Exo 2. Etude d’un oscillateur linéaire amorti a un degré de liberté

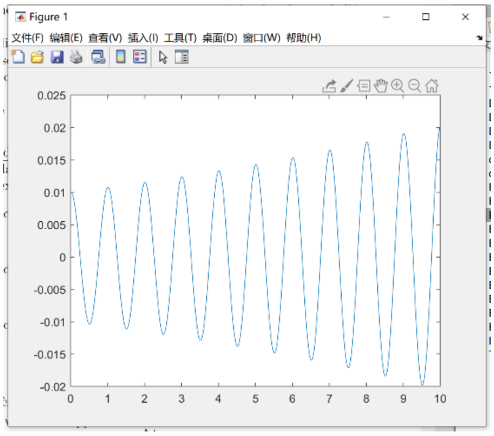
1.1 Le terme $\frac{2\epsilon}{\omega_0}$ vaut 0.0064

Le code de MATLAB :

```
T0 = 1;
omega=2*pi;
epsilon=0.02;
interval=10*T0;
pas = 0.01;
steps = interval/pas;
m = [0,1;-omega^2,-2*epsilon*omega];
M = eye(2)+pas*m;    % Matrice d'amplification
res = zeros(2,steps);
res(:,1) = [0.01;0];
for i=1:(steps-1)
    res(:,i+1) = M*res(:,i);
end
t = pas:pas:interval;
plot(t, res(1,:))
```

1.1.a $\frac{2\epsilon}{\omega_0} < \Delta t$

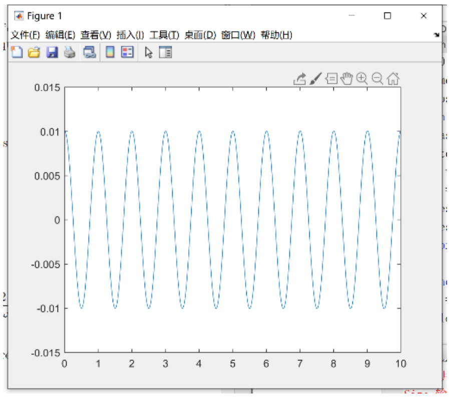
Je choisis Δt= 0.01s > 0.0064s



Le résultat est divergent, ce qui est impossible pour un oscillateur amorti.

1.1.b $\Delta t = \frac{2\epsilon}{\omega_0}$

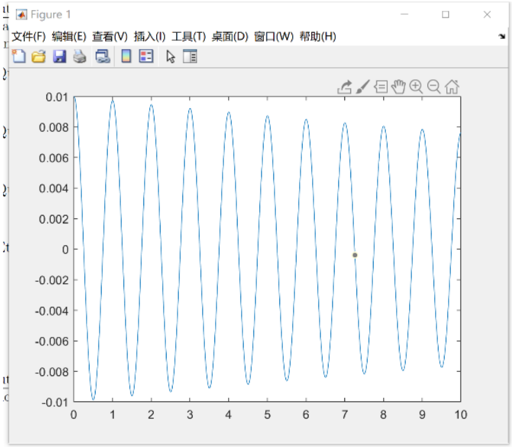
Je choisis Δt= 0.0064s



Le résultat est stable, ce qui est impossible pour un oscillateur amorti.

1.1.c $\Delta t = 0.8 \times \frac{2\epsilon}{\omega_0}$

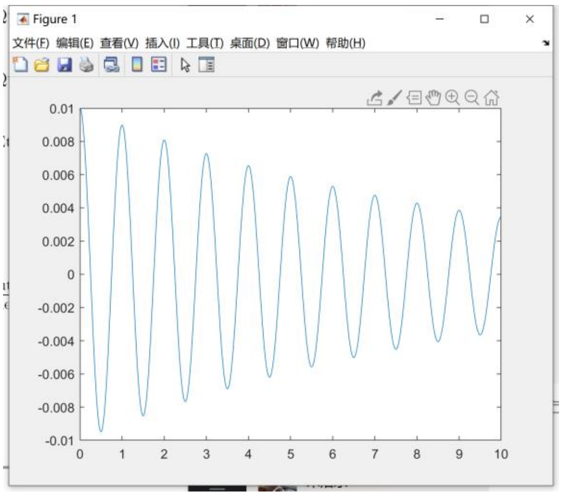
Je choisis Δt= 0.8 * 0.0064s =0.005s



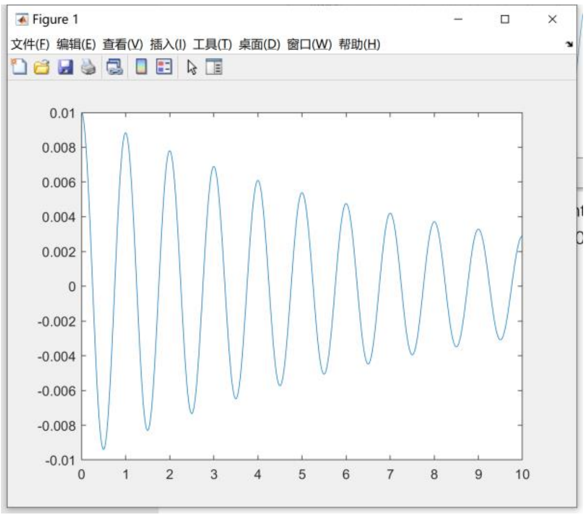
Le résultat est un peu convergent, ce qui correspond à l'oscillateur amorti.

1.1. d

1) Plus le pas de temps est petit, plus le résultat est précis.
2) Quand je prends le pas de temps à 0.001s



Le résultat est encore plus convergent.
Mais si je prends le pas de temps à 0.0001s :



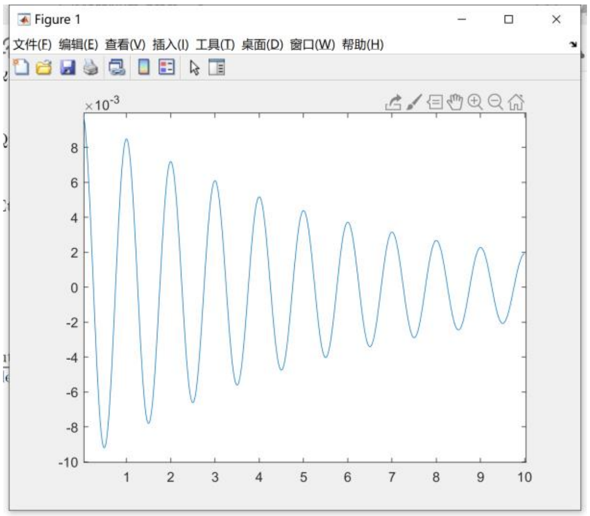
Le résultat est peu changé.
Donc à partir de 0.2 du rapport que la solution calculée présente-t-elle une précision suffisante.

1.2 Résolution avec un schéma d'EULER implicite

Le code de Matlab :

```
T0 = 1;
omega=2*pi;
epsilon=0.02;
interval=10*T0;
pas = 0.002;
steps = floor(interval/pas);
m = [0,1;-omega^2,-2*epsilon*omega];
M = inv(eye(2)-pas*m);    % Matrice d'amplification
res = zeros(2,steps);
res(:,1) = [0.01;0];
for i=1:(steps-1)
    res(:,i+1) = M*res(:,i);
end
t = pas:pas:interval;
plot(t, res(1,:))
```

D'après 1.1)
Le pas de temps critique est aussi 0.0064s :



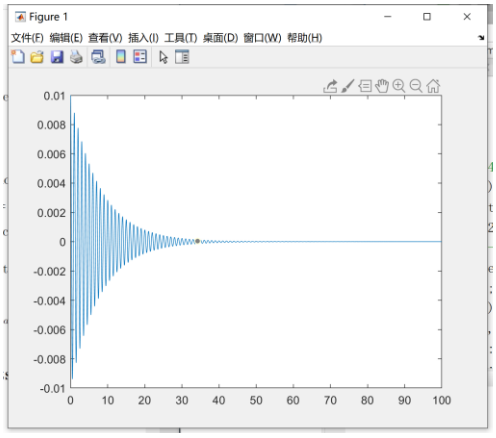
1.3 Résolution avec un schéma de RUNGE KUTTA

Le code de Matlab :

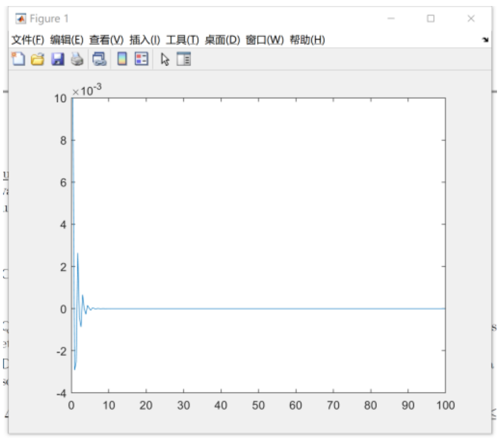
```
T0 = 1;
omega=2*pi;
epsilon=0.02;
interval=100*T0;
h=0.04; % h= 0.04 0.96 1.04
pas = h*2*sqrt(2)/omega;
steps = floor(interval/pas);
m = [0,1;-omega^2,-2*epsilon*omega];
res = zeros(2,steps);
res(:,1) = [0.01;0];
for i=1:(steps-1)
    k1 = m*res(:,i);
    k2 = m*(res(:,i)+k1*pas/2);
    k3 = m*(res(:,i)+k2*pas/2);
    k4 = m*(res(:,i)+k3*pas);
    K = (k1+2*k2+2*k3+k4)/6;
    res(:,i+1) = res(:,i)+K * pas;
end
t = pas:pas:interval;
plot(t, res(1,:))
```

1.3(a)

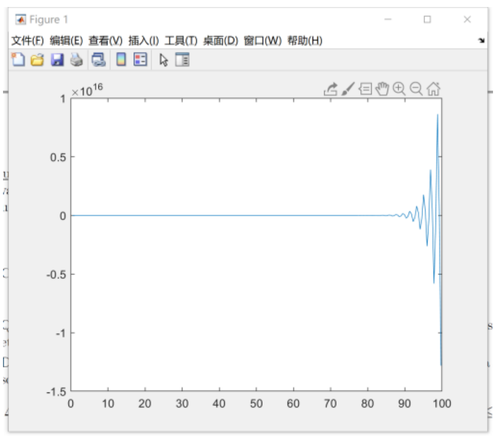
h = 0.04 :



h = 0.96 :



h = 1.04 :



Conclusion : Plus le pas du temps est petit, plus le résultat est précis. Si le pas du temps est plus grand qu'une valeur critique, ce sera la catastrophe – le résultat va exploser.

1.3 (b)

Après beaucoup d'essais, je vois une valeur critique est 0.001. Donc je peux fixer h par : h_{min}=0, h_{max}=0.001

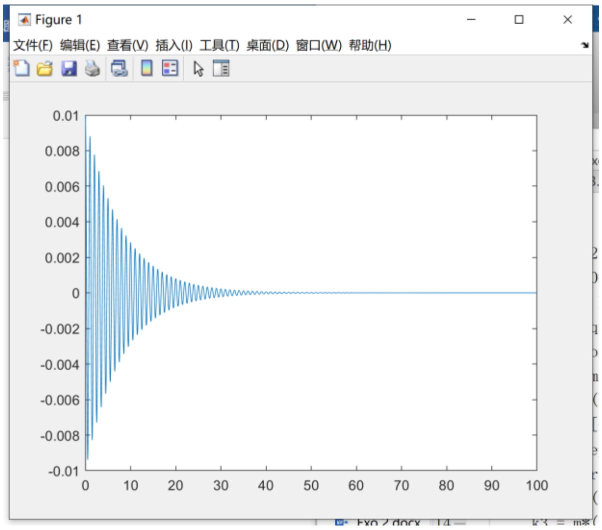


Figure 1
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
0.01
0.008
0.006
0.004
0.002
0
-0.002
-0.004
-0.006
-0.008
-0.01
0
10
20
30
40
50
60
70
80
90
100