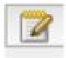
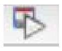
TP 1-Révisions

1 Avant de commencer

1.1 Enregistrez votre travail

1.2 SciNotes

On utilisera systématiquement **SciNotes**, l'éditeur de texte intégré à Scilab. Pour y accéder, une fois Scilab ouvert :

- Placez-vous sur la console, cliquez sur l'icône  .
- Dans « fichier », sélectionnez « Enregistrez-sous » pour sauvegarder ce fichier dans le dossier du TP1 sous le format *.sci* : ce fichier contiendra toutes les fonctions du TP1.
- Dans fichier, sélectionnez « Nouveau » pour créer un nouveau fichier et enregistrez-le dans le dossier du TP1 sous le format *.sce* : ce fichier contiendra tous les scripts du TP1.
- Pour exécuter un script ou une fonction, dans SciNotes vous pouvez, au choix sélectionner « Exécuter fichier sans écho » ou cliquer sur l'icône  qui enregistre et exécute.

1.3 Commentaires

Beaucoup de TP dureront plusieurs séances; il est donc important, en plus de sauvegarder votre travail, que votre code soit « propre » :

- donnez des noms explicites à vos fonctions et vos variables;
- commentez votre code : la commande // permet d'apposer des commentaires (les expressions qui suivent // sont ignorées par Scilab).

Exemple 1

```
// Exemple 1

function y = exemple1(x)    //ceci est un commentaire
    y  = 2x
endfunction
```

1.4 Aide

En cas de doute, vous pouvez consulter l'aide de Scilab grâce à la commande (dans la console) :

```
help fonction
```

où `fonction` est le nom de la commande dont on cherche les fonctionnalités.

2 Vecteurs, matrices

Définir une matrice

- Un **vecteur ligne** est défini par une liste d'éléments entre crochets séparés par des virgules.
- Un **vecteur colonne** est défini par une liste d'éléments entre crochets séparés par des points-virgules.
- Pour définir une matrice de taille $n \times p$ on combine les deux syntaxes.

► Dans la console, taper

```
A1=[1,2,3;4,5,6]
```

et expliquer ce que cette ligne de commande réalise.

► Créer la matrice $B1 = \begin{pmatrix} 1 & -1 & 0 \\ 3 & 2 & -1 \\ 1 & 1 & 1 \end{pmatrix}$ et recopier le code :

► Avec l'aide Scilab, déterminer le rôle des commandes `zeros`, `ones`, `eye`, `linspace` : **ces commandes et leurs arguments sont exigibles !**

Définir une matrice

On peut aussi construire des matrices « par blocs » en concaténant (c'est-à-dire juxtaposant) des vecteurs et/ou matrices, si leur taille le permet, verticalement ou horizontalement.

Par exemple, si A et B sont des matrices avec le même nombre de colonnes, `[A;B]` est la concaténation verticale de A et B (la matrice $\begin{pmatrix} A \\ B \end{pmatrix}$).

On considère les matrices suivantes :

$$C1 = \begin{pmatrix}1\\0\\0\end{pmatrix}, \quad C2 = \begin{pmatrix}0\\1\\0\end{pmatrix}, \quad C3 = \begin{pmatrix}1\\0\\1\end{pmatrix}, \quad D = \begin{pmatrix}1 & 0 & 1\\0 & 1 & 0\\0 & 0 & 1\end{pmatrix}.$$

► Définir les matrices C1, C2 et C3 dans Scilab.

► Construire la matrice D en concaténant C1, C2 et C3 et recopier la ligne de code :

Manipuler des matrices

Soient $A, B \in \mathcal{M}_{n,p}(\mathbb{R})$.

- Extraction et modification des éléments.
  1. L'instruction $A(i, j)$ extrait le coefficient de la $i^{\text{ème}}$ ligne et de la $j^{\text{ème}}$ colonne de A.
     L'instruction $A(i, j) = a$ remplace le coefficient de la $i^{\text{ème}}$ ligne et de la $j^{\text{ème}}$ colonne de A par a.
  2. L'instruction $A(i,:)$ extrait la $i^{\text{ème}}$ ligne de A.
     L'instruction $A(i,:) = L$ remplace la $i^{\text{ème}}$ ligne de A par L.
  3. L'instruction $A(:, j)$ extrait la $j^{\text{ème}}$ colonne de A.
     L'instruction $A(:, j) = C$ remplace la $j^{\text{ème}}$ colonne de A par C.
  4. L'instruction `size(A)` renvoie la taille de la matrice et `rank(A)` son rang (voir le chapitre 5).
- Opérations sur les matrices.

Syntaxe	`A+B`	`A-B`	`A*B`	`inv(A)`	`A'`	`A^k`
Matrice renvoyée	$A+B$	$A-B$	AB	A^{-1}	${}^tA$	A^k

On considère la matrice :

$$A2 = \begin{pmatrix}1 & \frac{2}{3} & \frac{1}{3}\\0 & 1 & 1\\0 & 0 & 1\end{pmatrix}$$

► Compléter la commande suivante afin de créer A2 :

```
A2=[                                  ]
```

► Donner la commande permettant d'obtenir le coefficient de la $2^{\text{ème}}$ ligne et $3^{\text{ème}}$ colonne de A2. Donner la commande permettant d'obtenir la $3^{\text{ème}}$ ligne de A2.

► (D'après EDHEC 2017) Compléter les commandes suivantes pour que soit affichée la matrice $(A2)^n$ pour une valeur n entrée par l'utilisateur :

```
n=input('Entrez une valeur pour n:')
A2=[                                        ]
disp(           )
```

3 Boucles `while` et `for`

3.1 Boucle `for`

Boucle `for`

La syntaxe est la suivante :

```
for var=vect
    instruction
end
```

où

- `vect` est un vecteur de valeurs,
- `var` est une variable qui prend successivement les valeurs contenues dans `vect`,
- `instruction` est un bloc d'instructions qui va être exécuté successivement pour chaque valeurs de `var`.

En général, `var` est de la forme `deb:pas:fin` c'est-à-dire le vecteur prenant les valeurs entre `deb` et `fin` espacées de `pas`.

Soit F la fonction définie sur $\mathbb{R}$ par

$$\forall x \in \mathbb{R},\ \mathrm{F}(x) = 1 - e^{-x}.$$

On considère la suite $(u_n)_{n\geqslant 1}$ définie par $u_1 = 1$ et pour tout $n \geqslant 1$, $u_{n+1} = \mathrm{F}(u_n)$ et on admet qu'elle est bien définie.

► Compléter le programme suivant qui permet de représenter les cent premiers termes de la suite.

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

► A l'aide de la représentation graphique obtenue, que peut-on conjecturer quant à la monotonie et la limite de la suite ?

3.2 Boucle `while`

Boucle `while`

La syntaxe est la suivante :

```
while condition
    instruction
end
```

où

- `condition` est une condition (c'est-à-dire une expression qui prend la valeur vraie ou fausse),
- `instruction` est un bloc d'instructions qui va être exécuté tant que `condition` est vraie.

On considère une suite $(u_n)_{n\in\mathbb{N}}$ telle que

$$\forall n \in \mathbb{N},\ e^{-\sqrt{u_n}} \leqslant u_n - n \leqslant e^{-\sqrt{n}}.$$

► Compléter le script Scilab suivant afin qu'il affiche un entier n pour lequel $u_n - n$ est inférieur ou égal à 10^{-4}.

```
n=0
while -----
    n = -----
end
disp(n)
```

4 Fonctions

Les fonctions

La syntaxe d'une fonction est la suivante :

```
function sortie=maFonction(entree)
    corps
endfunction
```

où

- `sortie` : le ou les éléments renvoyés par la fonction (zéro, une ou plusieurs variables),
- `maFonction` : le nom de la fonction,
- `entree` : le ou les arguments d'entrée de la fonction (zéro, uen ou plusieurs variables),
- `corps` : la suite finie d'instructions qui permet à la fonction de calculer `sortie` à partir de `entree`.

On considère la $A = \begin{pmatrix} 2 & 1 & -2 \\ 0 & 3 & 0 \\ 1 & -1 & 5 \end{pmatrix}$ et on pose $X_0 = \begin{pmatrix} 3 \\ 0 \\ -1 \end{pmatrix}$. Pour tout entier naturel n, on pose $X_{n+1} = \frac{1}{6} AX_n$.

► Compléter et recopier dans le fichier .sci la fonction ci-dessous qui prend en argument un entier supérieur ou égal à 1 et qui renvoie X_n :

```
function res=X(n)
    X=[3;0;-1]
    A=[2,1,-2;0,3,0;1,-1,5]
    for i=1:n
        X = ---------
    end
    res= ---------
endfunction
```

5 Graphiques

La commande `plot2d`

- Si X et Y sont deux vecteurs lignes ou deux vecteurs colonnes de même taille, l'instruction `plot2d(X,Y)` trace la ligne brisée reliant les points dont les abscisses sont données par X et les ordonnées par Y.
- Si X, Y et Z sont trois **vecteurs colonnes de même taille**, l'instruction `plot2d(X,[Y,Z])` trace Y en fonction de X puis Z en fonction de X. Cela se généralise avec la commande `plot2d(X,[Y_1,Y_2,...,Y_n])`
- La commande `clf` permet de supprimer le contenu de la fenêtre graphique courante.

On considère la suite $(x_n)_{n\in\mathbb{N}}$ définie pour tout entier naturel n par $x_{n+1} = \frac{1}{2}x_n + 3$ et $x_0 = 1$.

► Tracer les 10 premiers termes de la suite et recopier la commande :

Tracer de fonctions

Si X est un vecteur ligne et `f` une fonction préalablement créée, l'instruction `fplot2d(x,f)` permet de tracer la ligne brisée reliant les points $\{(x, f(x)),\ x$ coefficient de X$\}$.

Exercice 1

Soit f la fonction définie sur $[-1,4]$ par

$$\forall x \in [-1,4]\,,\ f(x) = \sqrt{1+x}.$$

1. *Définir une fonction `f` qui prend en argument un nombre x dans $[-1,4]$ et qui renvoie $\sqrt{x+1}$.*
2. *Définir une fonction `graphe` qui prend en argument un entier naturel n (et ne possède pas de sortie) et qui affiche la représentation graphique de `f` avec n points (on pourra utiliser la commande `linspace` pour obtenir un vecteur de taille n constitué de nombre uniformément espacés entre -1 et 4).*
3. *Modifier la fonction `graphe` pour qu'elle affiche aussi, sur la même fenêtre graphique, la courbe de la fonction identité.*
4. *On s'intéresse à la suite $(u_n)_{n\in\mathbb{N}}$ définie par $u_0 = -0.5$ et $\forall n \in \mathbb{N}$, $u_{n+1} = f(u_n)$.*
   (a) *On considère les points $A_0 = \begin{pmatrix} u_0 \\ u_0 \end{pmatrix}$, $A'_0 = \begin{pmatrix} u_0 \\ u_1 \end{pmatrix}$, $A_1 = \begin{pmatrix} u_1 \\ u_1 \end{pmatrix}$, $A'_1 = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \ldots$, $A_n = \begin{pmatrix} u_n \\ u_n \end{pmatrix}$, $A'_n = \begin{pmatrix} u_n \\ u_{n+1} \end{pmatrix}$. Écrire une fonction qui prend en argument un entier naturel n et qui renvoie la matrice*

   $$\begin{pmatrix} A_0 & A'_0 & \ldots & A_n & A'_n \end{pmatrix} = \begin{pmatrix} u_0 & u_0 & u_1 & u_1 & \ldots & u_n & u_n \\ u_0 & u_1 & u_1 & u_2 & \ldots & u_n & u_{n+1} \end{pmatrix}$$

   (b) *Écrire une fonction qui prend en argument un entier naturel n et qui affiche, sur une même fenêtre graphique, la courbe représentative de f, de la fonction identité et la ligne brisée reliant les points A_0, $A'_0, \ldots, A_n, A'_n$.*
   (c) *Tester la fonction avec $n = 100$ ou $n = 1000$. Quel résultat du cours cela illustre-t-il?*

6 Éléments du programme officiel

1. Les commandes suivantes, rencontrées durant ce TP, ainsi que leurs arguments sont exigibles :

   `zeros` , `ones` , `eye`

2. Les commandes suivantes, rencontrées durant ce TP, doivent savoir être manipulées mais la connaissance de leurs arguments n'est pas exigible :

   `plot2d` , `fplot2d`

3. Savoir-faire exigibles (première année) :
   - calcul des termes d'une suite
   - calcul de la valeur approchée de la limite d'une suite ou de la somme d'une série.