Compte rendu du TP 1 MATH 4

par HARTMANN Matthias et BIENVENU Victor groupe 2A

Sommaire

Partie 1 :

1)

Les formules f_1, f_2 et f_3 en python :

```python
def f_1(x : float)-> float:
    """!
    @brief Formule de la fonction 1

    Paramètres :
        @param x : float => la variable x
    Retour de la fonction :
        @return float => l'image de x par la fonction 1

    """
    return x**2 + 1

def f_2(x : float) -> float :
    """!
    @brief Formule de la fonction 2

    Paramètres :
        @param x : float => la variable x
    Retour de la fonction :
        @return float => l'image de x par la fonction 2
    """
    return 2 / ( x**2 + 3)

def f_3(x : float) -> float:
    """!
    @brief Formule de la Fonction 3

    Paramètres :
        @param x : float => la variable x
    Retour de la fonction :
        @return float => l'image de x par la fonction 3

    """
    return 5 / ((x + 2) **2 )
```

La fonction « somme_rectangles » :

```python
def somme_rectangle(f, n : int) -> float:
    """!
    @brief Première approximation de l'air sous la courbe d'une fonction

    Paramètres :
        @param f => Une fonction f
        @param n : int => le nombre de rectangles
    Retour de la fonction :
        @return float => l'approximation de l'air sous la courbe d'une fonction

    """
    return sum([f(i) * 1 for i in range(n)])
```

Execution du code :

```python
if __name__ == "__main__":
    #Partie 1 : 1
    print(somme_rectangle(f_1, 4))
    print(somme_rectangle(f_2, 4))
```

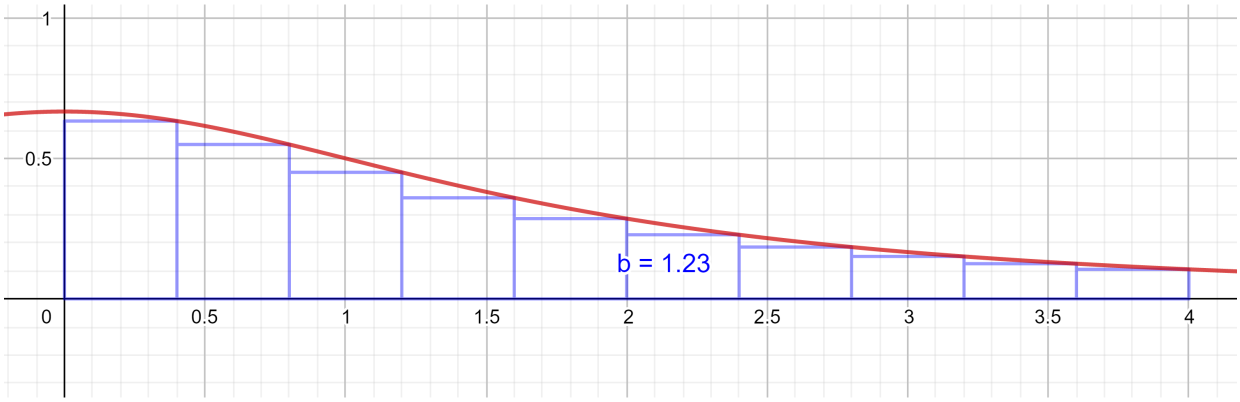
Le résultat semble cohérent avec le résultat obtenu en le calculant à la main. Le programme retourne 18 et le calcule à la main retourne 18 aussi.

S’il on teste cette même fonction avec une fonction décroissante, nous pourrons observer que les rectangles suivent la courbe.

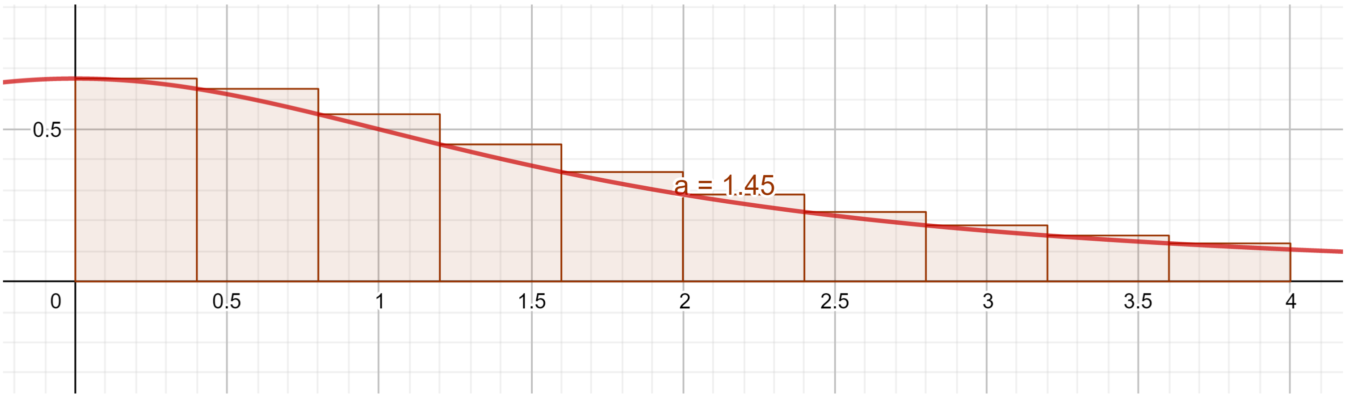
2)

A) les schéma de l'intégrale de la fonction 2

Par défaut :



Par Excès :



B) La largeur des rectangles en fonctions de b et n

Pour obtenir la largeur, nous devons faire la distance entre 0 et b puis la diviser par le nombre de rectangles. La distance entre 0 et b est équivalente à la valeur de b donc cela donne :

$$largeur = \frac{b}{n}$$

C) La hauteur du k^{ième} rectangle

Pour obtenir la hauteur du k^{ième} rectangle il faut faire le calcul suivant :

$$hauteur(k) = fonction(k)$$

D) La fonction python associée

```python
def methode_rectangle(f,b,n) -> float:
    """!
    @brief Deuxième approximation de l'air sous la courbe d'une fonction

    Paramètres :
        @param f => Une fonction f
        @param b => La borne haute de l'intégrale
        @param n => Le nombre de rectangles
    Retour de la fonction :
        @return float => l'approximation de l'air sous la courbe d'une fonction

    """

    largeur : float = b/n
    somme : float = 0
    iterator : float = 0
    for i in range(n):
        somme += f(iterator)*largeur
        iterator += largeur
    return somme
```

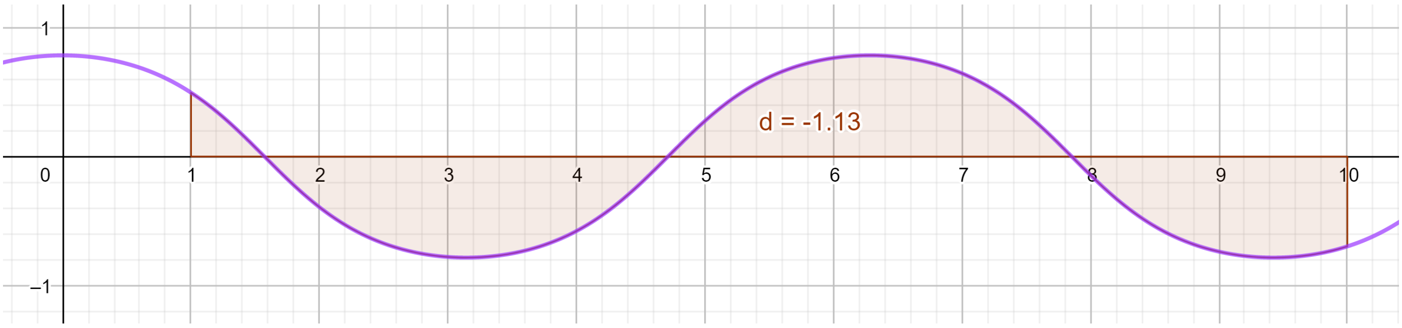
E) Le tableau de valeurs

	$\int_0^4 (x^2+1)\,dx$	$J_1=\int_0^5 (\frac{2}{x^2+3})\,dx$	$J_2=\int_0^{20} (\frac{5}{(x+2)^2})\,dx$
n = 4	18	1,80	7.02
n = 10	22.24	1.58	3.87
n = 100	25.01	1.44	2.40
n = 1000	25.30	1.43	2.28
Valeur réelle	25,33	1,43	2,27

On peut conclure que le nombre de rectangles influe la précision de la valeur d'aire. Plus il y a de rectangles, plus l'aire est précise.

3)

A)



B) La largeur des rectangles en fonctions de a, b et n

Pour obtenir la largeur, nous devons faire la distance entre a et b puis la diviser par le nombre de rectangles:

$$largeur = \frac{b-a}{n}$$

C) Exprimer a_k en fonction de a, b, n et k :

$$a_k = \frac{b-a}{n} \cdot k$$

D) En déduire l'aire A_k du $k^{ième}$ rectangle :

On en déduit la formule suivante :

$$A_k = \frac{(b-a)}{n} \cdot f\left(\frac{(b-a)}{n} \cdot k\right)$$

E) la fonction python

```python
def methode_rectangle_v2(f,a,b,n) -> float:
    """!
    @brief Généralisation de l'approximation de l'air sous la courbe d'une fonction

    Paramètres :
        @param f => Une fonction f
        @param a => la borne basse de l'intégrale
        @param b => La borne haute de l'intégrale
        @param n => Le nombre de rectangles
    Retour de la fonction :
        @return float => l'approximation de l'air sous la courbe d'une fonction

    """

    largeur : float = (b-a)/n
    somme : float = 0
    iterator : float = 0
    while iterator < b:
        somme += f(iterator)*largeur
        iterator += largeur
    return somme
```

F) Le tableau de valeur

	$J_3 = \int_1^{10} \arctan(\cos(x))\,dx$
n = 4	0.9739302067788755
	-0.5317241832872797
	-1.1377426607432766
	-1.1238968609042392
n = 10	-0.5317241832872797
n = 100	-1.1377426607432766
n = 1000	-1.1238968609042392
Valeur réelle	-1,13