Evaluation en Calcul Scientifique 3A9 corrig?

December 23, 2019

Ecole Supérieure PRivée d'Ingénierie et de Technologies
Durée de l'évaluation : 30 minutes
NOM et Prénom :

Question 1 : Importer les modules `numpy`, `matplotlib.pyplot` et `sympy` en utilisant des alias différents.

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import sympy as sp
```

Question 2 : Créer une liste `t` composée de `1000` points répartis uniformément sur $[-2\pi, 2\pi]$.

[2]:
```
t=np.linspace(-2*np.pi,2*np.pi,10**3)
```

Question 3 : On considère la fonction f définie par :

$$f(t) = \frac{1}{2}\big(\sin(t)^2 + \cos(t)\big),\ \forall t \in [-2\pi, 2\pi]$$

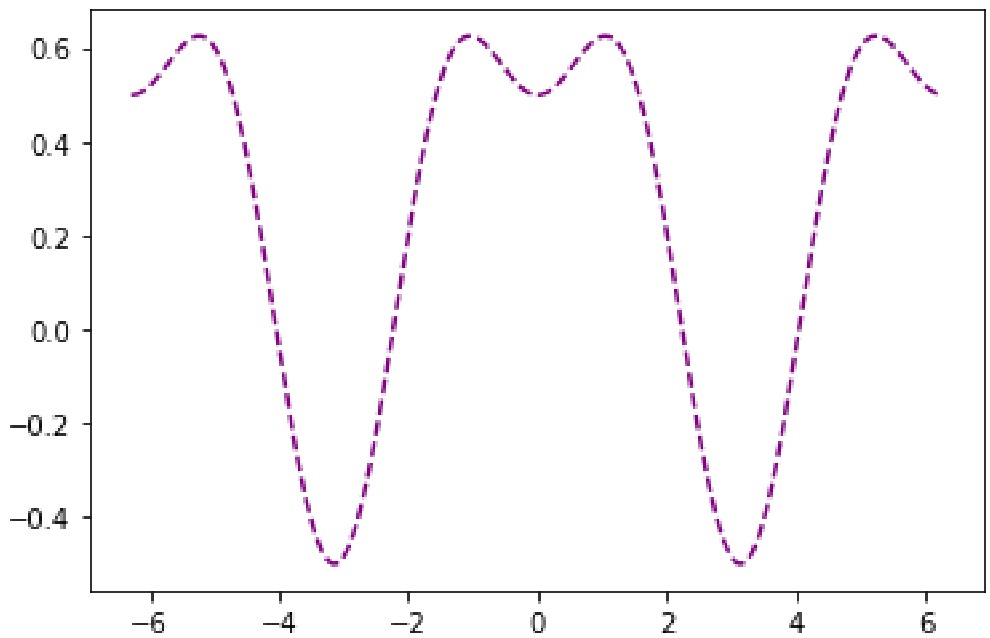
Programmer la fonction f.

[3]:
```
f=lambda t : (np.sin(t)**2+np.cos(t))/2
```

Question 4 : Donner l'instruction de la représentation graphique de la fonction f sur $[-2\pi, 2\pi]$. Utiliser la couleur `purple` et une ligne discontinue pour la courbe.

[4]:
```
plt.plot(t,f(t),color="purple",linestyle="dashed")
```

[4]:
```
[<matplotlib.lines.Line2D at 0x24841d5f080>]
```



Question 5 : Observer la courbe représentative de f et indiquer le nombre de racines de f sur $[-2\pi, 2\pi]$. Justifier votre réponse.

[5]:
```
# 4 racines : 4 passages par zéro
```

Question 6 : Soit

$$I(f) = \int_{-2\pi}^{2\pi} f(x)dx$$

Donner la valeur de $I(f)$, qu'on notera I, en utilisant la fonction integrate du module sympy. Donner les instructions nécessaires.

[6]:
```
x=sp.Symbol('x')
I=sp.integrate((sp.sin(x)**2+sp.cos(x))/2,(x,-2*np.pi,2*np.pi))
I
```

[6]: 3.14159265358979

Question 7 : On souhaite maintenant approcher $I(f)$ par $I^c_{Rm}(f)$, la valeur donnée par la méthode d'intégration composite du rectangle du milieu dont la formule est rappelée ci-dessous :

$$I^c_{Rm}(f) = 2h \sum_{i=0}^{p-1} f(x_{2i+1})$$

avec h désigne le pas de discrétisation de l'intervalle $[a, b]$ sur lequel f est définie, $n = 2p$ le nombre de sous intervalles et x_i, $i \in \{0, \cdots, n\}$, les points d'intégration.

Ecrire une fonction RM(f,a,b,n) qui renvoie la valeur de $I^c_{Rm}(f)$.

[7]:
```
def RM(f,a,b,n):
    h=(b-a)/n
    p=n/2
    S=0
    for i in np.arange(0,p):
        S+=f(a+(2*i+1)*h)
    return 2*h*S
```

Question 8 : Pour $n = 10$, utiliser la fonction RM pour approcher $I(f)$ de la question 6. Comparer le résultat avec la valeur exacte en terme d'erreur d'intégration.

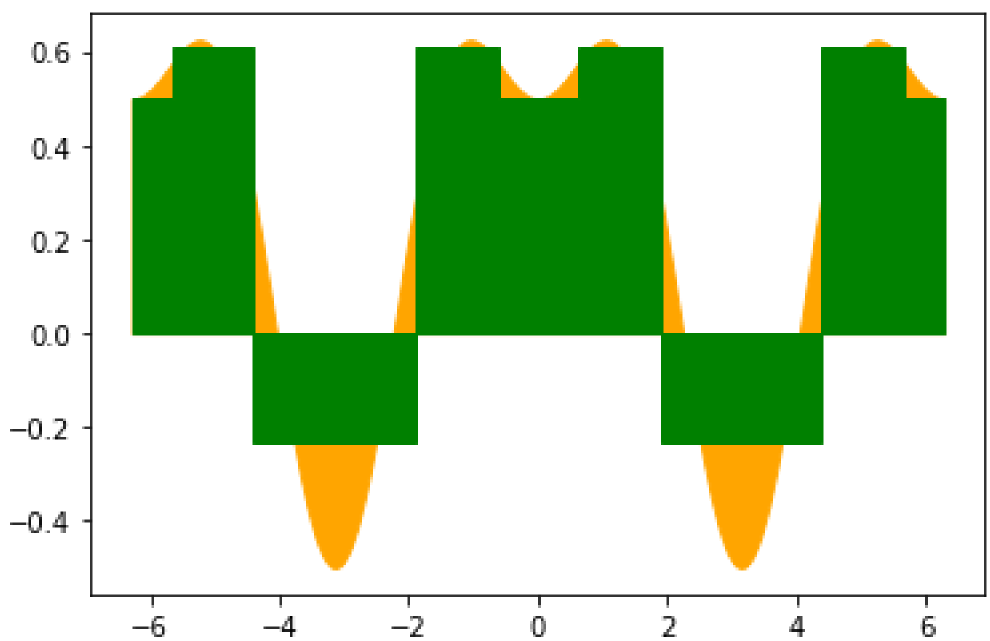
[8]:
```
RM(f,-2*np.pi,2*np.pi,10)
# Erreur d'intégration
np.abs(I-RM(f,-2*np.pi,2*np.pi,10))
```

[8]: $8.88178419700125 \cdot 10^{-16}$

Question 9 : Sur une même figure, et pour $n = 10$, représenter en orange la valeur exacte $I(f)$ et en vert la valeur approchée $I^c_{Rm}(f)$. Ecrire les instructions nécessaires.

[9]:
```
plt.fill_between(t,f(t),color="orange")
plt.fill_between(np.linspace(-2*np.pi,2*np.pi,11),f(np.linspace(-2*np.pi,2*np.
 ↪pi,11)),step="mid",color="green")
```

[9]: <matplotlib.collections.PolyCollection at 0x24841e29668>



Question 10 : Refaire la question 9 pour $n = 100$. Observer le graphique, que remarquez vous?

[10]:
```
plt.fill_between(t,f(t),color="orange")
plt.fill_between(np.linspace(-2*np.pi,2*np.pi,101),f(np.linspace(-2*np.pi,2*np.
 ↪pi,101)),step="mid",color="green")
# L'approximation a bien été améliorée. Les aires (en orange et vert) sont␣
 ↪presque confondues.
```

[10]:
```
<matplotlib.collections.PolyCollection at 0x24841e8fcc0>
```