TP°2 : affichage de données – partie 1

Elément de cours : Afficher des courbes

Affichage basique

Octave a des fonctionnalités avancées d'affichage graphique de données grâce à un module externe : **GNUPLOT** (`http://www.gnuplot.org`).

Les commandes de base sont **`plot(x,y)`** [affichage de fonction continue] et **`stem(x,y)`** [affichage de fonction discrète] où x et y sont des **coordonnées** ou **des vecteurs** passés en paramètres. Il existe d'autres commandes pour afficher des données sous forme d'histogrammes ou de barres (**`bar`**, **`bar3`**, **`barh`**, **`bar3h`**, **`hist`**, **`pie`**, **`pie3`** …)

Affichage évolué

Bien évidemment, il est possible d'améliorer le graphe de base pour y ajouter des labels, titres

Titre

`title``('titre')` → ajoute un titre au graphique

Labels

`xlabel``(' label x')` → ajoute un label sur l'axe `x`

`ylabel``('label y')` → ajoute un label sur l'axe `y`

Légende

`legend``('legende')` → ajoute une légende au graphe

Axes

`axis``([x1 x2 y1 y2])` → affiche la courbe entre `x1` et `x2` sur l'axe `x` et `y1` et `y2` sur l'axe `y`

`axis equal` → rend les axes isométriques

Quadrillage

Afin d'afficher les résultats avec un quadrillage, utilisez la commande `grid on | off`

Plusieurs graphes dans une même fenêtre

`subplot``(nombre_lignes, nombre_colonnes, selection)`

L'argument `selection` permet de sélectionner le graphe courant dans la fenêtre. Les sous-fenêtres sont numérotées à partir du haut à gauche avec un parcours ligne par ligne.

Attention : A chaque commande `plot`, l'affichage est réinitialisé ! Une commande permet d'éviter ce (léger) problème : `hold on` pour empêcher la réinitialisation et `hold off` pour la permettre à nouveau.

Exercice 1 : Fonctions trigonométriques

Afficher dans une même fenêtre les fonctions `cos(x)`, `sin(x)` entre `0` et `2π` en utilisant l'instruction `hold on`)

Ajouter maintenant des labels sur les axes, un titre et une légende au graphique généré.

Elément de cours : Afficher des courbes … suite

Affichage encore plus évolué

La fonction `plot` permet aussi de définir vos propres styles. Pour cela, il faut introduire une chaine de caractères après chaque paire de coordonnées :

```
plot(x,y,"formatage")
```

La chaine de formatage peut contenir :

- un des caractères `. + * ^ o x h d s`, ce qui va donner un graphe à différents points
- le caractère `-`, ce qui va donner un graphe à lignes ou, si utilisé en conjonction avec un des caractères précédents, un graphe à lignes et points
- un numéro ou une lettre définissant la couleur : **`k`** = noir, **`r`** = rouge, **`g`** = vert, **`b`** = bleu, **`m`** = magenta, **`c`** = bleu clair, **`y`** = jaune, **`w`** = blanc.

Exercice 2 : plot et subplot

On cherche à obtenir une représentation graphique de la fonction $f(x) = e^{-x}.\sin(4x)$ sur l'intervalle $[0,2]$.

1. Créer les tableaux X et Y via les commandes suivantes :

   `X=linspace(0,2*pi,101);` → créé un vecteur de 101 valeurs entre 0 et 2π

   `Y=exp(-X).*sin(4*X);` → l'ensemble des résultats de la fonction

2. Tracer alors la représentation graphique de la fonction `f` associée aux tableaux `X` et `Y`. En utilisant le zoom, déterminer une valeur approchée du maximum de f sur `[0,2]`. Comment peut-on affiner le tracé pour préciser ce maximum ?

3. En utilisant l'instruction `subplot`, tracer sur une même figure une représentation graphique des fonctions `f(x)`, $g(x) = x^2$ et $h(x) = x^2\sin(x)\exp(-x)$ sur l'intervalle `[-1,1]` en utilisant des couleurs différentes pour chacune d'entre elles.

Exercice 3 : Le Papillon de T. Fay

Ecrire un programme qui trace le « *Papillon de T. Fay* », courbe paramétrée en coordonnées polaires pour le paramètre theta variant de 0 à 2π

$$r = e^{\cos(\theta)} - 2\cos(4\theta)$$

Nota : penser à repasser en coordonnées cartésiennes avant d'afficher la courbe !

Exercice 4 : dichotomie & racine

Supposons une fonction $\mathbf{f(x) = e^{3x} - x^2}$.

La clé de la recherche des racines repose sur l'existence d'un encadrement préalable de cette racine.

S'il existe un couple `(a,b)` tel que le produit $f(a)f(b) < 0$ et si la fonction est continue, le théorème de la valeur intermédiaire nous dit que fonction s'annule au moins une fois à l'intérieur de cet intervalle.
La méthode de dichotomie est une méthode qui ne peut pas échouer, mais sa rapidité de convergence n'est pas la meilleure en comparaison avec d'autres méthodes.

Le principe de cette méthode est le suivant : soit une fonction `f` monotone sur un intervalle `[a0, b0]` telle que `f(a0).f(b0) < 0`. On sait alors qu'il existe une et une seule racine comprise dans cet intervalle. L´algorithme de la méthode de dichotomie est la suivante :
Tout d'abord, on calcule `f((a0 + b0)/2)`

- Si $f((a_0 + b_0)/2).f(a_0) < 0$, on définit un nouvel encadrement de la racine par le couple (a_1, b_1) tel que :
  $a_1 = a_0$
  $b_1 = (a_0 + b_0)/2$

- Si $f((a_0 + b_0)/2).f(a_0) > 0$, on définit un nouvel encadrement de la racine par le couple (a_1, b_1) tel que :
  $a_1 = (a_0 + b_0)/2$
  $b_1 = b_0$

Créer un programme qui :

- Demande à l'utilisateur d'entrer les valeurs d'un intervalle $[a_0, b_0]$ ainsi que le nombre `N` d'itérations de calcul et qui indique si la racine se trouve ou non dans cet intervalle.
- Donne la valeur approchée de la racine après les N itérations
- Affiche sous forme de graphique la courbe centrée sur la racine trouvée.

Modifier le programme afin de donner la valeur approchée de la racine lorsque la différence entre les bornes a_n et b_n devient inférieure à une certaine précision et qui sera demandée à l'opérateur.

Dans l'algorithme, il y a une opération que l'on répète tout le temps…
Comment pourrions-nous résoudre ce problème ? Coder la solution

Exercice 5 : calcul du cosinus par série entière

Ecrire un programme qui demande à l'utilisateur un entier naturel non nul `n` et un réel `x` permettant de calculer une approximation de la valeur de la fonction `cos(x)` en ne conservant que les n premiers termes du développement en série entière :

$$\cos(x) = 1 - \frac{x^2}{2!} + \cdots + (-1)^k \frac{x^{2k}}{(2k)!} + \cdots$$

Comparer graphiquement les différents résultats obtenus avec ceux de la fonction `cos(x)` de GNU Octave avec la commande `subplot` pour 1, 5 puis 20 termes du calcul.