Annexe 1 : Utilisation de la fonction bisect pour résoudre une équation

▪La fonction bisect doit être chargée à partir du module scipy.optimize :

Chargement du module : from scipy.optimize **import** bisect

▪Cette méthode de résolution numérique concerne une équation du type f (x) = 0 , il faut préciser les bornes de l'intervalle [a,b] dans lequel on cherche la solution .

▪*Utilisation de bisect*

① *Définir la fonction f*

② *Introduire la fonction bisect sachant qu'elle renvoie un nombre ,*
les paramètres d'entrée de la fonction bisect sont la fonction f , les bornes de l'intervalle

solution = bisect (f,a ,b)

▪*Exemple : pH d'une solution d'acide éthanoïque de concnetration C = 0,01 molL^{-1}*

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import bisect

#Données
pKa= 4.8
C=0.01 #concentration initiale en mol/L

Ka= 10**(-pKa)

#equation vérifiée par x = [H3O+] écrite sous la forme f(x)=0

def f(x):
return(Ka*(C-x)-x**2)

#résolution avec bisect
solution = bisect(f, 0,14)

#Valeur du pH
pH= -np.log10(solution)
```

bisect

Definition : `bisect(f, a, b, args=(), xtol=1e-12, rtol=4.4408920985006262e-16, maxiter=100, full_output=False, disp=True)`
Type : Function of scipy.optimize.zeros module

Find root of a function within an interval.

Basic bisection routine to find a zero of the function *f* between the arguments *a* and *b*. *f(a)* and *f(b)* can not have the same signs. Slow but sure.

Parameters

f : function
Python function returning a number. *f* must be continuous, and f(a) and f(b) must have opposite signs.
a : number
One end of the bracketing interval [a,b].
b : number
The other end of the bracketing interval [a,b].
xtol : number, optional
The routine converges when a root is known to lie within *xtol* of the value return. Should be >= 0. The routine modifies this to take into account the relative precision of doubles.
rtol : number, optional
The routine converges when a root is known to lie within *rtol* times the value returned of the value returned. Should be >= 0. Defaults to `np.finfo(float).eps * 2`.
maxiter : number, optional
if convergence is not achieved in *maxiter* iterations, and error is raised. Must be >= 0.
args : tuple, optional
containing extra arguments for the function *f*. *f* is called by `apply(f, (x)+args)`.
full_output : bool, optional
If *full_output* is False, the root is returned. If *full_output* is True, the return value is `(x, r)`, where x is the root, and r is a *RootResults* object.
disp : bool, optional
If True, raise RuntimeError if the algorithm didn't converge.

Returns

x0 : float
Zero of *f* between a and *b*.
r : RootResults (present if `full_output = True`)
Object containing information about the convergence. In particular, `r.converged` is True if the routine converged.

See Also

brentq, brenth, bisect, newton fixed_point : scalar fixed-point finder fsolve : n-dimensional root-finding