Corrigé de la séance 3

Comme tous les corrigés des séances d'exercices le corrigé de la séance 3 est partiel. Vous trouverez ci-dessous quelques programmes qui répondent en partie aux questions posées dans l'énoncé de la séance et qui fournissent éventuellement des résultats supplémentaires. Ces programmes n'ont pas la prétention d'être optimaux et de répondre aux canons de l'informatique...

Toute information complémentaire peut-être obtenue soit auprès de la personne qui encadre les travaux pratiques du groupe auquel vous appartenez, soit en adressant vos demandes par courriel (au moyen de votre adresse ulb) à mtolley@ulb.ac.be, soit aux "intercours" ou lors des permanences organisées "à la carte" (cf. vos délégués d'année).

Méthode du point fixe pour résoudre l'équation (7)

```
% Chenilles-Sapins-Oiseaux
% Méthode du point fixe pour résoudre l'équation (7)
clc; clear
display('Méthode du point fixe - Deux fonctions d''itération distinctes')
% Les deux fonctions d'itération utilisées dans la méthode du point fixe
f1  = @(x,rho,q) rho*(1+x.*x).*(1-x/q);
f2  = @(x,rho,q) q*(1 - x./(rho*(1+x.*x)));
% On choisit q
q = 10;
% Les valeurs de départ de l'algorithme sont choisies égales à zéro ou q,
% ou à partir du graphe fourni par le programme Appelle_Newton_Bud.m
u_R(1,1:10)=0;
u_P(1,1:10)=q;
% On traite dix valeurs différentes de rho
for k=1:10
rho(k)=k*0.05;
% On fait 19 itérations sans tester la convergence
for n=2:20
    u_R(n,k)=f1(u_R(n-1,k),rho(k),q);
    u_P(n,k)=f2(u_P(n-1,k),rho(k),q);
end
end
```

```
% On affiche les résultats.
% Pour certaines valeurs de rho, ça "foire", pourquoi ?
% On choisit un format qui permette de lire le résultat dans la fenêtre de
% commande
format short e
rho
u_R
u_P
% Pour voir plus de décimales on utilise le format long e
% Ici, la convergence est linéaire et nettement moins bonne qu'avec la
% méthode de Newton qui converge quadratiquement pour les racines simples.
format long e
u_R
u_P
```

Méthode de Newton ou de la tangente

```
% Chenilles-Sapins-Oiseaux
% Méthode de Newton ou de la tangente
% Cette fonction est appelée par le programme nommé
% Appelle_Newton_Bud.m
function U=Newton_Bud(rho,q,U0)
% La fonction dont on cherche les zéros
zer  = @(x,rho,q) rho*(1-x/q) - x./(1+x.*x);
% La dérivée de la fonction dont on cherche les zéros
dzer = @(x,rho,q) -rho/q + (x.*x-1)./(1+x.*x).^2;
% U0 est la valeur de départ de l'algorithme
U(1)=U0;
% On fait maximum 20 itérations et on teste la convergence
for n=2:21
    U(n)=U(n-1) - zer(U(n-1),rho,q)/dzer(U(n-1),rho,q);
    if abs(U(n)-U(n-1)) < 1e-14
        break
    end
end
U=U';
```

textbfProgramme qui appelle la fonction qui résout l'équation (7) par la méthode de Newton

```
% Chenilles-Sapins-Oiseaux
% Programme qui appelle la fonction qui résout l'équation (7) par la
% méthode de Newton pour rho, q et U0 donnés
% U0 est la valeur initiale du processus itératif qu'il faut choisir selon
% les valeurs de rho et q que l'on se donne ainsi que la valeur de u_eq
```

```
% escomptée et que l'on peut estimer à partir d'un graphe.
% S'il n'y a qu'un seul zéro réel on peut aussi trouver les racines
% complexes à condition de choisir un bon "U0"
% Essayer q= 10, rho = 0.3 et U0 = 1 + 0.5i
clc; clear
% La fonction qui définit la courbe y=g(x)
g   = @(x) x./(1+x.*x);
% La fonction qui définit la droite y=d(x)
d   = @(x,rho,q) rho*(1-x/q);
% Tracé de la courbe et de la droite correspondant aux rho et q donnés
% On choisit 101 abscisses équiréparties sur [0,10]
u=0:0.01:10;
% On choisit q et rho
q=input('Donnez q  ')
rho=input('Donnez rho  ')
% Tracé de la droite d(u)
figure(1)
plot(u,d(u,rho,q),'r','LineWidth',1.5);hold on
% Dessin de la courbe g(u) et notations le long des axes
plot(u,g(u),'LineWidth',1.5)
grid on % On place une grille sur le graphe
title('Densité de population à l''équilibre','FontSize',14)
xlabel('Densité de population réduite  u','FontSize',14)
ylabel('Fonctions d(u) et g(u)','FontSize',14)
zoom on
display('Zoomez sur le dessin pour déterminer une approximation de u_eq')
U0=input('Introduisez la valeur de u_eq comme valeur de départ du calcul ')
format long e
% Appel de la fonction qui résout l'équation (7)
U=Newton_Bud(rho,q,U0)
```

Programme qui effectue le tracé partiel des graphes des figure 2 et 4 de la "note". On utilise notamment les fonctions "fzero" et "roots" de Matlab.

```
% Chenilles-Sapins-Oiseaux
% Tracé partiel des graphes des figure 2 et 4 de la "note"
% On utilise notamment les fonctions "fzero" et "roots" de Matlab
% Choristoneura et sapins baumiers
% Ludwig et al. 1978 - Voir note ad hoc (appelée note ci-dessous)
% Les vecteurs et matrices ont des dimensions raisonnables on ne se
% préoccupe donc pas des préallocations
clear
% La fonction qui définit la courbe y=g(x)
g   = @(x) x./(1+x.*x);
% La fonction qui définit la droite y=d(x)
```

```
d   = @(x,rho,q) rho*(1-x/q);
% La fonction dont on cherche les zéros
zer  = @(x,rho,q) rho*(1-x/q) - x./(1+x.*x);
% Tracé des courbes et droites de la figure 2 de la note
% On choisit 101 abscisses équiréparties sur [0,10]
u=0:0.01:10;
% On choisit q = 10 partout dans la suite
q = 10;
% Images des abscisses, dessin de la courbe notations le long des axes
fu=g(u);
figure(1)
plot(u,fu,'LineWidth',1.5)
grid on % On place une grille sur le graphe
title('Densité de population à l''équilibre','FontSize',14)
xlabel('Densité de population réduite  u','FontSize',14)
ylabel('Fonctions d(u) et g(u)','FontSize',14)
hold on % On "fixe" la figure pour faire d'autres graphes sur le précédent
% Trois droites de "travail" particulières en rouge
for rho=[0.30 0.45 0.60]
d1=d(u,rho,q);
plot(u,d1,'r','LineWidth',1.5);
end
% Les deux tangentes à la courbe en pointillé noir
plot(u,d(u,0.5595,q),'k:','LineWidth',1.5)
plot(u,d(u,0.3840,q),'k:','LineWidth',1.5)
% On fabrique 1001 valeurs de rho
for n=1:1001
rho(n)=(n-1)/1000;
end
% Calcul des valeurs d'équilibre pour 1001 valeurs de rho allant de 0 à 1
% On utilise "fzero" pour trouver le plus petit zéro, u_R associé à rho
% "roots" calcule les trois zéros dont on stocke le zéro intermédiaire dans
% un vecteur nomé u_I
u_R(1)=0;
for n=2:1001
a=rho(n);
u_R(n)=fzero(@(x) zer(x,a,q),u_R(n-1),q);
pp=[1,-q,1+q/rho(n),-q];
rac=roots(pp);
rac'
u_I(n)=rac(2);
end
% Ouverture d'une nouvelle figure - Courbe d'hystérèse
figure(2)
% Tracé de la partie correspondant aux équilibres stables de refuge u_R
```

```
plot(rho,u_R,'LineWidth',1.5)
grid on
hold on
% Tracé de la partie correspondant aux équilibres instables u_I
plot(rho(384:559),u_I(384:559),'r:','LineWidth',1.5)
% Calcul des valeurs d'équilibre pour 1001 valeurs de rho allant de 1 à 0
% On utilise "fzero" pour trouver le plus petit zéro, u_R associé à rho
u_P(1001)=0;
for n=1:1001
a=rho(1002-n);
u_P(n+1001)=fzero(@(x) zer(x,a,q),u_P(n+1000));
end
% Partie correspondant aux équilibres stables de pullulation u_P
plot(1.0-rho,u_P(1002:end),'LineWidth',1.5)
% Labels des axes et titre de la figure
title('Phénomène d''hystérèse','FontSize',14)
xlabel('\rho','FontSize',14)
ylabel('u_R','FontSize',14)
```