TP2 – *Eigenfaces*

Ce TP s'inspire d'un article intitulé *Eigenfaces for recognition*, écrit par Turk et Pentland et publié dans le *Journal of Cognitive Neuroscience* en 1991.

Description des données

Vous disposez de n images de visages d'un ensemble d'individus. Chaque individu est photographié sous le même nombre de postures faciales (gauche, face, trois quart face, etc.). Chacune de ces n images en niveaux de gris est stockée dans une matrice bidimensionnelle de taille 480×640. Ces n images constituent les *images d'apprentissage*. En les vectorisant, vous pouvez donc représenter ces images par des vecteurs colonnes de $\mathbb{R}^p$, où $p = 480 \times 640 = 307200$ est le nombre de pixels commun à toutes les images. Alors que dans le TP1, chaque pixel d'une image couleur constitue un point de $\mathbb{R}^3$, ici c'est chaque image qui constitue un point d'un espace affine $\mathbb{R}^p$ de dimension très élevée.

La matrice des données X, de taille $n \times p$, contient sur chaque ligne la transposée d'une image vectorisée. Lancez le script `donnees.m` afin de créer cette matrice et de la stocker dans un fichier au format Matlab, de nom `donnees.mat`. **Attention** : ne recopiez pas les images sur votre compte, afin de préserver votre quota !

Différences importantes avec le TP1

- Dans le TP1, la matrice des données centrées X_c est de taille $p \times 3$, où p désigne le nombre de pixels de l'image RVB. Or, le rang d'une matrice est inférieur à sa plus petite dimension. Comme $p \gg 3$, cela signifie que $\mathrm{rg}(\mathrm{X}_c) \leqslant 3$. Pour une image naturelle, le coefficient de corrélation linéaire entre les 3 canaux RVB n'est jamais exactement égal à ± 1. Cela signifie que les 3 colonnes de X_c sont linéairement indépendantes, donc que $\mathrm{rg}(\mathrm{X}_c) = 3$ (on dit que X_c est *de rang maximal*). Une conséquence de ce résultat est que la matrice de variance/covariance $\Sigma = \mathrm{X}_c^\top \mathrm{X}_c / p$ est également de rang 3. Comme elle est de taille 3×3, cette matrice est inversible.

- Dans le TP2, la matrice X_c des données centrées, obtenue en retranchant à chaque ligne de X l'*individu moyen* $\overline{\mathrm{X}}$ (égal à la moyenne des lignes de X), est de taille $n \times p$, où n désigne le nombre d'images et p le nombre de pixels commun à toutes ces images. Comme $p \gg n$, on en déduit que $\mathrm{rg}(\mathrm{X}_c) \leqslant n$. Pour que cette matrice soit de rang maximal, il faudrait que ses n lignes soient linéairement indépendantes. Or, leur somme est égale au vecteur nul, puisque $\overline{\mathrm{X}}$ est égal à la moyenne des n lignes de X. Pour des images naturelles, on en déduit que $\mathrm{rg}(\mathrm{X}_c) = n - 1$. La matrice de variance/covariance $\Sigma = \mathrm{X}_c^\top \mathrm{X}_c / n$ est donc elle aussi de rang $n - 1$. Comme elle est de taille $p \times p$, et que $p \gg n$, cette matrice n'est pas inversible. En l'occurrence, le noyau de Σ est de dimension $p - n + 1$.

- Une autre différence avec le TP1 vient de ce que, dans le TP2, la fonction `eig` ne peut pas être directement appliquée à Σ. En effet, sa taille $p \times p$ est gigantesque ($p = 307200$). Or, pour une matrice M quelconque, $\mathrm{M}^\top \mathrm{M}$ et $\mathrm{M}\,\mathrm{M}^\top$ ont les mêmes valeurs propres *non nulles*. On peut donc appliquer la fonction `eig` à $\Sigma_2 = \mathrm{X}_c\,\mathrm{X}_c^\top / n$, de taille $n \times n$ beaucoup plus petite, pour calculer les valeurs propres non nulles de Σ.

- Si Y est un vecteur propre de Σ_2 associé à une des $n - 1$ valeurs propres λ non nulles, alors par définition $(\mathrm{X}_c\,\mathrm{X}_c^\top / n)\,\mathrm{Y} = \lambda\,\mathrm{Y}$, d'où $(\mathrm{X}_c^\top\,\mathrm{X}_c / n)\,\mathrm{X}_c^\top\,\mathrm{Y} = \lambda\,\mathrm{X}_c^\top\,\mathrm{Y}$. D'autre part, $\mathrm{X}_c^\top\,\mathrm{Y}$ est un vecteur non nul : sinon, cela impliquerait que le vecteur $\lambda\,\mathrm{Y}$ est nul, ce qui est impossible puisque $\lambda \neq 0$ par hypothèse et que, étant un vecteur propre, Y ne peut pas être un vecteur nul. Par conséquent, $\mathrm{X}_c^\top\,\mathrm{Y}$ est un vecteur propre de $\Sigma = \mathrm{X}_c^\top \mathrm{X}_c / n$ associé à la valeur propre λ. Il est facile de montrer que les $n - 1$ vecteurs $\mathrm{X}_c^\top\,\mathrm{Y}$ sont orthogonaux deux à deux. En les normalisant, on obtient donc une base orthonormée $\mathcal{B}$ de $\mathrm{Im}(\Sigma)$.

Exercice 1 : analyse en composantes principales

Complétez le script `exercice_1.m`, qui vise à calculer les axes principaux des images d'apprentissage à partir des vecteurs propres associés aux $n-1$ valeurs propres non nulles de la matrice de variance/covariance Σ des données. Ces axes principaux sont appelés *eigenfaces* par Turk et Pentland, par contraction des mots anglais *eigenvectors* et *faces*.

Exercice 2 : projection des images sur les *eigenfaces*

Une fois connues les $n-1$ *eigenfaces*, on peut calculer les composantes principales. Complétez le script `exercice_2.m`, de manière à afficher les images d'apprentissage reconstruites à l'aide des q premières *eigenfaces* et des q premières composantes principales, pour $q \in [0, n-1]$. **Attention** : n'oubliez pas d'ajouter l'individu moyen. Ce script doit également afficher l'évolution, en fonction de q, de la racine carrée de l'erreur quadratique moyenne (*Root Mean Square Error*, ou RMSE) entre les images originales et les images ainsi reconstruites.

Exercice 3 : application à la reconnaissance de visages

Le script `clusters.m` calcule les composantes principales des n images d'apprentissage, puis affiche sous la forme d'un nuage de n points de $\mathbb{R}^2$ leurs deux premières composantes principales. Chaque couleur correspond à un même individu de la *base d'apprentissage*. Ce nuage fait apparaître des groupes de points (ou *clusters*) de couleur uniforme, ce qui montre que chaque *cluster* correspond aux différentes postures d'un même individu. Il semble donc possible d'utiliser les *eigenfaces* pour la reconnaissance de visages (comme l'indique le titre de l'article ayant inspiré ce TP : *Eigenfaces for recognition*), en calculant les deux premières composantes principales d'une image, dite *image de test*, n'appartenant pas forcément à la base d'apprentissage, et en cherchant de quelle image d'apprentissage cette image est la plus proche, donc à quel individu elle correspond.

Mais il suffit de remplacer la posture numéro 6 par la posture numéro 5 dans le script `donnees.m`, puis de relancer le script `exercice_1.m`, pour comprendre pourquoi cette idée ne peut pas fonctionner ainsi. En calculant la proportion de contraste (cf. TP1) correspondant aux deux premières composantes principales, vous constaterez que cette proportion n'est pas suffisamment proche de 1 pour qu'il soit possible de correctement discriminer la totalité des images de la base de données à l'aide de deux composantes principales seulement. Il faut utiliser plus de composantes principales.

Le script `exercice_3.m` tire aléatoirement (à l'aide de la fonction `randi` de Matlab) une image de test, parmi les quinze individus et les six postures faciales disponibles dans la base de données. Complétez ce script, qui commence par déterminer le nombre N de composantes principales à utiliser pour que la proportion de contraste soit supérieure à 95%, et qui sélectionne les N premières composantes principales des n images d'apprentissage et de l'image de test (n'oubliez pas de retrancher l'individu moyen). Ce script doit ensuite calculer la distance euclidienne de l'image de test (considérée comme un point de $\mathbb{R}^N$) à chacune des n images d'apprentissage. En notant $d_{\min}$ la plus petite de ces distances, deux cas se présentent :

- Si $d_{\min} < s$, où s désigne un seuil, l'individu a été reconnu : son numéro doit être affiché.
- Sinon, l'individu n'a pas été reconnu : cela doit également être indiqué.

Ajustez la valeur du seuil s de manière à minimiser le nombre de *faux négatifs*, tout en veillant à ce qu'il n'y ait aucun *faux positif*. La valeur de s fournie par défaut a été grossièrement ajustée pour le cas où la base d'apprentissage contient les quatre premières postures des sept premiers individus.

Question facultative – Effectuez une copie du script `exercice_3.m`, de nom `exercice_3_bis.m`, dans laquelle tout individu non reconnu est ajouté à la base d'apprentissage, sous les mêmes postures que les individus déjà présents dans cette base, puis recalcule les *eigenfaces*.