Correction - TP 2

Révisions : Les librairies de Python

Exercice 1

On peut utiliser les commandes suivantes :

```
>>> A = 3*np.ones((3,3))+2*np.eye(3,3) ; A
>>> B = np.eye(4,4) ; B[:,0:1] = np.ones((4,1)) ; B
```

Exercice 2

On a :

$$\begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} \times \begin{pmatrix} 1 & \dots & 1 \end{pmatrix} = \begin{pmatrix} a_1 & a_1 & \dots & a_1 \\ a_2 & a_2 & \dots & a_2 \\ \vdots & \vdots & & \vdots \\ a_{10} & a_{10} & \dots & a_{10} \end{pmatrix}.$$

On en déduit les commandes suivantes :

```
>>> C = np.transpose([np.arange(1,11)]) ; L = np.ones(1,10) ; np.dot(C,L)
```

Exercice 3

1. (a) On fait des opérations pointées (coefficients par coefficients).

   ```
   >>> x = np.arange(1.,11.)**(-2)
   ```

   (b) On applique la fonction `sum` :

   ```
   >>> x = np.arange(1.,11.)**(-2) ; np.sum(x)
   ```

2. Sur le modèle de la question précédente :

   ```
   >>> x = np.sum((1/2)**np.arange(0,11))
   ```

3. (a) `x` est le vecteur de taille 10 contenant que des 1. `y` est un vecteur de taille 10 dont la i-ème composante contient :

   $$y_i = \sum_{j=1}^{i} x_i = \sum_{j=1}^{i} 1 = i.$$

   (b) `z` contient le réel :

   $$\sum_{i=1}^{10} i = \frac{10 \times 11}{2} = 55.$$

   (c) `t` contient cette fois le réel :

   $$\sum_{i=1}^{10} \sum_{j=1}^{i} j = \sum_{i=1}^{10} \frac{i(i+1)}{2} = \frac{1}{2} \sum_{i=1}^{10} (i + i^2) = \frac{10 \times 11}{4} + \frac{10 \times 11 \times (2 \times 10 + 1)}{12}.$$

Exercice 4

1. On obtient le nombre maximal de véhicules circulant en une journée grâce à la commande `np.max(A)`.

   Pour obtenir le mercredi où il y a eu le plus de circulation, on utilise la commande `np.max(A[:,2])`.

2. Il faut pour cela sommer suivant les lignes de A pour obtenir le nombre de véhicule pour chaque semaine. On cherche ensuite les indices des lignes où le coefficient de C est maximal. On peut par exemple faire :

```
1 S = np.sum(A,1)
2 m = np.max(S)
3 for k in range(52) :
4     if S[k] == m :
5         print(k+1)
```

3. On parcourt toute la matrice A et on compte le nombre de jours où la fréquentation est minimale :

```
m = np.mim(A)
k = 0
for i in range(52) :
    for j in range(7) :
        if A[i,j] == m :
            k = k+1
print(k)
```

4. Nous n'aurons qu'une approximation de cette probabilité, mais on peut compter le nombre de jours où 8 voitures sont passées et diviser ce nombre par 364.

```
k = 0
for i in range(52) :
    for j in range(7) :
        if A[i,j] >= 8 :
            k = k+1
print(k/364)
```

En notant X une variable aléatoire qui suit une loi de Poisson $\mathscr{P}(5)$, on a :

$$P(X \geq 8) = 1 - P(X \leq 7) = 1 - \sum_{k=0}^{7} P(X = k) = 1 - \sum_{k=0}^{7} \frac{5^k}{k!} e^{-5}.$$

Exercice 5

On peut utiliser le programme suivant :

```
def trace(A)
     n,p = np.shape(A)
     if n != p :
          return("A n'est pas carrée")
     else :
          T = 0
          for k in range(n):
               T = T + A[k,k]
          return(T)
```

Exercice 6

1. On peut procéder ainsi :

```
u = ((-1)**np.arange(1,51))/(np.arange(1,51)**2)
```

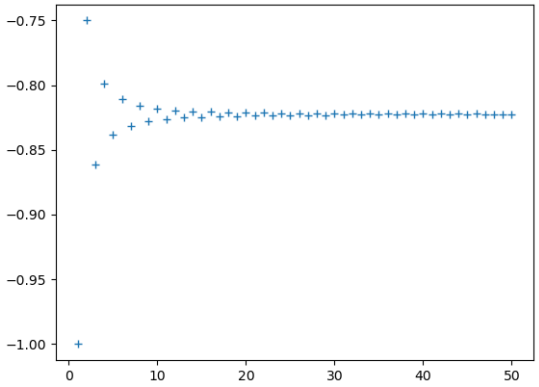
2. On utilise la commande `np.cumsum` :

```
v = np.cumsum(u)
```

3. On utilise pour cela la ligne de commandes suivante :

```
x = np.arange(1,51)
plt.plot(x,v,'x')
plt.show()
```

On obtient le graphe suivant :



Par lecture graphique de la représentation obtenue, on observe que la série converge et que sa somme est environ égale à 0,82. Plus précisément, on observe que les suites extraites paires et impaires de la suite des sommes partielles de la série sont adjacentes et converge vers la somme de la série.

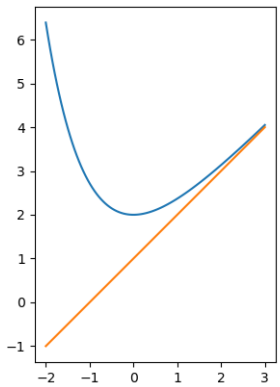
On vérifie facilement à la main que la série converge. En effet, elle est absolument convergente (car la série $\sum \frac{1}{n^2}$ converge en tant que série de Riemann avec $\alpha = 2 > 1$), donc convergente.

Exercice 7

1. $\lim\limits_{x\to+\infty} f(x) - (x+1) = \lim\limits_{x\to+\infty} e^{-x} = 0$ donc $\mathscr{D}$ est bien asymptote oblique à $\mathscr{C}_f$ en $+\infty$
2. Voici une proposition :

```
def f(x) :
    return(x+1+np.exp(-x))

def delta(x) :
    return(x+1)

x = np.linspace(-2,3,100)
plt.plot(x,f(x))
plt.plot(x,delta(x))
plt.axis('scaled')
plt.show()
```

On obtient la représentation graphique suivante :



Exercice 8

1. f est continue sur $\mathbb{R}$ comme composée de fonctions continues. Elle est de plus dérivable pour les mêmes raisons, et on a :

$$\forall x \in \mathbb{R}, \quad f'(x) = \frac{e^x + e^{-x}}{2} > 0.$$

Donc f est strictement monotone sur $\mathbb{R}$, et on a $\lim\limits_{x\to\pm\infty} f(x) = \pm\infty$. Par le théorème de la bijection, f réalise une bijection de $\mathbb{R}$ sur $\mathbb{R}$.

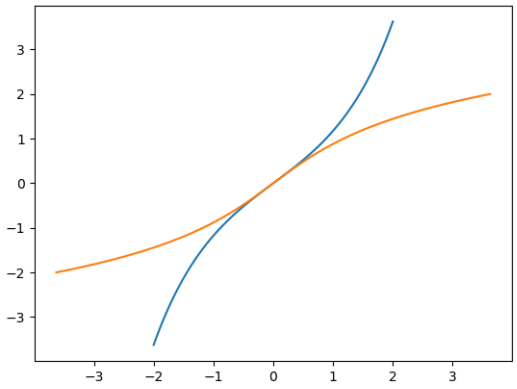
2. On peut utiliser les instructions :

```
x = np.linspace(-2,2,100)

def f(x)
     return (np.exp(x)-np.exp(-x))/2

plt.plot(x,f(x))
```

3. On sait que le graphe de f^{-1} est le symétrique de celui de f par rapport à la droite $y = x$. On utilise ce résultat ici en ajoutant à ce qui précède l'instruction :

```
plt.plot(f(x),x)
plt.show()
```

On obtient la représentation graphique suivante :



Exercice 9

1. (a) f est définie, continue et dérivable sur $\mathbb{R}$ et $f'(x) = -e^{-x} < 0$. Donc f est strictement décroissante sur $\mathbb{R}$.
   On a donc : $f([1,2]) = [f(2), f(1)] = \left[e^{-2}+1, e^{-1}+1\right] \subset [1,2]$. Donc $[1,2]$ est stable par f.

   (b) On pose $g(x) = f(x) - x$. g est définie, continue et dérivable sur $[1,2]$ et $g'(x) = -e^{-x} - 1 < 0$.
   Donc g est continue et strictement décroissante sur l'intervalle $[1,2]$ et réalise donc une bijection de $[1,2]$ dans $[g(2), g(1)] = [e^{-2}-1, e^{-1}]$.
   Comme $0 \in [e^{-2}-1, e^{-1}]$, 0 admet un unique antécédent $\alpha \in [1,2]$ par g. Donc l'équation $f(x) = x$ admet une unique solution $\alpha \in [1,2]$.

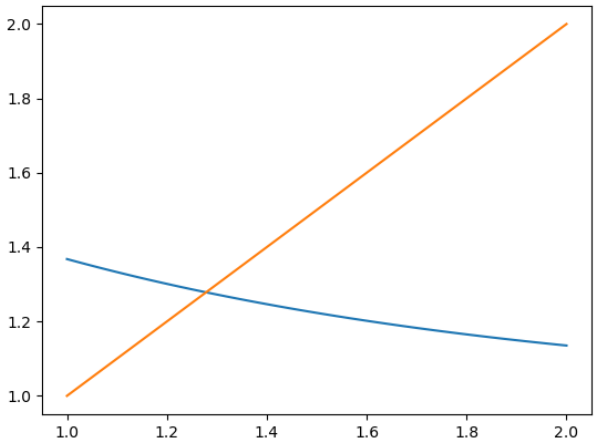
   (c) Voici les instructions pour définir f :

```
def f(x) :
    return(np.exp(-x)+1)
```

   (d) Pour tracer le graphique sur `Python`, on ajoute les instructions suivantes (dans l'éditeur ou la console) :

```
x = np.linspace(1,2,100)
y = f(x)
plt.plot(x,y) //courbe de f
plt.plot(x,x) //droite y=x
plt.show()
```

On obtient la figure suivante :



Pour obtenir une approximation de α, il faut déterminer l'abscisse du point d'intersection de $\mathscr{C}_f$ avec la droite d'équation $y = x$. On trouve $\alpha \simeq 1,28$.

2. (a) Notons $\mathscr{P}(n)$ la propriété : "$u_n \in [1,2]$. Montrons que $\mathscr{P}(n)$ est vraie pour tout $n \in \mathbb{N}$.

Ini. $u_0 = 1 \in [1,2]$ donc $\mathscr{P}(0)$ est vraie.

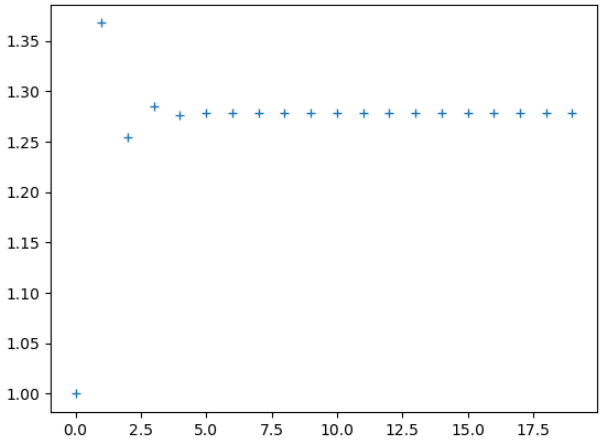
Héré. Soit $n \in \mathbb{N}$. Supposons que $\mathscr{P}(n)$ est vraie et montrons $\mathscr{P}(n+1)$.
Par hypothèse de récurrence, $u_n \in [1,2]$. Donc $u_{n+1} = f(u_n) \in f([1,2]) \subset [1,2]$ d'après la question 1.(a). Donc $\mathscr{P}(n+1)$ est vraie.

Ccl. Par récurrence, $u_n \in [1,2]$ pour tout $n \in \mathbb{N}$.

(b) Voici la fonction `SuiteU` :

```
1 def SuiteU(n) :
2     u=1
3     for k in range(1,n+1) :
4         u = f(u)
5     return(u)
```

(c) On obtient le graphique suivant :



Ce programme permet de tracer les 20 premiers termes de la suite (u_n) :

- `x = np.arange(n)` donne une matrice de taille 1×20 contenant les entiers de 0 à 19.
- `y = np.zeros(n)` donne une matrice de taille 1×20 remplie de 0. On affecte la valeur $u_0 = 1$ au premier coefficient puis, avec la boucle `for`, on met à jour les coefficients d'indice 1 à 19 de la matrice afin qu'elle contienne en sortie de boucle $[u_0, u_1, \ldots, u_{19}]$.
- Enfin, la fonction `plt.plot` permet la représentation graphique des points de coordonnées (k, u_k) pour k variant de 0 à 19. Elle prend en paramètre deux matrices lignes `x` (qui contient les abscisses successives des points à représenter, les éléments k) et `y` (qui contient les ordonnées successives des points à représenter, les éléments u_k). L'option `'+'` permet de faire apparaitre les pour sous forme de croix. Le fait de spécifier la forme de ces points permet aussi qu'ils ne soient pas reliés (fonctionnalités par défaut sous `Python`).

3. (a) f est dérivable sur $[1,2]$ et pour tout $x \in [1,2]$, on a : $|f'(x)| = e^{-x} \leq e^{-1}$. D'après l'inégalité des accroissements finis,
$$\forall a, b \in [1,2], \qquad |f(b) - f(a)| \leq \frac{1}{e}|b-a|.$$
En prenant $b = u_n \in [1,2]$ (d'après la question 2.(a)) et $a = \alpha \in [1,2]$ (d'après la question 1.(a)), on obtient :
$$|f(u_n) - f(\alpha)| \leq \frac{1}{e}|u_n - \alpha| \qquad \Leftrightarrow \qquad |u_{n+1} - \alpha| \leq \frac{1}{e}|u_n - \alpha|.$$

(b) Notons $\mathscr{P}(n)$ la propriété : "$|u_n - \alpha| \leq \dfrac{1}{e^n}$. Montrons que $\mathscr{P}(n)$ est vraie pour tout $n \in \mathbb{N}$.

Ini. $|u_0 - \alpha| = |1 - \alpha| \leq 1 = \dfrac{1}{e^0}$ car $\alpha \in [1,2]$ donc $\mathscr{P}(0)$ est vraie.

Héré. Soit $n \in \mathbb{N}$. Supposons que $\mathscr{P}(n)$ est vraie et montrons $\mathscr{P}(n+1)$.
En utilisant la question précédente puis l'hypothèse de récurrence, on a :
$$|u_{n+1} - \alpha| \leq \frac{1}{e}|u_n - \alpha| \leq \frac{1}{e} \times \frac{1}{e^n} = \frac{1}{e^{n+1}}.$$
Donc $\mathscr{P}(n+1)$ est vraie.

Ccl. Par récurrence, $|u_n - \alpha| \leq \dfrac{1}{e^n}$ pour tout $n \in \mathbb{N}$.

Comme $0 \leq |u_n - \alpha| \leq \dfrac{1}{e^n}$ et $\lim\limits_{n\to+\infty} \dfrac{1}{e^n} = 0$, on a par encadrement $\lim\limits_{n\to+\infty} \leq |u_n - \alpha| = 0$ donc $\lim\limits_{n\to+\infty} u_n = \alpha$.

(c) Il suffit de déterminer n tel que $e^{-n} \leq \varepsilon$. Voici le programme demandé :

```
eps = input('Donner epsilon : ')
n = 0
while exp(-n)>eps :
    n = n+1
print(SuiteU(n))
```

Exercice 10

1. (a) On pose $f_n(x) = x^n + x - 1$. f_n est définie, continue et dérivable sur $\mathbb{R}_+$ et $f_n'(x) = nx^{n-1} + 1 > 0$. Donc f_n est croissante sur $\mathbb{R}_+$.

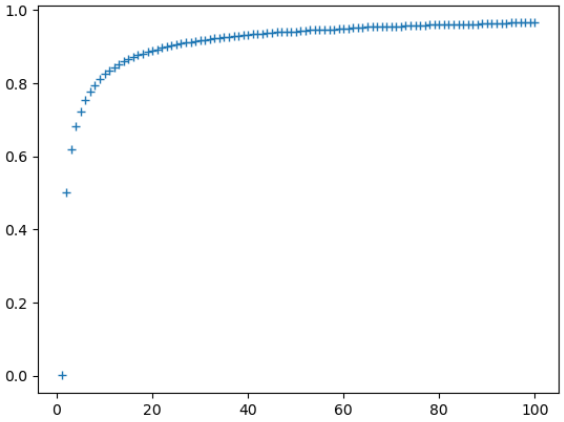
(b) f_n est continue et strictement croissante sur $\mathbb{R}_+$, elle réalise une bijection de $\mathbb{R}_+$ sur $f_n(\mathbb{R}_+) = [-1, +\infty[$.
Comme $0 \in [-1, +\infty[$, 0 admet un unique antécédent $u_n \in \mathbb{R}_+$ par f_n. L'équation (E_n) admet donc une unique solution u_n sur $\mathbb{R}_+$.

(c) $f_n(0) = -1$, $f_n(u_n) = 0$ (par définition de u_n) et $f_n(1) = 1$.
Donc $f_n(0) \leq f_n(u_n) \leq f_n(1)$ et comme f_n est croissante sur $\mathbb{R}_+$, $0 \leq u_n \leq 1$.

2. (a) Voici le programme complété :

```
def valeur_approchee(n) :
    a = 0
    b = 1
    while (b-a) > 10**(-3) :
        c = (a+b)/2
        if (c**n+c-1)>0 :
            b = c
        else
            a = c
    return(c)
```

(b) On obtient le graphique suivant :



Ce programme permet de tracer les 100 premiers termes de la suite (u_n) :

- `x = np.arange(1,n)` donne une matrice de taille 1×100 contenant les entiers de 1 à 100.
- `y = np.zeros(n)` donne une matrice de taille 1×100 remplie de 0. Avec la boucle `for`, on met à jour tous les coefficients de la matrice afin qu'elle contienne en sortie de boucle $[u_1, u_2, \dots, u_{100}]$.
- Enfin, la fonction `plt.plot` permet la représentation graphique des points de coordonnées (k, u_k) pour k variant de 1 à 100. Elle prend en paramètre deux matrices lignes `x` (qui contient les abscisses successives des points à représenter, les éléments k) et `y` (qui contient les ordonnées successives des points à représenter, les éléments u_k). L'option `'+'` permet de faire apparaitre les pour sous forme de croix. Le fait de spécifier la forme de ces points permet aussi qu'ils ne soient pas reliés (fonctionnalités par défaut sous `Python`).

Au vu du graphique obtenu, on peut conjecturer que la suite (u_n) est croissante et qu'elle converge vers une limite $\ell \simeq 1$.

3. (a) Comme $f_{n+1}(x) - f_n(x) = x^{n+1} - x^n = x^n(x-1)$, on a donc que :
   - $f_{n+1}(x) - f_n(x) \leq 0$ pour tout $x \in [0,1]$;
   - $f_{n+1}(x) - f_n(x) \geq 0$ pour tout $x \in [1, +\infty[$.

   (b) En prenant $x = u_n \in [0,1]$, on obtient $f_{n+1}(u_n) - f_n(u_n) \leq 0$ donc $f_{n+1}(u_n) \leq 0$ (par définition de u_n).
   Comme $f_{n+1}(u_{n+1}) = 0$ (par définition de u_{n+1}), $f_{n+1}(u_n) \leq f_{n+1}(u_{n+1})$.
   Or, d'après la question 1.(a), f_{n+1} est croissante sur $\mathbb{R}_+$. Donc $u_n \leq u_{n+1}$ et la suite (u_n) est croissante.

   (c) On a démontré que (u_n) est croissante (question 3.(b)) et majorée par 1 (question 1.(c)). Par le théorème des suites monotones, (u_n) converge vers une limite ℓ.
   Pour tout $n \in \mathbb{N}$, $u_n \in [0,1]$ (toujours question 1.(c)) donc $\ell \in [0,1]$.

   (d) Supposons que $\ell \neq 1$. Alors $\ell \in [0,1[$. Comme (u_n) est croissante, on a pour tout $n \in \mathbb{N}^*$, $0 \leq u_n \leq \ell$.
   Par croissance de $x \mapsto x^n$ sur $\mathbb{R}_+$, $0 \leq u_n^n \leq \ell^n$ pour tout $n \in \mathbb{N}^*$. Comme $\ell \in [0,1[$, $\lim\limits_{n\to+\infty} \ell^n = 0$.
   Donc par encadrement, $\lim\limits_{n\to+\infty} (u_n)^n = 0$.
   Par définition de u_n, $u_n^n + u_n - 1 = 0$ pour tout $n \in \mathbb{N}^*$. Par passage à la limite, on obtient $\ell - 1 = 0$ donc $\ell = 1$ ce qui contredit l'hypothèse de départ où on avait supposé $\ell \neq 1$.
   Donc $\ell = 1$ et $\lim\limits_{n\to+\infty} u_n = 1$.