S14 - *Kernel PCA*

Juan Carlos Martinez-Ovando
ITAM

En la sesion de hoy estudiaremos una variante del analisis de componentes principales basado en la nocion de similaridad en los datos, el cual puede ser asociado con estructuras complejas de datos.

Keywords: Kernel methods, singular value decomposition, manifolds.

Intuicion en PCA

Recordemos que PCA es un procedimiento de ortogonalizacion de una matriz de datos $Y_{(n\times p)}$, con $n >> p$, basada en la descomposicion en valores singulares,

$$Y_{(n\times p)} = U_{(n\times p)} D_{(p\times p)} V_{(p\times p)}.$$

A partir de esta descomposicion, podemos calcular las siguientes matrices cuadraticas

$$\begin{aligned} S_{(p\times p)} &= Y'_{(n\times p)} Y_{(n\times p)} = V'_{(p\times p)} D^2_{(p\times p)} V_{(p\times p)} && (1)\\ K_{(n\times n)} &= Y_{(n\times p)} Y'_{(n\times p)} = U_{(n\times p)} D^2_{(p\times p)} U'_{(p\times n)} && (2) \end{aligned}$$

La matrix S corresponde a la sum de cuadrados de Y –cuando los datos han sido estandarizados previamente–, mientras que la matriz K es referida como la *matriz de Gram*.

Recordemos que el primer componente principalesta dado por la siguiente transformacion

$$\begin{aligned} f_1 &= Yv_1 && (3)\\ &= UDV'v_1 && (4)\\ &= u_1 d_1, && (5) \end{aligned}$$

donde v_1 es un vector de dimension $(p \times 1)$ correspondiente al eigenvector asociado con el primer eigenvalor de Y.

Como sabemos, el primer componente principal puede obtenerse de tres formas alternaivas:

a. Como el producto de Y con el primer eigenvector de S

b. A partir de la descomposicion en valores singulaes de Y (descrito lineas arriba)

c. A traves de la descomposicion singular de K.

Asi, pues no se necesita saber Y directamente, sino que basta con conocer S o K para producir los componentes principales de un conjunto de datos. En particular, el primer componente principal de un vector p-dimensional y puede obtenerse como la proyeccion sobre eleje del primer componente, i.e.

$$f = v'_1 y.$$

la expresion anterior puede calcularse directamente, o *indirectamente* empleando la expresion alternativa

$$f = u_1' Y y / d_1 = \sum_{i=1}^{n} \left(\frac{u_{i1}}{d_1} \right) y_i' y.$$

Expresiones semjantes se obtienen de manera analoga para los demas componente principales (no solo el primero).

De esta forma podemos ver que es necesario conocer los productor interiores $(y_i' y)_{i=1}^n$ solamente.

Como antes mencionamos, el calculo de f_1 puede obtenerse de dos formas:

`Forma.1`.- Calcular $S = Y'Y$, obteniendo el primer eigenvector de esta matriz, v_i de S, y calcular

$$f_1 = Y v_1.$$

`Forma.2`.- Empleando la matriz de Gram, calculando YY', obteniendo el primer eigenvector de esta matriz, u_1 y su correspondiente eigenvalor, d_1, y calculando

$$f_1 = u_1 d_1.$$

La `Forma.1` es particularmente util cuando $n >> p$, mientras que la `Forma.2` lo es para el caso $n << p$.

Al final del dia, PCA descansa en el calculo de los prodcutos interiores $(y_i' y)_{i=1}^n$, el cual puede interpretarse como una medida de 'similaridad euclidiana' entre objetos p-dimensionales.

La idea entonces de **Kernel PCA** es la de relajar el supuesto de "similaridad euclidiano" para otras medidas de similaridad. Esto en particular cuando los objetos y residan en *sub-espacios no lienales* de R^p (curvas, superficies o *manifolds*).

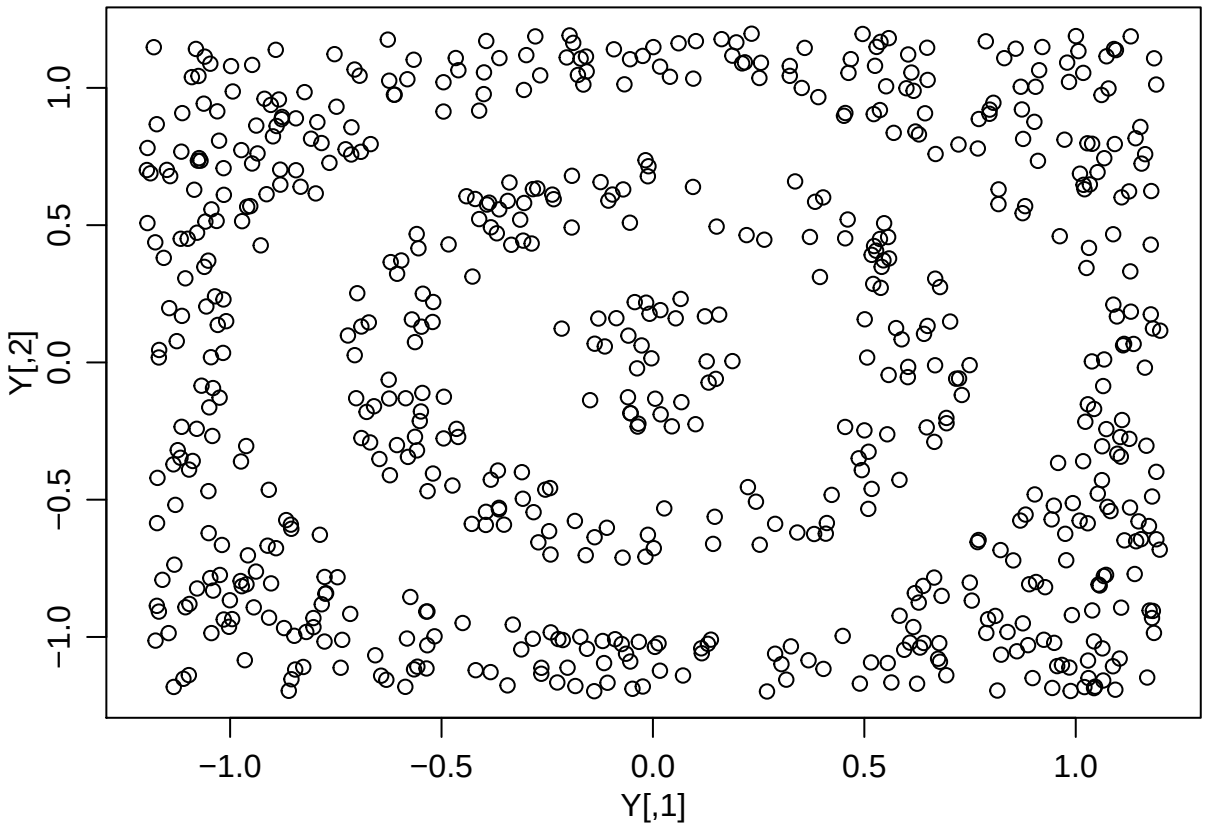
Caracterizacion

¿Como son esos sub-espacios?

Veamos un ejemplo con el siguiente diagrama de datos sinteticos.

```
set.seed(1)
n <- 1000
Y <- matrix(runif(n*2,-1.2,1.2),n,2)
r <- sqrt(apply(Y^2,1,sum))
Y <- Y[  r<.25 | (r>.5 & r<.75) | r>1   ,]
r <- sqrt(apply(Y^2,1,sum))

r <- sqrt(apply(Y^2,1,sum))
clr <- rgb( (r/max(r))^.7,(1-r/max(r))^.7,.5)
par(mar=c(3,3,1,1),mgp=c(1.75,.75,0))
#plot(Y,col=clr)
plot(Y)
```

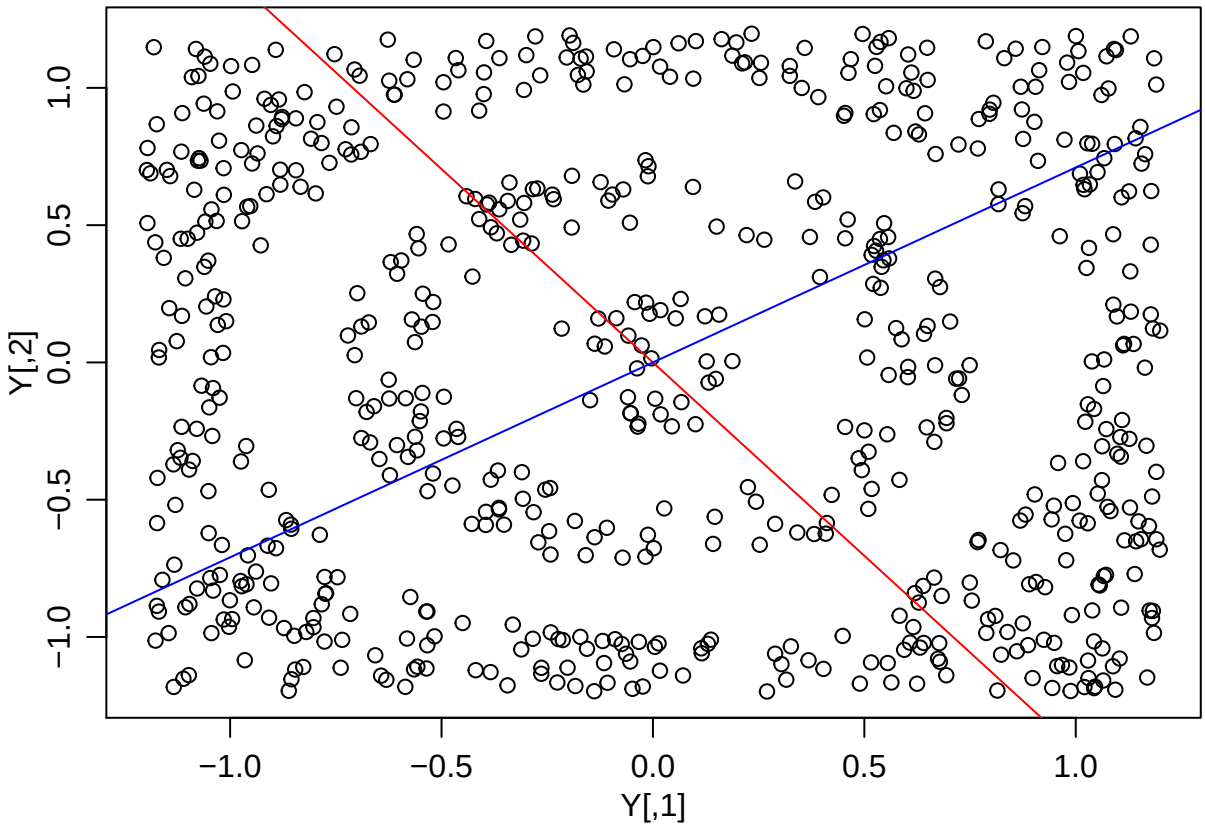


En este caso, la matriz de Gram deseable sera tal que mida alternativamente la similaridad entre las y_is.

PCA convencional

En caso de realizar PCA convencional en este conjunto de datos, se obtendrian resultados confusos, pues al parecer no habria ortogonalizacion que realizar. Veamos los siguientes resultados.

```
par(mar=c(3,3,1,1),mgp=c(1.75,.75,0))
plot(Y)
#plot(Y,col=clr)
sY <- svd(Y)
V <- sY$v
abline(0,V[2,1]/V[1,1],col="red")
abline(0,V[2,2]/V[1,2],col="blue")
```

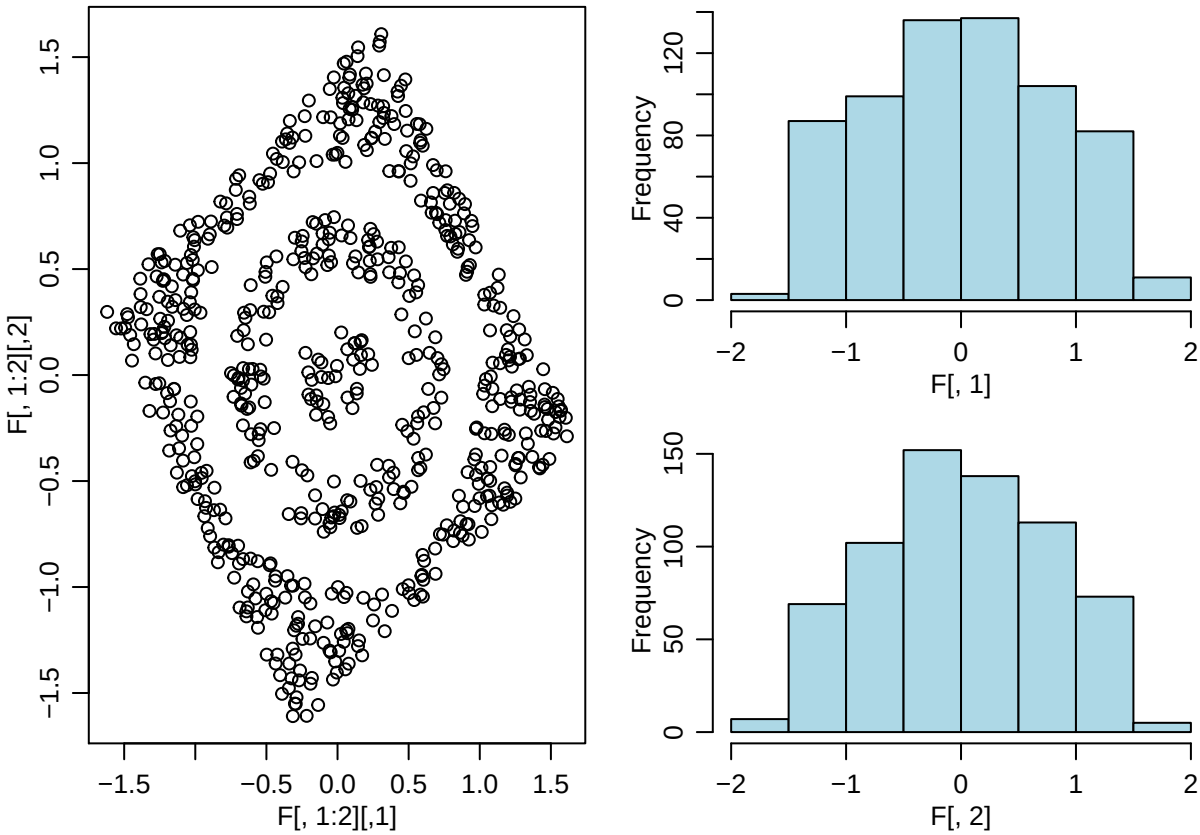


En el grafico anterior, las rectas representan los *ejes del PCA*. Como observamos, los datos de *componentes principales* son iguales a Y. Esto es porque la *medida de similaridad* empleada es la euclidiana. El resultado es, en este caso, una rotacion de Y solamente.

```
sY <- svd(Y)
F <- sY$u%*%diag(sY$d)

par(mar=c(3,3,1,1),mgp=c(1.75,.75,0))
layout(matrix(c(1,1,2,3),2,2))
plot(F[,1:2])

hist(F[,1],main="",col="lightblue")
hist(F[,2],main="",col="lightblue")
```



Nota: El resultado anterior se obtiene adoptando la matriz $K = YY'$ como la matriz de Gram (i.e. la matriz de similaridad entre datos y_i).

PCA por kernel

Ahora, si modificamos la no nocion de similaridad por la siguiente metrica,

$$d(y_i, y_j) = \left(y_i' y_j + 1\right)^2,$$

la matriz de Gram asociada seria

$$K = \left(YY' + 1\right)^2.$$

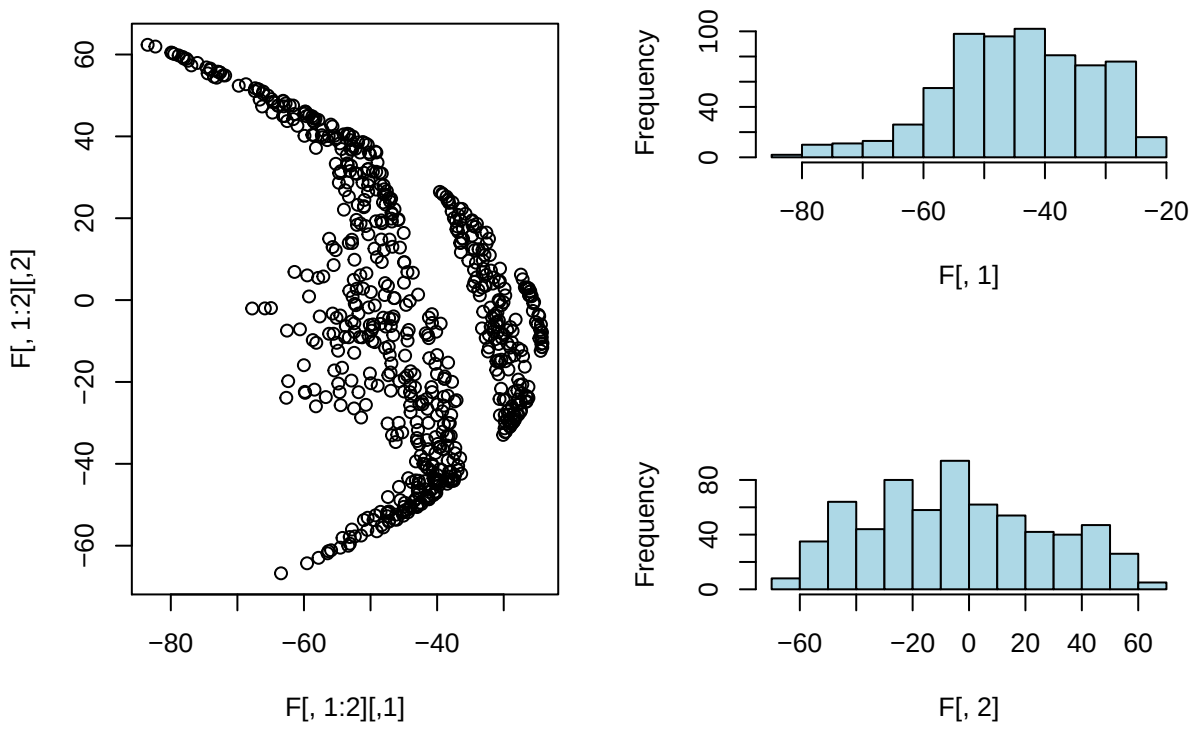
En este caso, realizando la descomposicion singular de K resultaria en el siguiente PCA.

```
K <- (tcrossprod(Y) + 1)^2

sK <- svd(K)
F <- sK$u%*%diag(sK$d)

layout(matrix(c(1,1,2,3),2,2))
plot(F[,1:2])

hist(F[,1],main="",col="lightblue")
hist(F[,2],main="",col="lightblue")
```

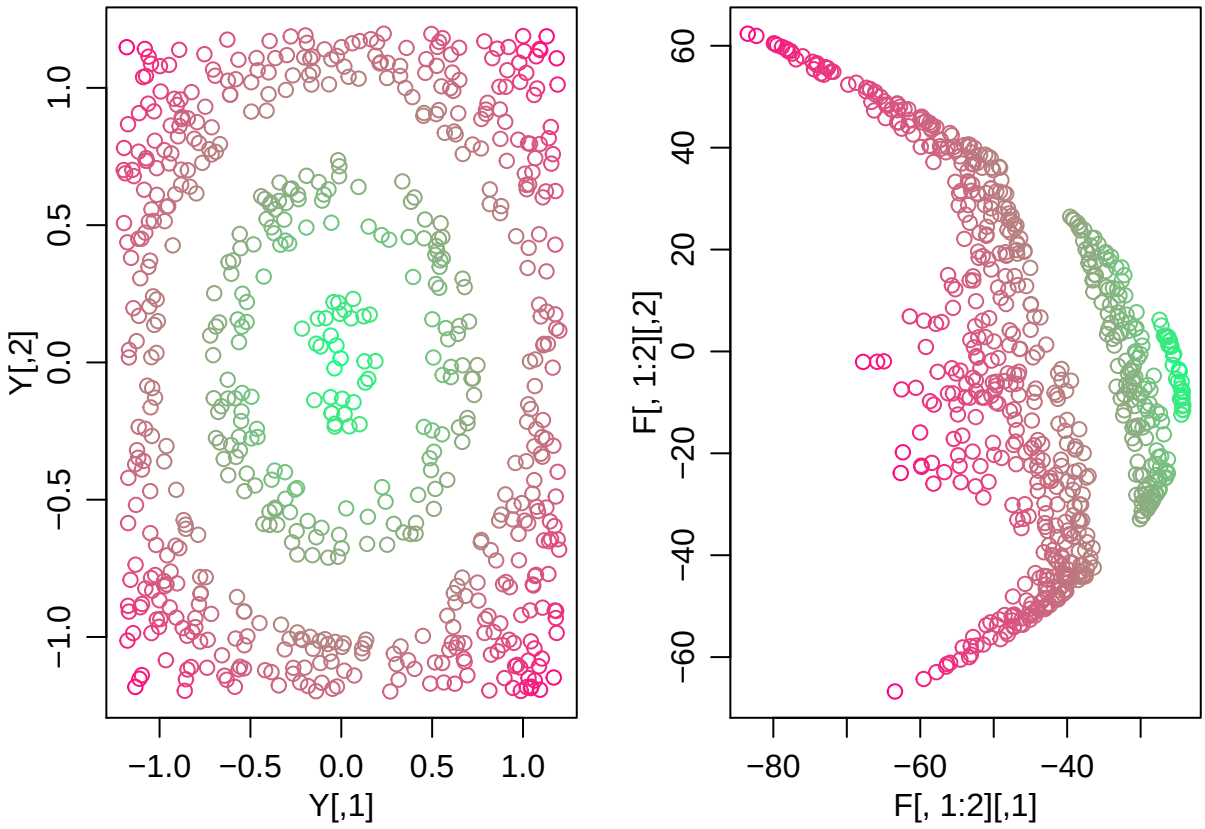


Como es evidente, los datos tranformados (panel izquierdo en la grafica), responden la evidente separacion de anillos que visualiamos en los datos originales.

Las proyecciones de los datos en los *nuevos ejes principales* es tambien distinta, mostrando sesgo en este caso.

La asociacion entre las regiones de los datos originales, Y, y los datos transformados, F, se muestra en el siguiente grafico.

```
par(mfrow=c(1,2),mar=c(3,3,1,1),mgp=c(1.75,.75,0))
plot(Y,col=clr)
plot(F[,1:2],col=clr)
```



A continuacion describimos el marco general de descomposicion PCA basado en kernels.

Kernel PCA

Al modificar la nocion de similaridad como antes, se introduce la nocion de *kernel*. en un contexto general, el *kernel* servira como la medida de similaridad. Asi, entre dos puntos y_i y y_j la similaridad en kernel se definira como

$$k(y_i, y_j) = \phi(y_i)'\phi(y_j),$$

donde ϕ es una funcion que mapea de $\mathbb{R}^p$ a $\mathbb{R}^q$ (donde tipicamente $q >> p$).

El kernel, asi definido, puede interpretarse como el producto interioreuclidiano no de los datos originales y_is, sino de los datos modificados por una cierta funcion ϕ.

En el ejemplo anterior, la funcion ϕ empleada es

$$\phi(y) = \left(1, \sqrt{(2)}y_1, \ldots, \sqrt{(2)}y_p, y_1^2, \ldots, y_p^2, \sqrt{2}y_1y_2, \ldots, \sqrt{2}y_{p-1}y_p\right).$$

Ejercicio: Verifiquen que $d(y_i, y_j)$ visto antes es igual a $\phi(y_i)'\phi(y_j)$ para todo y_i, y_j en $\mathbb{R}^p$.

Contextos

¿En que contextos *Kernel PCA* es util?

- Procesamiento de textos

- Procesamiento de imagenes
- Procesamiento de audio

En `R` se pueden encontrar varias implementaciones confiables en el paquete '`kernlab`.

Observacion: Casi todos los procedimientoes vistos en este curso donde se emplea el producto interior euclidiano pueden modificarse para definirse en terminos de *kernels*. Regresion es un ejemplo, derivando en modelos de expansiones de bases de kernels. Eso lo veremos en la siguiente semana.

Referencias

- **Wang** (2014) *Kernel Principal Component Analysis and its Applications in Face Recognition and Shape Models*
- **Bishop** (2006) *Pattern Recognition and Machine Learning*