Equations Différentielles Ordinaires (EDO)

TP MATLAB

Exercice 1:

clear variables;

close all;

tmin=0;tmax=1;

f=@(t,y)((t.^2).\*exp(t)-y.\*(3\*t.^2-1));%fonction y'(t)

y0=-1;

yExact=@(t)(exp(t).\*(1-4\*exp(-t.^3))/3);%solution exact du problème

% solution approchï¿½e de l'eq. diff.

% 1. mï¿½thode d'Euler (h=0.1)

h=0.1;

[yEuler1,t1]=fct\_Euler(y0,tmin,tmax,h,f);

eps1=abs(yEuler1-yExact(t1)); % erreur

% 2. mï¿½thode d'Euler (h=0.05)

h=0.05;

[yEuler2,t2]=fct\_Euler(y0,tmin,tmax,h,f);

eps2=abs(yEuler2-yExact(t2)); % erreur

% 1. mï¿½thode RK2 (h=0.1 et beta=1)

h=0.1;beta=1;

[yRK,t3]=fct\_RK2(y0,tmin,tmax,h,beta,f);

eps3=abs(yRK-yExact(t3)); % erreur

%gestion de l'affichage des courbes

figure(1);

subplot(1,2,1);

hold on;

plot(t1,yEuler1,'\*-');

plot(t1,yExact(t1));

plot(t3,yRK,'\*-');

plot(t2,yEuler2,'\*-');

legend('Euler pas de 0.1','Solution Exacte','Runge-Kutta pas de 0.05','Euler pas de 0.05');

title('Solution exacte, Solution approchée par méthode d Euler et Rung-Kutta 2');

xlabel(' t ');

ylabel ('y(t)');

subplot(1,2,2)

hold on;

plot(t1,eps1,'x-');

plot(t2,eps2,'x-');

plot(t3,eps3,'x-');

legend('Courbe erreur Euler pas de 0.1','Courbe erreur Euler pas de 0.05','Courbe erreur Runge-Kutta pas de 0.05');

title('Courbes des erreurs associées aux solutions approchées');

xlabel(' t ');

ylabel ('erreur');

Fonction qui calcul la solution avec la méthode d’Euler:

function[y,t]=fct\_Euler(y0,tmin,tmax,h,f)

t=tmin:h:tmax;%création d'un tableau pour la variable t avec pas de h

y=zeros(1,length(t));%tableau de 0

y(1)=y0;%initialisation du premier terme d'indice 1 = à y0

for k=2 : length(t) %boucle calculant les termes d'indices superieur

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

end

end

Fonction qui calcul la solution avec la méthode de Rung-Kutta d’ordre 2:

function[y,t]=fct\_RK2(y0,tmin,tmax,h,beta,f)

t=tmin:h:tmax;

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

y(1)=y0;

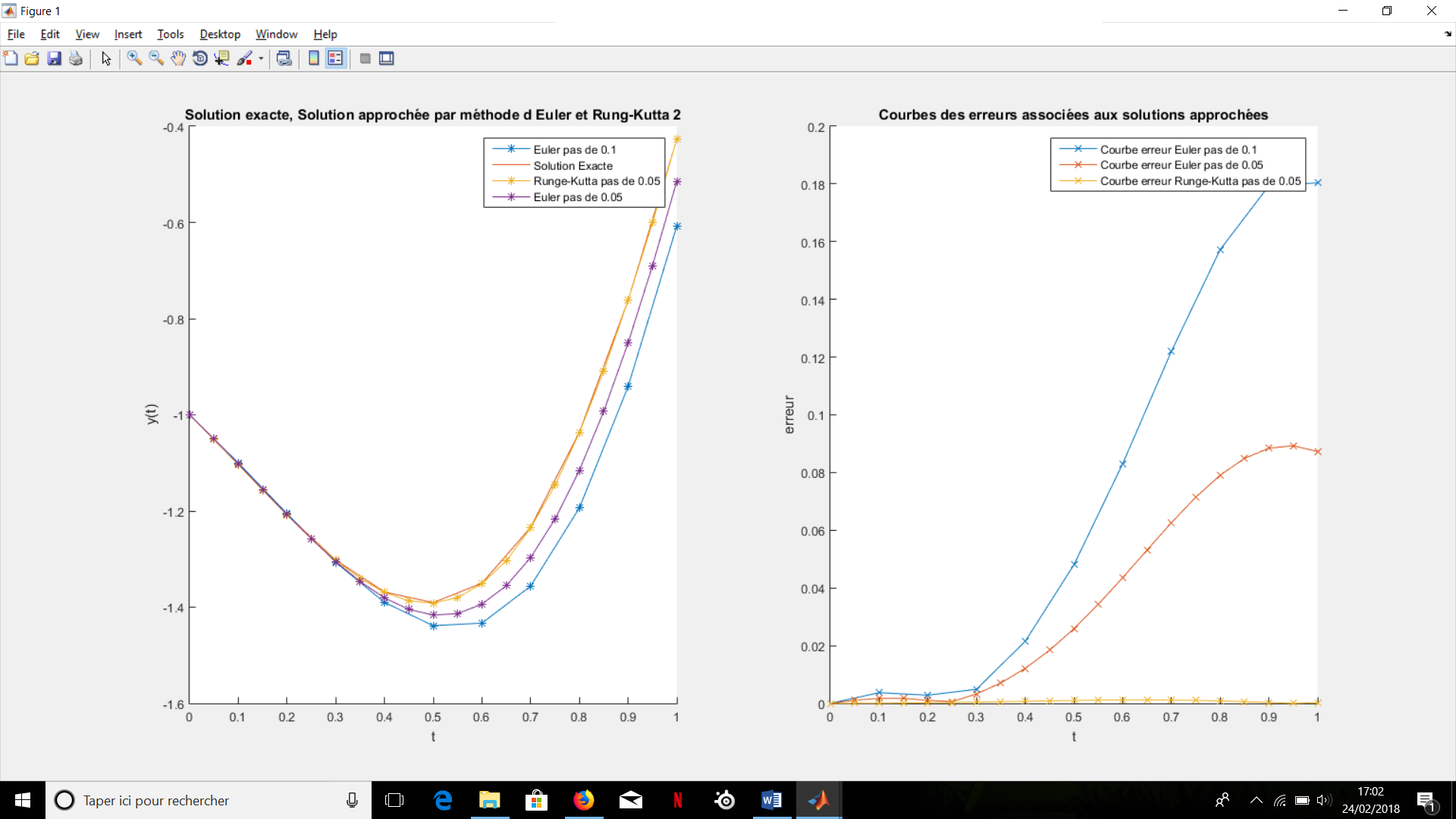
for k=2 : length(t)

H=h/(2\*beta)

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

end

end

On obtient la figure suivante:

Lors de ce TP, l’objectif était de resoudre le problème de Cauchy suivant :

Pour résoudre ce système nous nous sommes intéressés à différentes méthodes : la méthode d’Euler ainsi que la méthode de Runge-Kutta d’ordre 2. Nous avons donc codé sur Matlab une fonction pour chaque méthode que nous avons appelé dans un programme principal.

La solution exacte du problème nous était donné, le second objectif était donc de comparer les deux méthodes pour pouvoir déterminer laquelle est la plus efficace et présente le moins d’erreur.

Pour réliser la méthode d’Euler avec Matlab, nous avons crée une fonction prenant en paramètre les bornes max et min de l’intervalle sur lequel on étudie la fonction (ici entre 0 et 1). De même on renseigne le pas que l’on souhaite attribué pour savoir combien de points on souhaite calculer pour tracer la solution. On prendra aussi en paramètre de notre fonction d’Euler, la fonction qui donne en fontion de et , c’est à dire . On défini cette fonction dans notre programme principale ([f=@(t,y)((t.^2).\*exp(t)-y.\*(3\*t.^2-1))](mailto:f=@(t,y)((t.%5e2).*exp(t)-y.*(3*t.%5e2-1)))).

La fonction d’Euler nous retournera ensuite les points au point grâce à la solution au point et à la connaissance du schéma Euler explicite :

On trace ensuite la courbe des solutions en fonction de . On effectue exactement le même travail pour trouver la solution du système avec la méthode d’Euler pour un pas différents en agissant sur le paramètre dans notre programme principal.

Pour tracer ensuite la solution par méthode de Rung-Kutta d’ordre 2 avec un pas de 0.05 et en fixant on suit le même raisonnement que précédemment en changeant le schéma Euler explicite dans la boucle par celui de la méthode Rung-Kutta 2 :

De même on trace ensuite la courbe des solutions en fonction de .

Pour pouvoir comparer les différentes méthodes est savoir laquelle est la plus adapté on s’intéresse à l’erreur. L’erreur associée à une solution est définie en traçant la différence entre la solution exacte du problème est la solution associée à une méthode.

Après avoir tracé tous les graphiques on peut en déduire que la méthode d’Euler avec un pas de 0.1 est la méthode qui suit le moins la courbe de la solution exacte. En revanche la méthode de Rung-Kutta d’ordre 2 est celle qui possède le moins d’erreur et donc qui suit pour le mieux la solution exacte.

On peut retenir que ces deux méthodes sont faciles à mettre en œuvre et requiert peut de code et que la méthode d’Euler, qui est plus simple à mettre en œuvre, est moins précise que celle de Rung-kutta.

Programme calculant l’erreur en fonction du pas ainsi que la RMS :

maxerr = [];

hi = [];

for h = 0.01 : 0.01 : 0.1

hi = [hi, h]; % tableau des différentes valeurs du pas

[yRK4,t4]=fct\_RK4(y0,tmin,tmax,h,f); %calcul de la solution

maxerr = [maxerr, max(yRK4-yExact(t4))]; %calcul de l'erreur maximale

end

% affichage de l'erreur en fonction du pas

figure(2);hold on;

plot(hi, maxerr);

title('courbe de l erreur en fonction du pas avec méthode Runge-Kutta ordre 4');

legend('erreur maximale entre la solution et la solution approchée');

ylabel('erreur');

xlabel(' pas ');

%%détermination du polynome à l'aide de polyfit

somme=0;

imax=9;

rms=[];

for i=1:1:imax

P = polyfit(hi, maxerr, i);

somme=(maxerr(i)-P(i)).^2;

rms=[rms, sqrt(somme./imax)];

end

figure(3);

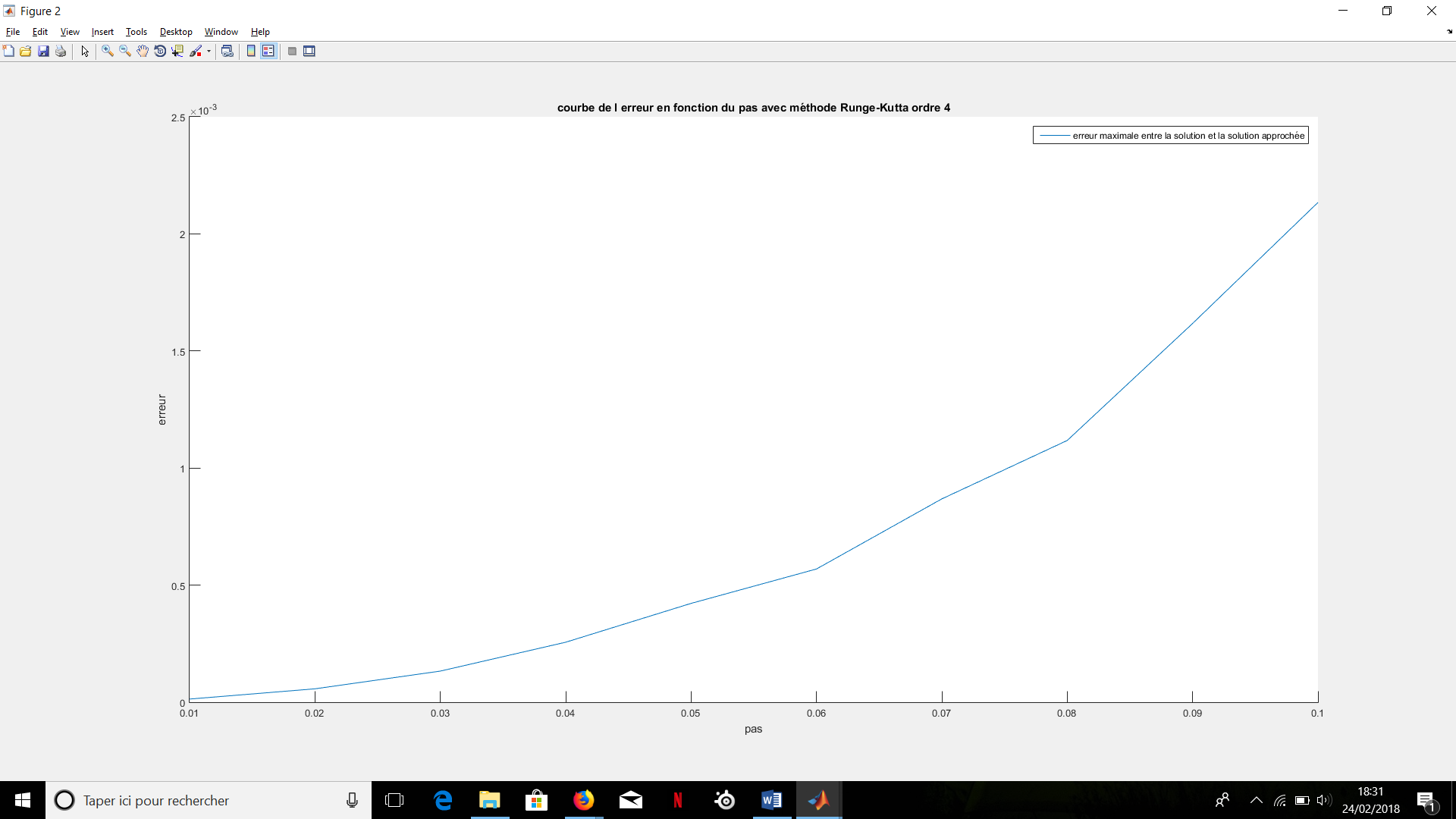
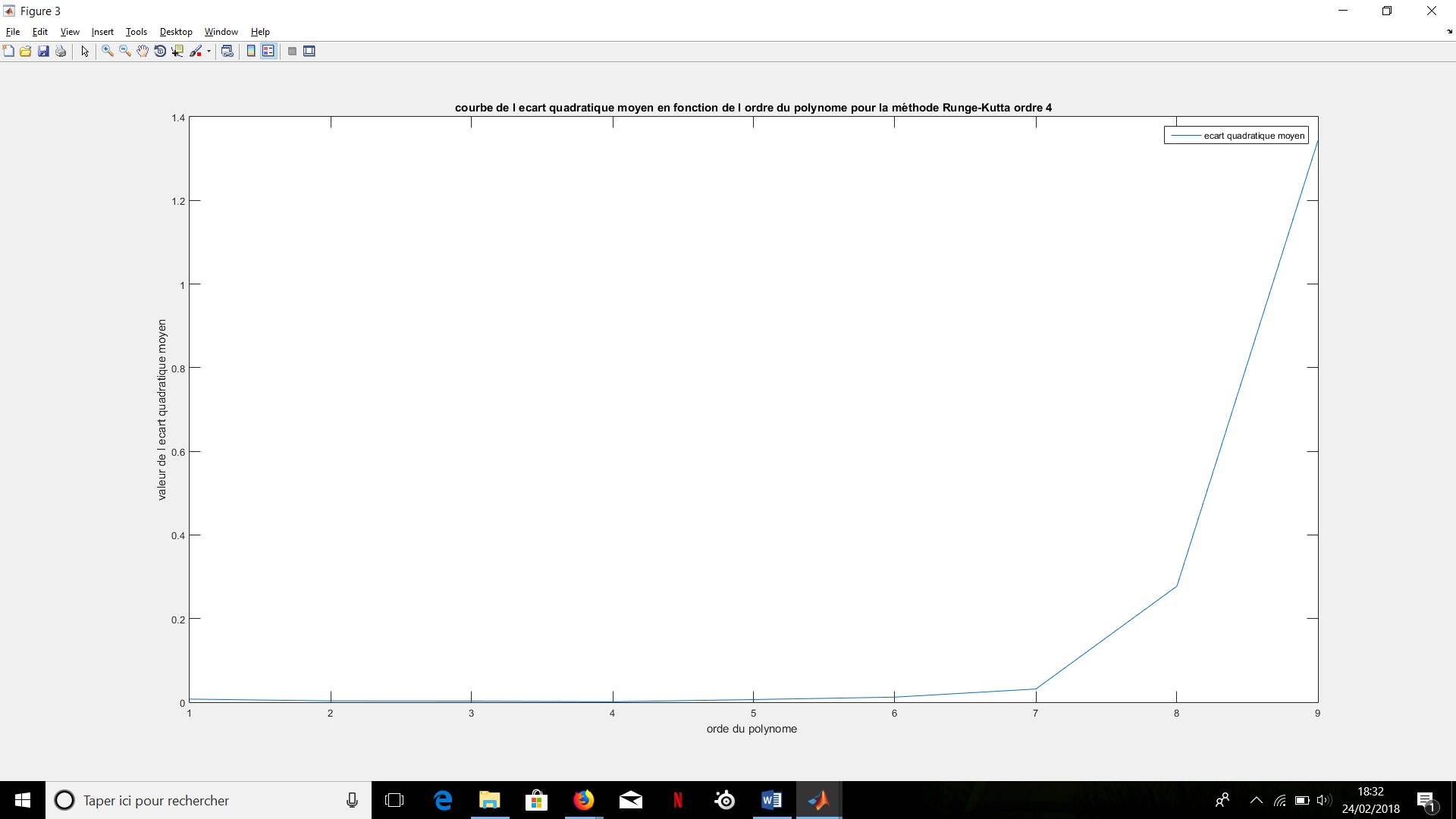
plot([1:imax],rms);

title('courbe de l ecart quadratique moyen en fonction de l ordre du polynome pour la méthode Runge-Kutta ordre 4');

legend('ecart quadratique moyen');

ylabel('valeur de l ecart quadratique moyen');

xlabel('orde du polynome ');



Dans un second temps nous allons mettre en œuvre la méthode de Rung-Kutta d’ordre 4 et nous intéresser à l’influence des paramètres de la méthode. La fonction Matlab de cette méthode nous est donnée et nous allons calculer la solution approchée du problème de Cauchy en faisant varier le pas de 0.01 à 0.1 avec un pas de 0.01. On aura donc 9 valeurs de pas.

Ensuite nous calculons pour chaque valeur de pas l’erreur maximale entre la solution exacte et la solution approchée. On affiche ensuite sur un graphique l’erreur en fonction du pas.

On remarque sur ce graphique que plus le pas est élevé plus l’erreur est importante. Il faut donc minimiser le pas pour avoir une solution proche de la solution exacte.

Par la suite nous allons développer une méthode permettant de minimiser l’écart quadratique moyen afin de minimiser la distance entre chaque polynôme et la courbe de l’erreur en fonction du pas. Nous allons tout d’abord utiliser la fonction de Matlab pour trouver le polynôme au plus près des 9 points de la courbe tracée précédemment (erreur en fonction du pas) afin de calculer la RMS. On trace ensuite la valeur de la RMS en fonction de l’ordre du polynôme et on obtient la figure ci-dessus.

On remarque que la valeur de l’écart quadratique moyen est faible lorsque l’ordre du polynôme l’est aussi et que plus l’ordre du polynôme est élevé plus la valeur de la RMS augmente.

Pour conclure, on peut dire que dans ce TP nous avons vu comment mettre en œuvre différentes méthodes afin de résoudre des équations différentielles ordinaires. De plus nous avons aussi vu les limites de certains paramètres et comment les régler afin d’avoir de meilleurs résultats.