TD machine 2 - Mose 1003

E. Ringeisen

Octobre 2015

Tracé de courbes en Scilab

Mise en route

Démarrez le programme Scilab et ouvrez l'éditeur SciNotes comme on l'a vu au premier TD. Sauvegardez un nouveau fichier de commandes dont le nom se termine par `.sce` dans votre dossier `TPmose1003`. Les premières lignes de ce fichier seront de la forme

```
// Blanc-Sec Adèle 00112233
// Hochet Ric 33221100
clear;
xdel(winsid());
```

Cette dernière instruction sert à détruire toutes les fenêtres graphiques existantes entre deux exécutions du fichier. L'instruction `clear` servant quant à elle à détruire toutes les variables existant dans l'environnement d'exécution.
Ajoutez les commandes

```
fenetre = figure("Figure_name", "Une première figure",..
                           "position", [100, 50, 1000, 600]);
fenetre.BackgroundColor = name2rgb('aliceblue')/256;
set("current_figure", fenetre);
```

qui permettent de créer une première fenêtre graphique (une *figure* dans le jargon scilab), en précisant son titre et sa position sur l'écran. Cette position est donnée par une matrice ligne de 4 entiers qui sont le nombre de points de l'écran à l'ouest et au nord de la fenêtre, et la largeur et la hauteur de la (partie graphique de la) fenêtre en pixels également (nombre de points sur l'écran). On voit également comment donner une couleur de fond à la fenêtre en modifiant sa propriété `BackgroundColor`. On a choisi ici la couleur nommée `'aliceblue'`. La liste des couleurs nommées connues de scilab se trouve sur la page http://help.scilab.org/docs/5.5.2/fr_FR/color_list.html Enfin, la troisième instruction dit que cette fenêtre devient la *figure courante* sur laquelle s'effectuent les tracés.

Tracé d'une fonction réelle

Pour tracer la fonction $f(x) = (1-x)\sin(2x)$ sur un intervalle, on utilise la fonction `plot2d()` de scilab. Dans le bloc de code suivant, on définit une matrice colonne `x` qui contient 100 abscisses découpant l'intervalle $[-5, 5]$, et une matrice `y` qui contient les images de ces abscisses par la fonction f. Noter la transposition qui transforme `x` en matrice colonne.

```
x = linspace(-5,  5, 100)';
y = (1-x) .* sin(2*x);
plot2d(x, y, style=[color("forestgreen")]);
```

Par un second appel à `plot2d()`, ajoutez sur le même graphique la courbe représentative de $g(x) = \sqrt{e^x}\cos(x)$ (on pourra réutiliser la même colonne `x`).

Stylage du tracé

L'objet scilab contenant les courbes s'appelle un système de coordonnées (*axes* en jargon scilab). On peut obtenir le système de coordonnées courant comme ceci

```
coor = get("current_axes");
```

Il peut servir à **donner un titre au dessin**, avec une certaine taille des caractères par exemple

```
coor.title.text = "Quelques courbes";
coor.title.font_size = 3;
```

Pour **mettre une couleur de fond**, on utilisera

```
coor.filled = "on";
coor.background = color("seashell1");
```

On peut également mettre un **titre sur l'axe** des x et sur l'axe des y en modifiant les propriétés de `x_label` et `y_label`

```
coor.x_label.text= "Axe des abscisses";
coor.x_label.font_size = 2;
coor.x_label.font_color = color("dark violet");

coor.y_label.text= "Axe des ordonnées";
coor.y_label.font_size = 2;
coor.y_label.font_color = color("dark violet");
```

On peut **positionner un axe** pour le faire passer par l'origine en modifiant la propriété `x_location` (dont les valeurs possibles sont `"bottom"`, `"top"`, `"middle"` et `"origin"`)

```
coor.x_location = "origin";
coor.x_label.position = [4, -0.5];
```

Enfin, on peut **ajouter une grille** par dessus le dessin si on veut

```
coor.grid = [-1, color("blue"), -1];
```

Les 3 valeurs sont les couleurs de la grille selon les 3 axes Ox, Oy et Oz, bien que l'axe Oz n'intervienne pas dans un dessin 2D. Une valeur de -1 indique l'absence de grille dans cette direction. Testez différentes valeurs pour voir l'effet produit.

Zoom

Un clic `Outils->Zoomer` de la fenêtre graphique déclenche un outil permettant en deux clics de sélectionner une zone rectangulaire du dessin sur laquelle on peut zoomer.

Divisions d'une fenêtre

Diviser une fenêtre permet de réaliser plusieurs tracés sur la même fenêtre. Commençons par créer cette deuxième fenêtre, en la décalant un peu sur l'écran par rapport à la première

```
fenetre2 = figure("Figure_name", "Deuxième figure", "position", [150, 100, 1000, 600]);
set("current_figure", fenetre2);
```

Subplot

L'instruction `subplot(2,2,1)` ci-dessous permet de sélectionner comme zone de dessin courante la première d'une série de 4 zones organisées en 2 lignes et 2 colonnes dans la fenêtre courante.

```
subplot(2,2,1);
coor = get("current_axes");
coor.filled= "on";
coor.background= color("seashell1");
```

Comme on le voit à l'exécution, le système de coordonnées obtenu ne concerne que le quart de la fenêtre.
On calcule ici deux matrices colonnes qui sont les valeurs des dérivées $f'(x)$ et $f''(x)$ pour la même matrice `x` que ci–dessus (on a calculé les expressions des dérivées à la main)

```
valeursA = 2 * (1 - x) .* cos(2*x) - sin(2*x);
valeursB = - 4 * (1 - x) .* sin(2*x) - 4 * cos(2*x);
```

Un seul appel de `plot2d()` permet de tracer simultanément les deux courbes

```
plot2d(x, [valeursA, valeursB],..
            style=[color("darkgreen"), color("orchid")],..
            rect=[-3, -15, 3, 10]);
```

On note le nouveau paramètre `rect` de `plot2d` qu'on a utilisé ici. Il permet de définir un rectangle de coordonnées pour la zone qu'on veut tracer, sous forme d'une matrice ligne `[xmin, ymin, xmax, ymax]`. Ici on verra donc l'intervalle $[-3, 3]$ en abscisses et l'intervalle $[-15, 10]$ en ordonnées.

Légende

La fonction `legend()` permet d'ajouter une légende sur le dessin. On passe une chaine de caractères pour chaque courbe.

```
legend("Dérivée", "Dérivée Seconde");
```

(attention, si on voulait mettre f' et f'' comme légende, il faudrait écrire `legend("f''", "f''''")` en doublant les primes)

Stylage supplémentaire des courbes

Une fois les courbes tracées, on peut les styler davantage de la façon suivante

```
courbe = coor.children(2).children(2);
courbe.line_style = 5; // un entier de 0 à 10
courbe.foreground = color("blue");
courbe.thickness = 2;
```

En effet, il y a une hiérarchie des éléments graphiques dessinés : la fenêtre est le parent du système de coordonnées, qui admet lui même comme enfants la légende qu'on a tracé et une composition dont les enfants sont les courbes. On a ici changé de style de ligne en mettant des pointillés, ainsi que la couleur et l'épaisseur du trait.

Définir des fonctions Scilab

Considérons deux points $A(t, \cos(5t))$ et $B(\theta t, \cos(5\theta t))$ qui se déplacent dans le plan. Ici, $t \in \mathbb{R}$ est le temps, et $\theta \in [0, 1]$ est un nombre fixé, par exemple $\theta = 0.4$. La distance entre A et B à l'instant t est

$$D(t) = \sqrt{(1-\theta)^2 t^2 + (\cos(5t) - \cos(5\theta t))^2}$$

On peut appeler A le lièvre et B la tortue.
Il est parfaitement possible de définir dans Scilab une fonction qui calcule la distance entre les deux points à l'instant t. Pour cela, on utilise la syntaxe `function ... endfunction` comme ceci

```
theta = 0.4

function d = distanceLT(t)
    d = sqrt(((1-theta)*t).^2 + (cos(5*t)-cos(5*theta*t)).^2);
endfunction
```

Ici, `distanceLT` est le *nom* de la fonction, `d` est la *valeur calculée* et `t` est le *paramètre* (il pourrait y avoir plusieurs paramètres, séparés par des virgules). Entre la ligne `function ...` et la ligne `endfunction`, on peut mettre des instructions scilab, l'une d'entre elle fixant la valeur calculée. Un appel de fonction tel que

```
distanceLT([1 3 4])
```

donne alors la distance entre les deux points aux instants 1, 3 et 4 secondes. On peut tracer la courbe représentative de cette fonction dans la fenêtre courante. Exécutez et commentez le morceau de code suivant

```
subplot(2, 2, 2);
x = linspace(0, 4, 200)';
fplot2d(x, distanceLT);

coor = get("current_axes");
coor.filled= "on";
coor.background= color("white");
coor.x_label.text= "Secondes";
coor.y_label.text= "Mètres";
legend("dist lièvre-tortue", pos = [2.5, 0.4]);
```

On a utilisé ici une variante de `plot2d()` nommée `fplot2d()` qui permet d'utiliser une fonction comme deuxième argument au lieu d'une matrice. La fonction est évaluée sur chacune des abscisses contenues dans la matrice colonne `x`.

Votre tracé supplémentaire

On montre assez facilement que les vitesses du lièvre et de la tortue à l'instant t sont

$$V_L = \sqrt{1 + (5\sin(5t))^2} \qquad \text{et} \qquad V_T = \sqrt{\theta^2 + (5\theta\sin(5\theta t))^2}$$

1. Définissez deux fonctions scilab nommées `vitesseL()` et `vitesseT()` qui calculent ces vitesses à tout instant t.

2. Dans un troisième subplot, effectuez un appel de la forme `plot2d(vitesseL(x), vitesseT(x));` pour réaliser un tracé où la vitesse du lièvre est en abscisse et la vitesse de la tortue en ordonnée.

3. A l'aide de `plot2d`, tracer sur le même graphique le segment de droite liant les points $(0, 0)$ et $(6, 6)$ du plan. Utiliser l'argument `rect = [0.8, 0.3, 5.5, 2.3]` et indiquer son effet en commentaire.

4. Modifiez votre code pour que le trait de la courbe soit en rouge et ajoutez des titres sur les axes.