tp2

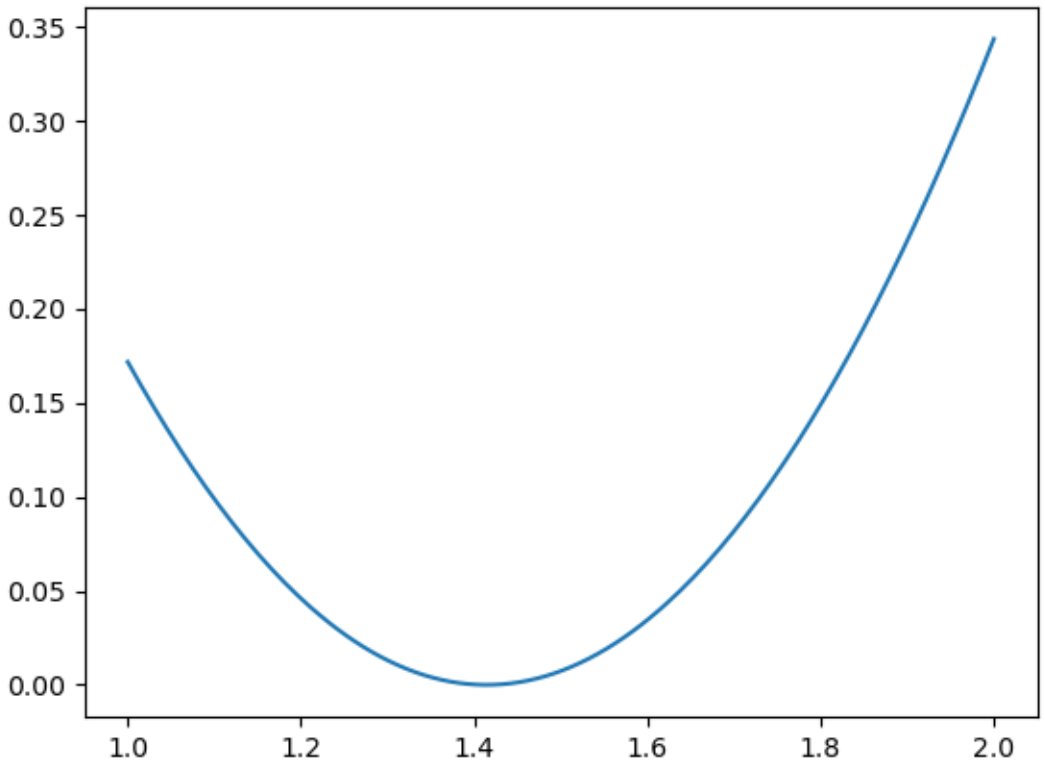
October 13, 2023

1 TP 2

1.0.1 Exercice 1:

[11]:
```
import numpy as np
import matplotlib.pyplot as plt

def f(x) :
    return (x - np.sqrt(2))**2

x=np.linspace(1,2,200)
plt.plot(x,f(x))
```

[11]: [<matplotlib.lines.Line2D at 0x7f3afe2a4490>]



1.0.2 Question 1:

Passer de [a,b] à [a,y] revient à avoir $longueur(I) = y - a = (b - a)/\tau$

Passer de [a,b] à [x,b] revient à avoir $longueur(I) = b_x = $

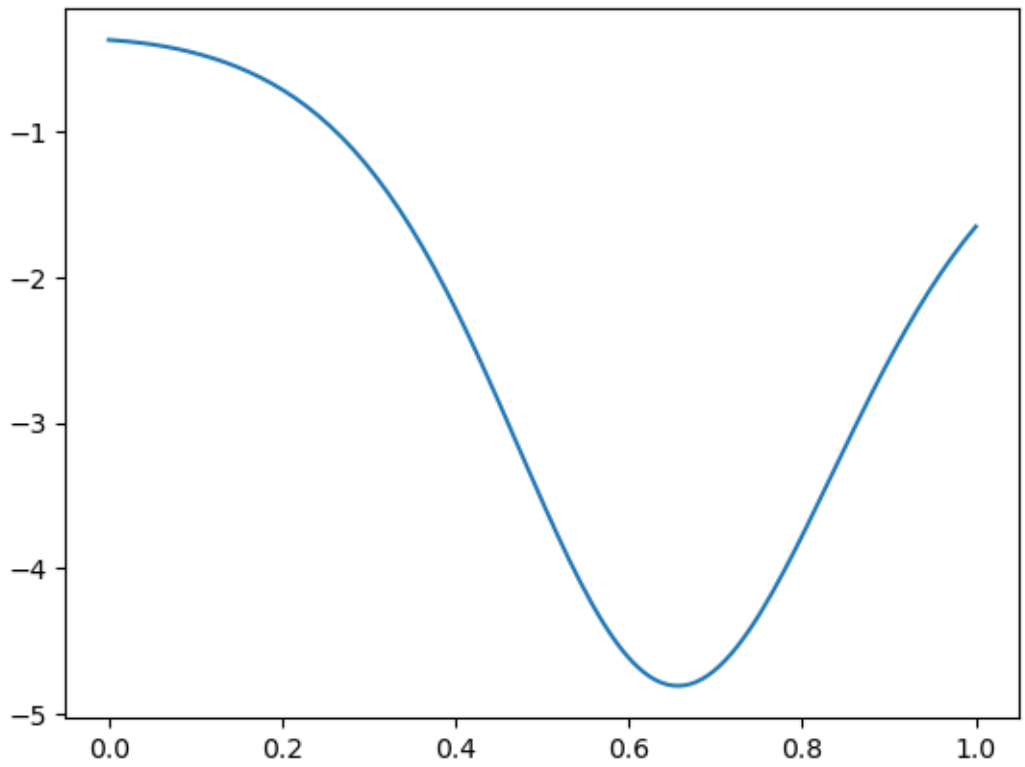
1.0.3 Question 2:

[17]:
```
def section(f, a0, b0, tol):
    a = a0
    b= b0
    count=0
    t = (1+5**0.5)/2
    while b - a > tol :
        x = a + (b-a)/t**2
        y = a + (b-a)/t
        if f(x)< f(y):
            a, b = a, y
        else:
            a, b = x, b
        count +=1
    return (a+b)/2, count

section(f,1,2,0.003)
```

[17]: (1.4142619343463463, 13)

[22]:
```
def f2(x):
    return - np.exp(np.arctan(x)- np.cos((5*x)))

X = np.linspace(0,1,150)
plt.plot(X, f2(X))

section(f2, 0,1,0.003)
```

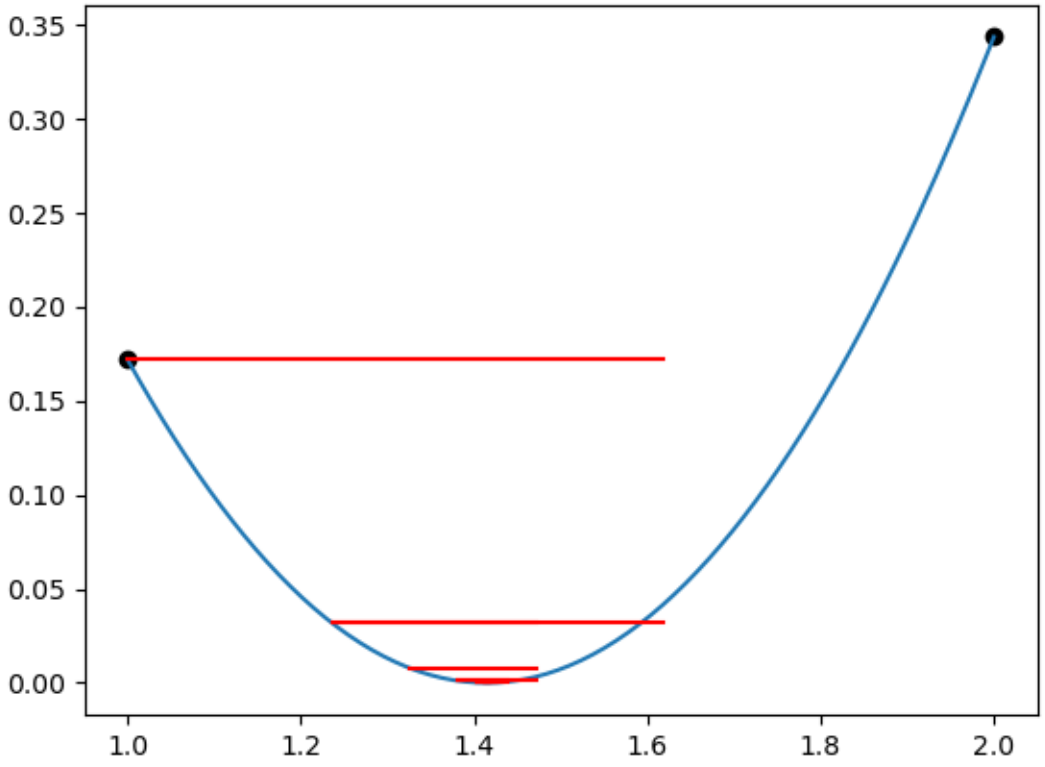
[22]: (0.6565411518764196, 13)



[41]:
```
#Visualisation de l'algorithme
def section2(f, a0, b0, tol):
    a = a0
    b = b0
    count = 0
    t =(1+5**0.5)/2

    x = np.linspace(a,b,150)
    plt.plot(x,f(x))
    plt.scatter([a,b],[f(a),f(b)], c='black')

    while (b-a)>tol:
        x=a+(b-a)/t**2
        y=a+(b-a)/t
        if f(x)<f(y):
            a,b = a,y
        else:
            a,b = x,b
        count+=1

        #plt.scatter([a,b],[f(a),f(b)],c='r')
        plt.plot([a,b],[f(a),f(a)],c='r')
```

```
    return((a+b)/2,count)

section2(f, 1, 2, 0.001)
```

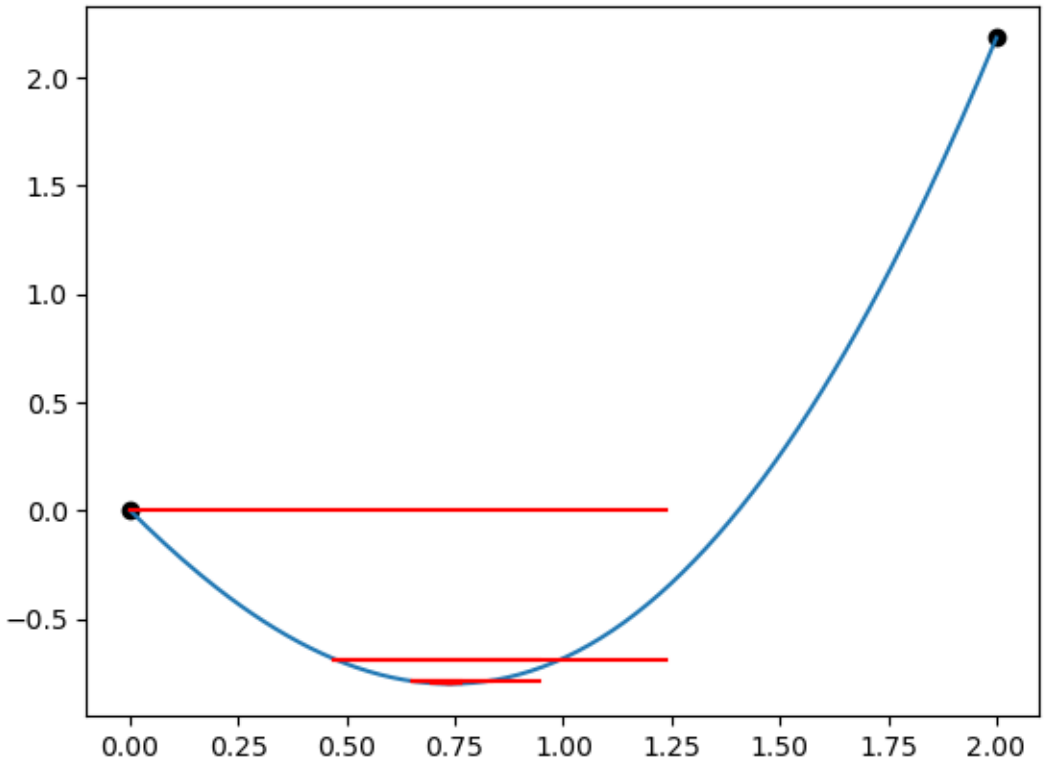
[41]: (1.4141219175553097, 15)



[ ]:
```
#Comparaison avec la fonction "minimize_scalar" de scipy
from scipy.optimize import minimize_scalar
minimize_scalar(f, method='bounded', bounds=(0,1))
```

[47]:
```
# Autre exemple
def f3(x):
    return x**2 - 2 * np.sin(x)

section2(f3, 0,2,0.001)
```

[47]: (0.738980099068715, 16)

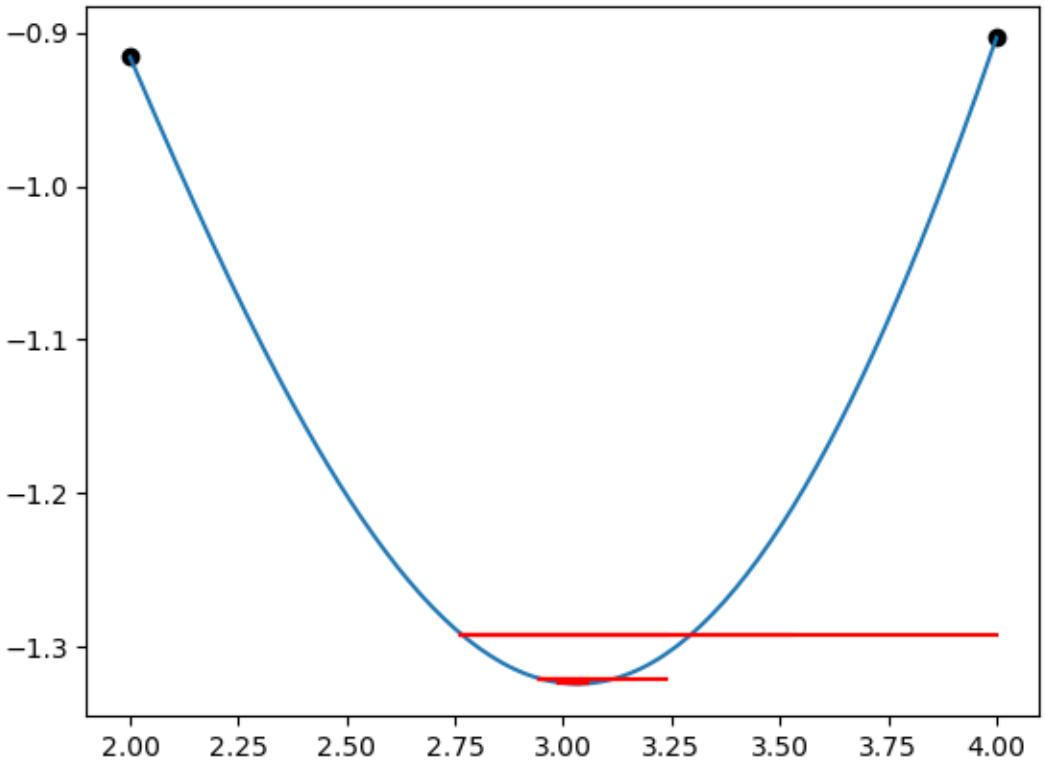


[48]:
```
#Comparaison avec la fonction "minimize_scalar" de scipy
from scipy.optimize import minimize_scalar
minimize_scalar(f3, method='bounded', bounds=(0,1))
```

[48]:
```
 message: Solution found.
 success: True
  status: 0
     fun: -0.8009772242267538
       x: 0.7390850838167129
     nit: 8
    nfev: 8
```

[49]:
```
def f4(x):
    return - 1/x + np.cos(x)

section2(f4, 2,4,0.001)
```

[49]:
```
(3.0328025086052057, 16)
```

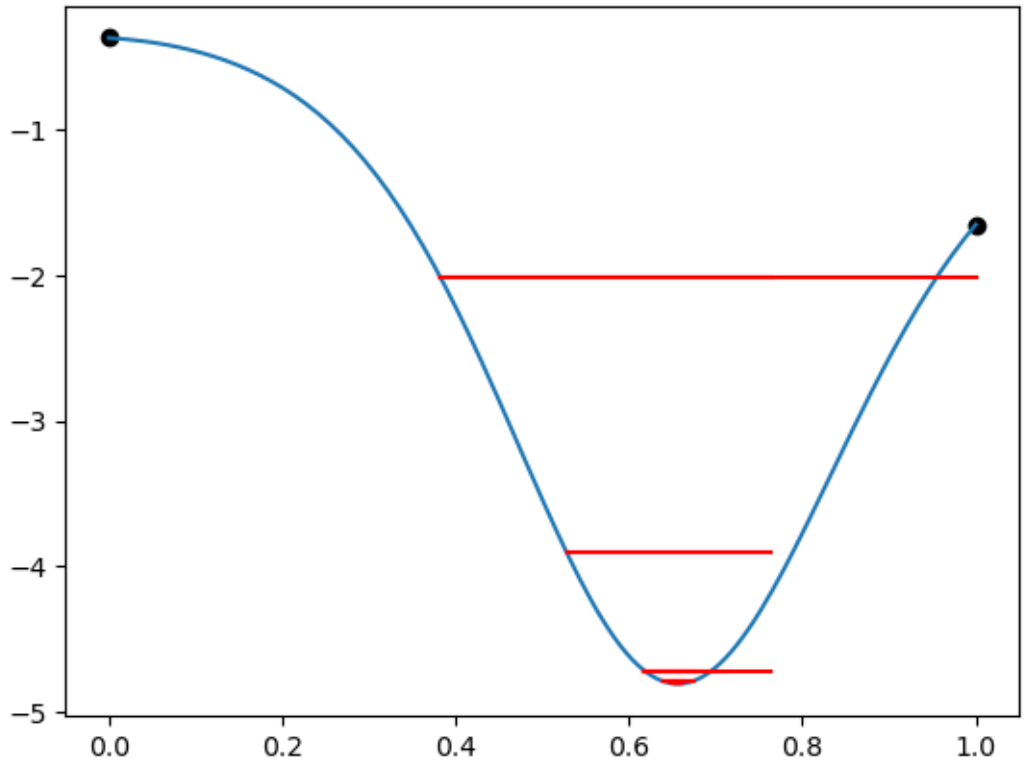


[50]:
```
#Comparaison avec la fonction "minimize_scalar" de scipy
from scipy.optimize import minimize_scalar
minimize_scalar(f4, method='bounded', bounds=(0,1))
```

[50]:
```
 message: Solution found.
 success: True
  status: 0
     fun: -167760.00000596052
       x: 5.9608609865491405e-06
    nit: 25
   nfev: 25
```

[51]:
```
section2(f2, 0,1,0.001)
```

[51]:
```
(0.6564011350853833, 15)
```



[52]:
```
#Comparaison avec la fonction "minimize_scalar" de scipy
from scipy.optimize import minimize_scalar
minimize_scalar(f4, method='bounded', bounds=(0,1))
```

[52]:
```
 message: Solution found.
 success: True
  status: 0
     fun: -167760.00000596052
       x: 5.9608609865491405e-06
    nit: 25
   nfev: 25
```

1.0.4 Exercice 2:

[ ]:
```

```

[ ]:
```

```