Calcul Formel et Numérique, TD 2

Avant de lancer Octave, il convient de se placer dans un répertoire de travail où les fichiers de définitions de fonctions seront sauvegardés. Pour cela, créer un répertoire calcul, puis un sous-répertoire TD2. Placez vous dans ce répertoire et lancez Octave. Vous créerez un fichier TD2.m qui permettra à l'enseignant de vérifier votre travail à la fin de la séance.

1 Fonctions

La fonction sinus cardinal
On considère la fonction suivante

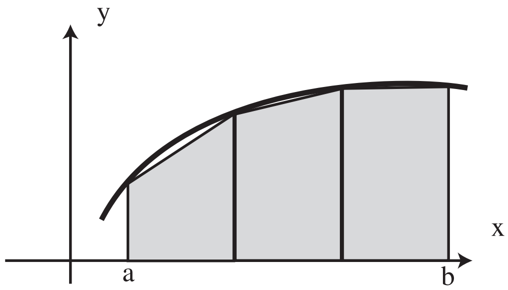
```
function p=mysinc(v)
% p=mysinc(v).
%Calcule p = sin(pi*v)/(pi*v)
        p = sin(pi*v)/(pi*v);
end
```

† A l'aide d'un éditeur de texte, écrire cette fonction dans le fichier *mysinc.m* que vous sauvegarderez dans le répertoire TD2.
Question 1.1: *Que se passe-t-il lorsque v est un vecteur ?* ....
† Modifier la fonction *mysinc* pour que l'on puisse utiliser un vecteur en entrée.
Question 1.2: *Que se passe-t-il alors lorsque v est une matrice* ....
On prend $x = [-0.3, 0, 0.3; 0, 0.1, 0.2]$
Question 1.3: *Que se passe-t-il lorsqu'on exécute $mysinc(x)$? Pour quelle raison a-t-on ce comportement ?* ....
Nous allons modifier la fonction *mysinc* pour que le calcul de $mysinc(x)$ retourne le bon résultat. A l'aide de la fonction `find`, nous allons chercher à déterminer la position des éléments non nuls de v. Dans la fonction *mysinc*, nous allons modifier v afin de l'organiser sous forme de vecteur.
† Affecter à nl et nc le nombre de lignes et le nombre de colonnes de v (utiliser la fonction `size`).
† Utiliser la fonction `reshape` pour modifier v et en faire un vecteur de $nl * nc$ lignes et 1 colonne.
† A l'aide de la fonction `find`, construire le vecteur ind contenant les positions des éventuels éléments non nuls de v
On remarque que le sinus cardinal de 0 est 1. Il suffit donc pour p de partir d'un vecteur de la taille de v plein de 1 et de ne modifier que les éléments d'indice appartenant à ind.
† Construire le vecteur p dont tous les éléments sont 1 et qui a la même taille que v.
† Remplir le vecteur p pour qu'il soit égal à $sin(pi * v)/(pi * v)$.

† Modifier la forme de y à l'aide de *reshape* pour revenir à la forme de v.
Question 1.4: *Calculer $mysinc(x)$ où x est le vecteur défini ci dessus.* . . . .

2 Calcul approché d'intégrales

Pour calculer l'aire sous une courbe, on peut remplacer la surface complète par un domaine constitué de trapèzes. Nous allons écrire une fonction permettant cette opération puis nous testerons l'erreur commise.



Fonction de calcul d'aire
Nous cherchons à calculer l'aire sous la fonction $f : x \mapsto \exp(x) - 2x^2$ pour x allant de -1 à 3. Pour cela nous diviserons l'intervalle $[-1, 3]$ en n sous intervalles.
Question 2.1: *Ecrire la formule donnant l'aire du trapèze délimité par les sommets $(x_i, 0)$, $(x_i, f(x_i))$, $(x_{i+1}, f(x_{i+1}))$, $(x_{i+1}, 0)$* . . . .
† Créer une fonction *integexp* avec une variable d'entrée n qui calcule l'aire sous la courbe à partir de la méthode des trapèzes. On calculera d'abord le vecteur A qui contient l'aire de chacun des trapèzes.
Erreur en fonction du pas
Nous cherchons à estimer l'erreur commise en fonction de la largeur des trapèzes h. Pour cela, nous exécuterons le même calcul avec un nombre d'intervalles différents. Dans un script, on fera successivement le calcul pour tous les nombres d'intervalles dans le vecteur $nv = logspace(1, 5, 20)$.
Question 2.2: *Quelle est la valeur théorique de l'aire sous la courbe ?* . . . .
† Dans un script, calculer l'erreur pour les différents nombre d'intervalles du vecteur nv (on fera une boucle de 1 à $length(nv)$). On calculera deux vecteurs : le vecteur h qui contient la largeur des trapèzes et le vecteur err qui contient les erreurs pour chacun de ces nombres d'intervalles.
† Tracer err en fonction de h en échelle log-log (on pourra utiliser la fonction `loglog`).
Question 2.3: *Quelle est l'allure de la courbe obtenue ?* . . . .
Question 2.4: *En déduire que l'erreur suit une loi en $err = e_0 h^n$. Précisez la valeur de n.* . . . .