Université de Poitiers | **Analyse de données 2021–2022**

TP 2: ACP

F. Enikeeva, P. Carré | M1 STDV, M1 Info, M1 IOI

1 ACP : Exemple simple

Nous voulons étudier un tableau de notes.

1.1 Pré-étude des données

- Charger les données qui sont dans le tableau notes "notes.csv".

Pour cela on utilise la commande :

```
X = np.transpose(pd.read_csv("notes.csv", A compléter ...))
#On transpose car les individus doivent être en ligne pour utiliser les fonctions de Sklearn.
```

Il s'agit d'un tableau de notes obtenues par 9 élèves dans les matières suivantes : mathématiques, sciences, français, latin et dessin.

Nous allons tout d'abord étudier graphiquement et statistiquement ces données notées par `X`. On va d'abord créer les vecteur des noms des individus et des noms des variables:

```
nomi = list(X.index)    # noms de variables
nomv = list(X.columns)  # noms des individus
```

Pour chaque graphique demandé, extraire de l'information sur les données et confirmer ces constatations visuelles par des mesures statistiques (par exemple écart-type):

- Représenter graphiquement la matière "français". (on peut utiliser la commande matplotlib.pyplot.hist).
- Représenter graphiquement la matière "latin".
- Représenter dans un plan le nuage de points caractérisés par les deux variables "mathématique" et "sciences".

  Si vous souhaitez ajouter les noms des individus vous pouvez utiliser les commandes suivantes

  ```
  for i in range(0,n):
      plt.scatter(Data[i,0],Data[i,1])
      plt.text(Data[i,0],Data[i,1],nomi[i])
  ```

- Représenter dans un plan le nuage de points caractérisés par les deux variables "mathématique" et "dessin".

1.2 Calcul de l'ACP

Soit `X` les données de dimension p rangées dans un tableau de taille $n \times p$. On peut effectuer l'ACP en Python en utilisant, par exemple, la commande `PCA` de `scikit-learn`:

```
from sklearn.decomposition import PCA

acp = PCA(n_components=p)
cc = acp.fit_transform(X) ## cc contient les projections
```

Quelques options et attributs de la classe `PCA`:

- **Options:**
  - `n_components` est le nombre de composantes à garder, si le paramètre n'est pas défini, toutes les composantes seront calculées; avec l'option `n_components='mle'` le nombre de composantes à garder est calculé automatiquement
  - `whiten` est le flag de l'ACP sur la matrice de corrélation: `whiten='TRUE'`
- **Attribus de la structure stockée dans acp**
  - `components_` correspond aux axes principales (vecteurs propres de la matrice de covariance)
  - `explained_variance_` la quantité de variance (d'inertie) expliquée par chaque axe et donc la valeur propre associée à chaque axe;

Si vous souhaitez projeter un nouveau nuage de points dans les axes ainsi définis par l'ACP,

```
data2proj = acp.transform(data2)
```

1.3 Représentation

Les différentes étapes qui suivent doivent vous permettre de comprendre la structuration de l'ensemble d'apprentissage en classes. Ces classes vont correspondre à autant de "modes de fonctionnement". Pour chaque étape, analyser les différents graphiques et mesures afin de conclure sur l'organisation des données.

- Afficher l'évolution de l'inertie expliquée (variance) cumulée selon le nombre d'axes (scree–plot). Choisir le nombre d'axes à conserver.
- Représenter les individus dans les plans $E_1 \cup E_2$ et $E_1 \cup E_3$ (représentation 1). Ici E_i correspond à la projection sur l'axe i.
- Calculer les corrélations entre les variables initiales ξ_k, $k = 1, \ldots, p$ et les composantes principales Y_j, $j = 1, \ldots, p$:

  $$\mathrm{Corr}(\xi_k, Y_j) = \mathrm{acp.components}[j,k] \times \sqrt{\frac{\lambda_j}{\sigma_k^2}}.$$

  Ici $\mathrm{Var}[\xi_k] = \sigma_k^2$ et λ_j est la variance (valeur propre) correspondant à la composante Y_j. Attention, la matrice dite de rotation `acp.components_` est rangée par ligne (chaque ligne correspond au vecteur propre).

- étudier les valeurs de corrélation entre les matières initiales et les trois premiers axes factoriels. Commenter.
- Tracer la représentation simultanée des individus et des variables (biplot) dans les plans $E_1 \cup E_2$ et $E_1 \cup E_3$ (représentation 2)

  Pour cela, nous vous proposons la fonction `my_biplot` :

  ```
  from affichage_acp import my_biplot

  my_biplot(score=cc[:,0:2],coeff=np.transpose(acp.components_[0:2,:]),coeff_labels=nomv,...
  ...score_labels=nomi,nomx = "PC1",nomy = "PC2")
  ```

- Analyser les résultats.

2 Données réelles

Nous vous proposons de travailler sur des données réelles dans le cadre d'une application classique de l'ACP : l'analyse d'image multispectrale.

Une image multispectrale est une image dans laquelle les données correspondant à des mesures prises à différentes longueurs d'onde sur le spectre électromagnétique. Les longueurs d'onde peuvent correspondent à l'intervalle du spectre visible mais aussi de l'infra-rouge. L'imagerie multispectrale permet l'extraction d'information complémentaire à l'oeil humain qui se limite à la capture d'information avec ces trois récepteurs. A l'origine, l'imagerie multispectrale était utilisée pour l'observation de la terre que ce soit pour des applications militaires ou civil : en effet, la mesure multispectrale permet de différentier les différents composants d'un scène (végétaux, eaux, terre, bâtiment ...)

La difficulté de ce type d'imagerie est que en chaque position (pixel), l'information est caractérisée par un grand nombre de valeurs (dans notre exemple 200). Cela rend difficile alors la manipulation, l'interprétation et bien sûr la visualisation. L'ACP appliquée sur l'image devient alors un outil central pour traiter ces images.

Nous proposons de travailler sur un jeu de données d'exemple décrit ci-dessous

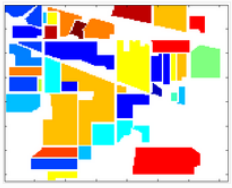
Indian Pines

This scene was gathered by AVIRIS sensor over the Indian Pines test site in North-western Indiana and consists of 145\times145 pixels and 224 spectral reflectance bands in the wavelength range 0.4–2.5 10^(-6) meters. This scene is a subset of a larger one. The Indian Pines scene contains two-thirds agriculture, and one-third forest or other natural perennial vegetation. There are two major dual lane highways, a rail line, as well as some low density housing, other built structures, and smaller roads. Since the scene is taken in June some of the crops present, corn, soybeans, are in early stages of growth with less than 5% coverage. The ground truth available is designated into sixteen classes and is not all mutually exclusive. We have also reduced the number of bands to 200 by removing bands covering the region of water absorption: [104-108], [150-163], 220. Indian Pines data are available through Pursue's univeristy MultiSpec site.

Groundtruth classes for the Indian Pines scene and their respective samples number

#	Class	Samples
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93

Sample band of Indian Pines dataset



Groundtruth of Indian Pines dataset

Figure 1: Description du jeu de données

Charger les données avec les instructions :

```
from scipy.io import loadmat

imgtmp = loadmat("Indian_pines_corrected.mat")
img = np.float32(imgtmp['indian_pines_corrected'])

maptmp = loadmat("Indian_pines_gt.mat")
map = (maptmp['indian_pines_gt'])
```

La variable `img` est un tableau de taille $145 \times 145 \times 200$ de réels correspondant aux mesures pour différentes longueurs d'ondes. La variable `map` est un tableau de taille $145 \times 145 \times 1$ de `uchar8` correspondant aux labels que chaque pixels selon le tableau présenté ci-dessus (c'est la vérité terrain).

Vous pouvez afficher une composante de l'image :

```
res = img[:,:,18]
plt.imshow((res-np.min(res))/(np.max(res)-np.min(res)))
# On normalise avant affichage car ce sont des réels


plt.imshow(map) #Pas besoin de normaliser
```

- Appliquer une ACP sur la tableau correspondant à l'image multispectrale (n'oubliez pas de mettre l'image sous la forme d'un tableau de 145×145 lignes (individus) pour 200 colonnes (mesures) grâce à l'instruction reshape).
- Evaluer combien d'axes sont nécessaires pour conserver une grande partie de l'information. Que pouvez-vous conclure sur l'information initiale.
- Visualiser sous forme d'une image en niveaux de gris la projection sur le principal axe factoriel. Visualiser sous la forme d'une image couleur la projection sur les principaux axes factoriels. (n'oubliez pas de mettre le tableau de projection 145×145 lignes (individus) pour 3 colonnes (mesures) sous la forme d'une image de taille $145 \times 145 \times 3$ grâce à l'instruction reshape)
- Comparer avec la réalité terrain et conclure.

  Il peut être important d'avoir le lien entre ces composantes et les longueurs d'ondes initiales afin de déterminer la signature spectrale de l'information ainsi extraite. Cela indique les longueurs d'ondes mise en valeurs suivant l'axe factoriel retenu. Nous pouvons calculer les corrélations entre les variables initiales (longueur d'ondes) et les composantes principales retenues, ou d'une manière plus simple le vectuer directeur de l'axe factoriel.
- Afficher avec l'instruction `plt.plot(pca.components_[k,:])` pour l'axe k la signature spectrale de l'axe. Commenter.

3 Programmation d'une ACP avec Python

Nous proposons maintenant de réaliser sous `Python` les différentes étapes de calcul de l'ACP. Pour cela, vous utiliser la bilbiothèque `Numpy` qui vous permet de réaliser toutes les opérations sur les matrices et vecteurs et notamment vous avez l'instruction :

```
(numpy.linalg.eigh(s))
```

qui permet de calculer les vecteurs et valeurs propres.

Réaliser une fonction `my_acp` qui a comme paramètre d'entrée le tableau `X` de données qui ess de taille $n \times p$ et qui retourne

- Le tableau `C` contenant les projections sur les axes factoriels qui est de taille $n \times p$
- le vecteur `lamb` contenant les parts d'inertie expliquées par chacun des axes factoriels, de taille p
- le tableau `u` contenant les axes factoriels, de taille $p \times p$.

Tester votre fonction avec l'exemple simple du tableau de notes.