SY09 Printemps 2019

TP 4
Analyse en composantes principales

1 Données `Notes`

Données

Le fichier `sy02-p2016.csv` contient des informations relatives aux étudiants ayant suivi l'UV SY02 au semestre de printemps 2016. On commencera par charger le jeu de données et déclarer les variables qualitatives comme telles. Les modalités doivent être les mêmes pour les correcteurs du médian et du final, et niveau comme résultat ECTS doivent être ordonnés comme il se doit.

1.1 Analyse

On pourra adopter la stratégie d'étude suivante.

1. Faire une analyse descriptive générale des données, en présentant les informations disponibles (nature, domaine de définition, volume, etc). Identifier les valeurs manquantes et les expliquer si possible. Mentionner les variables qui sont a priori liées.
2. Étudier les liens statistiques entre variables. On pourra en particulier étudier si la réussite à l'UV est influencée par la formation d'origine des étudiants, leur branche, ou leur niveau. On pourra de même étudier l'influence du correcteur sur la note obtenue.

1.2 ACP « à la main »

On s'intéresse à présent plus particulièrement aux correcteurs du jeu de données `notes` présenté ci-dessus. On commencera par constituer un nouveau jeu de données au moyen du code suivant :

```
> moy.median <- aggregate(note.median~correcteur.median, data=notes, FUN=mean)
> names(moy.median) <- c("correcteur","moy.median")
> std.median <- aggregate(note.median~correcteur.median, data=notes, FUN=sd)
> names(std.median) <- c("correcteur","std.median")
> median <- merge(moy.median, std.median)
> moy.final <- aggregate(note.final~correcteur.final, data=notes, FUN=mean)
> names(moy.final) <- c("correcteur","moy.final")
> std.final <- aggregate(note.final~correcteur.final, data=notes, FUN=sd)
> names(std.final) <- c("correcteur","std.final")
> final <- merge(moy.final, std.final)
> correcteurs <- merge(median, final, all=T)
```

Ce jeu de données contient les moyennes et les écarts-types par correcteur, pour le médian et le final. On remarquera qu'il manque des informations pour deux correcteurs ; on les écartera donc dans un premier temps du jeu de données :

```
> corr.acp <- correcteursX[-which(is.na(correcteursX), arr.ind=T)[,1],]
```

On associera dans cet exercice les mêmes pondérations à tous les individus, et on munira $\mathbb{R}^p$ de la métrique euclidienne.

1. Calculer les axes factoriels de l'ACP du nuage de points défini par les quatre variables quantitatives. Quels sont les pourcentages d'inertie expliquée par chacun de ces axes ?

```
> Xc <- scale(corr.acp[,2:5], scale=F)
> S <- t(Xc)%*%Xc/dim(Xc)[1]
> ACP <- NULL
> ACP <- eigen(S)
> ACP$pourc <- ACP$values/sum(ACP$values)*100
```

Les axes factoriels sont donc donnés par `ACP$vectors`, les inerties correspondantes par `ACP$values` et les pourcentages associés par `ACP$pourc`. On trouve les résultats suivants :

$$\texttt{ACP\$vectors} = \begin{pmatrix} -0.0368 & 0.7043 & -0.2332 & 0.6695 \\ 0.2942 & 0.6469 & -0.0943 & -0.6972 \\ -0.9550 & 0.1720 & -0.0206 & -0.2406 \\ -0.0006 & 0.2364 & 0.9676 & 0.0883 \end{pmatrix}$$

$$\begin{aligned} \texttt{ACP\$values} &= (0.9799, 0.3675, 0.0832, 0.0520)\,, \\ \texttt{ACP\$pourc} &= (66.10, 24.79, 5.61, 3.51)\,. \end{aligned}$$

2. Calculer les composantes principales ; en déduire la représentation des six individus dans le premier plan factoriel.

On travaille avec la métrique euclidienne ; on peut donc calculer les composantes principales et les représenter de la manière suivante :

```
> C <- Xc%*%ACP$vectors
> plot(C[,1:2])
> text(C[,1:2], labels=corr.acp[,1], pos=4)
```

3. Tracer la représentation des quatre variables dans le premier plan factoriel.

Il s'agit de calculer les corrélations entre anciennes et nouvelles variables :

```
> Dp <- diag(dim(Xc)[1])/dim(Xc)[1]
> Cor <- t(Xc)%*%Dp%*%C/
>
      ↪  matrix(sqrt(diag(S)),nrow=dim(Xc)[2])%*%matrix(sqrt(ACP$values),nrow=1)
> plot(Cor[,1:2])
> text(Cor[,1:2], labels=names(corr.acp)[2:5], pos=1)
```

4. Calculer l'expression $\sum_{\alpha=1}^{k} \mathbf{c}_\alpha \mathbf{u}_\alpha^T$ pour les valeurs $k = 1, 2$ et 3. À quoi correspond cette somme lorsque $k = 4$?

On peut calculer les quatre matrices de la reconstruction de la manière suivante :

```
> Xr <- NULL
> Xr$Xr1 <-
>     matrix(C[,1],nrow=dim(Xc)[1])%*%t(matrix(ACP$vectors[,1],
>         nrow=dim(Xc)[2]))
> Xr$Xr2 <- Xr$Xr1+
>     matrix(C[,2],nrow=dim(Xc)[1])%*%t(matrix(ACP$vectors[,2],
>         nrow=dim(Xc)[2]))
> Xr$Xr3 <- Xr$Xr2+
>     matrix(C[,3],nrow=dim(Xc)[1])%*%t(matrix(ACP$vectors[,3],
>         nrow=dim(Xc)[2]))
> Xr$Xr4 <- Xr$Xr3+
```

```
> matrix(C[,4],nrow=dim(Xc)[1])%*%t(matrix(ACP$vectors[,4],
>     nrow=dim(Xc)[2]))
```

On pourra vérifier que `Xr4` correspond bien à la matrice centrée.

5. On souhaite représenter les individus initialement écartés de l'ACP. Remplacer chacune de leurs valeurs manquantes par la moyenne de la variable correspondante sur les données conservées [1], puis représenter ces individus dans les deux premiers plans factoriels.

Il suffit tout simplement de projeter les individus extraits de la matrice initiale (après centrage par rapport aux moyennes calculées sur cette même matrice initiale) sur les axes factoriels :

```
> corr2 <- correcteurs[2,]
> corr3 <- correcteurs[3,]
> corr2[,2:3] <- corr2[,2:3] - apply(corr.acp[,2:3], 2, mean)
> corr2[,4:5] <- 0
> corr3[,4:5] <- corr3[,4:5] - apply(corr.acp[,4:5], 2, mean)
> corr3[,2:3] <- 0
> corr23 <- rbind(corr2, corr3)
> corr23.pc <- as.matrix(corr23[,2:5])%*%ACP$vectors
>
> plot(rbind(C,corr23.pc)[,1:2])
> text(rbind(C,corr23.pc)[,1:2], labels=c(corr.acp[,1],corr23[,1]),
  ↪  pos=4)
```

2 ACP avec R

L'objectif de cet exercice est de se familiariser avec les fonctions R permettant d'effectuer une ACP, en particulier les fonctions `princomp`, `summary`, `loadings`, `plot` et `biplot`. Remarquons qu'il existe une autre fonction `prcomp` qui effectue les calculs de manière différente ; on ne l'utilisera pas ici.

— En utilisant ces fonctions, effectuer l'ACP du jeu de données notes étudiées en cours. Montrer comment on peut retrouver tous les résultats alors obtenus (valeurs propres, axes principaux, composantes principales, représentations graphiques, ...).

— On s'intéresse à l'affichage des résultats de la fonction `princomp`. Qu'affichent les fonctions `plot` et `biplot` ? Détailler plus particulièrement le fonctionnement de la fonction `biplot` définie pour la classe `princomp` (accessible par `biplot.princomp`) et de ses différentes options.

3 Données `Crabs`

Le jeu de données considéré, disponible dans la bibliothèque de fonctions `MASS`, est constitué de 200 crabes décrits par huit variables (trois variables qualitatives, et cinq quantitatives). Charger le jeu de données et sélectionner les variables quantitatives en utilisant le code R suivant :

```
> library(MASS)
> data(crabs)
> crabsquant <- crabs[,4:8]
```

1. Cette opération d'*imputation par la moyenne*, qui consiste à remplacer la valeur manquante prise par une variable X^j pour un individu i donné par la moyenne empirique des valeurs correspondantes $x_{i'j}$ ($i' \neq i$) observées sur les autres individus, est une stratégie classique (quoique naïve) de gestion des données manquantes.

3.1 Analyse

1. Effectuer dans un premier temps une analyse descriptive des données. On s'interrogera notamment sur les différences de caractéristiques morphologiques, en particulier selon l'espèce ou le sexe : semble-t-il possible d'identifier l'une ou l'autre à partir d'une ou plusieurs caractéristiques morphologiques ?

Le tracé des boxplots confirme que l'analyse monovariée permet de mettre en évidence des différences significatives au niveau des distributions (médianes significativement différentes) mais pas d'identifier l'espèce ou le sexe à partir d'une unique variable. Cela justifie une approche multivariée.

L'analyse des données brutes par un graphique de dispersion par paires [a] met en évidence une très forte corrélation des différentes variables les unes aux autres. On est en présence d'un effet taille. Néanmoins, il est possible d'identifier des variables ou paires de variables pour lesquelles la distinction par espèce ou par sexe est visible :

```
> pairs(crabsquant, col=c("red","green")[crabs$sp])
> quartz()
> pairs(crabsquant, col=c("red","green")[crabs$sex])
```

Les couples CW/FL, et dans une moindre mesure BD/CW, semblent permettre de distinguer les deux espèces, qui restent néanmoins difficiles à séparer. Les combinaisons de RW avec chacune des autres variables semblent elles permettre de distinguer les crabes mâles des crabes femelles.

a. `pairs(crabsquant)`

2. Dans un second temps, on étudiera la corrélation entre les différentes variables. Quelle en est vraisemblablement la cause ? Quel traitement est-il possible d'appliquer aux données pour s'affranchir de ce phénomène ?

Les graphiques de dispersion par paires comme le calcul des corrélations [a] met en évidence les très grandes corrélations des variables les unes avec les autres. On est en présence d'un "effet taille", toutes ces variables représentant des mesures absolues (et non relatives à la taille globale du crabe).

L'idée pour s'affranchir de cet effet taille est donc de travailler sur des grandeurs relatives afin de mettre en évidence les différences morphologiques liées à l'espèce ou au sexe. Il s'agit donc de définir une variable que l'on assimilera à la taille globale du crabe, pour ensuite calculer les grandeurs relatives à cette variable. On pourra, au choix, définir la taille

— comme la somme des variables en présence :

```
> TA <- crabs$FL+crabs$RW+crabs$CL+crabs$CW+crabs$BD
```

— comme la variable la plus corrélée aux autres :

```
> TA <- crabsquant[,which.max(apply(cor(crabsquant),2,sum))]
```

On utilisera ensuite cette information de taille de manière à travailler sur des grandeurs relatives de la manière suivante :

```
> crabsquant2 <- crabsquant/
>     matrix(rep(TA,dim(crabsquant)[2]),nrow=dim(crabsquant)[1],byrow=F)
```

Si l'on a défini la taille comme étant la variable la plus corrélée aux autres, on prendra soin de la supprimer du tableau de données résultant :

```
> crabsquant2 <- crabsquant2[,-which.max(apply(cor(crabsquant),2,sum))]
```

a. `cor(crabsquant)`

3.2 ACP

Cette étude vise à utiliser l'ACP pour trouver une représentation des crabes qui permettent de distinguer visuellement différents groupes, liés à l'espèce et au sexe.

1. Tester tout d'abord l'ACP sur `crabsquant` sans traitement préalable. Que constatez-vous ? Comment pouvez-vous expliquer ce phénomène à la lumière de l'analyse exploratoire de ces données menée préalablement ?

   L'application de l'ACP sur les données `crabs` sans pré-traitement donne une représentation dans laquelle le premier axe porte une très grande partie de l'inertie. Ce phénomène est dû à l'effet taille évoqué plus haut.

   Le tableau de données contient en effet des mesures morphologiques relatives à 200 individus de toutes tailles : cette information est donc sous-jacente à chacune des grandeurs consignées dans le tableau. En conséquence, c'est principalement cette information de taille qui se trouve à l'origine de la variation des mesures morphologiques dans le jeu de données.

2. Trouver une solution pour améliorer la qualité de votre représentation en termes de visualisation des différents groupes.

   Il convient donc :
   - soit de travailler sur les données pré-traitées comme suggéré dans la première partie du TP, la taille étant estimée par la somme des variables en présence,
   - soit de supprimer simplement le premier axe factoriel et de représenter les données dans le plan formé par les second et troisième axes.

   Les deux stratégies permettent de distinguer les groupes de crabes correspondant aux différentes combinaisons espèce-sexe.

4 Données `Pima`

4.1 Analyse exploratoire

Le jeu de données considéré, disponible dans le fichier `Pima.csv`, est constitué de 532 individus de sexe féminin décrits par huit variables (dont une qualitative) :
- nombre de grossesses (`npreg`),
- taux plasmatique de glucose (`glu`),
- pression artérielle diastolique (`bp`),
- épaisseur du pli cutané au niveau du triceps (`skin`),
- indice de masse corporelle (`bmi`),
- fonction de pedigree du diabète (`ped` : mesure de l'influence génétique espérée des proches, affectés ou non par le diabète, sur le risque éventuel du sujet),
- âge (`age`),
- catégorie (`z`, diabétique si $z = 2$).

Charger le jeu de données en utilisant le code R suivant :

```
> Pima <- read.csv("Pima.csv", header=T)
> Pima$z <- factor(Pima$z)
```

Analyse

1. Effectuer dans un premier temps une analyse descriptive des données.
2. On tentera ensuite d'identifier les liens statistiques forts entre variables. En particulier, on s'intéressera au facteur « diabète », et à son influence potentielle sur les indicateurs numériques présents dans le jeu de données.

4.2 ACP

On cherche ici à représenter les données de manière à distinguer visuellement les groupes de patientes diabétiques et non diabétiques.

Tenter une ACP sur les données. Que constate-t-on ? Semble-t-il possible de trouver une représentation simple qui permette de distinguer les deux catégories de patientes ?

5 AFTD sur données de `Mutations`

On rappelle que l'AFTD calcule une représentation multidimensionnelle, dans un espace euclidien de dimension $p \leq n$, de données se présentant sous la forme d'un tableau $n \times n$ de dissimilarités δ_{ij} entre n individus ($i, j \in \{1, \ldots, n\}$), dont le tableau de dissimilarités ne donne qu'une description implicite. Cette représentation est exacte lorsque les dissimilarités sont des distances euclidiennes.

Une fois que des variables ont été retenues, la qualité de la représentation peut être évaluée numériquement par un critère similaire au pourcentage d'inertie de l'ACP, ou graphiquement au moyen d'un *diagramme de Shepard* représentant la distance $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ entre les représentations de $\boldsymbol{x}_i$ et $\boldsymbol{x}_j$ déterminées par l'AFTD en fonction de la dissimilarité initiale δ_{ij}, pour tout $(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Sous R, on pourra utiliser les fonctions `cmdscale` ainsi que `Shepard` (bibliothèque `MASS`).

Charger les données de `Mutations` et déclarer les données comme tableau de dissimilarités :

```
> mut <- read.csv("mutations2.csv", header=T, row.names=1)
> mut <- as.dist(mut, diag=T, upper=T)
```

Calculer une représentation euclidienne des données en $d = 2$ variables par AFTD ; l'afficher ; calculer la qualité de la représentation, et afficher le diagramme de Shepard. Que peut-on dire ? Recommencer avec $d \in \{3, 4, 5\}$. Interpréter les résultats.

On pourra utiliser le code suivant. On notera que la qualité de représentation est calculée par rapport à la somme des valeurs absolues des valeurs propres de $\frac{1}{n}W$, de manière à tenir compte des valeurs propres négatives (le tableau de dissimilarités n'étant ici pas un tableau de distances euclidiennes).

```
> repr <- cmdscale(mut, k=2, eig=T)
> plot(repr$points)
> text(repr$points, labels=row.names(repr$points))
>
> qual <- repr$eig[1:2]/sum(abs(repr$eig))
>
> plot(Shepard(mut, repr$points))
> abline(0,1)
```

En ré-exécutant ces instructions pour des valeurs croissantes de k, on constate que les points sont de plus en plus proches de la première bissectrice.

Mutation distances among 20 species (Fitch and Margoliash)

The source of this data is a paper by Fitch and Margoliash in Science(1967). For a more recent reference see Scientific American (1972 ?). Every species has a protein molecule, Cytochrome c, which varies from species to species but has a similar function for all. It consists of a long chain of amino acids. There are only a few acids, but different molecules are obtained by varying the acids in each position in the chain. The number of positions with different acids measures distance between two species. These distances are given in the data below. For example, the amino acids in Cytochrome c for two species look like this : Moth XXYVPLY .....SEXI Screwworm fly XXYVPLY .....LSEI where the whole chain is 110 in length, and the letters represent particular amino acids. Each difference contributes to mutation distance according to the minimum number of nucleotides that would need to be changed to convert one into the other. Fitch & Margoliash used these data to construct a phylogenetic tree.

Science, vol. 155, pp. 279-284.

```
> Man                0
> Monkey            01  0
> Dog               13 12  0
> Horse             17 16 10  0
> Donkey            16 15 08 01  0
> Pig               13 12 04 05 04  0
> Rabbit            12 11 06 11 10 06  0
> Kangaroo          12 13 07 11 12 07 07  0
> Pekin Duck        17 16 12 16 15 13 10 14  0
> Pigeon            16 15 12 16 15 13 08 14 03  0
> Chicken           18 17 14 16 15 13 11 15 03 04  0
> King Penguin      18 17 14 17 16 14 11 13 03 04 02  0
> Snapping Turtle   19 18 13 16 15 13 11 14 07 08 08 08  0
> Rattlesnake       20 21 30 32 31 30 25 30 24 24 28 28 30  0
> Tuna              31 32 29 27 26 25 26 27 27 27 26 27 27 38  0
> Screwworm Fly     33 32 24 24 25 26 23 26 26 26 26 28 30 40 34  0
> Moth              36 35 28 33 32 31 29 31 30 30 31 30 33 41 41 16  0
> Bakers Mould      63 62 64 64 64 64 62 66 59 59 61 62 65 61 72 58 59  0
> Bread Yeast       56 57 61 60 59 59 59 58 62 62 62 61 64 61 66 63 60 57  0
> Skin Fungus       66 65 66 68 67 67 67 68 66 66 66 65 67 69 69 65 61 61 41 0
```