Département de Mathématiques 2018/2019
Module: Analyse numérique 1
Fiche de T.P. N°3

Tracé de courbes

Pour tracer des courbes, Python n'est pas suffisant et nous avons besoin des bibliothèques NumPy et matplotlib.

Utilisation de plot()

L'instruction plot() permet de tracer des courbes qui relient des points dont les abscisses et ordonnées sont fournies dans des tableaux.

Exemple 1

```
import numpy as np
import matplotlib.pyplot as plt
x = np.array([1, 3, 4, 6,7,8])
y = np.array([2, 3, 5, 1,0.5,-1])
plt.plot(x, y)
plt.show() # affiche la figure à l'écran
```

Exemple 2

```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0, 2*np.pi, 30)
y = np.cos(x)
plt.plot(x, y)
```

Délimiter les axes du repère

Il est possible fixer indépendamment les domaines des abscisses et des ordonnées en utilisant les fonctions **xlim(xmin, xmax)** et **ylim(ymin, ymax)** par exemple:

```
plt.xlim(0, 2*np.pi)
plt.ylim(-2, 2)
```

On peut ajouter un titre grâce à l'instruction **title()**.

```
plt.title("Fonction cosinus")
```

Des labels sur les axes peuvent être ajoutés avec les fonctions **xlabel()** et **ylabel().**

```
plt.xlabel("abscisses")
plt.ylabel("ordonnees")
```

Affichage de plusieurs courbes

Pour afficher plusieurs courbes sur un même graphe, on peut procéder de la façon suivante :

```
x = np.linspace(0, 2*np.pi, 30)
y1 = np.cos(x)
y2 = np.sin(x)
plt.plot(x, y1)
plt.plot(x, y2)
plt.show()
```

Exemple avec un légende

```
x = np.linspace(0, 2*np.pi, 30)
y1 = np.cos(x)
y2 = np.sin(x)
plt.plot(x, y1, label="cos(x)")
plt.plot(x, y2, label="sin(x)")
```

plt.legend()
plt.show()

Formats de courbes

Il est possible de préciser la couleur, le style de ligne et de symbole ("marker") en ajoutant une chaîne de caractères de la façon suivante : **plot(abscisse, ordonnée, 'couleur forme_point forme_trait', paramètres)**

plt.plot(x, y1, "r- -", label="cos(x)")
plt.plot(x, y2, "b:o", label="sin(x)")
plt.legend()

Tracé multiples

Il est souvent utile de pouvoir avoir plusieurs figures à côté l'une de l'autre pour les comparer. 2 possibilités : 2 fenêtres contenant 1 figure, 1 fenêtre contenant 2 figures :

Possibilité 1 : 2 fenêtres contenant 1 figure :

*x = np.linspace(0, 2*np.pi, 30)*
y1 = np.cos(x)
y2 = np.sin(x)
plt.figure(1)
plt.plot(x, y1, 'r')
plt.figure(2)
plt.plot(x, y2, 'g')
plt.show()

Possibilité 2 : 1 fenêtre contenant 2 figures :

Nous allons utiliser la fonction subplot qui divise une fenêtre en plusieurs figures :

plt.figure(1)
plt.subplot(121)
plt.plot(x, y1, 'r')
plt.subplot(122)
plt.plot(x, y2, 'g')
plt.show()

Pour donner des titres aux différents subplot il faut passer par :

sub1 = plt.subplot(121)
plt.plot(x, y1, 'r')
sub1.set_title("Cos(x)")
sub2 = plt.subplot(122)
plt.plot(x, y2, 'g')
sub2.set_title("Sin(x)")

Exercice 1

1. Représenter dans un même figure les fonctions $\sin(x^2)$ et $1/x$ entre 0.2 et 2.
2. Tracer sur l'intervalle $[-5, 0]$ la fonction $x^2\cos(x)$ en trait plein bleu et la fonction $x\cos(x)$ en trait pointillé rouge.Ajouter un titre et une légende.

Exercice 2 Comparer une fonction et ses DL :

1. Représenter sur le même graphique la fonction exp (en trait plein) ainsi que les fonctions $x \to 1 + x$ et $x \to 1 + x + x^2/2$ (en pointillés) sur l'intervalle $[-2, 2]$.
2. De même, représenter sur l'intervalle $[-\pi, \pi]$ la fonction cos ainsi que les polynômes de Taylor de degré 2 et 4 associés à la fonction cos au voisinage de 0.