Universitat Politècnica de Catalunya

Visió per Computador
Grau en Enginyeria Informàtica

EXERCICI 1 DE LABORATORI

PRINCIPAL COMPONENT ANALYSIS

Autor:
Daniel DONATE

Professor:
Manuel FRIGOLA
Q2 Curs 2020-2021

24 de febrer del 2021

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

Enunciat de l'exercici

L'objectiu del problema és transformar un núvol de punts aleatoris (Fig. 1-a) per a què estigui centrat en l'eix horitzontal (Fig. 1-b), tal i com es suggereix a les següents figures.

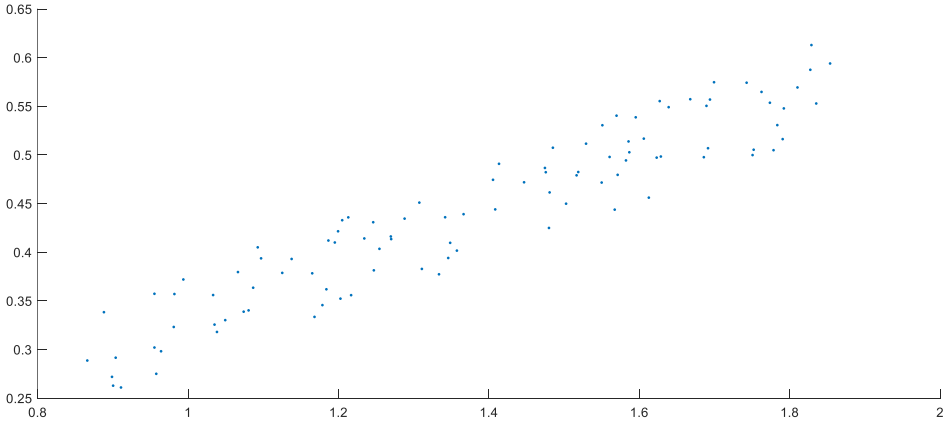


Figura 1-a: Núvol de punts aleatori

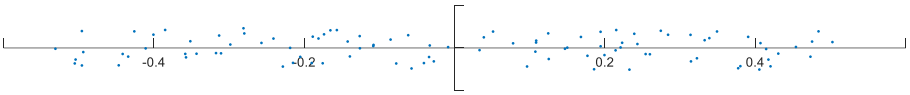


Figura 1-b: Núvol de punts de 1-a centrat en l'eix horitzontal (objectiu de l'exercici)

Solució en 2D

La solució proposada a l'enunciat per resoldre aquest exercici es basa en una tècnica que no s'esmenta en el document i que rep el nom de PCA (anàlisi de components principals). Per això, començarem l'informe donant una breu descripció del funcionament de la tècnica mitjançant un exemple[1] que penso que està molt ben construït i que ens servirà per justificar la validesa de la nostra solució.

Imaginem que tenim un conjunt de vins i volem, per cadascun d'ells, elaborar una llista amb les seves *característiques* (color, edat, acidesa, etc). Notem que podem fer servir una gran quantitat de característiques, però moltes d'elles mesuraran propietats relacionades i, per tant, seran redundants. La tècnica PCA ens permet *resumir* el producte amb les *millors* característiques possibles, fent ús de combinacions lineals de les propietats.

Observem que la nostra resposta ha de contenir propietats del vi que difereixin molt entre diferents vins. Si utilitzéssim una propietat que és comuna a casi tots els vins, aleshores aquesta propietat, sens dubte, no aportaria informació útil en el nostre resum. Per això, la tècnica PCA busca propietats que mostrin tanta **variació** entre vins com sigui possible. Per exemple, considerem dues propietats concretes del vi, com ara la foscor del vi (eix x) i el contingut alcohòlic (eix y). Un *scatter plot* de diferents vins (notem que cada punt blau de la figura és un vi) podria ser com el que es mostra a continuació:

[1] L'explicació donada sobre la tècnica PCA està inspirada en un exemple que es desenvolupa al respecte a la xarxa StackExchange. Pot trobar-se a: https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues

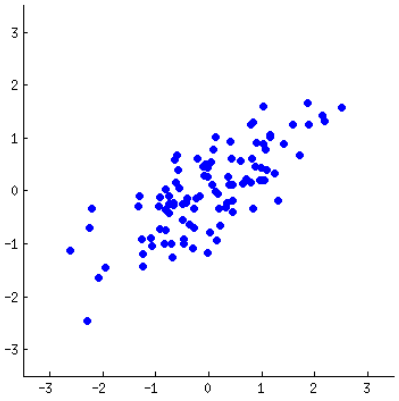


Figura 2: Núvol de punts de diferents vins, segons dos de les seves propietats

Com veiem, el núvol suggereix que les dues característiques tenen **correlació**. De manera que podem construir una nova propietat a partir de les dues traçant una recta a través del centre del núvol i projectant tots els punts sobre aquesta recta. Aquesta nova propietat ve donada per una combinació lineal $w_1 x + w_2 y$, essent cada recta una assignació particular de w_1 i w_2.
Així, podem veure que la tècnica PCA trobarà la *millor* recta d'acord amb el criteri que hem anunciat abans. És a dir, la variació dels valors en aquesta línia serà màxima. En les següents imatges podem veure com canvia la **variància** dels punts vermells (punts projectats en la recta) a mesura que rotem la recta.

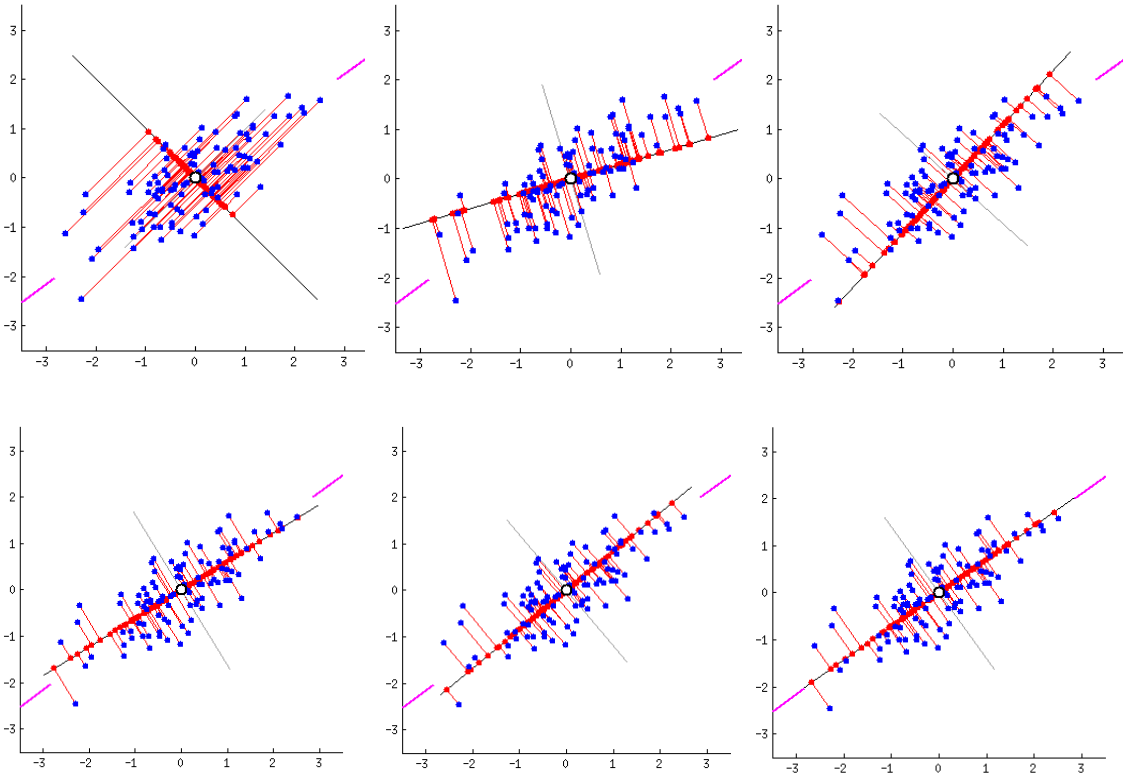


Figura 3: Projeccions dels vins sobre diferents rectes que passen pel centre del núvol

D'altra banda, si *reconstruïm* les dues característiques originals d'un vi (punt blau) a partir d'una nova posició (punt vermell), l'error de la reconstrucció vindrà donat per la recta vermella que projecta el punts blau. És fàcil veure que la *màxima variació* i *l'error mínim* s'assoleixen, simultàniament, quan la recta passa per les marques magenta de la imatge (figura d'avall a la

dreta). Aquesta recta es correspon a una nova propietat dels vins construïda per la PCA que anomenem ***component principal***.
Matemàticament, l'extensió de cada punt vermell es mesura com la distància quadràtica mitjana des del centre del núvol fins a cada punt vermell. D'això se'n diu **variància**. L'error, d'altra banda, és la longitud quadràtica mitjana de les corresponents rectes vermelles. Com l'angle entre les rectes vermelles i la recta negre és sempre $\pi/2$, la suma d'aquestes dues quantitats és igual a la distància promig al quadrat entre el centre del núvol i cada punt blau. Aquesta distància no depèn de la orientació de la recta negra, és constant, de manera que com més gran és la variància, menor és l'error.
Per una gràfica amb dos dimensions, com la que considerem, tindrem una matriu de covariància de 2x2 amb les següents entrades:

$$C = \begin{bmatrix} Var(x) & Cov(x,y) \\ Cov(y,x) & Var(y) \end{bmatrix}$$

Concretament, pel núvol de punts de la Figura 2, tenim una matriu amb aquests valors:

$$C = \begin{bmatrix} 1.07 & 0.63 \\ 0.63 & 0.64 \end{bmatrix}$$

Com *C* és una matriu quadrada simètrica, aquesta pot diagonalitzar-se escollint un nou sistema de coordenades ortogonals donat pels seus *eigenvectors*; els *valors propis* corresponents s'ubicaran en la diagonal (teorema de descomposició espectral). En aquest nou sistema de coordenades, *C* es transforma en *C'*, que és diagonal i té aquests valors:

$$C' = \begin{bmatrix} 1.52 & 0 \\ 0 & 0.19 \end{bmatrix}$$

Ara és clar que la variància de qualsevol projecció vindrà donada per una mitjana ponderada dels *eigenvalues*. En conseqüència, la màxima variància possible (1.52) s'assolirà si prenem la projecció en el primer eix de coordenades. D'aquí es dedueix que **la direcció del primer component principal ve donada pel primer eigenvector de C**.

El procés que hem descrit mitjançant l'exemple dels vins és el mateix que es porta a terme en aquestes línies de MATLAB. Estem calculant la component principal del núvol de punts aleatoris que hem generat en les línies anteriors.

```
15 -    c = cov(xp, yp);
16 -    [evectors, evalues] = eig(c);
17
18      % Determine which dimension has the major variance
19 -    [val,ind] = max(diag(evalues));
```

Un cop trobada la component principal, podem calcular l'angle θ (que forma la recta que representa aquesta component amb l'eix d'abscisses) per rotar cada punt del núvol i centrar-lo sobre l'eix horitzontal (*reduint-ne* la dimensió, com es demana a l'enunciat del problema). L'angle pot calcular-se com $\theta = \pi + \tan^{-1}(x_a/y_a)$, on x_a i y_a són, respectivament, el primer i segon component del vector *evector* corresponent al component principal. De forma equivalent, podem expressar l'angle com: $\theta = -\pi/2 - \tan^{-1}(y_a/x_a)$.

```
22 -    theta = -pi/2 - atan2(evectors(1,ind),evectors(2,ind));
```

Per realitzar la rotació esmentada, farem ús d'una matriu de rotació *R(θ)*, que representa una rotació del pla de θ graus en sentit antihorari. Aquesta matriu té els següents elements:

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

A mode de justificació informal, considerem la següent figura, que rota el nostre sistema de coordenades, en base canònica (blau) θ graus, fins al sistema de coordenades verd.

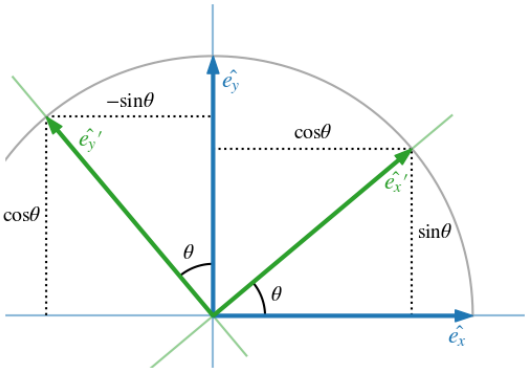


Figura 4: Rotació de θ graus dels vectors $\hat{e}_x$ i $\hat{e}_y$

Notem que, efectivament, en produir-se la rotació, els components (x, y) originals es transformen en els nous (x', y') com es suggereix al dibuix:

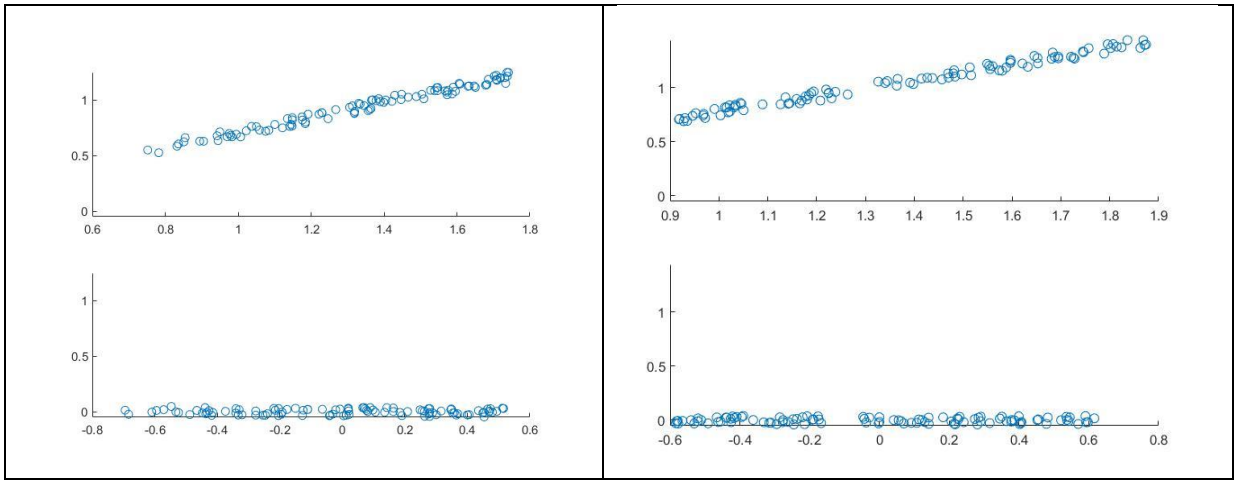
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} = \begin{bmatrix} x \cdot \cos(\theta) - y \cdot \sin(\theta) \\ x \cdot \sin(\theta) + y \cdot \cos(\theta) \end{bmatrix}$$

Notem, però, que la rotació que volem aplicar al nostre núvol de punts és una rotació en sentit horari, de manera que la matriu *R* que fem servir és lleugerament diferent:

$$R(-\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}$$

```
24    % Create clockwise rotation matrix
25    R = [cos(theta) sin(theta); -sin(theta) cos(theta)];
26
27    % Rotate the points
28    rp = R * [xp;yp];
```

Finalment, fem un *scatter plot* del nostre núvol original, junt amb el nostre núvol rotat, *rp*. Per evitar que es re-escalin les dades fem servir la funció *linkaxes* de MATLAB, que ens permet sincronitzar l'eix 'y' dels dos *plots*. A la següent figura mostrem diferents execucions.



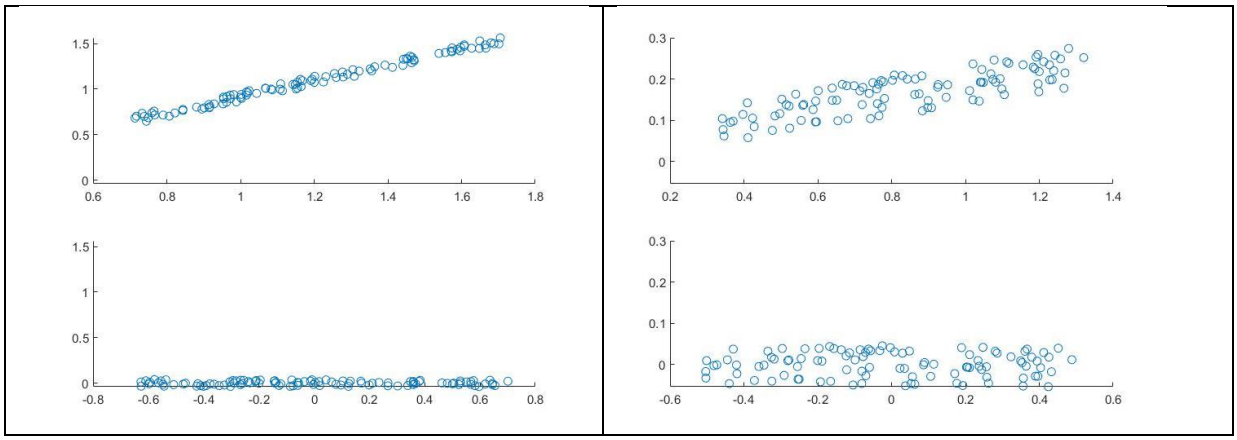


Figura 5: Execució del programa amb diferents núvols aleatoris

Com s'aprecia en els *subplots* inferiors, el núvol transformat (que està centrat en l'eix horitzontal a causa de la resta de la mitjana de les components 'x' a l'inici de l'script) *col·lapsa* en l'eix 'x' quan el núvol mostra una clara correlació, mentre que es presenta més *dispers* quan el núvol inicial no té una correlació lineal tan acusada (imatge d'avall, a la dreta).

Proposta d'extensió en 3D

El procediment que hem enunciat mitjançant l'exemple dels vins convida a pensar en una fàcil extensió de la tècnica a problemes amb més de dues variables. Si considerem un núvol en 3D, per exemple, aleshores enlloc de buscar fer una projecció sobre una recta, com hem fet amb el núvol 2D, el que faríem és una projecció sobre un pla, i obtindríem enlloc de dos, tres *eigen-vectors*, que apunten, respectivament, en la primera, segona i tercera direcció de major variància. Un exemple d'aquest procés es troba en la Figura 6, on l'autovector verd es correspon al primer dels *eigenvectors* descrits, el blau al segon, i el vermell al tercer.

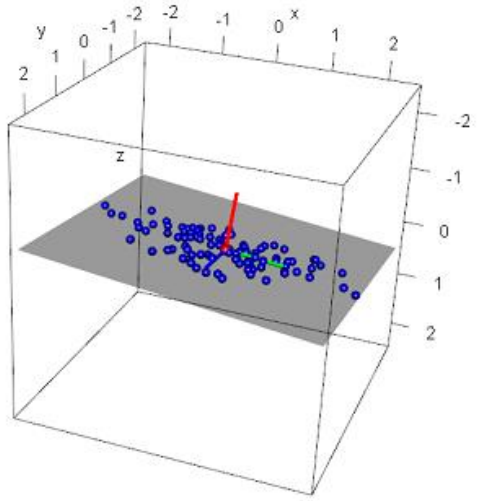


Figura 6: Representació dels tres eigenvectors d'un núvol de punts 3D

A partir d'aquests autovectors podem trobar dos angles que ens permetran rotar el núvol de punts en l'espai 3D. El procediment pot detallar-se més, però penso que amb l'anàlisi que hem fet del cas en 2D, ja queda clara la idea base.

Codi complert de l'exercici

```
clear;
close all;

% Point cloud creation
x = rand(1,100) + rand();   % 100 punts aleatoris amb un offset també aleatori
y = rand().*x  + rand(1,100)/10; % pendent de valor aleatori i offset també aleatori
ax(1) = subplot(2,1,1);
scatter(x,y);

% Cloud point centering
xp = x - mean(x);
yp = y - mean(y);

% Covariance and eigen values
c = cov(xp, yp);
[evectors, evalues] = eig(c);

% Determine which dimension has the major variance
[val,ind] = max(diag(evalues));

% Extract the angle of the % Extract the angle of the 'major axis'major axis'
theta = -pi/2 - atan2(evectors(1,ind),evectors(2,ind));

% Create clockwise rotation matrix
R = [cos(theta) sin(theta); -sin(theta) cos(theta)];

% Rotate the points
rp = R * [xp;yp];

ax(2) = subplot(2,1,2);
scatter(rp(1,:),rp(2,:));
linkaxes(ax,'y');
```