1 - Rappels

1.1 Introduction au calcul avec matlab

Remarque préléminaire : Pour l'utilisation de matlab utiliser le document disponible sur moodle et la commande help de matlab pour obtenir toutes les informations utiles et nécessaires sur l'utilisation de matlab et des fonctions disponibles.

1.1.1 Opérations sur les matrices

1. Soit la matrice $A = \begin{pmatrix} 1 & 2 & -1 & 3 \\ -1 & 0 & 2 & 1 \end{pmatrix}$
   Calculer puis vérifier avec matlab AA^T , A^TA
2. Ecrire une fonction matlab qui permet de permuter deux lignes d'une matrice et de créer la matrice de permutation. Vérifier les résultats en utilisant la matrice de permutation.
3. Ecrire une fonction matlab qui permet de générer une matrice symétrique aléatoire de taille n. Un deuxième paramètre i permet d'ajuster l'intervalle des valeurs entre $\pm i$
4. Ecrire une fonction matlab qui permet de générer une matrice inversible de taille n avec le même paramètre i.
5. Ecrire une fonction matlab qui calcule le déterminant d'une matrice en utilisant la définition. Mesurer le temps de calcul de cette fonction pour des tailles de matrice de 2 à 10 inclus, et comparer avec le temps de calcul de la fonction 'det' de matlab.
6. Ecrire une fonction matlab qui calcule la comatrice d'une matrice A.
7. Soit la matrice $A = \begin{pmatrix} 0 & -1 & 2 \\ 0 & 0 & 3 \\ 0 & 0 & 0 \end{pmatrix}$
   (a) Montrer que A est nilpotente.
   (b) Ecrire un programme matlab qui permet de vérifier qu'une matrice est nilpotente.
8. Soient les matrices :
   $A = \begin{pmatrix} 1 & 2 & 1 & 3 \\ 1 & 5 & 0 & 2 \\ 2 & 4 & 2 & 6 \\ 3 & 0 & 1 & 1 \end{pmatrix}$ et $B = \begin{pmatrix} 1 & 2 & 1 & 3 \\ 5 & 4 & 3 & 7 \\ 2 & 4 & 2 & 6 \\ 3 & 0 & 1 & 1 \end{pmatrix}$
   Pour chaque matrice :
   (a) Calculer le déterminant, la comatrice et le rang.
   (b) La matrice est-elle inversible ? pouvait-on le prévoir ?
   (c) Ecrire une fonction matlab qui permet de calculer une comatrice.
   (d) Vérifier ces calculs avec matlab

1.1.2 Valeurs propres et vecteurs propres

1. Soit la matrice :
   $A = \begin{pmatrix} 1 & 4 & -2 \\ -1 & 6 & -2 \\ -1 & 5 & -1 \end{pmatrix}$
   Calculer les valeurs propres et vecteurs propres, puis vérifier le résultat.
2. Ecrire une fonction matlab qui retourne les matrices des valeurs propres et des vecteurs propres.

1.1.3 Pseudo-inverse

1. Ecrire une fonction matlab qui réalise la décomposition en valeurs singulières
2. Ecrire une fonction matlab qui calcule la pseudo-inverse.

1.1.4 Factorisation LU

1. Soient les matrices : $A = \begin{pmatrix} 4 & -9 & 2 \\ 2 & -4 & 4 \\ -1 & 2 & 2 \end{pmatrix} \Rightarrow L = \begin{pmatrix} 1 & 0 & 0 \\ 0.5 & 1 & 0 \\ -0.25 & -0.5 & 1 \end{pmatrix}, U = \begin{pmatrix} 4 & -9 & 2 \\ 0 & 0.5 & 3 \\ 0 & 0 & 4 \end{pmatrix}$
2. Ecrire le code d'une fonction qui calcule la décomposition LU en utilisant le principe décrit en cours. tester cette fonction avec les matrices ci-dessus.
3. Estimer les performances de ces algorithmes pour des tailles de matrice allant jusqu'à 1000.
4. Ecrire le code d'une fonction qui calcule le déterminant d'une matrice en utilisant la décomposition LU.

1.1.5 Puissance

1. Soit la matrice : $A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$
   (a) Montrer :
   $A^n = \frac{1}{3}\begin{pmatrix} 2+4^n & -1+4^n & -1+4^n \\ -1+4^n & 2+4^n & -1+4^n \\ -1+4^n & -1+4^n & 2+4^n \end{pmatrix}$
   (b) Ecrire le code qui calcule A^n avec les deux méthodes
   (c) Vérifier les résultats pour plusieurs valeurs de n.
2. Soit la matrice :
   $A = \begin{pmatrix} 1 & 7 & 8 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} = N + D = \begin{pmatrix} 0 & 7 & 8 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
   (a) Vérifier que N et D commutent.
   (b) Calculer A^n en utilisant le binôme de Newton :
   $(N+D)^n = \sum_{k=0}^{n} \binom{n}{k} N^k D^{n-k}$
   (c) Ecrire le code d'une fonction qui permet de calculer A^n en utilisant le binôme de Newton.

1.1.6 Exponentielle

1. Soit la fonction $f_x(n) = exp(x) - \sum_{k=0}^{n} \frac{\mathrm{x}^k}{k!}$,
   (a) tracer cette fonction pour $n \in [1, 20]$ avec plusieurs valeurs de x (exemple, 1 à 10).
   (b) Conclusion.
2. Cas d'une matrice diagonalisable :
   (a) Ecrire la fonction qui calcule l'exponentielle d'une matrice diagonalisable en utilisant l'exponentielle standard.
   (b) On remplace exp(x) par une approximation de Padé de la forme : $\frac{n_{p,q}(x)}{d_{p,q}(x)} = \frac{\sum_{i=0}^{p} a_i x^i}{\sum_{j=0}^{q} a_j x^j}$ avec
   $n_{p,q}(x) = \sum_{i=0}^{p} \frac{\frac{p!}{(p-i)!}}{\frac{(p+q)!}{(p+q-i)!}} \frac{x^i}{i!}$ et $d_{p,q}(x) = \sum_{i=0}^{q} (-1)^j \frac{\frac{q!}{(q-j)!}}{\frac{(p+q)!}{(p+q-j)!}} \frac{x^j}{j!}$
   Exemple avec $p = q = 3$:
   $\frac{n_{3,3}(x)}{d_{3,3}(x)} = \frac{1 + \frac{1}{2}x + \frac{1}{10}x^2 + \frac{1}{120}x^3}{1 - \frac{1}{2}x + \frac{1}{10}x^2 - \frac{1}{120}x^3}$
      i. Tracer l'approximation de Padé pour les valeurs $p = q = 1$ à $p = q = 3$ et x compris entre 0 et 5.
      ii. Comparer les résultats avec ceux obtenus avec l'exponentielle. Conclusion.
      iii. Dans la fonction de calcul de l'exponentielle d'une matrice diagonalisable remplacer le calcul de l'exponentielle standard par l'approximation de Padé.

2 - Introduction aux réseaux de neurones

2.1 Droites de séparation de données

1. Ecrire une fonction matlab qui permet d'afficher les données ainsi que la droite de séparation de données.
2. Utiliser cette fonction avec les fonctions OU et ET du cours.
3. Utiliser cette fonction avec les différentes couches de la fonction OU exclusif du cours.

2.2 Apprentissage des réseaux simple couches

2.2.1 Règle de Hebb

1. Réaliser l'apprentissage des fonctions ET et OU du cours. Tracer la droite de séparation.
2. On possède une ensemble d'échantillons de données suivantes :
   $x_1 = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix}$ $x_2 = \begin{pmatrix} -1 \\ -1 \\ 1 \\ -1 \end{pmatrix}$ $x_3 = \begin{pmatrix} -1 \\ -1 \\ -1 \\ -1 \end{pmatrix}$ $x_4 = \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix}$ $x_5 = \begin{pmatrix} -1 \\ 1 \\ -1 \\ -1 \end{pmatrix}$
   associées aux classes de sorties suivantes :
   $y_1 = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}$ $y_2 = \begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix}$ $y_3 = \begin{pmatrix} -1 \\ -1 \\ 1 \end{pmatrix}$ $y_4 = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$ $y_5 = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}$
   (a) Réaliser l'apprentissage et donner la matrice W
   (b) Vérifier le résultat obtenu
3. On cherche à détecter des classes de caractères qui sont représentés graphiquement par une matrice, on traitera les caractères 'o' et 'x'.
   $O = \begin{bmatrix} -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 & 1 \\ 1 & -1 & -1 & -1 & 1 \\ -1 & 1 & 1 & 1 & -1 \end{bmatrix}$ $X = \begin{bmatrix} 1 & -1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 & -1 \\ -1 & -1 & 1 & -1 & -1 \\ -1 & 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & -1 & 1 \end{bmatrix}$ $Z = \begin{bmatrix} -1 & 1 \end{bmatrix}$: -1 pour le caractère 'o' et 1 pour le caractère 'x'
   (a) Réaliser l'apprentissage et donner la matrice W
   (b) Vérifier le résultat obtenu
   (c) Appliquer la matrice W aux caractères dégradés comme par exemple :
   $Od = \begin{bmatrix} -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 & -1 \end{bmatrix}$ $Xd = \begin{bmatrix} 1 & -1 & -1 & -1 & 1 \\ -1 & 1 & -1 & 1 & -1 \\ -1 & -1 & 1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 \\ 1 & -1 & -1 & -1 & -1 \end{bmatrix}$
   (d) Appliquer la matrice W en dégradant encore plus les caractères. Conclusion ?
4. identification d'images
   (a) Charger les images lena et barbara, redimenssionner les images afin d'éviter des calculs importants (ex 64x64).
   (b) Réaliser l'apprentissage afin d'avoir une classe pour chaque image et vérifier les résultats.
   (c) Dégrader les images et observer les résultats

2.2.2 Perceptron

1. Ecrire une fonction matlab qui réalise l'apprentissage en utilisant la règle du perceptron.
2. Réaliser l'apprentissage du cours et vérifier les résultats
3. Traitement de données
   (a) Réaliser l'apprentissage de données définies dans $\mathbb{R}^2$ à partir de la base d'apprentissage fournie dans le fichier baseapprentissage.csv (la première colonne correspond à X, la deuxième à Y et la troisième au résultat attendu).
   (b) Vérifier les résultats avec la base de vérification baseverification.csv.
4. Réaliser l'apprentissage des données des questions 2 à 4 du 2.2.1 et vérifier les résultats. Pour l'identification des caractères, vérifier les résultats sur des données dégradés.

2.2.3 Perceptron multicouche

1. Ecrire la fonction matlab feedforward donnée en cours.
2. Ecrire une fonction matlab qui réalise l'apprentissage pour le perceptron multicouche.
3. Réaliser l'apprentissage du ou exclusif et vérifier les résultats.
4. Réaliser l'apprentissage des données d'affichage sept segments, puis vérifier les résultats.

2.2.4 RBF

1. Ecrire une fonction matlab feedforward. Vérifier avec les deux versions du OU exclusif.
2. Ecrire une fonction matlab qui réalise l'apprentissage pour le RBF en utilisant les algorithmes du cours.
3. Réaliser l'apprentissage sur les données étiquetées du fichier datasrbfsource.txt puis vérifier les résultats.
4. Réaliser l'apprentissage sur les données étiquetées avec le MLP.
5. Comparer les résultats obtenus entre RBF et MLP.

3 - Traitement du signal et des images

3.1 Filtrage de signaux à une dimension

1. A partir du masque de filtre $h = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \end{pmatrix}$
   Proposer une matrice qui permette de donner les coefficients des filtres suivants :
   $h = \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \end{pmatrix}$ $h = \begin{pmatrix} \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}$ $h = \begin{pmatrix} \frac{1}{2} & 0 & -\frac{1}{2} \end{pmatrix}$
2. Générer le signal carré, et vérifier le résultat des différents filtres.
3. Afficher les spectres des signaux d'entrée et de sortie.
4. Ecrire le code matlab qui permet d'appliquer les différents masques de convolution à l'image fournie.

3.2 Filtrage de signaux à deux dimensions

1. Charger l'image fournie sur moodle.
2. Charger l'image avec matlab.
3. Ecrire le code matlab qui permet d'appliquer les filtres passe-bas suivants :
   $h_1 = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$ $h_2 = \frac{1}{10}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$ $h_3 = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$
   Conclusion ?
4. Ecrire le code matlab qui permet d'appliquer les filtres passe-bas suivants :
   $h_1 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$ $h_2 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$
   $h_3 = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$ $h_4 = \begin{pmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{pmatrix}$ $h_5 = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$
   Conclusion ?
5. Ecrire le code matlab qui permet de calculer les filtres de gabor et de les appliquer à l'image et calculer l'image résultante.

3.3 OpenCV

3.3.1 Traitement d'image

1. Ecrire un programme qui permet d'appliquer les différents filtre à une image et d'afficher le résultat.
2. Ecrire un programme qui permet d'appliquer le filtre de Gabor à une image et d'afficher le résultat.

3.3.2 Traitement vidéo

1. Ecrire un programme qui permet d'appliquer les différents filtre sur une vidéo.
2. Ecrire un programme qui permet de réaliser une binarisation sur une vidéo.
3. Ecrire un programme qui permet de détecter un mouvement sur une vidéo.