TP2 : Scilab et Approximation de fonctions

Ce TP sera noté : à la fin de TP, envoyez un mail à `tibor.stanko@inria.fr`. N'oubliez pas d'inclure vos noms et 1-2 images commentées pour chacune des exercices 4 et 5.

1 Prise en main de *Scilab*

Lorsque vous cherchez des renseignements sur une commande ou fonction Scilab prédéfinie, utilisez la commande `help` suivi du nom de la fonction recherchée.

```
// ceci est un commentaire
// exemple : documentation de commande zeros'
help
```

Une bonne référence en français pour découvrir Scilab est le pdf "Scilab pour les vrais débutants" disponible en ligne [1].

1.1 Interface

Scilab est essentiellement une grande calculatrice. Il y a deux modes principales pour executer le code :

1. **la console**, utile pour tester des commandes rapidement ;
2. **l'éditeur SciNotes**, utile si on veut sauvegarder une séquence des commandes dans un fichier. Scilab utilise les extensions `.sce` (scripts) et `.sci` (fonctions). On va utiliser que des `.sce` dans ce TP.

Remarque 1. *Si l'éditeur SciNotes n'est pas visible dans l'interface, on peut le démarrer en exécutant la commande `editor` dans la console.*

Remarque 2. *Utilisez la touche `F5` pour sauvegarder et exécuter un fichier `.sce` dans Scinotes.*

Remarque 3. *Pour repositionner les éléments de l'interface (console, editeur, . . .), clicquez sur la barre horizontale (bleue ou noire) et déplacez la flèche de la souris dans la fenêtre souhaitée.*

Quelques commandes utiles :

```
clc    // effacer la console
clear  // supprimer toutes les variables dans workspace
clf    // effacer la figure
```

1.2 Définir des variables

Des variables dans Scilab sont des matrices, des vecteurs (matrices avec une dimension égale à 1) et des scalaires (matrices avec les deux dimensions égales à 1). Les crochets permettent de définir des matrices. Un espace ou une virgule permet de passer d'une colonne à la suivante et un point-virgule d'une ligne à l'autre. Voici quelques examples.

```
a = 0              // un scalaire
b = [1 2 3]        // un vecteur ligne (1x3)
b = [1, 2, 3]      // un vecteur ligne (1x3)
c = [4; 5; 6]      // un vecteur colonne (3x1)
```

```
A = [1 2 3; 4 5 6]  // une matrice 2x3
R = rand(2,3)       // une matrice 2x3 aleatoire, valeurs dans [0,1]
Z = zeros(2,3)      // la matrice 2x3 des 0
O = ones(2,3)       // la matrice 2x3 des 1
I = eye(3,3)        // la matrice 3x3 identite
```

Exercice 1. Testez les commandes pour définir des matrices.

Remarque 4. *Taper le nom d'une variable affiche sa valeur, sauf avec « ; » en fin de commande.*

1.3 Opérateurs arithmétiques

```
// entre matrice et scalaire
A+2
A-2
A.*2
A./2
A.^2
// entre deux matrices, A et R doivent avoir les memes dimensions
A+R
A-R
A.*R
A./R
A.^R
// matrice transposee
Rt = R'
// produit matriciel
A*Rt
// resolution de systeme lineaire A*X = B
B = rand(2,1)
X = A\B
// les commandes suivantes donnent des erreurs
A+b
b+c
A*R
```

Exercice 2. Testez les commandes pour calcul matriciel.

1.4 Accéder aux éléments

Les parenthèses permettent d'accéder aux éléments ou de les modifier.

```
A(1,3)
A(1,3) = 30
```

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

```
A(2,:)   // la deuxieme ligne
A(:,1)   // la premiere colonne
A(\$,:)  // derniere ligne
A(:,\$)  // derniere colonne
```

Remarque 5. *La commande `size(A)` sert à determiner les dimensions de matrice `A`.*

1.5 Échantillonnage

L'opérateur « : » permet de définir des vecteurs de nombres dont les coordonnées sont en suite arithmétique. On donne « la première valeur : le pas : la dernière valeur » (pas forcément atteinte). Si le pas n'est pas mentionné, sa valeur est 1 par défaut.

```
// exemples
0 : 10
0 : 2 :  10
0 : 0.5 : 10
0 : %pi : 10
```

De manière similaire, la commande `linspace(a,b,n)` est utilisé pour avoir une échantillonnage uniforme d'un interval `[a,b]` où `n` est le nombre de valeurs calculées.

```
// echantillonner [0,2*pi] avec 100 valeurs uniformes
linspace(0,2*%pi,100)
```

Exercice 3. Testez les commandes pour accéder aux éléments d'une matrice et les commandes pour échantillonner un interval [a,b].

2 Méthode des moindres carrés

Exercice 4 (Équations normales)**.** On veut approximer un ensemble des données $(x_i, y_i) \in \mathbb{R}^2,\ i = 1, \ldots, N$ par un polynome $p(x)$ de degrée n

$$p(x) = c_0 + c_1 x + \cdots + c_n x^n = \sum_{k=0}^{n} c_k x^k$$

en minimisant la fonction de cout

$$S(c_0, \ldots, c_n) = \sum_{i=1}^{N} \|p(x_i) - y_i\|^2 .$$

Si on définit les matrices

$$\mathbf{J} = \begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^n \\ 1 & x_2 & x_2^2 & \ldots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^2 & \ldots & x_N^n \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{pmatrix}, \quad \mathbf{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix},$$

le S s'écrit sous la forme matricielle comme

$$S(c_0, \ldots, c_n) = (\mathbf{JC} - \mathbf{Y})^\top (\mathbf{JC} - \mathbf{Y}).$$

En dérivant, on obtient

$$\nabla S = \left(\frac{\partial S}{\partial c_0}, \frac{\partial S}{\partial c_1}, \ldots, \frac{\partial S}{\partial c_n} \right) = 2\mathbf{J}^\top \mathbf{JC} - 2\mathbf{J}^T \mathbf{Y}.$$

Pour trouver le minimum, on met $\nabla S = 0$ ce qui nous donne les *équations normales* :

$$\mathbf{C} = (\mathbf{J}^\top \mathbf{J})^{-1} (\mathbf{J}^\top \mathbf{Y}).$$

1. Téléchargez et examinez le fichier `exo4.sce` disponible sur le site du TP [2].
2. Implementez la résolution des équations normales. Pour cela, il suffit de remplir les colonnes de la matrice `J` ; tout le rest est déjà fourni dans le code.
3. Testez avec l'Ensemble 1 (données de temperature mondiale 1880-2016) et varier le degré de polynome (variable `deg` dans le code). À votre avis, quel degré est nécessaire pour avoir une bonne approximation ?

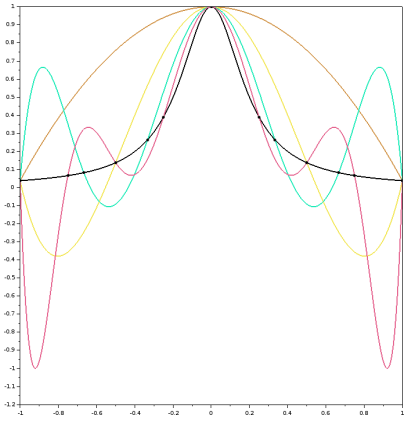
   Compte rendu : inclure un image pour le degré que vous trouvez satisfaisant, avec un commentaire.
4. Question 3 pour l'Ensemble 2 (fonctions analytiques – exp, sin).

   Compte rendu : inclure un image pour une des fonctions et pour le degré que vous trouvez satisfaisant, avec un commentaire.

3 Interpolation

Exercice 5 (Phénomène de Runge).

1. Téléchargez et examinez le fichier `exo5.sce` disponible sur le site du TP [2].
2. Complétez le code : il faut d'abord evaluer la fonction de Runge sur l'abscisse `xp`, puis calculer le polynome `P` qui interpole les points `(xp,yp)` – utilisez les fonctions `Runge` et `Lagrange`. Puis, faitez un `plot` de points `(xp,yp)` et de la fonction `P(x)` – pour cela, il faudra échantillonner l'interval `[-1,1]` de nouveau avec la commande `linspace`, puis appeler la commande `horner(P,x)`.
3. Variez la taille d'abscisse `xp` (variable `n`) et observez le phénomène de Runge.



Compte rendu : inclure un image similaire à la graphe à droite.

Références

[1] Scilab pour les vrais débutants.
http://www.scilab.org/fr/content/download/849/7897/file/Scilab_debutant.pdf

[2] https://tiborstanko.sk/teaching/analyse-num-2017/