ACTIVITÉ

LANGAGE PYTHON
CALCUL SCIENTIFIQUE

1 - INTERPOLATION DE COURBES

En utilisant Spyder recopier et exécuter le programme ci-dessous.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import interpolate
x=np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
y=np.array([1., 0.720, 0.511, 0.371, 0.260, 0.190, 0.133, 0.099, 0.069, 0.048])
f = interpolate.interp1d(x, y, kind='cubic')     # ou kind = 'linear'
xnew = np.linspace(0, 9, 50)                     # Nouvelle abscisse pour f(x)
ynew = f(xnew)                                   # Calcul des ordonnées de f(x)
plt.figure()                                     # Initialisation du tracé
plt.plot(xnew, ynew, '-')                        # Trace de la fonction
plt.plot(x, y, 'x')                              # Trace des points
plt.show()                                       # Affichage
```

Sauvegarder ce programme dans un dossier nommé SciPython sous le nom prg8.py.

Remarque :

- La fonction scipy.interpolate.interp1d() retourne une fonction (mathématique) à une dimension interpolée à partir d'une série de points.

ACTIVITÉ

LANGAGE PYTHON
CALCUL SCIENTIFIQUE

2 - RÉGRESSION LINÉAIRE

En utilisant Spyder recopier et exécuter le programme ci-dessous.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import linregress
x = np.array([0, 1.01, 2.02, 2.99, 3.98])
y = np.array([10.02, 7.96, 6.03, 4.04, 2.01])
a, b, rho ,_ ,_ = linregress(x, y)  # Regression lineaire
print("a = ", a)                    # Affichage de coefficient directeur
print("b = ", b)                    # Affichage de l'ordonnee a l'origine
print("rho = ", rho)                # Affichage du coefficient de correlation
xnew = np.linspace(0, 4, 50)        # Nouvelle abscisse pour la modelisation
ynew = a*xnew + b                   # Ordonnees de la fonction affine
plt.figure()                        # Initialisatin du tracé
plt.plot(xnew, ynew, '-')           # Trace de la droite
plt.plot(x, y, 'x')                 # Trace des points et de la fonction affine
plt.title('Régression linéaire')    # Titre
plt.xlabel('x')                     # Etiquette en abscisse
plt.xlim(-1,5)                      # Echelle en abscisse
plt.ylabel('y')                     # Etiquette en ordonnee
plt.ylim(0, 12)                     # Echelle en ordonnee
plt.show()                          # Affichage
```

Sauvegarder ce programme dans le dossier nommé SciPython sous le nom prg9.py.

Remarque :

- La fonction scipy.stats.linregress() retourne les paramètres de la régression linéaire d'une série de points.

ACTIVITÉ

LANGAGE PYTHON
CALCUL SCIENTIFIQUE

3 - MODÉLISATION À PARTIR D'UN POLYNÔME

En utilisant Spyder recopier et exécuter le programme ci-dessous.

```
# Données expérimentales
T = [0.0, 0.04, 0.08, 0.12, 0.16, 0.2, 0.24, 0.28, 0.32, 0.36, 0.4, 0.44, 0.48,
0.52, 0.56, 0.6, 0.64, 0.68, 0.72, 0.76, 0.8, 0.84, 0.88, 0.92]
X = [-0.95, -0.88, -0.80, -0.72, -0.64, -0.56, -0.48, -0.39, -0.31, -0.23,-0.15,
-0.07, 0.01, 0.10, 0.18, 0.26, 0.34, 0.43, 0.51, 0.59, 0.66, 0.74, 0.82, 0.89]
Y = [-0.05, 0.07, 0.17, 0.25, 0.33, 0.39, 0.43, 0.47, 0.48, 0.49, 0.49, 0.47,
0.43, 0.39, 0.32, 0.25, 0.16, 0.05, -0.06, -0.19, -0.33, -0.49, -0.65, -0.79]

a, b, c = np.polyfit(X, Y, 2)                    # Modelisation par un polynome de degre 2
print("a = ", a)                                 # Affichage
print("b = ", b)                                 # Affichage
print("c = ", c)                                 # Affichage
X = np.array(X)                                  # Creation d'un tableau Numpy pour X
Y_modele = a*X**2+b*X+c                          # Creation d'un tableau Numpy pour Y du modele
plt.figure()                                     # Initialisatin du tracé
plt.plot(X,Y,'b+', label = 'trajectoire')        # Courbe des mesures
plt.plot(X, Y_modele, 'r-', label = "modèle")    # Courbe du modèle
plt.xlabel("x")                                  # Etiquette en abscisse
plt.ylabel("y")                                  # Etiquette en ordonnée
plt.title("Trajectoire et modèle associé")       # Titre
plt.legend()                                     # Affichage legend
plt.show()                                       # Affichage fenetre
```

Sauvegarder ce programme dans le dossier nommé SciPython sous le nom prg10.py.

QUESTION **Quel type de courbe obtient-on ?** _____

ACTIVITÉ

LANGAGE PYTHON
CALCUL SCIENTIFIQUE

4 - MODÉLISATION À PARTIR D'UNE FONCTION QUELCONQUE

En utilisant Spyder recopier et exécuter le programme ci-dessous.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
x=np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
y=np.array([0., 3.935, 6.321, 7.769, 8.647, 9.179, 9.502, 9.698, 9.817, 9.889])
def fct(x, A, tau):                          # Definition de la fonction
    return A*(1-np.exp(-x/tau))              # Expression du modele
(A, tau), pcov = curve_fit(fct, x, y)        # Determination des parametres du modele
print("A = ", A)                             # Affichage de A
print("tau =", tau)                          # Affichage de tau
xnew = np.linspace(0, 10, 50)
ynew = fct(xnew, A, tau)
plt.figure()                                 # Initialisation du tracé
plt.plot(xnew, ynew, '-')                    # Courbe des mesures
plt.plot(x, y, 'x')                          # Tracé des valeurs
plt.show()                                   # Affichage fenetre
```

Remarque :

- La fonction scipy.optimize.curve_fit() retourne les paramètres de modélisation à partir d'une fonction quelconque.

Sauvegarder ce programme dans le dossier nommé SciPython sous le nom prg11.py.