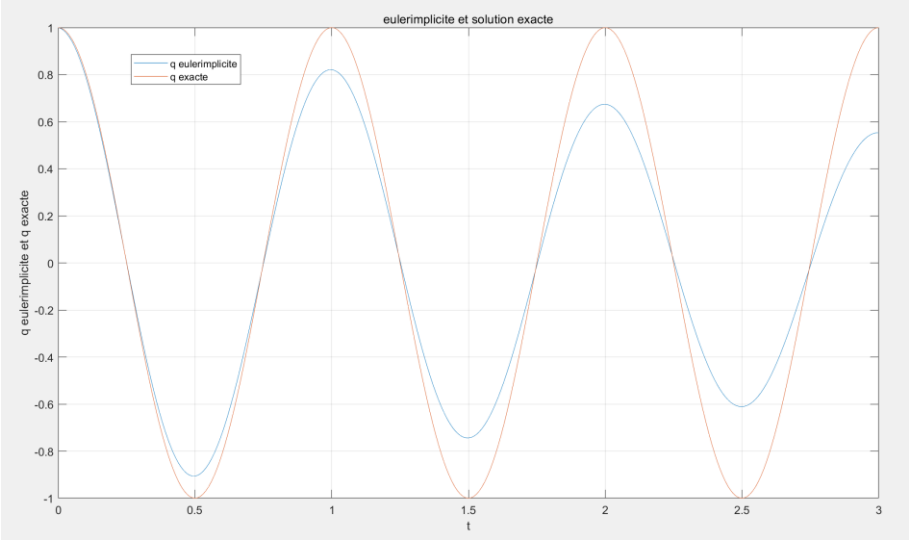
Mécanique Numérique DM3

3.Résolution avec schéma de Euler Implicite

3.1 Programme de la solution du problème à l'aide d'Euler implicite en utilisant la méthode de matrice A

Programmation en Matlab est comme ca:

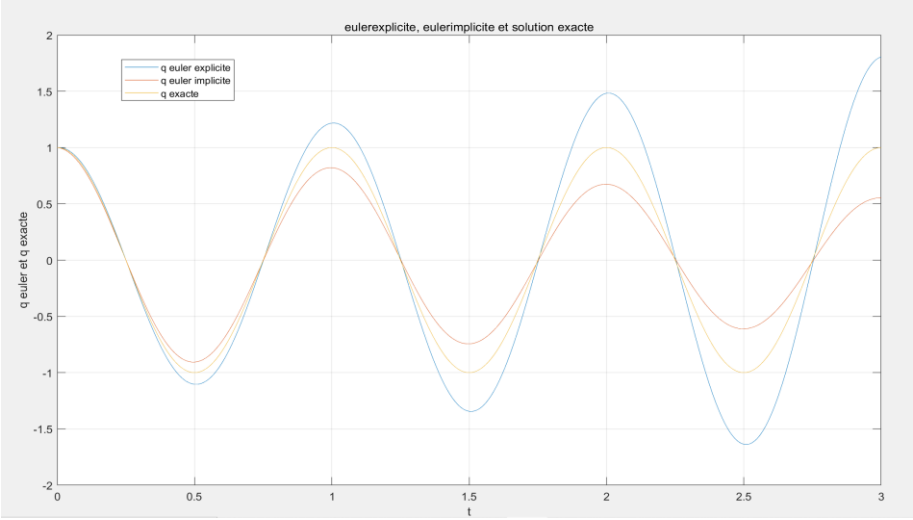
```
q0=1;dq0=0;w0=2*pi;t0=3;dt=0.01;
t=(0:dt:t0)';
nb=size(t,1);
q=[q0;dq0];
q1b=zeros(nb,1);
q1b(1)=q0;
A=(1/(1+(w0*dt)^2))*[1,dt;-(w0^2)*dt 1];
for i=2:nb
    q=A*q;
    q1b(i)=q(1);
end
plot(t,q1b),hold on
plot(t,cos(2*pi*t))
grid on;
xlabel('t');
ylabel('q euler implicite et q exacte');
title('euler implicite et solution exacte')
```



On peut avoir le dessin comme ca:

On voit qu'il existe un décalage entre la solution exacte et la solution obtenue à l'aide d'Euler implicite, et la décalage devient de plus en plus grand quand le temps augemente.

3.2 Comparaison des 3 solutions(solution exacte,euler explicite et euler implicite) avec dt=0.01



On met la solution exacte, euler explicite et euler implicite dans la même image:

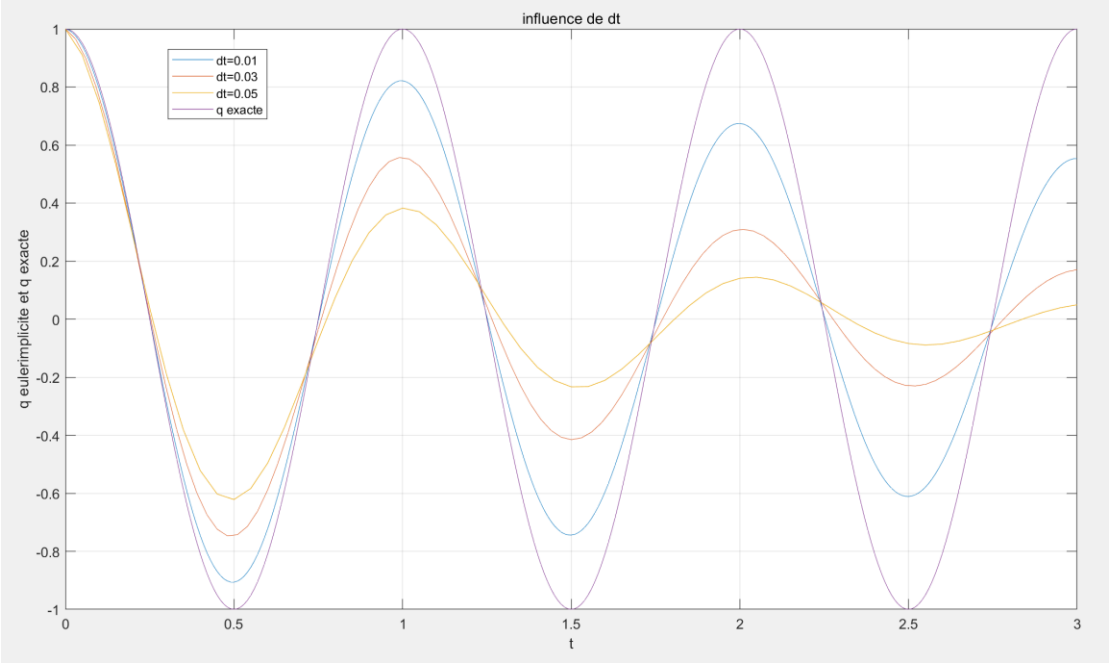
Alors, q_euler explicite est plus grand que q_exacte, et q_excate est plus grand que q_euler implicite,c'est-à-dire que :

q_euler implicite< q_exacte< q_euler explicite

3.3 L'influence du pas de temps

On teste et compare l'influence de 3 pas de temps, dt=0.01, 0.03 et 0.05, Programmation en Matlab est comme ca:

```
q0=1;dq0=0;w0=2*pi;T0=3;
for dt=[0.01 0.03 0.05 ]
    t=(0:dt:T0)';
    nb=size(t,1);
    q=[q0;dq0];
    q1b=zeros(nb,1);
    q1b(1)=q0;
    A=(1/(1+w0*w0*dt*dt))*[1,dt;-(w0*w0)*dt 1];
    for i=2:nb
        q=A*q;
        q1b(i)=q(1);
    end
    plot(t,q1b),hold on
end
t=(0:0.01:3);
plot(t,cos(2*pi*t))
grid on;
xlabel('t');
ylabel('q eulerimplicite et q exacte');
title('influence de dt')
```



Ici, on sait que la solution numérique obtenue a l'aide de ce schéma d'intégration introduit un amortissement numérique. En comparant les lignes dans le dessin , on a cependant que plus le pas de temps est petit, plus atténuation des oscillations est faible .

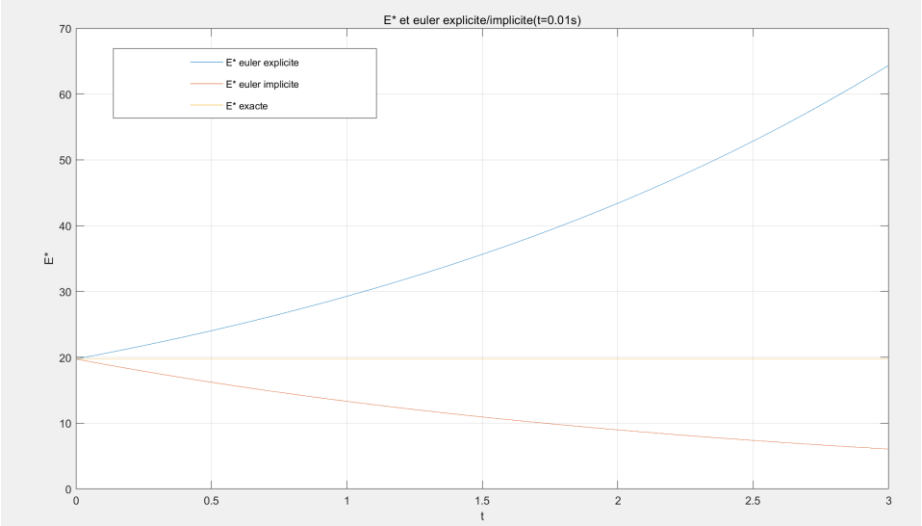
3.4.1 Calcul du E*(dt=0.01) à l'aide d'Euler implicite et Comparaison avec E* exacte et E*_euler explicite

Programmation en Matlab est comme ca:

```
q0=1;dq0=0;w0=2*pi;T0=3;dt=0.01;
t=(0:dt:T0)';
nb=size(t,1);
q=[q0;dq0];
q1b=zeros(nb,1);
dq1b=zeros(nb,1);
E_star_euler_im=zeros(nb,1);
q1b(1)=q0;
dq1b(1)=dq0;
E_star_euler_im(1)=2*pi*pi;
A=(1/(1+w0*w0*dt*dt))*[1,dt;-(w0*w0)*dt 1];
for i=2:nb
    q=A*q;
    q1b(i)=q(1);
    dq1b(i)=q(2);

    E_star_euler_im(i)=1/2*(dq1b(i).*dq1b(i)+(2*pi*q1b(
    i))^2);
end
```

```
plot(t,E_star_euler_im),hold on
plot(t,t*0+2*pi*pi)
grid on;
xlabel('t');
ylabel('E*');
title('E* et euler explicite
/implicite (t=0.01s)')
```



On met les E*_exacte, E*_euler explicite et E*_euler implicite dans la même image, on sait que E*(dt=0.01)_euler implicite et E*(dt=0.01)_euler explicite ne sont pas, au contraire E*_exacte est un constant, et

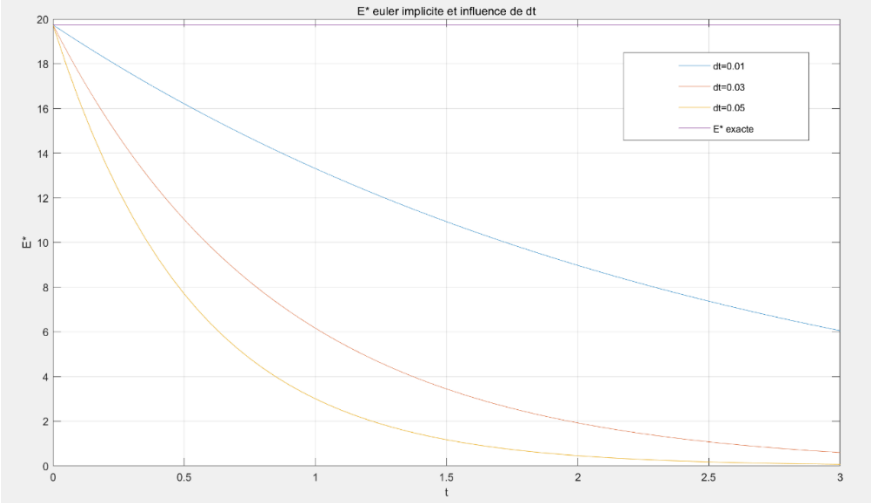
E*(dt=0.01)_euler explicite> E*_exacte> E*(dt=0.01)_euler implicite

3.4.2 L'influence du dt sur E*_euler implicite

Programmation en Matlab est comme ca:

```
q0=1;dq0=0;w0=2*pi;T0=3;
for dt=[0.01 0.03 0.05]
   t=(0:dt:T0)';
   nb=size(t,1);
   q=[q0;dq0];
   q1b=zeros(nb,1);
   dq1b=zeros(nb,1);
 E_star_euler_im=zeros(nb,1);
   q1b(1)=q0;
   dq1b(1)=dq0;
   E_star_euler_im(1)=2*pi*pi;
   A=(1/(1+w0*w0*dt*dt))*[1,dt;-(w0*w0)*dt 1];
   for i=2:nb
      q=A*q;
      q1b(i)=q(1);
      dq1b(i)=q(2);
```

```
        E_star_euler_im(i)=1/2*(dq1b(i).*dq1b(i)+(2*pi*q
        1b(i))^2);
    end
plot(t,E_star_euler_im),hold on
end
plot(t,t*0+2*pi*pi)
grid on;
xlabel('t');
ylabel('E*');
title('E* euler implicite et influence de dt')
```



On obtient le dessin qui montre que plus le pas de temps dt est petit, plus la vitesse de diminution du E*_euler implicite est faible. Donc c'est mieux de choisir un pas de temps(dt) plus petit.

3.5 Les valeurs propres de matrice d'amplification d'Euler implicite

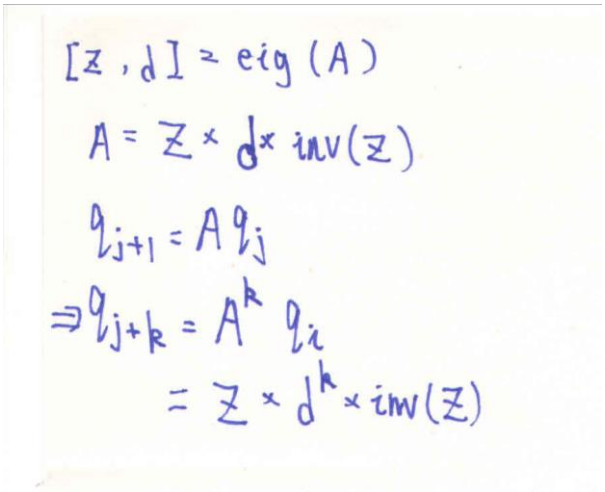
Calcul les valeurs propres de A

```
syms dt w0
A=(1/(1+w0*w0*dt*dt))*[1 dt;-(w0*w0)*dt 1]
[z,d]=eig(A)
mo=abs(d)
```

On a donc les valeurs propres :

d =[1i/(dt*w0 + 1i) , 0]
[0, 1/(1 + dt*w0*1i)]
=[(1-i*w0*dt)/(1+w0^2*dt^2), 0]
[0, (1+i*w0*dt)/(1+w0^2*dt^2)]

et m0=abs(d) est absolument plus petit que 1.

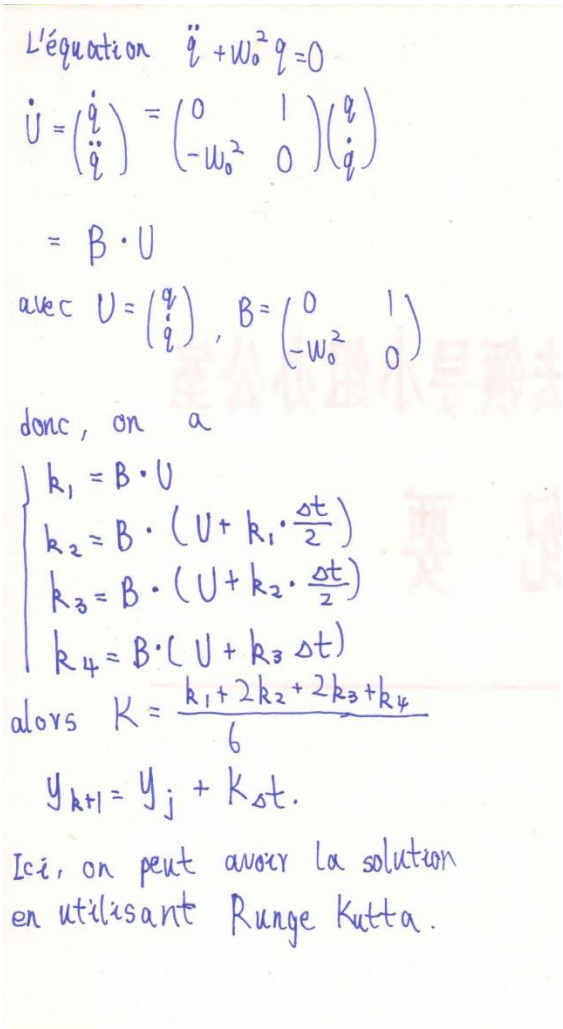


d<1, d^k converge.

Donc la solution obtenue par Euler implicite est inconditionnellement convergente.

4. Résolution avec un schéma de Runge Kutta

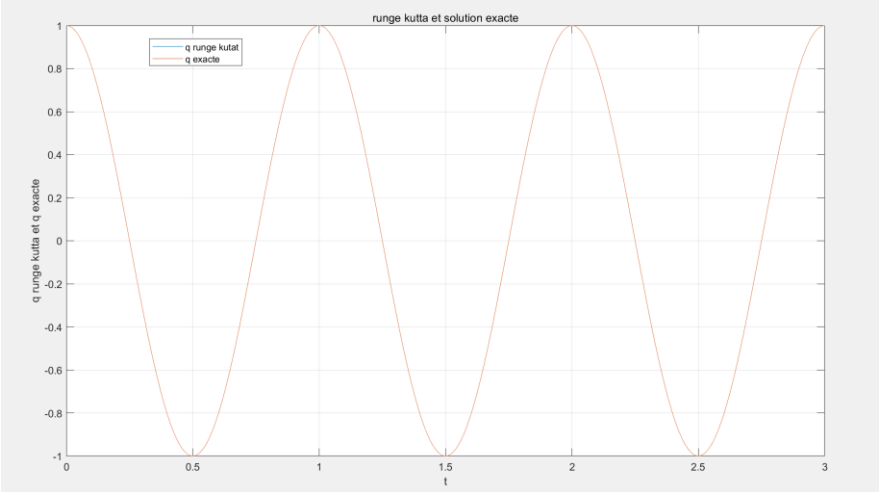
4.1 Transformation de (1) à Runge Kutta



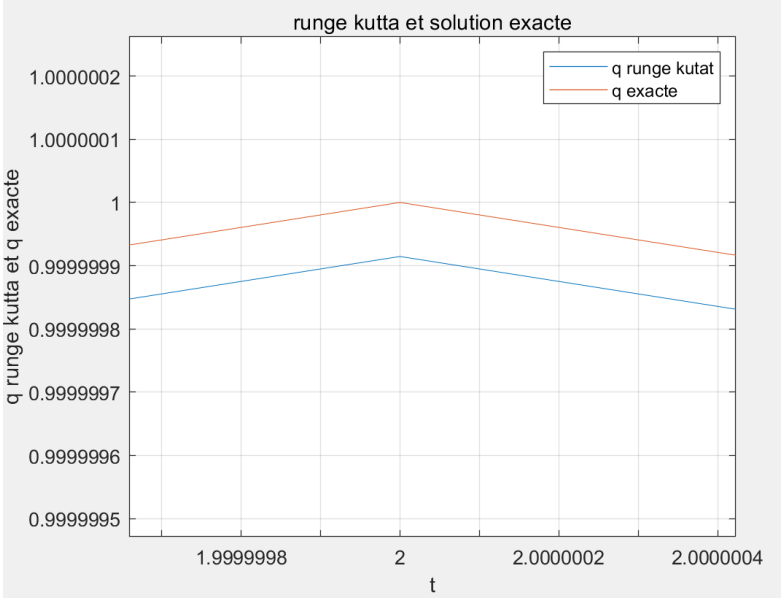
4.2 Solution à l'aide de Runge Kutta

Programmation en Matlab est comme ca:

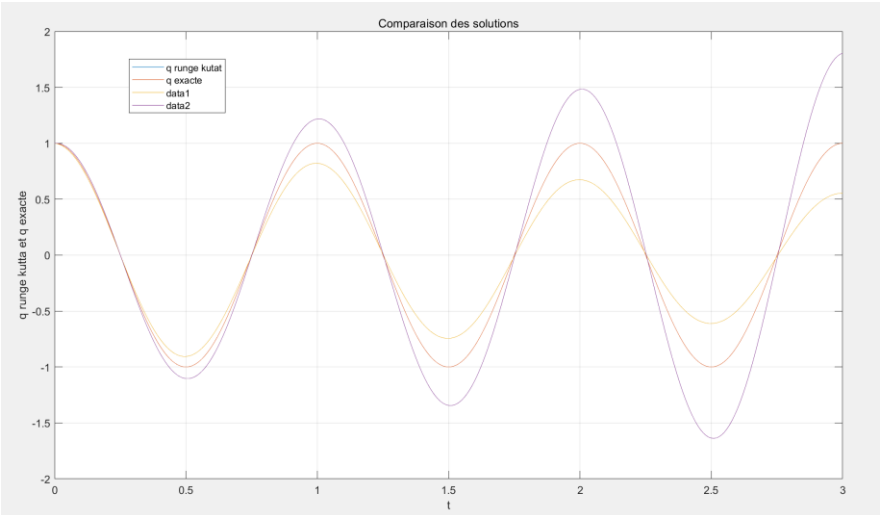
```
q0=1;dq0=0;w0=2*pi;T0=3;dt=0.01;
t=(0:dt:T0)';
nb=size(t,1);
q=[q0;dq0];
q1b=zeros(nb,1);
q1b(1)=q0;
B=[0 1;-w0*w0 0];
for i=2:nb
    k1=B*q;
    k2=B*(q+k1*dt/2);
    k3=B*(q+k2*dt/2);
    k4=B*(q+k3*dt);
    k=(k1+2*k2+2*k3+k4)/6;
    q=q+k*dt;
    q1b(i)=q(1);
end
plot(t,q1b),hold on
plot(t,cos(2*pi*t))
grid on;
xlabel('t');
ylabel('q runge kutta et q exacte');
title('runge kutta et solution exacte')
```



Donc on a obtient la solution à l'aide de Runge Kutta, il est très proche de la solution exacte.



4.3 Comparaison des solutions avec dt=0.01s



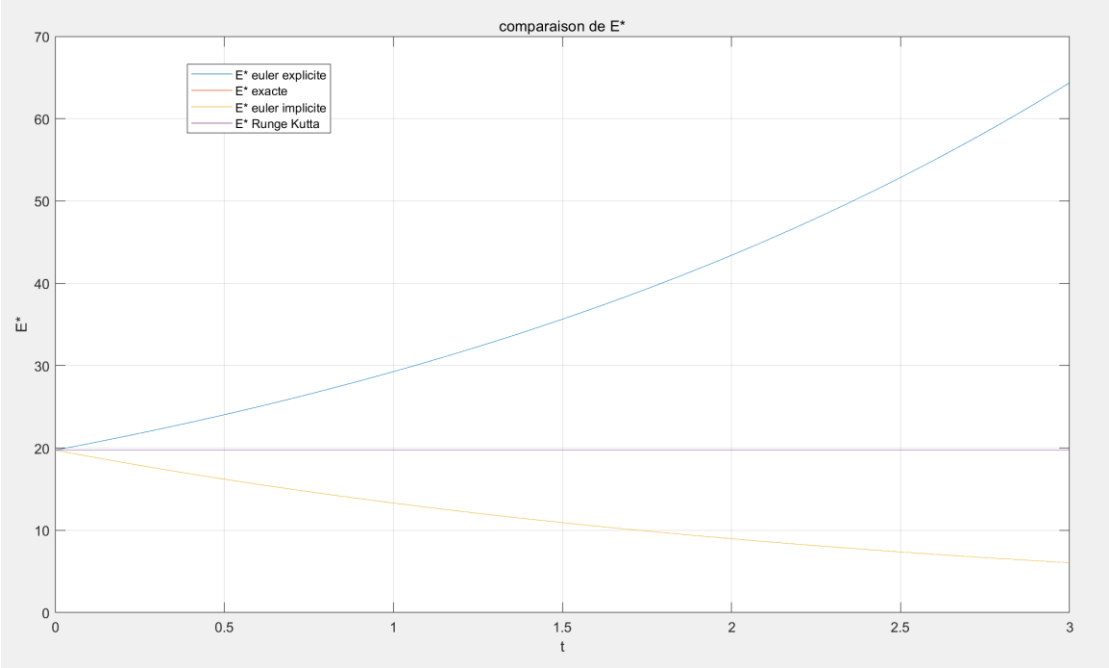
On met les solutions dans une même image. On voit que la solution de Runge Kutta est plus proche de la solution exacte que la solution d'Euler explicite et la solution d'Euler implicite.

4.4 Comparaison des E* à l'aide de façons différents

Programmation en Matlab est comme ca:

```
q0=1;dq0=0;w0=2*pi;T0=3;dt=0.01;
t=(0:dt:T0)';
nb=size(t,1);
q=[q0;dq0];
q1b=zeros(nb,1);
dq1b=zeros(nb,1);
E_star_rungekutta=zeros(nb,1);
q1b(1)=q0;
```

```
B=[0 1;-w0*w0 0];
E_star_rungekutta(1)=2*pi*pi;
for i=2:nb
    k1=B*q;
    k2=B*(q+k1*dt/2);
    k3=B*(q+k2*dt/2);
    k4=B*(q+k3*dt);
    k=(k1+2*k2+2*k3+k4)/6;
    q=q+k*dt;
    q1b(i)=q(1);
    dq1b(i)=q(2);
    E_star_rungekutta(i)=1/2*(dq1b(i).*dq1b(i)+(2*pi*q1
    b(i))^2);
end
plot(t,E_star_rungekutta),hold on
grid on;
xlabel('t');
ylabel('E*');
title('comparaison de E*')
```



Et on met tous les E* dans la même image. On voit clairement que E*_rungekutta est beaucoup plus proche de E*_exacte que les deux autres, E*_euler explicite et E*_euler implicite(dt=0.01s). Donc le facon de Runge Kutta est une façon plus précise qui fonctionne mieux.