TP de mathématiques

ISBS

Thomas Richard

11 octobre 2016

1 Déroulement

Votre UE de mathématiques comporte trois séances de travaux pratiques. Le but de ces séances est multiple :

— aborder certaines notions du cours de mathématiques sous un angle plus concret/visuel.
— vous donner une première approche d'un logiciel professionnel de calcul scientifique.
— vous donner un exemple d'utilisation d'un modèle mathématique.

Votre travail sera organisé de la façon suivante :

1. Lors des deux premières séance, vous vous familiarisez avec les fonctions de bases de SCILAB, tout en illustrant certains point du cours de mathématiques.
2. À la fin de la deuxième séance, vous choisissez un binôme un projet sur lequel vous travaillerez.
3. Lors des séances suivantes, vous répartirez votre travail entre exercices dévelloppant certains point du cours et projet.
4. Deux semaines après la dernière séance, vous me rendez par mail un compte rendu accompagné des programmes que vous avez produits.

2 Initiation à SCILAB

SCILAB dispose d'une aide très utile, que vous pouvez lancer soit via le menu aide, soit via l'invite de commande en tapant `help` suivi de la commande sur laquelle vous voulez de l'aide.

2.1 Une grosse calculatrice...

SCILAB est un outil de calcul numérique. Il permet d'effectuer toutes sortes de calculs.

1. SCILAB peut effectuer des calculs simples.
   a) Tapez `2+2`, `exp(1)`, `sin(pi/3)`, `log(2)` dans la console de SCILAB.
   b) Tapez `5e-2`, puis `3.2e3`. Quelle est cette écriture ?
   c) Effectuez à la main et à l'aide de SCILAB le calcul suivant $(1+10^{-20})-1$. Que constatez vous ? *Ce comportement est dû au fait que SCILAB ne travaille qu'avec des valeurs approchées.*
   d) On peut stocker des valeurs dans des variables, voici un exemple : tapez les commades suivantes à la suite : `a=1`, `b=4`, `c=-8`, `d=b^2-4*a*c`. Qu'a-t-calculé ?
   e) Continuez la séquence d'instruction précédente pour trouver les racines du polynôme x^2+4x-8. Vous stockerez les valeurs dans les variables `x1` et `x2`.
2. SCILAB est un logiciel qui bien utilisé permet d'effectuer de façon relativement efficace des opérations sur des tableaux, des vecteurs ou des matrices.
   a) Tapez `u=[4,5,6]`, `v=[1,2,4]`, `w=[1;2;4]`. Quelle est la différence entre `v` et `w` ?
   b) À quoi servent les commandes `u(2)` ? `w(3)` ?
   c) Que renvoie `2*u` ? `u+v` ? `u+w` ?
   d) Que produit selon vous la commande `M=[1,2,3;4,5,6;7,8,9]` ? Vérifiez votre hyptohèse. Testez l'effet de la commande `M(2,3)`.
   e) Construisez les matrices $A=\begin{pmatrix} 3 & \pi \\ -1 & 2 \\ 5 & 10^{-12} \end{pmatrix}$ et $B=\begin{pmatrix} 4 & 5 & 6 \\ -1 & -2 & -3 \end{pmatrix}$.
   f) Exécutez les commandes `A*B`, `A*M`, `M*A`, `M*v`, `M*w`, `v*v`... Lesquelles renvoient une erreur ?
   g) Calculez `y=M*w`, puis `M\y`. Que fait la commande « `\` » ?

h) Réessayez les commandes précédentes en remplaçant « `*` » par « `.*` ».

i) Testez l'effet des opérateurs « `^` », « `.^` », « `/` » et « `./` » sur les matrices et les vecteurs.

j) Décrivez l'effet des commandes suivantes `0:10`, `2:0.5:6`, `1:0.3:5`.

k) Construisez en une seule ligne un vecteur ligne contenant les carrés de tout les multiples de 3 compris entre 0 et 100. Construisez un vecteur ligne contenant les 20 premières puissances de 2.

l) Si `v` est un vecteur, `sum(v)` renvoie la somme de toutes les composantes de `v`. Utilisez cette fonction pour évaluer $\sum_{k=1}^{10^5} \frac{1}{k^2}$.

2.2 Graphiques

SCILAB est capable de créer des représentations graphiques diverses. La commande de base pour les graphiques en 2D est la commande `plot`. Voyons un exemple :

```
X=(0:0.05:1)';
Y1=X;
Y2=X.^2;
Y3=X.^3;
Y4=X.^4;
plot2d(X,[Y1,Y2,Y3,Y4],strf="041");
title('Graphes de fonctions puissances y=x^n');
xlabel('x');
ylabel('y');
legend('n=1','n=2','n=3','n=4');
```

1. Exécutez les commandes ci-dessus. Essayez de comprendre à quoi sert chaque ligne.
2. Créez un graphique représentant les fonction sinus et cosinus sur l'intervalle $[0, 2\pi]$.

Bien des variations sont possibles, on peut en particulier ne mettre que des croix aux points de la courbe (au lieu de les relier), ce qui est utile pour représenter des données issues de l'expérience, ou encore représenter plusieurs graphiques sur la même figure. Expérimentez avec les commandes suivantes :

```
x = 0:0.1:10';
y1 = sin(x);
y2 = sin(2*x);
y3 = sin(4*x);
y4 = sin(8*x);

subplot(2,2,1);
plot2d(x,y1,-1);
title('Subplot 1: sin(x)')

subplot(2,2,2);
plot2d(x,y2,-2);
title('Subplot 2: sin(2x)')

subplot(2,2,3)
plot2d(x,y3,-3);
title('Subplot 3: sin(4x)')

subplot(2,2,4)
plot2d(x,y4);
title('Subplot 4: sin(8x)')
```

SCILAB est aussi capable de représenter graphiquement les fonctions de deux variables de la forme $z = f(x, y)$. Soit sous forme de surface 3D (commande `surf`), soit sous forme de lignes de niveaux (commandes `contour` et `contourf`). Voici un exemple :

```
x=-2:0.2:2;
y=-3:0.2:3;
Xt=x'*ones(1,length(y));
Yt=ones(length(x),1)*y;
Z = Xt.^2 - Yt.^2;
xset("colormap",jetcolormap(16));
subplot(1,2,1);
plot3d1(x,y,Z);
title('Graphe de f(x,y)=x^2-y^2');
subplot(1,2,2);
contour2d(x,y,Z,16,strf="041");
title('Lignes de niveaux de f(x,y)=x^2-y^2');
```

2.3 Programmation

Quand on veut effectuer une suite d'opérations assez longue dans SCILAB, il est utile de pouvoir stocker les suites d'instructions pour pouvoir les réutiliser plus tard. C'est le rôle des scripts. Créez un fichier `script.sci` puis entrez le code ci-dessous dans le fichier :

```
// Ce script illustre les capacités
// de MATLAB en terme de représentation
// graphique des fonctions de plusieurs
// variables.
x=-2:0.2:2;
y=-3:0.2:3;
Xt=x'*ones(1,length(y));
Yt=ones(length(x),1)*y;
Z = Xt.^2 - Yt.^2;
xset("colormap",jetcolormap(16));
subplot(1,2,1);
plot3d1(x,y,Z);
disp('Représentation du graphe');
pause;
title('Graphe de f(x,y)=x^2-y^2');
subplot(1,2,2);
contour(x,y,Z,16,strf="041");
title('Lignes de niveaux de f(x,y)=x^2-y^2');
disp('Représentation des lignes de niveaux');
```

Exécutez le. Remarquez les commentaires (précédés du symbole `%`) et l'usage des commandes `disp` et `pause` pour donner des informations à l'utilisateur.

Les structures de contrôles habituelles en programmation sont disponibles : `if`, `while`, `for`. Voici un exemple de script qui calcule les solution d'une équation du second degré :

```
// Ce script demande à l'utilisateur d'entrer les valeurs des
// coefficients a, b et c et affiche si elles existent les solutions
// réelles de l'équation $ax^2+bx+c=0.
disp('Entrez les valeurs a,b et c dans ax^2+bx+c=0');
a=input('Valeur de a ? ');
b=input('Valeur de b ? ');
```

```
c=input('Valeur de c ? ');
strEq=[num2str(a),'x^2+',num2str(b),'x+',num2str(c),'=0'];
d=b^2-4*a*c;
if d>0
    x1=(-b-sqrt(d))/(2*a);
    x2=(-b+sqrt(d))/(2*a);
    disp('L''équation ');
    disp(strEq);
    disp('a deux solutions :');
    disp(x1);
    disp(x2);
    else if d==0
       x0=-b/(2*a);
       disp('L''équation ');
       disp(strEq);
       disp('a une seule solution :');
       disp(x0);
        else if d<0
             disp('L''équation ');
             disp(strEq);
             disp('n''a pas de solutions réelles.');
             end
        end
end
```

Pour vous entraîner, vous pouvez programmer le "jeu" suivant : le script choisi un entier au hasard qu'il garde secret (utiliser `floor(100*rand(1))`) puis demande à l'utilisateur de le deviner. Si l'utilisateur a trouvé le script affiche 'Gagné', sinon il affiche 'Trop grand' ou 'Trop petit' et redemande un essai à l'utilisateur.

Pour effectuer des tâches qui interviennent à plusieurs moments, on peut utiliser des fonctions. La syntaxe générale pour définir une fonction est :

```
function [y1,...,ym]=mafonction(x1,...,xn)
  // On effectue les
  // calculs nécessaires.
  // Il peut y avoir des if, for, while...
  y1= // On affecte les valeurs de
  ... // sortie au variables
  ym= // de sortie yi
```

```
endfunction
```

— Les `xi` sont les variables d'entrées,
— Les `yi` sont les variables de sorties.
Elles peuvent être des nombres, des vecteurs, des matrices, des chaines de caractères.

Remarque 2.1. Certaines fonctions avancées de scilab, par exemple les fonctions `fsolve`, `ode`, `fminsearch`, demandent comme variable d'entrée des **fonctions**. Il faut lire l'aide pour voir comment les utiliser. Une source d'erreur est souvent que, lorsque l'on veut utiliser ces fonctions pour résoudre des problèmes impliquant des fonctions à *plusieurs variables réelles*, il faut écrire ces fonctions comme des fonctions d'une *seule variable vectorielle.*

Si l'on exécute la fonction en tapant :

```
mafonction[x1,...,xn]
```

seule la valeure `y1` sera renvoyée. Pour obtenir tous les `yi`, il faut exécuter :

```
[y1,...,ym]=mafonction[x1,...,xn]
```

Si on veut écrire une fonction simple qui ne prend qu'une seule variable en entrée et ne renvoie qu'une seule variable, cela devient `function y=mafonction(x)`. Pour pouvoir être exécutée par SCILAB la fonction doit être enregistrées dans un fichier fichier (par exemple `mafonction.sci`). Avant d'utiliser une fonction il faut exécuter le fichier `.sci` contenant sa définition.

Les variables d'entrée et de sortie peuvent être des nombres, mais aussi des vecteurs, des matrices ou des chaînes de caractères.

Donnons quelques exemples de fonctions. La fonction suivante renvoie x^2 si $x > 0$ et 0 sinon :

```
function y=carre_positif(x)
   // Cette fonction renvoie le
   // carré de x si x est positif,
   // et 0 si x est négatif.
   if x>0
      y=x.^2;
      else y=0
   end
endfunction
```

Il est parfois possible de se passer de structures de contrôles (`if`,`for`), par exemple on peut aussi écrire la fonction précédente ainsi :

```
function y=carre_positif2(x)
   // Cette fonction renvoie le
   // carré de x si x est positif,
   // et 0 si x est négatif.
   y=(x>0).*(x.^2)
endfunction
```

Si genre d'écriture peut sembler cryptique il a plusieurs avantage :

- Les opérations `.*` et `.^` fonctionnent naturellement sur les tableaux. Essayez d'exécuter `carre_positif([-1,1,4])` puis `carre_positif2([-1,1,4])`
- La manière dont est écrite la deuxième fonction amène souvent à des temps de calcul plus courts.

La fonction suivante renvoie, étant donnés u_1, u_2 et N le vecteur $(u_1, u_2, \ldots, u_N)$ des $N+1$ premiers termes de la suite de Fibonnacci, définie par la relation de récurrence $u_{n+1} = u_n + u_{n-1}$.

```
function Uvect=fibonacci(u1,u2,N)
   // Renvoie les (N+1) premiers termes
   // de la suite de fibonacci sous formes
   // d'un vecteur.
   Uvect(1)=u1;
   Uvect(2)=u2;
   for i=3:N
     Uvect(i)=Uvect(i-1)+Uvect(i-2);
   end
endfunction
```

Étant donnés un entier N la fonction suivante renvoie le plus grand entier k tel que $2^k \leq N$, ainsi que la valeur 2^k.

```
function [k,p]=disc_log2(N)
  // Logarithme discret en base 2.
  k=0;
  p=1;
  while p<N
    k=k+1;
    p=2*p;
  end
end
```

Il est aussi possible de définir des fonctions simples en une ligne , la commande `deff('y=f(x)','y=x.^2+2.*x+2')` définit la fonction $f(x) = x^2 + 2x + 1$.

2.4 Quelques pistes

SCILAB sais faire beaucoup de choses. Pour évaluer des intégrales, jetez un oeil à `intg` ou à `inttrap`, pour résoudre un système non-linéaire regardez du côté de `fsolve`, pour minimiser une fonction regardez `fminsearch`, pour résoudre une équation différentielles utilisez `ode`. Cependant, avant d'utiliser ces commandes lisez l'aide en ligne et examinez les exemples pour comprendre comment fonctionnent ces commandes.

2.5 Petits exercices

1. Utilisez scilab pour résoudre le système linéaire :

$$\begin{cases} 3x + 4y - z = 5 \\ x + 2y + 3z = -1 \\ x - y - z = 3 \end{cases}$$

2. Tracez les lignes de niveau de la fonction $f(x, y) = (1 - x)^2 + 100(y - x^2)^2$ sur $[-2, 2]^2$. Trouvez son minimum.
3. Résolvez à l'aide de matlab l'équation $x^2 - 2 = 0$.
4. Résolvez numériquement l'équation différentielle $x'(t) = -x(t)+\sin(t)$ avec la condition initiale $x(0) = 1$.
5. Tracez sur un même graphique les lignes de niveaux de $f(x, y) = x^2 - 2y^2$ et son gradient comme champ de vecteurs (regardez l'aide de la fonction `champ`).
6. On s'intéresse au calcul approché de l'inégrale d'une fonction $f : [a, b] \to \mathbb{R}$. On considère pour cela deux méthodes :
   — La méthodes des rectangles :

$$\int_a^b f(x)dx \simeq h \sum_{k=0}^{N-1} f\left(a + k(b - a)h\right)$$

   où $h = 1/N$.

— La méthode des trapèzes :

$$\begin{aligned}\int_a^b f(x)dx \simeq & \frac{h}{2}\sum_{k=0}^{N-1} f\left(a+k(b-a)h\right)+f\left(a+(k+1)(b-a)h\right)\\ =& h\left(\frac{f(a)+f(b)}{2}+\sum_{k=1}^{N-1} f\left(a+k(b-a)h\right)\right)\end{aligned}$$

où $h = 1/N$

Testez ces méthodes sur la fonction exp : $[0,1] \to \mathbb{R}$. Tracez sur un même graphique l'erreur commise en fonction de N pour chacune de ces méthodes. Laquelle est la meilleure ?

7. On cherche ici à remplacer une fonction connue seulement en un nombre fini de points par un polynôme. Sont donnés N points (x_i, y_i) où les x_i sont tous différents et on cherche un polynôme $P(x)$ de degré $N+1$ vérifiant $P(x_i) = y_i$.
   a) Déterminez la matrice de l'application linéaire qui à $(a_0, \ldots, a_N) \in \mathbb{R}^{N+1}$ associe $(P(x_0), \ldots, P(x_N)) \in \mathbb{R}^{N+1}$ où $P(x) = a_N x^N + \cdots + a_0$.
   b) Écrire un programme MATLAB qui donne connaissant les $N+1$ points $(x_i, y_i)_{0 \leq i \leq N}$ calcule les coefficients du polynôme P de degré N tels que $P(x_i) = y_i$.
   c) Testez ce programme pour des points (x_i, y_i) tels que les x_i sont équirépartis de -1 à 1 et $y_i = 1/(1+x_i^2)$, pour $N = 3, 5, 7, 9....$ Que constatez vous ?
8. Soit $f : \mathbb{R} \to \mathbb{R}$ une fonction impaire et 2π-périodique. Ces coefficients de Fourier b_k sont définis par :

$$b_k = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\sin(kt)dt.$$

   La série de Fourier de f est la somme :

$$S_f(t) = \sum_{k=1}^{+\infty} b_k \sin(kt).$$

   a) Écrivez un programme permettant de calculer b_k étant donnés k et f.
   b) Pour la fonction impaire et 2π-périodique égale à 1 sur $[0,\pi]$, représentez sur un même graphique f et $\sum_{k=1}^{N} b_k \sin(kt)$. pour N entre 1 et 5.

c) Même question pour la fonction affine par morceaux, 2π périodique et impaire s'annulant en 0 et en π et valant $\pi/2$ en $\pi/2$.

3 Projets

3.1 Population structurée en âges, modèle de Leslie

On s'intéresse à l'évolution d'années en années d'une population animale divisée en trois catégories : les jeunes (dont la population à l'année n est notée j_n), les adultes (notés a_n) et les viellissants (notés v_n). On fait les hypothèses suivantes :

— Chaque spécimens de l'espèce en question vit au plus 3 ans.

— Au bout d'un an, s'il survit, un jeune devient adulte et un adulte devient vieillissant.

— Le taux de survie des jeunes (resp. des adultes) est noté s_j (resp. s_a).

— Le taux de fécondité des adultes (resp. des viellissants) est noté f_a (resp. f_v).

Sous ces hypothèses, on obtient que si on note P_n le vecteur $\begin{pmatrix} j_n \\ a_n \\ v_n \end{pmatrix}$, alors $P_{n+1} = MP_n$ où M est la matrice définie par :

$$M = \begin{pmatrix} 0 & f_a & f_v \\ s_j & 0 & 0 \\ 0 & s_a & 0 \end{pmatrix}.$$

Pour différentes valeurs des paramètres f_a, f_v, s_j, s_a :

— calculez les 20 premières valeurs des suites j_n, a_n et v_n pour différentes données initiales j_0, a_0 et v_0. Le comportement en temps long de ces suites dépend-il des données initiales ?

— calculez le taux d'accroissement τ_n à l'année n (c'est le quotient de la population totale à l'année n par la population totale à l'année $n-1$).

— calculez les valeurs propres de la matrice M (commande `spec` de SCILAB).

Que remarquez vous ?

On suppose que chaque année on prélève (chasse ou pêche par exemple) une proportion $p \in [0, 1]$ des individus adultes et des viellissant. Comment cela affecte-t-il les paramètres du modèle (attention, on obtient des résultat différents si on suppose que le prélèvement a lieu avant ou après que les animaux se soient reproproduits).

Comment feriez-vous pour déterminer le taux de prélèvement qui permet à la population globale d'être stable ?

On a effectué les comptages suivants (tableau 1) en milieu naturel. On souhaite autoriser la chasse de l'espèce en question pour en stabiliser la population.

Année	0	1	2	3	4	5	6	7	8	9	10
Jeunes	19	91	34	80	65	77	85	87	97	102	115
Adultes	10	4	19	7	17	14	16	17	18	20	22
Vielliessants	22	8	4	16	6	14	11	13	14	15	16

Table 1 – Population mesurée au cours des dix premières années

Estimez les paramètres f_a, f_v, s_j, s_a pour la population considérée, vous pouvez par exemple considérer la fonction $E(f_a, f_v, s_j, s_a) = \sum_k \|P_{k+1} - MP_k\|^2$ et en chercher le minimum.

Déterminez quelle pourcentage des adultes et des viellissants doivent être chassés pour stabiliser la population, on distinguera les cas où la chasse est autorisée avant ou après la période de reproduction.

3.2 Estimation des paramètres dans un modèle logistique

Le but est d'essayer de modéliser l'évolution d'une population par différent modèles. On se basera sur les données de la population mondiale données dans le tableau 2.

Années t_i	1950	1955	1960	1965	1970	1975	1980
Population p_i	2,5	2,7	3,0	3,3	3,6	4,0	4,4
Années t_i	1985	1990	1995	2000	2005	2010	2015
Population p_i	4,8	5,3	5,7	6,1	6,5	6,9	7,3

Table 2 – Population Mondiale de 1950 à 2015 (En milliards, source ONU)

Vous confronterez trois modèles différents.

On commence par supposer qu'il existe une relation affine en t_i et p_i, c'est à dire que $p_i \simeq at_i + b$ (modèle linéaire). Le but du jeu est de trouver les meilleurs a et b possibles. Pour cela, on définit une fonction $E : \mathbb{R}^2 \to \mathbb{R}$ par :

$$E(a,b) = \sum_i (p_i - at_i - b)^2,$$

et on cherche a et b qui minimisent la fonction E. Pour cela on peut :

— procéder graphiquement (en traçant les lignes de niveaux de la fonction E par exemple).

— écrire le système d'équations $\frac{\partial E}{\partial a} = 0$, $\frac{\partial E}{\partial b} = 0$ et le résoudre à l'aide de SCILAB. (La condition préccédente n'est qu'une condition nécessaire pour être un minimum, on pourrait aussi tomber sur un maximum ou un point selle, cependant la forme particulière de la fonction utilisée ici assure que l'on obtiendra un minimum.)

— utiliser la fonction `reglin` de SCILAB.

— utiliser la fonction `fminsearch` de SCILAB.

On regarde ensuite si l'on obtient de meilleurs résultats en cherchant C et α tels que $p_i \simeq Ce^{\alpha t_i}$. Pour cela on peut par exemple observer que la relation ci-dessus est équivalente à $\log(p_i) \simeq \alpha t_i + log(C)$ et procéder comme dans le cas précédent. On remarquera que la fonction $p(t) = Ce^{\alpha t}$ est solution de l'équation différentielle $p' = \alpha p$.

Le dernier modèle que l'on considère est le modèle logistique (dit aussi modèle de Verhulst). Les fonctions logistiques sont les solutions des équations différentielles de la forme $p' = \alpha p\left(1 - \frac{p}{p_{max}}\right)$. Elles sont de la forme :

$$p_{\alpha,p_{max},C}(t) = \frac{p_{max}}{1 + \left(\frac{C}{p_{max}} - 1\right)e^{-\alpha t}}.$$

On peut vérifier, numériquement ou sur papier, que les fonctions $p_{\alpha,p_{max},C}$ sont bien solutions de l'équation différentielle ci dessus.

On cherche une fois de plus à trouver les meilleurs paramètres α, p_{max} et C tels que $p_i \simeq p_{\alpha,p_{max},C}(t_i)$. On pourra définir, en s'inspirant des exemples précédents une fonction « erreur » $E(\alpha, p_{max}, C)$ et chercher les valeurs de α, p_{max} et C qui minimisent cette erreur.

On concluera en comparant les résultats donnés par ces différents modèles en 2100 avec les prévisions officielles de l'ONU pour l'évolution de la population mondiale, dont la variante moyenne donne une population de 11,2 milliards en 2100.

3.3 Modèle proies-prédateurs en écologie

On s'intéresse à l'évolution au cours du temps t de deux populations animales : les proies $(x(t))$ et les prédateurs $(y(t))$.

On suppose qu'en l'absence de prédateurs les proies croissent de façon exponentielle (i.e. $x'(t) = \alpha x(t)$) et les prédateurs décroissent de façon exponentielle (i.e. $y'(t) = -\delta y(t)$).

Pour tenir compte de la prédation, on suppose que la présence de prédateurs contrarie la croissance des proies et on remplace l'équation $x'(t) = \alpha x(t)$ par $x'(t) = (\alpha - \beta y(t))x(t)$. De même la présence de proies favorise la croissance des prédateurs et l'équation pour $y(t)$ devient alors : $y'(t) = (\gamma x(t) - \delta)y(t)$.

On obtient donc les système d'équations différentielles suivant :

$$\begin{cases} x' = (\alpha - \beta y)x \\ y' = (\gamma x - \delta)y. \end{cases}$$

À l'aide de SCILAB simulez et représentez graphiquement les solutions de cette équation différentielle. (On pourra prendre tout les paramètre égaux à 1, et $x(0) = 1$, $y(0) = 3$, $t \in [0, 20]$.) Observez le fait que les solutions sont périodiques.

Évaluez les valeurs moyennes de x et de y, définie par $\bar{x} = \frac{1}{T}\int_0^T x(t)dt$ et $\bar{y} = \frac{1}{T}\int_0^T y(t)dt$.

On suppose que la population est soumise à un prélévement extérieur (pêche ou chasse) indiscriminé. On peut modéliser ce changement par le fait de diminuer la valeur de α et d'augmenter la valeur de δ. Quel effet cela a-t-il sur les population moyennes de prédateurs ?

L'hypothèse que la population de proies croît exponentiellement en l'absence de prédateurs n'est réaliste que si l'espace et la nourriture est disponible en quantité quasi illimitée pour les proies.

Pour obtenir un résultat plus réaliste, on suppose qu'en l'absence de prédateurs, le nombre de proies évolue selon l'équation $x' = \alpha(1 - x/x_{max})x$. Simulez les solutions de cette équation sous MATLAB, que se passe-t-il si la condition initiale $x(0)$ est supérieure à x_{max} ? inférieure à x_{max} ?

En tenant compte de cet effet, le système proies/prédateurs complet devient :

$$\begin{cases} x' = \alpha(1 - x/x_{max})x - \beta yx \\ y' = (\gamma x - \delta)y. \end{cases}$$

Simulez et visualisez les solutions de ce nouveau système et comparez les aux solutions du système précédent. Le caractère périodique des solutions est-il conservé ? Les prédateurs survivent-ils toujours indépendemment de la limitaion du milieu pour les proies ?

3.4 Épidémiologie, modèle SIR

Le modèle SIR est un modèle essayant de décrire la propagation au fil du temps d'une maladie dans une population. Pour cela on divise la population totale en 3 catégories :

- Les « susceptibles », qui n'ont jamais attrapé la maladie, dont la population au temps t est notée $S(t)$.
- Les « infectés », ou malades, notés $I(t)$.
- Les « résistants », qui sont immunisés contre l'infection soit car ils l'ont déjà eu soit car ils ont été vaccinés, notés $R(t)$.

On fait les hypothèses suivantes :

- Plus il y a de susceptibles et d'infectés, plus les susceptibles deviennent infectés.
- Les infectés se soignent à une vitesse proportionnelle à leur nombre.

Ceci se traduit par le système d'équations différentielles suivant :

$$\begin{cases} S'(t) = -\alpha I(t)S(t) \\ I'(t) = \alpha I(t)S(t) - \beta I(t) \\ R'(t) = \beta I(t) \end{cases}$$

où α représente un « taux de contagion » et β une vitesse de guérison. On suppose que $\alpha = 1.4 \times 10^{-5}$ et $\beta = 0.2$.

On suppose qu'au temps $t = 0$, la population est constituée de 50000 individus dont 5 sont infectés et aucun n'est résistant.

Utiliser SCILAB pour visualiser l'évolution de $S(t)$, $I(t)$ et $R(t)$ pour $t \in [0, 100]$.

On dit qu'une épidémie est hors de contrôle à un instant donné si le nombre de d'infectés croit à cet instant. L'épidémie est-elle hors de contrôle dans l'exemple précédent ?

Pour contrôler une épidémie, on peut procéder de plusieurs manières :

- on peut isoler les infectés du reste de la population (en leur donnant un arrêt maladie par exemple), ceci diminue les risques de contagion et donc la valeur de α.

— on peut prescrire des médicaments aux infectés qui accélèrent la guérison et augmentent donc la valeur de β.

— on peut enfin vacciner en amont une partie de la population, ce qui diminue $S(0)$ et augmente $R(0)$.

Pour chacune des ces méthodes, déterminez dans quelle mesure il faut modifier les paramètres pour mettre l'épidémie sous contrôle.

On a mesuré le nombre d'infectés tous les 10 jours pour l'épidémie considérée, on obtient les chiffres suivants :

Temps (en jours)	0	10	20	30	40	50	60	70	80	90	100
Nombre d'infectés	5	50	400	2500	5000	2800	900	300	90	20	7

Table 3 – Nombre d'infectés

Ces données sont elles en accord avec vos simulations ?

On fait l'hypothèse que les différences entre les données et la simulation sont dues au fait qu'une partie de la population était naturellement immunisée contre la population. Estimez la proportion d'immunisés dans la population initiale.