

Travaux Dirigés

Initiation à Matlab

I Exercices simples

Après avoir été testés sur la ligne de commande, toutes les instructions effectuées lors de ce TD seront placées dans des fichiers (scripts Matlab).

I.1 Manipulation des scalaires, vecteurs et matrices

Scalaires

- + Calculer $\sqrt{10}$ et vérifier que $\sqrt{10}^2 = 10$.
- + Afficher i et vérifier $i^2 = -1$.
- + Calculer $x = (1 + 3i)(2 + 2i)(3 + i)$ ainsi que son module et son argument en degré.
- + Afficher $\cos(\pi/2), \sin(\pi/2)$ et $\exp(i\pi/2)$. Conclure...

Vecteurs

- + Construire le vecteur ligne v contenant les nombres entiers pairs de 1 à 20. L'utiliser pour calculer la racine carrée de ces nombres.
- + Construire le vecteur colonne v_2 contenant les mêmes éléments que v . Calculer et donner les dimensions de v_2v et vv_2 .
- + Ajouter 22 et 24 au vecteur v . A partir de ce nouveau vecteur construire en utilisant la fonction `reshape` la matrice : $A = \begin{bmatrix} 2 & 4 & 6 & 8 \\ 10 & 12 & 14 & 16 \\ 18 & 20 & 22 & 24 \end{bmatrix}$
- + Pour N quelconque construire le vecteur : $v_3 = [0 \ 1 \ 2 \ \dots \ N-1 \ N \ N-1 \ \dots \ 2 \ 1]$. À partir de v_3 ne conserver qu'un élément sur deux ; enfin, toujours à partir de v_3 , construire le vecteur : $[N \ N-1 \ \dots \ 1 \ 0 \ 1 \ \dots \ N-1]$. Refaire les mêmes manipulations pour un autre N (impair si le premier était pair et vice versa).

Matrices

- + Ajouter la ligne $[26 \ 28 \ 30 \ 32]$ à la matrice A .
- + Afficher la troisième ligne de A , sa deuxième colonne, puis les éléments des 1ère et 3ème lignes et 2ème et 4ème colonnes.
- + Construire B en supprimant la dernière colonne et la deuxième ligne de A .
- + Calculer la matrice des éléments de A au carré, ajouter un à chacun de ces éléments.
- + Construire la matrice $C = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$, puis calculer $B * C$ et $B.*C$

- + Construire une matrice similaire à C mais de dimension 999×999 .
- Créer la matrice M de taille 10×10 utilisant la fonction `magic`. Calculer la somme des lignes de M et la somme de ses colonnes. Pourquoi cette matrice est-elle magique ?

- Soit la matrice $D = \begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix}$. Calculer l'inverse de D . Calculer la solution

du système $Dx = y$ pour $y = [32 \ 23 \ 33 \ 31]^T$ puis pour un second membre légèrement perturbé : $y_b = y + 0,1[1 \ -1 \ 1 \ -1]^T$. Faire de même en perturbant cette fois-ci la

$$\text{matrice : } D_b = \begin{bmatrix} 10 & 7 & 8,1 & 7,2 \\ 7,08 & 5,04 & 6 & 5 \\ 8 & 5,98 & 9,89 & 9 \\ 6,99 & 4,99 & 9 & 9,98 \end{bmatrix} \text{ Conclusions.}$$

I.2 Affichage graphique et alphanumérique

En ce qui concerne l'affichage graphique, il est indispensable de placer le titre et les légendes nécessaires à la compréhension de chaque figure.

1D

- + Pour un vecteur représentant le temps $t = [0 \ 0,01 \ 0,02 \ \dots \ 1]$ (en secondes). Tracer une sinusoïde de fréquence 5 Hz.
- + Pour une exponentielle complexe de même fréquence et pour le même vecteur du temps, tracer dans une même fenêtre graphique quatre courbes représentant la partie réelle, la partie imaginaire, le module et l'argument. Faire de même pour une exponentielle complexe amortie (coefficent d'amortissement $a = 1$).
- Tracer une distribution gaussienne centrée normalisée sur un intervalle de trois écart-type σ . Vérifier par intégration numérique simple¹ que l'on retrouve 68% de la répartition dans $[-\sigma, \sigma]$, 95% dans $[-2\sigma, 2\sigma]$, et 99% dans $[-3\sigma, 3\sigma]$. Tracer la fonction de répartition approchée de cette gaussienne en utilisant la fonction `cumsum`.

2D

- + La fonction de Rosenbrock $f(x, y) = (x-1)^2 + p(x^2-y)^2$ (avec p pris par exemple à 100) est souvent utilisée en optimisation pour comparer différents algorithmes. Tracer cette fonction sur un intervalle judicieux et justifier son surnom de "banana function". Tracer le logarithme de cette fonction pour faire apparaître plus clairement son minimum².
- Charger les images contenues dans les fichiers `clown.mat` et `gatlin.mat` et les rendre de même taille en supprimant les lignes et les colonnes excédentaires. En interchangeant les modules et les phases des transformées de Fourier discrètes³ de ces images, construire deux nouvelles images (la transformée de Fourier discrète 2D est calculée par `fft2` et son inverse par `ifft2`). Afficher côté à côté les quatre images et conclure.

1. Par la méthode des rectangles, on considère par exemple que pour a et b proches : $\int_a^b f(x)dx \approx f(a)(b-a)$.
 2. Utiliser éventuellement la fonction `view` pour visualiser cette fonction d'un point de vue différent.
 3. On ne rentrera pas ici dans le détails de ces opérations, l'objectif étant de programmer en Matlab.

I.3 Programmation et utilisation de fonctions

min2D

La fonction `min` de Matlab utilisée pour un vecteur renvoie la plus petite valeur contenue dans le vecteur et éventuellement sa position. Créer une fonction Matlab `min2D` qui pour une matrice renvoie la plus petite valeur contenue dans cette matrice ainsi que sa position en ligne et en colonne. Chercher en utilisant votre fonction la position du minimum de la matrice `magic(22)`.

Sinus cardinal

La fonction sinus cardinal $f(x) = \frac{\sin(\pi x)}{\pi x}$ doit son nom au fait qu'elle s'annule pour les entiers cardinaux. Créer une fonction Matlab `sinc` qui renvoie la valeur du sinus cardinal de son argument. Bien évidemment l'argument peut être un simple scalaire mais également un vecteur ou une matrice.

II Quelques problèmes numériques simples

II.1 Dérivation numérique

Il peut être utile lors de la résolution de problèmes de physique de connaître la dérivée d'une fonction à partir de sa valeur en certains points (dérivée numérique). Cela peut être le cas si l'on cherche le champ qui dérive d'un potentiel ou si l'on cherche la vitesse d'un objet à partir de sa position.

Dérivation d'une fonction en un point

On va essayer de calculer numériquement la dérivée d'une fonction en un point à partir des relations :

$$f'(t) = \lim_{h \rightarrow 0} \frac{f(t+h) - f(t)}{h} \quad \text{et} \quad f'(t) = \lim_{h \rightarrow 0} \frac{f(t+h) - f(t-h)}{2h}$$

- Créer une fonction Matlab simple nommé `fun` qui renvoie la valeur de la fonction mathématique étudiée (par exemple une sinusoïde).
- Pour différentes valeurs de h approcher numériquement la dérivée de cette fonction et comparer le résultat à sa dérivée calculée analytiquement. Faire varier h de 1 à 10^{-12} (on ne peut malheureusement aller jusqu'à zéro) et comparer les erreurs sur l'approximation pour les deux formules.
- Tracer l'erreur en fonction de h en échelle linéaire et en échelle loglog sur la même figure. Conclusion ?

Calcul de la vitesse à partir de la trajectoire

On dispose maintenant dans un vecteur x de la position d'un objet à certains instants t et l'on va à partir des relations précédentes de calculer sa vitesse.

- Construire un vecteur du temps et un vecteur de position correspondant à une trajectoire de votre choix.
- À partir des approximations sur les deux formules précédentes, calculer la vitesse de l'objet aux instants t . Peut-on aisément calculer la vitesse à tout instant t ? Comment régler le problème en $t = 0$ et en $t = t_{\max}$.
- Afficher les vitesses calculées numériquement, ainsi que la vitesse calculée analytiquement. Que remarque-t-on ? Afficher l'erreur sur les vitesses calculées.
- Tracer la trajectoire de l'objet dans le plan de phase (vitesse en fonction de la position).

II.2 Intégration numérique : à la recherche de π

La résolution de problèmes physiques amènent parfois au calcul d'intégrales à une ou plusieurs dimension. Dans le meilleur cas, l'intégrale peut être calculée analytiquement mais ce n'est pas toujours le cas. Il faut alors faire appel à des méthodes numériques d'intégration.

On va s'appuyer sur un exemple simple pour illustrer différentes méthodes d'intégration numériques. On sait qu'un disque de rayon unité a une surface de π . En exploitant les symétries et en considérant uniquement un quart de disque cela s'écrit sous forme intégrale : $\int_0^1 \sqrt{1-x^2} dx = \frac{\pi}{4}$.

Intégration par une technique de Monte-Carlo

Les techniques de Monte-Carlo sont des techniques aléatoires très utilisées pour calculer numériquement des intégrales multidimensionnelles. On va chercher à calculer π par une technique de ce type.

Si l'on tire aléatoirement un grand nombre de points dans le carré $[0, 1] \times [0, 1]$, le rapport entre le nombre de points appartenant au disque centré de rayon unité et le nombre total de point devrait tendre vers le rapport des deux surfaces à savoir $\frac{\pi}{4}$. Pour effectuer et illustrer cela sous Matlab :

- tirer de façon aléatoire un nombre N de points dans $[0, 1] \times [0, 1]$;
- compter le nombre de points appartenant au disque unité et en déduire π ;
- afficher ces points dans une figure (une croix pour chaque point);
- afficher en vert le quart de cercle unité dans $[0, 1] \times [0, 1]$;
- afficher en rouge les points intérieurs au cercle unité;
- afficher en bleu les points extérieurs au cercle unité;
- afficher la convergence de la valeur de π estimée en fonction du nombre de points N .

Intégration par les méthodes des rectangles et des trapèzes

On peut chercher à calculer l'intégrale par une technique d'intégration déterministe visant à découper l'intervalle d'intégration en petits intervalles.

- La méthode des rectangles avant (forward) considère que pour a et b petit : $\int_a^b f(x) dx \approx f(a)(b-a)$. Estimer π en calculant l'intégrale par cette méthode.
- La méthode des trapèzes considère que pour a et b petit : $\int_a^b f(x) dx \approx \frac{f(a)+f(b)}{2}(b-a)$. Estimer π en calculant l'intégrale par cette méthode.
- Pour ces deux méthodes, tracer graphiquement l'aire réellement calculée. La différence entre ces méthodes apparaîtra pour un nombre petit d'intervalles d'intégration.
- Par ces méthodes, combien faut-il prendre de points pour avoir une erreur inférieure à 10^{-2} .