

Fiche méthode n° 4

Optimisation : *curve fitting* avec python**Principe**

On dispose de deux listes de données $X = [x_1, x_2, \dots, x_n]$ et $Y = [y_1, y_2, \dots, y_n]$, représentant n valeurs des variables x et y .

On se donne une fonction f de la variable x , dépendant de paramètres $p_1, p_2, \dots, p_k$.

L'optimisation consiste à chercher le jeu de valeurs des paramètres p_i telle que la fonction $y = f(x, p_1, \dots, p_k)$ s'approche « au mieux » des n points donnés.

Utilisation de `curve_fit` de la bibliothèque `scipy`

La bibliothèque `scipy` met à notre disposition un outil permettant cette optimisation.

```
from scipy.optimize import curve_fit
```

La syntaxe d'utilisation est

```
curve_fit(f, X, Y, p0 = [p1, ..., pk])
```

où

f est la fonction utilisée pour l'optimisation

X est la liste des données $X = [x_1, x_2, \dots, x_n]$

Y est la liste des données $Y = [y_1, y_2, \dots, y_n]$

$[p_1, \dots, p_k]$ est la liste des estimations initiales des paramètres $p_1, \dots, p_k$.

➤ on peut ne pas fournir d'estimation initiale des paramètres avec l'option `p0 = None`.

La fonction `curve_fit` retourne deux listes :

- la liste des paramètres p_k qui optimise la fonction par rapport à l'ensemble des données;
- un tableau qui donne la covariance de la liste des paramètres.

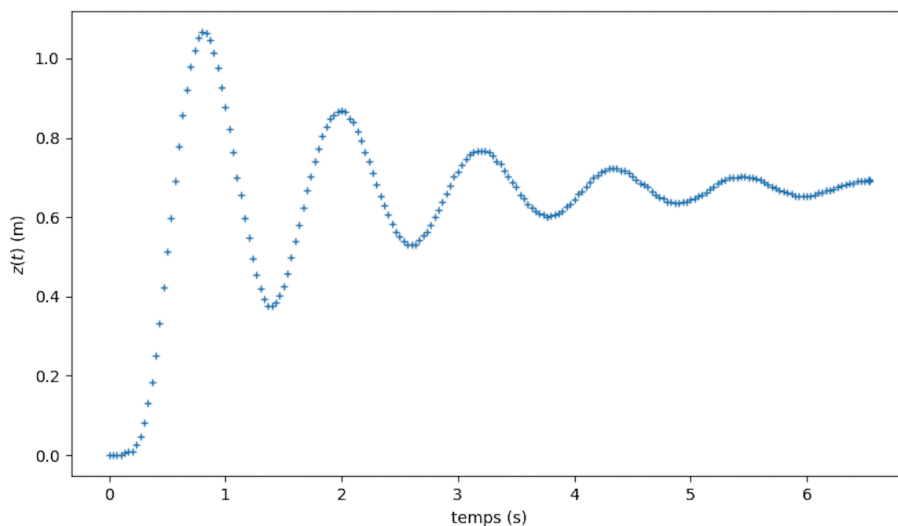
➤ Nous n'utiliserons que la première liste, que l'on appelle usuellement `popt`, la seconde étant appelée `pcov`.

La syntaxe d'appel est donc typiquement

```
popt, pcov = curve_fit(f, X, Y, p0 = [p1, ..., pk])
```

Exemple

On dispose d'un enregistrement des oscillations amorties d'un ressort :



Les données sont enregistrées dans le fichier `pointage.csv`.

Les instants de pointage sont dans la première colonne, et les altitudes dans la troisième colonne (la seconde donne la position horizontale, qui ne nous intéresse pas ici).

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 import scipy.optimize
4
5 fichier = open("pointage.csv","r")
6 donnees_brutes = fichier.readlines()
7 fichier.close()
8
9 donnees = []
10 for i in range(len(donnees_brutes)):
11     donnees_brutes[i] = donnees_brutes[i][:-1]
12     donnees.append(donnees_brutes[i].split(";"))
13
14 T,Z = [],[]
15 for i in range(2,len(donnees_brutes)):
16     T.append(float(donnees[i][0].replace(",",".")))
17     Z.append(float(donnees[i][2].replace(",",".")))
18
19 def modele(t,A,B,w,phi,tau):
20     return A+B*np.cos(w*t+phi)*np.exp(-t/tau)
21
22 p,pcov = sp.optimize.curve_fit(modele,T,Z,p0 = None)
23
24 Z_fit = []
25 for t in T:
26     Z_fit.append(modele(t,p[0],p[1],p[2],p[3],p[4]))
27
28 plt.close()
29 plt.plot(T,Z,linestyle='',marker='+',markersize=4)
30 plt.plot(T,Z_fit,linewidth=1)
31 plt.xlabel('temps (s)')
32 plt.ylabel('$z(t)$ (m)')
33 plt.show()

```

La liste `p` contient les valeurs optimisées des paramètres `A`, `B`, `w`, `phi`, `tau`.

