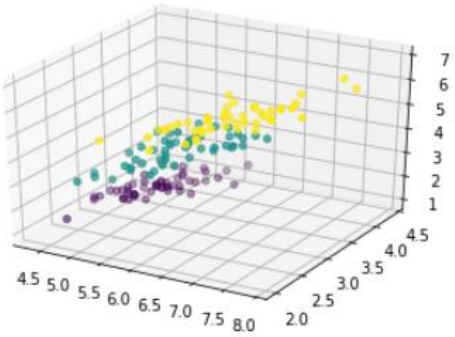
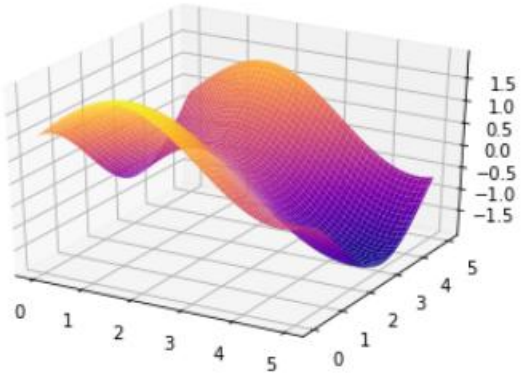
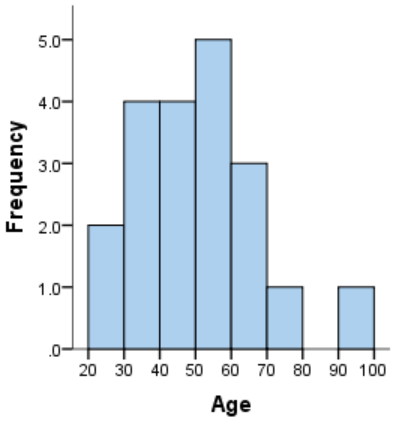
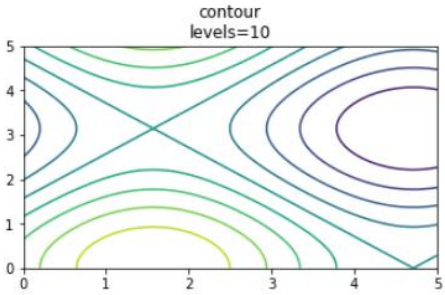
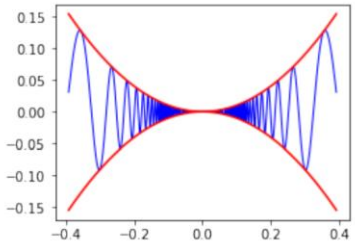
- Matplolib -









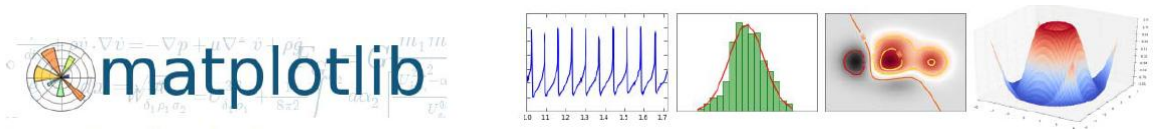


Le module Matplotlib

1. Le tracé de courbes dans Matplotlib

Le tracé de courbes est effectué grâce au riche module matplotlib, qui propose des outils de tracés multiples. Le site internet dédié à ce module (http://matplotlib.org/) vous montrera entre autre la galerie impressionnante de tracés possibles.

Nous allons vous montrer ceux qui sont les plus utilisés en ingénierie numérique et attendus aux concours.



Le module matplotlib, comme tout autre module de Python (numpy, math…) se charge grâce à la commande **import**. C'est la fonction **pyplot** de ce module qui permet de tracer des courbes.

```
import matplotlib.pyplot as plt
import numpy as np
```

2. Découverte de Matplotlib

Afin de personnaliser vos tracés (légendes, titres, axes….), il est important de maîtriser les outils suivants présents de base dans le module matplotlib.

Légendes : plt.xlabel('axe des abscisses') # Légende sur l'axe (Ox)
plt.ylabel('fonction sinus') # Légende sur l'axe (Oy)

Grille : plt.grid()

Bornes : plt.axis([xmin,xmax,ymin,ymax])

Couleurs du tracé : g = green, r = red, k = noir, b =bleu, c = cyan, m =magenta, y = yellow

ex : plt.plot(x,y,'r') trace en rouge

Symboles : mettre des symboles aux points tracés : symboles= *, - , + , . , etc...

ex : plt.plot(x,y,marker='*') trace avec * aux points de données et relie les points.

plt.plot(x,y,'*) trace les points en les marquants par * sans les relier.

Style du tracé : plt.plot(x,y,ls='style') style = - - tirets, -. points-tirets, : pointillés (ls signifie linestyle)

Épaisseur du trait : plt.plot(x,y,lw=flottant) (lw signifie linewidth)

Ajouter un titre : plt.title('Titre')

Ajouter du texte en un point donné: plt.text(1,-1,' Texte ', color='b')

Mettre une légende dans la fenêtre : legend()

ex : plt.plot(X,Y,label='cosinus') ou plt.legend(loc = 'upper left') # place la légende cosinus en haut à gauche

Le module Matplotlib

Découvrons à présent matplotlib avec quelques petits exercices simples avec les apports méthodologiques.

Tracer une ligne brisée

Pour tracer une ligne brisée reliant les points $A1(x_1,y_1)\dots An(x_n,y_n)$ **:**

```
X = [x1,...,xn]   # X est la liste des abscisses des points.
Y = [y1,...,yn]   # Y est la liste des ordonnées des points.
plt.plot(X,Y)     # trace une ligne brisée reliant les points A1, …, An
plt.show()        # affiche la fenêtre de graphique
```

Exercice 0 :
Tracer une ligne brisée reliant les points de coordonnées A(1,0), B(3,6), C(5,2) et D(7,3)

Tracer une courbe d'équation y=f(x)

Pour tracer la courbe d'équation *y = f(x)* :

```
X = np.linspace(a,b,N)          # np.linspace crée un vecteur de N (entier) valeurs équidistantes de a à b inclus !
Y = [f(x) for x in X]           # liste en compréhension, contenant les images par f des valeurs de X
plt.plot(X,Y)                   # trace les points de coordonnées (X[i],Y[i]) et les relie
plt.show()                      # affiche la fenêtre de graphique
plt.savefig('test1.pdf',format='pdf')  # f.pdf est au même endroit que le code python qui l'a généré !
```

Exercice 1:
Tracer le graphe de la fonction sinus sur [0,2π] , avec une légende "sinus" marquée en haut à droite et sauver fichier en pdf.

Tracer plusieurs courbes dans une même fenêtre

Pour tracer les courbes d'équations *y = f(x)* et *y = g(x)* dans une même fenêtre:

```
X = np.linspace(a,b,N)
Y = [f(x) for x in X]
Z = [g(x) for x in X]
plt.plot(X,Y)
plt.plot(X,Z)                   # on peut superposer ainsi autant de graphes que l'on souhaite !
plt.show()
plt.savefig('test2.pdf',format='pdf')
```

Exercice 2 :
Tracer le graphe des fonctions sinus, cosinus sur [0,2π] avec 2 légendes centrées à gauche dans une même fenêtre.

Exercice 3:
Tracer plusieurs graphes de la fonction sinus sur [0,2π] , un pour chaque valeur de *N* dans [1,10] où N précise le nombre de valeurs équidistantes de 0 à 2π .

Le module Matplotlib

Tracer sur plusieurs fenêtres

Ouvrir plusieurs fenêtres à l'écran :

```
X = np.linspace(a,b,N)
Y = [f(x) for x in X]
Z = [g(x) for x in X]
plt.figure()              # permet d'ouvrir une fenêtre
plt.plot(X,Y)

plt.figure()              # permet d'ouvrir une autre fenêtre etc...
plt.plot(X,Z)
plt.show()
```

Exercice 4:
Tracer le graphe des fonctions cosinus, sinus et exponentielle avec des légendes de votre choix sur chacune d'elle dans trois fenêtres différentes et sauvegarder les figures en pdf.

Exercice 5 :
Tracer cosinus et sinus dans une même fenêtre, cosinus en rouge et ligne continue, sinus en bleu et pointillés , mettre une légende "axe des abscisses" et en ordonnée: "en bleu =cosinus, en rouge=sinus". Afficher aussi la grille.

Tracer plusieurs sous-graphiques dans une même fenêtre

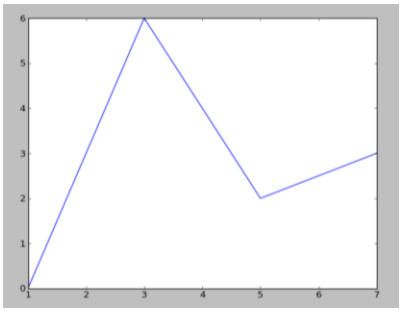
Plusieurs sous-graphiques dans une même fenêtre :

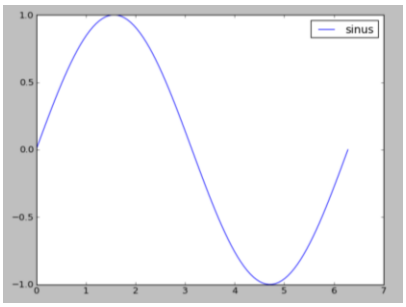
```
T = np.linspace(a,b,N)
plt.subplot(3,2,1)  # 1er sous-graphique parmi troix lignes et deux colonnes
Y = [f(t) for t in T]
plt.plot(T,Y)
plt.grid()

plt.subplot(3,2,2)  # 2e sous-graphique, parmi trois lignes et deux colonnes
Z = [g(t) for t in T]
plt.plot(T,Z)
plt.grid

plt.subplot(3,2,3) # 3e sous-graphique, parmi trois lignes et deux colonnes
plt.subplot(3,2,4) # inutile de les mettre, juste pour illustrer
plt.subplot(3,2,5)
plt.subplot(3,2,6)
plt.show()
```

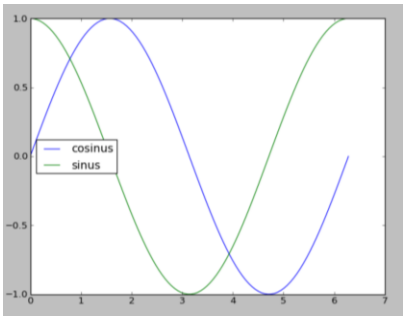
Exercice 6 :
Tracer les fonctions cosinus, sinus, log, exponentielle dans des sous-graphiques d'une même fenêtre. On ne veut pas plus de quatre sous-graphiques. Mettre les titres centre haut des tracés et la grille pour chacun.

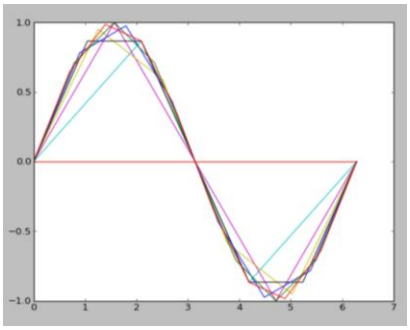
Le module Matplotlib

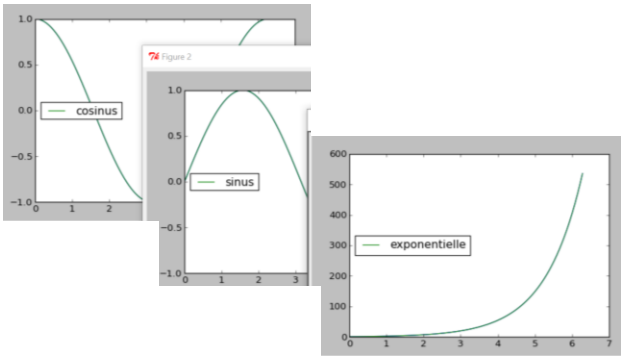
Eléments de correction :





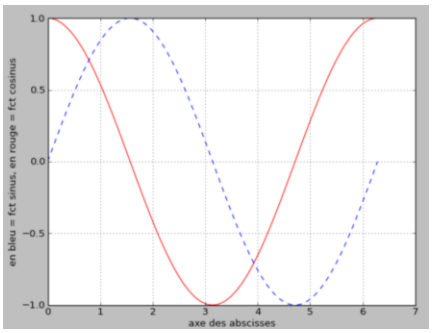






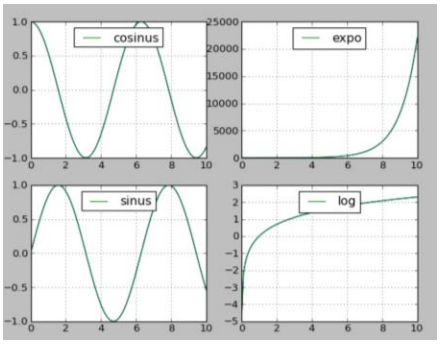

```
from math import *
import matplotlib.pyplot as plt
import numpy as np

## exercice 0

X = [1,3,5,7]
Y = [0,6,2,3]
plt.plot(X,Y)
plt.show()

## exercice 1

X = np.linspace(0,2*pi,100)
Y = [sin(x) for x in X]
plt.plot(X,Y,label='sinus')
plt.legend(loc = 'upper right')
plt.savefig('sinus.pdf',format='pdf')
plt.show()


## exercice 2

plt.figure()
X = np.linspace(0,2*pi,100)
Y = [sin(x) for x in X]
Z = [cos(x) for x in X]
plt.plot(X,Y,label = 'cosinus')
plt.plot(X,Z,label = 'sinus')
plt.legend(loc = 'center left')
plt.show()


## exercice 3

for N in range(1,11,1):
    X = np.linspace(0,2*pi,N)
    Y = [sin(x) for x in X]
    plt.plot(X,Y)
plt.show()

# exercice 4

X = np.linspace(0,2*pi,100)
Y = [cos(x) for x in X]
Z = [sin(x) for x in X]
T = [exp(x) for x in X]

plt.figure()
plt.plot(X,Y)
plt.plot(X,Y,label = 'cosinus')
plt.legend(loc = 'center left')
plt.savefig('cosinus.pdf',format='pdf')

plt.figure()
plt.plot(X,Z)
plt.plot(X,Z,label = 'sinus')
plt.legend(loc = 'center left')
plt.savefig('sinus.pdf',format='pdf')

plt.figure()
plt.plot(X,T)
plt.plot(X,T,label = 'exponentielle')
plt.legend(loc = 'center left')
plt.savefig('exp.pdf',format='pdf')
plt.show()
```

Le module Matplotlib




```
# exercice 5

plt.figure()
X = np.linspace(0,2*pi,100)
Y = [sin(x) for x in X]
Z = [cos(x) for x in X]
plt.plot(X,Z,'r')
plt.plot(X,Y,'b',ls='--')
plt.xlabel('axe des abscisses')
plt.ylabel('en bleu = fct sinus, en rouge = fct cosinus')
plt.grid()
plt.show()
```

```
# exercice 6


T = np.linspace(0.01,10.,100)
Y = [cos(t) for t in T]
Z = [exp(t) for t in T]
U = [sin(t) for t in T]
V = [log(t) for t in T]

plt.subplot(2,2,1)
plt.plot(T,Y)
plt.plot(T,Y,label = 'cosinus')
plt.legend(loc = 'upper center')
plt.grid()

plt.subplot(2,2,2)
plt.plot(T,Z)
plt.plot(T,Z,label = 'expo')
plt.legend(loc = 'upper center')
plt.grid()

plt.subplot(2,2,3)
plt.plot(T,U)
plt.plot(T,U,label = 'sinus')
plt.legend(loc = 'upper center')
plt.grid()

plt.subplot(2,2,4)
plt.plot(T,V)
plt.plot(T,V,label = 'log')
plt.legend(loc = 'upper center')
plt.grid()

plt.show()
```



3. Quelques exercices d'approfondissement

Exercice 1 :

Ecrire un script pour tracer, en bleu, la courbe représentant la fonction :

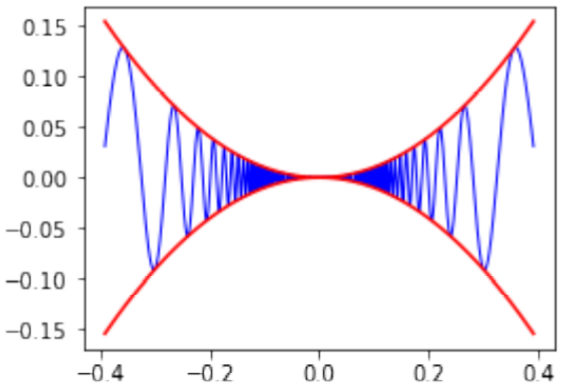
$f : x \to x^2 sin(1/x^2)$ *sur l'intervalle* $I = [-\pi/8 \quad \pi/8]$

Tracer sur le même graphe, en rouge, les courbes représentant les fonctions :
$g : x \to x^2$ *et* $-g$

On prendra N=1000 points d'échantillonnage pour le tracé

Solution :

```
N=1000 # nb de points d'échantillonnage pour le tracé de f
xi=np.linspace(-np.pi/8,np.pi/8,N) # vecteur des valeurs d'abscisses
fi=xi**2*np.sin(1/xi**2) # vecteur de valeurs yi=f(xi)
plt.plot(xi,fi,'b',linewidth=1) # en bleu
plt.plot(xi,xi**2,'r')          # fonction g
plt.plot(xi,-xi**2,'r')         # fonction -g
plt.show()
```



Exercice 2 :

Vous allez tracer la courbe paramétrée de Lissajous.

$$\begin{cases} x(t) &= a\sin(t) \\ y(t) &= b\sin(nt+\varphi) \end{cases}$$

Ecrire une fonction *traceLissajous(n,phi)* qui trace la courbe paramétrique pour t variant dans l'intervalle [0 100π], avec 2000 points de calculs, pour a=b=1. Puis tester là avec n=8/5 et phi=π/6

Solution :

```
def traceLissajous(n,phi):
    ti=np.linspace(0,100*np.pi,int(2e3)) # 2000 points de calcul
    xi=np.sin(ti)
    yi=np.sin(n*ti+phi)
    plt.plot(xi,yi,'k')
    plt.axis('equal')
    plt.show()
```

```
traceLissajous(8/5,np.pi/6) # n=8/5 et phi=pi/6
```