Equations Différentielles Ordinaires (EDO)

TP MATLAB

Exercice 2 :

Lors de cet exercice nous allons étudier le système d’équations différentielles de Lokta-Volterra suivant :

avec

On a le programme principal Matlab suivant :

clear;close all;

tmin=0;

tmax=15;

% param?tres du mod?le proies-pr?dateurs

alpha=1; % taux de reproduction des proies

beta=0.5; % taux de mortalit? des proies

gamma=2; % taux de reproduction des pr?dateurs

delta=1; % taux de mortalit? des pr?dateurs

% conditions initiales

x0=2; % proies

y0=0.5; % predateurs

% second membre de l'?qu. diff. (x',y')=(f(t,x,y),g(t,x,y))

f=@(t,x,y)(x.\*(alpha - beta\*y));

g=@(t,x,y)(-y.\*(gamma - delta\*x));

% Calcul des populations des proies et des pr?dateurs

h=0.01; % pas temporel

% 1. méthode d'Euler

[xEuler,yEuler,t]=Euler\_2D(x0,y0,tmin,tmax,f,g,h);

% 2. méthode RK2

beta=1;

[xRK2,yRK2,t1]=RK2\_2D(x0,y0,tmin,tmax,h,beta,f,g);

%modification des Conditions initiales

[xRK2bis,yRK2bis,t1]=RK2\_2D(3,2,tmin,tmax,h,beta,f,g);

% Affichage des populations des proies et des pr?dateurs

% en fonction du temps

figure(1);hold on;

plot(t,xEuler,'r');

plot(t,yEuler,'b');

plot(t1,xRK2,'g');

plot(t1, yRK2, 'm');

title('affichage des populations des proies et des prédateurs');

xlabel('base de temps');

ylabel('valeur des populations');

legend('affichage proies par la méthode Euler ','affichage prédateurs par la méthode Euler','affichage proies par la methode Runge-Kutta ordre2 x0=2 ', 'affichage prédateurs par la méthode Runge-Kutta ordre2 y0=0.5 ');

figure(2);hold on;

plot(t1,xRK2,'g');

plot(t1, yRK2, 'm');

plot(t1,xRK2bis,'y');

plot(t1, yRK2bis, 'k');

title('affichage des populations des proies et des prédateurs');

xlabel('base de temps');

ylabel('valeur des populations');

legend('affichage proies par la methode Runge-Kutta ordre2 x0=2 ', 'affichage prédateurs par la méthode Runge-Kutta ordre2 y0=0.5 ','affichage proies par la methode Runge-Kutta ordre2 x0=2 ', 'affichage prédateurs par la méthode Runge-Kutta ordre2 y0=3');

% affichage de la trajectoire proies-pr?dateurs (tangente au champ de

% vecteurs d?fini par la fonction (x,y)->(f(t,x,y),g(t,x,y))

figure(3);hold on;

% champ de vecteurs (x,y)->(f(t,x,y),g(t,x,y))

N=40;

ux=linspace(0,7,N);

uy=linspace(0,7,N);

[x,y]=meshgrid(ux,uy); % grille de coordonn?es (ux,uy)

fxy=f(t,x,y);gxy=g(t,x,y); % calcul du champ de vecteurs

norme=(fxy.^2+gxy.^2).^0.5; % normalisation des vecteurs

fxy=fxy./norme;gxy=gxy./norme;

quiver(x,y,fxy,gxy); % affichage de fxy et gxy sous forme

% de champ de vecteurs

% trajectoires proies-pr?dateurs

% 1. m?thode de d'Euler

plot(xEuler,yEuler, 'r');

% 2a. méthode RK2

%plot(xRK2, yRK2, 'g');

plot(x0,y0,'r\*');

title('trajectoires proies-prédateurs avec x0=2 et y0=0.5');

xlabel('nombre de proies');

ylabel('nombre de prédateurs');

legend('champ de vecteurs','trajectoires pour la méthode d Euler');

% 2b. méthode RK2 et nouvelles conditions initiales

[xRK2bis,yRK2bis,t1]=RK2\_2D(3,2,tmin,tmax,h,beta,f,g);

figure(4);hold on;

N=40;

ux=linspace(0,7,N);

uy=linspace(0,7,N);

[x,y]=meshgrid(ux,uy); % grille de coordonn?es (ux,uy)

fxy=f(t,x,y);gxy=g(t,x,y); % calcul du champ de vecteurs

norme=(fxy.^2+gxy.^2).^0.5; % normalisation des vecteurs

fxy=fxy./norme;gxy=gxy./norme;

quiver(x,y,fxy,gxy); % affichage de fxy et gxy sous forme

% de champ de vecteurs

plot(xRK2, yRK2, 'g');

plot(xRK2bis, yRK2bis, 'r');

plot(x0,y0,'g\*');

plot(3,2,'r\*');

title('trajectoires proies-prédateurs ');

xlabel('nombre de proies');

ylabel('nombre de prédateurs');

legend('champ de vecteurs','trajectoires pour la méthode Runge-Kutta pour x0=2 et y0=0.5','trajectoires pour la méthode Runge-Kutta pour x0=3 et y0=2');

On code la fonction de la méthode d’Euler avec le programme suivant :

function[x,y,t]=Euler\_2D(y0,x0,tmin,tmax,f,g,h)

t=tmin:h:tmax;

y=zeros(1,length(t)); %tableau de 0 que l’on va remplir dans la boucle for

x=zeros(1,length(t));

y(1)=y0; %initialisation de la première valeur

x(1)=x0;

for k=2 : length(t)

y(k)=(y(k-1)) + h\*g(t(k-1),x(k-1),y(k-1)); %schéma d’Euler explicite appliqué aux 2 EDO

x(k)=(x(k-1)) + h\*f(t(k-1),x(k-1),y(k-1));

end

end

On code la fonction de la méthode de Runge-Kutta avec le programme suivant :

function [x,y,t] = RK2\_2D(x0,y0,tmin,tmax,h,beta,f,g)

t=tmin:h:tmax;

y=zeros(1,length(t));

x=zeros(1,length(t));

y(1)=y0; %initialisation de la première valeur des EDO

x(1)=x0;

%application du schéma de Runge Kutta aux 2 EDO

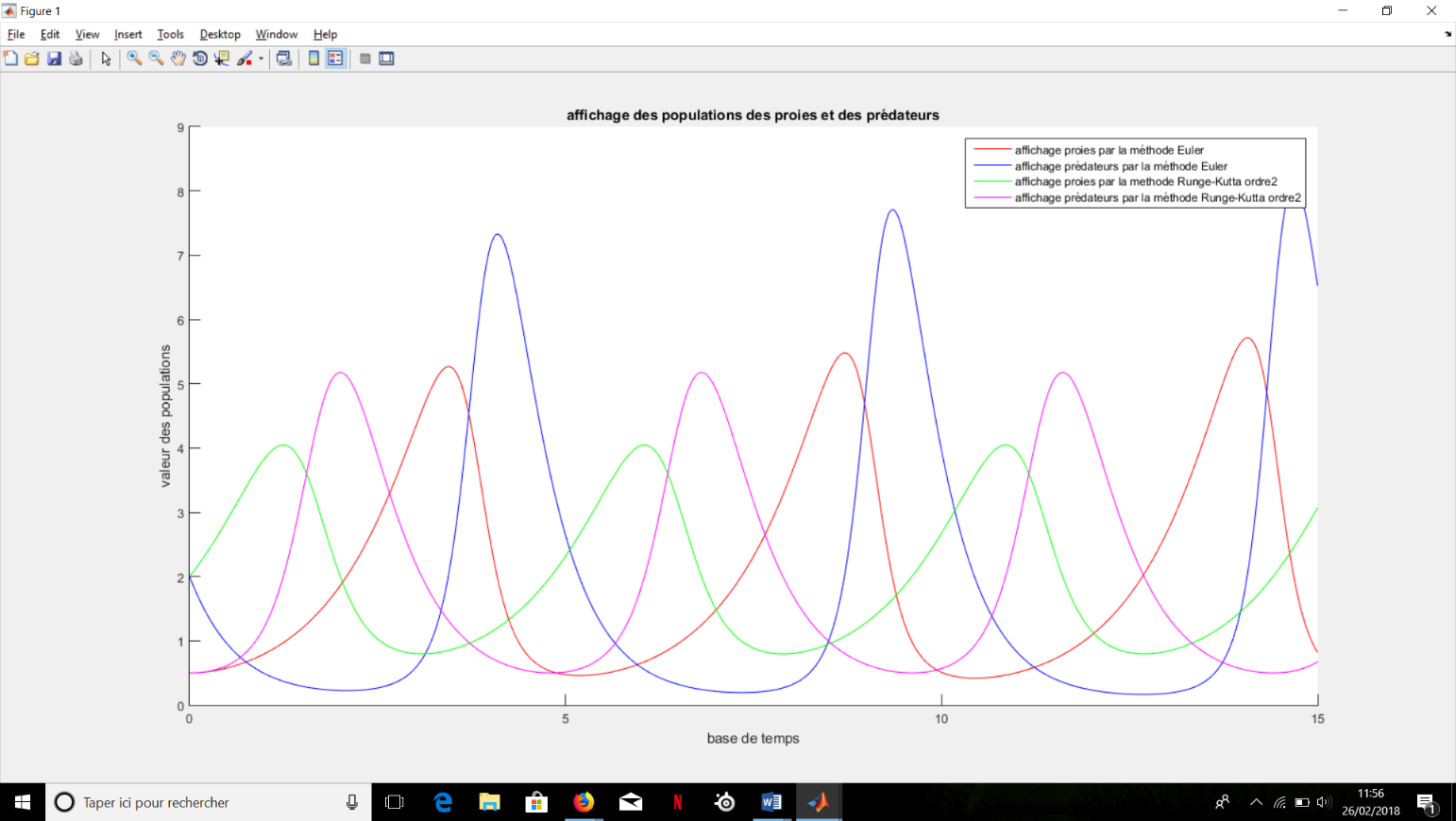
for k=2 : length(t)

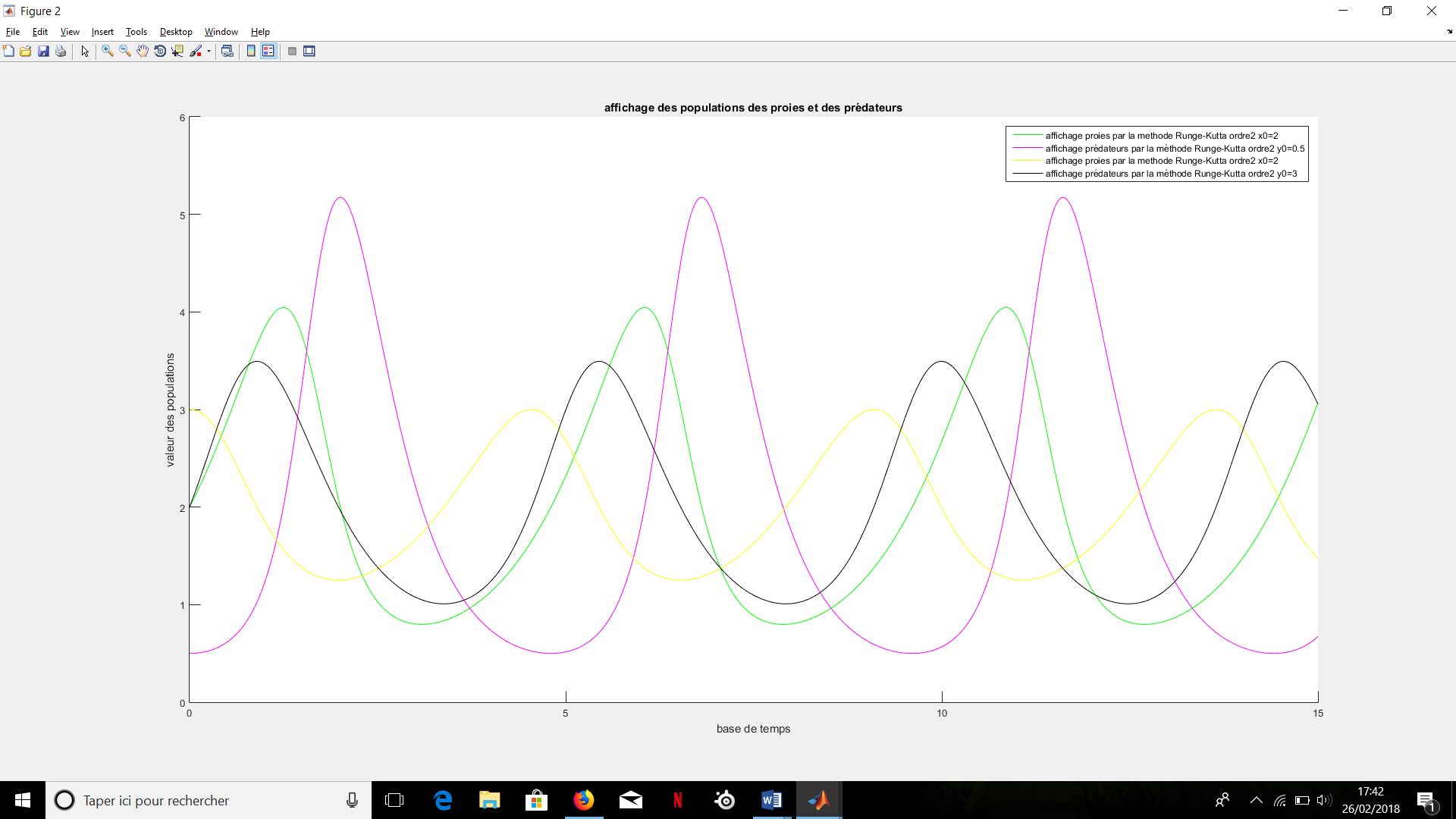
x(k) = x(k-1) + h \* ((1-beta) \* f(t(k-1),x(k-1),y(k-1)) + beta \* f(t(k-1) + h / 2 \* beta,x(k-1) + (h / 2 \* beta)\*f(t(k- 1),x(k-1),y(k-1)), y(k-1) + (h / 2 \* beta)\*g(t(k-1),x(k-1),y(k-1))));

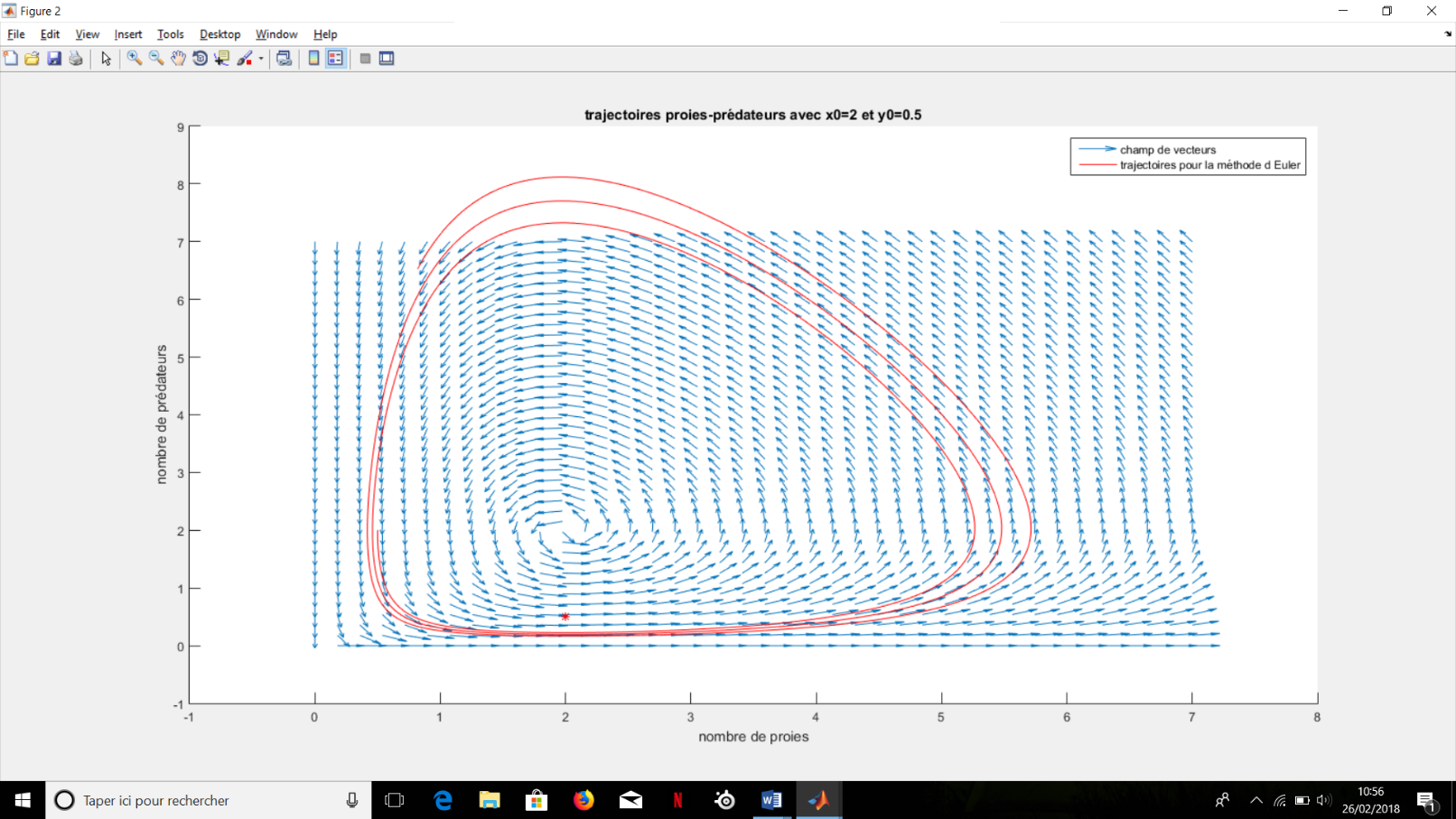
y(k) = y(k-1) + h \* ((1-beta) \* g(t(k-1),x(k-1),y(k-1)) + beta \* g(t(k-1) + h / 2 \* beta,x(k-1) + (h / 2 \* beta)\*f(t(k-1),x(k-1),y(k-1)), y(k-1) + (h / 2 \* beta)\*g(t(k-1),x(k-1),y(k-1))));

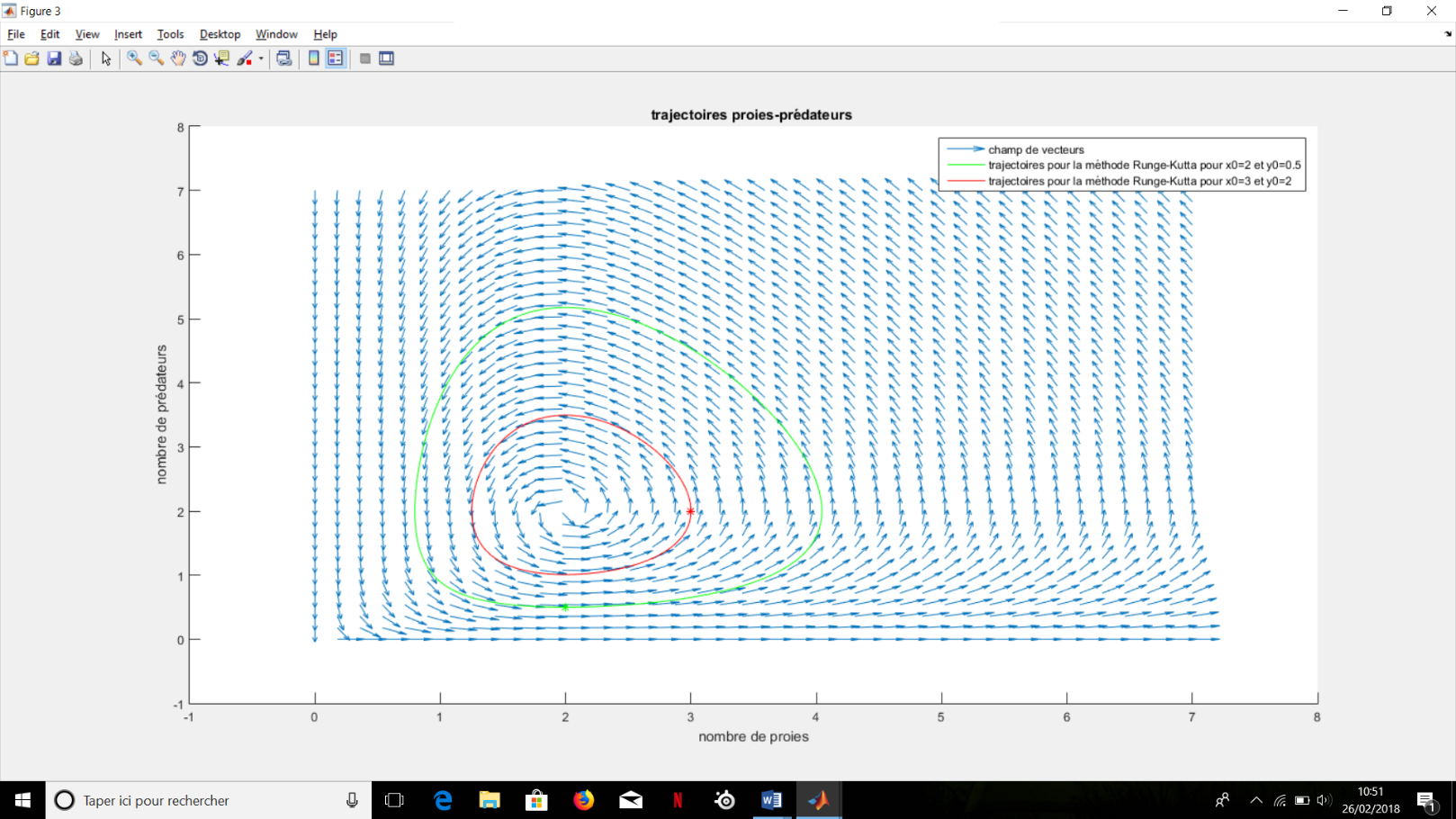
end

end









Lors de cet exercice nous allons résoudre le problème de Lokta-Volterra avec la méthode d’Euler ainsi qu’avec la méthode de Runge-Kutta aux deux équations différentielles ordinaires sur l’intervalle [0 ;15]. On donne les conditions initiales suivantes : et un pas de 0.01. Puis nous modifierons ces conditions initiales pour voir leur influence dans la méthode de Runge-Kutta.

L’objectif de cet exercice est donc de comparer les deux méthodes de résolution ainsi que l’influence des conditions initiales.

Nous codons donc ces deux méthodes à l’aide de Matlab dans les fonctions et données ci-dessus.

On cherche à afficher la courbe des populations des proies et des prédateurs en fonction du temps, ainsi que les trajectoires proies-prédateurs. Pour cela on affiche les valeurs des proies et prédateurs renvoyées par les deux fonctions en fonction du temps. On remarque qu’avec la méthode d’Euler lorsque le nombre de proies augmente le nombre de prédateurs augmente aussi, réciproquement lorsque le nombre de proies baisse celui de prédateur baisse aussi. On observe le même phénomène avec la méthode de Runge Kutta avec un nombre de proies et prédateurs moins important. On remarque aussi que la courbe des proies et des prédateurs de la méthode d’Euler a tendance à osciller légèrement.

Ensuite nous avons comparé la trajectoire des proies-prédateurs en fonction de la méthode utilisée et en traçant le champ de vecteur. On remarque que pour la méthode d’Euler, la trajectoire constituée des couples solutions ne se superpose pas et à tendance à diverger. Contrairement, la trajectoire constituée des couples solutions par la méthode de Runge-Kutta est stable et forme un cercle continu. Lorsqu’on change les conditions initiales dans la méthode de Runge-Kutta on remarque que cela à une influence sur la trajectoire des couples solutions mais la trajectoire est toujours continue. De même sur la figure représentant l’évolution des proies et des prédateurs en fonction du temps, on remarque que les conditions initiales ont une influence (ici le nombre de proies et de prédateur et moins important lorsqu’on augmente et qu’on ne bouge pas .

En conclusion on peut dire que la méthode de Runge Kutta semble plus stable que la méthode d’Euler au vu de la trajectoire des couples solutions. On a pu aussi voir que la modification des conditions initiales a une influence sur l’évolution des proies-prédateurs en fonction du temps et sur la trajectoire des couples solutions mais sans influencer la stabilité.