TD ACP

1 Codons l'ACP sous Matlab

1. Développer une fonction `[valprop, U, moy] = mypca(X)` permettant de calculer les k premières composantes principales d'un ensemble de données $X \in \mathbb{R}^{N \times D}$.
   `X` : données
   `valprop` : valeurs propres de la matrice de covariance triées dans l'ordre décroissant
   `U` : la matrice des vecteurs propres (respectant l'ordre des valeurs propres)
   `moy` : la valeur moyenne des D variables x^j

2. Coder une fonction `Ct = projpca(X, moy, P)` permettant de projeter une matrice de données $X \in \mathbb{R}^{N \times D}$ sur les d premiers vecteurs propres.
   `moy` : la valeur moyenne fournie par la fonction `mypca`
   `P` : la matrice contenant les d premiers vecteurs propres fournis par la fonction `mypca`
   `C` : matrice des composantes principales (projection des x_i sur P)
3. Coder une fonction `Xhat = reconstructpca(C, P, moy)` permettant de calculer $\hat{X}$, la reconstruction de X connaissant C.
   `P` : la matrice contenant les d premiers vecteurs propres fournis par la fonction `mypca`
   `C` : matrice des composantes principales fournie par la fonction `projpca`

2 Tests de l'ACP

1. Récupérer les données USPS disponibles sur Moodle. Choisir un caractère entre 1 et 9 et extraire les données correspondant à ce caractère.
2. Tracer la matrice de corrélation (instruction `corrcoef`) et analyser la
3. Tester votre méthode ACP sur ces données (déterminer la matrice P, projeter vos données sur P).
4. Visualiser la projection des données en 2D (deux premières composantes de C)
5. Représenter la reconstruction d'un des caractères sur les 10 premières composantes principales. Quel est selon vous la valeur de d appropriée ?

3 Visages propres

On dispose d'un ensemble d'images étiquetées de taille 50×50 représentant les visages de 10 personnes sous différentes conditions d'illuminations. Chaque image a été transformée en un vecteur de dimension 2500. L'objectif est de reconnaître les visages en utilisant l'algorithme des k-ppv appliqué aux images projetées par ACP afin d'éviter de calculer des distances euclidiennes sur des vecteurs de dimension 2500. Pour l'apprentissage, on utilisera la base `subset1faces.mat`.

1. Appliquer la fonction `mypca` aux données `subset1faces.mat`. Afficher le "visage moyen" et les $d = 9$ premiers "visages propres" (il s'agit des vecteurs propres).
   Instruction matlab : `imagesc(reshape(moy, 50, 50)); colormap(gray)`

2. Pour réaliser la classification par k-ppv, les bases `subset2faces.mat` et `subset3faces.mat` serviront de validation ou de test
   (a) Pour une valeur d fixée, former la matrice P en utilisant les résultats de la question 1
   (b) Projeter les données d'apprentissage, de validation et de test en utilisant P et `moy` de la question 1. On notera $Capp$, $Cval$ et $Ctest$ les matrices obtenues.
   (c) Centrer et réduire ces matrices (utiliser la fonction `normalizemeanstd`)
   (d) Mettre en oeuvre alors votre classification par k-ppv en s'inspirant du TD 1. Quelle valeur de k donne les meilleures performances ?
3. Comment évolue vos performances si on varie d ? Comment automatiser la détermination des valeurs optimales de k et d.

4 Question subsidiaire

1. Représenter en utilisant différentes couleurs la projection en 2D des visages du jeu d'apprentissage