



Département  
Licence

K1BE6W14      Biomodélisation  
Mathématiques    TP machine 1  
Ph. Thieullen

## TP 1 : Prise en main du logiciel Scilab (Lecture du document mathL3MatScilabFINAL)

### 1 Démarrer Scilab

Avant toute utilisation d'un logiciel de calcul scientifique, il est nécessaire de mettre en place un environnement fonctionnel de travail.

- Créez un répertoire K1BE6W14 puis copiez le lien du logiciel Scilab dans ce répertoire.

- Modifiez les propriétés du lien avec un clic droit : -> Propriétés -> Raccourci puis supprimez le chemin d'accès du champ Démarrer dans ou cible. Cliquez sur Appliquer puis quittez par OK.

- Ouvrez le logiciel Scilab depuis le répertoire K1BE6W14. Dans la fenêtre console, exécutez quelques opérations sur les fichiers :

- `ls` ou `listfiles()` : lister des fichiers

- `pwd()` : afficher le chemin du répertoire courant

- `chdir(...)` : changer de répertoire

- Il est important de savoir rechercher des informations dans l'aide en ligne : cliquer sur le bouton du menu  puis recherchez comment utiliser `chdir`

- La fenêtre de console ne sert qu'à exécuter des commandes générales. Il est préférable de travailler sur un fichier script distinct. Avec le menu Applications -> SciNotes, ouvrez un script que l'on sauvegardera par exemple sous le nom TP01.sci Vérifiez sur la console en tapant `ls()` que le script est bien dans votre répertoire de travail.

- Chaque TP comporte plusieurs problèmes. Il sera commode d'arranger tous les scripts de la manière suivante. Tapez dans le script les lignes ci-dessous puis revenez sur la console et tapez `exec('TP01.sci')`. Constatez que la fonction `disp()` a affiché du texte. En fait `function prob01() ... endfunction` est un sous-programme qui ne fait rien si on ne l'appelle pas. C'est l'instruction `prob01()` qui appelle ce sous-programme. Le script ne fait rien non plus si on l'exécute pas ; c'est l'instruction sur la console `exec('TP01.sci')` qui exécute ce script.

```
// Numero du TP, Nom de l'etudiant, prenom, ...
function prob01()
    disp('Probleme 01 :');
endfunction
```

```

function prob02()
    disp('Probleme 02 :');
endfunction

prob01();
//prob02();

```

Constater que la première ligne du script (commençant par //) est une ligne de commentaire qui n'est pas exécutée. Constatez aussi que toute chaîne de caractères se met entre deux apostrophes '...'. Revenez sur la console et utilisez la touche  $\uparrow$  : constatez qu'elle rappelle la commande précédente. On pourra aussi utiliser la touche  $\rightarrow$  pour compléter des noms de fichiers ou de répertoire sur la console.

## 2 Graphique 1D

Scilab permet de visualiser très simplement des résultats numériques. On cherche à tracer les graphes de fonctions usuelles. Ces fonctions peuvent par exemple modéliser le nombre d'individus  $N_t$  en fonction du temps  $t$ . On demande de tracer sur la même figure les 6 graphes des fonctions suivantes

$$\begin{array}{ll}
 N_t = k_1 e^{k_2 t}, & \text{(exponentiel)} \\
 N_t = k_1 t^{k_2}, & \text{(puissance)} \\
 N_t = k_1 t / (k_2 + t), & \text{(saturation)} \\
 N_t = k_1 / (1 + k_2 \exp(-k_3 t)), & \text{(Richards)} \\
 N_t = k_1 t^{k_2} (1 - t^{k_3}), & \text{(Blumberg)} \\
 N_t = k_1 t^{k_2} \exp(-k_3 t), & \text{(maximum)}
 \end{array}$$

Ecrivez le code suivant (dans `function prob01() ... endfunction`) pour la première fonction en le complétant pour les autres fonctions. `clf()` efface tous les graphiques précédents, `subplot(x,y,z)` partage la figure en  $x*y$  sous-graphiques avec  $x$  lignes,  $y$  colonnes et  $z$ , un numéro pour chaque graphique. `t = 0:0.01:50;` crée un vecteur (horizontal) entre  $[0.1, 50]$  de pas 0.01. Constatez dans `N_t1` que les opérations arithmétiques sont faites terme à terme. `N_t1'` transpose verticalement `N_t1` et `[N_t1', N_t2', N_t3']` crée une matrice avec comme colonne `N_t1'`

```

clf()
t = 0.1:0.01:50;
subplot(2,3,1)
k1 = 10; k2 = -0.1; N_t1 = k1*exp(k2*t);
k1 = 10; k2 = -0.5; N_t2 = k1*exp(k2*t);
k1 = 0.1; k2 = 0.1; N_t3 = k1*exp(k2*t);
plot(t, [N_t1', N_t2', N_t3'])
legend(['k1=10 ; k2=-0.1', 'k1=10 ; k2=-0.5', 'k1=0.1 ; k2=0.1'], [5,13])
xstring(5,13.5, 'Exponentiel')
subplot(2,3,2)
...

```

On trouvera la figure 1. Pour le deuxième graphique, on force le domaine des abscisses et des ordonnées par la commande

```
a=gca(); a.data_bounds=[t(1),0;t($),100];
```

Remarquez que  $t(1)$  est la première composante de  $t$  et  $t(\$)$  est la dernière. Consultez l'aide sur `axes_properties` et repérez `data.bounds`. Pour le troisième graphique, Scilab permet de diviser terme à terme des vecteurs avec  $k1*t./(k2+t)$ . Remarquez l'opérateur `./` dans l'expression. On aura besoin des opérateurs `^` ou `*` qui agissent aussi terme à terme.

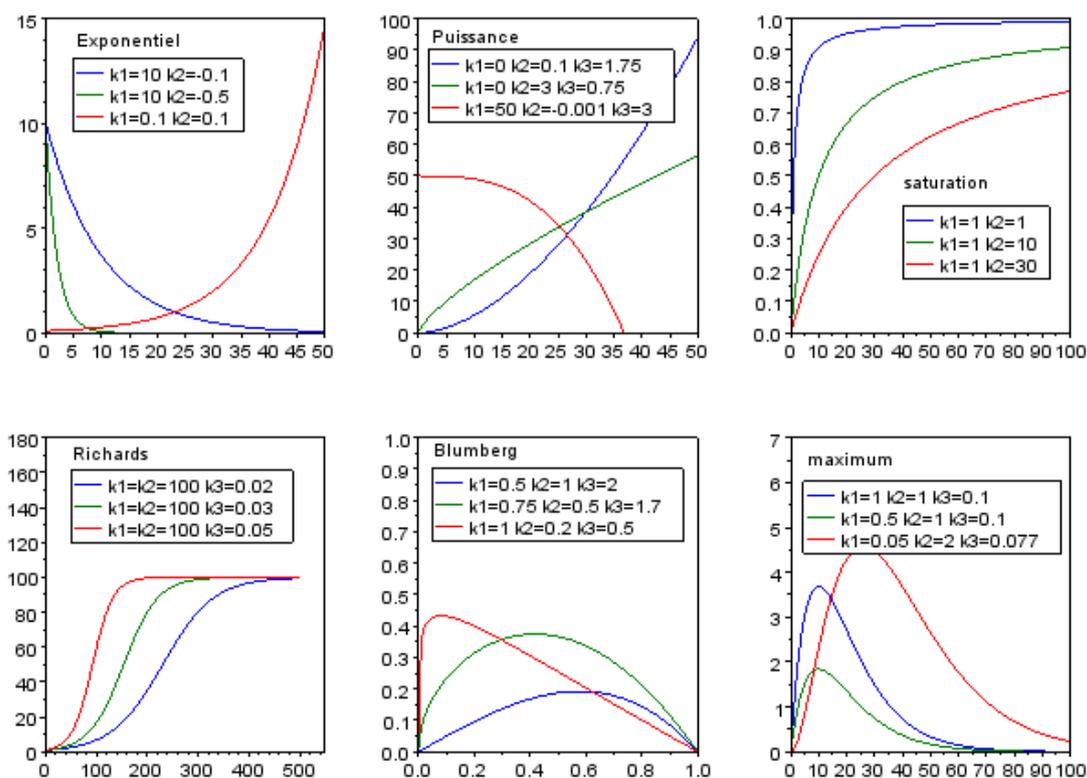


FIGURE 1 – Quelques fonctions usuelles

### 3 Graphique 2D

On cherche à représenter une distribution aléatoire en espace d'individus (par exemple, une distribution d'arbres dans une forêt). Commencez par regarder l'aide en ligne de la fonction `grand()`. Puis tapez le code suivant (dans `function prob02() ... endfunction`).

```
rand('seed',getdate('s'))
```

```
x = grand(1000,1,'nor',0,3);
y = grand(1000,1,'nor',0,3);
xx = x+2*y; yy = x+y;
clf()
subplot(1,2,1)
plot(x,y,'.k','MarkSize',3)
xlabel('X et Y non correlees')
a = gca(); a.isoview = 'on'; a.data_bounds = [-10,-10;10,10];
subplot(1,2,2)
//idem pour plot(xx,yy, ...)
```

On trouvera la figure 2. Remarquez l'utilisation de `a.isoview = 'on'` qui permet d'obtenir une figure où l'unité de longueur est la même pour les deux axes. Remarquer aussi dans la fonction `plot` l'utilisation de l'option `'.k', 'MarkSize', 3` qui permet d'afficher des points `.` de couleur noire `k` et de taille 3. Consultez l'aide en ligne de `plot()` et les options `LineStyle` et `GlobalProperty`.

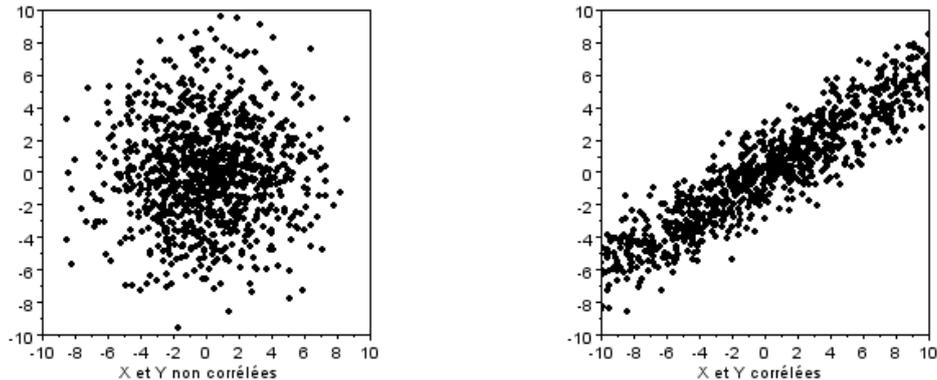


FIGURE 2 – Corrélacion ou non