Présentation de Scilab

Anne-Marie Aebischer
Françoise De Labachelerie
IREM de Franche-Comté

18 octobre 2013

Scilab est un lobiciel de calcul numérique librement téléchargeable à l'adresse `http://www.scilab.org/`. Ce document, largement inspiré du tutoriel *Scilab pour les vrais débutants*[1] propose quelques conseils pour une prise en main rapide du logiciel Scilab orientée vers le traitement d'images numériques.

1. Scilab

a) Démarrer Scilab

Lorsqu'on ouvre Scilab, une fenêtre nommée **Console Scilab** apparaît.
Dans la console, il suffit de saisir une commande après l'invite de commandes $-->$ et d'appuyer sur la touche « Entrée » du clavier pour obtenir le résultat correspondant.

Exemple : la commande $4+3$*, suivie de « Entrée » donne « ans=7 ».*

En ajoutant **;** à la fin d'une ligne, le calcul est effectué, mais le résultat ne s'affiche pas. On ne place qu'une instruction par ligne, ou alors on les sépare par un ;.
Une instruction qui a été exécutée ne peut être modifiée. Si on souhaite l'exécuter à nouveau après lui avoir apporté quelques modifications, on peut faire un copier-coller sur une nouvelle ligne ou utiliser, en positionnant le curseur sur une nouvelle ligne, la touche **Flèche vers le haut** qui fait réapparaître successivement les instructions précédentes.

L'instruction **Effacer la console** du menu Préférences permet de récupérer une page blanche.

Lorsqu'on souhaite rédiger et tester une suite d'instructions complexe (un programme !), il est conseillé d'ouvrir, pour travailler une fenêtre nommée « **Scinotes** » qui sert d'éditeur de textes. Pour ouvrir une fenêtre Scinotes, on peut cliquer sur l'icône de gauche du menu représentant une feuille blanche ou bien choisir Scinotes dans le menu Applications.

b) Variables

Les **variables** n'ont pas besoin d'être déclarées à l'avance, toutefois, toute variable doit avoir une valeur. On peut donner à une variable n'importe quel nom non utilisé par Scilab. Attention toutefois au cas des matrices : il ne faut pas utiliser de notation incluant une apostrophe (qui a une signification particulière dans ce contexte). Si on ne donne pas de nom à une variable, sa valeur est affectée à la variable **ans**.
L'affectation d'une valeur se fait par le signe égal. **L'affichage** de la valeur d'une variable se fait en écrivant le nom de cette variable.

La fonction **disp( )** permet l'affichage d'une valeur ou d'un texte. Dans ce dernier cas, il faut entourer le texte de guillemets.

1. `http://www.scilab.org/fr/community/news/20130214`

c) Vecteurs, matrices

Les **vecteurs lignes ou colonnes** sont considérés comme des matrices et s'écrivent entre crochet. Les valeurs sur une même ligne sont séparées par des virgules ou des espaces et le point virgule indique le passage à la ligne suivante.

–> A=[1 3 -5 ;7 6 2.4] *enregistre la matrice* $A = \begin{pmatrix} 1 & 3 & -5 \\ 7 & 6 & 2,4 \end{pmatrix}$.

–> A(2,3) appelle l'élément de la deuxième ligne et de la troisième colonne : 2,4.

L'opérateur : sert à désigner toutes les lignes ou toutes les colonnes d'une matrice.

- –> A(2, :) donne le résultat 7. 6. 2,4.

Pour obtenir la **transposée** d'une matrice ou d'un vecteur, on utilise l'apostrophe '.
Les opérations * et / sont les opérations matricielles. On peut les transformer en des opérations éléments par éléments en les faisant précéder d'un point : **.*** et **./**.
M étant une matrice et p un réel, l'instruction M+p ajoute p à chaque éléments de la matrice M.

La fonction **length** retourne le nombre de coordonnées d'un vecteur. La fonction **size** retourne la taille d'une matrice sous forme d'un vecteur.

Pour l'exemple précédent –>size(A) retourne 2. 3.. Si on nomme le vecteur par l'instruction –> T=size(A), on pourra ensuite faire invenir T(1) ou T(2), première ou deuxième composante du vecteur T.

d) Créer une fonction

Scilab possède une liste de fonctions prêtes à être utilisées. On peut aussi créer ses propres fonctions sur le modèle suivant :

function [valeurs renvoyées]=<nom>(arguments d'entrée)

Une fonction peut avoir plusieurs arguments d'entrée et renvoyer plusieurs valeurs.
Exemple :

```
function res=normes(x,y,z)
res(1)=abs(x)+abs(y)+abs(z)
res(2)=sqrt(x^2+y^2+z^2)
endfunction
```

Dans cette écriture, le nom de la fonction est normes. Les variables d'entrées x, y et z sont des réels (au vu du traitement qui leur est ensuite appliqué). La valeur renvoyée est res, mais la première ligne ne donne pas de renseignement sur sa nature.
Le calcul res(1) et res(2) qui est opéré ensuite montre que res est un vecteur de dimension 2.
La fonction normes renvoie donc le couple $(|x|+|y|+|z|, \sqrt{x^2+y^2+z^2})$ *qui donne les normes 1 et 2 du vecteur* (x, y, z).

e) Programmer avec Scilab

La structure de boucles la plus simple pour un nombre connu d'itérations s'écrit : **for... end**.

L'instruction **i=1 :10** permet de définir un compteur i (en fait un vecteur) qui parcourt tous les entiers entre 1 et 10. On peut préciser le pas de l'incrémentation en écrivant par exemple **i=1 :2 :10**. Dans ce cas, la suite de nombre obtenue est 1, 3, 5, 7, 9. L'incrémentation peut être négative.

On peut aussi utiliser une boucle **while ...end**.

On peut opérer les **tests** suivants :
== **Égal** ; <> **Différent** : < **Inférieur** ; > **Supérieur** ; <= **Inférieur ou égal** ; >= ; %T **Vrai** ; %F **Faux** ; & **Et** ; | **Ou** ; ~ **Non**

Autre structure classique : **if . . .then . . .else . . .end**. If et then doivent être écrits sur une même ligne.

2. Module SIVP (Scilab Image and Processing Video)

Le module SIVP permet de traiter des images et des videos. Il donne accès, en particulier, aux fonctions **imread**, **imshow** et **imwrite** liées aux traitement des images. Il n'est pas installé systématiquement au moment où l'on télécharge Scilab, et pour l'installer, on peut :
– ou bien choisir le menu : Applications ; Gestionnaire de modules - ATOMS ; sélectionner SIVP et cliquer sur le bouton installer ;
– ou bien écrire dans la console : atomsinstall SIVP.
On redémarre ensuite Scilab, et le module sera alors définitivement chargé : à chaque démarrage, on verra écrit dans la console « Start SIVP . . . ».

a) Les images dans Scilab

Une **image numérique en noir et blanc**, c'est-à-dire en 256 teintes de gris, est représentée par une matrice $n * p$ dont les nombres sont des éléments de $\mathbb{Z}/256\mathbb{Z}$. Chaque nombre de la matrice correspond à la teinte d'un pixel de l'image. 0 correspond au noir, 255 au blanc.

Une image numérique en couleur, est caractérisée par la teneur en Rouge, Vert et Bleu de ses pixels (codage RGB). Elle est représentée par une hypermatrice $n * p * 3$, c'est à dire 3 matrices $n * p$ qui correspondent chacune à l'une des trois couleurs. Pour chaque matrice, les nombres sont des éléments de $\mathbb{Z}/256\mathbb{Z}$.

b) Lecture de la matrice d'une image

La fonction **imread** permet de faire afficher la matrice associée à une image donnée.
Le nom du fichier est entre apostrophes, on écrit tout le « chemin », sans oublier l'extension.

Exemple : si l'image s'appelle Musiciens21x32-gris*, sauvegardée au format png, dans le dossier Photos, lui-même sur une clé USB, cette clé étant connectée au port E de l'ordinateur, en écrivant :* G=imread('E :\Photos\Musiciens21x32-gris.png')*, on obtient une matrice G de 21 lignes et 32 colonnes.*

Si l'image est en couleur, la fonction imread va renvoyer une hypermatrice à 3 dimensions. Ces matrices donnent, pour l'ensemble des pixels, les valeurs de Rouge (matrice (:, :,1)) , de Vert (matrice (:, :,2)) et de Bleu (matrice (:, :,3)). Pour un triplet donné, le dernier terme indique la couleur concernée, les 2 premiers termes indiquent de façon usuelle la position du terme dans la matrice choisie.

c) Création d'une matrice d' image

Image en noir et blanc Pour créer une image en noir et blanc, il suffit donc de se donner une matrice dont les termes sont des éléments de $\mathbb{Z}/256\mathbb{Z}$, ou leurs représentants, c'est-à-dire des entiers entre 0 et 255. Pour créer une image en couleur, il suffit de créer de même 3 matrices de même taille constituant l'hypermatrice de l'image.
Sans précaution, ce système ne fonctionne pourtant pas :

La ligne –> M=[12 140 213 67 157; 63 230 145 36 169]

renvoie $M=\begin{pmatrix} 12. & 140. & 213. & 67. & 157. \\ 63. & 230. & 145. & 36. & 169. \end{pmatrix}$

La visualisation de cette matrice (voir plus loin), donne une image blanche.

Le problème vient de la façon dont Scilab traite les nombres.
Les nombres entiers compris entre 0 et 255 devraient être enregistrés au format simple sur 8 bits ($256 = 2^8$). Or, par défaut Scilab écrit les nombres réels au format « double », sur 64 bits.
Le logiciel a donc enregisté une matrice de nombre réels et ne reconnaît pas qu'il s'agit d'une matrice d'image.
Pour que Scilab reconnaisse que la matrice est une matrice d'image, nous pouvons transformer les nombres en éléments de $\mathbb{Z}/256\mathbb{Z}$, en utilisant la fonction **uint8** :

–> M=uint8(M)

Cela permet de réaffecter à M des nombres (au format simple) considérés comme des entiers entre 0 et 255.

On pourrait aussi faire reconnaître par Scilab comme une matrice d'images une matrice dont les éléments sont des nombres compris entre 0 et 1, écrits au format « double ».
Nous avons fait ce choix du format « uint8 » d'une part parce que lorsque l'on lit (avec la fonction « imread ») la matrice d'une image, c'est dans le format « uint8 » qu'elle est affichée ; et d'autre part parce que cela nous paraît préférable pour comprendre l'aspect discret du codage des couleurs.

Il faudra donc être attentif au type de nombre sur lesquels on travaille, et pour calculer k*M, on commencera par convertir la matrice en une matrice de nombres au format « double », et on reviendra ensuite à une matrice d'image avec la fonction uint8.

Image en couleurs Pour créer une image en couleur, il faut donc créer (dans les conditions numériques évoquées précédemment) une matrice à 3 dimensions, c'est à dire un ensemble de triplet (i, j, k), où k prend les valeurs 1, 2 et 3 et où $1 \leqslant i \leqslant n$ et $1 \leqslant j \leqslant p$ si l'image a la dimension $n * p$ (lignes x colonnes).