Mécanique Numérique DM3

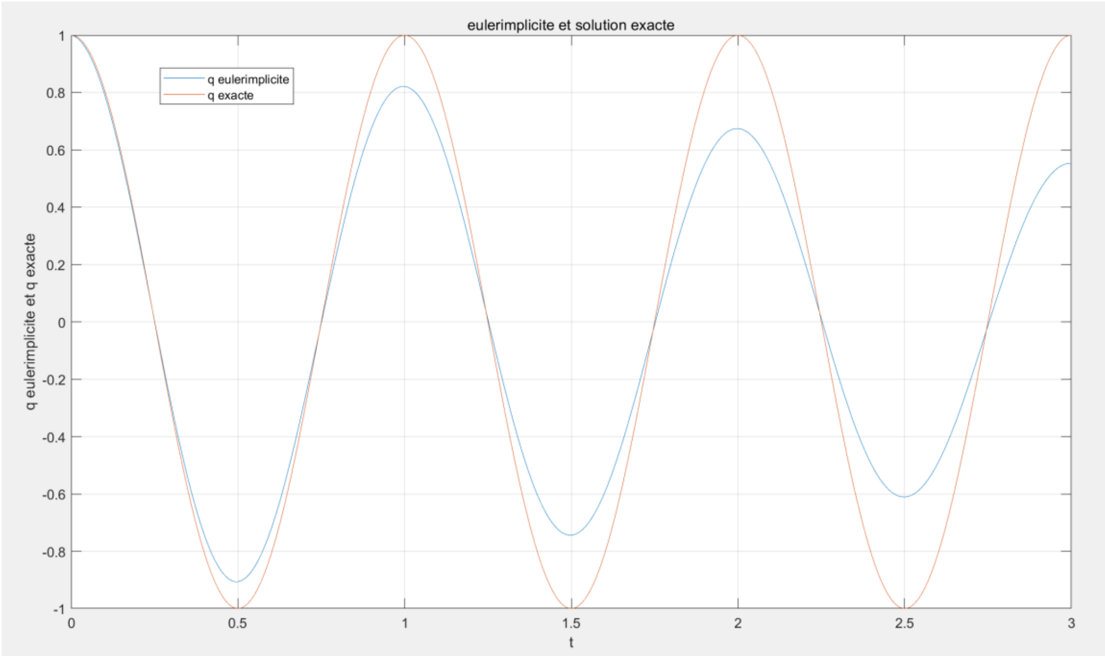
3.Résolution avec schéma de Euler Implicite

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

Script Matlab

```
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*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
plot(t,cos(2*pi*t))
grid on;
xlabel('t');
ylabel('q euler implicite et q exacte');
title('euler implicite et solution exacte')
```

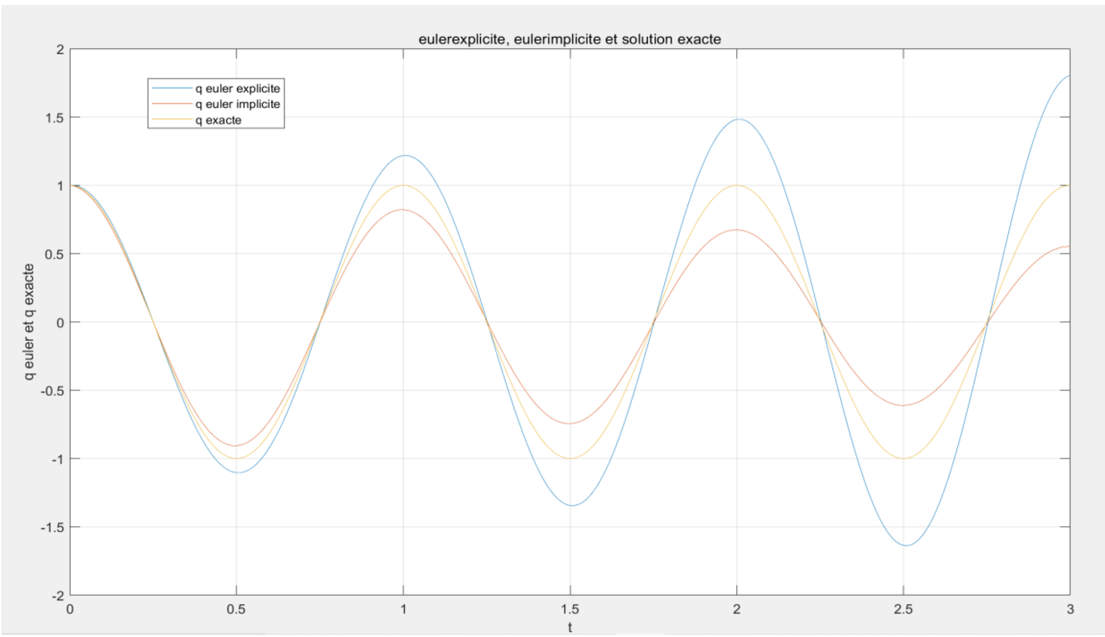
On obtient le dessin dessous :



Il y a un décalage entre la solution exacte et la solution obtenue à l'aide d'Euler implicite, et la décalage agrandit en fonction du t.

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

On met les 3 solution dans une même image :



Alors, q_eulerexplicite est plus grand que q_exacte, et q_excate est plus grand que q_eulerimplicite,c'est-à-dire que :

q_eulerexplicite> q_exacte> q_eulerimplicite.

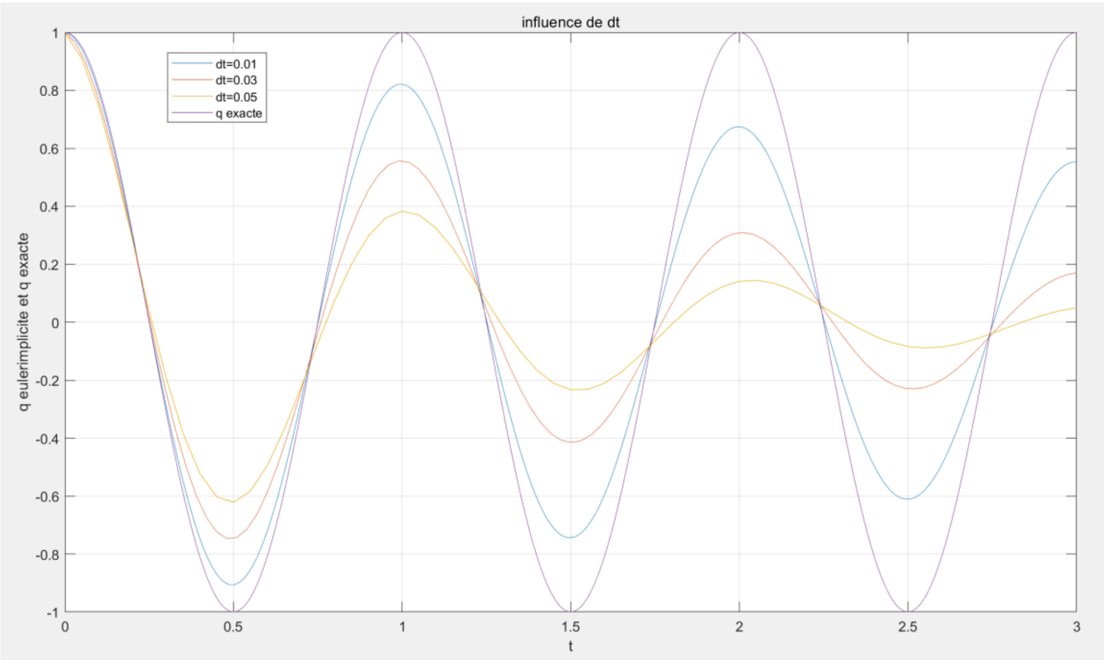
3.3 L'influence du pas de temps

On teste et compare l'influence de 3 pas de temps, dt=0.01, 0.03 et 0.05, script Matlab comme dessous :

```
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')
```

Bien entendu, la solution numérique obtenue par ce schéma d'intégration introduit un amortissement numérique. En comparant les lignes dans le dessin dessous, 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*_eulerexplicite

Scripte Matlab

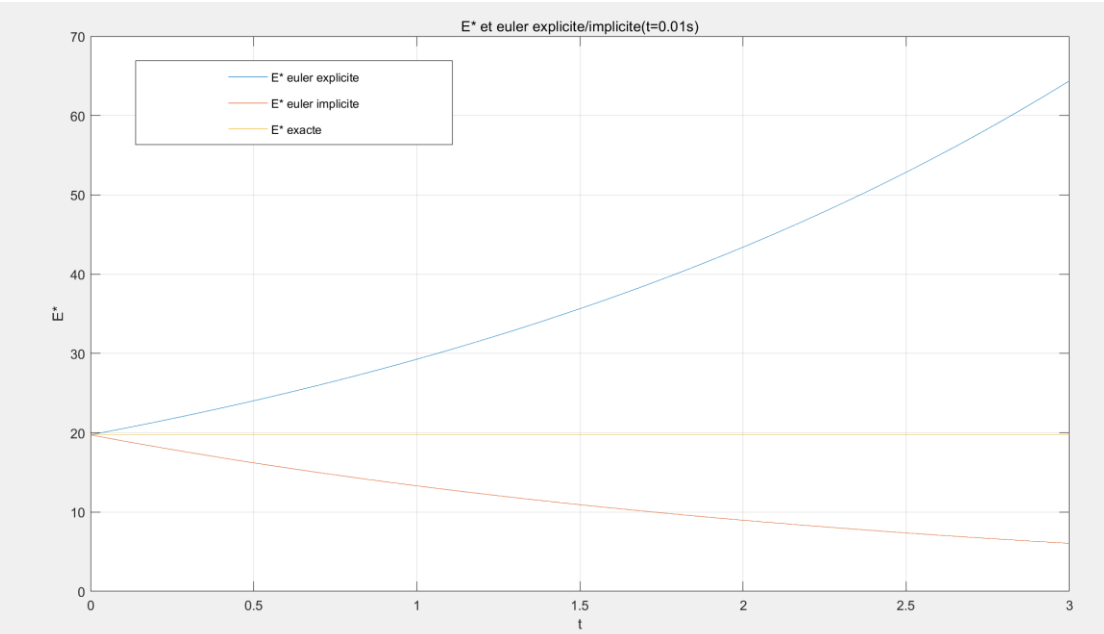
```
q0=1;dq0=0;w0=2*pi;T0=3;dt=0.01;

t=(0:dt:T0)';

nb=size(t,1);
```

```
q=[q0;dq0];
q1b=zeros(nb,1);
dq1b=zeros(nb,1);
E_etoile_euler_implicite=zeros(nb,1);
q1b(1)=q0;
dq1b(1)=dq0;
E_etoile_euler_implicite(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_etoile_euler_implicite(i)=1/2*(dq1b(i).*dq
   1b(i)+(2*pi*q1b(i))^2);
end
plot(t,E_etoile_euler_implicite),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*_eulerexplicite et E*_eulerimplicite dans la même image, on sait que E*(dt=0.01)_eulerimplicite et E*(dt=0.01)_eulerexplicite ne sont pas, au contraire E*_exacte est un constant, et

E(dt=0.01)_eulerexplicite> E*_exacte> E*(dt=0.01)_eulerimplicite*



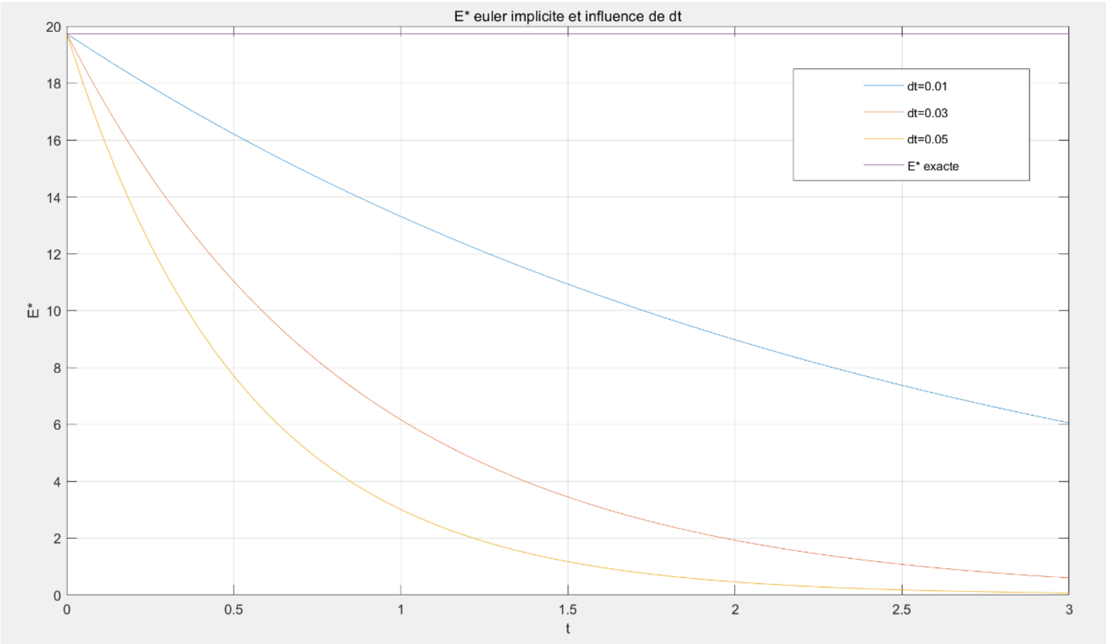
3.4.2 L'influence du dt sur E*_eulerimplicite

Script Matlab

```
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_etoile_euler_implicite=zeros(nb,1);
   q1b(1)=q0;
   dq1b(1)=dq0;
   E_etoile_euler_implicite(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_etoile_euler_implicite(i)=1/2*(dq1b(i).
      *dq1b(i)+(2*pi*q1b(i))^2);
   end
   plot(t,E_etoile_euler_implicite),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*_eulerimplicite est faible. Il nous indique que c'est mieux de choisir un pas de temps(dt) assez 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. Or

$$[z, d] = eig(A)$$
$$A = z * d * inv(z)$$
$$q_{j+1} = A q_j$$
$$q_{j+k} = A^k q_j$$
$$= z * d^k * inv(z)$$

et 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

4.1 L'équation (1) $\ddot{q} + w_0^2 q = 0$

On écrit $U = \begin{pmatrix} q \\ \dot{q} \end{pmatrix}$

alors $\dot{U} = \begin{pmatrix} \dot{q} \\ \ddot{q} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -w_0^2 & 0 \end{pmatrix} \begin{pmatrix} q \\ \dot{q} \end{pmatrix}$

$= \begin{pmatrix} 0 & 1 \\ -w_0^2 & 0 \end{pmatrix} \cdot U = B \cdot U$

C'est la même forme de $\dot{y}(t) = f(y(t), t)$

d'où on a

$$\begin{cases} k_1 = B \cdot U \\ k_2 = B \cdot (U + k_1 \cdot \frac{\Delta t}{2}) \\ k_3 = B \cdot (U + k_2 \cdot \frac{\Delta t}{2}) \\ k_4 = B(U + k_3 \cdot \Delta t) \end{cases}$$

alors $K = \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$

et $y_{k+1} = y_j + K\Delta t$.

On obtient donc la solution à l'aide de Runge Kutta.

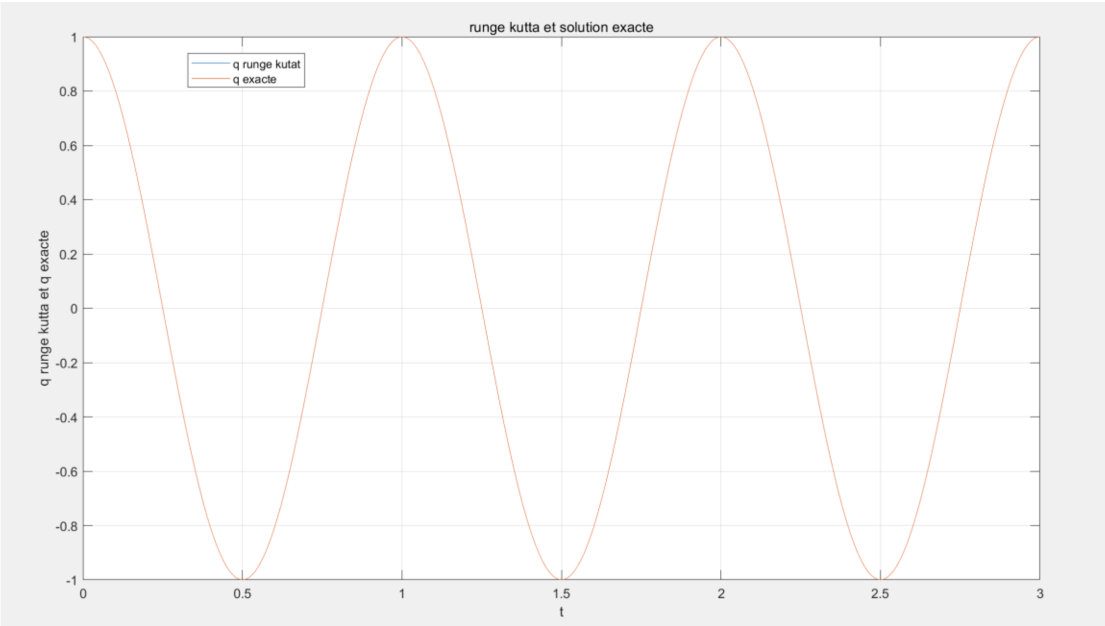
4.2 Solution à l'aide de Runge Kutta

Script Matlab

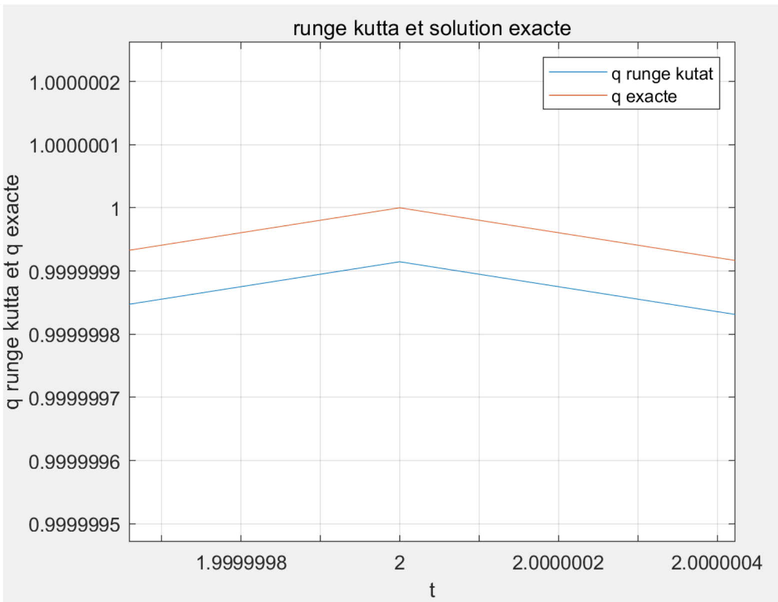
```
q0=1;dq0=0;w0=2*pi;T0=3;dt=0.01;

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')
```

La solution obtenue à l'aide de Runge Kutta est très proche de la solution exacte.

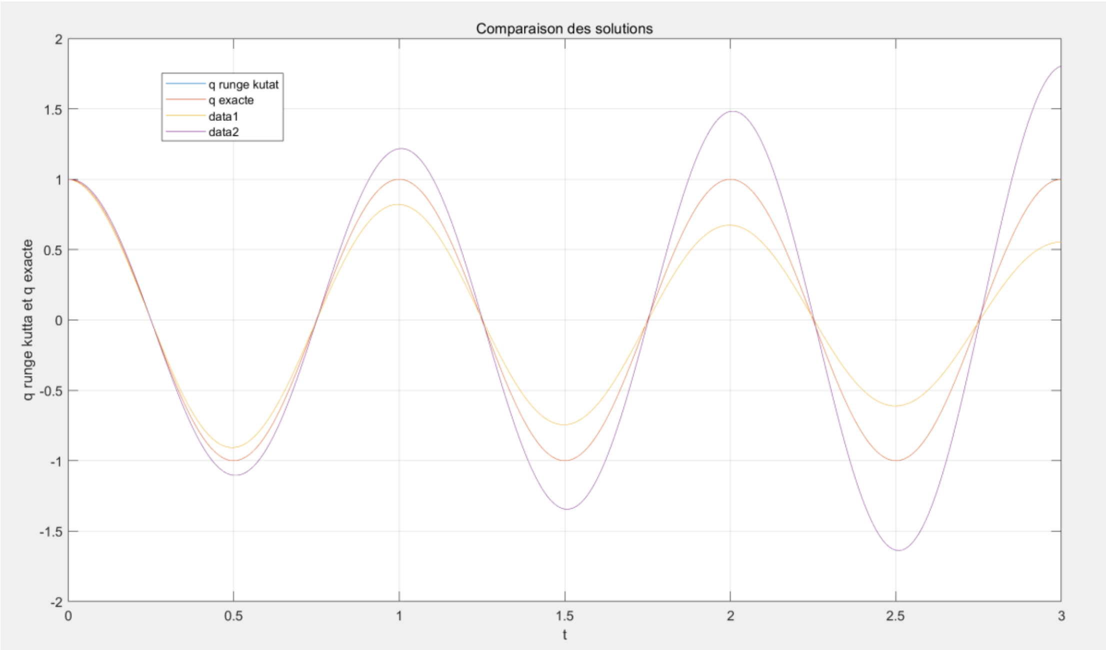


Les deux sont très proche :



4.3 Comparaison des solutions avec dt=0.01s

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



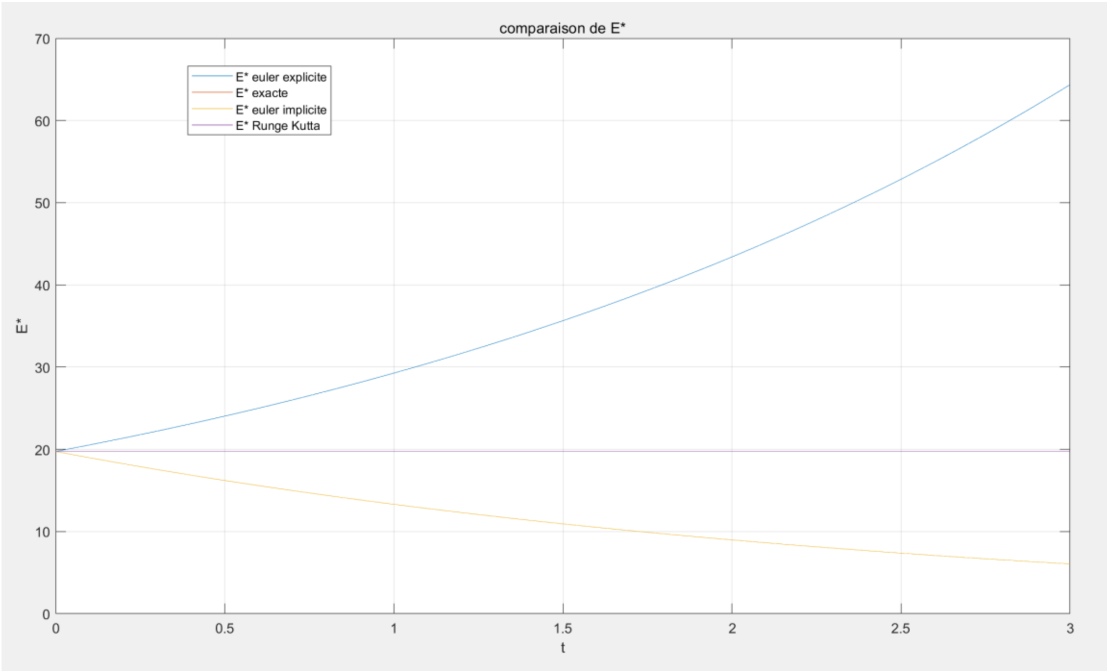
4.4 Comparaison des E* à l'aide de façons différents

Script Matlab

```
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_etoile_rungekutta=zeros(nb,1);

q1b(1)=q0;

B=[0 1;-w0*w0 0];

E_etoile_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_etoile_rungekutta(i)=1/2*(dq1b(i).*dq1b(i)
    +(2*pi*q1b(i))^2);
end
plot(t,E_etoile_rungekutta),hold on
grid on;
xlabel('t');
ylabel('E*');
title('comparaison de E*')
```

Et on met les E* obtenue par façons différentes dans la même image suivante, E*_rungekutta est beaucoup plus proche de E*_exacte que E*_eulerexplicite et E*_eulerimplicite(dt=0.01s). Le schéma de Runge Kutta est une façon plus précise qui fonctionne très bien.


comparaison de E*
E* euler explicite
E* exacte
E* euler implicite
E* Runge Kutta
E*
t