, sigue la dirección de V2 y de V4 en los porcentajes indicados. Esto quiere decir que la variación de los datos, depende, principalmente de las variables F y PLT (Fuerza aplicada a los rodillos de laminación y espesor de la chapa a laminar resp.).

Las componentes principales nº 2 y nº 3, con el 1.057% y 0.371% de importancia relativa siguen, mayormente, la dirección de V1 y V4 (Par aplicado a los rodillos de laminación y espesor de la chapa a laminar) en los porcentajes indicados.

La componente principal nº 4 y nº 5, con el 0.054% y 0.022% de importancia relativa, se orientan según V5 y V6 (radio de los rodillos de laminación y grado de carbono del acero)

La componente principal nº 6, con el 0.002% de importancia relativa, sigue la dirección de V3 y V1 (diferencia de espesores obtenidos en la chapa durante la pasada de laminación y par aplicado a los rodillos)

Referencia Física.	Comp. 1	Comp. 2	Comp. 3	Comp. 4	Comp. 5	Comp. 6
TRRE	3.989	**57.377**	**36.255**	0.490	0.928	**5.271**
F	**89.875**	0.110	5.105	0.046	0.249	0.553
DR	0.382	3.812	1.531	1.888	1.897	**89.371**
PLT	**5.729**	**37.970**	**54.928**	0.004	2.673	0.970
R	0.012	0.280	0.038	**89.034**	**8.214**	2.092
TRCEQ	0.049	0.450	2.142	**8.536**	**86.039**	1.742

Tabla 4

3.2.3. *Conclusiones*

En el análisis previo de los datos del proceso se considerarán únicamente las dos primeras componentes principales, a la vista de la relevancia de sus autovalores frente al resto. Despreciar el resto de componentes principales supone perder el 0.449% de información en total, porcentaje éste que consideramos asumible.

A la hora de modelizar el proceso, bien por técnicas clásicas, bien por técnicas neuronales, será necesario realizar un nuevo análisis de componentes principales donde no se tenga en cuenta la variable a explicar TRRE, puesto que si se pretende introducir las componentes principales como variables de entrada al modelo, no se podrá contar con el valor de TRRE a priori.

Es curioso comprobar que la variable TRCEQ, correspondiente al contenido en carbono de la chapa, carece de influencia significativa sobre el par aplicado durante su laminación.

Capítulo 4

Análisis exploratorio de las observaciones

Ahora que conocemos con mayor precisión la “estructura” de los datos soporte del modelo que deseamos “afinar” vamos a analizar en detalle la distribución de cada componente. **Resultará de especial importancia conocer, entre otros aspectos, si la población de observaciones sigue una distribución normal multivariante,** dada la importancia de esta distribución y de las herramientas estadísticas disponibles bajo condiciones de normalidad.

Los gráficos con los que básicamente se trabajará son los gráficos de cajas, o *box plots*, los histogramas, las estimaciones de la función de densidad de probabilidad y las rectas cuantil-cuantil. Una muestra previa de todas ellas como se aprecia en la Fig. 14, correspondientes a determinada variable, puede realizarse con una función similar a la propuesta en **[14].**

```
> eda.shape <- function(x) {
+ par(mfrow=c(2,2)); hist(x,col="red", main="Histograma")
+ boxplot(x,main="Gráfico de Cajas")
+ iqd <- summary(x)[5] - summary(x)[2]
+ plot(density(x, width=2*iqd), xlab="x",ylab="",
+ main="f.d.p. estimada",type="l")
+ qqnorm(x, pch=".",col="red",main="Gráfico Cuantil-Cuantil",
+ xlab="Cuantiles esperados",ylab="Cuantiles observados")
+ qqline(x) }
```

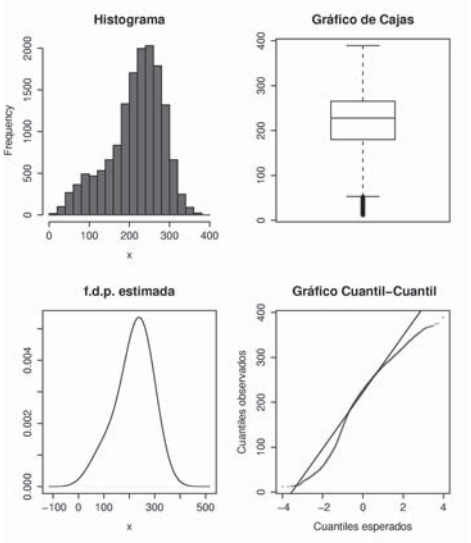


Fig. 14: Cuatro tipos básicos de gráficos utilizados en el análisis de datos.

4.1. Gráficos de distribución

4.1.1. *Gráficos de cajas*

Los gráficos de cajas, corrientemente denominados "*Box Plot*" resultan útiles a la hora de representar de forma compacta y resumida la distribución de una determinada muestra. La caja rectangular representa el rango intercuartílico Q (distancia entre el primer y tercer cuartil), con una línea de separación interna que representa la ubicación de la mediana. Si esta es próxima a los extremos de la caja la distribución presenta asimetría. La zona de estrechamiento de la caja en torno a la mediana indica el intervalo de confianza de su verdadero valor. La caja contiene el 50% de los datos centrales de la distribución. Las líneas que nacen en la caja se llaman *whiskers* o *bigotes* y se extienden en ambas direcciones hasta una distancia máxima de 1.5 veces la distancia intercuartílica, $1.5Q$. Si los valores máximo o mínimo se encuentran a menor distancia que la máxima de los *whiskers*, estos marcan su fin. Las observaciones que residan fuera del intervalo indicado por los *whiskers* son espureos potenciales que habrán de ser estudiados de forma separada a fin de contrastar si deben o no ser eliminados. Estas observaciones se representarán con un pequeño círculo si están dentro del rango $3Q$ y con un asterisco si están más allá de estos límites.

Para una distribución normal, la distancia intercuartílica viene dada por $Q = 1.35\sigma$. El suplemento de $1.5Q$ añadido en ambos extremos de la caja proporciona un rango de $4Q = 5.40\sigma$. Los umbrales de detección de los outliers son por tanto $\mu \pm 2.70\sigma$, lo que implica que este rango $4Q = 5.40\sigma$ represente un intervalo del 99.3% de confianza.

Cabe preguntarse si los puntos que en la representación Sammon pueden ser interpretados como espureos, son también identificados como tales mediante los gráficos de distribución. Esto puede verse en el siguiente ejemplo:

Se consideran dos variables (x,y) cuyos 100 primeros valores siguen una distribución normal y sus 10 últimos una $\chi^2_{df=3}$. *En la Fig. 15 se observa la distribución de ambas variables. Como la distribución es bidimensional no es necesario aplicar el algoritmo de Sammon para obtener su representación 2D (Fig. 16). En ella se observa como los puntos correspondientes verdes correspondientes a la* $\chi^2_{df=3}$ *se encuentran suficientemente alejados, en su mayoría, y que más de uno podría ser considerado como espureo visualmente. Sin embargo. En los gráficos de distribución (Fig. 17) no aparece más que uno de los puntos como posible espureo.*

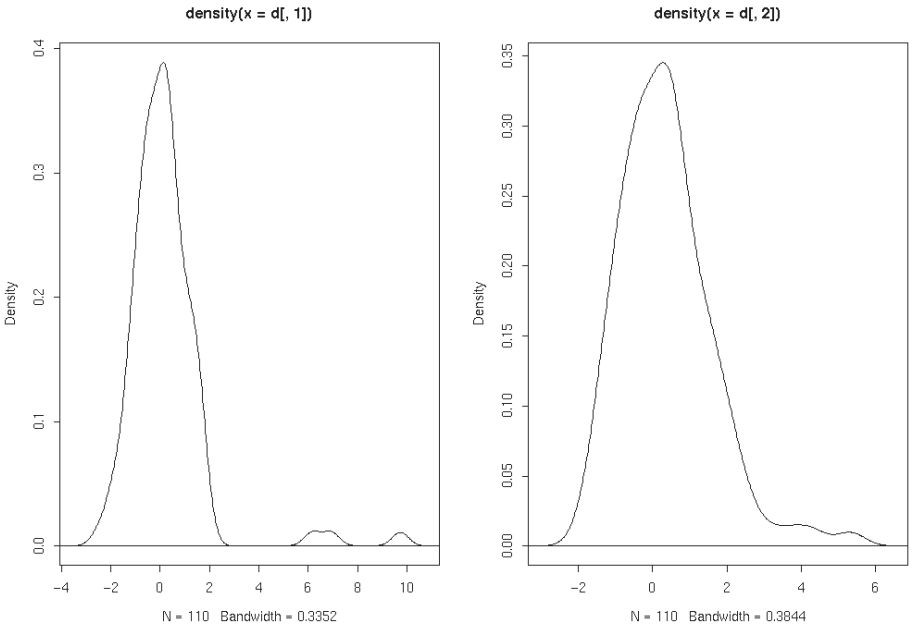


Fig. 15: Función de densidad de probabilidad de las variables *(x,y)* del ejemplo.

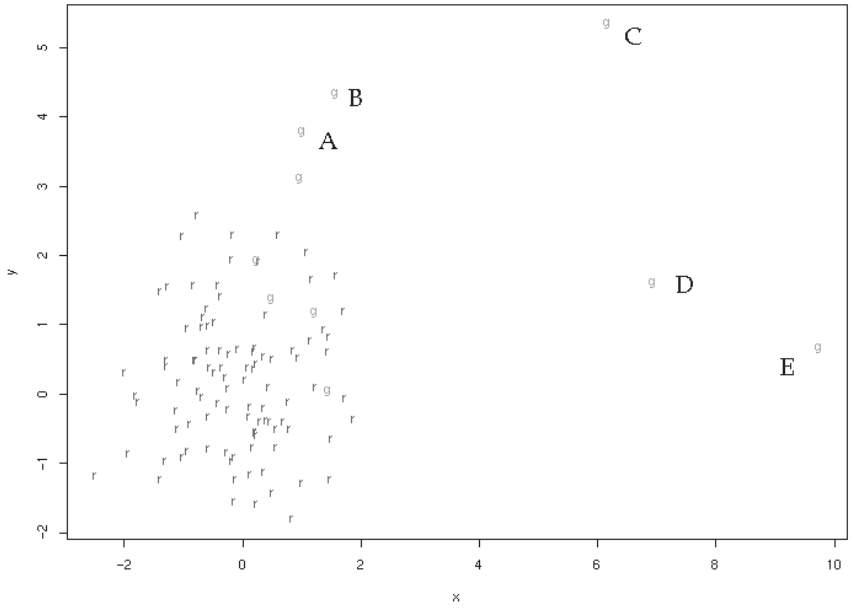


Fig. 16: Representación bidimensional de las variables *(x,y)* del ejemplo.

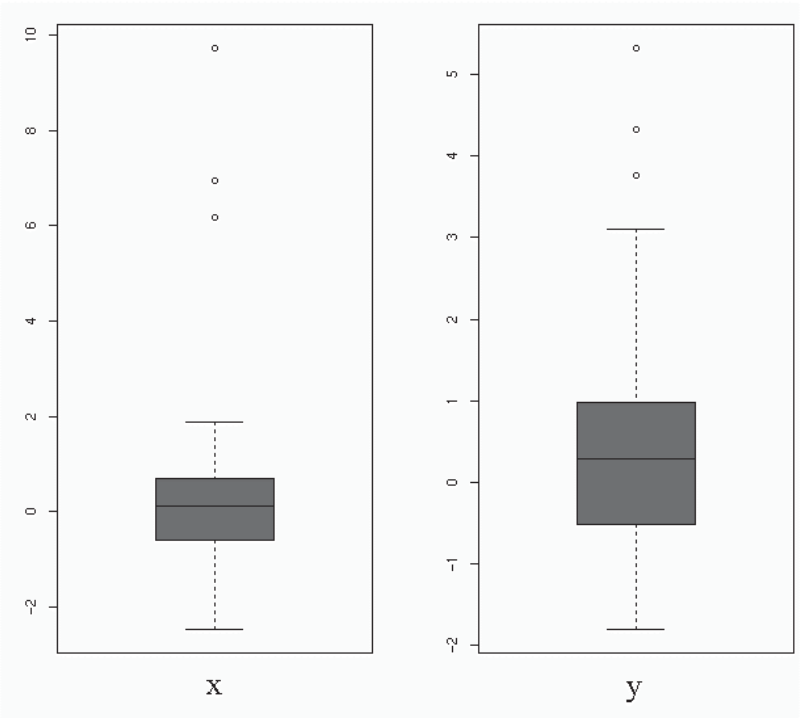


Fig. 17: Gráficos de cajas de las variables *(x,y)* del ejemplo.

Los puntos A, B y C quedan al descubierto con esta técnica como se ve en el diagrama "y" de la Fig. 17. El diagrama "x" detecta los puntos C, D y E. No obstante la representación Sammon permite detectar aquellas observaciones que no resultan atípicas en sus componentes individuales pero si en su conjunto. Un ejemplo de esto puede verse en la Fig. 18. En ella se señala la presencia de una observación que si bien se sale del margen de confianza del 95%, sin embargo no resulta atípico en sus componentes individuales y por tanto no es detectado como tal en los gráficos de cajas.

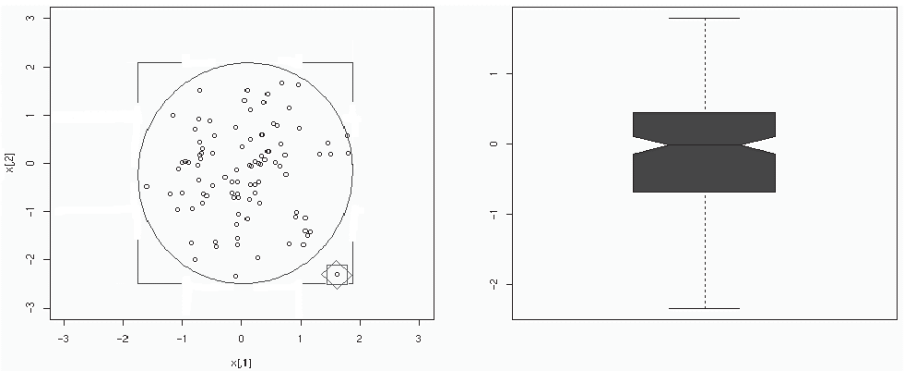


Fig. 18: Diferencia entre detección univariante y detección conjunta multivariante.

En primer lugar, si existe una división en clases basada únicamente en una única variable, como la potencialmente originada en este caso por el contenido en carbono, resulta conveniente representar para cada variable los “Box Plot” de cada variable restante contra cada clase de acero.

4.1.1.1. Implementación en **R**.

En *clase.datos* recogemos las diferentes clases correspondientes a cada observación de la matriz *datos.nuevos*, que recordemos, era el resultado de suprimir los puntos más aislados de la muestra.

```
> summary(datos.nuevos)
TRRE            F               DR              PLT
Min.   : 12     Min.   : 281    Min.   : 0.03   Min.   :  5.47
1st Qu.:180     1st Qu.:1764    1st Qu.: 9.60   1st Qu.: 41.40
Median :228     Median :2091    Median :14.03   Median : 91.97
Mean   :216     Mean   :2175    Mean   :12.75   Mean   : 99.12
3rd Qu.:265     3rd Qu.:2590    3rd Qu.:16.92   3rd Qu.:145.30
Max.   :389     Max.   :4158    Max.   :24.94   Max.   :276.60

R                TRCEQ
Min.   :425.6   Min.   :14.00
1st Qu.:441.9   1st Qu.:26.00
Median :462.9   Median :33.00
Mean   :457.3   Mean   :34.45
3rd Qu.:472.4   3rd Qu.:42.00
Max.   :475.6   Max.   :62.00

> lapply(split(datos.nuevos,cortes),summary)
$"(13,20]"
      TRRE             F              DR             PLT
 Min.   : 18.0   Min.   : 552   Min.   : 2.10   Min.   : 14.14
 1st Qu.:187.0   1st Qu.:1677   1st Qu.:13.42   1st Qu.: 85.44
 Median :220.0   Median :1926   Median :15.57   Median :147.80
 Mean   :214.7   Mean   :1966   Mean   :15.14   Mean   :141.70
 3rd Qu.:248.5   3rd Qu.:2182   3rd Qu.:17.65   3rd Qu.:200.40
 Max.   :336.0   Max.   :3762   Max.   :24.73   Max.   :268.70
       R            TRCEQ
 Min.   :437.3   Min.   :14.00
 1st Qu.:440.3   1st Qu.:14.00
 Median :458.1   Median :20.00
 Mean   :454.9   Mean   :17.85
 3rd Qu.:463.3   3rd Qu.:20.00
 Max.   :475.6   Max.   :20.00

$"(20,24]"
      TRRE             F              DR             PLT
 Min.   : 13.0   Min.   : 496   Min.   : 0.25   Min.   :  6.15
 1st Qu.:182.0   1st Qu.:1736   1st Qu.:10.60   1st Qu.: 46.77
```

```
 Median :223.0   Median :2058   Median :14.53   Median :101.90
 Mean   :213.4   Mean   :2143   Mean   :13.36   Mean   :107.60
 3rd Qu.:259.0   3rd Qu.:2505   3rd Qu.:17.13   3rd Qu.:158.30
 Max.   :354.0   Max.   :4141   Max.   :24.94   Max.   :265.40
       R             TRCEQ
 Min.   :425.6   Min.   :21.00
 1st Qu.:441.9   1st Qu.:22.00
 Median :466.0   Median :23.00
 Mean   :459.1   Mean   :22.92
 3rd Qu.:472.5   3rd Qu.:24.00
 Max.   :474.7   Max.   :24.00

$"(24,28]"
      TRRE             F              DR             PLT
 Min.   : 18.0   Min.   : 391   Min.   : 0.35   Min.   :  5.47
 1st Qu.:178.0   1st Qu.:1752   1st Qu.: 8.98   1st Qu.: 36.25
 Median :227.0   Median :2098   Median :14.10   Median : 88.46
 Mean   :217.1   Mean   :2211   Mean   :12.67   Mean   : 96.06
 3rd Qu.:267.0   3rd Qu.:2678   3rd Qu.:17.04   3rd Qu.:143.00
 Max.   :375.0   Max.   :4139   Max.   :23.71   Max.   :268.70
       R             TRCEQ
 Min.   :425.6   Min.   :25.00
 1st Qu.:442.8   1st Qu.:25.00
 Median :463.2   Median :26.00
 Mean   :458.5   Mean   :26.26
 3rd Qu.:473.1   3rd Qu.:27.00
 Max.   :475.6   Max.   :28.00

$"(28,32]"
      TRRE             F              DR             PLT
 Min.   : 12.0   Min.   : 383   Min.   : 0.340   Min.   :  6.41
 1st Qu.:187.0   1st Qu.:1801   1st Qu.: 9.625   1st Qu.: 38.60
 Median :229.0   Median :2147   Median :13.940   Median : 88.57
 Mean   :217.2   Mean   :2234   Mean   :12.610   Mean   : 95.40
 3rd Qu.:261.0   3rd Qu.:2680   3rd Qu.:16.430   3rd Qu.:141.50
 Max.   :352.0   Max.   :4158   Max.   :23.750   Max.   :266.50
```

```
       R             TRCEQ
 Min.   :428.0   Min.   :29.00
 1st Qu.:441.9   1st Qu.:29.00
 Median :463.2   Median :30.00
 Mean   :457.9   Mean   :30.23
 3rd Qu.:470.4   3rd Qu.:31.00
 Max.   :474.7   Max.   :32.00

$"(32,36]"
      TRRE             F              DR              PLT
 Min.   : 18.0   Min.   : 315   Min.   : 0.380   Min.   :  6.28
 1st Qu.:175.0   1st Qu.:1785   1st Qu.: 8.285   1st Qu.: 37.93
 Median :228.0   Median :2096   Median :13.540   Median : 88.19
 Mean   :213.9   Mean   :2197   Mean   :12.240   Mean   : 98.18
 3rd Qu.:266.0   3rd Qu.:2627   3rd Qu.:16.550   3rd Qu.:148.30
 Max.   :347.0   Max.   :4154   Max.   :22.570   Max.   :272.60
       R             TRCEQ
 Min.   :428.0   Min.   :33
 1st Qu.:441.9   1st Qu.:33
 Median :462.9   Median :34
 Mean   :457.0   Mean   :34
 3rd Qu.:470.4   3rd Qu.:35
 Max.   :474.7   Max.   :35

$"(36,40]"
      TRRE             F              DR              PLT
 Min.   : 13.0   Min.   : 367   Min.   : 0.320   Min.   :  5.53
 1st Qu.:183.0   1st Qu.:1838   1st Qu.: 8.145   1st Qu.: 39.80
 Median :234.0   Median :2177   Median :13.420   Median : 89.81
 Mean   :219.8   Mean   :2247   Mean   :12.120   Mean   : 98.35
 3rd Qu.:270.0   3rd Qu.:2695   3rd Qu.:16.200   3rd Qu.:145.00
 Max.   :389.0   Max.   :4119   Max.   :23.010   Max.   :276.60
       R             TRCEQ
 Min.   :425.6   Min.   :37.00
 1st Qu.:441.9   1st Qu.:38.00
 Median :463.2   Median :39.00
 Mean   :458.4   Mean   :38.71
 3rd Qu.:472.4   3rd Qu.:40.00
 Max.   :474.7   Max.   :40.00
```

```
$"(40,44]"
      TRRE             F              DR              PLT
 Min.   : 12.0   Min.   : 281   Min.   : 0.030   Min.   :  6.44
 1st Qu.:175.0   1st Qu.:1776   1st Qu.: 8.695   1st Qu.: 39.81
 Median :226.0   Median :2102   Median :13.360   Median : 88.44
 Mean   :215.2   Mean   :2169   Mean   :12.320   Mean   : 93.06
 3rd Qu.:265.3   3rd Qu.:2579   3rd Qu.:16.440   3rd Qu.:139.10
 Max.   :370.0   Max.   :4081   Max.   :24.180   Max.   :276.60
       R            TRCEQ
 Min.   :425.6   Min.   :42.00
 1st Qu.:441.9   1st Qu.:42.00
 Median :462.9   Median :43.00
 Mean   :456.5   Mean   :42.99
 3rd Qu.:472.4   3rd Qu.:44.00
 Max.   :475.6   Max.   :44.00

$"(44,48]"
      TRRE             F              DR              PLT
 Min.   : 33.0   Min.   : 629   Min.   : 0.36   Min.   :  6.26
 1st Qu.:185.0   1st Qu.:1723   1st Qu.:10.29   1st Qu.: 47.68
 Median :237.0   Median :2005   Median :14.83   Median : 89.06
 Mean   :227.4   Mean   :2067   Mean   :13.52   Mean   : 87.43
 3rd Qu.:280.0   3rd Qu.:2413   3rd Qu.:17.44   3rd Qu.:129.70
 Max.   :365.0   Max.   :3723   Max.   :22.67   Max.   :165.90
       R            TRCEQ
 Min.   :428.0   Min.   :45
 1st Qu.:439.5   1st Qu.:45
 Median :460.0   Median :45
 Mean   :455.8   Mean   :45
 3rd Qu.:472.6   3rd Qu.:45
 Max.   :474.7   Max.   :45

$"(48,63]"
      TRRE             F              DR              PLT
 Min.   : 42.0   Min.   : 570   Min.   : 0.91   Min.   :  8.71
 1st Qu.:168.5   1st Qu.:1624   1st Qu.:10.84   1st Qu.: 47.30
 Median :220.0   Median :1978   Median :15.59   Median : 98.49
 Mean   :208.4   Mean   :1985   Mean   :13.98   Mean   :102.80
 3rd Qu.:258.0   3rd Qu.:2282   3rd Qu.:18.45   3rd Qu.:144.50
 Max.   :361.0   Max.   :3880   Max.   :23.08   Max.   :269.10
```

```
       R              TRCEQ
 Min.   :425.6   Min.   :51.00
 1st Qu.:438.4   1st Qu.:59.00
 Median :442.5   Median :59.00
 Mean   :451.7   Mean   :57.82
 3rd Qu.:470.4   3rd Qu.:59.00
 Max.   :474.5   Max.   :62.00
```

Esta información aparece ordenada por variables en las siguientes tablas.

TRRE									
Clase	**(0,20]**	**(20,24]**	**(24,28]**	**(28,32]**	**(32,36]**	**(36,40]**	**(40,44]**	**(44,48]**	**(52,…]**
Min.	18.0	13.0	18.0	12.0	18.0	13.0	12.0	33.0	42.0
1Q	187.0	182.0	178.0	187.0	175.0	183.0	175.0	185.0	168.5
Mediana	220.0	223.0	227.0	229.0	228.0	234.0	226.0	237.0	220.0
Media	214.7	213.4	217.1	217.2	213.9	219.8	215.2	227.4	208.4
3Q	248.5	259.0	267.0	261.0	266.0	270.0	265.3	280.0	258.0
Max	336.0	354.0	375.0	352.0	347.0	389.0	370.0	365.0	361.0

F									
Clase	**(0,20]**	**(20,24]**	**(24,28]**	**(28,32]**	**(32,36]**	**(36,40]**	**(40,44]**	**(44,48]**	**(52,…]**
Min.	552	496	391	383	315	367	281	629	570
1Q	1677	1736	1752	1801	1785	1838	1776	1723	1624
Mediana	1926	2058	2098	2147	2096	2177	2102	2005	1978
Media	1966	2143	2211	2234	2197	2247	2169	2067	1985
3Q	2182	2505	2678	2680	2627	2695	2579	2413	2282
Max	3762	4141	4139	4158	4154	4119	4081	3723	3880

DR									
Clase	**(0,20]**	**(20,24]**	**(24,28]**	**(28,32]**	**(32,36]**	**(36,40]**	**(40,44]**	**(44,48]**	**(52,…]**
Min.	2.10	0.25	0.35	0.34	0.38	0.32	0.03	0.36	0.91
1Q	13.42	10.60	8.98	9.62	8.28	8.14	8.69	10.29	10.84
Mediana	15.57	14.53	14.10	13.94	13.54	13.42	13.36	14.83	15.59
Media	15.14	13.36	12.67	12.61	12.24	12.12	12.32	13.52	13.98
3Q	17.65	17.13	17.04	16.43	16.55	16.20	16.44	17.44	18.45
Max	24.73	24.94	23.71	23.75	22.57	23.01	24.18	22.67	23.08

PLT									
Clase	**(0,20]**	**(20,24]**	**(24,28]**	**(28,32]**	**(32,36]**	**(36,40]**	**(40,44]**	**(44,48]**	**(52,…]**
Min.	14.1	6.15	5.47	6.41	6.28	5.53	6.44	6.26	8.71
1Q	85.4	46.77	36.25	38.60	37.93	39.80	39.81	47.68	47.30
Mediana	147.8	101.90	88.46	88.57	88.19	89.81	88.44	89.06	98.49
Media	141.7	107.60	96.06	95.40	98.18	98.35	93.06	87.43	102.80
3Q	200.4	158.30	143.00	141.50	148.30	145.00	139.10	129.70	144.50
Max	268.7	265.40	268.70	266.50	272.60	276.60	276.60	165.90	269.10

R									
Clase	**(0,20]**	**(20,24]**	**(24,28]**	**(28,32]**	**(32,36]**	**(36,40]**	**(40,44]**	**(44,48]**	**(52,…]**
Min.	437.3	425.6	425.6	428.0	428.0	425.6	425.6	428.0	425.6
1Q	440.3	441.9	442.8	441.9	441.9	441.9	441.9	439.5	438.4
Mediana	458.1	466.0	463.2	463.2	462.9	463.2	462.9	460.0	442.5
Media	454.9	459.1	458.5	457.9	457.0	458.4	456.5	455.8	451.7
3Q	463.3	472.5	473.1	470.4	470.4	472.4	472.4	472.6	470.4
Max	475.6	474.7	475.6	474.7	474.7	474.7	475.6	474.7	474.5

TRCEQ									
Clase	**(0,20]**	**(20,24]**	**(24,28]**	**(28,32]**	**(32,36]**	**(36,40]**	**(40,44]**	**(44,48]**	**(52,…]**
Min.	14.00	21.00	25.00	29.00	33	37.00	42.00	45	51.00
1Q	14.00	22.00	25.00	29.00	33	38.00	42.00	45	59.00
Mediana	20.00	23.00	26.00	30.00	34	39.00	43.00	45	59.00
Media	17.85	22.92	26.26	30.23	34	38.71	42.99	45	57.82
3Q	20.00	24.00	27.00	31.00	35	40.00	44.00	45	59.00
Max	20.00	24.00	28.00	32.00	35	40.00	44.00	45	62.00

Parece claro que la interpretación de estos parámetros resulta más clara a través de gráficos como los que se muestran en la en la Fig. .

```
> attach(datos.nuevos)
> clase ← cut(TRCEQ, c(13,20,24,28,32,36,40,44,48,63))
> par(mfrow=c(3,2), cex=1)
> boxplot(split(TRRE, clase),col="red", notch=TRUE, xlab="TRRE")
> boxplot(split(F,    clase),col="red", notch=TRUE, xlab="F")
> boxplot(split(DR,   clase),col="red", notch=TRUE, xlab="DR")
> boxplot(split(PLT,  clase),col="red", notch=TRUE, xlab="PLT")
> boxplot(split(R,    clase),col="red", notch=TRUE, xlab="R")
> detach(datos.nuevos)
```

Cada clase recoge el número de observaciones indicado en la Tabla 5

Clase nº	1	2	3	4	5	6	7	8	9
Contenido en carbono	(13,20]	(20,24]	(24,28]	(28,32]	(32,36]	(36,40]	(40,44]	(44,48]	(48,63]
Nº observ.	455	1785	2649	2171	1738	2328	2244	473	1099

Tabla 5

El origen de la mayor parte de los de potenciales *outliers* univariantes recae en las variables 1 y 2. Las clases 1, 8 y 9 presentan menor número de elementos que el resto. Ello repercute en una menor significación de los resultados de los test aplicados sobre ellos, frente a la significación del resto.

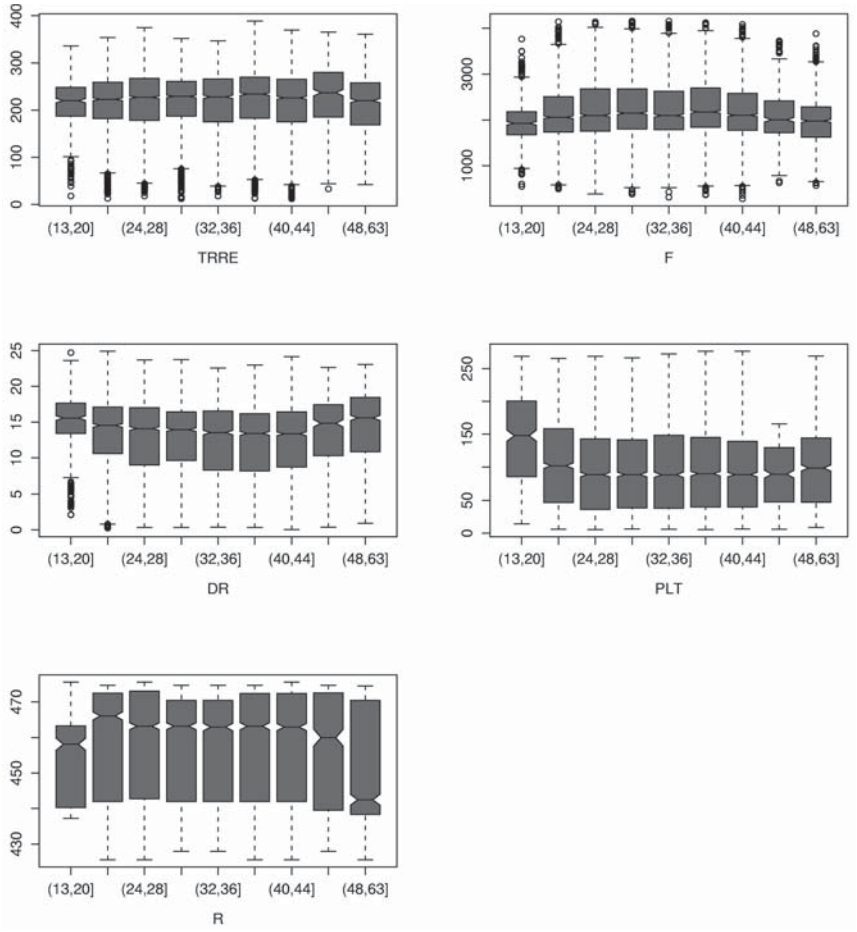


Fig. 19: Gráficos de distribución de cada variable en las distintas clases.

En la Fig. 19 se observa que las variables presentan un comportamiento uniforme a lo largo de las presuntas distintas clases, lo que apunta hacia el rechazo de su posible existencia. Esto se reafirma en mayor medida en la Fig.

20, donde se ha representado los Box Plot de las dos componentes principales más significativas.

```
> cortes  <- c(13,20,24,28,32,36,40,44,48,63)
> clase ← cut(datos.nuevos$TRCEQ, cortes)
> par(mfrow=c(3,2), cex=1)
> boxplot(split(datos.nuevos.pca$rproj[,1], clase), col="red",
+ notch=TRUE, xlab="CP1")
> boxplot(split(datos.nuevos.pca$rproj[,2], clase), col="red",
+ notch=TRUE, xlab="CP2")
```

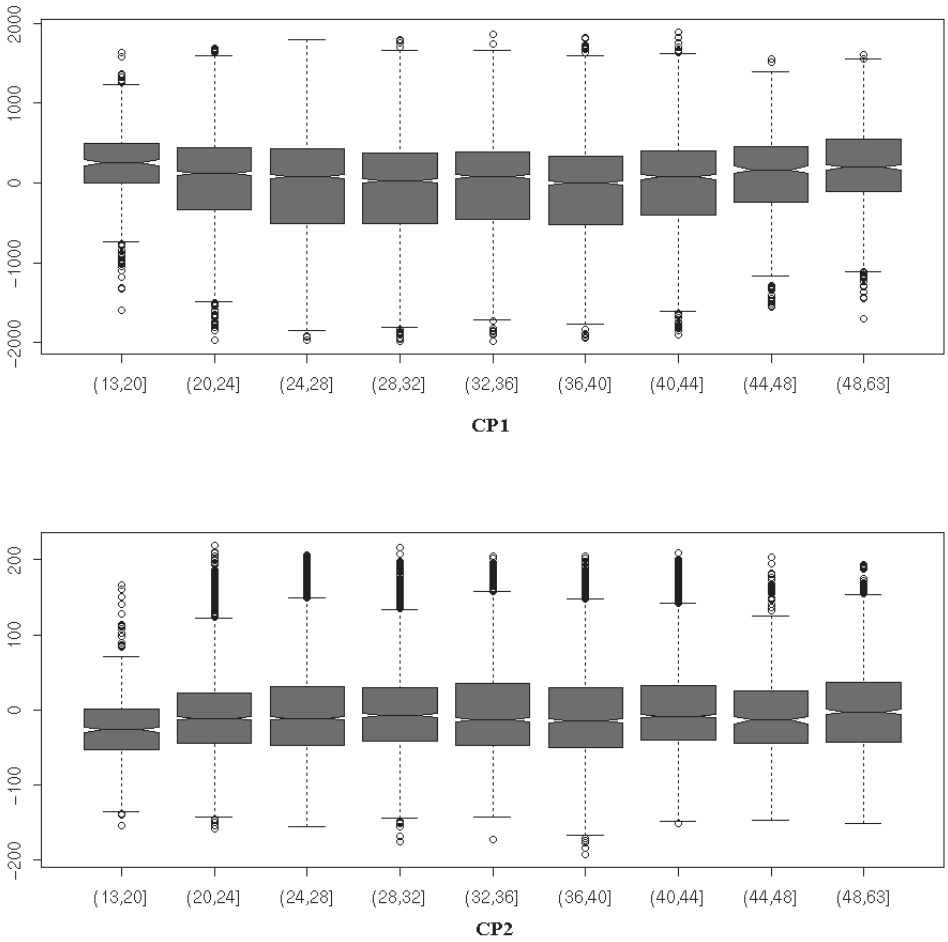


Fig. 20: Las 2 Comp. Princ. más significativas en las distintas clases.

4.1.2. *Gráficas de frecuencia totales*

4.1.2.1. Implementación en **R**.

En las gráficas de frecuencias totales se representa la distribución que sigue cada variable de la muestra completa considerando de forma conjunta todas las clases (Fig. 21).

```
> par(mfrow=c(3,2),col="red")
> hist(TRRE, main = "")
> hist(F, main = "")
> hist(DR, main = "")
> hist(PLT,main = "")
> hist(R,main = "")
```

Una representación preferible a los histogramas es la proporcionada por la función *density*, cuya representación ya no depende del número de intervalos seleccionados (Fig. 22).

```
> par(mfrow=c(3,2), cex=1)
> plot(density(TRRE), main="TRRE")
> plot(density(F),main="F")
> plot(density(DR),main="DR")
> plot(density(PLT),main="PLT")
> plot(density(R),main="R")
```

El parámetro *bandwidth* de los gráficos de densidad afecta a su forma y suavidad. Se ha optado por escoger el valor propuesto por la función para nuestros datos.

A la vista de los gráficos de las Fig. 21 y Fig. 22, las variables no presentan un comportamiento normal univariante. Es interesante representar las distribuciones correspondientes a las componentes principales (Fig. 23).

```
>par(mfrow=c(3,2), cex=1)
>for (i in 1:5)
+  plot(density(datos.nuevos.pca$rproj[,1]), main=paste("CP",i)
```

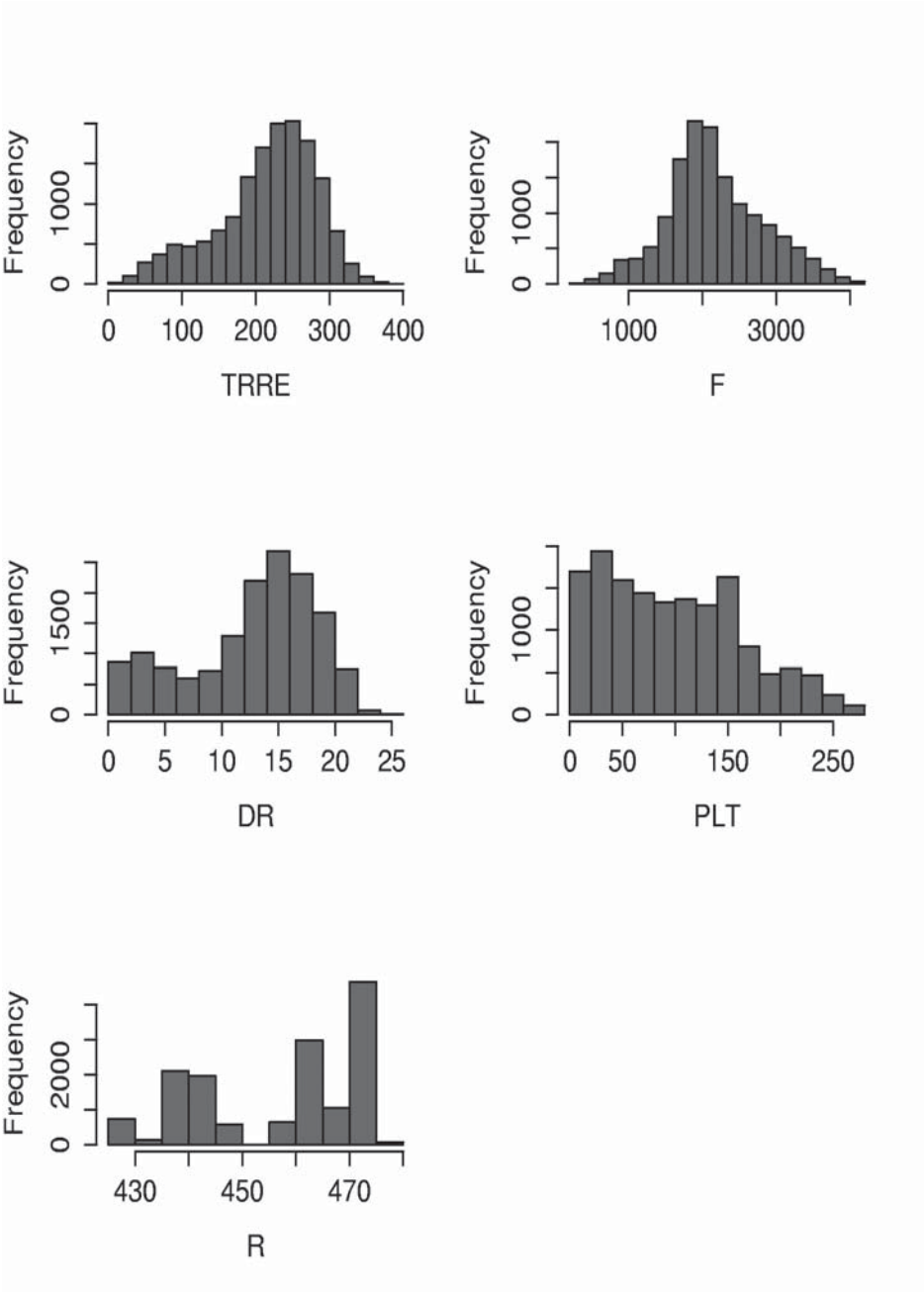


Fig. 21: Histogramas totales de cada variable.

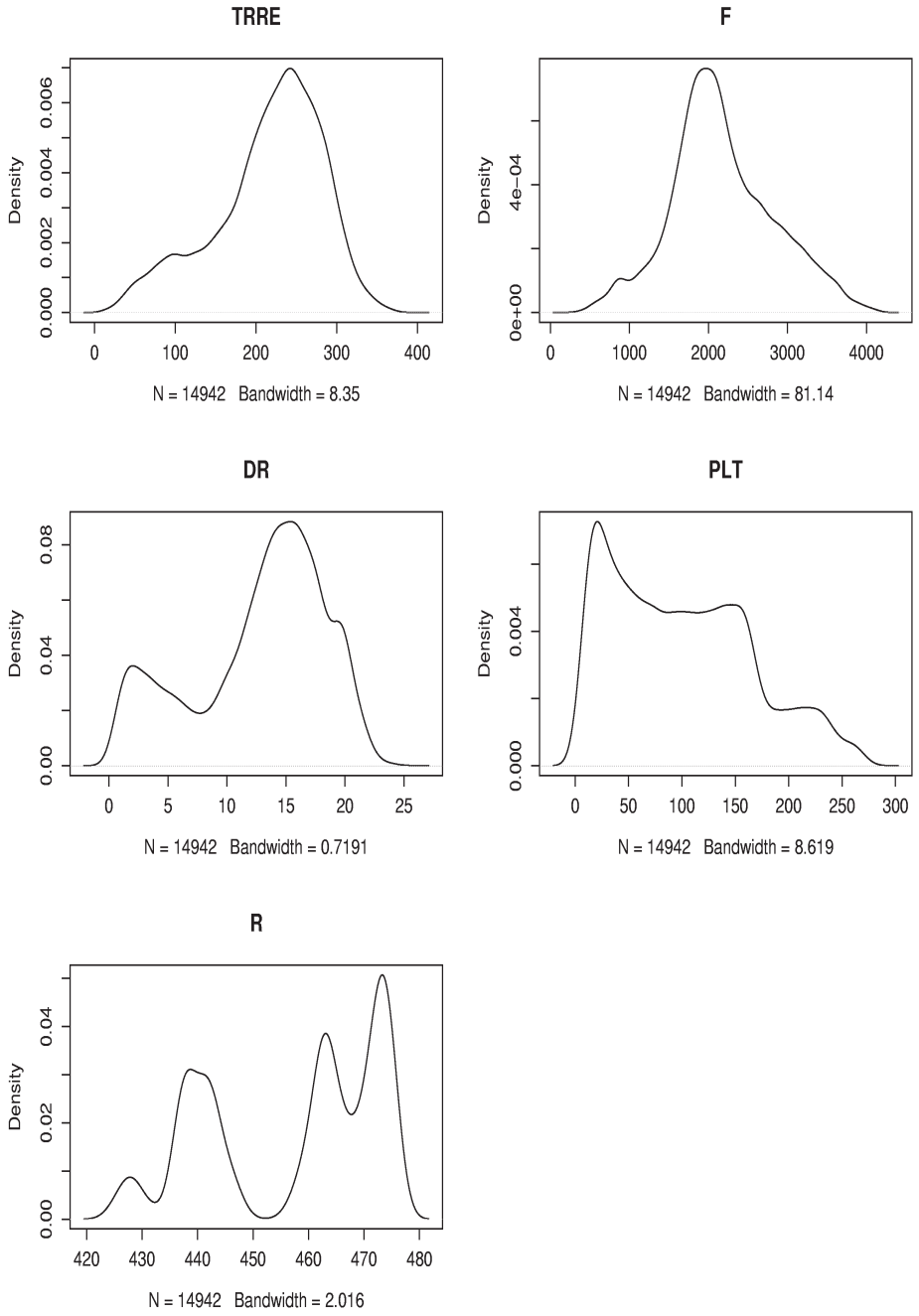


Fig. 22: Densidad de distribución total de cada variable.

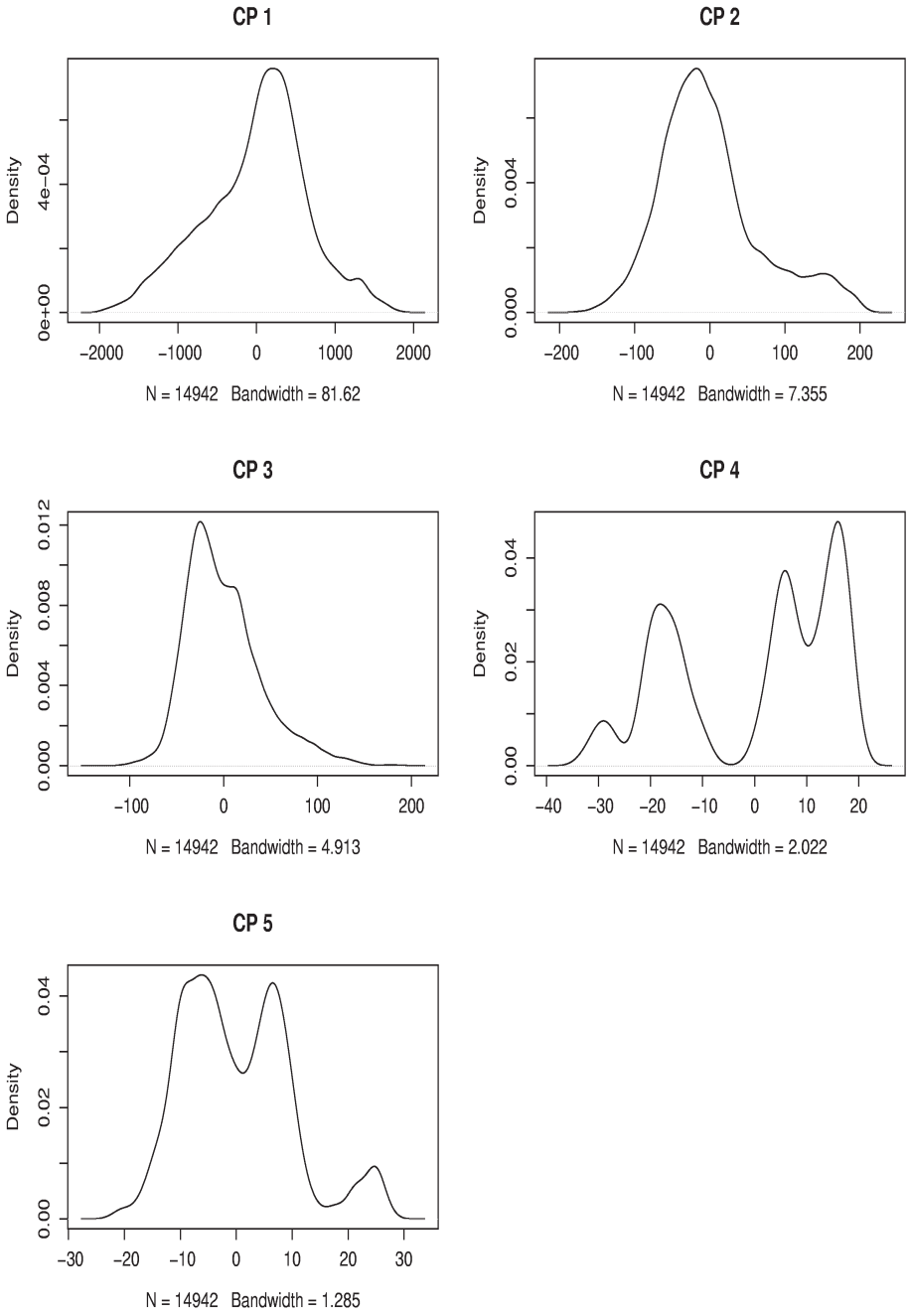


Fig. 23: Densidad de distribución total de las componentes principales.

La no normalidad de las componentes principales asegura la no normalidad multivariante. Concretamente, las dos primeras componentes principales, de mayor interés en este estudio, no presentan perfil normal. De haberlo presentado se hubiera aceptado la normalidad multivariante, por ser estas componentes las de mayor peso. Por ello no se podrán aplicar a los datos de este proceso las técnicas estadísticas que asuman normalidad de la población multivariada.

4.1.3. *Gráficas de frecuencia por clase*

Permite contrastar si el comportamiento de las variables dentro de las distintas clases resulta acorde con el patrón general, o si por el contrario, el comportamiento individual dentro de cada una de las clases es claramente distinto.

4.1.3.1. Implementación en **R**.

La función *split* devuelve una lista cuyas componentes hacen referencia a las distintas clases. La lista contiene tantas componentes como clases haya y cada una integra los elementos de la matriz de entrada de su clase particular.

```
> clase.datos ← factor(datos.nuevos[,6])
> v1 ← split(datos.nuevos[,1], clase.datos)
> v2 ← split(datos.nuevos[,2], clase.datos)
> v3 ← split(datos.nuevos[,3], clase.datos)
> v4 ← split(datos.nuevos[,4], clase.datos)
> v5 ← split(datos.nuevos[,5], clase.datos)
> par(mfrow=c(5,2), cex=1)
> for( i in 1:9)
+   plot(density(v1[[i]]))
```

En los gráficos de la Fig. 24 se muestra el comportamiento de la variable nº 1 a lo largo de las distintas clases. Se ha obviado la inclusión de los gráficos correspondientes al resto de variables pues todas ellas presentan un comportamiento que apunta hacia la misma conclusión y se opta por representar una de ellas, la primera, a modo de ejemplo.

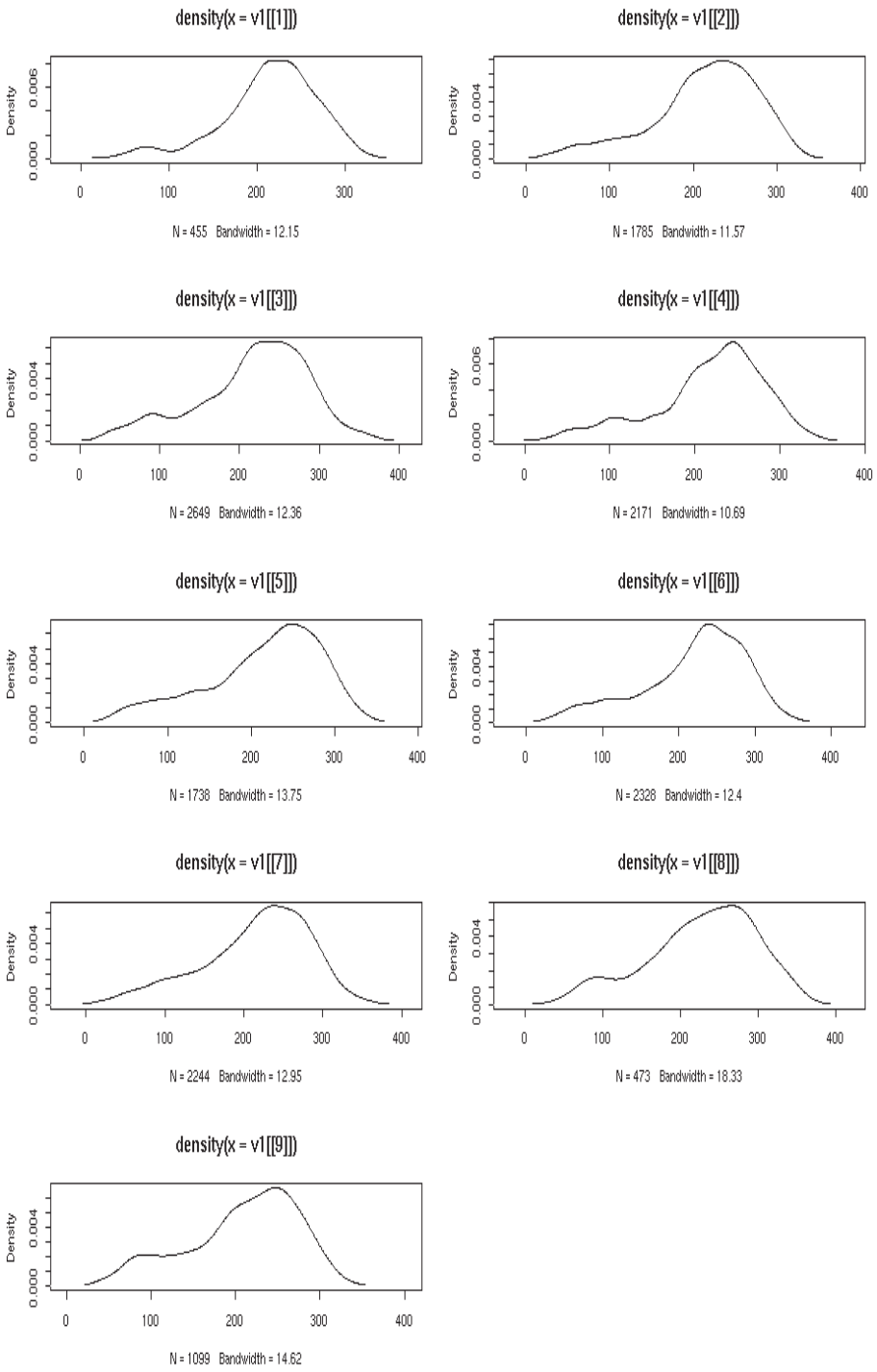


Fig. 24: Densidad de distribución de la variable nº 1 en las diferentes clases.

Es interesante observar asimismo, el aspecto de estos gráficos para las componentes principales de mayor importancia.

```
> clase.datos <- factor(datos.nuevos[,6])
> cp1 <-  split(datos.nuevos[,1], clase.datos)
> cp2 <-  split(datos.nuevos[,2], clase.datos)
> par(mfrow=c(5,2), cex=1)
> for( i in 1:9)
+  plot(density(cp1[[i]]))
> par(mfrow=c(5,2), cex=1)
> for( i in 1:9)
+  plot(density(cp2[[i]]))
```

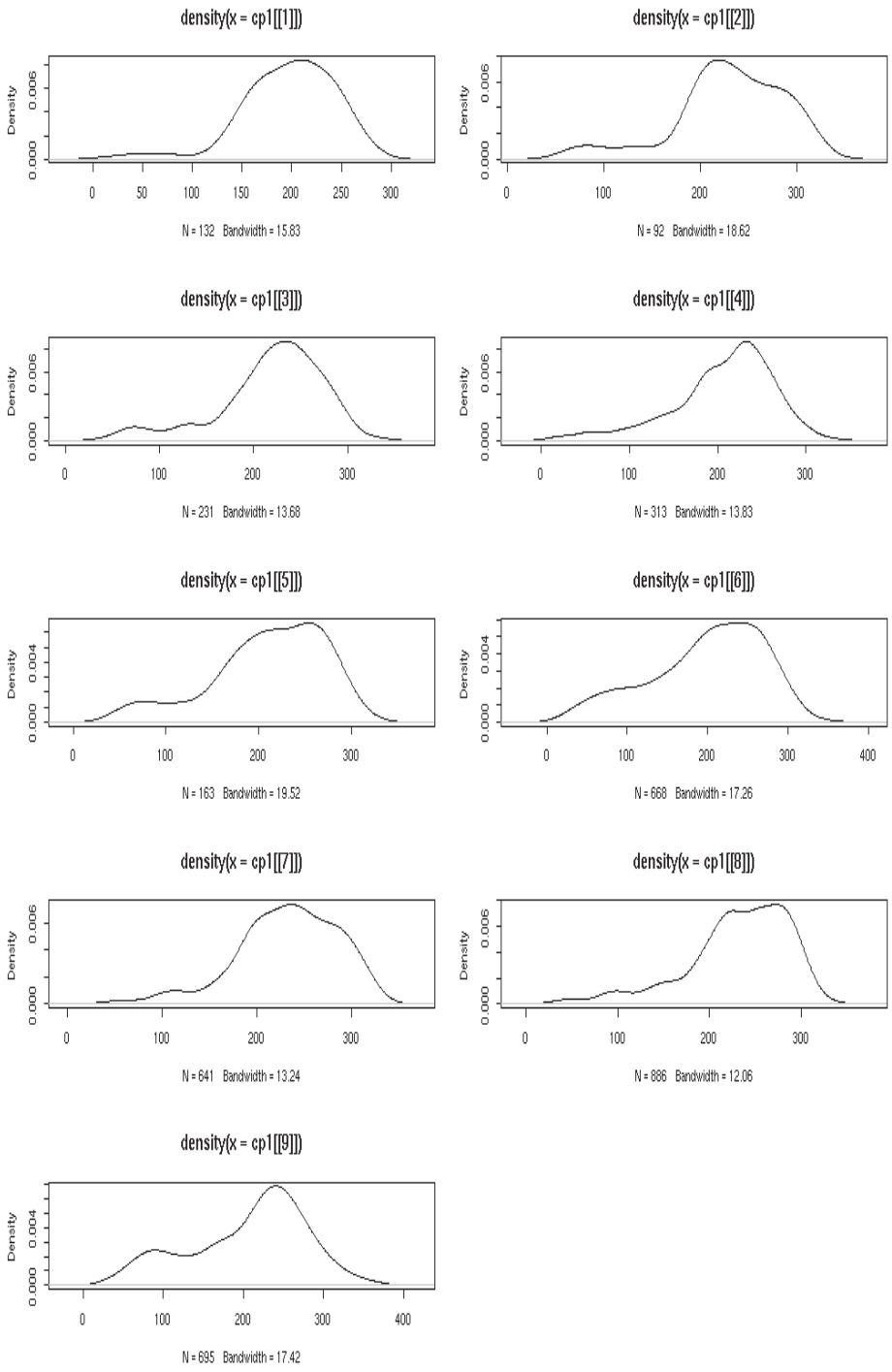


Fig. 25: Densidad de distribución de la comp. principal nº 1 en las diferentes clases.

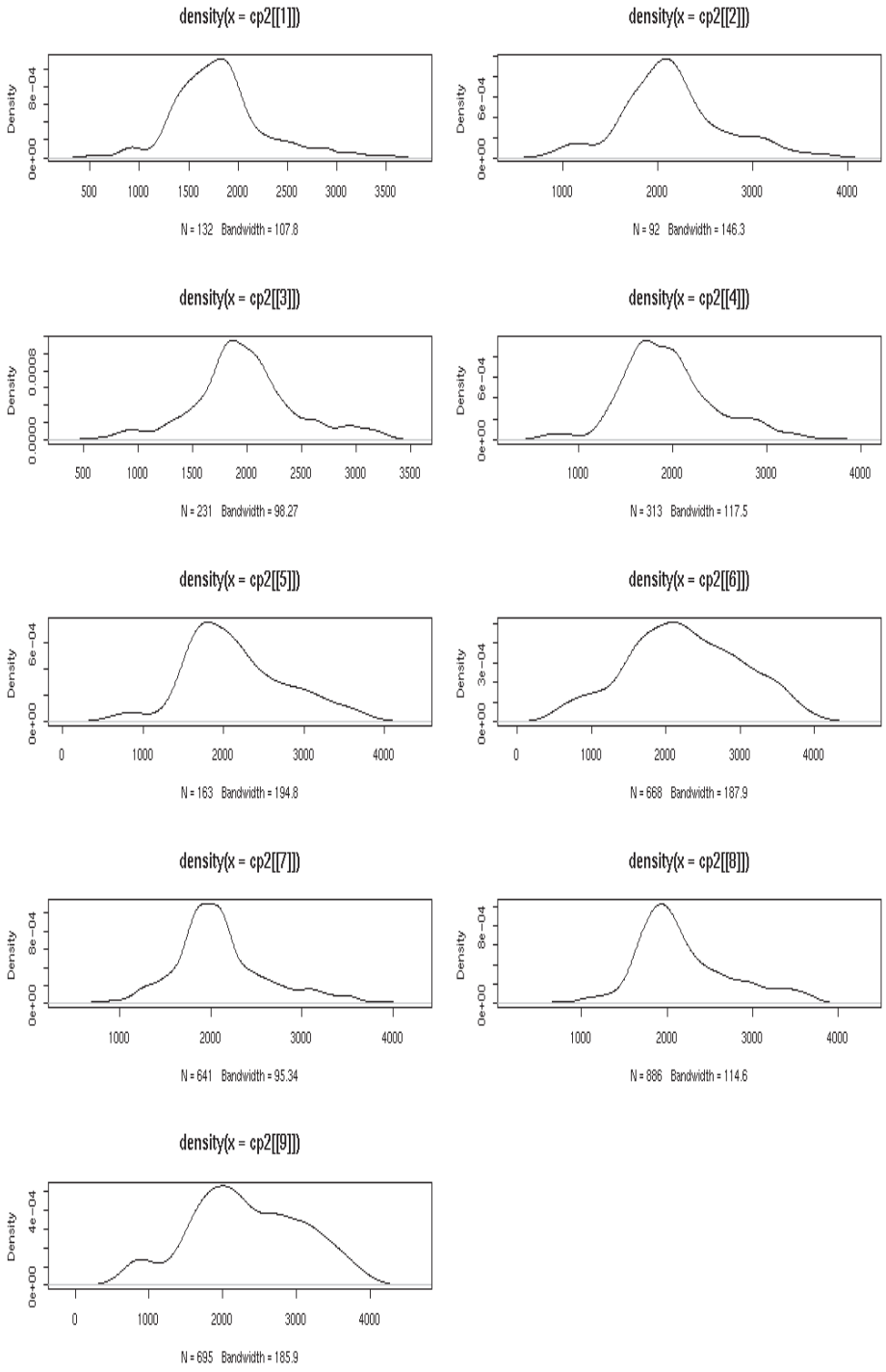


Fig. 26: Densidad de distribución de la comp..principal nº 2 en las diferentes clases.

4.1.4. *Conclusiones*

Las distribuciones de las observaciones del proceso no siguen una distribución normal multivariante. A simple vista no se aprecia un comportamiento claramente distinto entre las distintas clases, lo que apunta al rechazo de la hipótesis de su existencia, al menos tal y como se supone hasta ahora, en función del contenido en carbono del acero. La diferenciación entre clases queda, por tanto, cuestionada a la espera de técnicas estadísticas, no visuales, que posteriormente se aplicaran.

4.2. Correlación entre variables

4.2.1. *Fundamentos*

La matriz de covarianzas Σ de la población, esto es, de todas las posibles observaciones que pudieran presentarse, proporciona conocimiento acerca de la ligazón existente entre las distintas variables. Si la matriz de covarianzas es diagonal indica que, bajo condiciones de normalidad multivariante, las variables son independientes entre sí. Si además los elementos de la diagonal fueran iguales unos a otros, $\Sigma = \sigma \cdot I$, se estaría frente a una distribución hiper-esférica de datos.

La matriz de covarianzas Σ está formada por:

- $\Sigma_{ij} = \sigma_{ij} = E\left[(x_i - \mu_i)(x_j - \mu_j)\right]$, $(i \neq j)$: Covarianza entre las variables x_i, x_j.

- $\Sigma_{ij} = \sigma_i^2 = E\left[(x_i - \mu_i)^2\right]$, $(i = j)$: Varianza de la variable x_i.

Donde μ representa la esperanza matemática de la distribución considerada, a diferencia de $\bar{x}$, que representa la media muestral de las observaciones recogidas.

Las relaciones entre variables resultan más evidentes al ser observadas en la matriz de correlaciones ρ, formada por:

- $\rho_{ij} = \dfrac{\sigma_{ij}}{\sigma_{ii}\sigma_{jj}}$, $(i \neq j)$

- $\rho_{ij} = 1$, $(i = j)$

Estos elementos serán tanto más próximos a 1 (o -1) cuanto mayor sea la relación entre las variables, o lo que es lo mismo, cuanto mayor sea el parecido de la información contenida en ambas señales. Su signo es el correspondiente al producto de sus incrementos. Los términos nulos indican la ausencia de colinealidad de las variables involucradas. Habitualmente se considera como

variables altamente correlacionadas aquellas con ρ_{ij} mayor que un valor umbral que típicamente se adopta dentro del entorno 0.7 a 0.8.

Es fácil comprobar cómo puede obtenerse también la matriz ρ a partir de Σ mediante

$$\rho = D^{-\frac{1}{2}} \Sigma D^{-\frac{1}{2}}$$

La matriz D es una matriz diagonal cuyos elementos son las varianzas de cada una de las variables. Por ser diagonal, $D^{-\frac{1}{2}}$ se obtiene directamente elevando a $-\frac{1}{2}$ cada uno de sus elementos.

En el caso de una muestra, esto es, de un subconjunto aleatorio de la población, pueden definirse magnitudes análogas a las correspondientes a las distribuciones de la población total. De esta manera se define la matriz de covarianzas muestrales S, formada por:

- $S_{ij} = s_{ij} = \frac{1}{N-1} \sum_{h=1}^{N} (x_{ih} - \bar{x}_i)(x_{jh} - \bar{x}_j), \ (i \neq j)$: Covarianza muestral entre las variables x_i, x_j.

- $S_{ij} = s_i^{\,2} = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x}_i)^2$, $(i = j)$: Varianza de la variable x_i.

Análogamente la matriz de correlaciones muestral R puede definirse como

- $R_{ij} = \frac{s_{ij}}{s_{ii} s_{jj}}, \ (i \neq j)$

- $R_{ij} = 1, \ (i = j)$

Para muestras grandes S es un estimador de Σ, así como R lo es de ρ.

Se sigue cumpliendo que $R = D^{-\frac{1}{2}} S D^{-\frac{1}{2}}$

4.2.2. *Implementación en* **R**.

Para calcular en **R** la matriz de varianzas y covarianzas, en este caso para la familia 5, se pueden utilizar los siguientes comandos:

```
# Es útil una función que genere listas vacías de
# un número parametrizado de componentes.
> HazLista ← function(longitud, raiz.nombre)
+{
+   aux ← list('nombre'=NULL)
+   if (longitud>1)
+     for (i in 2:longitud)
+        aux ← c(aux, list('nombre'=NULL))
+   for (i in 1:longitud)
+        attributes(aux)$names[i] ← paste(raiz.nombre, i)
+   return(aux)
+}

# HazMclase genera listas cuyas componentes son arrays.
# Cada componente recoge los datos de una clase.
> HazMclase <- function(data.in, mylevels)
+ {
+ numclases <- length(levels(mylevels))
+ numvars   <- ncol(data.in)
+ Mclase <- HazLista(numclases,"Clase")
+ vars   <- HazLista(numvars,"var")
+ for (i in 1:numvar)
+   vars[[i]] <- split(data.in[,i], mylevels)
+ for (i in 1:numclases)
+   Mclase[[i]] <- matrix(0, ncol=numvars,
+                            nrow=length(vars[[1]][[i]]))
+ for (i in 1:numclases)
+   for (j in 1:numvars)
+     Mclase[[i]][,j]  <-as.vector(vars[[j]][[i]])
+ return(Mclase)
+}

> cortes  <- c(13,20,24,28,32,36,40,44,48,63)
> clase.datos <- cut(datos.nuevos[,6], cortes)
> Mclases ← HazMclases(datos.nuevos, clase.datos)
> cov(Mclases[[5]])
        [,1]   [,2]    [,3]   [,4]   [,5]   [,6]
[1,]  4797.7  21399   160.8    412   6.48   9.47
[2,] 21399.4 458135 -1833.4 -26982 303.83 -57.06
```

```
[3,]   160.8  -1833    33.1    245  -4.66   1.07
[4,]   412.0 -26982   245.3   4548  -5.05  23.10
[5,]     6.5    304    -4.7     -5 218.62  -0.32
[6,]     9.5    -57     1.1     23  -0.32   0.80
```

Si lo que se desea es la matriz de correlaciones se puede partir de la matriz de covarianzas [3]…

```
#Se calcula la matriz D^-1/2 .
> D05 ← diag((diag(var(Mclases[[5]])))^-0.5)

# La correlación muestral se obtiene como R = D^-1/2 S D^-1/2
> R <-  D05 %*% var(Mclases[[5]]) %*% D05
       [,1]    [,2]    [,3]    [,4]    [,5]   [,6]
[1,] 1.0000  0.456  0.404  0.0882  0.0063  0.153
[2,] 0.4564  1.000 -0.471 -0.5911  0.0304 -0.094
[3,] 0.4039 -0.471  1.000  0.6327 -0.0548  0.208
[4,] 0.0882 -0.591  0.633  1.0000 -0.0051  0.384
[5,] 0.0063  0.030 -0.055 -0.0051  1.0000 -0.024
```

Se puede comprobar que para el resto de familias se observa gran similitud. Puede utilizarse además la instrucción *corr* para obtener el mismo resultado directamente.

```
> cor(Mclases[[5]])
```

Analizando la matriz R se observa que; para la familia 5, que es a la que pertenece esta matriz de correlación; la variable 2 está fuertemente correlacionada, $R_{24} = 0.72$, con la variable 4. De la misma manera la variable 4 esta correlacionada con la 3 de forma significativa $R_{34} = 0.72$ lo que permitiría agrupar estas variables como un único *cluster* de variables.

La dependencia entre variables puede en ocasiones resultar más clara a los sentidos si se representa gráficamente la evolución de unas respecto a las otras. Los siguientes gráficos de dispersión representan estas relaciones para las distintas familias, observándose idénticas relaciones indiferentemente de la clase considerada. Es interesante la capacidad que proporciona la función *pairs* de personalizar el contenido de los gráficos que aparecen en el triángulo inferior, mediante la función *lower.panel*, las casillas de la diagonal, *diag.panel*, y el triángulo superior, *upper.panel*. Los gráficos generados han sido enriquecidos con información adicional gracias a la personalización de estas funciones. En el triangulo inferior del gráfico se muestran las relaciones entre variables indicando una aproximación polinomial a su naturaleza. En la diagonal se representa el histograma de cada variable dentro de esa clase y en

[3] El operador `%*%` es el producto de matrices. El operador `*` realizaría el producto de Hadamard (multiplicación elemento a elemento de las matrices).

el triangulo superior del gráfico el valor absoluto de las correlaciones con un tamaño proporcional a su magnitud.

```
# Función que representará en la diagonal el histograma de cada
# variable.
> panel.hist <- function(x, ...)
+ {
+   usr <- par("usr"); on.exit(par(usr))
+   par(usr = c(usr[1:2], 0, 1.5) )
+   h <- hist(x, plot = FALSE)
+   breaks <- h$breaks; nB <- length(breaks)
+   y <- h$counts; y <- y/max(y)
+   rect(breaks[-nB], 0, breaks[-1], y, col="cyan", ...)
+ }
# Función que representará el valor absoluto de las
# correlaciones.
> panel.cor <- function(x, y, digits=2, prefix="", cex.cor)
+  {
+   usr <- par("usr"); on.exit(par(usr))
+   par(usr = c(0, 1, 0, 1))
+   r <- abs(cor(x, y))
+   txt <- format(c(r, 0.123456789), digits=digits)[1]
+   txt <- paste(prefix, txt, sep="")
+   if(missing(cex.cor)) cex <- 0.8/strwidth(txt)
+   text(0.5, 0.5, txt, cex = cex * r)
+  }

> for (i in 1:numclases)
> pairs(Mclases[[i]],lower.panel=panel.smooth,
+ diag.panel=panel.hist, upper.panel=panel.cor)
```

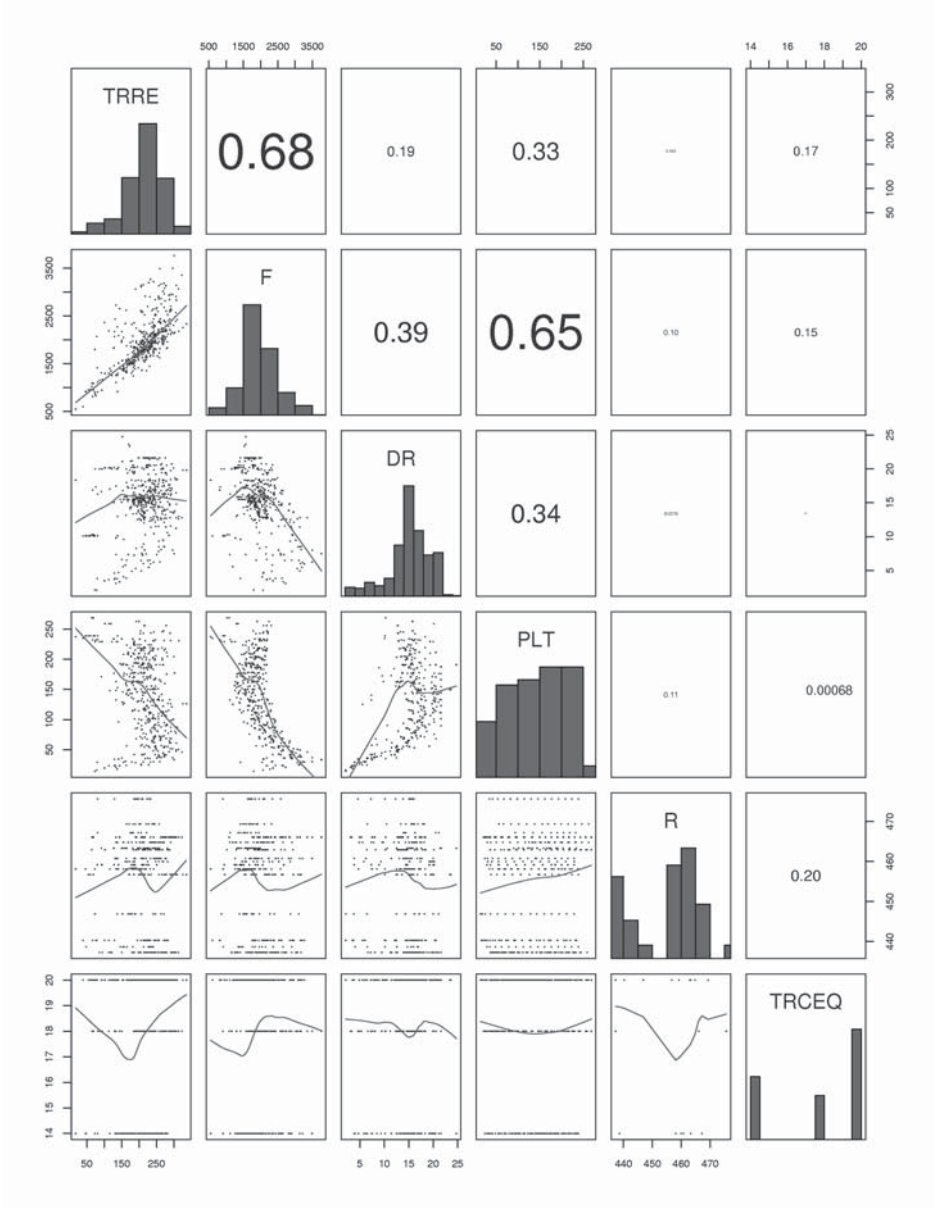


Fig. 27: Relación entre variables. Familia nº 1.

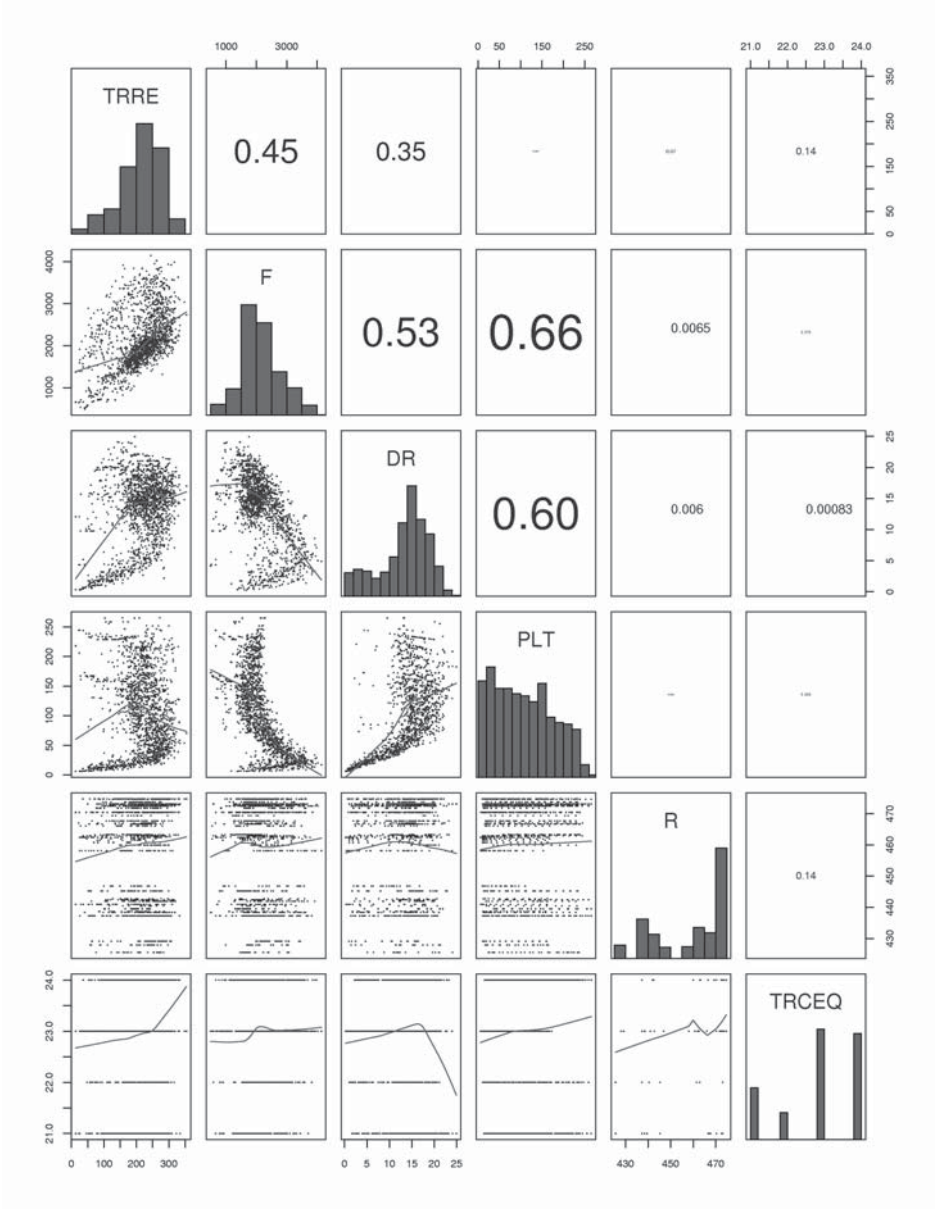


Fig. 28: Relación entre variables. Familia nº 2.

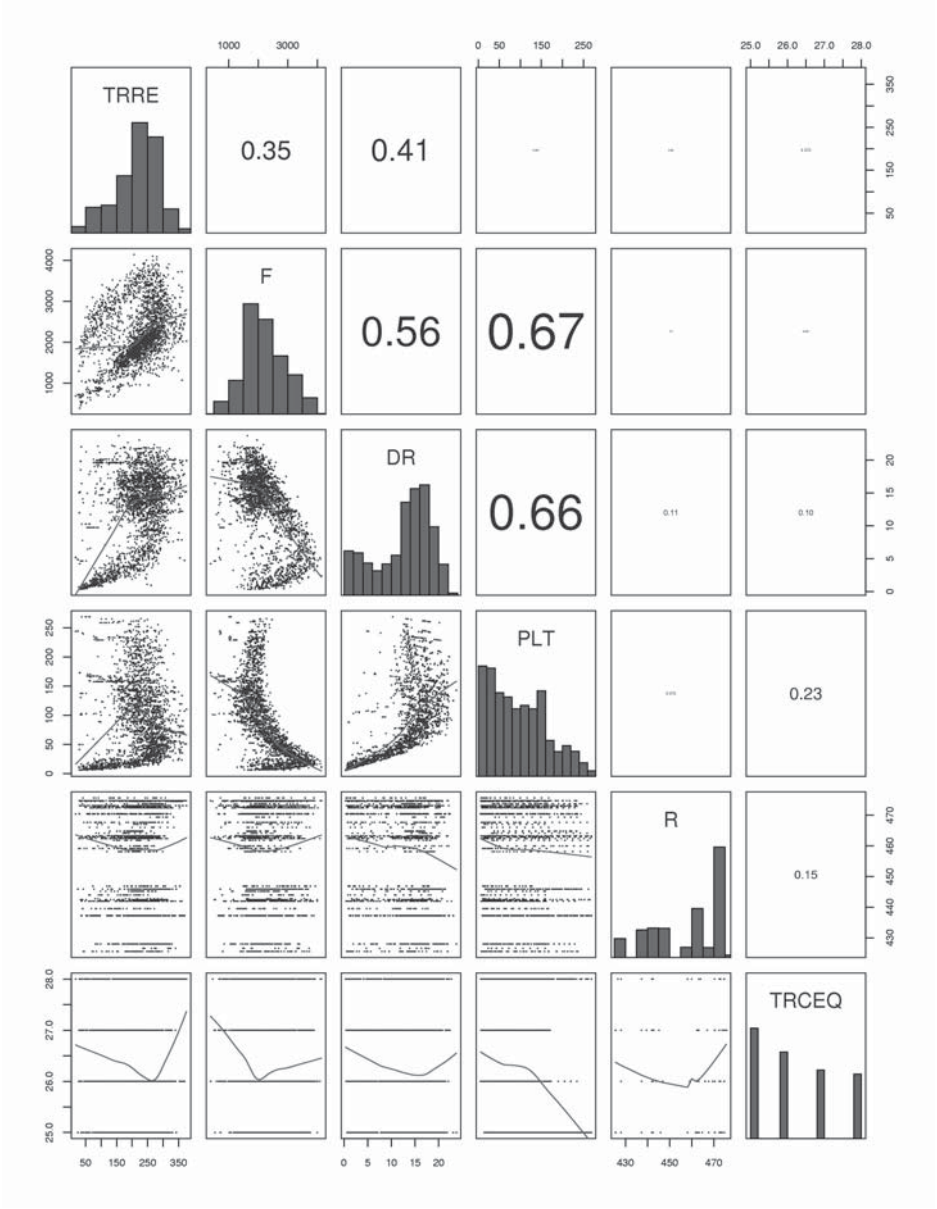


Fig. 29: Relación entre variables. Familia nº 3.

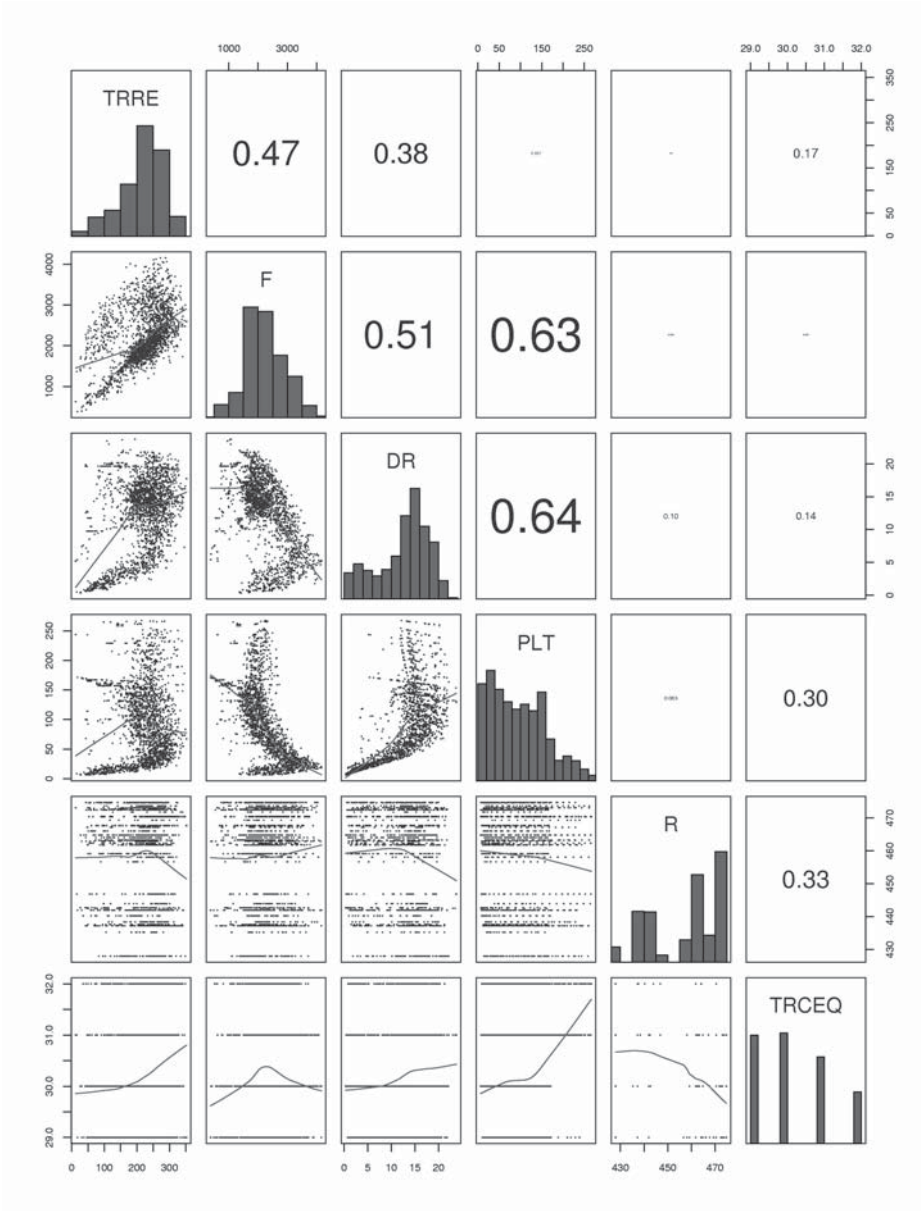


Fig. 30: Relación entre variables. Familia nº 4.

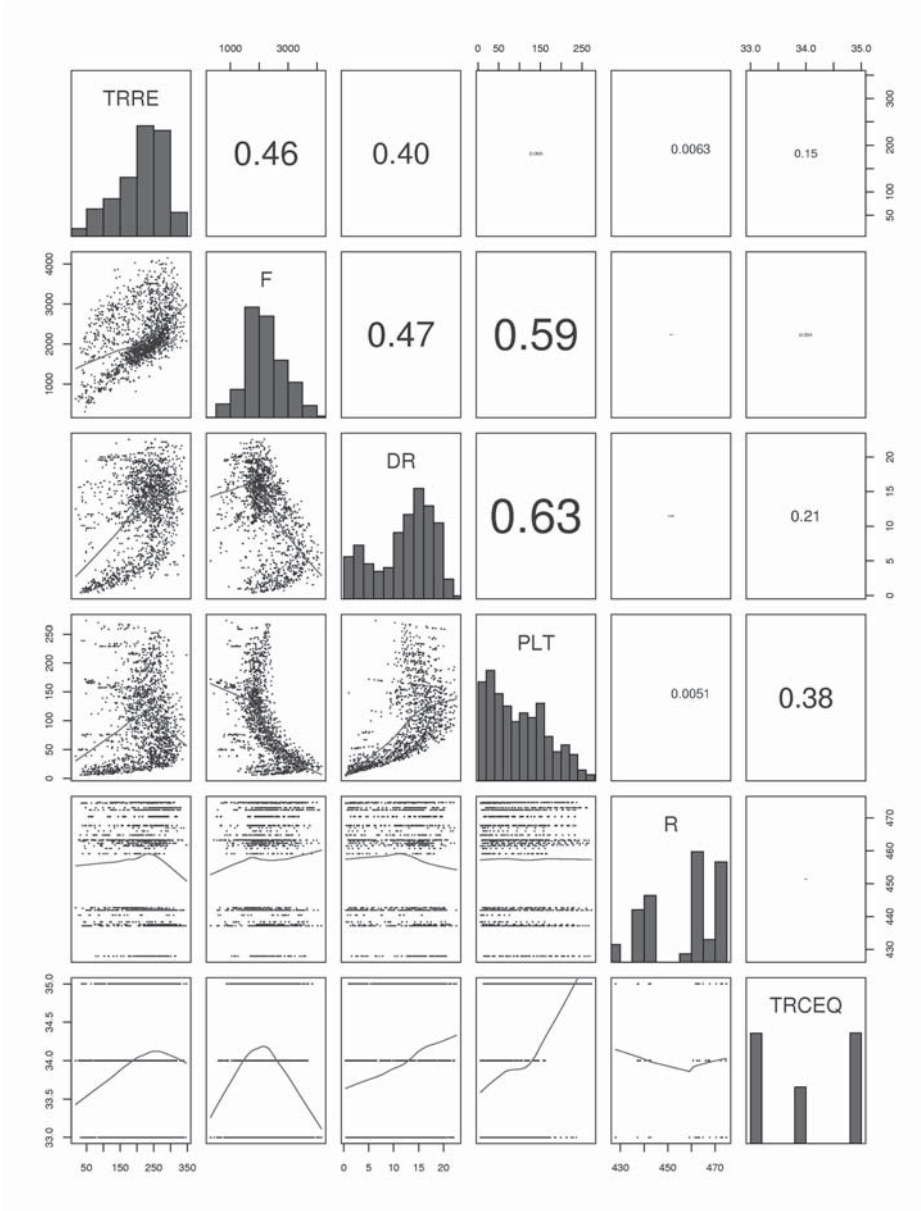


Fig. 31: Relación entre variables. Familia nº 5.

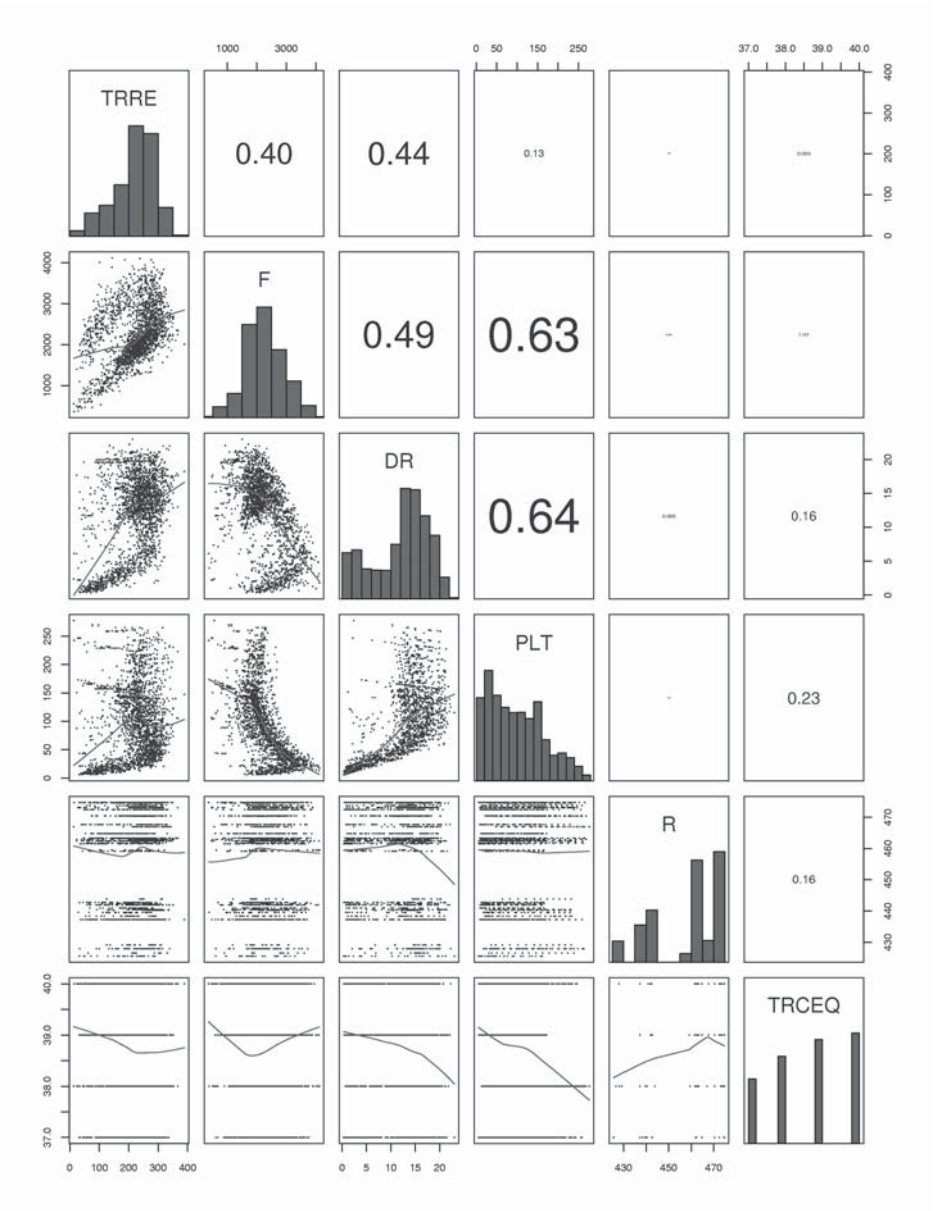


Fig. 32: Relación entre variables. Familia nº 6.

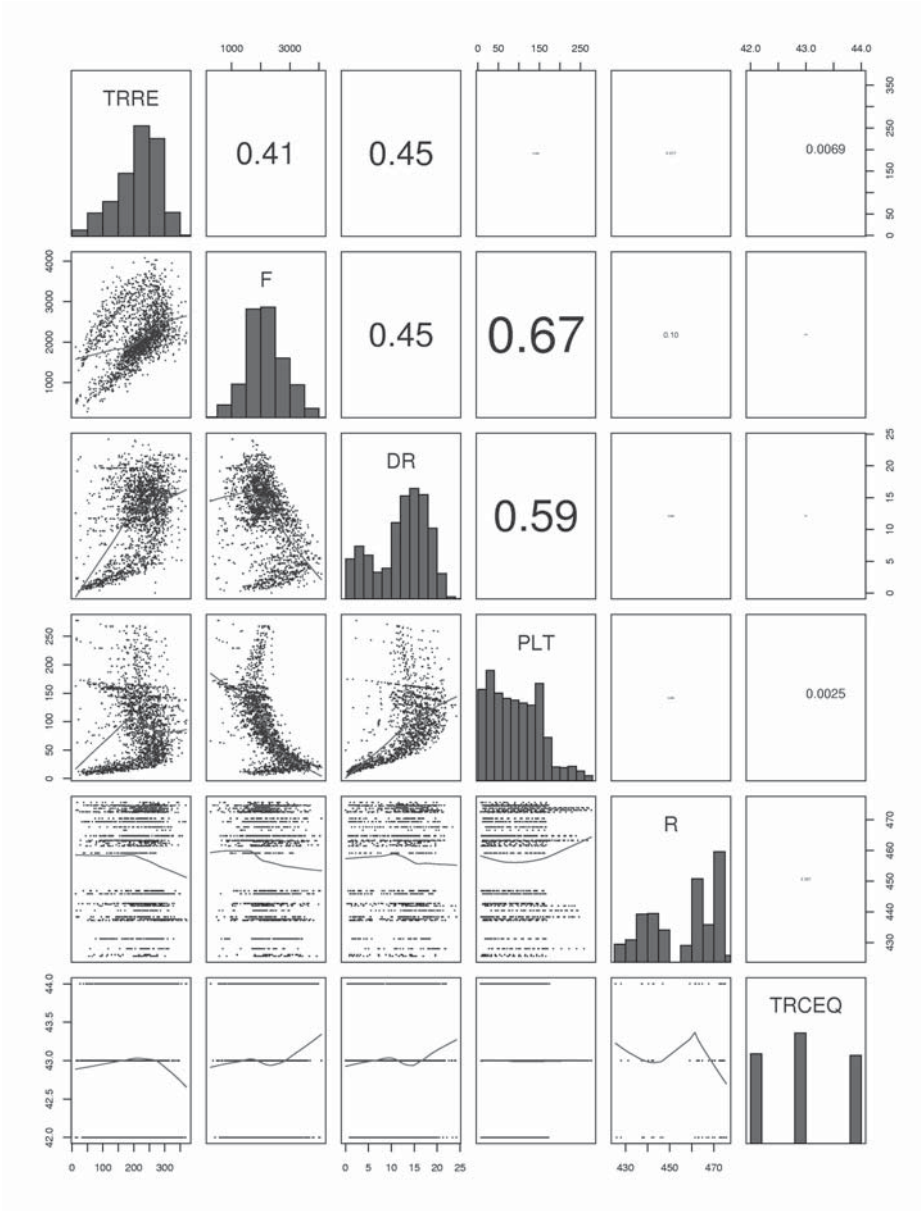


Fig. 33: Relación entre variables. Familia nº 7.

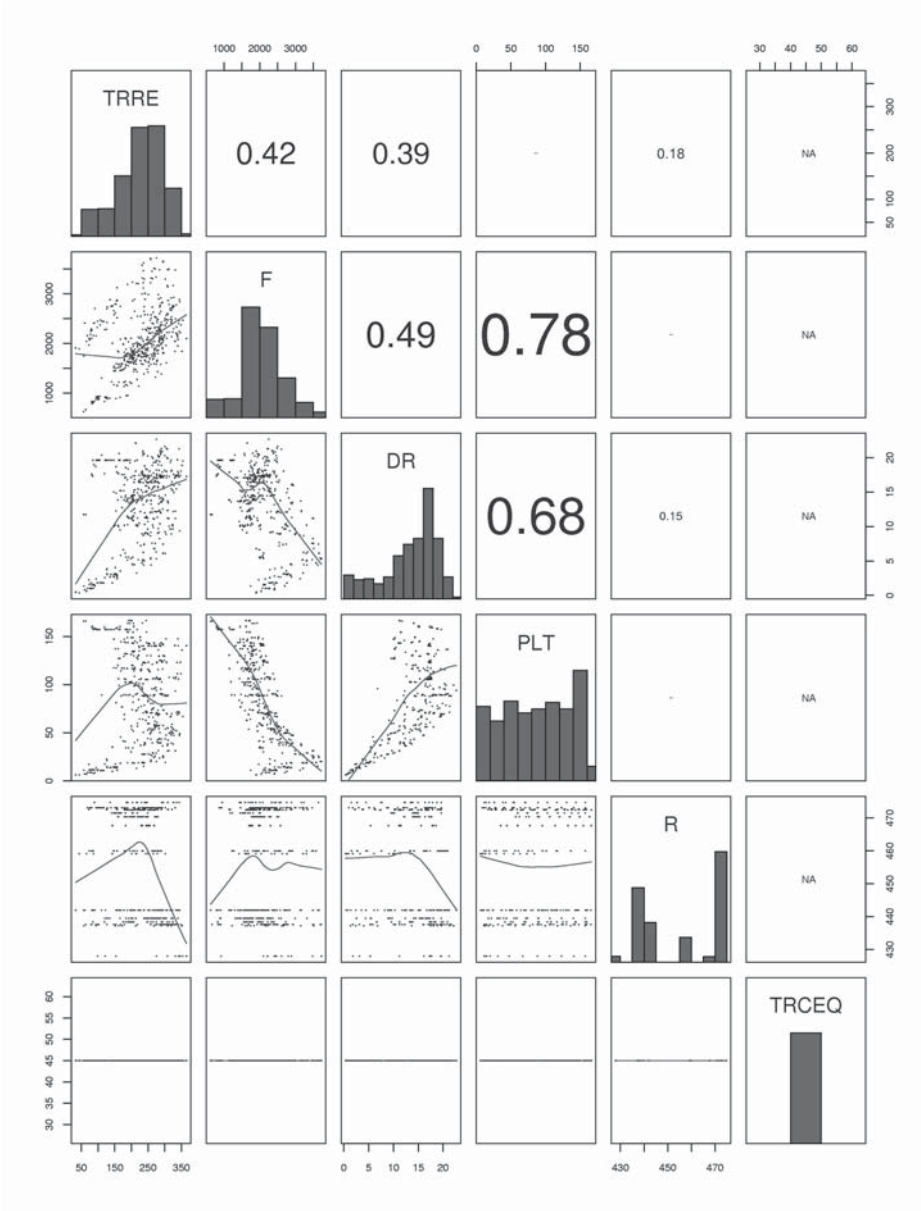


Fig. 34: Relación entre variables. Familia nº 8.

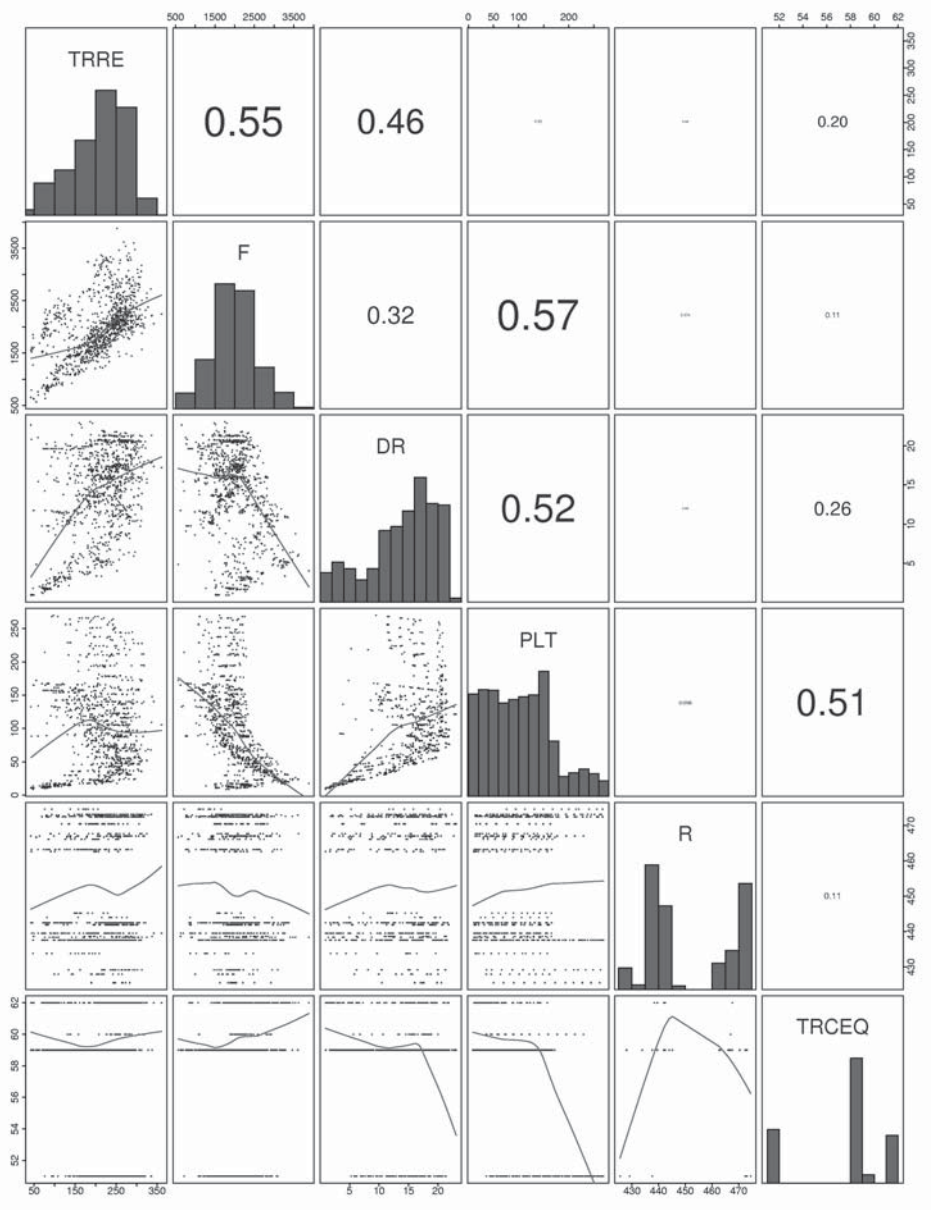


Fig. 35: Relación entre variables. Familia nº 9.

4.2.3. *Conclusiones*

Se ve claramente que las clases nº 1 y nº 2, por recoger un número de puntos inferior al del resto de clases, muestran menor densidad de puntos en sus gráficos. La variable nº 5 es categórica y adopta valores discretos que podrían corresponderse con diferentes consignas o puntos de funcionamiento. Las relaciones de las demás variables entre sí son claramente no lineales.

4.3. Análisis de Normalidad Multivariante

Es interesante comprobar si el comportamiento de las muestras sigue una distribución normal multivariante dado que gran número de herramientas asume condiciones de normalidad multivariante para la validez de sus resultados.

Se dice que una muestra es normal multivariante si cada una de sus n componentes esta formada por combinación lineal de distribuciones normales. Si Z es un vector cuyas componentes z_i siguen, cada una, una distribución normal $N(\mu_i,\sigma_i)$, y A es una matriz de coeficientes constantes, entonces la matriz X, definida como $X = A \cdot Z$ sigue una distribución normal multivariante.

El análisis de normalidad multivariante puede abordarse mediante métodos gráficos y utilizando técnicas estadísticas; ya sean estas de naturaleza univariante, aplicadas sobre las componentes principales; o de carácter multidimensional.

En el caso de las técnicas estadísticas de contraste de hipótesis, éstas se aceptarán o rechazarán en función del valor que adopten los estadísticos seleccionados, y del grado de confianza que deseemos adoptar en nuestras decisiones. El concepto de grado de confianza, o significación, está íntimamente ligado al concepto estadístico de intervalo de confianza, y dado el gran número de ocasiones en las que aparecerá en distintas formas (1 ó 2 colas) aclararemos previamente su significado.

Supongamos que determinado fenómeno tiene asociada una función de densidad de probabilidad como la de la Fig. 36, por ejemplo. Si conocemos el valor medio de la distribución podremos asegurar que las observaciones que se realicen se encontrarán en un entorno de ese valor medio cuya amplitud aumenta a medida que depositamos mayores grados de confianza en nuestras aseveraciones. En la medida en que se amplíe el *intervalo de confianza*, con mayor probabilidad se recogerán en él las observaciones realizadas, cuantificándose esa probabilidad como el área bajo la curva de densidad de probabilidad que se abarca en el intervalo.

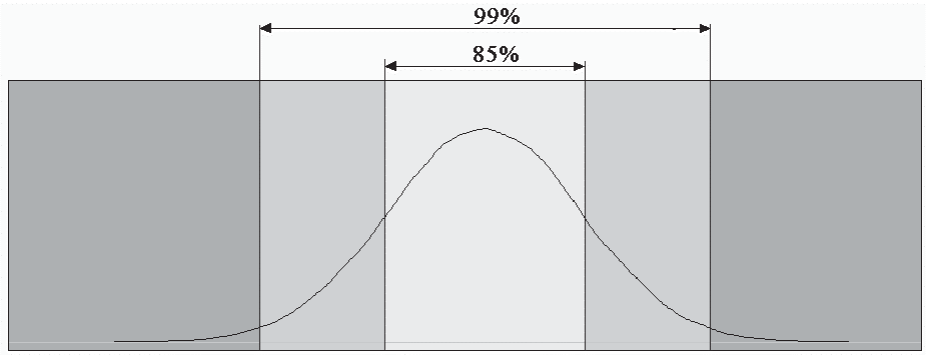


Fig. 36: Diversos intervalos de confianza en una distribución normal.

Los contrastes pueden ser paramétricos o no paramétricos. El término paramétrico hace mención a la necesidad de datos distribuidos normalmente, mientras que los contrastes no paramétricos son válidos aun cuando la distribución real de los datos se aleja en gran medida de la normalidad.

En función de la hipótesis alternativa; hipótesis que se confirma al rechazar la nula; podemos tener distintos tipos de tests:

- $\theta_1 \neq \theta_2$: Contraste a dos colas (en el argot de **R** "*Two sided*").
- $\theta_1 > \theta_2$: Contraste a una cola positivo (en **R** "G*reater* ").
- $\theta_1 < \theta_2$: Contraste a una cola negativo. (en **R** "*Less*").

Para el contraste de determinada hipótesis, el valor de la significación (α) adoptada, representa el mínimo valor del grado de confianza que deberíamos haber escogido para nuestro intervalo de modo que una observación posterior hubiera recaído en su interior. Cuanto mayor sea entonces la significación menor será este grado de confianza necesario para abarcar las observaciones en el intervalo, y por tanto, cuanto mayor sea la significación mayor validez deberemos dar a la hipótesis de partida. La significación se cuantifica como el área abarcada por las colas de la distribución considerada, definiéndose estas como el área de rechazo de la hipótesis nula. El p-value del estadístico de una muestra concreta expresa el valor de la significación límite para que la hipótesis correspondiente a ese estadístico sea aceptada. El número de colas del contraste depende de si estamos en un contraste "Two sided", con lo que tendríamos dos colas, o en un "greater than", "less than" con lo que sólo es precisa una de las colas.

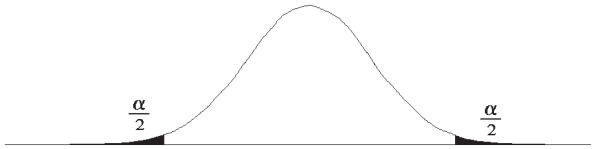


Fig. 37: Dos colas en una distribución normal.

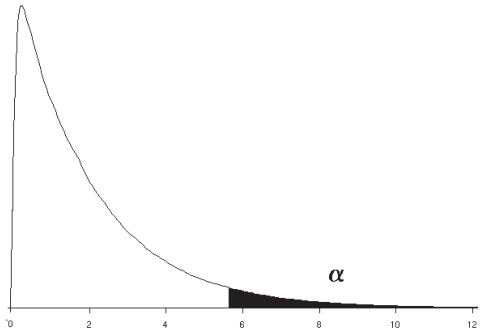


Fig. 38: Una cola en una distribución χ^2

En el caso de un estadístico que siguiera una distribución normal como la de la Fig. 37, la significación aumentaría a la par que el área de las dos colas como se muestra en la figura.

En el caso, por ejemplo, de la detección de observaciones atípicas, en el que se calcula un estadístico que sigue una distribución χ^2, como la de la Fig. 38, la significación únicamente puede crecer por una de las colas, puesto que no es físicamente posible que tal estadístico (realmente distancia de Mahalanobis, positiva por definición) adopte valores negativos. Un valor, por ejemplo de $\alpha = 0.16$, indicaría que se hubiera aceptado como válida la distribución supuesta incluso con un grado de confianza del 85% (caso b de la Fig. 39), lo cual es claramente mejor que otro valor de $\alpha = 0.001$ (caso a de la Fig. 39), que únicamente hubiera aceptado la hipótesis a partir de grados de confianza del 99.9%, lo cual es ciertamente restrictivo.

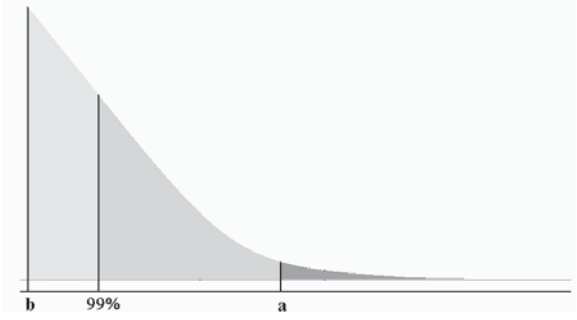


Fig. 39: Distintos grados de significación de un estadístico.

4.3.1. *Análisis multidimensional directo*

Este método aborda el análisis de normalidad mediante técnicas propiamente multidimensionales, basándose en parámetros propios de la distribución multivariante.

Si se definen los coeficientes multivariados de asimetría y kurtosis **[12**, Pág. 148**]** $b_{1,n}$ y $b_{2,n}$, respectivamente como

- $b_{1,n} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} m_{ij}^3$
- $b_{2,n} = \frac{1}{N}\sum_{i=1}^{N} m_i^2$

donde

$$m_i = (x_i - \bar{x})^t S^{-1}(x_i - \bar{x})$$

$$m_{ij} = (x_i - \bar{x})^t S^{-1}(x_j - \bar{x})$$

se tiene que para muestras grandes el coeficiente $\frac{N}{6}b_{1,n}$ sigue una distribución χ^2 con $\frac{n(n+1)(n+2)}{6}$ grados de libertad. También para muestras grandes, el coeficiente $b_{2,n}$ sigue una distribución normal univariante de media $n(n+2)$ y varianza $\frac{8n(n+2)}{N}$. Ambos coeficientes son invariantes por transformaciones lineales y se utilizarán como estadísticos para el test de normalidad.

4.3.2. *Implementación en* **R**

```
> asimetria <- function(data.in)
+ {
+ N <- nrow(data.in); n <- ncol(data.in)
+ mij3 <- 0
+ invS <- solve(cov(data.in))
+ for (i in 1:N)
+     for (j in 1:N)
+        mij3 <- mij3 + (data.in[i,] %*% invS %*% data.in[j,])^3
+ Xalpha0.01 <- qchisq(0.99, n*(n+1)*(n+6)/6)
> return (list(N_b1n_6=mij3/(N*6), Xalpha0.01=Xalpha0.01 ))
> }
> datos.nuevos.b1n <- asimetria(datos.nuevos)
> unlist(datos.nuevos.b1n )
```

```
$N_b1n_6                $Xalpha0.01
[1,] 2.794357e+12       [1] 117.0565
```

El valor del estadístico cae en la zona de rechazo de la hipótesis habiéndose considerado un grado de confianza en la zona de aceptación de la hipótesis del 99%. Su significación es inferior al 1%.

No obstante se preferirán los siguientes métodos frente a éste por su elevado coste computacional.

```
> kurtosis <- function(data.in)
> {
> n <- ncol(data.in)
> N <- nrow(data.in)
> mahadatos.nuevos <- mahalanobis(datos.nuevos,
+   apply(datos.nuevos, 2, mean), cov(datos.nuevos))
> Xalpha0.01 <- qnorm(0.99, n*(n+2), sqrt(8*n*(n+2)/N))
>  return(list(b2n = sum(mahadatos.nuevos^2)/N),
+              Xalpha0.01=Xalpha0.01)
}
> datos.nuevos.kurtosis <- kurtosis(datos.nuevos)
> unlist(datos.nuevos.kurtosis)
$b2n                    $Xalpha0.01
[1] 51.07926            [1] 48.37294
```

El estadístico $b_{2,n}$ se escapa del umbral de referencia del 99% y por tanto cae en la región de rechazo. Su significación es inferior al 0.01%.

4.3.3. *Análisis de la normalidad de las componentes principales*

Puede demostrarse que si las observaciones correspondientes a cada una de las componentes principales sigue una distribución normal univariante, el conjunto sigue una distribución normal multivariante. Nótese que en una base no principal, que cada coordenada supere los test de normalidad, es una condición necesaria pero no suficiente para asegurar la normalidad multivariante.

La normalidad de cada componente puede ser contrastada utilizando distintos tests no paramétricos como el χ^2, Shapiro-Wilk, Kolmogorov-Smirnov. Este último será el aquí utilizado. Su uso no se restringe únicamente a distribuciones normales sino que puede ser utilizado para cualquier tipo de función de probabilidad teórica de tipo continuo.

El test de Kolmogorov-Smirnov (K-S), se basa en la comparación de la función de probabilidad acumulada teórica y la observada. Se calculan primeramente los estadísticos normalizados $z_i = \frac{x_i - \overline{x}}{s}$. A continuación, se calcula la función de densidad de probabilidad acumulada teórica correspondiente a z_i ; considerando que siguen una distribución normal; $F(i) = \Phi(i)$, donde $\Phi(i)$

representa la función de densidad de probabilidad acumulada de la distribución normal standard. El valor del estadístico $D = \max\left\{\left|\frac{i}{N - F(i)}\right|\right\}$, $i = 1,2,\ldots,N$, multiplicado por el factor $\left(\sqrt{N} - 0.01 + \frac{0.85}{\sqrt{N}}\right)$, usando como valores críticos los definidos en **[13]**.

Es aconsejable aplicar las desigualdades de Bonferroni. Si $H_1, H_2, \ldots, H_n$ representan las afirmaciones "$\overline{w}_1$ es normal", "$\overline{w}_2$ es normal", …, "$\overline{w}_n$ es normal", ($\overline{w}_1, \overline{w}_2, \ldots, \overline{w}_n$ son las componentes principales de la muestra), entonces se verifica

$$p(H_1 \cap \ldots \cap H_n) \geq 1 - \sum_{i=1}^{n} p(\overline{H}_i)$$

siendo $p(\overline{H}_i)$ la probabilidad de que la afirmación $\overline{H}_i$ sea rechazada cuando es cierta. $p(H_1 \cap \ldots \cap H_n)$ es la probabilidad de que las n afirmaciones sean aceptadas cuando son ciertas, y exista por tanto, normalidad multivariante. Si se considera entonces que el nivel de significación para el test de normalidad sobre cada componente es $\frac{\varepsilon}{n}$; $p(\overline{H}_i) = \frac{\varepsilon}{n}$ (probabilidad de equivocarnos al rechazar que $\overline{w}_i$ es normal). La probabilidad de acertar al afirmar que la distribución es normal multivariante será:

$p(H_1 \cap \ldots \cap H_n) \geq 1 - n\frac{\varepsilon}{n} = 1 - \varepsilon$ lo que se traduce en que el nivel de significación para esta prueba conjunta es inferior a ε. Si deseamos, por ejemplo, comprobar la normalidad multivariante de la muestra con un nivel de significación $\varepsilon = 0.05$, con 5 variables tendremos que comprobar la normalidad de cada componente principal con un nivel de significación $\frac{\varepsilon}{5} = \frac{0.05}{5} = 0.01$.

4.3.4. *Implementación en* **R.**

```
# Test de Kolmogorov-Smirnov para la 1ª y 2ª comp. principales.
> MD ← pca(datos.nuevos,2)
> ks.test(MD[[1]][,1], "pnorm", mean=mean( MD[[1]][,1] ), sd=sqrt(var(MD[[1]][,1])))
        One-sample Kolmogorov-Smirnov test
data:  MD[[1]][, 1]
D = 0.0634, p-value = < 2.2e-16
alternative hypothesis: two.sided

# Test de Kolmogorov-Smirnov para la 2ª componente principal.
> ks.test(MD[[1]][,2], "pnorm", mean=mean( MD[[1]][,2] ), sd=sqrt(var(MD[[1]][,2])))
        One-sample Kolmogorov-Smirnov test
data:  MD[[1]][, 2]
D = 0.0891, p-value = < 2.2e-16
alternative hypothesis: two.sided

# Test de Kolmogorov-Smirnov para la 3 componente principal.
> ks.test(MD[[1]][,3], "pnorm", mean=mean( MD[[1]][,3] ), sd=sqrt(var(MD[[1]][,3])))
        One-sample Kolmogorov-Smirnov test
data:  MD[[1]][, 3]
D = 0.0682, p-value = < 2.2e-16
alternative hypothesis: two.sided

# Test de Kolmogorov-Smirnov para la 4 componente principal.
> ks.test(MD[[1]][,4], "pnorm", mean=mean( MD[[1]][,4] ), sd=sqrt(var(MD[[1]][,4])))
        One-sample Kolmogorov-Smirnov test
data:  MD[[1]][, 4]
D = 0.1727, p-value = < 2.2e-16
alternative hypothesis: two.sided

# Test de Kolmogorov-Smirnov para la 5 componente principal.
> ks.test(MD[[1]][,5], "pnorm", mean=mean( MD[[1]][,5] ), sd=sqrt(var(MD[[1]][,5])))
        One-sample Kolmogorov-Smirnov test
data:  MD[[1]][, 5]
D = 0.0691, p-value = < 2.2e-16
alternative hypothesis: two.sided
```

En este caso los distintos test rechazan las hipótesis de normalidad univariante de las componentes principales con valores de significación prácticamente nulos y por tanto la normalidad multivariante debe ser rechazada.

Existen otros tests de normalidad, como el de Wilk-Shapiro **[11**, Pág 67**]**, implementado en **R** como *shapiro.test()*. Al aplicarse sobre los datos considerados proporciona la misma conclusión que el de Kolmogorov-Smirnov.

4.3.5. *Métodos visuales*

Los dos tipos de gráficos más utilizados en la comparación de dos muestras son las curvas de probabilidad acumulada y la recta cuantil cuantil.

En el primero de ellos se trazan las curvas de probabilidad acumulada de ambas distribuciones y se compara la forma de ambas. En la Fig. 40 se muestran las curvas correspondientes a una $N(0,1)$ frente a una $\chi^2_{df=2}$.

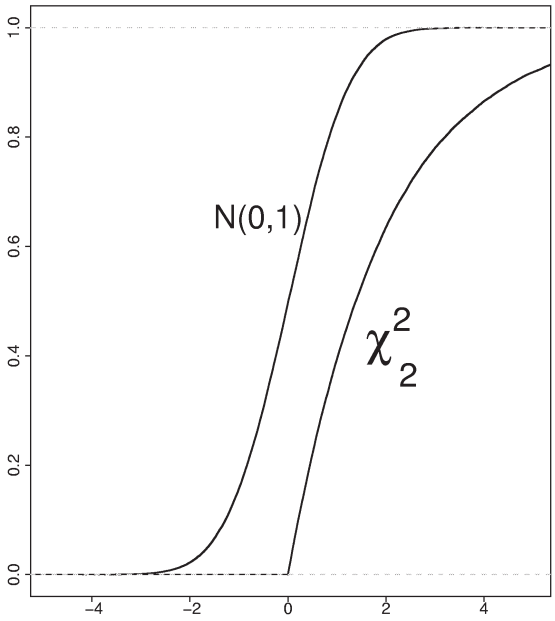


Fig. 40: Curvas *ecdf* para dos distribuciones

La recta cuantil-cuantil traza los cuantilies observados frente a los esperados. Dada una muestra de N observaciones $\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_N$ de una variable aleatoria unidimensional, se denominan cuantiles a los valores de estas observaciones reordenados de menor a mayor valor $\bar{z}_{(1)}, \bar{z}_{(2)}, \ldots, \bar{z}_{(N)}$.

Si se representan gráficamente los valores observados de los cuantiles frente a los valores esperados para ese tipo concreto de distribución de probabilidad, debería obtenerse una línea recta del tipo identidad. Si el resultado (gráfico cuantil-cuantil o *Q-Q Plot*) no es dicha línea recta la distribución de la muestra no se corresponde con la tomada como referencia.

Si la recta no pasa por el origen, la muestra tiene su media desplazada frente a la de referencia. Si la pendiente no es unitaria significa que existe un factor de escala entre ambas distribuciones.

De la forma del gráfico Q-Q se puede extraer información adicional. Si por ejemplo apareciese un trazo anormalmente brusco en alguno de los extremos de la recta, indicaría la presencia de espureos en la muestra. La curvatura del gráfico en los extremos indica kurtosis, mientras que la convexidad o concavidad en el gráfico sugieren falta de simetría en la distribución de la muestra.

4.3.6. *Implementación en* **R**.

La generación del gráfico ecdf es inmediata gracias a la función *ecdf* de la librería *stepfun*.

```
> library(stepfun)
> par(mfrow=c(2,3))
>  for  ( i in 1:6)
+ {
+ par(col="black",pch=".")
+ plot(ecdf(datos.nuevos[,i]), pch=".",
+      main=names(datos.nuevos)[i])
+ par(col="red",pch=".");
+ plot(ecdf(rnorm(40000,mean(datos.nuevos[,i]),
+      sd(datos.nuevos[,i]))),add=T)
+ }
```

Las curvas obtenidas se muestran en la Fig. 41. Se observa como las variables recogidas presentan un comportamiento claramente no normal. Sus curvas ecdf (en trazo negro) son distintas a las ecdf normales que les corresponderían (trazo rojo).

Es interesante, asimismo, obtener las curvas *ecdf* correspondientes a las componentes principales.

```
> par(mfrow=c(2,3))
> for  ( i in 1:6)
+ {
+ par(col="black",pch=".")
+ plot(ecdf(datos.nuevos.pca$rproj[,i]), main=paste("CP", i))
+ par(col="red",pch=".")
+ plot(ecdf(rnorm(40000,mean(datos.nuevos.pca$rproj[,i]),
+      sd(datos.nuevos.pca$rproj[,i]))),add=T)
+ }
```

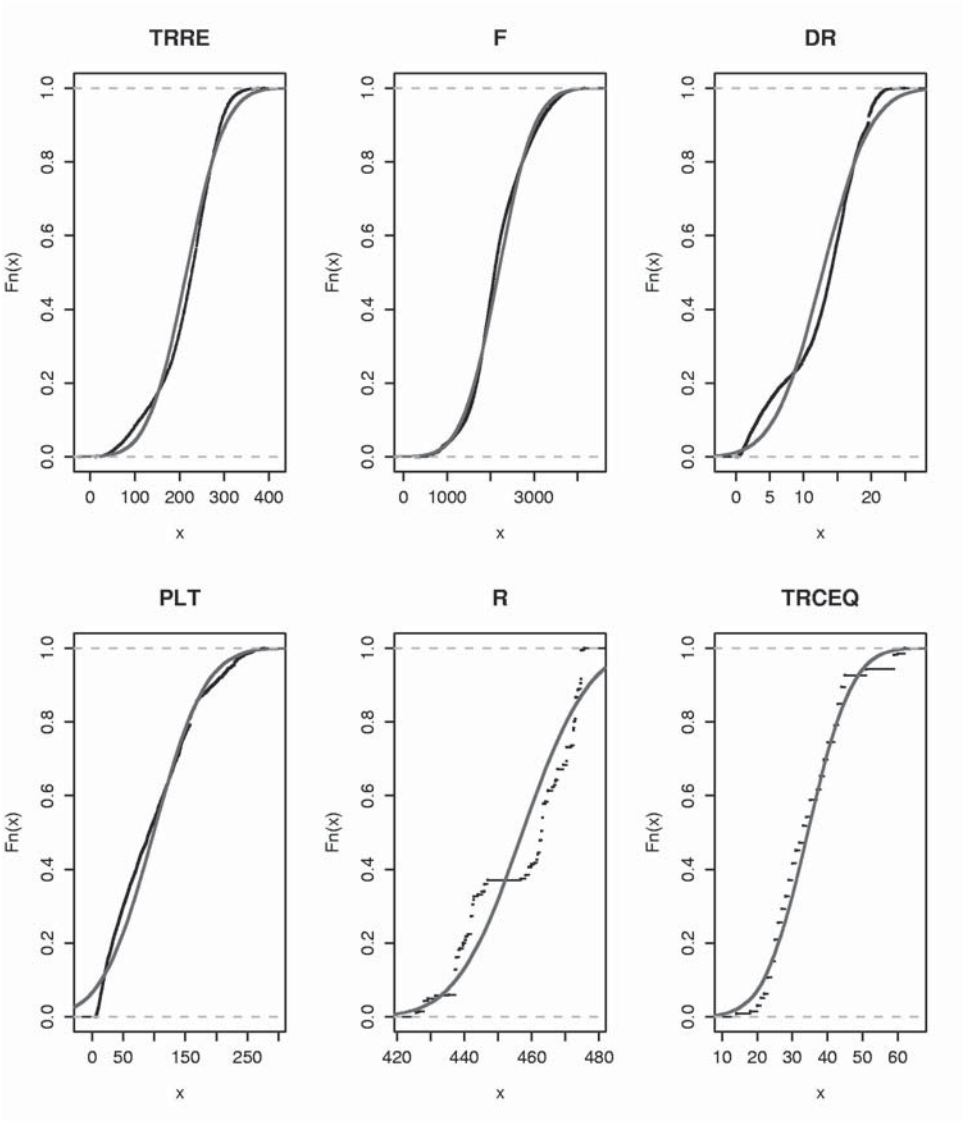


Fig. 41: Curvas *ecdf* de las variables del proceso frente a distribuciones normales.

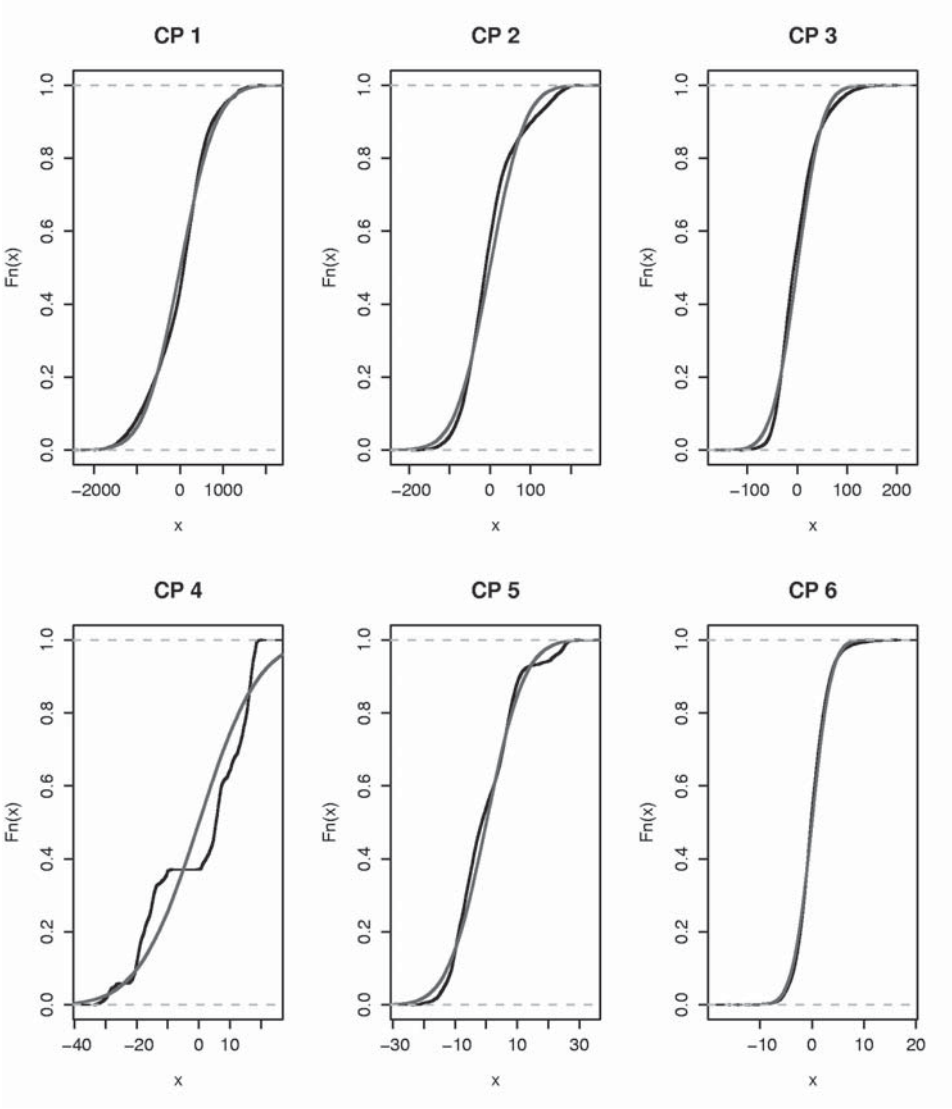


Fig. 42: Curvas *ecdf* de las comp. principales del proceso frente a ecdfs normales.

Las componentes principales tampoco muestran un comportamiento normal. Merece la pena, eso sí, observar más de cerca la curva correspondiente a la primera componente principal, por su mayor relevancia (98%). Se diferencia con mayor detalle su trayectoria de la esperada si fuera normal.

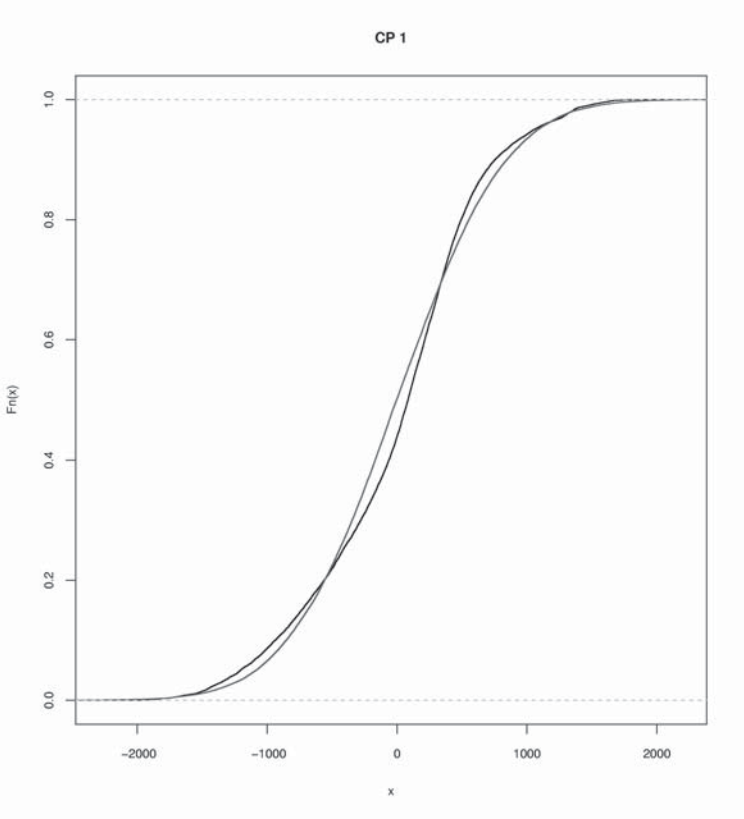


Fig. 43:Curva ecdf para la primera Comp. Princ. frente a la normal correspondiente.

La función *mahalanobis* calcula la distancia de Mahalanobis de cada observación al centro de gravedad de la muestra Si los datos de trabajo proceden de una distribución normal multivariada, las distancias de Mahalanobis de sus observaciones centradas sigue una χ^2 cuyo grado de libertad es igual al número de variables de la muestra. Se compararán los cuantiles correspondientes a la distribución χ^2 esperada y los obtenidos en el muestreo.

```
> mediadatos ← apply(datos.nuevos,2,mean)
> covdatos ← var(datos.nuevos)
> D ← mahalanobis(datos, mediadatos, covdatos)
> n ← numvars; N ← length(D)
> qqplot(qchisq(ppoints(N), df=numvars),D)
```

```
# La funcion ppoints genera una secuencia equiespaciada y
# ordenada de observaciones unidimensionales.
> abline(0,1,"Red")        #Recta y=x
> abline(0,0.85,"Blue")    #Recta y=1.5x
> abline(0,1.15,"Blue")    #Recta y=0.85x
```

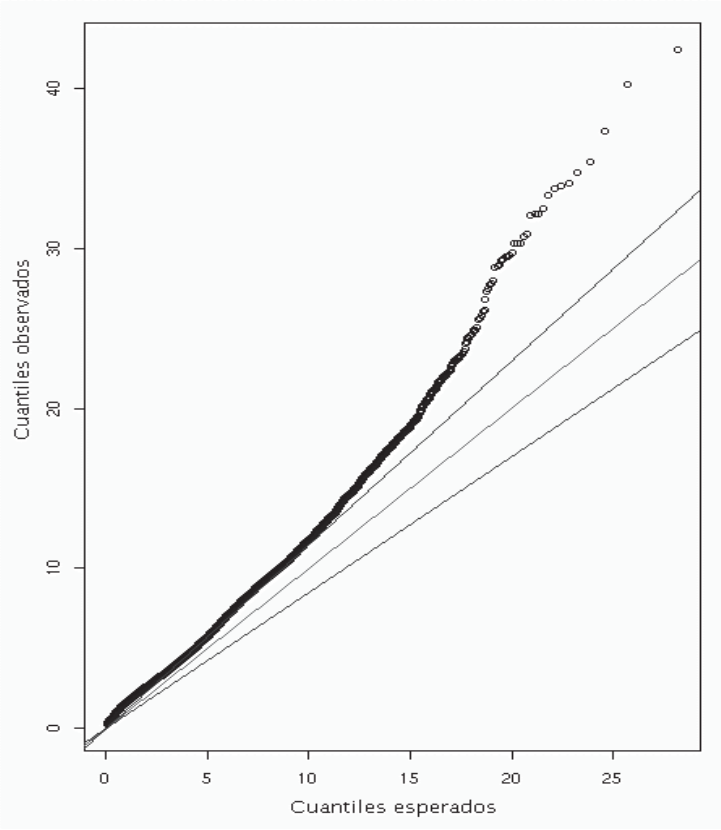


Fig. 44: Gráfico Cuantil-Cuantil

Se observa que la función de probabilidad acumulada de las distancias observadas al cuadrado respecto a los valores esperados resulta monótona creciente pero su pendiente no se ajusta a la unitaria, (ni siquiera permanece en la banda del 15% de tolerancia) con lo que se rechaza la hipótesis de normalidad multivariante anteriormente aceptada.

4.3.7. *Conclusiones*

Las pruebas realizadas sobre la muestra indican la no normalidad de la misma. No son, por tanto, aplicables, aquellas herramientas estadísticas que supongan normalidad de la distribución multivariante.

Capítulo 5

Análisis temporal

Los tests parámetricos como el *t* de Student, el ANOVA, MANOVA e incluso otros test no paramétricos, como el de Wilcoxon, pueden ver afectada su correcta aplicación si existe correlación seriada de sus componentes. Esto es, que la información contenida en una determinada variable, no guarde relación estrecha consigo misma retrasada un cierto número de muestra.

El análisis temporal de una serie de datos cobra especial relevancia en el ajuste de modelos ARIMA (Auto Regressive Integrated Moving Average). Las características principales de una serie temporal son la tendencia, la estacionalidad, la ciclicidad y la aleatoriedad.

5.1. Implementación en *R.*

Las dos funciones básicas implementadas en **R** que permiten el análisis temporal son *ts.plot* y *acf*. La primera de ellas muestra la secuencia temporal de los valores que sucesivamente han sido recogidos en una variable. La segunda muestra los valores de la autocorrelación parcial.

Los efectos temporales que principalmente pueden estar presentes en una serie temporal son la periodicidad, la estacionalidad, y la tendencia.

En los datos del proceso no es posible que afecten tales efectos puesto que fueron desordenados temporalmente en una etapa previa a su análisis.

Sin embargo, puede mostrarse el resultado de estos gráficos con los siguientes ejemplos.

```
> t <- 1:100; V <- sin(t); ts.plot(V); acf(V)
```

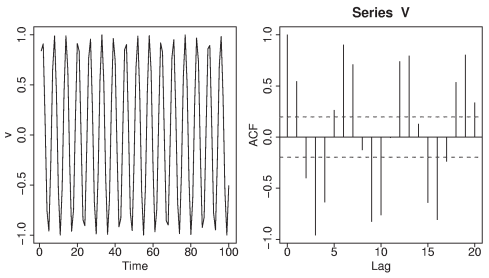


Fig. 45: ACF de una señal senoidal

La serie temporal analizada presenta una clara periodicidad manifiesta tanto en su gráfico temporal como en la periodicidad de los valores de su acf. Presenta además multitud de valores significativos, (aquellos que sobrepasan los umbrales horizontales de línea discontinua), lo que revela la existencia de estructura temporal remanente.

Capítulo 6

Clasificación de las observaciones en familias

La clasificación de las observaciones en *clusters*, o clases, permite separar aquellas observaciones que entre sí muestran una naturaleza diferenciada de tal forma que cada clase presente máxima homogeneidad interna y máxima heterogeneidad externa respecto al resto de las clases.

La clasificación se basa únicamente en la estructura de las observaciones y de su "proximidad" relativa. Debe distinguirse la *clasificación* del agrupamiento previo que, en base a un conocimiento tecnológico del proceso, como en este caso la suposición de la existencia de distintas clases de comportamiento atendiendo al grado de acero, pueda realizarse de antemano.

Las principales herramientas de clasificación se basan en algoritmos basados en técnicas estadísticas o se aprovechan de la capacidad de las redes neuronales de clasificación no supervisada.

6.1. Métodos Estadísticos

6.1.1. *Fundamentos*

Principalmente existen dos tipos de algoritmos: los basados en la "aglomeración jerárquica" y los que utilizan técnicas de "reubicación iterativa".

En la aglomeración jerárquica se comienza con tantas clases como observaciones. Las clases semejantes son aglutinadas en una única clase. De esta forma se genera sucesivamente un número cada vez inferior de clusters que aglutinan las observaciones semejantes entre sí. **R** implementa este algoritmo a través de la función *hclust*. Este método resulta más adecuado para clasificar variables cualitativas o de tipo categórico.

La reubicación iterativa parte de un número determinado de clusters en los que intenta optimizar la semejanza interna según criterios de distancias. La función correspondiente en **R** es *kmeans*., así como también la función más general de clasificación *cclust*, que por defecto utiliza el algoritmo *kmeans*.

6.1.2. *Implementación en R*

```
> dcclust ← cclust(datos.nuevos, mean(datos.nuevos))
> dcclust
```

```
                Clustering on Training Set
Number of Clusters:  9
Sizes  of Clusters:  1050 1595 662 1943 2132 2785 1249 641 2885
Algorithm converged after 71 iterations.
Changes: 13653 2834 1749 1331 1201 1175 1090 976 870 753 551 469 384
346 304 285 264 245 237 225 212 197 168 147 125 125 131 109 111 106
106 109 97 91 89 82 78 70 61 52 54 51 53 39 23 17 16 16 12 11 7 3 3 3
4 2 2 2 2 4 3 3 4 2 2 2 1 1 2 3 2
```

La clasificación así obtenida (Fig. 46) atiende a los valores de la variable número 3, correspondiente a DR (diferencia de espesores entre pasadas). La representación Sammon coloreada atendiendo a esta clasificación se muestra en la Fig. 47.

```
> subsample <- function(data.in, cuantos)
+ {
+ aux <- cbind(runif(nrow(data.in)),data.in)
+  aux <- as.matrix(aux)
+  ordenado <- order(aux[,1])
+  aux <- as.matrix(aux[ordenado,-1])
+  return(list(A=aux[1:cuantos,],B=aux[(cuantos+1):nrow(aux),]))
+ }
> datos.nuevos.sam <-sammon(subsample(datos.nuevos,2000)$A, 2,
+  maxit=1000, tol=0.001)
```

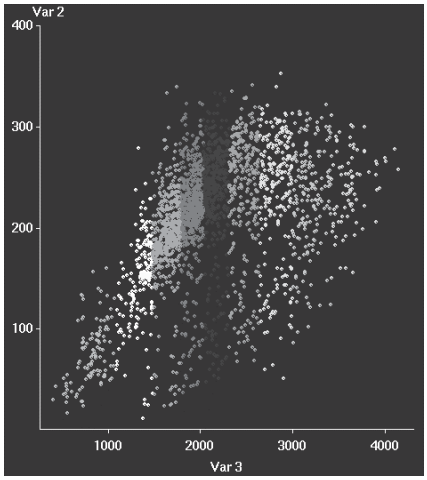


Fig. 46: F vs. DR

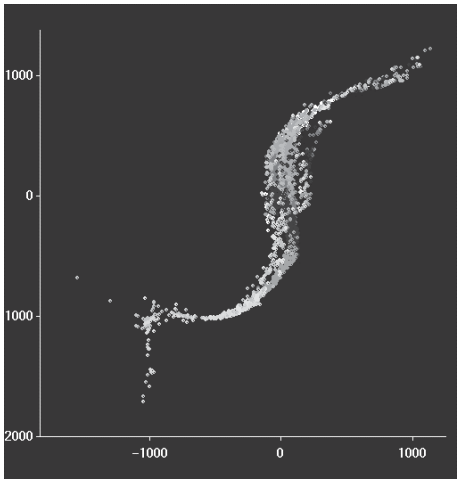


Fig. 47: Proyección Sammon.

En la proyección Sammon de los datos se distinguen distintas zonas correspondientes a los distintas clases generadas. A simple vista, la clasificación obtenida con este criterio, estratifica las observaciones con mayor calidad que la obtenida con la clasificación obtenida con los criterios basados en el contenido en carbono.

6.2. Métodos Neuronales:Redes SOM

Las SOM (self organizing maps o redes neuronales autorganizativas) consisten en un conjunto de algoritmos de proyección no lineal de la función de densidad de un vector de entrada multidimensional sobre una retícula bidimensional discreta.

Puede imaginarse la estructura abstracta como un conjunto de neuronas de entrada con total conexión sobre una retícula neuronal bidimensional. Cada una de las celdas de esta retícula está caracterizada por un vector de pesos habitualmente llamado *codebook*.

En la etapa de aprendizaje, entrenamiento, o de ajuste de pesos, cada vector de entrada es comparado con los *codebooks* de las celdas existentes en la retícula. Se persigue identificar la celda que mayor parecido guarda con el vector de entrada. Esta relación de similitud puede ser, por ejemplo, medida a través del ángulo que forman ambos vectores, con lo que la operación de comparación puede ser realizada simplemente con el producto escalar del vector de entrada y el *codebook* de la celda cuestionada.

Con la etapa de aprendizaje se espera obtener una red neuronal que automáticamente asigne a cada vector de entrada dos coordenadas (x, y) correspondientes a los índices de una celda concreta dentro de la estructura con la que la función de similitud se hace máxima.

De esta manera se cubriría el objetivo de clasificación de las distintas observaciones.

La clasificación de los patrones por vía neuronal la hemos realizado con el paquete `SOM-PACK`[4] , desarrollado por un equipo de la universidad tecnológica de Helsinki.

6.2.1. *Implementación*

Generamos a continuación los ficheros de datos con los que trabajarán los algoritmos ejecutables del `SOM-PACK`.

```
> aux   <- subsample(datos.nuevos, 10000)
> train <- aux$A
> test  <- aux$B
> sink("datos.nuevos.som")
> datos.nuevos[,1:6]
> sink("train.som")
> train
> sink("test.som")
> test
> sink()
```

[4] Disponible en http://cochlea.hut.fi/research/som_lvq_pak.shtml

Con lo que se obtienen dos ficheros con un formato como el siguiente:

```
    TRRE    F    DR    PLT       R TRCEQ
1    150 1956 15.08 233.82 463.295    14
2    203 1922 15.35 218.74 463.295    14
3    195 1835 12.54 203.39 463.295    14
4    152 1567 24.73 190.85 463.295    14
5    259 1630 21.60 166.12 463.295    14
6    212 1569 21.61 144.52 463.295    14
7    223 1631 21.60 122.91 463.295    14
```

Antes de utilizarlo con el SOM-PACK es necesario eliminar la primera columna, innecesaria en este caso por ser un simple índice, y substituir la primera línea por el número de dimensiones de los vectores de entrada.

```
6
150 1956 15.08 233.82 463.295 14
203 1922 15.35 218.74 463.295 14
195 1835 12.54 203.39 463.295 14
```

```
$ vfind
$ visual -din train.som -dout train.out -cin train.cod
$ visual -din test.som -dout test.out   -cin train.cod
$visual -din datos.nuevos.som -dout datos.nuevos.out
        -cin train.cod
```

En train.out y test.out se encuentran los ficheros con las clases correspondientes a los vectores de los ficheros de entrenamiento y de test.

```
3 hexa 3 3 gaussian
2 1 239.365
2 1 141.919
0 2 74.3517
```

La primera línea del fichero informa de la estructura de la SOM entrenada. En este caso se trata de una estructura de celda con vecindad hexagonal (la alternativa sería una estructura rectangular) de dimensiones 3x3 y cuya función de vecindad es de tipo gaussiano (puede elegirse también una función tipo *bubble* o *step*).

Es necesario antes de cargar los datos en **R,** editar el fichero y eliminar la primera línea de información.

```
> train.out <- read.table("train.out")
> test.out <- read.table("test.out")
> datos.nuevos.out <- read.table("datos.nuevos.out")
```

La indexación del array de neuronas de la SOM comienza con 0. En **R** los arrays comienzan su indexación con 1. Por tanto habrá que sumar una unidad a las dos primeras columas de train.out y test.out.

```
> train.out[,1:2] <- train.out[,1:2]+1
> test.out[,1:2] <- test.out[,1:2]+1
> datos.nuevos.out <- datos.nuevos.out[,1:2] + 1
```

Para representar en **R** las clases formadas utilizando distintos colores en su representación se crea una función que asigna un código de color a cada clase.

```
> matriz.colores <- matrix(c("black","blue","yellow","red",
+           "cyan","white","green3","magenta","white"), ncol=3)
> matriz.colores
     [,1]      [,2]      [,3]
[1,] "black"   "blue"    "yellow"
[2,] "red"     "cyan"    "white"
[3,] "green3"  "magenta" "white"
> train.colores <- matriz.colores[
+            cbind(train.out[,1], train.out[,2])]
> test.colores <- matriz.colores[
+            cbind(test.out[,1], test.out[,2])]
> datos.nuevos.som.colores <- matriz.colores[
+            cbind(datos.nuevos.out[,1], datos.nuevos.out[,2])]
```

En la Fig. 48 se muestra los gráficos correspondientes a las clasificaciones obtenidas con las técnicas SOM para los vectores de entrenamiento y de test. Se observa que la SOM ha sido capaz de generalizar y clasifica los patrones de test conforme al mismo criterio que el utilizado para los patrones de entrenamiento. El criterio utilizado parece ser predominantemente los distintos niveles de fuerza de apriete entre rodillos de laminación.

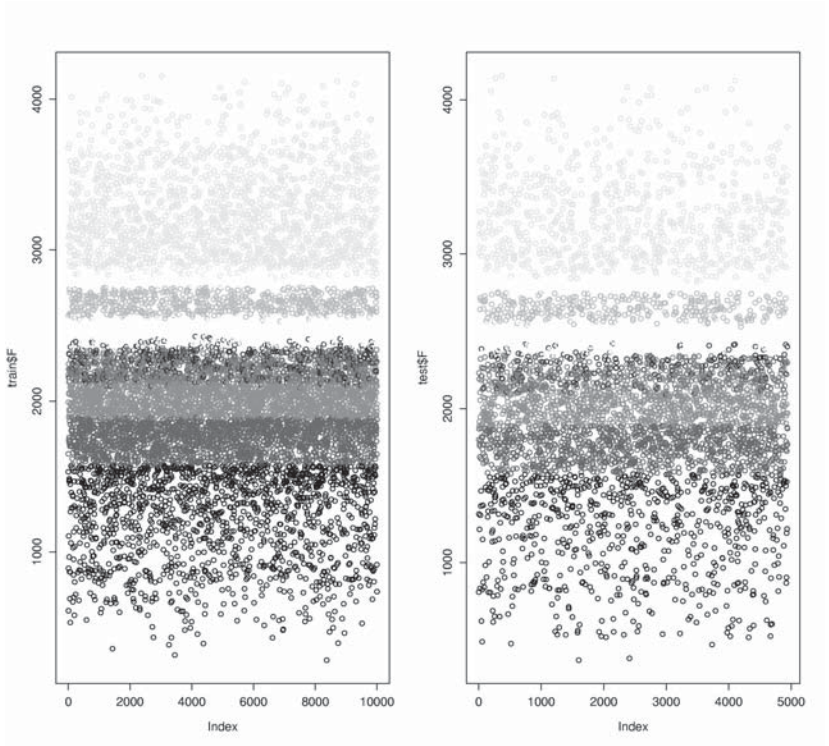


Fig. 48: Valores de fuerza coloreadas según clasificación realizada por técnicas SOM.

6.3. Conclusiones

Los datos del proceso han sido clasificados por dos vías algorítmicas distintas. Ninguna de ellas ofrece resultados que se ajusten al criterio hasta ahora utilizado en el que se clasificaban las observaciones atendiendo al contenido en carbono. Las clasificaciones obtenidas utilizan criterios basados en el valor de otras variables, como la reducción de espesor entre pasadas; en el caso del algoritmo k-means; y el valor de la fuerza de apriete entre rodillos; caso de la clasificación por SOM.

Capítulo 7

Test sobre la existencia de distintas clases

El esfuerzo necesario para verificar la normalidad múltiple se ve justificado a la hora de comprobar si las clases propuestas, bien por la existencia de conocimiento previo del proceso tecnológico que conduzca a un agrupamiento dirigido, bien por algoritmos de clasificación, resultan válidas.

Esta validez vendrá determinada, necesariamente, por la comprobación de la suficiente heterogeneidad entre clases que apunte hacia el rechazo de la hipótesis nula: "todas las observaciones proceden de una población de idénticos parámetros, independientemente de que pertenezcan a una presunta clase o a otra".

Los algoritmos y los métodos de contraste de hipótesis, que sobre la existencia de distintas clases han de aplicarse, marcan como requisito indispensable la verificación y cumplimiento de una serie de condiciones de partida. Entre ellas, quizás la más importante y la que los diferencia en dos tipos claros de métodos, es la exigencia de trabajar con observaciones procedentes de muestras normales multivariantes. Este es uno de los factores que marcan la importancia del análisis previo de normalidad multivariante. La normalidad multivariante, o su ausencia marcan los métodos y algoritmos a aplicar en la detección de casos atípicos y la validación de las clases de comportamiento propuestas por los algoritmos de clasificación.

En el caso de trabajar con muestras normales univariantes lo habitual es utilizar el método ANOVA **[11,** pág 399**]** (análisis de la varianza); implementado en **R** como *anova*; o el contraste de la t de Student **[11]]**; implementado como *t.test*; como caso particular para dos clases, grupos, tratamientos, o como quiera que se vengan denominando los agrupamientos de datos en según que campo de conocimiento trabajemos.

Para muestras normales multivariantes la extensión natural del ANOVA cuando ha de considerarse más de una variable de criterio es el MANOVA **[11] [17]**; (análisis de varianza multidimensional), que se desarrolla en el siguiente punto. Tanto en el ANOVA como el MANOVA es necesario comprobar la homogeneidad de la matriz de covarianzas de la muestra, por ser ésta, así como la normalidad de la muestra, una de las suposiciones de trabajo de ambos algoritmos.

En el caso de muestras no normales univariantes existen alternativas al ANOVA, como el test no paramétrico de Kruskal-Wallis **[11,** pág 425**]**. En el caso de considerar únicamente dos muestras es de aplicación asimismo el test de Wilcoxon, que presenta ventajas frente a no normalidades y casos atípicos.

7.1. Existencia de clases en muestras normales multivariantes

7.1.1. *MANOVA*

El análisis MANOVA supone que las distribuciones de cada uno de los grupos son normales multivariadas y con misma varianza, cuestionándose como hipótesis nula si el valor del vector de medias puede considerarse común para todos los grupos.

En el ANOVA de un factor se comparaba la ubicación de las medias de distintas distribuciones con varianza común (Fig. 49). El MANOVA de un factor puede verse como una comparación de la ubicación espacial de los centroides de los elipsoides que forma cada distribución, de idénticas dimensiones (varianza común), pero posibles distintos centros.

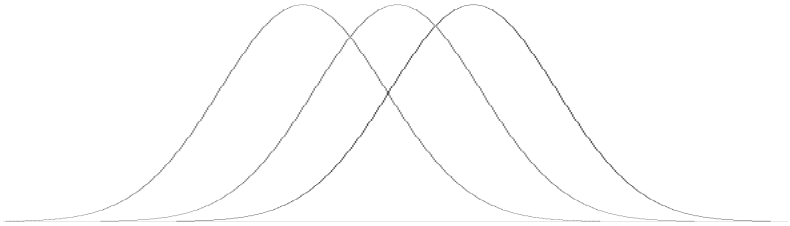


Fig. 49: Distribuciones normales univariantes de varianza común y distinta media

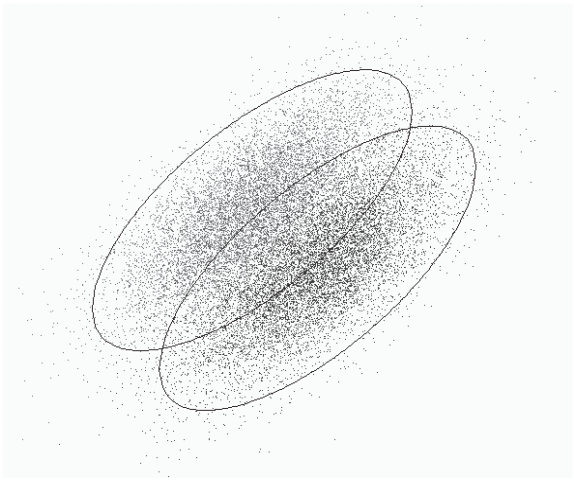


Fig. 50: Distribuciones binormales de varianza común pero distinto vector de medias.

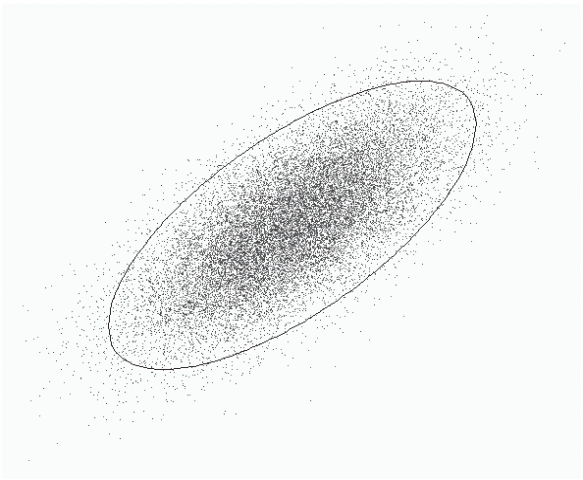


Fig. 51: Distribuciones binormales de varianza y vector de medias común.

La variación de los individuos respecto a la media total puede verse como la contribución de dos factores: la variación entre grupos y la variación dentro de grupos. El valor de la suma de los cuadrados de las diferencias correspondientes a cada uno de estos factores proporciona un indicador de la homogeneidad de los distintos grupos entre sí. En el caso univariado estos valores pueden ser contrastados con una distribución F de Snedecor. Grandes variaciones entre grupos respecto a las variaciones dentro de grupos derivan en grandes valores de F y por tanto el rechazo de la hipótesis nula.

Cada observación puede considerarse como un vector,

$$x_{c_j} = \bar{x} + (\bar{x}_c - \bar{x}) + (\bar{x}_{x_j} - \bar{x}_c)$$

expresando así los distintos términos que lo forman:

$$x_{c_j} = \mu_{global} + \tau_{offset} + \varepsilon_{residuo}$$

Llamando a:

- n : nº de variables.
- g : nº de clases.
- n_c : nº de observaciones disponibles en cada clase.
- N : nº total de observaciones.
- j : Indice de ordenación de las observaciones.
- c : Indice de ordenación de las clases.
- S_j : Matriz de covarianzas muestral de la clases j .
- $s = N - g$: Grados de libertad de la matriz de dispersión residual.
- $t = g - 1$: Grados de libertad de la hipótesis.

Si los distintos grupos deben tener un vector de medias común, entonces la hipótesis a contrastar es $H_0 \Rightarrow \bar{\tau}_1 = \bar{\tau}_2 = \ldots = \bar{\tau}_g = \vec{0}$.

La suma de los cuadrados dentro del grupo se define como

$$W = \sum_{c=1}^{g} \sum_{j=1}^{n_c} (x_{c_j} - \bar{x}_c)(x_{c_j} - \bar{x}_c)^T$$

esta matriz puede asimismo expresarse como la suma ponderada de las matrices de covarianzas de cada grupo

$$W = (n_1 - 1)S_1 + (n_2 - 1)S_2 + \ldots + (n_g - 1)S_g$$

La suma de cuadrados dentro de cada grupo se define como

$$W = \sum_{c=1}^{g} n_c (\bar{x}_c - \bar{x})(\bar{x}_c - \bar{x})^T$$

La matriz de dispersión total es $T = B + W$.

La matriz $C = \frac{W}{s}$ proporciona un estimador de la matriz común de covarianzas bajo la hipótesis de homogeneidad de la matriz de covarianzas.

La matriz W sigue una distribución de Wishart[5] **[11] [12]**, $W_n(C,s)$. La matriz B sigue una distribución de Wishart $W_n(C,t)$ y por tanto el estadístico $\Lambda = \frac{|W|}{|W+B|}$ sigue una distribución de lambda de Wilk's, $\Lambda(n,s,t)$.

Para un número de observaciones suficientemente grande puede utilizarse la aproximación asintótica de Bartlett:

$$\left[-\left(n - 1 - \frac{p+g}{2} \right) \ln\left(\frac{|W|}{|W+B|} \right) \right] < \chi^2_{p(g-1)}(\alpha)$$

si bien, la aproximación de Rao proporciona una mejor aproximación para valores pequeños de g .

$$v = \frac{ab - 2c}{nt} \frac{1 - \Lambda^{\frac{1}{b}}}{\Lambda^{\frac{1}{b}}}$$

[5] Si $X_{(N \times n)}$ es una muestra aleatoria correspondiente a una normal multivariante $N(0,\Sigma)$, entonces $X^t X$ sigue una distribución de Wishart de dimensión n , $W_n(\Sigma, N)$, con N grados de libertad.

con

- $a = s + t - \frac{n+t+1}{2}$
- $b = \sqrt{\frac{n^2 t^2 - 4}{n^2 + t^2 - 5}}$
- $c = \frac{nt - 2}{4}$

Así definida, v sigue una distribución $F(nt, ab - 2c)$.

Puede ser de utilidad calcular el valor de un coeficiente de correlación al igual que en el caso univariado, como indicador de la fortaleza de la ligazón existente entre los distintos grupos. En el caso del MANOVA puede considerarse la definición de este coeficiente como el valor de $(1-\Lambda)$. De esta forma se valoraría la relación existente entre la suma de cuadrados dentro del grupo respecto a la suma de cuadrados total. A mayor valor de $(1-\Lambda)$, menor será la homogeneidad de los grupos. Sin embargo esta medida así considerada es sesgada **[12,** pag 218**].** Una estimación aproximadamente insesgada es

$$w_c^2 = w^2 - \frac{n^2 + t^2}{3N}(1 - w^2)$$

con $w^2 = 1 - \frac{N\Lambda}{s + \Lambda}$.

En conjunción con el MANOVA, puede utilizarse el MANCOVA (análisis multivariante de la covarianza) para eliminar el efecto (comprobado con el MANOVA) de cualquier variable independiente no controlada sobre la respuesta del sistema estudiado.

7.1.2. *Implementación en **R**.*

Puesto que los datos reales no siguen una distribución normal multivariante, se emplearán otros datos para mostrar la implementación del código necesario.

```
# Se define la función determinante …
> det <- function(x, method = c("qr","eigenvalues"))
+ {
+ if(!is.matrix(x))
+ stop("x debe ser de tipo matrix")
+ method <- match.arg(method)
+ if(method == "qr")
+ prod(diag(qr(x)$qr)) *(-1)^(ncol(x)-1)
```

```
+ else ## if(method == "eigenvalues")
+ Re(prod(eigen(x, only.values=TRUE)$values))
+ }

# Se define la función que evalúa el Test comentado.
> TestOmega ← function (datos.in)
+  {
+   numclases ← length(datos.in)
+   numvars ← ncol(datos.in[[1]])
+   mediaQ ← HazLista(numclases, "Clase" )
+   for (i in 1:numclases)
+    mediaQ[[i]] ← apply(datos.in[[i]],2,mean)
+ SQ ← lapply(datos.in,var)
+ W ← SQ[[1]]*(nrow(datos.in[[1]])-1)
+ for (i in 2:numclases)
+   W ← W + SQ[[i]]*(nrow(datos.in[[i]])-1)
+   mediaTotal ← mediaQ[[1]]*nrow(datos.in[[1]])
+  for (i in 2:numclases)
+    mediaTotal ← mediaTotal + mediaQ[[i]]* nrow(datos.in[[i]])
+    mediaTotal ← mediaTotal / sum(unlist(
+      lapply(datos.in,nrow)))
+ B ← nrow(datos.in[[1]]) * as.matrix(mediaQ[[1]] - mediaTotal)
+   %*% t(as.matrix(mediaQ[[1]] - mediaTotal))
+ for (i in 2:numclases)
+  B ← B + nrow(datos.in[[i]]) * as.matrix(mediaQ[[i]] -
+   mediaTotal) %*% t(as.matrix(mediaQ[[i]] - mediaTotal))
+ n ← numvars
+ s ← sum(unlist(lapply(datos.in,nrow))) - numclases
+ t ← numclases -1
+ Lambda ← det(W)/det(B+W)
+ w2 ← 1- (nrow(datos.in[[i]])*Lambda)/(s+Lambda)
+ w  ← w2 - (n^2+t^2)/(nrow(datos.in[[i]])*3)*(1-w2)
+ if ((n^2+t^2-5) != 0)
+   {
+     a ← s + t - (n + t + 1)/2
+     b ← sqrt((n^2*t^2-4)/(n^2+t^2-5))
+     c ← (n*t-2)/4
+     duda ← (a*b - 2*c) /(n*t) * (Lambda^(-1/b)-1)
+     return (list( valor=duda, F=qf(0.99, n*t, (a*b - 2*c)),
+       L=(duda < qf(0.99, n*t, (a*b - 2*c))), w=w))
```

```
+   }
+    else
+   {
+     duda ← -( s - 0.5*(n-t+1))*log(Lambda)
+     return (list( valor=duda, ji2=qchisq(0.99, n*t),
+       L=(duda < qchisq(0.99, n*t)),w=w))
+   }
+}
```

Como ejemplo de uso de la función se consideran dos distribuciones binormales, *clase1* y *clase2*, con distinto vector de medias pero idéntica matriz de covarianzas (Fig. 52).

```
> clase1_v1 <- rnorm(5000, mean=0, sd=1)
> clase1_v2 <- rnorm(5000, mean=0, sd=1)
> clase1    <- cbind(clase1_v1,clase1_v2)
> clase2_v1 <- rnorm(5000, mean=1.5, sd=1)
> clase2_v2 <- rnorm(5000, mean=2, sd=1)
> clase2    <- cbind(clase2_v1, clase2_v2)
# Se representan las dos clases
> plot(rbind(clase1,clase2),
+   col=c(rep("Blue",length(clase1[,1])),
+         rep("Red",length(clase2[,1]))), pch=".")
> vcellipse(apply(clase1,2,mean), cov(clase1), col="Blue")
> vcellipse(apply(clase2,2,mean), cov(clase2), col="Red")
```

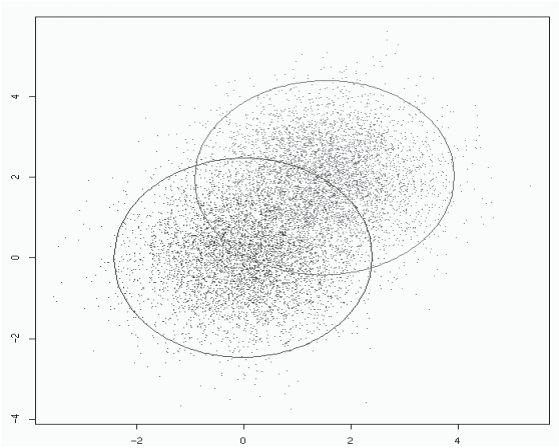


Fig. 52: Ejemplo de datos con dos clases de comportamiento bien diferenciadas.

```
# Se aplica el Test
> dosclases <- list(clase1, clase2)
> Testdosclases ← TestOmega(dosclases)
> unlist(Testdosclases)
       valor          ji2            L            w
9403.4041430    9.2103404        FALSE    0.8047111
```

Puesto que para las distribuciones del ejemplo se cumplen las condiciones de normalidad, este resultado rechaza la hipótesis nula de homogeneidad y por tanto se identifican distintas clases de comportamientos.

7.1.3. *Comprobación de homogeneidad de las matrices de covarianzas*

7.1.3.1. Homogeneidad de la matriz de covarianzas para muestras normales

En el MANOVA se parte de la suposición de que los grupos que forman las muestras siguen distribuciones de tipo normal multivariante y que la matriz de covarianzas es común a todos los grupos. Parece pues razonable no sólo comprobar la normalidad de las muestras sino también la bondad de la suposición de igualdad de matriz de covarianzas para cada caso concreto. Esto es particularmente necesario cuando las distintas clases recogen un número muy dispar de muestras.

Se debe comprobar la hipótesis nula $H_0 \Rightarrow \Sigma_1 = \Sigma_2 = \ldots = \Sigma_g = \vec{0}$. Bajo la hipótesis de normalidad multivariante de la muestra se define **[12**, pág. 221**]**.

$$M = \frac{\prod_{k=1}^{g} |S_k|^{\frac{n_c-1}{2}}}{|\overline{S}|^{\frac{s}{2}}}$$

con $\overline{S} = \frac{1}{N-g}\sum_{c=1}^{g}(n_c-1)S_c$

Existen dos aproximaciones asintóticas a la distribución de M.

- $-2(1-c_1)\ln(M)$ es aproximadamente $\chi^2_{\frac{1}{2}n(n+1)}(\alpha)$ con

$$c_1 = \frac{2n^2+3n-1}{6(n+1)t}\left[\sum_{k=1}^{g}\frac{1}{n_c-1} - \frac{1}{s}\right]$$

- $-2b\log_{10}(M)$ sigue aproximadamente una distribución F con v_1 y v_2 grados de libertad, donde …

  - $b = \frac{1-c_1-\frac{v_1}{v_2}}{v_1}$

- $c_1 = \dfrac{2n^2+3n-1}{6(n+1)t}\left[\sum_{k=1}^{g}\dfrac{1}{n_c-1}-\dfrac{1}{s}\right]$
- $v_1 = \dfrac{1}{2}n(n+1)t$
- $v_2 = \dfrac{v_1+2}{\left|c_2-c_1^2\right|}$
- $c_2 = \dfrac{(n-1)(n+2)}{6t}\left[\sum_{k=1}^{g}\dfrac{1}{(n_c-1)^2}-\dfrac{1}{s^2}\right]$

Si $c_2 - c_1 < 0$, entonces $\dfrac{-2b_1 v_2 \ln(M)}{v_1 + 2b_1 v_1 \ln(M)}$ sigue, aproximadamente, una distribución F con v_1 y v_2 grados de libertad, con $b_1 = \dfrac{1-c_1-\dfrac{2}{v_2}}{v_2}$. La aproximación F suele ser mejor que la χ^2.

Alternativamente, bajo condiciones de normalidad y para muestras univariantes, se puede contrastar la hipótesis de homogeneidad de la varianza utilizando el test de Bartlett, implementado en **R** dentro de la librería *ctest* como *bartlett.test()*. En el test de Bartlett **[11,** pág. 427**]**, bajo la hipótesis nula, el estadístico $\dfrac{M}{c+1}$ sigue una distribución χ^2 con $(g-1)$ grados de libertad para muestras grandes, donde

$$M = \sum_{j=1}^{g}(N_j - 1)\ln\left(\frac{s^2}{s_j^2}\right)$$

$$c = \left(\frac{1}{3(g-1)}\right)\left[\sum_{j=1}^{g}\left(\frac{1}{n_j-1}\right)-\frac{1}{\sum_{j=1}^{g}(n_j-1)}\right]$$

recordemos que $s_j^2 = \frac{1}{N_j - 1}\sum_{i=1}^{n_j}\left(y_{ij} - \bar{y}_{\cdot j}\right)^2$ es la varianza muestral de la clase j y $s^2 = \frac{1}{N-g}\sum_{j=1}^{g}\left[\left(N_j - 1\right)s_j^2\right]$ es la varianza muestral conjunta.

Asimismo, para este test puede utilizarse un estadístico que sigue, aproximadamente, una distribución F de $(g-1)$ y $d = \frac{(g+1)}{c^2}$. Este es

$$F = \frac{dM}{(g-1)\left(\frac{d}{1-c+\frac{2}{d}} - M\right)}$$

Adicionalmente, se puede comentar que en el caso de clases con idéntico número de observaciones, se pueden emplear el test de Harley **[12,** pág. 427**]** y el test de Cochran **[12,** pág. 428**].** Ambos trabajan bajo condiciones de normalidad y están limitados al caso univariante.

7.1.3.2. Implementación en **R**

Puesto que los datos reales no siguen una distribución normal multivariante, se emplearán las distribuciones del ejemplo de la Pág. 103 para mostrar la implementación del código necesario.

```
 > n <- 2             # 2 variables
> numclases <- 2      # 2 clases
> nc <- as.vector(unlist(lapply(dosclases,nrow)))
     #nc -> número de observaciones por clase
> s <- sum(nc)-numclases
> S <- lapply(dosclases,cov)
> Smedia <- S[[1]]*(nc[1]-1)
> for (i in 2:numclases)
+   Smedia <- Smedia + S[[i]]*(nc[i]-1)
> Smedia <- Smedia/s
> LnM <- log(det(S[[1]]))*((nc[1]-1)/2)
> for (i in 2:numclases)
+ lnM <- lnM+ log(det(S[[i]]))*((nc[i]-1)/2)
> lnM <- lnM - s/2*log(det(Smedia))
> c1 <- (2*n^2+3*n-1)/(6*(n+1)* (numclases-1))*
+ (sum(1/(nc-1)^2)- 1/s )
> duda <- -2*(1-c1)*lnM
> referencia <- qchisq(0.99, 0.5*n*(n+1)*s)
> duda
```

```
[1] 1185.406
> referencia
[1] 30566.72
 duda < referencia
[1] TRUE
```

El resultado del estadístico (1185.406) es muy inferior al valor límite marcado por la χ^2 (30566.72) para un 99% de confianza, con lo que se acepta la hipótesis de homogeneidad de las matrices de covarianzas y es valido el análisis precedente de la Pág. 103 que presuponía la homogeneidad de las matrices de covarianzas.

Puede resultar más cómodo aplicar el test de Bartlett implementado en **R**. Este test sólo es válido bajo hipótesis de normalidad y aplicable únicamente al caso univariante.

```
> bartlett.test(list(rnorm(5000,0,1), rnorm(5000,30,1.5)))
        Bartlett test for homogeneity of variances
data:  list(rnorm(5000, 0, 1), rnorm(5000, 30, 1))
Bartlett's K-square = 0.0218, df = 1, p-value = 0.8827

> bartlett.test(list(rnorm(5000,0,1), rnorm(5000,30,1.5)))
        Bartlett test for homogeneity of variances
data:  list(rnorm(5000, 0, 1), rnorm(5000, 30, 1.5))
Bartlett's K-square = 895.2381, df = 1, p-value = < 2.2e-16
```

El test de Bartlett distingue correctamente ambas situaciones. Puede reflejarse su sensibilidad en un gráfico.

```
> d1 <- rnorm(1000, 0 , 1)
>  medias <- seq(from=-0.4, to=0.4, by=0.01)
>  varianzas <- seq(from=0.01, to=2, by=0.01)
>  desviaciones <- sqrt(varianzas)
>  m <- m2 <- matrix(1000, nrow=length(medias),
+             ncol=length(desviaciones))
>  for (i in 1:length(medias))
>  for (j in 1:length(desviaciones))
>   {
>    d2 <- rnorm(1000, medias[i] , desviaciones[j])
>    m[i,j] <-  bartlett.test(list(d1,d2))$p.value
>   }
> filled.contour(m,  levels=c(0,0.05,1))
```

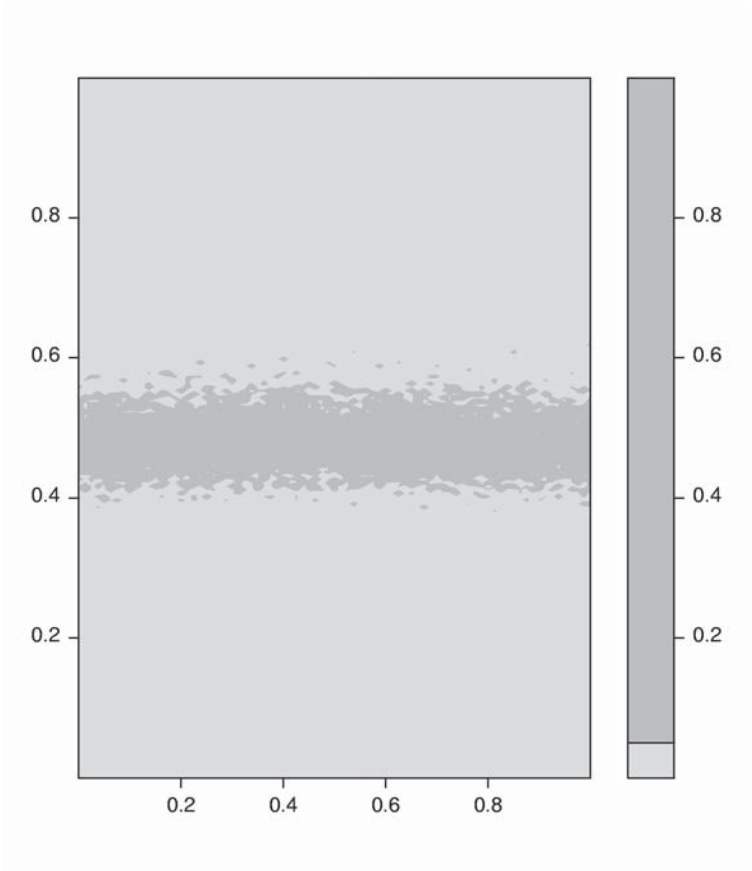


Fig. 53: Análisis de sensibilidad de l test de Bartlett.

7.1.3.3. Homogeneidad de la matriz de covarianzas para muestras no normales multivariantes.

La falta de normalidad en las distribuciones impide la aplicación del MANOVA, y por tanto no parece necesario comprobar la homogeneidad de las matrices de covarianzas, si bien se comentarán los métodos disponibles para distribuciones no normales.

Los tests de homogeneidad de la varianza, para distribuciones normales, comentados son extremadamente sensibles a la falta de normalidad. Gracias a la característica de que el ANOVA para la igualdad de medias es robusto a la no normalidad, varios test de homogeneidad de la varianza en distribuciones no normales hacen uso de esta robustez, como son el método de Levene **[11**, Pág. 428**]** y el método Jackknife **[11**, pág.29**]**. En el caso univariante el test de Fligner-Killeen se ha mostrado como uno de los más robustos frente a carencias de normalidad **[16]**.

7.1.3.4. Implementación en **R**.

El test de Fligner-Killen está implementado en **R** con la función *fligner.test*. Utilizando los mismos datos, del ejemplo anterior, pág. 103.

```
> fligner.test(list(rnorm(5000,0,1), rnorm(5000,30,1)))
        Fligner-Killeen test for homogeneity of variances
data:  list(rnorm(5000, 0, 1), rnorm(5000, 30, 1))
Fligner-Killeen:med chi-square = 2e-04, df = 1, p-value = 0.9882

> fligner.test(list(rnorm(5000,0,1), rnorm(5000,30,1.5)))
        Fligner-Killeen test for homogeneity of variances
data:  list(rnorm(5000, 0, 1), rnorm(5000, 30, 1.5))
Fligner-Killeen:med chi-square = 618.5719, df = 1, p-value = < 2.2e-16
```

7.2. Existencia de clases en muestras no normales multivariantes.

En el caso de que no se satisfagan las condiciones de normalidad o de homogeneidad la metodología anteriormente expuesta no resulta válida y deben usarse otras alternativas no paramétricas, como por ejemplo el test de Kruskal-Wallis **[11**, pag 425**]** que utiliza el rango de las observaciones. Esta función está implementada en **R** como *kruskal.test()*. Al ser un test univariante se deberá aplicar sobre cada componente de forma independiente y no sobre la distribución multivariante de forma conjunta.

El test sobre las g clases se lleva a cabo a través del estadístico $T = \frac{1}{s^2}\left[\sum_{j=1}^{g} R_{\cdot j}^2 - \frac{n(n+1)^2}{4}\right]$, donde $s^2 = \frac{1}{n-1}\left[\sum_{j=1}^{g}\sum_{i=1}^{n_j} R_{ij} - \frac{n(n+1)}{4}\right]$.

Bajo la hipótesis nula el estadístico T sigue una distribución con $g-1$ grados de libertad para muestras grandes. Las comparaciones entre distintos grupos pueden llevarse a cabo comparando $\left(\overline{R}_{\cdot j} - \overline{R}_{\cdot l}\right)$ frente a

$$t_{N-g}^{\frac{\alpha}{2}} \cdot \left(\frac{N-1-T}{N-g}\right)^{\frac{1}{2}} \left(\frac{1}{n_j} + \frac{1}{n_l}\right)^{\frac{1}{2}}$$

donde $t_{N-g}^{\frac{\alpha}{2}}$ es el valor de t para el que $P\left(t > t_{N-g}^{\frac{\alpha}{2}}\right) = \frac{\alpha}{2}$

Aun más robusto y preferible que el test de Kruskal-Wallis es el test de Wilcoxon, *wilcox.test*, que presenta mayor robustez frente a la falta de normalidad y a la presencia de *outliers*. La desventaja del test de Wilcoxon es que únicamente tiene aplicación en el caso de dos muestras.

7.3. Implementación en *R*

Dado que los datos obtenidos del proceso siguen una distribución no normal multivariante, son precisamente los test como el de Kuskal-Wallis y el de Wilcoxon, los indicados para comprobar si puede confirmarse la existencia de distintas clases a partir únicamente de los datos observados y sin tener en consideración ningún otro aspecto relativo al conocimiento previo.

En primer lugar se comprueba si existe falta de homogeneidad en los agrupamientos tal y como se proponen según las reglas de conocimiento previo, esto es, atendiendo al grado de carbono del acero.

Dado que las distintas clases se componen de distinto número de muestras, es necesario, para el correcto uso de los algoritmos, equilibrar el número de muestras en cada clase. Para ello se puede usar la función *subsample*, ya implementada anteriormente. Recordemos que cada clase recogía el número de muestras indicado en la Tabla 6. Puede homogeneizarse el número de observaciones a un número predeterminado de ellas, por ejemplo, en este caso, se elige que cada clase recoja 2000 muestras.

Clase nº	1	2	3	4	5	6	7	8	9
Contenido en carbono	(13,20]	(20,24]	(24,28]	(28,32]	(32,36]	(36,40]	(40,44]	(44,48]	(48,63]
Nº observ.	455	1785	2649	2171	1738	2328	2244	473	1099

Tabla 6

```
> numclases <- 9
> aux <- Mclases
> for (i in 1:numclases)
+     aux[[i]] <- subsample( rbind(Mclases[[i]],Mclases[[i]],
+     Mclases[[i]],Mclases[[i]],Mclases[[i]]), 2000)$A
```

Con lo que todas las clases contenidas en *aux,* constan del mismo número de observaciones.

Clase nº	1	2	3	4	5	6	7	8	9
Contenido en carbono	(13,20]	(20,24]	(24,28]	(28,32]	(32,36]	(36,40]	(40,44]	(44,48]	(48,63]
Nº observ.	2000	2000	2000	2000	2000	2000	2000	2000	2000

Tabla 7

Tal y como se encuentran estructurados los datos, resulta de utilidad plantear una función *aplica_kruskal()*, que contrasta la homogeneidad de una variable**; es por tanto un test univariante;** perteneciente a una determinada matriz a lo largo de las clases que componen una determinada lista de datos.

```
> aplica.kruskal <- function(data.in, variable)
+ {
+    kruskal.test( list(   data.in[[1]][,variable],
+    data.in[[2]][,variable],data.in[[3]][,variable],
+    data.in[[4]][,variable],data.in[[5]][,variable],
+    data.in[[6]][,variable],data.in[[7]][,variable],
+    data.in[[8]][,variable],data.in[[9]][,variable]))
+ }

> aplica.kruskal(aux,1)
Kruskal-Wallis chi-square = 121.0692, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,2)
Kruskal-Wallis chi-square = 476.9148, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,3)
Kruskal-Wallis chi-square = 554.4064, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,4)
Kruskal-Wallis chi-square = 852.954, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,5)
Kruskal-Wallis chi-square = 513.2187, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,6)
Kruskal-Wallis chi-square = 17823.51, df = 8, p-value = < 2.2e-16
```

Los resultados muestran que la división en clases según el grado de carbono es coherente y las clases formadas muestran la necesaria heterogeneidad como para que el test de Kruskal-Wallis detecte que no se trata del mismo patrón de comportamiento..

Como ejemplo de la utiización del test de Wilcoxon para dos muestras, puede aplicarse, por ejemplo, al valor de TRRE en las familias 1 y 6.

```
>  wilcox.test(aux[[1]][,1], aux[[6]][,1])
        Wilcoxon rank sum test with continuity correction
data:  aux[[1]][, 1] and aux[[6]][, 1]
W = 1800418, p-value = 4.624e-08
alternative hypothesis: true mu is not equal to 0
```

Los resultados obtenidos con el test de Wilcoxon apuntan en la misma dirección que los ya obtenidos con el método de Kruskal.

Es interesante comporobar los resultados obtenidos por la clasificación obtenida por el algoritmo de clasificación k-means.

```
> aux1 <- split(datos.nuevos[,1], dcclust$cluster)
> aux2 <- split(datos.nuevos[,2], dcclust$cluster)
> aux3 <- split(datos.nuevos[,3], dcclust$cluster)
> aux4 <- split(datos.nuevos[,4], dcclust$cluster)
> aux5 <- split(datos.nuevos[,5], dcclust$cluster)
> aux6 <- split(datos.nuevos[,6], dcclust$cluster)
```

```
> Nclases <- HazLista(9, "Clase" )
> Numvars <- 6
> for (i in 1:numclases)
+    Nclases[[i]] <- cbind(aux1[[i]], aux2[[i]], aux3[[i]],
+                     aux4[[i]], aux5[[i]], aux6[[i]])
```

Nuevamente se debe homogeneizar el número de muestras de cada clase.

```
> aux <- Nclases
> for (i in 1:numclases)
+ aux[[i]] <- subsample( rbind(Nclases[[i]],Nclases[[i]],
+             Nclases[[i]],Nclases[[i]],Nclases[[i]]), 2000)$A
> aplica.kruskal(aux,1)
Kruskal-Wallis chi-square = 7362.704, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,2)
Kruskal-Wallis chi-square = 17776.43, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,3)
Kruskal-Wallis chi-square = 6978.694, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,4)
Kruskal-Wallis chi-square = 10887.57, df = 8, p-value = < 2.2e-16
> aplica.kruskal(aux,5)
Kruskal-Wallis chi-square = 49.2898, df = 8, p-value = 5.594e-08
> aplica.kruskal(aux,6)
Kruskal-Wallis chi-square = 140.9727, df = 8, p-value = < 2.2e-16
```

Los resultados demuestran que la división en clases propuesta por el algoritmo *kmeans()* también es coherente y las clases formadas guardan suficiente heterogeneidad.

Puede ser interesante abundar en la sensibilidad de estos test frente a las distribuciones evaluadas.

```
> d1 <- rnorm(1000, 0 , 1)
> medias <- seq(from=-0.4, to=0.4, by=0.01)
> varianzas <- seq(from=0.01, to=2, by=0.01)
> desviaciones <- sqrt(varianzas)
> m <- m2 <- matrix(1000, nrow=length(medias),
+             ncol=length(desviaciones))
> for (i in 1:length(medias))
> for (j in 1:length(desviaciones))
+  {
+    d2 <- rnorm(1000, medias[i] , desviaciones[j])
+    m[i,j] <-  kruskal.test(list(d1,d2))$p.value
+    m2[i,j] <-  wilcox.test(d1,d2)$p.value
+  }
> par(mfrow=c(1,2))
> filled.contour(m,  levels=c(0,0.05,1))
> filled.contour(m2, levels=c(0,0.05,1))
```

En los gráficos resultantes se observa la sensibilidad de ambos test frente a variaciones en la media y en la varianza. El test de Wilcoxon proporciona una banda de aceptación de hipótesis más ancha El parámetro decisivo en la diferenciación entre distribuciones parece ser la media, tal y como se observa en las Fig. 54 y Fig. 55.

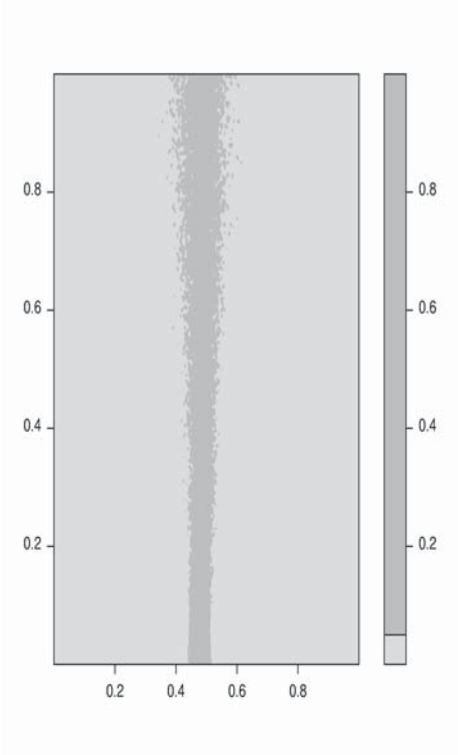


Fig. 54: Mapa de sensibilidad del test de Kruskal-Wallis

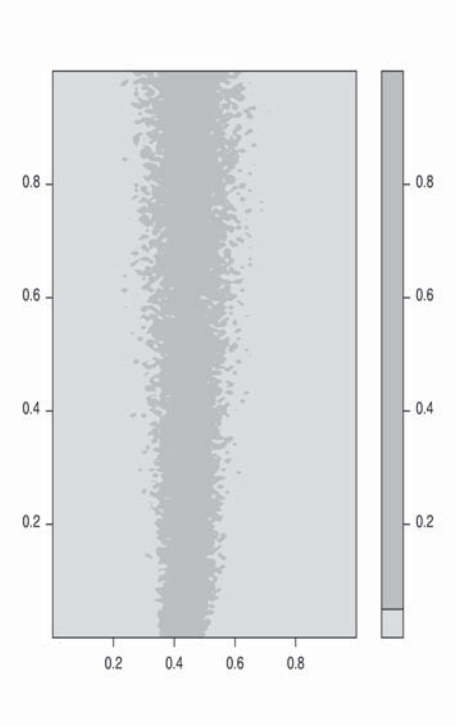


Fig. 55: Mapa de sensibilidad del test de Wilcoxon

7.4. Conclusiones

A pesar de la creencia que hasta ahora se mantenía acerca de la existencia de distintas clases de comportamiento atendiendo al valor del contenido en carbono, se ha comprobado en este trabajo que no son estos los criterios más adecuados para llevar a cabo la clasificación. Según la información extraída de la estructura intrínseca de los datos resulta más adecuada una clasificación realizada por los algoritmos de clasificación que se han mostrado. El test no ha rechazado ninguna de las clasificaciones, pues ambas contenían suficientes características que impedían aceptar la falta de comportamiento grupal. Ambas clasificaciones, por tanto, deben ser aceptadas como válidas, si bien se pone en cuestión su relevancia práctica final en la generación de modelos. El verdadero criterio que permitirá seleccionar la mejor clasificación será la propia

modelización. En la medida en que la propuesta de distintos modelos, atendiendo a diferentes clases de comportamiento, proporcione mejores resultados que aquellos que se obtendrían con un modelo más general e indiferente a estas clasificaciones, se podrá afirmar que determinada clasificación está dotada de verdadero significado práctico. La comparación de los resultados obtenidos a la hora de modelizar con estas clasificaciones revelará que clasificación; la hasta ahora adoptada y basada en valores del TRCEQ, o la obtenida por algoritmos de clasificación dirigidos por la propia estructura de los datos; debe ser considerada más adecuada.

Capítulo 8

Control de calidad sobre la muestra para el modelizado

8.1. Detección de espúreos mediante las distancias de Mahalanobis.

Este análisis es válido únicamente para muestras normales multivariantes.

Los espúreos (también llamados comúnmente *"outliers"*) son aquellas muestras cuya disposición espacial resulta sensiblemente extraña frente al comportamiento general del conjunto. Barnett y Lewis definieron al outlier como "una observación (o subconjunto de observaciones) que parecen ser inconsistentes con el resto de ese conjunto de datos". La aparente inconsistencia puede ser causada por la contaminación de las muestras recogidas por datos procedentes de una distribución de distinta naturaleza, o ser valores extremos de la distribución original. Este alejamiento del comportamiento general puede ser debido bien a errores de muestreo, con lo que habría que eliminar dicha observación, o bien puede darse el caso de que tal observación represente un comportamiento del sistema que si bien es raro e infrecuente no puede ser despreciado en los análisis.

El mayor problema que plantea la detección de estos puntos es el enmascaramiento. Este efecto se produce cuando existe un número considerable de observaciones espúreas concentradas en una zona concreta. Parece claro que el algoritmo de identificación tendrá cierta dificultad a la hora de discernir entre las dos situaciones que pueden presentarse; bien puede ser que sean observaciones perjudicadas por ruidos, o bien es posible que realmente esas observaciones, dado su numero, constituyan un grupo característico de comportamiento diferenciado.

El análisis de detección debe ser realizado dentro de cada clase. No puede ser de otra manera dado que las diferencias entre grupos pueden ser apreciables y un punto típico de una clase, con toda probabilidad tenga un comportamiento claramente anómalo frente a otra clase distinta a la suya.

Este estudio fundamentará la detección de espúreos en la distribución estadística que siguen las distancias generalizadas de Mahalanobis. Si X, matriz de observaciones muestrales, sigue una distribución normal multivariante (N observaciones de n componentes), se puede definir la distancia de Mahalanobis de la observación x_j al centro de la muestra como

$$d_j^2 = \left(x_j - \bar{x}\right) S^{-1} \left(x_j - \bar{x}\right)^t$$

Los N valores de las distancias d_j^2 siguen una distribución χ^2 con n grados de libertad. Si una muestra concreta se caracteriza por una distancia superior a la dada por la distribución χ^2 para un determinado nivel de confianza, puede aceptarse la clasificación de esa muestra como espúrea.

8.2. Implementación en *R*

La distribución de los datos del caso real que estamos analizando ha resultado ser no normal. No tiene sentido aplicar el algoritmo que a continuación se desarrolla a estos datos reales y por ello se ha optado por utilizar los datos procedentes del ejemplo, ya visto, de la Pág. 42. Se cumple así el objetivo de mostrar el modo de operación en aquellas distribuciones que cumplan las condiciones de normalidad necesarias para aplicar el algoritmo.

El desarrollo de la detección se desarrolla siguiendo los siguientes pasos:

1. Cálculo del vector de medias de la muestra $\overline{X}$.
2. Cálculo de la matriz de covarianzas S.
3. Cálculo de la distancia de Mahalanobis.
4. Test $\chi_n^2(\alpha)$ sobre las distancias de Mahalanobis.
5. Eliminación de aquellas observaciones que no superen el anterior test y repetición del proceso.
6. Finalmente comprobar con Sammon.

Recordemos la distribución de los datos del ejemplo de la Pág. 28 (Fig. 56). Los puntos A,B,C,D y E eran detectados como posibles espúreos con el Sammon o con los gráficos Box Plot. Veamos si el método propuesto coincide con estos resultados.

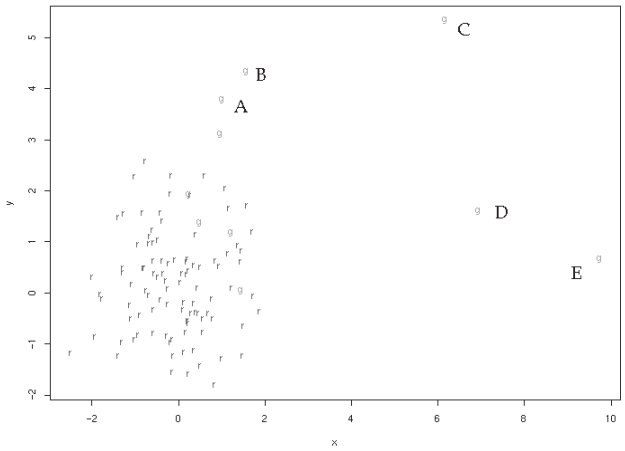


Fig. 56: Representación Bidimensional de las variables *(x,y)* del ejemplo

```
# Los datos del ejemplo se encuentran en la matriz de
# dos columnas y 110 observaciones datos.ejemplo.
# Las primeras 100 obervaciones son normales y se
# representan en rojo. No así las diez últimas que
# aparecen en azul (Array de cadenas de colores color).
# La función implementada admitirá como parámetros de
# entrada una lista donde los datos se recojan clasificados.

>M ← HazLista(1, "Clases")
>M[[1]] ← datos.ejemplo

# Se calculan las distancias de Mahalanobis de los datos.
>CalcDistancias ← function(data.in)
+{
+   longitud ← length(data.in)
+   distancias ← HazLista(longitud, "Clases")
+   for (i in longitud)
+      distancias[[i]] ← Mahalanobis (data.in[[i]],
+        apply(data.in[[i]], 2,mean), cov(data.in[[i]]))
+   return(distancias)
+ }

> plot(CalcDistancias(M)[[1]],col=color)
> referencia  <- qchisq(0.99, 2)
> abline(h=referencia, col="Green")
```

La línea horizontal verde de la Fig. 57 indica el valor de referencia para distancias que sigan una $\chi_2^2(0.99)$. Los puntos por encima de esta línea son espúreos. Se ve claramente que se detectan cuatro puntos como espúreos. Estos puntos se corresponden con los B, C, D y E. El punto A habría sido declarado espúreo si hubiésemos utilizado un nivel de confianza del 95%, en vez del 99%, como se ve en la Fig. 58 donde se han representado las elipses. En ella podemos ver no sólo los puntos de la distribución, sino también el lugar geométrico de los puntos que se encuentra dentro el margen de confianza del 95% y del 99%.

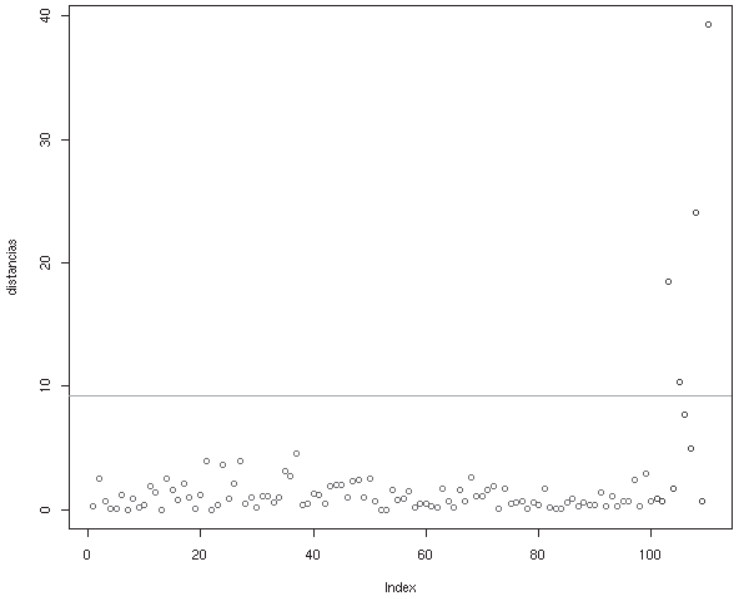


Fig. 57: Distancias de Mahalanobis de los datos del ejemplo.

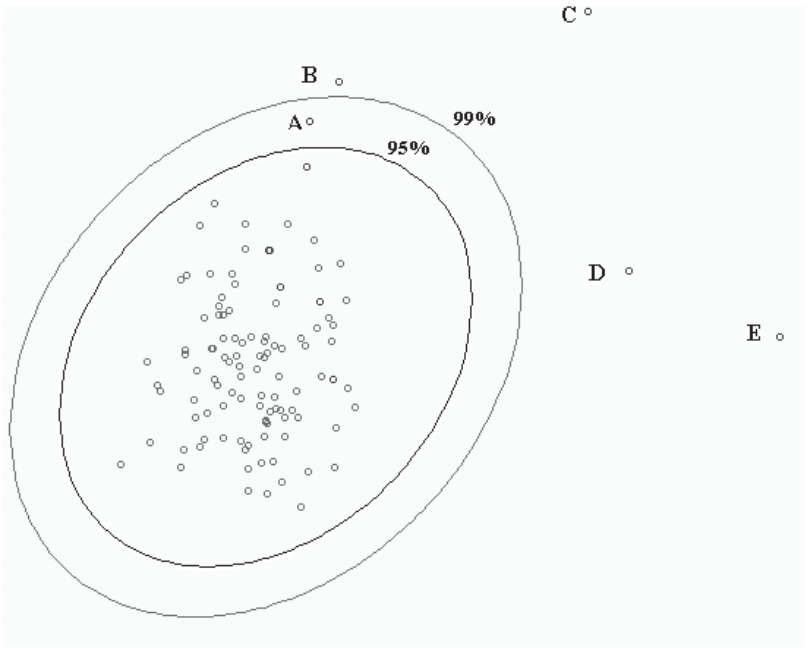


Fig. 58: Elipses del 95% y 99% de confianza.

Este método no detecta únicamente aquellos puntos que se detectarían visualmente con el Sammon, sino también aquellos que aun siendo espúreos, aparecen confundidos entre la nube de puntos del resto de observaciones en la proyección bidimensional Sammon. Por ejemplo, si dotamos de cierta profundidad (1 unidad) a uno de los puntos de la muestra, por ejemplo a la observación nº 50, el gráfico Sammon apenas cambia, y no es capaz de mostrar esta observación como espúrea. Sin embargo el criterio de las distancias de Mahalanobis lo distingue claramente (Fig. 59).

```
> datos.ejemplo2 <- cbind(datos.ejemplo,0)
> datos.ejemplo2[50,3] <- 1
> M[[1]]<- datos.ejemplo2
> plot(CalcDistancias(M)[[1]],col=color);
> abline(h=qchisq(0.99,3),col="Green")
```

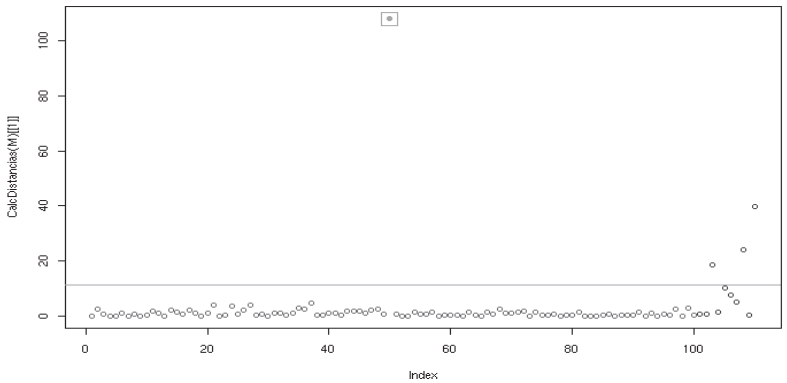


Fig. 59: Distancias de Mahalanobis de los datos del ejemplo modificados.

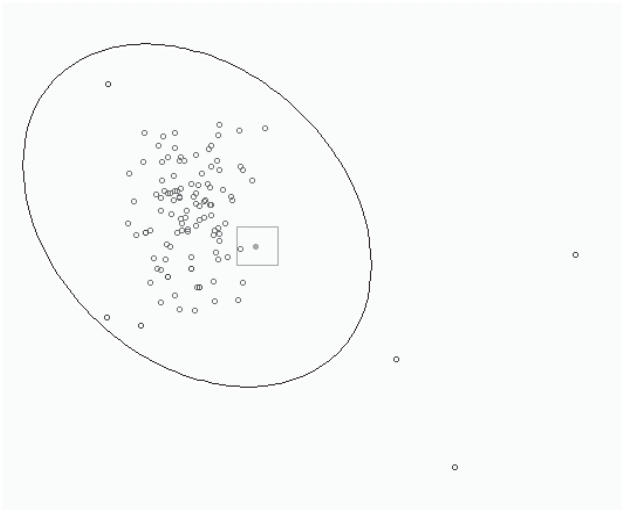


Fig. 60: Elipse del 99% de confianza.

```
> media ← HazLista(9, "Clase")
> Dx ← Mclases
> for (j in 1:numclases)
+   for (i in 1:numvars)
+     DX[[j]][,i] ← ( Mclases[[j]][,i]- media[[j]][[i]])

#Cálculo de varianzas en cada clase.
> S ← HazLista(9, "Clases")
> for (j in 1:numclases)
+    S[[j]] ← var(d[[j]]$rproj)

# Cálculo de las distancias de Mahalanobis.
# Podría haberse usado la función mahalanobis(), como en otras
# ocasiones.
> distancias ← list(C1=matrix(ncol=1),C2=matrix(ncol=1),
> C3=matrix(ncol=1),C4=matrix(ncol=1),C5=matrix(ncol=1),
> C6=matrix(ncol=1),C7=matrix(ncol=1),C8=matrix(ncol=1),
> C9=matrix(ncol=1))

> invS ← S
> for (j in 1:numclases)
+  {
+   invS[[j]] ← solve(S[[j]])
+   for (i in 1:nrow(DX[[j]]))
+    {
+     distancias[[j]][i] ← t(as.matrix(pca.DX[[j]]$rproj[i,]))
+        %*% invS[[j]] %*% as.matrix(pca.DX[[j]]$rproj[i,])
+    }
+ }
```

Se podría implementar una función en **R** para eliminar visualmente los puntos indeseados. Puede ser interesante realizar esta eliminación interactiva para tener, en un principio, una idea más aproximada del proceso que se está llevando a cabo.

```
> elimina ← function(datos.in, nivel)  {
+     plot(datos.in); curve(0*x+nivel,add=T)
+     malos ← identify(datos.in)
+     datos.out ← datos.in[-malos]
+     return (list(nuevo=datos.out,malos=malos))
+  }
```

Sin embargo resulta más cómodo realizar esta operación de forma transparente al usuario. Si se modifica la función *elimina* añadiendo la función

CualesT que extrae los índices de los elementos que cumplan las condiciones expresadas.

```
> cualesT ← function(tabla)
+{
+   cuales ← c()
+   for (index in 1:length(tabla))
+      if (tabla[index]==TRUE)
+          cuales ← c(cuales, index)
+   return (cuales)
+}
```

se puede modificar el código de *elimina* de esta manera…

```
> elimina ← function(datos.in, nivel)
+  {
+     malos ← cualesT( datos.in >  nivel)
+     if (is.null(malos))
+        return (list(nuevo=datos.in,malos=malos))
+     else
+      {
+        datos.out ← datos.in[-malos]
+        return (list(nuevo=datos.out,malos=malos))
+      }
+  }

> aux ←  list(C1 = list(nuevo=,malos=0),
+ C2=list(nuevo=,malos=0), C3=list(nuevo=,malos=0),
+ C4=list(nuevo=,malos=0), C5=list(nuevo=,malos=0),
+ C6=list(nuevo=,malos=0),  C7=list(nuevo=,malos=0),
+ C8=list(nuevo=,malos=0), C9=list(nuevo=,malos=0))

> referencia ← qchisq(0.99, numvars)
> for (i in 1:numclases)
+  aux[[i]] ← elimina(distancias[[i]],referencia)
```

8.3. Detección de espúreos mediante los autovalores de la muestra

Este análisis es válido únicamente para muestras normales multivariantes

En el análisis de componentes principales se determinaron las variables no correladas entre sí que en mayor medida influyen en el comportamiento de la variable a explicar. De las n dimensiones se consideran habitualmente r, $(r < n)$ componentes que expliquen suficientemente el modelo por ser las más significativas. Para la observación j expresada en sus componentes principales x_{jk}, la suma de los cuadrados de sus componentes principales divididas entre su varianza, $\sum_{k=1}^{n} \frac{x_{jk}^2}{\lambda_k}$, coincide con la distancia de Mahalanobis de la observación al centro geométrico de la muestra. Si se considera la suma parcial de las últimas $(p-r)$ componentes, $\sum_{k=r+1}^{n} \frac{x_{jk}^2}{\lambda_k}$, se determina cuantitativamente la proporción en la que la variación correspondiente a la observación j se distribuye en las últimas componentes. Si para una determinada observación, la variación se distribuye preferentemente entre estas últimas componentes, significa que esta observación es un espureo respecto a la estructura de correlación. Si la variación en una observación está dominada, en su mayor parte, por estas últimas componentes, esto indica que la observación considerada es claramente diferente al resto de observaciones. Se puede hacer un gráfico de control con confianza $(1-\alpha)100\%$ representando el valor $T_j^2 = \sum_{k=r+1}^{n} \frac{x_{jk}^2}{\lambda_k}$ de los autovalores no considerados. En este gráfico, aquellos puntos que quedan por encima del nivel de referencia marcado por el valor $\chi_{n-r}^2(\alpha)$ representan observaciones espurias.

8.4. Implementación en *R*

Puesto que los datos del caso real no siguen una distribución normal multivariante, se muestra la implementación del algoritmo únicamente a modo de ejemplo y sin considerar válidos sus resultados.

```
> MD <- pca(datos.nuevos,2)
> Tj2 ← matrix(ncol=1, nrow=nrow(MD$rproj))
> for ( i in 1:nrow(MD$rproj))
+   Tj2[i] ← sum(MD$rproj[i,5:6]^2/MD$evals[5:6])
> plot(Tj2,pch="*")
> referencia99 ← qchisq(0.01, df=2)
> abline(h=referencia99, col='red',add=T)
```

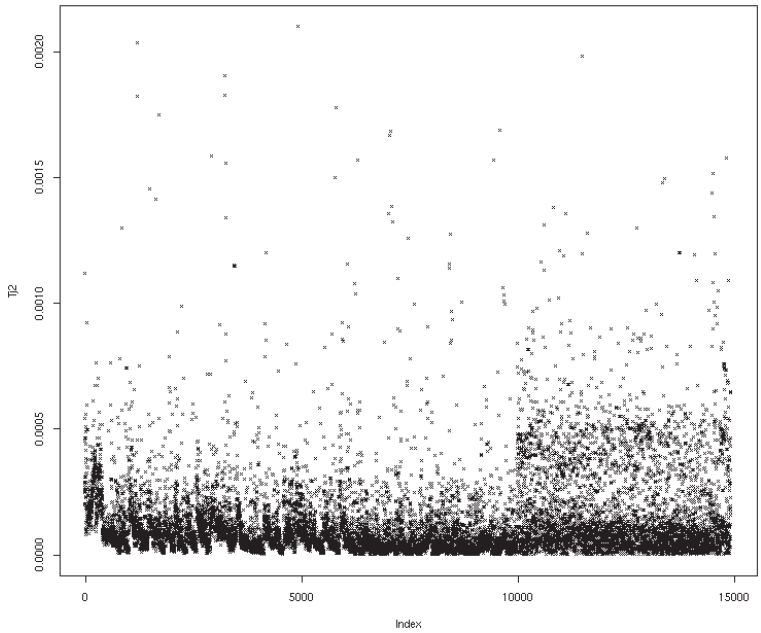


Fig. 61: Gráfico del estadístico T_j^2

En el caso considerado se constataría que al despreciar la contribución de las dos componentes principales menos significativas, ninguna de las observaciones se sale del rango admisible de confianza. La línea de referencia ni siquiera aparece en el gráfico por quedar más allá de los límites de sus ejes (referencia 99 = 0.0201). Esto hubiera confirmado que los datos con los que se trabaja están libres de espúreos.

Es curioso el efecto producido al examinar la misma situación si se considera únicamente la componente número 5, cuya dirección está marcada principalmente por la variable TRCEQ de cuyo valor se derivaban presuntas clases.

```
> for ( i in 1:nrow(MD$rproj))
+   Tj2[i] ← MD$rproj[i,5]^2/MD$evals[5]
> plot(Tj2)
> referencia99 ← qchisq(0.01, df=1)
> abline(h=referencia99, col='red',add=T)
```

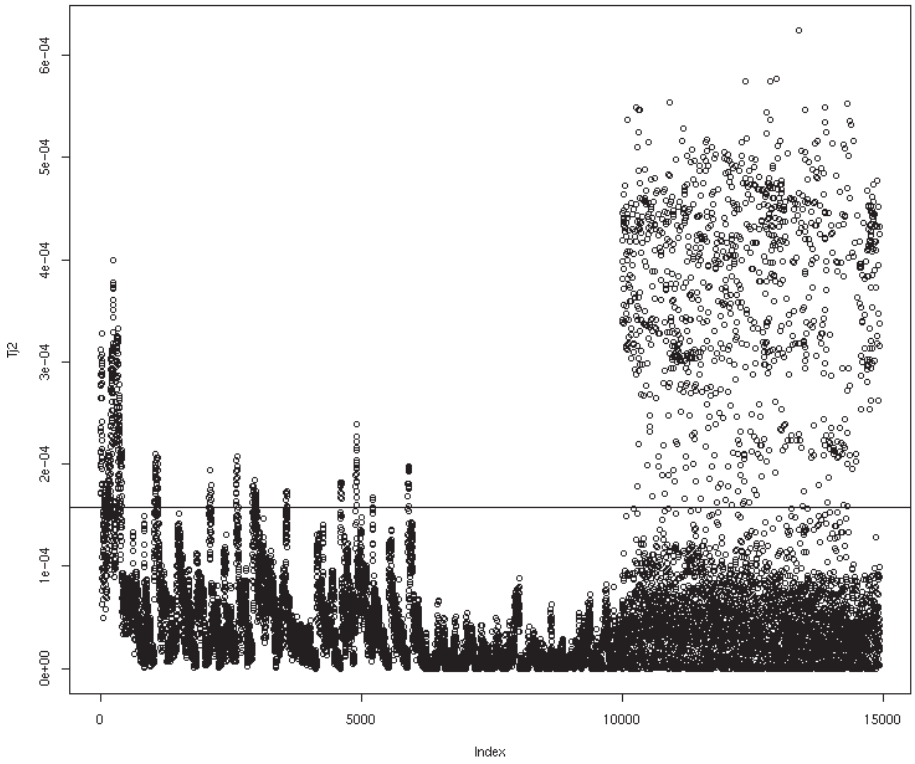


Fig. 62: Gráfico del estadístico T_j^2

En efecto es palpable la brusquedad en el comportamiento de esta variable. Si este análisis fuera válido serían muy numerosas las observaciones que deberían ser consideradas como espúreas. No se tiene en cuenta este resultado, no sólo por la falta de normalidad de la muestra, sino también por estar unida en este caso a un comportamiento categórico que invalida el procedimiento, puesto que los valores que adopta la variable pueden haber surgido de una escala arbitraria de asignaciones numéricas.

Capítulo 9

Cálculo de la Dimensión Fractal

9.1. Fundamentos

Mandelbrot proporciona la siguiente definición de fractal **[11]**

"*Se dice que una figura geométrica o un objeto natural es un fractal si combina las siguientes características*

1. *Sus partes tienen la misma forma o estructura que el conjunto completo, salvo que están a diferente escala y puede que ligeramente deformados.*
2. *Su forma es extremadamente irregular o extremadamente interrumpida o fragmentada, y así se mantiene independientemente de la escala a la que se examine.*
3. *Contiene distintos elementos, cuyas escalas son muy variadas y cubren un amplio rango.*"

Los fractales describen muchos objetos reales que no se corresponden con figuras geométricas simples como puede ser el contorno de una costa o el perfil de un cerebro. Se ha observado que este comportamiento inherentemente fractal es frecuente en la naturaleza.

De la mano del concepto de fractal viene el de su dimensión[6]. No parece adecuado asignar a estos objetos un valor dimensional entero, pues a la hora de movernos por su contorno los grados de libertad tampoco pueden ser caracterizados con números enteros. De una figura como el conocido triangulo de Sierpinsky (Fig. 63) difícilmente se puede calcular su área, pues carece de la compacidad necesaria. La dimensión fractal a asignar a esta figura debe entonces ser un número real entre 1 y 2, y el hecho de se constate que este valor es numéricamente más próximo al 2 que al 1 indica que guarda mayor semejanza con una superficie que con una línea caracterizable por un único parámetro.

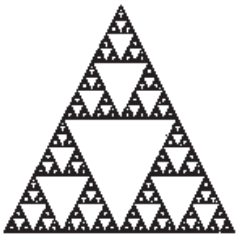


Fig. 63: Triángulo de Sierpinsky

[6] En este estudio se entenderá la dimensión fractal en el sentido de Hausdorff-Besicovich **[9][10]**.

La dimensión euclídea de figuras simples coincide numéricamente con su dimensión fractal, pudiéndose considerar entonces la dimensión fractal como una generalización del concepto que de la dimensión se tiene.

Otros ejemplos de fractales teóricos son el conjunto de Mandelbrot (Fig. 64) y el copo de nieve de Koch (Fig. 65):

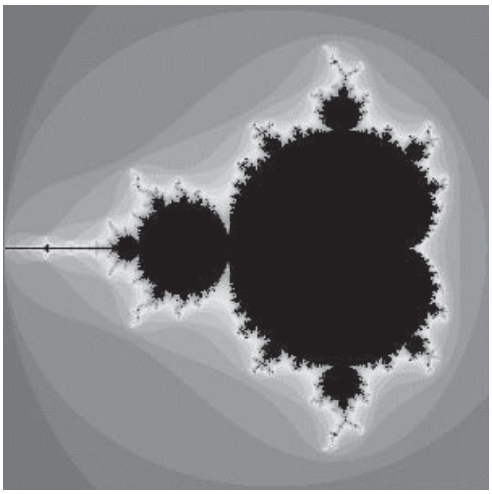


Fig. 64: Conjunto de Mandelbrot.

Fig. 65: Copo de nieve de koch.

De forma numérica, para la nube de puntos discretos que pueden formar un fractal, puede calcularse su dimensión mediante el método de box-counting **[9] [10]**. En este método se parcela el objeto fractal en una cuadrícula[7] que sucesivamente se va dividiendo de forma regular en cuadriculas de escalas sucesivamente inferiores. Si a medida que disminuye el lado de cada uno de esos entornos se considera el cociente existente entre el logaritmo del número de cuadriculas que contienen puntos del fractal en su interior y el logaritmo del lado de la cuadricula, se obtendrá la dimensión fractal. Nótese cierta analogía

[7] El método es válido para dimensión múltiple aun cuando se ilustre al lector con el caso bidimensional.

con el concepto de densidad. La dimensión fractal indica si a medida que cambia la escala con la que se examina el objeto, el numero de puntos existentes en su interior crece de una forma más parecida a una distribución lineal, superficial o hiper-volumétrica, es decir, se trata de presuponer que la densidad de puntos en $\Re^n$ tienen una estructura fractal intrínseca **[18]**.

En el caso de una distribución espacial correspondiente a un registro muestral de información real proveniente de sensores, el cálculo de la dimensión fractal permite estimar la dimensión intrínseca de la distribución espacial de los datos obtenidos, independientemente de la dimensión del espacio vectorial en el que se encuentran embebidos. Esto permite determinar que técnicas son más apropiadas para explicar el modelo subyacente a la estructura de los datos.

El análisis de componentes principales permite eliminar las componentes innecesarias indicando las dimensiones del espacio en el que se embebe la estructura de datos. No obstante la estructura del sistema no tiene necesariamente por qué tener intrínsecamente la dimensión del espacio en el que se inscribe. Una recta en el espacio tridimensional puede ser caracterizada por un único parámetro, una superficie con dos y sin embargo la dimensión completa del espacio que las acoge es tres.

9.2. Implementación en *R*

Para realizar este cálculo es necesario cargar la librería *fdim*[8] y recalcular el PCA al haber eliminado parte de las observaciones en el apartado anterior.

```
> library(fdim)
> datos.pca.nuevo ← pca(datos.nuevos, 2)
```

Se comprueba nuevamente la importancia relativa de cada componente, esta vez definiendo una función...

```
> importancia ← function(datos.in)
+{
+    datos.out ← datos.in/sum(datos.in)*100
+    return(datos.out)
+}
> importancia(datos.pca.nuevo$evals)
[1] 98.4930  1.0574  0.3720  0.0539 0.0217 0.0020
```

Nuevamente se constata como parece adecuado considerar únicamente las dos primeras componentes principales.

```
> datos.fdim ← dfractal(datos.pca.nuevo$rproj[,-6][,-5][,-4][,-3])
```

[8] Módulo de **R** desarrollado por el Área de Proyectos de Ingeniería, Departamento de Ingeniería Mecánica, de la Universidad de La Rioja y puesta a disposición de la comunidad de usuarios de **R**.

```
> datos.fdim
[[1]]
[1] 0

$fdim
      X1
1.595504

$points
  X1         X2
1 -4  -4.954196
2 -7 -10.037547
3 -8 -11.605942
4 -9 -12.796851

$slopeisOK
[1] TRUE

$coefficients
(Intercept)          X1
   1.319893    1.595504

$pointsdif
[1] 14942
$residual
         1          2          3          4
 0.1079260 -0.1889128 -0.1618044  0.2427912

$sumSQRresidual
[1] 0.09098911

$allpoints
  X1         X2
1 -1  -1.000000
2 -2  -2.000000
3 -3  -3.000000
4 -4  -4.954196
5 -5  -6.686501
6 -6  -8.339850
7 -7 -10.037547
8 -8 -11.605942
9 -9 -12.796851

$range
[1] -9 -1

$correlationdim
```

```
      [,1]        [,2]
 [1,]    1  -0.9547211
 [2,]    2  -1.3591849
 [3,]    3  -2.1903188
 [4,]    4  -3.5334847
 [5,]    5  -4.9639740
 [6,]    6  -6.7435327
 [7,]    7  -8.6117325
 [8,]    8 -10.4711406
 [9,]    9 -12.1061815

$informationdim
      [,1]        [,2]
 [1,]    1  -0.9768777
 [2,]    2  -1.5819245
 [3,]    3  -2.4669410
 [4,]    4  -4.0520822
 [5,]    5  -5.5210471
 [6,]    6  -7.3072181
 [7,]    7  -9.1576716
 [8,]    8 -10.9602809
 [9,]    9 -12.4517660
> plot(datos.fdim$points)
> abline(datos.fdim$coefficients,col=2,add=T)
```

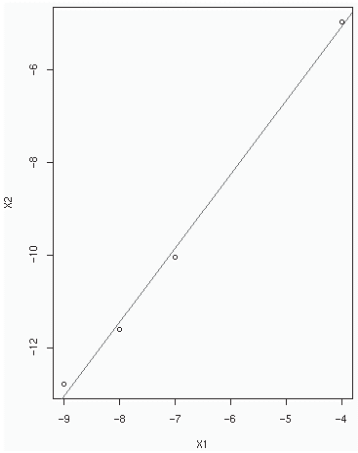


Fig. 66: Evolución del algoritmo fdim.

En la Fig. 66 se representa la evolución del logaritmo del número de celdas ocupadas con puntos de la muestra, frente al logaritmo de la longitud del lado de los sucesivos hiper-cubos en que se va progresivamente subdividiendo el espacio que acoge a la estructura de datos. La dimensión fractal es la pendiente de la recta representada correspondiente a ésta evolución del algoritmo. En este caso es 1.595504. De haber embebido los datos en la dimensión completa

```
> datos.fdim.6d ← dfractal(datos.pca.nuevo$rproj)
```

hubiera resultado un valor de 1.963177, que indicaría también bidimensionalidad.

9.3. Conclusiones

La dimensión deducida en el PCA es 2. La dimensión fractal es también próxima a 2. La estructura de los datos del proceso se corresponde con una variedad 2D “plegada” dentro del espacio n-dimensional original.

Capítulo 10

Ajuste de modelos por técnicas clásicas

Del modelo físico del rodillo de laminación, se puede formular analíticamente el par como:

$$TR = \int_0^{L_d} x \bullet P_v \bullet dx + \int_0^{L_d} y \bullet P_h \bullet dy$$

siendo:

- TR Par de laminación (m·T).
- P_v, P_h Componentes vertical y horizontal de la fuerza (T).
- x, y Distancia vertical y horizontal.
- L_d Longitud de contacto.

Para resolver esta ecuación es necesario obtener la distribución de las componentes vertical y horizontal de la fuerza. Con el ánimo de evitar esta complejidad se supone la fuerza concentrada en el punto neutro en el que la velocidad de cilindro y desbaste son iguales, no hay deslizamiento, pudiendo escribir entonces $TR = F \cdot X$.

donde X es la distancia entre el centro del cilindro y el punto neutro, que puede expresarse también como

$$X = kt \bullet L_d = kt \bullet \sqrt{R \bullet DR}$$

siendo

- kt Constante
- R Radio del cilindro (mm.)
- DR Reducción de la pasada (mm.)

y quedando, en definitiva,

$$TR = a \bullet F \bullet \sqrt{R \bullet DR}$$

donde

$$a = kt_0 + kt_1 * PLT + kt_2 * PLT^2 + kt_3 * PLT^3$$

siendo

- $kt_0 \text{ K } kt_3$ Constantes del modelo

Esta ecuación del par es la que actualmente se utiliza en la empresesa siderurgica en tres tipos de cálculos:

- **Cálculo Previo** del par de cada pasada:

  Esta etapa es previa al inicio de la laminación. La chapa aún no ha comenzado a laminarse y se planifican las actuaciones a realizar sobre ella.

- Cálculo del par **Precalculado** en cada pasada:

  En el precálculo de cada pasada se tiene en cuenta no solo la información procedente de los cálculos realizados en la etapa de cálculo previo, sino que interviene la información de las señales muestreadas en las pasadas anteriores.

- Cálculo del par **Postcalculado** en cada pasada.

  En el postcálculo se modifican nuevamente los valores de las variables explicadas a partir de los valores que realmente se han obtenido realmente en la pasada realizada.

El resultado de la fórmula se multiplica como en el cálculo previo del plan de pasadas por el coeficiente de aprendizaje entre desbastes (obtenido según el valor de la fuerza).

Además se multiplica por un segundo coeficiente, el coeficiente de aprendizaje entre pasadas, obtenido según se está laminando el desbaste en cuestión.

Así pues:

$$TR = TR \cdot CTRP \cdot CTRS$$

El ajuste de modelos por técnicas clásicas puede llevarse a cabo atendiendo a distintos patrones de ajuste. Básicamente se establecen cinco tipos de ajustes. Según la notación de **R**, así como de muchos otros paquetes matemáticos, se distinguen:

- Modelos lineales.
- Modelos de análisis de la varianza.
- Modelos lineales generalizados.
- Modelos aditivos generalizados.
- Modelos de regresión local.

Los modelos lineales de regresión son aplicables a predictores continuos y categóricos que explican variables continuas. Se formulan matemáticamente

como un modelo donde, para cada observación i, $i = 1..N$, el valor y_i de la variable a explicar, se ajusta linealmente a los valores observados de las variables predictoras x_{ij}. El error cometido por este ajuste se absorbe en el residuo ε_i.

$$y_i = \beta_0 + \sum_{j=1}^{N} \beta_j x_{ij} + \varepsilon_i = \hat{y}_i + \varepsilon_i$$

En media, la mejor predicción de la variable a explicar se obtiene mediante una ecuación del tipo

$$y_i = \beta_0 + \sum_{j=1}^{N} \beta_j x_{ij}$$

El siguiente paso a la hora de proponer un modelo más complejo es proponer la posible transformación de la variable a explicar con un modelo lineal generalizado. Estos modelos son aplicables a predictores tanto continuos como categóricos en la explicación de variables tanto continuas como categóricas. Se asume, eso sí, como condición que la varianza de la variable a explicar sea función de su media.

$$\eta(E(y)) = \beta_0 + \sum_{j=1}^{N} \beta_j x_j$$

$$\sigma_y^2 = \phi V(\mu)$$

Los modelos lineales generalizados permiten modelar datos que siguen distribuciones como la normal, binomial, Poisson, gamma y normal inversa, pero aun requieren la relación lineal en los parámetros.

Los modelos aditivos generalizados van más allá y permiten formular relaciones más complejas en las variables predictoras, a través de funciones no lineales.

$$\eta(E(y)) = \sum_{j=1}^{N} f_j(x_j)$$

El tipo de variables predictoras y explicadas es el mismo que para los modelos generalizados.

Si se permita la interacción entre distintas variables se llega a los modelos de regresión local generalizada.

$$y_i = g(x_{i1}, x_{i2}, \ldots, x_{in}) + \varepsilon$$

El tipo de variables predictoras y explicadas es el mismo que para los modelos generalizados y modelos aditivos.

Los modelos clásicos de regresión se basan en hipótesis que pudiera ser no cumplieran los datos, tal es el caso de la condición de normalidad. Además los modelos obtenidos pueden verse desplazados en presencia de outliers. En ese caso es adecuada la utilización de métodos robustos que no se vean afectados por la debilidad de los datos.

10.1. Modelos Lineales

10.1.1. *Fundamentos*

El ajuste de un modelo líneal a unos datos puede enfocarse desde dos puntos de vista.

- Ajuste de mínimos cuadrados.
- Regresión Lineal Multivariada.

Ambos enfoques conducen a un mismo modelo. Se detallará a continuación los fundamentos del ajuste desde el punto de vista de regresión lineal multivariada.

El vector correspondiente a una determinada observación x, de dimensión $(n\times 1)$ puede ser dividido en dos partes $x=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, donde x_1 hace referencia a las primeras q coordenadas de x, y x_2 denota las ultimas $s=(n-q)$ coordenadas. x sigue entonces una distribución cuyas matrices de media y covarianzas pueden ser expresadas como $\mu=\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$ y $\Sigma=\begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$, donde μ_1 es $(q\times 1)$, μ_2 es $(s\times 1)$, Σ_{11} es $(q\times q)$, Σ_{12} es $(q\times s)$ y $\Sigma_{21}=\Sigma_{12}^t$. Así definidas la distribución marginal que sigue x_1 es $N_q(\mu_1,\Sigma_{11})$. La distribución marginal de x_2 es $N_s(\mu_2,\Sigma_{22})$. Asimismo la distribución de x_2 condicionada a $x_1=x_1^*$ es una normal multivariante con vector de medias

$$\mu_{2\cdot 1}(x_1^*)=(\mu_2-\Sigma_{21}\Sigma_{11}^{-1}\mu_1)+\Sigma_{21}\Sigma_{21}^{-1}x_1^*$$

y matriz de covarianzas

$$\Sigma_{22\cdot 1}=(\Sigma_{22}-\Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12})$$

Nótese que el vector de medias condicionado es función de x_1^*, no así la matriz de varianzas condicionada.

La función de densidad de probabilidad x_2 condicionada a $x_1=x_1^*$ es:

$$f\left(x_2 \mid x_1 = x_1^*\right) = (2\pi)^{-\frac{s}{2}} \left|\Sigma_{22\cdot 1}\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(x_2-\mu_{2\cdot 1})' \Sigma_{22\cdot 1}^{-1}(x_2-\mu_{2\cdot 1})}$$

Por tanto la función de regresión multivariada de x_2 en función de x_1 es lineal de la forma $\mu_{2\cdot 1}(x_1) = \beta_0^t + x_1^t B^*$ donde:

$\beta_0^t = \left(\mu_2 - \Sigma_{21}\Sigma_{11}^{-1}\mu_1\right)^t$: Vector $(1 \times s)$ de intersección con el origen.

$B^* = \Sigma_{11}^{-1}\Sigma_{12}$: Matriz $(q \times s)$de coeficientes de pendiente.

Utilizandose de forma conjunta como $B = \begin{bmatrix} \beta_o^t \\ B^* \end{bmatrix}$.

Para utilizar una notación más convencional puede llamarse:

$$y_{(s\times 1)} = x_2$$

$$x_{((q+1)\times 1)} = \begin{bmatrix} 1 \\ x_1 \end{bmatrix}$$

$$\mu_{2\cdot 1}^t = xB$$

$$u^t = y^t - \mu_{2\cdot 1}$$

y entonces

$$y^t = x^t B + u^t$$

De esta manera el valor de y se descompone en dos partes: la proyección sobre el espacio vectorial generado por las columnas de x y el residuo u, complemento ortogonal al anterior.

La matriz de covarianzas de u es $\Sigma_{yy\cdot x} = \Sigma_{yy} - \Sigma_{yz_1}\Sigma_{x_1x_1}^{-1}\Sigma_{x_1y}$.

La componente correspondiente a los residuos debe ser analizada cuidadosamente una vez estimado el modelo, pues de su estudio se desprenderá la información necesaria para criticar la bondad del modelo.

La otra vía de aproximación al problema de la predicción lineal es el de ajuste por mínimos cuadrados. En él se plantea un modelo $Y_{(N\times s)} = X_{(N\times(q+1))}B_{((q+1)\times s)} + U_{(N\times s)}$. La solución óptima será aquella que minimice la suma de los cuadrados de las desviaciones producidas entre el valor estimado y el real de las variables a explicar. La solución emanará por tanto de

$\frac{\delta \sum_{j=1}^{s}\sum_{i=1}^{N}\left(y_{ij}-\bar{y}_s\right)^2}{\delta B}=0$. El estimador obtenido por esta vía de $B=\begin{bmatrix}\beta_o^t\\ B^*\end{bmatrix}$ es $\hat{B}=\left(X^tX\right)^{-1}X^tY$, que también puede ser escrito como $\hat{B}=\begin{bmatrix}\bar{y}^t-\bar{x}_{\cdot 1}^t S_{x_1x_1}^{-1}S_{x_1y}\\ S_{x_1x_1}^{-1}S_{x_1y}\end{bmatrix}$, resultado que concuerda con el obtenido por la vía estadística. Sin embargo, la aproximación estadística proporciona un entorno más robusto para profundizar en la estructura de los residuos y aplicar técnicas de inferencia estadística como, por ejemplo, la estimación de intervalos de confianza de los coeficientes obtenidos.

Un estimador no sesgado de la matriz de covarianzas del residuo, $\Gamma=\Sigma_{yy\cdot x_1}$, viene dado por $\hat{\Gamma}=\frac{Y-X\hat{B}}{N-q-1}$

A la hora de seleccionar que variables resultan más adecuadas en nuestro modelo, es útil definir el valor $AIC=\hat{\sigma}^2\left(C_p+N\right)$, estimación que depende del estadístico C_p de Mallow. Este valor indica que se debe hacer con cada una de las variables predictoras involucradas. El error cuadrático medio puede ser escrito como la suma de la varianza y el cuadrado del sesgo. No obstante, en una selección de variables es posible optimizar el error cuadrático medio y aun así continuar con un fuerte sesgo. El estadístico C_p de Mallow tiene en cuenta este efecto. El estadístico C_j representa el valor del estadístico C_p cuando se consideran j variables explicativas, $C_j=(N-j-1)\frac{MSE_j}{MSE}-[N-2(j+1)]$, donde MSE y MSE_j representan respectivamente el valor del error cuadrático medio del modelo completo y del modelo que considera únicamente j variables explicativas.

Si el valor AIC correspondiente a una determinada variable es inferior al correspondiente al modelo completo, resulta favorable su inclusión (función add1) o exclusión (función drop1). En este caso parece adecuado incluir en el modelo los efectos del resto de variables.

10.1.2. *Implementación en* ***R***

El ajuste de modelos en **R** se basa en las siguientes funciones básicas a utilizar:

- lm: Modelos lineales.
- aov: modelos de análisis de la varianza.

- glm: Modelos lineales generalizados.
- loess: Modelos de regresión local.

El ajuste por estos métodos no robustos asume la normalidad de los datos implicados. Al no ser normales multivariantes los datos del proceso objeto de estudio, el ajuste obtenido por fuerza no será optimo.

Empezaremos por un modelo conjunto donde no se distinga entre clases y únicamente conste de un término independiente. Puede servir como base para generar los siguientes modelos. Lo denominaremos modelo no agrupado.

```
> attach(datos.nuevos)
> lm1A <- lm(TRRE ~ 1, data=datos.nuevos)
> summary(lm1A)
Call:
lm(formula = TRRE ~ 1, data = datos.nuevos)
Residuals:
    Min      1Q  Median      3Q     Max
-204.02  -36.02   11.98   48.98  172.98
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 216.0199     0.5515   391.7   <2e-16 ***
---
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '  1
Residual standard error: 67.42 on 14941 degrees of freedom
```

La función *summary* proporciona información sobre los coeficientes, el error standard cometido y la significación de cada variable.

Por supuesto un modelo basado únicamente en una constante es inadecuado para el proceso que sometemos a estudio. La incorporación al modelo de una variable de interés no tiene porque significar la reescritura de todo el modelo. En **R** se puede *actualizar* un modelo existente gracias a la función *update*.

```
> lm1 <- lm(TRRE ~ F, data=datos.nuevos)
Call:
lm(formula = TRRE ~ F, data = datos.nuevos)
Residuals:
    Min      1Q  Median      3Q     Max
-196.04  -28.62   11.70   41.06  193.80
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.200e+02  1.728e+00   69.45   <2e-16 ***
F           4.413e-02  7.608e-04   58.01   <2e-16 ***
---
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '  1

Residual standard error: 60.91 on 14940 degrees of freedom
Multiple R-Squared: 0.1838,     Adjusted R-squared: 0.1838
F-statistic:  3365 on 1 and 14940 degrees of freedom,   p-value:     0
```

La incorporación de una variable al modelo debe realizarse según criterios que tengan en cuenta lo aportado al modelo por la nueva variable, de forma que su incorporación se vea justificada. Deben pues evaluarse ahora las variables disponibles para juzgar la idoneidad de su incorporación al modelo. Para ello se dispone de las funciones *add*, si queremos juzgar la incorporación de nuevas variables, y la función *drop* si deseamos evaluar la eliminación de alguna de las ya consideradas.

```
> lm1.add <- add1(lm1A, c("F","DR","PLT","R"))
> lm1.add
Single term additions
Model:
TRRE ~ 1
       Df Sum of Sq      RSS      AIC
<none>              67904785   125839
F       1  12484293 55420493   122805
DR      1  11009144 56895641   123198
PLT     1     97105 67807680   125819
R       1      6987 67897798   125839
```

El valor del AIC correspondiente al modelo completo viene indicado en la fila encabezada por <*none*>. Aquellas variables que tengan asociado un valor inferior a este son susceptibles de ser añadidas al modelo (en la función *add*), o extraidas del mismo (en la función *drop).*

La función *add* considera adecuada la incorporación de nuevas variables.

```
> lm2 <- update(lm1,.~.+F+DR+PLT+R)
> summary(lm2)

Call:
lm(formula = TRRE ~ F + DR + PLT + R, data = datos.nuevos)
Residuals:
      Min        1Q    Median        3Q       Max
-215.7281  -19.5031    0.6498   19.2280  223.1334
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.969e+02  9.361e+00 -21.031   <2e-16 ***
F            9.243e-02  6.095e-04 151.646   <2e-16 ***
DR           9.176e+00  6.957e-02 131.908   <2e-16 ***
PLT          1.439e-01  6.709e-03  21.450   <2e-16 ***
R            1.761e-01  1.985e-02   8.872   <2e-16 ***
---
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '
1
Residual standard error: 37.04 on 14937 degrees of freedom
Multiple R-Squared: 0.6982,     Adjusted R-squared: 0.6981
F-statistic:  8640 on 4 and 14937 degrees of freedom,   p-value:     0
```

La función *plot* puede extraer información de gran interés de un objeto de tipo modelo. Genera cuatro tipos distintos de gráficos que informan visualmente sobre la bondad del ajuste realizado.

1. Residuos frente a valores ajustados.

2. Gráfico cuantil-normal de los residuos. Permite evaluar la normalidad de los residuos. La carencia de esta hace sospechar de una estructura interna remanente por explicar.

3. Raiz cuadrada del valor absoluto de los residuos frente a los valores ajustados. Resulta útil a la hora de identificar outliers y visualizar su estructura.

4. Distancias de Cook. **[11,** pág. 158**]**: Representan la medida en que cada una de las observaciones influye en la regresión de los coeficientes.

```
> plot(lm2, pch=".", col="blue")
```

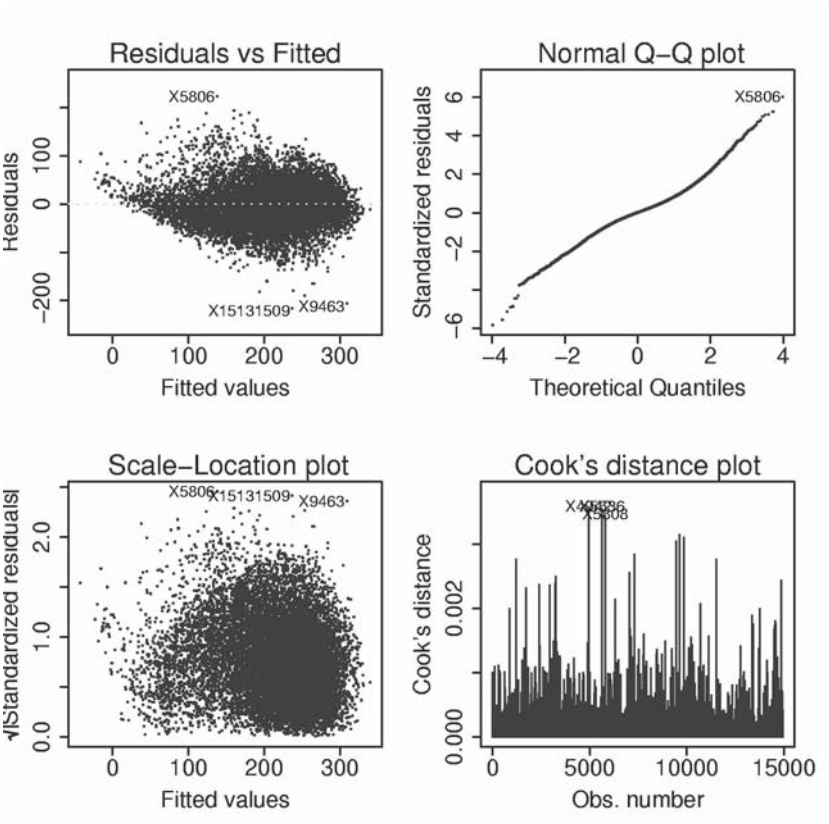


Fig. 67: Gráficos del ajuste lineal del modelo lm2.

A la vista de los gráficos correspondientes al modelo *lm2* se observan claras deficiencias en el ajuste. Entre ellas una clara falta de normalidad en los

residuos. Ello no es tampoco de extrañar si se considera que los datos introducidos proceden realmente de una distribución no normal multivariante.

```
> lm2.drop <- drop1(lm2)
> summary(lm2.drop)
       Df      Sum of Sq             RSS                AIC
 Min.   :1   Min.   :  108000   Min.   :20490000   Min.   :107900
 1st Qu.:1   1st Qu.:  500400   1st Qu.:20600000   1st Qu.:108000
 Median :1   Median :12250000   Median :21120000   Median :108400
 Mean   :1   Mean   :14040000   Mean   :31720000   Mean   :113100
 3rd Qu.:1   3rd Qu.:25790000   3rd Qu.:44360000   3rd Qu.:119500
 Max.   :1   Max.   :31550000   Max.   :52040000   Max.   :121900
 NA's   :1   NA's   :       1
> detach(datos.nuevos)
```

Las funciones *add1* y *drop1* pueden resultan cómodas a la hora de comprobar el efecto de unos pocos términos. En el caso de manejar un número considerable de variables, resulta más adecuado la utilización de la orden *step*.

```
> step(lm2)
Start:  AIC= 107945.3
 TRRE ~ F + DR + PLT + R
       Df Sum of Sq      RSS      AIC
<none>              20492088   107945
- R     1    107974 20600062   108022
- PLT   1    631187 21123275   108397
- DR    1  23870582 44362670   119484
- F     1  31548907 52040995   121869
Call:
lm(formula = TRRE ~ F + DR + PLT + R, data = datos.nuevos)
Coefficients:
(Intercept)            F           DR          PLT            R
 -196.87706      0.09243      9.17649      0.14391      0.17614
```

Se podría aplicar un test de normalidad a los residuos que refuerce lo visualmente observado.

```
> ks.test(lm2$residuals,"pnorm",mean= mean(lm2$residuals),
+ sd=sqrt(var(lm2$residuals)))
One-sample Kolmogorov-Smirnov test
data:  lm2$residuals
D = 0.0534, p-value = < 2.2e-16
```

El test rechaza igualmente la posibilidad de que el residuo carezca de estructura. Debe mejorarse el modelo propuesto utilizando otras técnicas.

Previamente se propondrá un modelo que atienda a la clasificación antigua y otros que utilicen las clasificaciones obtenidas anteriormente con los algoritmos de clusterización k-means y SOM, a fin de compararlos entre sí y con referencia también al modelo general sin distinción alguna entre clases.

Añadiremos nuevas componentes al data.frame *datos.nuevos* que reflejen las clasificaciones realizadas hasta ahora.

```
> datos.nuevos$Cceq <- cut(datos.nuevos$TRCEQ, cortes)
> datos.nuevos$Ckmeans <- factor(dcclust$cluster)
> datos.nuevos$Csom <- factor(matrix(1:9, ncol=3)[
+             cbind(datos.nuevos.out[,1], datos.nuevos.out[,2])])

> datos.nuevos[1,]
  TRRE    F    DR    PLT       R TRCEQ    Cold Cnew Csom
1  150 1956 15.08 233.82 463.295    14 (13,20]    9    7
```

Se proponen modelos distintos para cada clasificación

```
> lm2.ceq <- update(lm1,.~.+F+DR+PLT+R + Cceq + Cceq:F + Cceq:DR
+                     + Cceq:PLT + Cceq:R)
> lm2.kmeans <- update(lm1,.~.+F+DR+PLT+R + Ckmeans +Ckmeans:F +
+                 Ckmeans:DR + Ckmeans:PLT +Ckmeans:R)
> lm2.som <- update(lm1,.~.+F+DR+PLT+R + Csom + Csom:F + Csom:DR
+                     + Csom:PLT + Csom:R)
> lm2.ceq
Call:
lm(formula = TRRE ~ F + DR + PLT + R + Cceq + F:Cceq + DR:Cceq +     PLT:Cceq
+             + R:Cceq, data = datos.nuevos)

Coefficients:
    (Intercept)                F               DR              PLT
     -2.477e+02        1.064e-01        6.880e+00        8.914e-02
              R      Cceq(20,24]      Cceq(24,28]      Cceq(28,32]
      3.000e-01        2.544e+01        1.570e+01        4.189e+01
    Cceq(32,36]      Cceq(36,40]      Cceq(40,44]      Cceq(44,48]
      9.641e+01       -1.444e+01        4.024e+01        2.799e+02
    Cceq(48,63]    F.Cceq(20,24]    F.Cceq(24,28]    F.Cceq(28,32]
      2.415e+01       -1.420e-02       -1.892e-02       -1.120e-02
  F.Cceq(32,36]    F.Cceq(36,40]    F.Cceq(40,44]    F.Cceq(44,48]
     -1.543e-02       -1.020e-02       -1.416e-02        3.304e-03
  F.Cceq(48,63]   DR.Cceq(20,24]   DR.Cceq(24,28]   DR.Cceq(28,32]
      4.192e-05        2.234e+00        3.749e+00        2.283e+00
 DR.Cceq(32,36]   DR.Cceq(36,40]   DR.Cceq(40,44]   DR.Cceq(44,48]
      1.869e+00        2.185e+00        2.278e+00        1.560e+00
 DR.Cceq(48,63]  PLT.Cceq(20,24]  PLT.Cceq(24,28]  PLT.Cceq(28,32]
      4.110e-01       -1.541e-03       -4.572e-02        1.112e-01
PLT.Cceq(32,36]  PLT.Cceq(36,40]  PLT.Cceq(40,44]  PLT.Cceq(44,48]
      6.948e-02        1.500e-01        1.057e-01        2.947e-01
PLT.Cceq(48,63]    R.Cceq(20,24]    R.Cceq(24,28]    R.Cceq(28,32]
      1.310e-01       -6.732e-02       -4.528e-02       -1.349e-01
  R.Cceq(32,36]    R.Cceq(36,40]    R.Cceq(40,44]    R.Cceq(44,48]
     -2.068e-01       -1.160e-02       -9.946e-02       -6.934e-01
  R.Cceq(48,63]
     -8.757e-02
```

```
> lm2.kmeans

Call:
lm(formula = TRRE ~ F + DR + PLT + R + Ckmeans + F:Ckmeans + DR:Ckmeans +
PLT:Ckmeans + R:Ckmeans, data = datos.nuevos)


Coefficients:
(Intercept)                 F                DR               PLT                 R
Ckmeans2
 -242.04183       0.11885       3.84783      -0.07281       0.42503       7.99464
      Ckmeans3         Ckmeans4         Ckmeans5         Ckmeans6         Ckmeans7
Ckmeans8
  132.29328      31.04777      65.59934      44.71005      37.53695      93.01825
      Ckmeans9       F.Ckmeans2       F.Ckmeans3       F.Ckmeans4       F.Ckmeans5
F.Ckmeans6
   62.87105      -0.02246      -0.05532      -0.02372      -0.02022      -0.02580
    F.Ckmeans7       F.Ckmeans8       F.Ckmeans9      DR.Ckmeans2      DR.Ckmeans3
DR.Ckmeans4
   -0.02688      -0.01993      -0.02905       7.49498       8.20971       6.28468
   DR.Ckmeans5      DR.Ckmeans6      DR.Ckmeans7      DR.Ckmeans8      DR.Ckmeans9
PLT.Ckmeans2
    1.90590       5.54740       5.84616      -1.42543       3.61164       0.67286
  PLT.Ckmeans3     PLT.Ckmeans4     PLT.Ckmeans5     PLT.Ckmeans6     PLT.Ckmeans7
PLT.Ckmeans8
    1.48806       0.41693       0.21931       0.21602       1.63589       0.10154
  PLT.Ckmeans9       R.Ckmeans2       R.Ckmeans3       R.Ckmeans4       R.Ckmeans5
R.Ckmeans6
    0.22652      -0.27498      -0.37277      -0.26124      -0.20218      -0.23591
    R.Ckmeans7       R.Ckmeans8       R.Ckmeans9
   -0.34947      -0.17323      -0.21228
> lm2.som <- update(lm1,.~.+F+DR+PLT+R + Csom + Csom:F + Csom:DR +
Csom:PLT + Csom:R)


> lm2.som

Call:
lm(formula = TRRE ~ F + DR + PLT + R + Csom + F:Csom + DR:Csom +
PLT:Csom + R:Csom, data = datos.nuevos)


Coefficients:
(Intercept)            F           DR          PLT            R        Csom2
 -2.259e+02    1.271e-01    3.535e+00    3.864e-02    3.362e-01    1.009e+02
      Csom3        Csom4        Csom5        Csom6        Csom7        Csom8
  7.050e+01    6.014e+01   -2.279e+01   -2.911e+02   -1.624e+01    1.642e+02
      Csom9      F.Csom2      F.Csom3      F.Csom4      F.Csom5      F.Csom6
  7.535e+01   -6.726e-02   -4.767e-02   -3.866e-02   -1.882e-02    5.200e-02
    F.Csom7      F.Csom8      F.Csom9     DR.Csom2     DR.Csom3     DR.Csom4
 -9.594e-03   -1.192e-02   -4.256e-02    8.802e+00    6.719e+00    2.719e+00
   DR.Csom5     DR.Csom6     DR.Csom7     DR.Csom8     DR.Csom9     PLT.Csom2
  7.582e+00    5.798e+00    4.723e+00   -2.694e-01    5.760e+00   -2.580e-01
  PLT.Csom3    PLT.Csom4    PLT.Csom5    PLT.Csom6    PLT.Csom7    PLT.Csom8
```

```
  1.382e+00    1.183e-01    4.354e-01    8.171e-01    9.610e-02   -8.871e-02
  PLT.Csom9      R.Csom2      R.Csom3      R.Csom4      R.Csom5      R.Csom6
  3.094e-01   -1.870e-01   -2.854e-01   -1.146e-01   -2.064e-01   -4.967e-02
    R.Csom7      R.Csom8      R.Csom9
 -1.226e-01   -2.697e-01   -2.193e-01
```

Comparemos los resultados obtenidos por los modelos propuestos. Los resultados más interesantes para la empresa siderúrgica son el error medio y el valor de la varianza.

```
> ev <-function(modelo)
+{
+ return(unlist(list(error= mean(abs(modelo$residuals)),
+         varianza= var(abs(modelo$residuals)))))
+}

> ev(lm1)
  error        varianza
   47.2522   1476.3704
> ev(lm2)
  error        varianza
  26.9527     645.0382
> ev(lm2.ceq)
  error        varianza
 26.4062      609.5832
> ev(lm2.kmeans)
  error        varianza
 22.4229      460.7878
> ev(lm2.som)
  error        varianza
 22.16304     434.8805
```

Los modelos obtenidos con las “clasificaciones modernas” presentan no solo menor error medio, sino también, menor desviación típica.

Puede compararse la similitud de los modelos propuestos con un análisis anova de los resultados obtenidos con cada modelo.

```
> anova(lm2)
Analysis of Variance Table
Response: TRRE
             Df   Sum Sq  Mean Sq   F value    Pr(>F)
F             1 12484293 12484293  9099.994 < 2.2e-16 ***
DR            1 34174726 34174726 24910.486 < 2.2e-16 ***
PLT           1   645705   645705   470.664 < 2.2e-16 ***
R             1   107974   107974    78.704 < 2.2e-16 ***
Residuals 14937 20492088     1372
---
```

```
> anova(lm2.ceq)
Analysis of Variance Table
Response: TRRE
             Df   Sum Sq  Mean Sq    F value    Pr(>F)
F             1 12484293 12484293  9524.3416 < 2.2e-16 ***
DR            1 34174726 34174726 26072.1029 < 2.2e-16 ***
PLT           1   645705   645705   492.6120 < 2.2e-16 ***
R             1   107974   107974    82.3740 < 2.2e-16 ***
Cceq          8   339494    42437    32.3752 < 2.2e-16 ***
F:Cceq        8   221387    27673    21.1122 < 2.2e-16 ***
DR:Cceq       8   192947    24118    18.4001 < 2.2e-16 ***
PLT:Cceq      8   158392    19799    15.1048 < 2.2e-16 ***
R:Cceq        8    53216     6652     5.0748 2.528e-06 ***
Residuals 14897 19526652     1311
---
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '  1
> anova(lm2.kmeans)
Analysis of Variance Table
Response: TRRE
             Df   Sum Sq  Mean Sq    F value    Pr(>F)
F             1 12484293 12484293 12917.6334 < 2.2e-16 ***
DR            1 34174726 34174726 35360.9608 < 2.2e-16 ***
PLT           1   645705   645705   668.1177 < 2.2e-16 ***
R             1   107974   107974   111.7219 < 2.2e-16 ***
Ckmeans       8  2175886   271986   281.4266 < 2.2e-16 ***
F:Ckmeans     8   279094    34887    36.0976 < 2.2e-16 ***
DR:Ckmeans    8  3258948   407368   421.5086 < 2.2e-16 ***
PLT:Ckmeans   8   359239    44905    46.4636 < 2.2e-16 ***
R:Ckmeans     8    21663     2708     2.8019  0.004222 **
Residuals 14897 14397258      966
---
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '  1
> anova(lm2.som)
Analysis of Variance Table
Response: TRRE
             Df   Sum Sq  Mean Sq    F value    Pr(>F)
F             1 12484293 12484293 13440.6052 < 2.2e-16 ***
DR            1 34174726 34174726 36792.5531 < 2.2e-16 ***
PLT           1   645705   645705   695.1666 < 2.2e-16 ***
R             1   107974   107974   116.2449 < 2.2e-16 ***
Csom          8  2244240   280530   302.0189 < 2.2e-16 ***
F:Csom        8   413259    51657    55.6144 < 2.2e-16 ***
DR:Csom       8  3487637   435955   469.3493 < 2.2e-16 ***
PLT:Csom      8   479297    59912    64.5014 < 2.2e-16 ***
R:Csom        8    30592     3824     4.1169 6.404e-05 ***
Residuals 14897 13837063      929
---
```

```
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '
> par(mfrow=c(3,2))
> eqscplot(TRRE,predict.lm(lm1), pch=".",col="red")
> abline(0,1, col="blue",add=T)
> eqscplot(TRRE,predict.lm(lm2), pch=".",col="red")
> abline(0,1, col="blue",add=T)
> eqscplot(TRRE,predict.lm(lm2.ceq), pch=".",col="red")
> abline(0,1, col="blue",add=T)
> eqscplot(TRRE,predict.lm(lm2.kmeans), pch=".",col="red")
> abline(0,1, col="blue",add=T)
> eqscplot(TRRE,predict.lm(lm2.som), pch=".",col="red")
> abline(0,1, col="blue",add=T)
```

En la Fig. 68 destaca no sólo el mal ajuste realizado en el modelo lm1, donde únicamente se utilizaba la fuerza como variable de predicción, sino la escasa diferencia de comportamiento entre el modelo lm2 y el modelo lm2.old, que atendía a la clasificación por grados de acero. El ajuste correspondiente a lm2.new, correspondiente a la clasificación obtenida con el algoritmo k-means, se diferencia muy ligeramente del modelo global de datos y presenta mayor acercamiento general; menor varianza en el residuo; a la recta identidad.

```
> anova(lm2, lm2.ceq)
Analysis of Variance Table
Model 1: TRRE ~ F + DR + PLT + R
Model 2: TRRE ~ F + DR + PLT + R + Cceq + F:Cceq + DR:Cceq + PLT:Cceq +
  Res.Df Res.Sum Sq    Df   Sum Sq F value    Pr(>F)
1  14937   20492088
2  14897   19526652    40   965436  18.413 < 2.2e-16 ***
---
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '  1
> anova(lm2, lm2.kmeans)
Analysis of Variance Table
Model 1: TRRE ~ F + DR + PLT + R
Model 2: TRRE ~ F + DR + PLT + R + Ckmeans + F:Ckmeans + DR:Ckmeans +
PLT:Ckmeans +
  Res.Df Res.Sum Sq    Df   Sum Sq F value    Pr(>F)
1  14937   20492088
2  14897   14397258    40  6094830  157.66 < 2.2e-16 ***
Signif. codes:  0 `***' 0.001  `**'  0.01  `*'  0.05  `.'  0.1  ` '  1
```

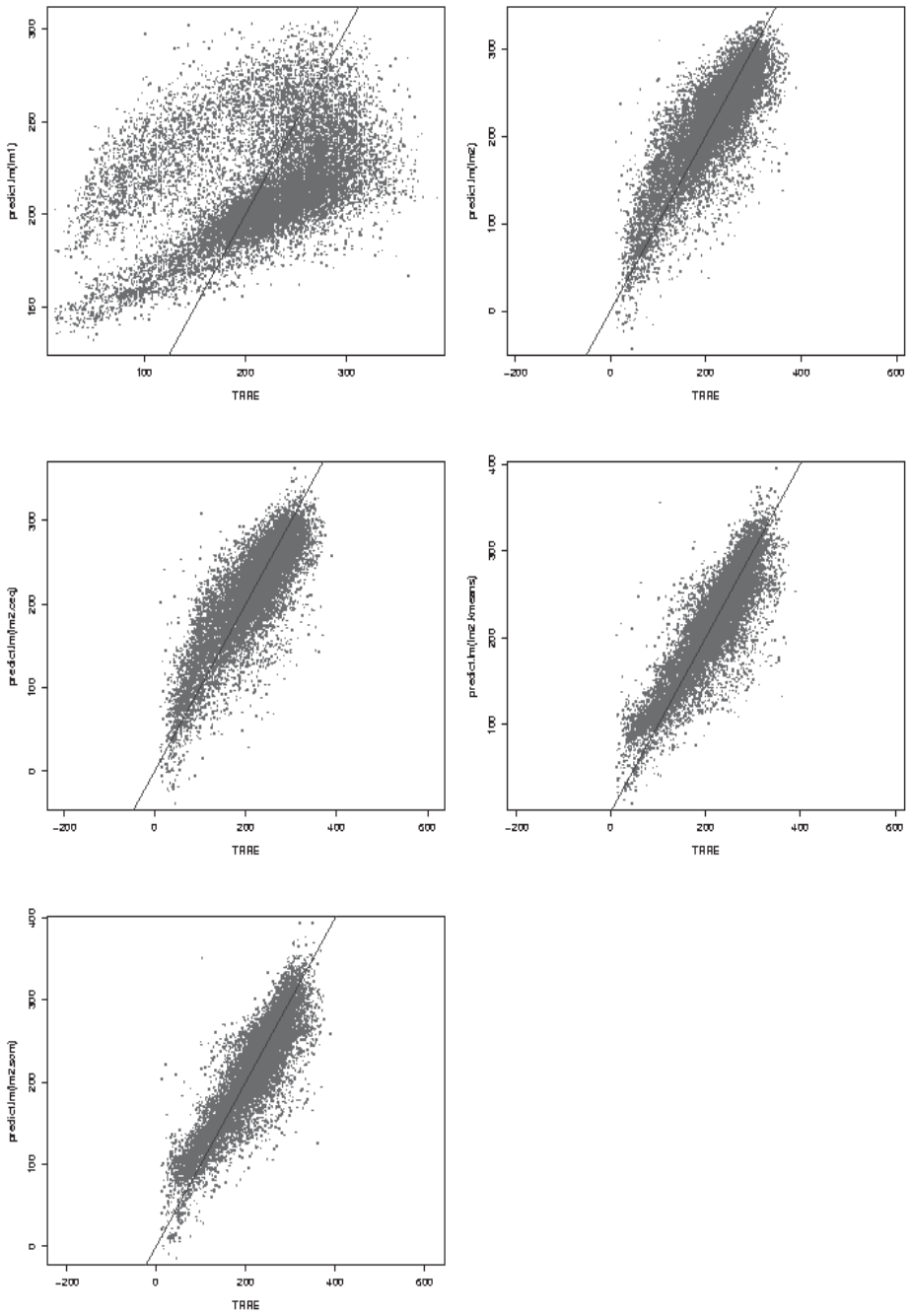


Fig. 68: Ajustes obtenidos con los modelos propuestos.

> anova(lm2, lm2.som)

```
>  anova(lm2, lm2.som)
Analysis of Variance Table
Model 1: TRRE ~ F + DR + PLT + R
Model 2: TRRE ~ F + DR + PLT + R + Csom + F:Csom + DR:Csom +PLT:Csom +
  Res.Df Res.Sum Sq    Df    Sum Sq F value    Pr(>F)
1  14937   20492088
2  14897   13837063    40  6655025  179.12 < 2.2e-16 ***
---
Signif. codes:  0  `***'  0.001  `**'  0.01  `*'  0.05  `.' 0.1 ` '  1
```

El siguiente paso a la hora de mejorar los modelos obtenidos puede ser incorporar ciertas combinaciones lineales de polinomios formados a partir de las variables explicativas de interés.

```
> polylm2 <- lm(TRRE~poly(F,2)+poly(DR,3)+poly(PLT,3)+R)
> polylm2.ceq <- lm(TRRE ~ poly(F,2) + poly(DR,3) +
+    poly(PLT,3) + R + Cceq + Cceq:poly(F,2) + Cceq:poly(DR,3) +
+    Cceq:poly(PLT,3) + Cceq:R)
> polylm2.kmeans <- lm(TRRE~poly(F,2)+poly(DR,3)+ poly(PLT,3)+
+     R + Ckmeans + Ckmeans:poly(F,2) + Ckmeans:poly(DR,3) +
+     Ckmeans:poly(PLT,3) + Ckmeans:R)
> polylm2.som <- lm(TRRE~poly(F,2)+poly(DR,3)+ poly(PLT,3)+
+     R + Csom + Csom:poly(F,2) + Csom:poly(DR,3) +
+     Csom:poly(PLT,3) + Csom:R)

> par(mfrow=c(4,1))
> plot(TRRE, predict(polylm2),pch=".", col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(polylm2.ceq),pch=".", col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(polylm2.kmeans),pch=".", col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(polylm2.som),pch=".", col="red")
> abline(0,1,col="blue")
```

En la Fig. 69 se muestra los valores reales de TRRE frente a los obtenidos con los modelos propuestos.

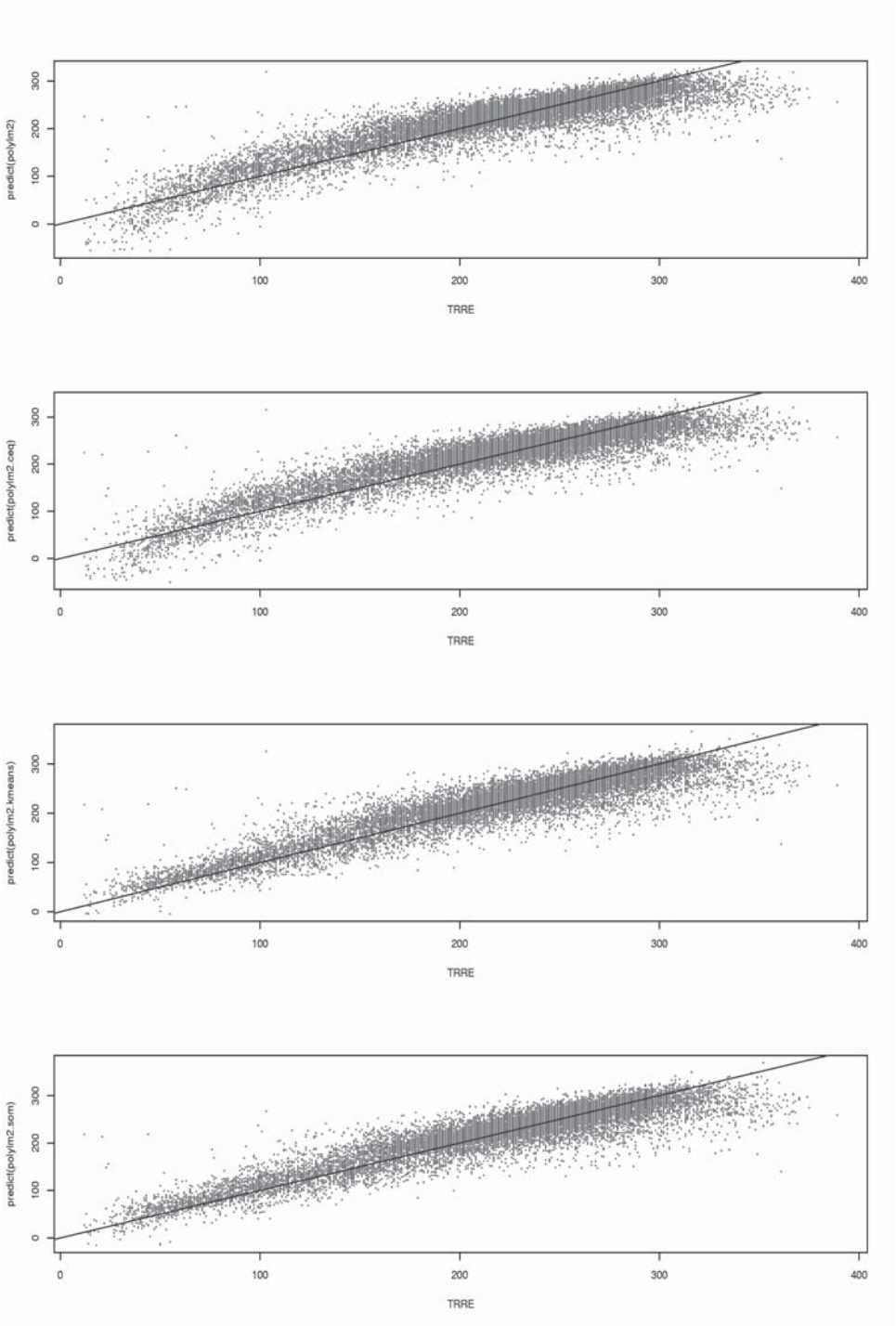


Fig. 69: Ajustes obtenidos con los ajustes polinomiales.

```
> ev(polylm2)
   error    varianza
 21.69547   369.67395
> ev(polylm2.ceq)
   error    varianza
 21.14931   343.47506
> ev(polylm2.kmeans)
   error    varianza
 19.76294   335.78992
> ev(polylm2.som)
    error   varianza
 19.44923   325.06203
```

La gráfica y los resultados numéricos del error medio cometido revelan la práctica total similitud en el caso del modelo sin clasificación previa y el modelo con clasificación tradicional. Únicamente el modelo con clasificación k-means y SOM se diferencia, eso sí, tan ligeramente, que cuestiona la propia necesidad de la clasificación.

Puede evaluarse el resultado obtenido si se proponen polinomios de orden superior, digamos 3, utilizando las componentes principales.

```
> aux <- nuevo.pca$rproj
> lm3 <- lm(TRRE ~poly( aux[,1],3) + poly(aux[,2],3) +
+   poly(aux[,3],3) + poly(aux[,4],3))
> lm3.ceq <- lm(TRRE ~poly( aux[,1],3) + poly(aux[,2],3) +
+   poly(aux[,3],3) + poly(aux[,4],3) + poly( aux[,1],3):Cceq +
+   poly(aux[,2],3):Cceq + poly(aux[,3],3):Cceq +
+   poly(aux[,4],3):Cceq  )
> lm3.kmeans <- lm(TRRE ~poly(aux[,1],3) + poly(aux[,2],3) +
+    poly(aux[,3],3) +poly(aux[,4],3) + poly( aux[,1],3):Ckmeans
+
+   poly(aux[,2],3):Ckmeans + poly(aux[,3],3):Ckmeans +
+   poly(aux[,4],3):Ckmeans  )
> lm3.som <- lm(TRRE ~poly(aux[,1],3) + poly(aux[,2],3) +
+   poly(aux[,3],3) +poly(aux[,4],3) + poly( aux[,1],3):Csom +
+ poly(aux[,2],3):Csom + poly(aux[,3],3):Csom +
+ poly(aux[,4],3):Csom  )
```

```
> ev(lm3)
    error      varianza
 24.96337     528.64365
> ev(lm3.ceq)
    error      varianza
 24.21955    488.37145
> ev(lm3.kmeans)
    error      varianza
 20.14775    337.11621
> ev(lm3.som)
    error      varianza
 19.87637     325.45112

> par(mfrow=c(4,1), pch=".")
> plot(TRRE, predict(lm3)    ,col="red"); abline(0,1,col="blue")
> plot(TRRE, predict(lm3.ceq),col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(lm3.kmeans), col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(lm3.som), col="red")
> abline(0,1,col="blue")
```

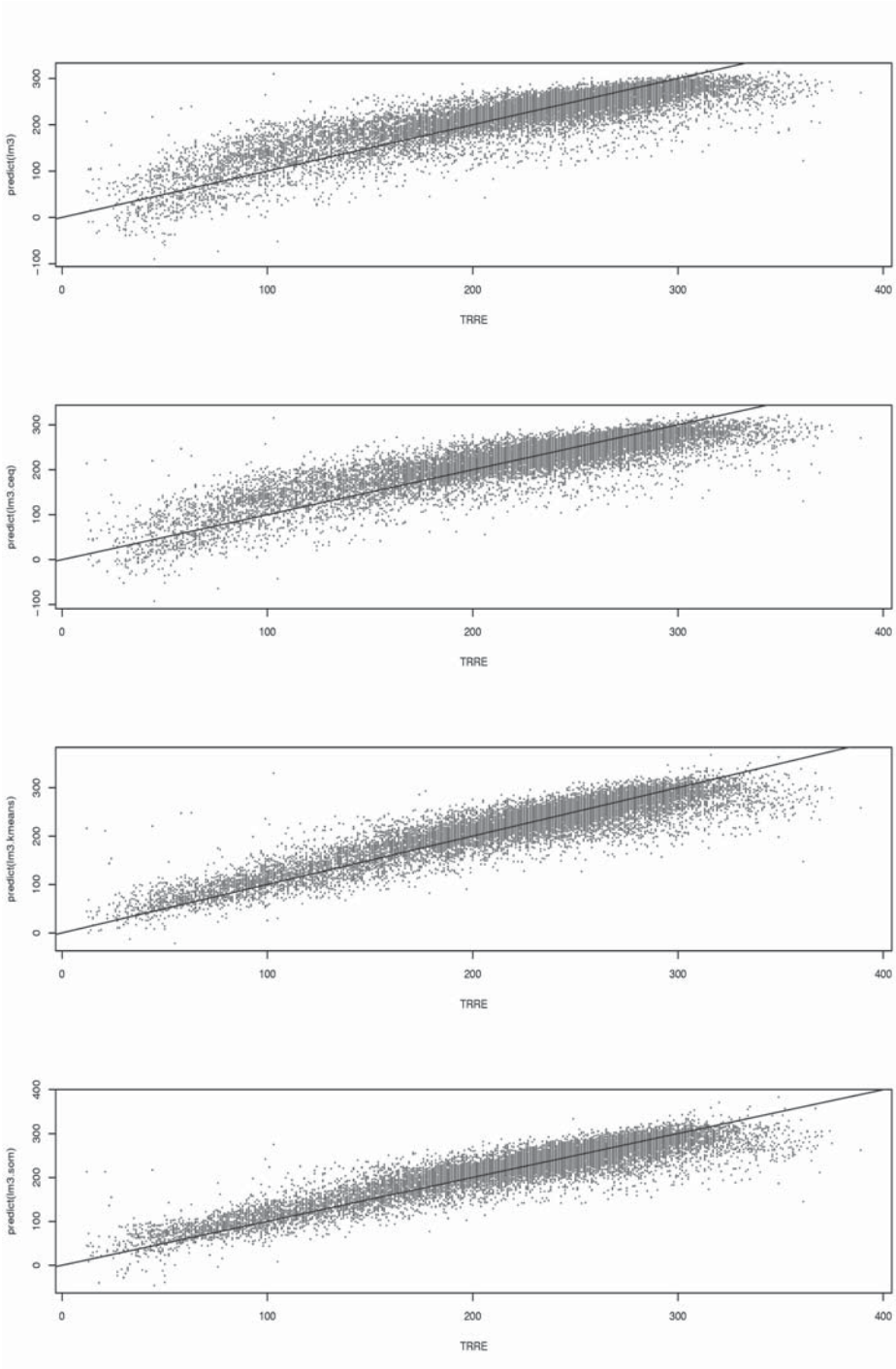


Fig. 70: Ajustes obtenidos con las componentes principales.

Al igual que en el caso anterior no se perciben diferencias apreciables entre el modelo general sin clases y el modelo obtenido con la clasificación clásica. Las clasificaciones algorítmicas modernas logran ligeras mejoras frente a estos otros dos modelos (sin clases y con clases según carbono), pero no se justifica claramente la necesidad de proyectar las componentes principales a efectos de introducir variables en el modelo lineal.

10.2. Modelos Lineales generalizados

Los modelos lineales generalizados son una extensión de los modelos lineales que permiten utilizar tanto distribuciones no normales como transformaciones hacia la linealidad. Las condiciones que debe cumpluir un modelo lineal generalizado son:

- La variable explicada y es independiente de las variables predictoras.
- Las variables predictoras, x_1,K,x_n, únicamente pueden influir en la distribución de y a través de una función lineal $\eta = \beta_1 x_1 + \Lambda + \beta_n x_n$, denominada predictor lineal.
- La distribución de y sigue una función de densidad de probabilidad de la forma:

$$f(y_i;\theta_i;\varphi) = e^{A_i \frac{y_i\theta_i - \gamma(\theta_i)}{\varphi} + \tau\left(y_i, \frac{\varphi}{A_i}\right)}$$

donde φ es un parámetro de escala, A_i es un peso de ponderación y el parámetro θ_i controla la distribución de y_i

10.2.1. *Implementacion en R*

La función que permite la obtenciçon de modelos lineales generalizados es *glm*. Análogamente a lo hecho enlos modelos lineales se distinguiran distintos modelos según las distintas clasificaciones.

```
> glm2 <- glm(TRRE~poly(F,2)+poly(DR,3)+poly(PLT,3)+R)
> glm2.ceq <- glm(TRRE ~ poly(F,2) + poly(DR,3) +
+    poly(PLT,3) + R + Cceq + Cceq:poly(F,2) + Cceq:poly(DR,3) +
+    Cceq:poly(PLT,3) + Cceq:R)
> glm2.kmeans <- glm(TRRE~poly(F,2)+poly(DR,3)+ poly(PLT,3)+
+     R + Ckmeans + Ckmeans:poly(F,2) + Ckmeans:poly(DR,3) +
+     Ckmeans:poly(PLT,3) + Ckmeans:R)
> glm2.som <- glm(TRRE~poly(F,2)+poly(DR,3)+ poly(PLT,3)+
+     R + Csom + Csom:poly(F,2) + Csom:poly(DR,3) +
+     Csom:poly(PLT,3) + Csom:R)
```

Los resultados obtenidos son …

```
> ev(glm2)
error      varianza
21.69547   369.67395
> ev(glm2.ceq)
error      varianza
21.14931   343.47506
> ev(glm2.kmeans)
error      varianza
19.76294   335.78992
> ev(glm2.som)
    error      varianza
19.44923   325.06203
> par(mfrow=c(4,1))
> plot(TRRE, predict(polylm2),pch=".", col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(polylm2.ceq),pch=".", col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(polylm2.kmeans),pch=".", col="red")
> abline(0,1,col="blue")
> plot(TRRE, predict(polylm2.som),pch=".", col="red")
> abline(0,1,col="blue")
```

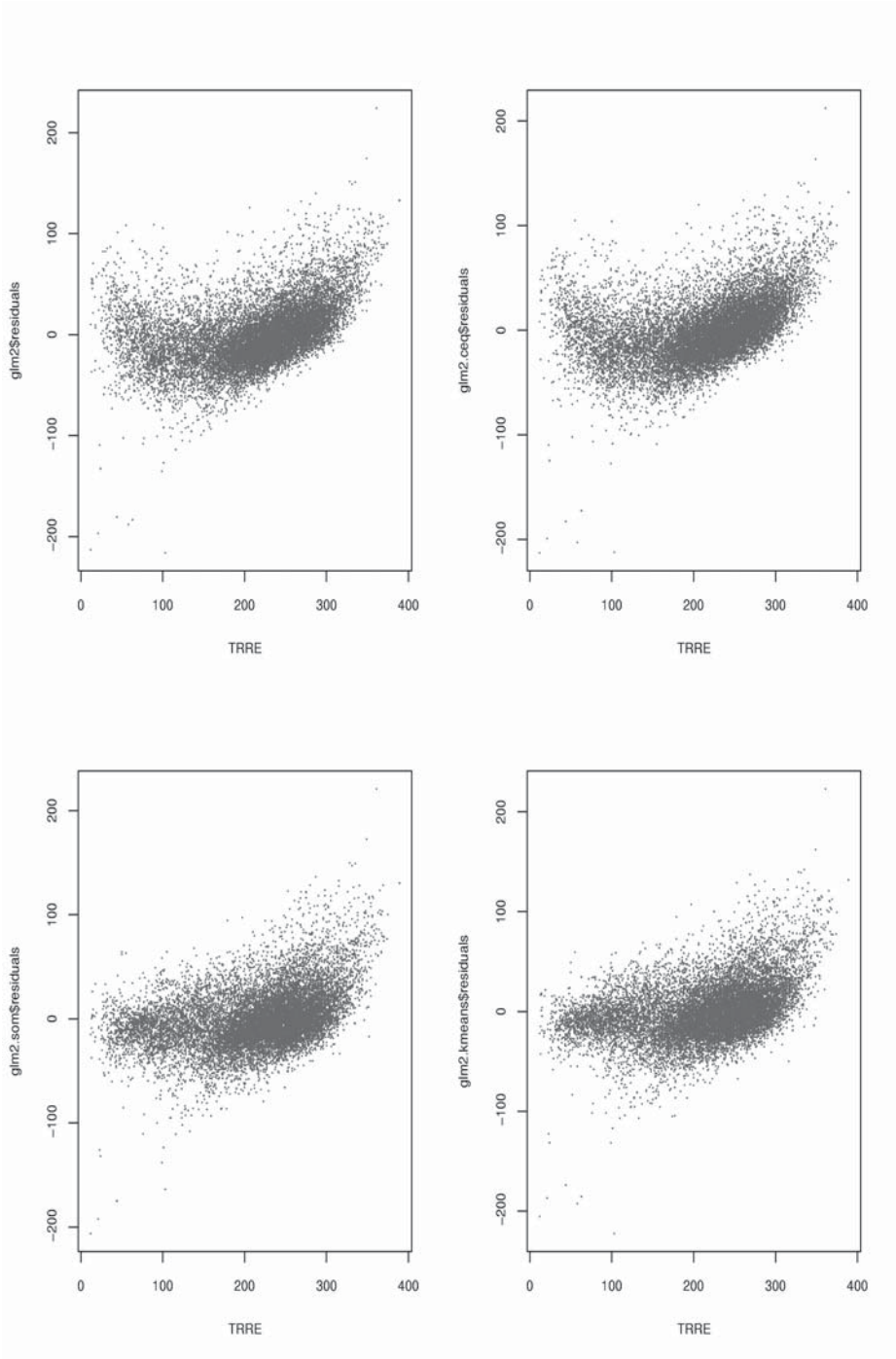


Fig. 71: Residuos de los modelos propuestos.

Dentro de la librería *lmtest* se encuentra la función *dwtest*, que realiza el test de Durbin-Watson que permite conocer si los residuos se encuentran autocorrelados.

Si se lo aplicamos a los modelos generalizados propuestos se obtiene…

```
> dwtest(glm2)
        Durbin-Watson-Test
data:
DW = 1.2236, p-value = < 2.2e-16
> dwtest(glm2.ceq)
        Durbin-Watson-Test
data:
DW = 1.2814, p-value = < 2.2e-16
> dwtest(glm2.kmeans)
        Durbin-Watson-Test
data:
DW = 1.2988, p-value = < 2.2e-16
> dwtest(glm2.som)
        Durbin-Watson-Test
data:
DW = 1.2878, p-value = < 2.2e-16
```

El test de Durbin-Watson no considera que los residuos exista correlación seriada.

10.3. Conclusión

Los modelos lineales propestos aun conservan estructura en sus residuos lo que da cabida a posibles mejoras introduciendo nuevos términos no lineales. En general se ha visto un comportamiento ligeramente superior por parte de los modelos aplicados a las clases obtenidas con algoritmos de clasificación automática, mientras que la clasificación en función del carbono equivalente no muestra mejoras apreciables frente al modelo agrupado.

La propuesta de modelos no lineales por técnicas clásicas se escapa del alcance de este trabajo, pues se ha preferido mostrar la aplicación de redes neuronales como aproximador no lineal. Por tanto se prevén posibles mejoras en este enfoque neuronal no lineal.

Capítulo 11

Aproximación Neuronal

Una vez desarrollado el ajuste de modelos por técnicas clásicas, se ensayará el aprendizaje de una red neuronal que tendrá como entradas las variables que influyen en el par y los errores cometidos en las pasadas anteriores en la laminación. La salida será única y lógicamente expresará el valor del par. El software elegido para el aprendizaje de la red neuronal es SNNS[9] (Stuttgart Neural Network Simulator) de libre distribución.

SNNS es un simulador de redes neuronales desarrollado por miembros de la universidad de Sttutgart. Su primera versión apareció en 1989. Se basa en cuatro componentes principales.

- Un kernel de simulación.
- Un interfaz gráfico con el usuario.
- Un interprete de comandos por lotes.
- Un traductor de redes a funciones en lenguaje C.

SNNS ha sido implementado completamente en lenguaje ANSI-C, lo que le facilita su portabilidad a multitud de arquitecturas, entre ellas Windows, Linux y múltiples `UNIX`.

El modelo neuronal propuesto pretende mejorar los datos obtenidos. Para ello, se añadirán a las variables de entrada que ya se usaron para el ajuste clásico, las siguientes:

TRRE-1 Par real en la pasada anterior (pasada n-1).

TRRE-2 Par real en la pasada antepenúltima (pasada n-2).

$\mathbf{T_{calculado}}$ Par calculado por el modelo clásico.

$\mathbf{T_{calculado\text{-}1}}$ Par calculado en la pasada anterior (pasada n-1).

$\mathbf{T_{calculado\text{-}2}}$ Par calculado en la pasada antepenúltima (pasada n-2).

Los valores de par calculados, tanto en la pasada *n*, como en las pasadas *n-1* y *n-2*, se obtienen aplicando la fórmula del par e introduciendo en los valores de las constantes los obtenidos como resultado de la estrategia evolutiva.

[9] Disponible en http://www-ra.informatik.uni-tuebingen.de/SNNS/

La red neuronal que ha dado los mejores resultados entre las probadas es una del tipo 10-15-1, es decir con diez neuronas en la capa de entrada, quince neuronas en la capa intermedia y una neurona en la capa de salida. En el entrenamiento la variable de salida es el par real, TRRE.

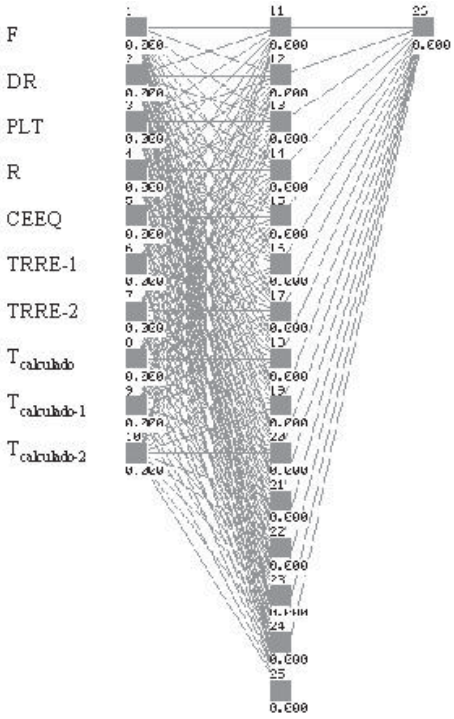


Fig. 72: Estructura de la red neuronal entrenada.

La configuración de los parámetros del entrenamiento es la siguiente:

- Función de inicio de pesos: Pesos aleatorios
- Número de parámetros de inicialización: 2
- Iniciación de parámetros: -0.3 +0.3
- Número máximo de ciclos: 300 000
- Regla de aprendizaje: Backpropagation

Para el entrenamiento de la red es necesario generar tres tipos de patrones.

- Patrones de entrenamiento
- Patrones de test:
- Patrones de validación.

La necesidad de estos tres patrones deriva del propio mecanismo de aprendizaje de la red neuronal.

El fichero de patrones de entrenamiento es utilizado en el ajuste de los pesos de las neuronas mediante la comparación del valor real que debería haber tomado la salida y el valor obtenido por la red en función de los pesos actuales de sus neuronas. De él depende la dirección en la que se ajusten los pesos.

El fichero de patrones de test es utilizado para comprobar cada cierto número de ciclos que el error obtenido con el patrón de entrenamiento es coherente con el error que vería la red neuronal frente a nuevos datos que no hubieran participado en el ajuste. El entrenamiento no finaliza cuando se obtiene el mínimo error en el patrón de entrenamiento sino cuando se minimiza el error procedente del patrón de test. Lo contrario es indeseable, pues sobreajusta los pesos de las neuronas de modo que no generalizan la información contenida en los datos del sistema, sino que se particularizan y especializan en obtener mínimo error concretamente en ese fichero de datos de entrenamiento. Su comportamiento frente a nuevos datos, diferentes a aquellos con los que se realiza el aprendizaje es sensiblemente peor que el obtenido en un entrenamiento sin sobreajuste. La regla de detención del aprendizaje es entonces mínimo error en el patrón de test, pues en ese instante la red está obteniendo minimizar el error frente a datos que no intervinieron en el aprendizaje.

Habitualmente se dividen los datos disponibles entre el fichero de patrones de entrenamiento y el fichero de patrones de test en porcentajes próximos a $\frac{2}{3}$ y $\frac{1}{3}$, respectivamente. La selección de qué vectores de entrada formarán parte de cada uno de estos ficheros es aleatoria. Para que los resultados del entrenamiento no dependan de una determinada selección, que al ser aleatoria pudiera no ser idónea en el contenido de su información, se genera por triplicado esta pareja de ficheros y se lleva a cabo el entrenamiento en estas tres parejas, seleccionando finalmente la red que mejor se haya adaptado a la información contenida en los datos.

La generación de los patrones puede realizarse en el entorno **R**. Es necesario, pues así lo requiere SNNS, que los datos se introduzcan normalizados por variables, de tal forma que el intervalo de valores en el que se encuentren los datos de cada variable sea [0.1, 0.9], para evitar las saturaciones de las funciones de activación de las neuronas, y procurar en todo momento trabajar

en zona lineal. Para normalizar por columnas puede implementarse una función como la aquí desarrollada, *normaliza.col*.

```
> normaliza.col <- function(data.in, a=0, b=1)
+ {
+  max.col <- min.col <- 0
+  data.out <- array(0,dim(data.in))
+ for (i in 1:ncol(data.in))
+ {
+  max.col[i] <- max(data.in[,i])
+  min.col[i] <- min(data.in[,i])
+  data.out[,i] <- a + (data.in[,i] - min.col[i])*(b-a)/
+                  ( max.col[i]-min.col[i])
+ }
+ return(data.out)
+}
```

Con ella se pueden normalizar entre 0.1 y 0.9 los datos del proceso.

```
> datos.red.normalizados <- normaliza.col(datos.red)
> apply(datos.red.normalizados,2,max)
[1] 1 1 1 1 1 1 1 1 1 1
> apply(datos.red.normalizados,2,min)
[1] 0 0 0 0 0 0 0 0 0 0
```

Una vez normalizados se puede generar el fichero de patrones. Resulta necesario primeramente dividir aleatóriamente los datos en dos tipos de patrones, los de entrenamiento y los de test. Para ello es útil recordar la función *subsample*, que dividía aleatóriamente un conjunto de datos en dos.

```
> subsample <- function(data.in, cuantos)
+ {
+  aux <- cbind(runif(nrow(data.in)),data.in)
+  aux <- as.matrix(aux)
+  ordenado <- order(aux[,1])
+  aux <- as.matrix(aux[ordenado,-1])
+  return(list(A=aux[1:cuantos,],B=aux[(cuantos+1):nrow(aux),]))
+  }
```

De esta manera se dividen los datos normalizados en dos conjuntos del tamaño deseado. Se generan tres parejas de ficheros de entrenamiento y de test.

```
> aux <- subsample(datos.red.normalizados, 10000)
> TR_15_20_1.sup.1.pat  <- aux$A      #Patrón de entrenamiento.
> pTR_15_20_1.sup.1.pat <- aux$B      #Patrón de Test.

> aux <- subsample(datos.red.normalizados, 10000)
> TR_15_20_1.sup.2.pat  <- aux$A      #Patrón de entrenamiento.
> pTR_15_20_1.sup.2.pat <- aux$B      #Patrón de test.
```

```
> aux <- subsample(datos.red.normalizados, 10000)
> TR_15_20_1.sup.3.pat  <- aux$A      #Patrón de entrenamiento.
> pTR_15_20_1.sup.3.pat <- aux$B      #Patrón de test.
```

Con ello tan solo falta generar los ficheros correspondientes en el formato propio de SNNS. Para ello se ha programado la función *data2snns*.

```
> data2snns<- function(file.name,data.matrix,number.of.output.units=1)
 {
  file.create(file.name)
  cat(" SNNS pattern definition file v3.2\n",file=file.name,append= T)
  cat(paste("  generated at ", date(), "\n\n\n\n\n\n\n"),
      file= file.name, append = T)
  cat(paste(" No. of patterns      :", nrow(data.matrix), "\n"),
       file = file.name, append = T)
  cat(paste(" No. of input units   :", ncol(data.matrix) -
       number.of.output.units, "\n"), file = file.name, append = T)
  cat(paste(" No. of output units  :", number.of.output.units,
       "\n\n"), file = file.name, append = T)
  for (i in 1:nrow(data.matrix)) {
    cat(paste("\n#", i, "\n"), file = file.name, append = T)
    numcol <- ncol(data.matrix)
    while (numcol > 10) {
       line <- as.character(data.matrix[i, 1])
      for (j in 2:10) line <- paste(line, data.matrix[i,j])
          cat(line, file = file.name, append = T)
      cat("\n", file = file.name, append = T)
      numcol <- numcol - 10
    }
    line <- as.character(data.matrix[i, 1])
    for (j in 2:numcol) line <- paste(line, data.matrix[i,j])
      cat(line, file = file.name, append = T)
    cat("\n", file = file.name, append = T)
  }
 }
```

La función crea un fichero de patrones a partir de un objeto tipo *matrix* o *data.frame*, adecuado para su utilización con el software SNNS. La función toma como argumentos el nombre del fichero de patrones a generar y el número de neuronas en la capa de salida de la red.

```
> data2snns("TR_15_20_1.sup.1.pat", 1)
> data2snns("TR_15_20_1.sup.2.pat", 1)
> data2snns("TR_15_20_1.sup.3.pat", 1)
> data2snns("pTR_15_20_1.sup.1.pat", 1)
> data2snns("pTR_15_20_1.sup.2.pat", 1)
> data2snns("pTR_15_20_1.sup.3.pat", 1)
```

De esta manera se procede a entrenar la red. El método elegido no es el interactivo, sino a través del interprete de comandos batchman. Este interprete procesa ficheros cuyo contenido especifica la sucesión de ordenes necesarias para el entrenamiento de la red. Entre estas instrucciones se hallan las necesarias para contemplar la grabación del estado de la red en aquellos momentos en que se minimiza el error en los patrones de test.

El fichero de instrucciones considera que los archivos guardan una cierta estructura dentro del disco. Tal y como está programado el fichero de instrucciones es necesaria la presencia de directorios denominados BATCH, REDES, PATRONES y RESULTADOS, que contendrán los ficheros de ordenes de aprendizaje, los ficheros con las redes entrenadas y sin entrenar, los patrones de aprendizaje, test y validación, y los resultados obtenidos en el aprendizaje, respectivamente.

El fichero del ejemplo contiene una cabecera configurable que indica cual es la raiz del nombre de los ficheros de entrenamiento (*nombre*), cuantos ciclos se desea entrenar (*maxciclos*) y el número de ciclos existente entre distintas comprobaciones del error cometido en el fichero de patrones de test (num_ciclos_seg).

Si como en el ejemplo se asigna a la variable *nombre* la cadena *TR_15_20_1,* deberán existir los siguientes ficheros en los directorios:

- `../BATCH/TR_15_20_1_300000.bat`
- `../REDES/TR_15_20_1.net`
- `../PATRONES/TR_15_20_1.sup.1.pat`
- `../PATRONES/TR_15_20_1.sup.2.pat`
- `../PATRONES/TR_15_20_1.sup.3.pat`
- `../PATRONES/pTR_15_20_1.sup.1.pat`
- `../PATRONES/pTR_15_20_1.sup.2.pat`
- `../PATRONES/pTR_15_20_1.sup.3.pat`

La red es sometida al algoritmo de aprendizaje hasta alcanzar un número máximo de iteraciones, generando un fichero de registro de las operaciones realizadas y los resultados obtenidos:

```
$ nohup batchman -q -l TR_15_20_1_300000.log -f TR_15_20_1_300000.bat &
```

```
############################################################
# Lanzador de redes con control de error en fichero de test
############################################################

#Parametros definidos por el usuario

nombre     = "TR_15_20_1"
maxciclos  = 300000
num_ciclos_seg  = 50
num_parejas = 3
#Codigo que procesara batchman

red= "../REDES/" + nombre + ".net"
print("#Fichero de Registro de la evolucion de la red neuronal
", red)

for conjunto_numero=1 to num_parejas do
     print("#Conjunto de datos numero: ",conjunto_numero)
     patron_de_entrenamiento= "../PATRONES/" +  nombre + ".sup."
+ conjunto_numero + ".pat"
     patron_de_test=  "../PATRONES/p"  +   nombre  +  ".sup."  +
conjunto_numero + ".pat"
     loadNet(red)
     loadPattern(patron_de_test)
     loadPattern(patron_de_entrenamiento)
     setInitFunc("Randomize_Weights", 0.3, -0.3)
     setLearnFunc("BackpropMomentum")
     initNet()

     minimo_error_de_test= 9999999999999999
     error_de_test_anterior= 99999999999999
     error_de_test= 9999999999999999
     anterior_fue_minimo=FALSE
     este_fue_minimo=FALSE
     while (CYCLES < maxciclos) do
        for contador= 1 to num_ciclos_seg do
          trainNet()
        endfor
        testNet()
        error_de_entrenamiento= MSE
        setPattern(patron_de_test)
        testNet()
        error_de_test= MSE
       print (CYCLES, ",Test_MSE ," ,error_de_test, ",
              Entrenamiento_MSE ," ,error_de_entrenamiento)
```

```
        if (error_de_test < minimo_error_de_test) then
          minimo_error_de_test=error_de_test
          este_fue_minimo=TRUE
          print("#Error minimo en ", CYCLES, " Test_MSE: ", MSE)
        else
           este_fue_minimo=FALSE
        endif
           if  ((error_de_test  >  error_de_test_anterior)  and
(anterior_fue_minimo == TRUE)) then
          print("Guardo esta red. El error de test fue mínimo
                 hasta el momento")
           red_entrenada= "../REDES/" + nombre + "_co_" + CYCLES
+ "_" + conjunto_numero + ".net"
          saveNet(red_entrenada)
        endif
       setPattern(patron_de_entrenamiento)
       error_de_test_anterior= error_de_test
       anterior_fue_minimo = este_fue_minimo
     endwhile
     red_final= "../REDES/" + nombre + "_final_" + CYCLES + "_"
+ conjunto_numero + ".net"
     saveNet(red_final)
     delPattern(patron_de_test)
     delPattern(patron_de_entrenamiento)
endfor
```

Fichero de entrenamiento de la red neuronal.

Durante el aprendizaje se generan diversos ficheros del tipo:

- `../REDES/TR_15_20_1_co_15928_1.net`
- `../REDES/TR_15_20_1_co_17502_1.net`
- `../REDES/TR_15_20_1_co_103867_1.net`
- `../REDES/TR_15_20_1_co_158225_1.net`

que contienen los estados de la red neuronal en aquellos puntos que hasta el momento habían alcanzado mínimo error en el fichero de patrones de test. Asimismo se genera un fichero al alcanzar el número máximo de ciclos, por si resultase deseable prolongar el aprendizaje de la red partiendo de este punto.

- `../REDES/TR_final_300000_1.net`

Tras analizar los errores cometidos por la red neuronal entrenada con mínimo error en el patrón de test (../REDES/TR_15_20_1_co_158225_1.net), se llega a los resultados que se indican en la Tabla 8, donde se compara el error cometido por el modelo en uso por la empresa siderurgico (Pág. 169), el error

obtenido con los modelos lineales (l*m3.som*) y el error obtenido por la aproximación neuronal.

Error	Modelo "Teórico"	Modelo Lineal	Modelo Neuronal
Máximo	232	215.88	136.28
Media	18.22	19.88	14.83
Desviación	19.56	18.04	13.58

Tabla 8

El modelo implementado pretende mejorar el valor medio del error precalculado sin que ello suponga un aumento de la desviación ni del valor máximo y se consigue una cierta mejora. En la Fig. 73 se muestra en nube de puntos los resultados del patrón de test. Analizando la figura se ve la mejora conseguida. Los errores del precálculo suelen ser por exceso mientras que el modelo actual comete más error por defecto.

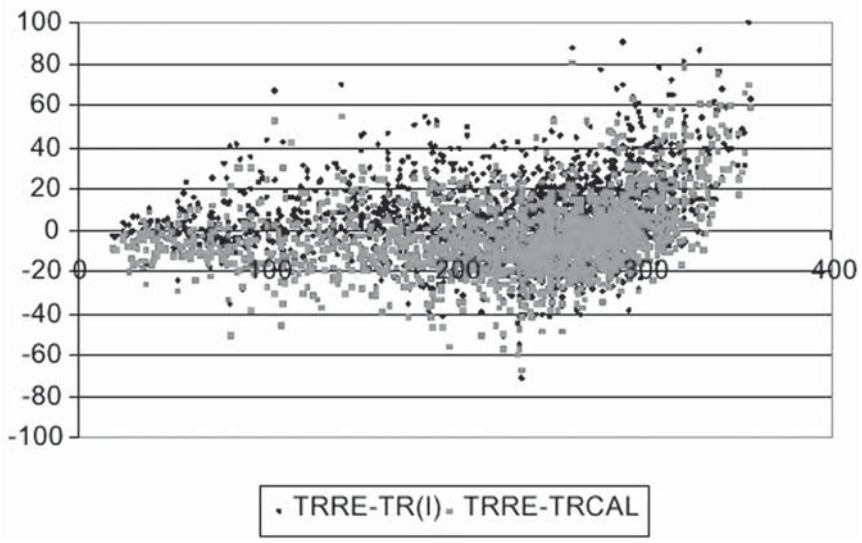


Fig 73: Residuos del modelo neuronal

Capítulo 12

Conclusiones finales

A lo largo del trabajo se han fundamentado e implementado en **R** las distintas herramientas de análisis multivariado aplicado a la modelización de procesos productivos, pieza clave del control de calidad industrial.

En el caso de técnicas no aplicables a los datos muestrales recogidos del proceso se ha procurado exponer, mediante datos ficticios a modo de ejemplo, el tratamiento que debería aplicarse a los datos procedentes de otros procesos donde dichas variantes de las técnicas utilizadas sí tuvieran cabida.

Se ha procurado seguir un estricto orden metodológico que sirva de guía en futuros proyectos de investigación en este campo, lo cual redundará en beneficio de los nuevos investigadores que se inicien en este tipo de proyectos industriales, facilitándoles la aproximación a las técnicas y su implementación práctica.

En el caso concreto estudiado, un proceso siderurgico, se ha observado la falta de normalidad multivariante en la distribución de los datos observados, lo que ha promovido la investigación en aquellas técnicas aplicables cuando el requisito de la normalidad multivariante no se cumplía, lo cual ha proporcionado al trabajo una mayor generalidad y un espectro más amplio de aplicación.

En particular se ha observado como la aplicación de algoritmos de clasificación que agrupan los datos observados en distintos modos de funcionamiento, redundan, efectivamente, en una minimización del error obtenido en cada una de las clases frente al error que se cometería al considerar un único modelo para todos los puntos del espacio de estados del sistema.

Al tratarse de un sistema real los resultados obtenidos con modelos lineales son de inferior calidad que los correspondientes a los modelos no lineales, pues estos recogen con mayor flexibilidad la información contenida en las obsevaciones muestrales.

Finalmente, gracias a la utilización de redes neuronales como aproximadores no lineales de la variable a explicar, se ha conseguido mejorar notablemente las estimaciones correspondientes al par motor aplicado en los rodillos de laminación, no sólo en error medio, sino en desviación típica del error y en magnitud máxima del mismo, lo cual supone un importante avance en la calidad del proceso productivo y un considerable ahorro energético y, por tanto, también económico.

Capítulo 13

Bibliografía

1. J.W. Sammon, Jr. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, C-18:401-409, 1969

2. Biswas, G., Jain, A.K., y Dubes, R.C. Evaluation of projection algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(6):701-708, 1981.

3. Kraaijveld, M.A., Mao, J., y Jain, A.K. A nonlinear projection method based on kohonen's topology preserving maps. *IEEE transactions on neural networks*, 6(6):548-559, 1995.

4. Mao, J. y Jain, A.K. Artificial neural networks for feature extraction and multivariate data projection. *IEEE Transactions on Neural Networks*, 6(2):296-317, march 1995.

5. R.C.T. Lee, J.R. Slagle, y H. Blum. A triangulation method for the sequential mapping of points from n-space to two-space. *IEEE Transactions on Computers*, C-26:288-292, 1977.

6. D. de Ridder y R.P.W. Duin. Sammon's mapping using neural networks: a comparison. *Pattern Recognition Letters*, 18(11-13):1307-1316, 1997.

7. M.F. Müller. Efficient Training of Feed-Forward Neural Networks. PhD thesis, Computer Science Department, Aarhus University, 1993.

8. Mandelbrot Benoit, La geometría fractal de la naturaleza / Benoît B. Barcelona: Tusquets, 1997 . ISBN 84-8310-549-7.

9. Faloutsos C y Gaede V. Analysis of n-dimensional quadtrees using the Hausdorff fractal dimension. VLDB 1996.

10. Belussi A. y Faloutsos C. Estimating the selectivity of spatial queries using the correlation fractal dimension. VLDB 1995.

11. Jobson. Multivariate Data Analysis. Volume I.

12. Jobson. Multivariate Data Analysis. Volume II.

13. Stephans, M.A. (1974) “EDF Statistics for Goodness of Fit and Some Comparisons”, Journal of American Statistical Association 69, 730-737.

14. S-PLUS 2000, Guide to Statistics, Volume 1. Data Analysis Products Division, MathSoft, Seattle, WA.

15. S-PLUS 2000, Guide to Statistics, Volume 2. Data Analysis Products Division, MathSoft, Seattle, WA.

16. W.J. Conover, Mark E. Jonson & Myrie M.Jonson (1981). A comparative study of test for homogeneity of variances, wirh applications to the outer continental shelf bidding data, Technometrics 23, 351-361.

17. J.F. Hair et al. (1999). Análisis multivariante. Prentice Hall Iberia.

18. F.J. Martinez de Pisón, J. Ordieres, M. Castejón. Estudio de la dimensión fractal como medida de la dimensión interna de los datos. Publicación interna DIM. 2000.

19. Durbin, J. and Watson, G.S., "Testing for Serial Correlation in Least Squares Regression I", Biometrika, Vol. 37, 1950, pp. 409-428.

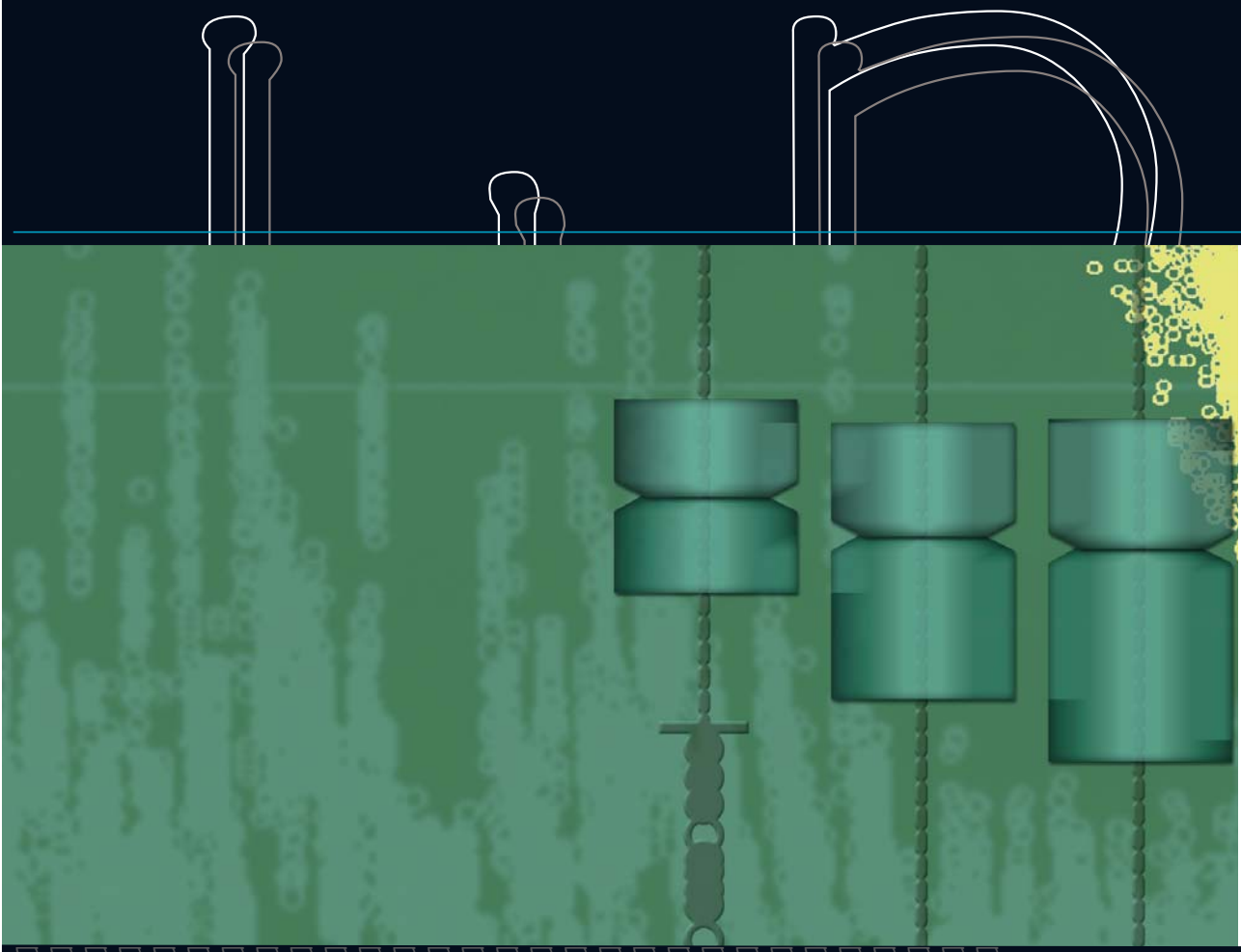
20. Barnett, V.; Lewis, T. *Outliers in Statistical Data,* 3rd ed.; John Wiley & Sons: New York, 1994: p 7.

21. Rousseauw, P.J. *Journal of American Statistycal Association*. 1984, 79. 871-880.

22. Atkinson, A.C. *Biometrika*.1986, 73, 533-541.

23. Gnanadesikan, R.; Kettering, J.R. *Biometrics* 1972, 28, 81-124.

24. Rocke, D.M.; Woodruff D.L. *Identification of Outliers in Multivariate Data.* Journal of the American tatisticaal Association. 1996, Vol 91, No. 435, Theory and Methods



UNIVERSIDAD
DE LA RIOJA