TP4/5 : Écriture de sommes finies en **Scilab**

Pré-requis : l'objectif des premières séances de TP est de faire le point sur des fonctionnalités importantes de **Scilab** qui ont été vues en première année. Je vous invite à consulter les chapitres de cours correspondants sur ma page : support informatique.
On pourra en particulier se reporter au « CH 7 : Les structures itératives ».

- Dans votre dossier `Info_2a`, créer le dossier `TP_4`.

I. Avant propos

On considère dans ce TP les suites $(u_n)_{n\in\mathbb{N}}$ et (v_n) suivantes :

$$\forall n \in \mathbb{N},\ u_n = \frac{n^2}{3^n} \qquad \text{et} \qquad \begin{cases} v_0 = 1 \\ \forall n \in \mathbb{N},\ v_{n+1} = \dfrac{2\,v_n}{e^{v_n} + e^{-v_n}} \end{cases}$$

Objectif du TP : il s'agit d'explorer les différentes méthodes permettant le calcul des sommes partielles d'ordre n : $S_n = \sum\limits_{k=0}^{n} u_k$ et $T_n = \sum\limits_{k=0}^{n} v_k$.

II. Calcul des sommes partielles d'ordre n

II.1. Calcul de S_n

Il s'agit ici d'illustrer le cas où la suite (u_n) est donnée sous forme explicite.

II.1.a) Méthode itérative

- Écrire une fonction `calculSn` qui :
  - prend en paramètre un entier `n`,
  - renvoie une variable `S`,
  - à l'aide d'une structure itérative, calcule la valeur de $S_{\mathtt{n}}$ et stocke le résultat dans `S`.

- Que vaut S_0 ? S_5 ? S_{10} ? S_{100} ? S_{1000} ?

II.1.b) Utilisation des fonctionnalités Scilab

- Écrire une fonction `premSuiteU` qui :
  - prend en paramètre un entier `n`,
  - renvoie une variable `U`, vecteur contenant initialement $\mathtt{n}+1$ zéros,
  - à l'aide d'une structure itérative, calcule les $\mathtt{n}+1$ premières valeurs de la suite (u_n) et stocke le résultat dans `U`.

- Que réalise l'appel `sum(1:5)` ? Détailler le rôle de la fonction `sum`.

- En déduire un appel permettant de calculer S_{10}.

- Que réalise l'appel `1 ./ (1:5)` ? Et `(4:8) ./ (1:5)` ? Détailler le rôle de l'opérateur `./`

- De même, que réalise l'appel `[1,2; 1,1] ^ 2` ? Et l'appel `[1,2; 1,1] .^ 2` ? Et `(1:5) ^ 2` ?

- Comment peut-on, par simple manipulation matricielle, créer le vecteur `U` des 101 premiers éléments de la suite (u_n) ? On commencera par stocker le vecteur `0:100` dans une variable `N`.

- Comment obtient-on alors S_{100} à l'aide de `U` ? Et comment récupérer S_5 ?

II.2. Calcul de T_n

Il s'agit ici d'illustrer le cas où la suite (u_n) est donnée sous forme récurrente.

II.2.a) Méthode itérative

- Écrire une fonction `calculTn` qui :
  - × prend en paramètre un entier `n`,
  - × renvoie une variable `T`,
  - × à l'aide d'une structure itérative, calcule la valeur de $S_{\mathtt{n}}$ et stocke le résultat dans `T`.

  On pourra utiliser une variable auxiliaire `v` afin de calculer les différents termes de (v_n).

- Que vaut T_0 ? T_{100} ? T_{10000} ?

II.2.b) Utilisation des fonctionnalités Scilab

- Écrire une fonction `premSuiteV` qui :
  - × prend en paramètre un entier `n`,
  - × renvoie une variable `V`, vecteur contenant initialement $\mathtt{n}+1$ zéros,
  - × à l'aide d'une structure itérative, calcule les $\mathtt{n}+1$ premières valeurs de la suite (v_n) et stocke le résultat dans `V`.

- En déduire un appel permettant de calculer T_{10}.

III. Calcul des n premières sommes partielles

III.1. Calcul des n premières sommes partielles de $\sum u_n$

- Soit $n \in \mathbb{N}$. Quel lien y a-t-il entre S_n et S_{n+1} ?

- En tirant profit de l'égalité précédente, écrire une fonction `premSn` qui :
  - × prend en paramètre une variable `n`,
  - × renvoie une variable `tabS`, vecteur contenant initialement $\texttt{n} + 1$ zéros,
  - × à l'aide d'une structure itérative, stocke la valeur de S_i dans la $i^{\text{ème}}$ case de `tabS`.

  On ne devra pas effectuer d'appel à `calculSn` mais on pourra s'inspirer de son code.

- Comme précédemment, on aurait aussi pu tirer parti des fonctionnalités de **Scilab**.
  Que réalise l'appel `cumsum(1:5)` ? Détailler le rôle de la fonction `cumsum`.

- Quel appel, tirant parti des fonctionnalités **Scilab**, permet d'obtenir les 101 premiers éléments de la suite (S_n) ? On utilisera la fonction `cumsum`.

III.2. Calcul des n premières sommes partielles de $\sum v_n$

- Soit $n \in \mathbb{N}$. Quel lien y a-t-il entre T_n et T_{n+1} ?

- En tirant profit de l'égalité précédente, écrire une fonction `premTn` qui :
  - × prend en paramètre une variable `n`,
  - × renvoie une variable `tabT`, vecteur contenant initialement $\mathtt{n}+1$ zéros,
  - × à l'aide d'une structure itérative, stocke la valeur de T_i dans la $i^{\text{ème}}$ case de `tabT`.

  On ne devra pas effectuer d'appel à `calculTn` mais on pourra s'inspirer de son code. On pourra notamment utiliser une variable `v` calculant les valeurs successives des termes de la suite (v_n).

À retenir : on a étudié deux méthodes en **Scilab** pour obtenir les éléments de (S_n).

1) La suite des sommes partielles étant une suite (grande nouvelle), on peut se servir des procédés vus en TP2.

2) On peut aussi préférer créer le vecteur des premiers éléments de la suite (u_n) et utiliser les instructions `sum` ou `cumsum` suivant ce que l'on cherche à obtenir. Encore une fois, il faut se reporter au TP2.

IV. Tracé des sommes partielles de $\sum u_n$

- Compléter le programme suivant afin qu'il permette d'effectuer le tracé des 51 premiers éléments de (S_n). On exécutera ce programme.

```
N = 0:50
U =
tabS =
plot(N, tabS, 'rx')
```

- Quelle conjecture peut-on émettre sur la nature de la série $\sum u_n$?

V. Suite des sommes partielles aux concours

V.1. ECRICOME 2015

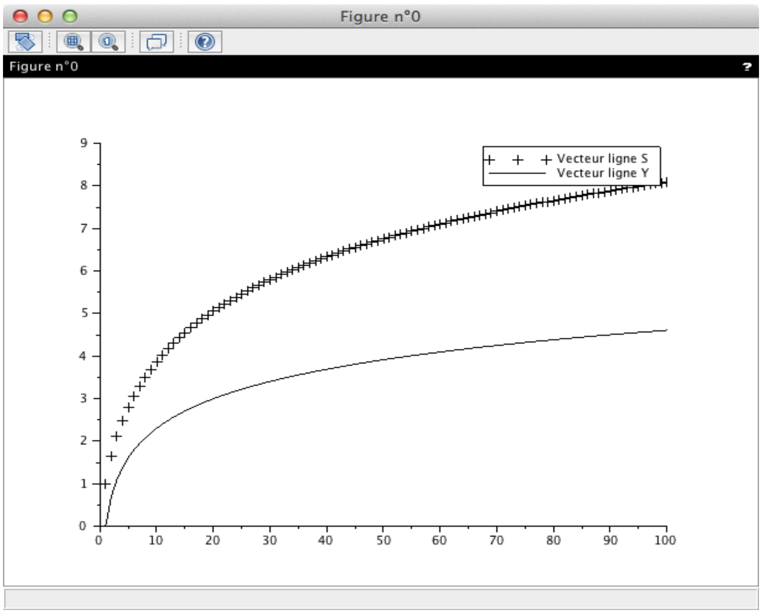
En TP2, on a déjà introduit l'épreuve `ECRICOME 2015` qui commençait par l'étude d'une suite récurrente $(u_n)_{n\in\mathbb{N}^*}$, définie par une relation de récurrence de type $u_{n+1} = F(u_n)$.
La première question, consistait à compléter le programme permettant le calcul des 100 premiers éléments de (u_n). On rappelle ce programme ci-dessous.

```
U = zeros(1,100)
U(1) = 1
for n = 1 : 99
  U(n+1) = 1 - exp(-U(n))
end
plot(U,"+")
```

- *On modifie le programme précédent en remplaçant la dernière ligne par :*

```
X = 1 : 100
S = cumsum(U)
Y = log(X)
plot2d(X,S)
plot2d(X,Y)
```

Le programme ci-dessus permet d'obtenir la représentation graphique suivante :



- *Que représente le vecteur-ligne* `S` *?*
  Quelle conjecture pouvez-vous émettre sur la nature de la série de terme général u_n ?

► *À l'aide de la question **2.h)*** (consistait à démontrer : $\forall n \in \mathbb{N}^*,\ u_n \geqslant \frac{1}{n}$) *établir la nature de la série de terme général u_n.*

V.2. EML 2015

On considère l'application $f : \mathbb{R} \to \mathbb{R}$, $x \mapsto f(x) = x^3\, e^x$
et la suite réelle $(u_n)_{n\in\mathbb{N}}$ définie par : $u_0 = 1$ et $\forall n \in \mathbb{N}, u_{n+1} = f(u_n)$.
Il était demandé de démontrer que la série $\sum\limits_{n\geqslant 1} \dfrac{1}{f(n)}$ converge (de somme S) et que :

$$\forall n \in \mathbb{N}^*,\ \left| S - \sum_{k=1}^{n} \frac{1}{f(k)} \right| \leqslant \frac{1}{(\mathrm{e}-1)\,\mathrm{e}^n}$$

► *En déduire une fonction **Scilab** qui calcule une valeur approchée de S à 10^{-4} près.* (*cf* TP3)

V.3. ECRICOME 2018

Pour tout entier naturel n non nul, on pose : $u_n = \sum_{k=1}^{n} \frac{1}{k} - \ln(n)$.

- *Écrire une fonction d'en-tête :* `function y = u(n)` *qui prend en argument un entier naturel n non nul et qui renvoie la valeur de u_n.*

Commentaire