Atelier Transfer : Initiation au Scilab

Aurélien GOUDJO[1]

28 novembre 2014

[1]Département de Mathématiques FAST/UAC

Table des matières

Chapitre 1

Introduction

L'objectif de cet atelier est d'initier à l'utilisation d'un puissant outil de calcul, d'un assistant mathématique qu'est le Scilab. On partira d'études de cas à priori sans interêt mathématique pour introduire des concepts qui seront modélisés à l'aide de cet outil.

Avant tout propos :

- C'est quoi exactement Scilab ?
- Que peut-t-on attendre du Scilab ?
- Qu'est ce qu'il ne faut pas expérer du Scilab ?
- Comment obtient-t-on Scilab ?

1.1 Les logiciels de calcul scientifique

Scilab est dans la famille des logiciels de calcul scientifique. C'est un outil de modélisation et de calcul. Contrairement à beaucoup de produits que l'on rencontre dans le monde industriel qui sont conçus pour un thématique précis, Scilab est généraliste et peut s'adapter à divers contexte par le développeement de boîte à outils.

Les logiciels de calcul scienfique se repartissent en général dans deux catégories :

- logiciels de calcul numérique
- logiciels de calcul formel

Les logiciels de calcul numérique sont ceux qui reçoivent en entrée des données numériques (en général des flottants 32 ou 64 bits) et qui crachent à la sortie des résulatats numériques. Dans cette catégorie on peut citer :

- R, SPSS : spcialisés en statistique
- Octave, Scilab, Matlab : généralistes
- Excel, OpenCalc : des avatars

Les logiciels de calcul formel sont ceux qui dans le traitement de certains objets mathématiques imitent les règles et procédures du mathématicien humain sans se préoccuper de l'aspect numérique. On peut citer dans cette catégorie :

- Maple, Mupad, Maxima

– Mathematika : un hybride

1.2 La position du Scilab

Scilab est un logiciel généraliste de calcul numérique développé autour d'une bibliothèque d'algèbre linéaire du domaine publique : **Lapack**.

Scilab n'est pas conçu pour le calcul formel, c'est à dire afficher par exemple que la dérivée de la function $x \mapsto \sin(x)$ est la function $x \mapsto \cos(x)$; par contre à l'aide d'un algorithme numérique il peut aider au calcul d'une approximation de cette dérivée et en afficher la courbe pour une suite de réels donnée.

Scilab un logiciel libre, gratuit, open source dont la licence est cependant propriété d'un consorsium chapeauté par l'INRIA. Ce n'est pas tout à fait un logiciel libre au sens **GPL**.

Scilab a été très proche de Matlab à l'origine. Les deux produits ont connu de légère différences du fait de la scission de l'équipe initiale de développement à l'issue d'un désaccord sur le type de licence. Matlab est passé sous licence commerciale.

Les différences sont surtout notables sur les outils graphiques et de programmation bien que la philosophie générale soit la même.

1.3 Les fonctionnalités du Scilab

Scilab dispose de :

1. Un interpreteur de commandes, **scilex** qui en fait un puissance langage de script
2. Un environnement graphique 2D et 3D permettant de tracer des courbes et des surfaces à partir de données numériques
3. Un environnement de modélisation par blocs et de simulation des systèmes dynamiques **scicos**
4. Un interface d'intégration logiciel permettant exploiter des bibliothèques C, Fortran et Java développées par l'utilisateur
5. Diverses boîtes à outils spécialisées couvrant beaucoup de champs d'applications

1.4 Espace de travail Scilab

On rentre dans l'espace de travail de Scilab en démarrant une session Sclab. Cet espace de travail tourne autour **scilex** son interprêteur de commandes. Il donne accès notamment à :

1. l'interprêteur qui traduit et exécute toutes commandes valides directement saisies au clavier. Au cours d'une session les commandes exécutées sont mémorisées dans un historique et les données manipulées sont empilées

dans la portion de la mémoire vive de votre ordinateur gérée par **scilex**. A la fin d'une session toute donnée non explicitement sauvegardée est perdue.

2. un **menu fichier** permettant entre autre de :
   - Sauvegarder une session
   - Recharger une session
   - Changer de repertoire de travail
   - Charger un fichier de fonctions (.sci)
   - Exécuter un script (.sce)
3. un éditeur de texte intégré le **scipad** pour saisir les scripts
4. un simulateur de systèmes dynamique le **scicos**
5. le menu d'**Aide en ligne**

1.4.1 Ouverture d'une session Scilab

Avant toute chose il faut vous assurer que Scilab est installé sur votre système. Une fois que cela est acquis, pour démarrer une session Scilab, selon la plateforme, on peut :

- soit utiliser le menu **Démarrer/Applications/...**
- soit utiliser la commande **scilab** dans une console

On a alors, selon la version installée, une fenêtre analogue à celle de la figure 1.1.

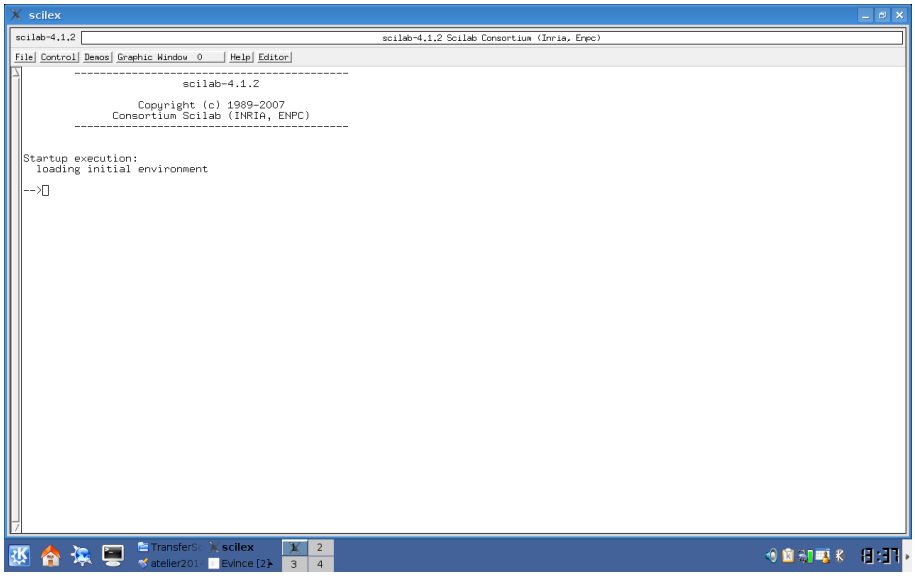

FIG. 1.1 – Fenêtre scilex

1.4.2 Fermeture d'une session Scilab

On a au moins deux alternatives :

- La façon la plus naturelle de fermer une session Scilab est de saisir la commande **quit** ou **exit** dans **scilex**.
- On peut aussi tout simplement tuer le processus en fermant la fenêtre de **scilex**.

Exercice Une visite des fonctions help, who et whos
C'est un exercice pour un premier contact avec la console Scilab.

1. Démarrer Scilab
2. Saisir la commande **who()**. Observer
3. Saisir la commande **whos()**. Observer et commenter
4. Saisir la commande **clc**. Commenter
5. Saisir la commande **help who**. Observer et commenter

Remarque :
Bien que Scilab a un mode d'usage interactif, dans toute la suite nous optons pour l'utilisation des fichier de script afin de garder trace de la gestion des erreurs de synthase et de logique.
Remarque :
Pour ne pas perdre du temps à chercher nos fichiers nous allons créer sur le **Bureau** de notre machine un dossier nommé **ExoScilab**. *Il serait judicieux de respecter la casse.*
Exercice Changement de repertoire de travail
Dans cet exercice on utilisera le menu **Fichier** de **scilex** pour nous positionner dans le repertoire **ExoScilab**. On vérifie ensuite avec la commande

```
pwd
```

Exercice Nommage des fichiers d'exercice
Pour faciliter la recherche, on adoptera le nommage incrémental suivant. En supposant que nous sommes le 10 Novembre 2014, le premier fichier d'exercice de la journée sera nommé : **exo2014111001** et le cinquième exercice de cette journée portera le nom **exo2014111005**. Nous utiliserons **scipad** pour cet exercice.

Chapitre 2

Scilab : Une calculatrice vectorielle

Etude de cas 1

On considère les notes de dix étudiants dans quatre Unités d'Enseignement listées dans la table 2.1 qui suit.

	FVR	ATO	LIN	CED
E1	17.50	15.94	14.46	12.00
E2	10.00	16.70	8.75	10.89
E3	13.30	10.38	9.50	13.43
E4	16.65	14.53	9.89	11.83
E5	16.30	10.78	9.33	15.43
E6	18.50	10.77	11.53	9.30
E7	16.85	10.59	10.86	12.37
E8	18.75	18.60	9.65	10.85
E9	10.50	15.83	11.75	11.90
E10	15.50	11.69	11.00	10.96

Tab. 2.1 – Données de l'Etude ce cas 1

Objectifs

A l'issue des manipulations l'apprenant sera capable de :

1. Ecrire un script scilab
2. Déclarer une variable scalaire
3. Déclarer un vecteur ligne, un vecteur colonne : les opérateurs de concaténation
4. Déclarer une matrice
5. Faire les opérations arithmétiques vectorielles standard
6. Modifier les composantes d'un tableau

7. Extraire des parties d'un tableau
8. Assembler des tableaux compatibles
9. Générer des tableaux à partir de vecteurs et matrices types
10. Utiliser quelques opérations non standard

2.1 Données usuelles en Scilab

Données de base et instructions

Scilab manipule principalement des valeurs constantes sous forme de :
- nombres réels codés flottants 64 bits
- nombres complexes également codés flottants

Accessoirement on peut avoir aussi des bouléens, des polynômes et des chaines de caractères.

Scilab utilise des constantes prédéfinies comme par exemple :

Constante	Représentée par
π	%pi
e	%e
$\imath$	%i
∞	%inf
Vrai	%t
Faux	%f

Scilab utilise les caractères de l'alphabet anglais, les chiffres, le blanc et un certain nombre de symboles pour ecrire les instructions.

Type de caractère	Liste
Alphabet	a b ... z A B ... Z
Chiffres	0 1 ... 9
Symboles	+ - * / ^ < > = $ — &
Ponctuations	, ; . !
Délimiteurs	( ) [ ]

Remarque :
Scilab fait la différence entre le majuscule et le minuscule (on dit que Scilab est sensible à la casse).

La touche Entrée notifie à **scilex** la fin d'une instruction et déclenche son exécution suivi de l'affiche du résultat.

Remarque :
Le caractère ; à la fin d'une instruction empêche l'affichage du résultat.

Remarque :
La séquence de caractères // permet de faire des commentaires. Sur une ligne, tout ce qui suit cette séquence est ignoré par **scilex**.

Déclaration de variable

De base toute donnée en Scilab est une matrice de données usuelles. On dit que *Scilab est vectoriel*.

La façon souple de manipuler une donnée en Scilab passe par une déclaration de variable qui se fait :

- soit par initialisation
- soit par affectation

Cette déclaration réserve dynamiquement de l'espace dans la mémoire vive et y stocke la donnée. Le type d'une variable peut changer de façon dynamique au gré des affectations comme en **Excel** par exemple. On dit que *Scilab est non typé*.

Exercice Type de donnée
Passe en revue la flexibilité de Scilab dans l'utilisation des variables. On regarde aussi la fonction **size()**.

2.2 Acquisition des vecteurs et matrices

Dans cette section la table de données 2.1 sera regardée sous trois angles :

1. Comme une succession de lignes
2. Comme une succession de colonnes
3. Comme une entité globale

Pour les manipulations qui vont suivre :

- Chaque ligne portera le nom libellé à gauche de cette ligne.
- Chaque colonne portera le nom libellé au dessus de cette colonne.
- L'entité globale portera le nom : Lab1.

Pour initialiser un vecteur ou une matrice on utilise en général un opérateur de concaténation : `[ ,..., ]` ou `[ ;... ; ]`

Opérateur de concaténation horizontale

L'opérateur de concaténation horizontale `[ ,..., ]` permet de définir un vecteur ligne. Par exemple le vecteur de $\mathbf{R}^5$ $u = (3, 5, 2, 9, 0)$ sera déclaré par :

```
u=[3,5,2,9,0]
```

On peut aussi simplement remplacer `,` par un blanc ; c'est à dire :

```
u=[3 5 2 9 0]
```

Exercice Vecteur ligne
Déclarer les lignes de la matrice Lab1 en utilisant l'opérateur de concaténation horizontale. Utiliser la fonction `size()` pour vérifier leurs dimensions.

Opérateur de concaténation verticale

L'opérateur de concaténation verticale [;. . . ;] permet de définir un vecteur colonne. Par exemple le vecteur de $\mathbf{R}^4$ $v = \begin{pmatrix} 0.5 \\ -1.75 \\ 2 \\ 0.9 \end{pmatrix}$ sera déclaré par :

```
v=[0.5 ;-1.75 ;2 ;0.9]
```

Exercice Vecteur colonne
Déclarer les colonnes de la matrice Lab1 en utilisant l'opérateur de concaténation verticale. Utiliser la fonction size() pour vérifier leurs dimensions.

Opération de transposition

La transposition d'un vecteur réel transforme un vecteur ligne en vecteur colonne et vice versa. L'opérateur Scilab correspondant est ' Par exemple on peut déclarer le vecteur ligne u par

```
u=[3 ; 5 ; 2 ; 9 ; 0]'
```

et le vecteur colonne v par

```
v=[0.5,-1.75,2,0.9]'
```

Acquisition de matrice

La construction de données matricielles sous Scilab peut se faire :
- Soit par concaténation verticale de vecteurs lignes de même dimension
- Soit par concaténation horizontale de vecteurs colonnes de même dimension
- Soit par concaténation mixte directe

Exercice Matrice

1. Construire sans afficher la donnée A par concaténation horizontale des vecteurs FVR, ATO, LIN et CED.
2. Construire sans afficher la donnée B par concaténation verticale des vecteurs $E1$, $E2$,. . .$E10$.
3. Calculer les dimensions de A et B puis comparer A et B en utilisant la commande Comp=isequal(A,B)
4. Construire sans afficher la donnée C par concaténation verticale des vecteurs FVR, ATO, LIN et CED. Calculer la dimension de C
5. Construire sans afficher la donnée D par concaténation horizontale des vecteurs $E1$, $E2$,. . .$E10$. Calculer la dimension de D

2.3 Opérateurs arithmétiques

Scilab dispose de toutes les opérations matricielles standard.

Opérateur	Action
A+B	Addition
A-B	Soustraction
k*A	Multiplication par un scalaire
A*B	Produit matriciel
A^k	Puissance matricielle

Il dispose également d'un certain nombre d'opérateurs exotiques comme :

Opérateur	Action
A.*B	Produit terme à terme de A par B
A./B	Quotient terme à terme
A.^n	Puissance terme à terme
A.^B	Puissance terme à terme
$x = A\backslash b$	Résolution de $Ax = b$
$x = b/A$	Résolution de $xA = b$

Remarque :
Scilab dispose également d'une vaste bibliothèque de fonctions relative au traitement des problèmes matriciels. On citera notamment la fonction `sum()`.

Exercice
On considère le tableau $Lab1$.

1. calculer le vecteur $S1$ contenant la somme des notes de chacun des étudiants en utilisant l'opérateur +
2. calculer le vecteur $S2$ contenant la somme des notes de chacun des étudiants en utilisant la fonction *sum*. Comparer $S1$ et $S2$ à l'aide la fonction *isequal*
3. calculer le vecteur $M1$ contenant la moyenne des notes de chacun des étudiants lorsque la pondération est uniforme pour toutes les UE.
4. calculer le vecteur $M2$ contenant la moyenne des notes de chacun des étudiants lorsque la table de pondération est donnée par vecteur $coef2 = (5, 3, 4, 4)$.
5. calculer le vecteur $M3$ contenant la moyenne des notes de chacun des étudiants lorsque la table de pondération est donnée par vecteur $coef3 = (3, 2, 4, 5)$.

2.4 Vecteurs et matrices spéciaux

2.4.1 Suite arithmétique

Scilab dispose de l'opérateur `:` à cet effet.

Commande	Action
$a : b$	suite arithmétique de raison $1 \leq b$ partant de a si $a \leq b$, sinon vide
$a : h : b$	suite arithmétique de raison h partant de a

2.4.2 Quelques matrices spéciales

Soit A un tableau de dimension $n \times m$

B=ones(A) ou B=ones(n,m)

génère un tableau B dont tous les composantes sont à 1

B=zeros(A) ou B=zeros(n,m)

génère un tableau B dont tous les composantes sont à 0

I=eye(A) ou I=eye(n,m)

génère un tableau $I = (I_{ij})$ dont les composantes vérifient $I_{ij} = \delta_{ij}$ symbole de Kronecker.

Remarque :
Parfois on a besoin d'un tableau vide. On l'obtient par [].

2.5 Manipulation des données

2.5.1 Réferencement et extraction

Cas d'un vecteur

Scilab référence la position d'un élément d'un vecteur $A = (A_i)_{i=1}^n \in \mathbf{R}^n$ par un entier positif compris entre 1 et n. On utilise la commande, pour $1 \leq k \leq n$,

A(k)

Remarque :
Lorsque l'indice k sort de la place $1, \ldots, n$, Scilab affiche un message d'erreur.

Soit $U = (3, 0.25, 4.7, -5\imath)$. On a

Commande	Affichage
U(1)	ans = 3.
U(3)	ans = 4.7
U(4)	ans = $-5.\imath$
U($)	ans = $-5.\imath$
U(5)	un message d'erreur

Remarque :
Comme les tableaux sont dynamiques en Scilab, la dimension d'un tableau peut varier ainsi que la valeur de l'indice final d'une ligne ou d'une colonne. Symboliquement on désigne cette valeur par $\boxed{\$}$.
Exercice
Faire $V = U'$ pour obtenir le transposé de U. Répéter les commandes précédentes en remplaçant U par V. Commenter

Cas d'un tableau rectangulaire

Pour une matrice $A = (A_{ij})_{\substack{1\leq i\leq n\\ 1\leq j\leq m}}$ on utilisera la commande

$$\boxed{A(l\,,\,k)}$$

où les indices l et k respectent les bonnes plages.
Remarque :
Scilab autorise la commande

$$\boxed{A(u)} \text{ ou } \boxed{A(u\,,\,v)}$$

où u et v sont des vecteurs d'entiers positifs dont les valeurs respectent les bonnes plages.
Exercice
On considère la matrice

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \end{bmatrix}$$

Commande	Affichage
A(1,3)	ans = 3.
A(3,4)	ans = 14.
A(1,$)	ans = 5.
A($,$)	ans =20.
A(2, :)	affiche la ligne 2
A(7, :)	affiche un message d'erreur
A(:,$)	affiche la dernière colonne
A(1 :3,[4,2])	affiche ans = 4. 2. 9. 7. 14. 12.

Exercice
On considère le tableau $A = Lab1$. Déterminer les tableaux suivants

1. U est formé des données de la ligne 5 en les disposant de la derniere colonne à la première
2. B est la matrice formée des lignes d'indice pair de A

3. C est obtenu en supprimant la colonne 2 et les lignes d'indice pair de A puis en réordonnant les colonnes dans l'ordre inverse.

Remarque :
Les manipulations de cette sous section ne modifient ni la structure ni le contenu d'un tableau

2.5.2 Affectation

Cette opération est invasive et altère le contenu des tableaux. Elle peut également affecter leur structure.

L'affectation permet de modifier la valeur d'un élément ou d'un ensemble contigu d'éléments d'un tableau.

Exercice
On reprend la matrice

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \end{bmatrix}$$

Soit $B = A$

Commande	Affichage
B(2,3)=-4	affiche B avec le contenu de l'élément référencé passé de 8 à -4
B(3, :)=[]	affiche B en supprimant la ligne 3 B n'a plus 3 lignes
B(4,2 :4)=[5,-1,3]	affiche B en recréant la ligne 4 en attribuant des valeurs aux éléments des colonnes 2, 3 et 4 et mettant les autres à 0
B(2 :3,3 :4)=[25,35 ;-3, -18]	affiche B en insèrant la matrice $\begin{matrix} 25 & 35 \\ -3 & -18 \end{matrix}$ à la place de la sous-matrice $\begin{matrix} 8 & 9 \\ 18 & 19 \end{matrix}$

2.5.3 Concaténation

Cette opération permettent de rajouter des lignes ou des colonnes à un tableau. Les opérandes dans cette opération doivent être compatibles au sens que :

- Pour la concaténation horizontale $\boxed{\text{[A,B]}}$, les tableaux A et B doivent avoir le même nombre de lignes
- Pour la concaténation verticale $\boxed{\text{[A ;B]}}$, les tableaux A et B doivent avoir le même nombre de colonnes

Ici le resultat peut être affecté à un autre tableau.

Exercice

On reprend la matrice

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \end{bmatrix}$$

Commande	Affichage
B=[A,sum(A,'c')]	affiche un tableau B contenant A et une colonne de somme à la fin
C=[sum(A,'2'),A]	affiche un tableau C qui comme B contient A et une colonne de somme placée au debut
D=[A ;sum(A,'1')]	affiche un tableau B contenant A et une ligne de somme à la fin
E=[sum(A,'r') ;A]	affiche un tableau C qui comme B contient A et une ligne de somme au debut

2.6 Exercices Pratiques

Exercice 1 (Table d'opération) *On considère deux vecteurs lignes :* $u = (u_i)_{i=1}^{5} \in \mathbf{R}^5$ *et* $v = (v_i)_{i=1}^{7} \in \mathbf{R}^7$.

On suppose que u *et* v *sont des suites arithmétiques de raison 1 de premier terme 1.*

Sans utiliser de boucle,

1. *Générer sans afficher les vecteurs* u *et* v.
2. *Générer la matrice* $M = (M_{ij})$ *définie par :* $M_{ij} = u_i * v_j$
3. *Générer la matrice* $A = (A_{ij})$ *définie par :* $A_{ij} = u_i + v_j$
4. *Générer la matrice* $B = (B_{ij})$ *définie par :* $B_{ij} = 5u_i - 2v_j$
5. *Générer la matrice* $C = (C_{ij})$ *définie par :* $C_{ij} = 2u_i^2 + v_j^2$

Exercice 2 (Approximation de dérivées) *On considère la fonction numérique* f *définie sur l'intervalle* $[-1\,,\,1]$ *par* $f(x) = -\dfrac{2}{x^2+1}$ *Sans utiliser de boucle,*

1. *Générer sans affichage la suite arithmétique* $x = (x_i)_{i=1}^{51}$ *telle que* $x_1 = -1$ *et* $x_{51} = 1$
2. *Générer sans affichage le vecteur* $h = (h_i)_{i\geq 1}$ *tel que* $h_i = x_{i+1} - x_i$
3. *Générer sans affichage le vecteur* $y = (y_i)_{i=1}^{51}$ *tel que* $y_i = f(x_i)$
4. *Générer le vecteur* $d = (d_i)_{i\geq 1}$ *tel que*

$$d_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

5. *Générer le vecteur* $D = (D_i)_{i\geq 1}$ *tel que*

$$D_i = \frac{y_{i-1} - 2y_i + y_{i+1}}{(x_{i+1} - x_i)^2}$$

Exercice 3 *Pour les vecteurs colonnes* $u = (u_i) \in \mathbf{R}^{10}$ *et* $v = (v_i) \in \mathbf{R}^{12}$ *tels que*

$$u_i = 1 + i^2 \text{ et } v_i = 4i - 1$$

calculer sans utiliser de boucle la matrice $A = (a_{ij})$ *avec* $a_{ij} = \frac{u_i}{v_j}$

Exercice 4 *On considère le tableau* $A = Lab1$*. Déterminer les tableaux suivants*

1. *B est la matrice formé des notes cœfficiées de A, sachant que la table des cœfficients est donnée par le vecteur ligne* $coeff = (5, 3, 4, 5)$*, complètées d'une colonne des moyennes cœfficiées*
2. *C est la matrice formée des lignes d'indice pair de A affectées des facteurs listés dans l'ordre dans le vecteur* $corrig = (1.25, 1, 0.8, 1.2, 1.3)$

Exercice 5 *Résoudre le système d'équations d'inconnues X, Y et Z*

$$\left\{ \begin{array}{rcrcrcrcr} 8X & + & 2Y & + & 3Z & = & A & + & 2B \\ X & - & 9Y & + & 2Z & = & 3A & - & 2J \\ 2X & + & 3Y & + & 6Z & = & A & - & B \end{array} \right.$$

avec

1. $$A = \begin{pmatrix} 3 & 2 \\ 7 & 5 \end{pmatrix}, B = \begin{pmatrix} 2 & -3 \\ 1 & 3 \end{pmatrix}, J = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
2. $$A = \begin{pmatrix} 3 & 2 \\ 7 & 5 \\ -3 & 1 \end{pmatrix}, B = \begin{pmatrix} 2 & -3 \\ 1 & 3 \\ 2 & 1 \end{pmatrix}, J = \begin{pmatrix} 0 & 1 \\ -1 & 0 \\ 0 & 1 \end{pmatrix}$$

Exercice 6 (Zooms successifs) *On considère la fonction réelle d'une variable r'eelle définie par :*

$$f(x) = x^4 - 4x^2 + x + 1$$

On se propose de rechercher par tabulation les solutions à 10^{-2} *près de* $f(x) = 0$*, en utilisant la technique de zooms successifs.*

1. *Soit a=-3 et b=3. Calculer la suite arithmétique* $x = (x_i)_{i=1}^{25}$*. Calculer la suite* $y = (y_i)_{i=1}^{25}$ *telle que* $y_i = f(x_i)$*. Puis afficher la matrice à deux colonnes* $Y = (x_i\,,\, y_i)_{i=1}^{25}$*, pour localiser les changements de signe dans la colonne 2. Tout changement de signe encadre un zéro de* f*. Il faut adapter la taille des tableaux affichés aux dimension du moniteur.*
2. *Chaque couple* $(x_k\,,\, x_{k+1})$ *localisant un changement de signe de y permet d'initialiser une nouvelle tabulation comme dans la question 1 avec* $a = x_k$ $b = x_{k+1}$*. On dit qu'on fait un* Zoom *ou un* Raffinement*. On arrête le cycle de raffinement lorsque* $|x_{k+1} - x_k| < 0.5\,10^{-2}$

Chapitre 3

Scilab : outil de graphisme numérique

Etude de cas 2

On considère les fonctions numériques définies sur l'intervalle $[-2\,,\,2]$ par : $f_1(x) = x^3 - 3x + 2$, $f_2(x) = \dfrac{2}{x^2+1}$ et $f_3(x) = \sin(2x)$

Dans un premier temps, on génère le vecteur ligne $X = (x_i)_{i=1}^{81}$ tel que $x_1 = -2$, $x_{81} = 2$ et les autres points sont équirépartis.

On calcule ensuite les vecteurs $Y_1 = f_1(X)$, $Y_2 = f_2(X)$ et $Y_3 = f_3(X)$.

Objectifs

A l'issue des manipulations l'apprenant sera capable de :

1. Construire une courbe associée aux données réelles $(x_i\,,\,y_i)_{i=1}^{n}$
2. Nommer les axes
3. Mettre un titre au graphique
4. Choisir et activer une fenêtre graphique
5. Construire plusieurs courbes dans la même fenêtre graphique
6. Choisir une couleur de trait ou un marqueur
7. Mettre une légende
8. Utiliser des sous-fenêtres graphiques

3.1 Primitives graphiques 2D : plot et plot2d

Pour tracer les graphiques 2D en Scilab on recommande **plot2d**. Cette fonction graphique gère en standard la surimpression. La subsistance de la fonction **plot** est une des rares manifestations de son héritage commun avec Matlab. Les plus grandes différences entre Scilab et Matlab se retrouvent au niveau de leurs environnements graphiques.

Cas d'une seule courbe

Pour tracer la courbe plane définie par $(x_i, y_i)_{i=1}^n$, on définit dans Scilab les vecteurs $x = (x_i)_{i=1}^n \in \mathbf{R}^n$ et $y = (y_i)_{i=1}^n \in \mathbf{R}^n$ et on utilise la commande `plot2d(x,y)` ou `plot(x,y)`

Par defaut Scilab trace dans la fenêtre active. A la première utilisation, sauf cas d'activation explicite, la fenêtre active par défaut est la fenêtre numero 0. Théoriquement Scilab peut gérer 256 fenêtres graphiques qui sont numerotées de 0 à 255.

Pour activer explicitement une fenêtre graphique on dispose de la commande `xset('window', k)` où k est le numero de la fenêtre.

Pour effacer le contenu de la fenêtre active on utiliser la commande `clf()` dans les versions de Scilab supérieure à 4 ; sinon on utilise `xbasc()`.

Exercice

On considère l'étude de cas 2.

1. Calculer les vecteurs X, $Y1$, $Y2$ et $Y3$ en utilisant des opérations matricielles
2. Tracer la courbe représentative de f_1 dans la fenêtre active.
3. Tracer la courbe représentative de f_2 dans la fenêtre 5.
4. Tracer la courbe représentative de f_3 dans la fenêtre 3.

Cas de plusieurs courbes

Pour tracer plusieurs courbes dans une fenêtre active on peut :

– soit utiliser la surimpression qui se fait par une succession de **plot2d** pour tracer les courbes individuellement.

– soit utiliser le tracer simultané de toutes les courbes

Pour le tracer simultané on utiliser la commande `plot2d(X,Y)` où

– Y est une matrice dont les colonnes représentent les courbes.

– X peut être un vecteur ou une matrice. Lorsque X est une matrice, elle est de même dimension que Y et ses colonnes correspondent aux courbes

Exercice

On reprend les données de l'étude de cas 2.

1. Activer la fenêtre 5 et tracer la courbe représentative de f_2 puis tracer la courbe représentative de f_1 .
2. Calculer la matrice Y ayant deux colonnes contenant respectivement les données de $Y2$ et $Y1$. Activer et effacer la fenêtre 3 puis utiliser la commande `plot2d(X, Y)`. Commenter

3.2 Paramétrage d'une fenêtre graphique

L'habillage correcte d'un graphique passe par un titre, le libellé des axes et la légende.

Titre d'une fenetre, titre d'un axe

Scilab dispose de la fonction **title** ou de la fonction **xtitle**.

Pour mettre juste un titre à la fenêtre graphique courante, on va se contenter de la syntaxe par défaut : `title('Nom du graphique')`

Lorsqu'on a besoin également de titre pour les axes, la syntaxe la plus indiquée est : `xtitle("Nom du graphique","Nom axe 1","Nom axe 2")`

Style et légende

La fonction graphique **plot2d** autorise un certain nombre de paramètres optionnels dont on retient ici *style* et *leg*.

On a par exemple : `plot2d(X,Y,style=TabStyle, leg=ChaineLeg)` où :

- *TabStyle* est un vecteur d'entiers comme `[2,-1,1]`, un entier positif pour la couleur d'un trait continu et un entier négatif ou nul la marque d'un trait discontinu
- *ChaineLeg* est une chaine de caractères contenant les légendes des courbes séparées par le caractère `@` dans la forme `"Leg1@Leg2@Leg3"` pour trois courbes

Exercice

On reprend les données de l'étude de cas 2.

1. Activer et effacer la fenêtre 5 pour y tracer la courbe représentative de f_1 avec le titre **Une cubique** et les axes portant les noms respectifs **abscisses** et **ordonées**.
2. Activer et effacer la fenêtre 3 puis tracer simultanément les courbes représentatives de f_1, f_2 et f_3 avec le titre **Trois couleurs** et les axes portant les noms respectifs **x** et **y**
3. Activer et effacer la fenêtre 3 puis tracer simultanément les courbes représentatives de f_1, f_2 et f_3 avec le titre **Trois marqueurs** et les axes portant les noms respectifs **x** et **y**. On utilisera `TabStyle=[-1,-2,-3]` et `ChaineLeg="f1@f2@f3"`

3.3 Sous-fenêtres d'une fenêtre graphique

Une fenêtre graphique peut se présenter comme une matrice de sous fenêtres. Pour cette tâche, Scilab dispose de la fonction **subplot**.

On utilise dans une fenêtre active, la syntaxe :

`subplot(Nbligne,Nbcolonne,NumSousfenetre)` suivie de la commande **plot2d**.

- *Nbligne* est le nombre de lignes
- *Nbcolonne* est le nombre de colonnes
- *NumSousfenetre* est le numero de la sous fenêtre choisie

Remarque :

Lorsqu'on utilise le sous-fenêtrage, il faut se garder d'effacer intempestivement

la fenêtre graphique. La fenêtre peut être éventuellement effacée juste à son activation.

Exercice Sous fenêtrage

On reprend les données de l'étude de cas 2.

1. Activer et effacer la fenêtre 2 puis subdiviser la en trois colonnes. Tracer la courbe représentative de
   - f_1 dans la sous fenêtre 1
   - f_2 dans la sous fenêtre 2
   - f_3 dans la sous fenêtre 3
2. Activer et effacer la fenêtre 3 puis subdiviser la en trois lignes. Tracer la courbe représentative de
   - f_1 dans la sous fenêtre 1
   - f_2 dans la sous fenêtre 2
   - f_3 dans la sous fenêtre 3
3. Activer et effacer la fenêtre 4 puis subdiviser la en deux lignes et deux colonnes. Tracer la courbe représentative de
   - f_1 dans la sous fenêtre 1
   - f_2 dans la sous fenêtre 2
   - f_3 dans la sous fenêtre 3

   Tracer sumultanément les courbes représentatives de f_1, f_2 et f_3 dans la sous fenêtre 4

3.4 Exercices Pratiques

Exercice 7 (Note d'étudiants) *On reprend la table* Lab1 *de l'étude de cas 1. On va représenter les notes obtenues dans une UE par un marqueur et l'étudiant Ei est représenté par l'entier i. Pour les quatre UE dans l'ordre FVR, ATO, LIN, CED on utilisera la vecteur de style* $(-1\,,\,-2\,,\,-3\,,\,-4)$

1. *Représenter les notes des quatre UE dans la fenêtre graphique 2.*
2. *Représenter les notes des quatre UE dans la fenêtre graphique 3 subdivisée en 2 lignes et 2 colonnes en remplaçant* ***plot2d*** *par :*
   (a) ***plot2d1*** *dans la sous fenêtre 1*
   (b) ***plot2d2*** *dans la sous fenêtre 2*
   (c) ***plot2d3*** *dans la sous fenêtre 3*
   (d) ***plot2d4*** *dans la sous fenêtre 4*

   Commenter

Exercice 8 (Zooms successifs) *On considère les fonctions réelles d'une variable r'eelle définie par :*

$$f(x) = x^4 - 3x^3 + 7x - 5 \text{ et } g(x) = 4x^3 - 9x^2 + 7$$

On se propose de rechercher graphiquement les solutions à 10^{-2} *près, en utilisant la technique de zooms successifs, de l'équation* $f(x) = g(x)$ *sur* $[-1\,,\,2]$.

1. *Tracer dans la fenêtre graphique 1 les graphes permettant la première localisation des solutions. On utilisera le fait que pour $x \in D_f \cap D_g$ on a l'équivalence*
$$f(x) = g(x) \Leftrightarrow \left\{ \begin{array}{l} y = f(x) \\ y = g(x) \end{array} \right.$$
2. *Chaque niveau de raffinement doit faire l'objet d'une fenêtre graphique où chaque solution est représentée dans une sous fenêtre ; les numeros de fenêtre étant régulièrement incrémentés.*

Exercice 9 (Approximation de dérivée) *Pour tout $x_i = -2 + 0.1(i-1) \in [-2, 2]$, $i = 1, \ldots, n$, et $y_i = f(x_i)$ où $f(x) = \frac{2}{1+x^2}$, déterminer $d = (d_i)$ une approximation de la dérivée première de f définie par :*

$$d_i = \left\{ \begin{array}{ll} \frac{1}{2h}(-3f(x_i) + 4f(x_{i+1}) - f(x_{i+2})) & si\ i = 1 \\ \frac{1}{2h}(f(x_{i+1}) - f(x_{i-1})) & si\ i = 2, \ldots, n-1 \\ \frac{1}{2h}(f(x_{i-2}) - 4f(x_{i-1}) + 3f(x_i)) & si\ i = n \end{array} \right.$$

Dans la fenêtre graphique numero 1 subdivisée horizontalement en deux sous fenêtres,

1. *Tracer dans la sous-fenêtre 1 le graphe $(x_i\,,\, f(x_i))_{i \geq 1}$*
2. *Tracer dans la sous-fenêtre 2 les graphes $(x_i\,,\, f'(x_i))_{i \geq 1}$ et $(x_i\,,\, d_i)_{i \geq 1}$*
3. *Proposer un titre et une légende puis conclure.*

Chapitre 4

Scilab : langage de programmation

Etude de cas 3 On reprend les données de la table 2.1 de l'étude de cas 1.

Dans un premier temps on va définir une fonction **moyenne** permettant de traiter la donnée *Lab*1 pour restituer un vecteur contenant la moyenne coefficiée des étudiants sont une table de coefficients donnée

Dans un second temps on définit une fonction **validation** permettant de traiter la donnée *Lab*1 pour restituer une matrice contenant 1 pour chaque UE avec une note supérieure ou égale à 10 et 0 sinon, puis un vecteur contenant le nombre total d'UE validées par étudiant.

Objectifs

A l'issue des manipulations l'apprenant sera capable de :

1. définir des fonctions en Scilab
2. utiliser l'instruction conditionnelle **if**
3. utiliser opportunément la boucle **for**

4.1 Fonction en Scilab

Le développement des boîtes à outils en Scilab passe par le concept de fonction qui permet de contruire des sous programmes.

Pour un utilisateur Scilab, la manière la plus commode de définir une fonction passe par la syntaxe :

function *Argument_sortie* = nomFonction(*Argument_entree*)
...
...
endfunction

Argument_sortie et *Argument_entree* sont des listes de variables qui peuvent être vides.

Lorsque *Argument_sortie* contient plus d'un élément il a la forme :

[sortie1 , sortie2, ... , sortieN]

Exercice

On considère la matrice $A = Lab1$.

1. Définir la fonction **moyense()** sans argument d'entrée retournant un vecteur contenant la moyenne arithmétique des notes de chaque étudiant.
2. Définir la fonction **moyenae(TabNote)** avec en entrée la table des notes et retournant un vecteur contenant la moyenne arithmétique des notes de chaque étudiant.Tester avec A
3. Définir la fonction **moyenne(TabNote, TabCoef)** avec en entrée les tables des notes et des coefficients et qui retourne un vecteur contenant la moyenne coefficiée des notes de chaque étudiant.Tester avec A et la table des coefficients prenant les valeurs $(1\,,\,1\,,\,1\,,\,1)$ et $(5\,,\,3\,,\,4\,,\,5)$

4.2 Instruction conditionnelle if

Il arrive souvent que l'on soit confronté à des traitements dépendants de la vérification d'une condition. Par exemple l'obtention d'une mention dépend de la note obtenue.

Pour gérer ces traitements, Scilab dispose de l'instruction **if** dont la syntaxe peut être :

```
if      (condition)
then
        instructions
end
```

ou pour deux alternatives :

```
if      (condition)
then
        instruction1
else
        instruction2
end
```

ou pour plusieurs alternatives :

```
if      (condition1)
then
        instruction1
elseif  (condition2)
then
        instruction2
else
        instruction3
end
```

Les conditions sont des instructions à valeurs booléennes en général obtenues à l'aide d'opérateurs de comparaison et d'opérateurs logiques que l'on peut trouver dans la table

Opérateur	Description
==	Egalité
!=	Non égalité
<	Inférieur à
>	Supérieur à
< =	Inférieur ou égal à
> =	Supérieur ou égal à
|	OU logique
&	ET logique
!	Négation

TAB. 4.1 – Opérateurs de comparaison et logiques

Exercice
On considère la matrice $A = Lab1$.

1. Définir la fonction **valider(note)** retournant la chaine
   – "Admis" si la note est supérieure ou égale à 10
   – "Refus" sinon
   Tester en utilisant les données de la matrice A.
2. Définir la fonction **mention(note)** retournant la chaine
   – "Passable" si la note est entre 10 et 12, 12 exclus
   – "ABien" si la note est entre 12 et 14, 14 exclus
   – "Bien" si la note est entre 14 et 16, 16 exclus
   – "TBien" si la note est entre 16 et 18, 18 exclus
   – "Excellent" si la note est entre 18 et 20.
   On utilisera à cet effet la table
   TabMention=["Passable","Abien","Bien","Tbien","Excellent"]
   Tester en utilisant les données de la matrice A.

4.3 Instruction de boucle for

Scilab est un interprèteur et de ce fait on doit faire un usage très très modéré des boucles car elles provoquent un ralentissement des traitements.

Il faut convenir cependant que pour le traitement de certains problèmes, le recours à une boucle est inévitable. Et comme tout bon langage informatique qui se respecte Scilab dispose d'un panopli d'instructions itératives dont l'instruction **for**.

L'instruction **for** est souvent utilisée avec la syntaxe :

```
for   i=1 : N,
      instructions
end
```

ou

```
for   x=TabX,
      instructions
end
```

où *TabX doit être absolument un vecteur ligne.*

Remarque :

Dans le bloc d'une boucle il faut empêcher l'affichage des resultats en terminant les instructions par un ; car cela peut geler la poursuite du traitement.

Exercice

On considère la matrice $A = Lab1$.

1. Définir la fonction **validation(TabNote)** retournant une table de même dimension que TabNote et contenant 1 pour les notes supérieure ou égale à 10 et 0 sinon.

   Tester en utilisant les données de la matrice A.

2. Définir la fonction **validationp(TabNote)** qui améliore la fonction **validation(TabNote)** de la question 1 en retournant un vecteur suplémentaire contenant le nombre d'UE validées par chaque étudiant.

   Tester en utilisant les données de la matrice A.

4.4 Exercices Pratiques

Exercice 10 *On considère la matrice $A = Lab1$.*

1. *Définir la fonction **Moyenne(TabNote, TabCoef)** avec en entrée les tables des notes et des coefficients et qui retourne une table contenant les notes coefficiées ainsi qu'un vecteur contenant la moyenne coefficiée des notes de chaque étudiant.Tester avec A et la table des coefficients prenant les valeurs* $(1\,,\,1\,,\,1\,,\,1)$ *et* $(5\,,\,3\,,\,4\,,\,5)$
2. *Définir la fonction **MFusion(TabNote, TabCoef)** avec en entrée les tables des notes et des coefficients et qui retourne une table contenant les notes coefficiées et, en dernière colonne, la moyenne coefficiée des notes*

de chaque étudiant.Tester avec A et la table des coefficients prenant les valeurs $(1\,,\,1\,,\,1\,,\,1)$ *et* $(5\,,\,3\,,\,4\,,\,5)$

Exercice 11 *On considère le problème de Cauchy*

$$\begin{cases} \frac{dx}{dt} & = 2x\left(1-\frac{x}{2}\right)-xy \quad t\in[0,15] \\ \frac{dy}{dt} & = 3y\left(1-\frac{y}{3}\right)-2xy \\ x(0) & = x_0 \\ y(0) & = y_0 \end{cases}$$

et la méthode de Runge Kutta définie pour l'équation différentielle $\frac{dx}{dt}=f(t,x)$ *par*

$$x(t+h)=x(t)+\frac{1}{6}(K_1+4K_3+K_4)$$

avec

$$\begin{cases} K_1 & = hf(t\,,\,x(t)) \\ K_2 & = hf\Big(t+\frac{1}{2}h\,,\,x(t)+\frac{1}{2}K_1\Big) \\ K_3 & = hf\Big(t+\frac{1}{2}h\,,\,x(t)+\frac{1}{4}K_1+\frac{1}{4}K_2\Big) \\ K_4 & = hf(t+h\,,\,x(t)-K_2+2K_3) \end{cases}$$

Après la résolution du problème de cauchy pour les couples

$$(x_0,y_0)\in\left\{\begin{array}{l}(0.1\,,\,0.2)\,,\,(0.2\,,\,0.1)\,,\,(0.3\,,\,0.3)\,,\\(0.5\,,\,0.2)\,,\,(1\,,\,3)\,,\,(2\,,\,3)\,,\\(2.2\,,\,3)\,,\,(3\,,\,3)\,,\,(3\,,\,0.75)\end{array}\right\}$$

1. *Tracer les courbes $x(t)$ dans la fenêtre graphique 2*
2. *Tracer les courbes $y(t)$ dans la fenêtre graphique 3*
3. *Tracer le portrait de phase $(x(t),y(t))$ dans la fenêtre graphique 4*

Exercice 12 *On considère le problème de Cauchy*

$$\begin{cases} \frac{dx}{dt} & = x+y-x^3 \quad t\in[0,15] \\ \frac{dy}{dt} & = -0.5x \\ x(0) & = x_0 \\ y(0) & = y_0 \end{cases}$$

et la méthode de Runge Kutta définie pour l'équation différentielle $\frac{dx}{dt}=f(t,x)$ *par*

$$x(t+h)=x(t)+\frac{11}{96}K_1+\frac{7}{24}K_2+\frac{35}{96}K_3+\frac{7}{48}K_4+\frac{1}{12}K_5$$

avec

$$\left\{\begin{array}{ll} K_1 & = hf(t\,,\,x(t)) \\ K_2 & = hf\big(t+\frac{2}{7}h\,,\,x(t)+\frac{2}{7}K_1\big) \\ K_3 & = hf\big(t+\frac{4}{7}h\,,\,x(t)-\frac{8}{35}K_1+\frac{4}{5}K_2\big) \\ K_4 & = hf\big(t+\frac{6}{7}h\,,\,x(t)+\frac{29}{42}K_1-\frac{2}{3}K_2+\frac{5}{6}K_3\big) \\ K_5 & = hf\big(t+h\,,\,x(t)+\frac{1}{6}K_1+\frac{1}{6}K_2+\frac{5}{12}K_3+\frac{1}{4}K_4\big) \end{array}\right.$$

Après la résolution du problème de cauchy avec un pas de temps $h = 0.05$ *et pour les couples*

$$(x_0, y_0) \in \left\{\begin{array}{l} (-2\,,\,-2)\,,\,(-1.5\,,\,1)\,,\,(-0.5\,,\,-1.5)\,, \\ (-0.75\,,\,0)\,,\,(0.25\,,\,0)\,,\,(0.5\,,\,0.5)\,, \\ (1.5\,,\,-0.5)\,,\,(1.5\,,\,0)\,,\,(2\,,\,2) \end{array}\right\}$$

1. *Tracer les courbes* $x(t)$ *dans la fenêtre graphique 2*
2. *Tracer les courbes* $y(t)$ *dans la fenêtre graphique 3*
3. *Tracer le portrait de phase* $(x(t), y(t))$ *dans la fenêtre graphique 4*

Conclusion

Scilab induit une nouvelle approche dans le traitement des problèmes. Il faut penser vecteur et matrice dans la modélisation des donnés à manipuler. C'est un mode de pensée auquel nous ne sommes pas traditionnellement habituées.

Comme tout outil informatique la maîtrise des fonctionnalités de Scilab passe par un apprentissage assidu et son utilisation régulière, disons au quotidien. Une fois le problème d'apprentissage réglé, Scilab peut, en contre-partie, permettre de traiter en quelques instructions simples, une question qui peut nécessiter par exemple plusieurs dizaines lignes de code C ou Fortran.

Bibliographie

[1] B. Ycart, *Démarrer en Scilab*, Université René Descartes, Paris.

[2] B. Pinçon, *Une introduction à Scilab*, Université Henri Poincaré, Nancy.