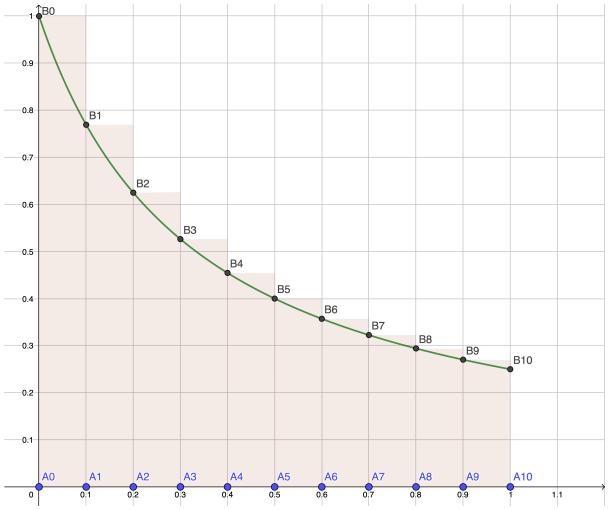
TP13 : éléments de correction

Préliminaires

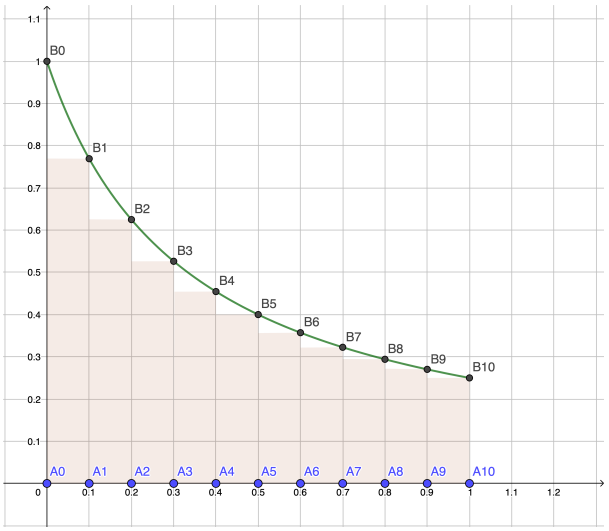
1. $I = \displaystyle\int_0^1 \frac{1}{3t+1}\,\mathrm{d}t = \left[\frac{\ln(3t+1)}{3}\right]_0^1 = \boxed{\frac{\ln(4)}{3} \simeq 0,4621}$

2. Calculer $J = \displaystyle\int_0^2 t^2+1\,\mathrm{d}t = \left[\frac{t^3}{3}+t\right]_0^2 = \frac{8}{3}+2 = \boxed{\frac{14}{3} \simeq 4,667}$

3. On obtient le tracé suivant :



4. On obtient le tracé suivant :



5. La fonction $\boxed{f_1 \text{ étant décroissante}}$ sur $[0,1]$, pour tout $k \in [\![0, n-1]\!]$ et pour tout $x \in \left[\frac{k}{n}, \frac{k+1}{n}\right]$:

$$f\left(\frac{k+1}{n}\right) \leq f(x) \leq f\left(\frac{k}{n}\right)$$

Par croissance de l'intégrale :

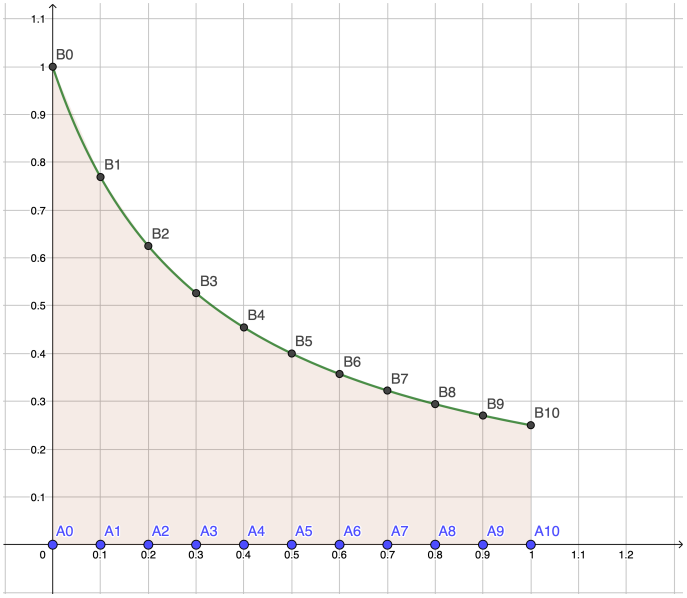
$$\frac{1}{n} f\left(\frac{k+1}{n}\right) \leq \int_{\frac{k}{n}}^{\frac{k+1}{n}} f(x)\,\mathrm{d}x \leq \frac{1}{n} f\left(\frac{k}{n}\right)$$

puis d'après la relation de Chasles et en sommant l'inégalité précédente pour $k \in [\![0, n-1]\!]$:

$$\underbrace{\frac{1}{n}\sum_{k=0}^{n-1} f\left(\frac{k+1}{n}\right)}_{=D_n} \leq \int_0^1 f(x)\,\mathrm{d}x \leq \underbrace{\frac{1}{n}\sum_{k=0}^{n-1} f\left(\frac{k}{n}\right)}_{=G_n}$$

D'où $\boxed{D_n \leq I \leq G_n}$.

6. On obtient le tracé suivant :



7. On peut obtenir une valeur approchée de $I = \int_0^1 f_1(t)\,\mathrm{d}t$ en sommant les aires des 10 trapèzes tracés. Graphiquement, remarquons que cette approximation semble de meilleure qualité qu'avec la méthode des rectangles.

Mise en place

1.
```
def f1(t) :
    return 1/(3*t+1)

def f2(t) :
    return t**2+1
```

Méthode de Monte-Carlo (approche probabiliste)

2. La probabilité que la flèchette atteigne la zone en-dessous de la courbe est égale à l'aire sous la courbe (qui vaut I) divisée par l'aire totale de la zone de tir (qui vaut 1) : donc $\boxed{\text{cette probabilité vaut } I = \int_0^1 f_1(t)\,\mathrm{d}t}$.

3. La proportion de fléchettes atteignant la zone sous la courbe est proportionnelle à l'aire de celle-ci donc

$$\boxed{I \simeq \frac{N_{dessous}}{N}}$$

4. Il s'agit de choisir une abscisse aléatoirement entre 0 et 1 puis une ordonnée aléatoirement entre 0 et 1. On peut écrire :

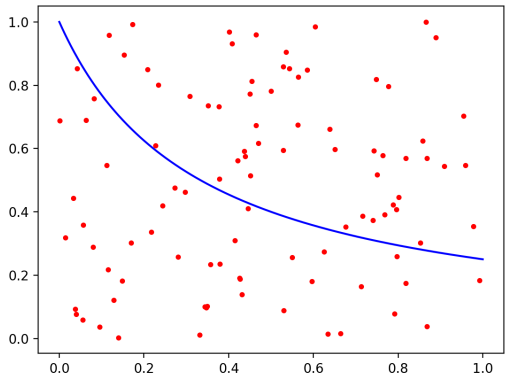
`x,y = rd.random(),rd.random()`

5.

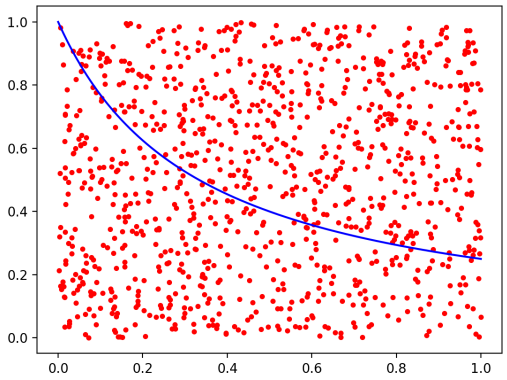
```
def monteCarloVisuel(f,N):
    listeX = []
    listeY = []
    # simulation des N tirs de flechettes
    for i in range(N):
        x,y = rd.random(),rd.random()
        listeX.append(x)
        listeY.append(y)
    # affichage des points
    plt.plot(listeX,listeY,"r.")
    # affichage de la courbe représentant f sur [0;1]
    abscisses = np.linspace(0,1,500)
    ordonnees = [f(x) for x in abscisses]
    plt.plot(abscisses, ordonnees, 'b')
    plt.show()
```

6. On obtient les résultats suivants :

```
>>> monteCarloVisuel(f1,100)
```




```
>>> monteCarloVisuel(f1,1000)
```

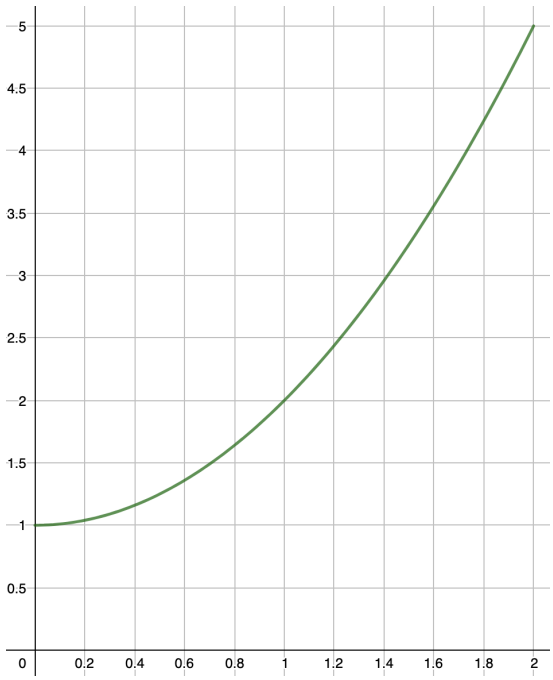


7.

```
def monteCarlo(f,N):
    Ndessous = 0
    # simulation des N tirs de flechettes
    for i in range(N):
        x,y = rd.random(),rd.random()
        # on compte les points en-dessous de la courbe representant f
        if y < f(x):
            Ndessous = Ndessous + 1
    I = Ndessous / N
    return I
```

8. On obtient :
   - `monteCarlo(f1,100)` $= 0,49$
   - `monteCarlo(f1,1000)` $= 0,433$

9. La courbe représentant la fonction f_2 a l'allure suivante :



On peut choisir la zone de tir comme suit :

$$a = 0, \quad b = 2, \quad M = 5$$

10. La proportion de fléchettes atteignant la zone sous la courbe est proportionnelle à l'aire de celle-ci donc

$$\boxed{\int_a^b f(t)\,\mathrm{d}t \simeq \frac{N_{dessous}}{N} \times \underbrace{(b-a)\times M}_{\text{aire de la zone de tir}}}$$

Par exemple,

$$\boxed{J \simeq \frac{N_{dessous}}{N} \times 10}$$

11. On peut écrire :

```
x,y = a+(b-a)*rd.random(),M*rd.random()
```

12.

```
def monteCarlo2(a,b,M,f,N):
    Ndessous = 0
    # simulation des N tirs de flechettes
    for i in range(N):
        x,y = a+(b-a)*rd.random(),M*rd.random()
        # on compte les points en-dessous de la courbe representant f
        if y < f(x):
            Ndessous = Ndessous + 1
    I = (Ndessous / N)*(b-a)*M
    return I
```

13. On obtient :
    - `monteCarlo2(0,2,5,f2,100)` $= 5,0$
    - `monteCarlo2(0,2,5,f2,1000)` $= 4,54$

Méthode des rectangles

14.
```
def valApprocheeRect(f,n):
    S1, S2 = 0 , 0
    for i in range(n):
        S1 = S1 + f(i/n)
    for i in range(1,n+1):
        S2 = S2 + f(i/n)
    Rg, Rd = S1/n, S2/n
    return Rg, Rd
```

15. On obtient

```
>>> valApprocheeRect(f1,1000)
(0.46247335474807333, 0.4617233547480733)
```

Donc $\boxed{0,4617 \leq I \leq 0,4625}$.

16.
```
def valApprocheeRect2(a,b,f,n):
    S1, S2 = 0 , 0
    for i in range(n):
        S1 = S1 + f(a+ i*(b-a)/n)
    for i in range(1,n+1):
        S2 = S2 + f(a+ i*(b-a)/n)
    Rg, Rd = S1*(b-a)/n, S2*(b-a)/n
    return Rg, Rd
```

17. On obtient

```
>>> valApprocheeRect2(0,2,f2,1000)
(4.662667999999996, 4.670667999999996)
```

Donc $\boxed{4,6626 \leq J \leq 0,4707}$.

Méthode des trapèzes

18.
```
def valApprocheeTrap(f,n):
    S = 0
    for i in range(n):
        S += f(i/n) + f((i+1)/n)
    return S/(2*n)
```

19. On obtient

```
>>> valApprocheeTrap(f1,1000)
0.4620983547480732
```

20.
```
def valApprocheeTrap2(a,b,f,n):
    S = 0
    for i in range(n):
        S += f(a+ i*(b-a)/n) + f(a+ (i+1)*(b-a)/n)
    return S*(b-a)/(2*n)
```

21. On obtient

```
>>> valApprocheeTrap2(0,2,f2,1000)
4.666668000000002
```

Bilan

22. On a pour la valeur exacte :

```
>>> np.log(4)/3
0.46209812037329684
```

	Valeur approchée par la méthode des rectangles à gauche	Valeur approchée par la méthode des rectangles à droite	Valeur approchée par la méthode des trapèzes	Valeur approchée par la méthode de Monte Carlo
$n = 100$	0.465871555633049	0.458371555633049	0.4621215556330487	0.56
$n = 1000$	0.46247335474807333	0.4617233547480733	0.4620983547480732	0.465
$n = 10000$	0.462135622717060	0.4620606227170602	0.4620981227170616	0.4649

23. On définit

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

	Valeur approchée par la méthode des rectangles à gauche	Valeur approchée par la méthode des rectangles à droite	Valeur approchée par la méthode des trapèzes	Valeur approchée par la méthode de Monte Carlo
$n = 100$	0.7499786042621127	0.7436573986738272	0.7468180014679695	0.76
$n = 1000$	0.747140131778599	0.7465080112197703	0.7468240714991843	0.735
$n = 10000$	0.7468557382272644	0.7467925261713813	0.7468241321993262	0.7401