DM 3

Nom et prénom: Maël - Ji Zhenhua

Numéro d'étudiant: SY1924113

Exercice 1: Oscillateur conservatif linéaire à un degré de liberté

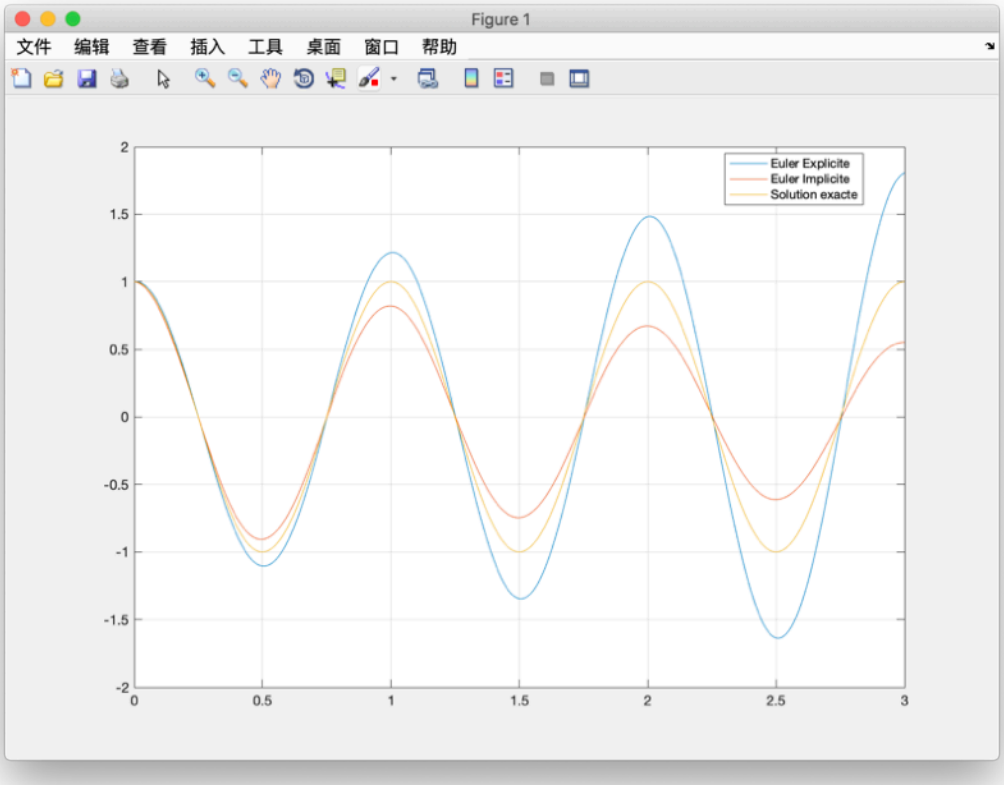
3. EULER implicite

3.1 Programmation sur Matlab

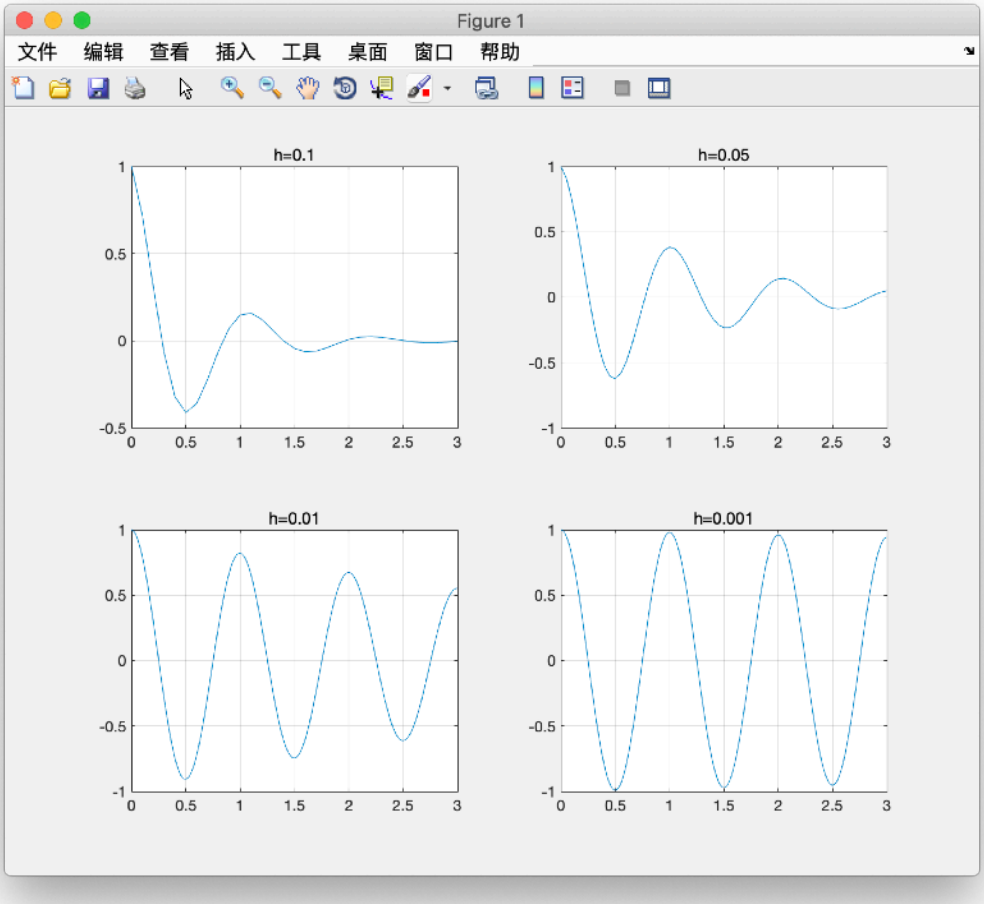
eulerIm.m

```
omega0 = 2 * pi;
T0 = 3;
h = 0.0001;

x = 0:h:T0;
y(1) = 1;
dy(1) = 0;

for n = 2:length(x)
    y(n) = (y(n-1) + h * dy(n-1))/(1 + omega0^2 * h^2);
    dy(n) = dy(n-1) + h * (-omega0^2 * y(n));
end

t = x';
q = y';

plot(t,q)
```

3.2 Dans les 3 situations, on prend le pas de temps $\Delta t = 0.01s$.



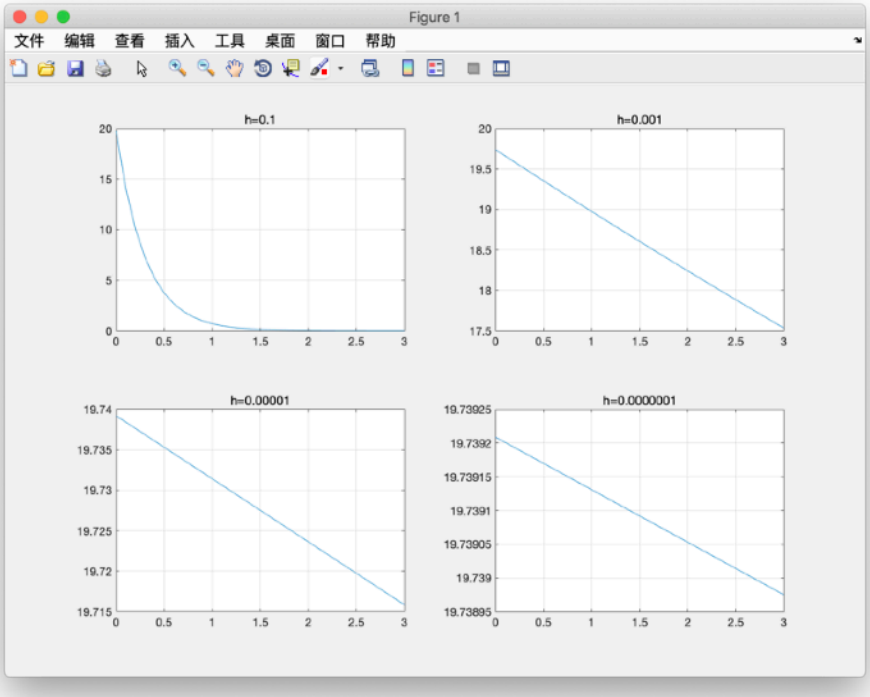
3.3 On a testé de différents pas de temps h, et on a obtenu le graphique suivant:



Dans ce graphique, on peut voir que le schéma introduit un amortissement numérique. Cependant, on peut trouver facilement que plus le pas de temps h est petit, plus l'atténuation des oscillations est faible.

3.4 De même, on a testé de différents pas de temps h pour calculer la quantité E^*, et on peut trouver le résultat.

La quantité E^* obtenue par la solution exacte est une constante, et cela obtenue par un schéma d'EULER explicite est une droite croissante, et le résultat obtenu par EULER implicite est une droite décroissante. Et plus le pas est petit, plus la pente de cette droite est petite.



3.5 Les valeurs propres de la matrice d'amplification.

La matrice d'amplification s'écrit $A = \frac{1}{1+\omega_0^2\Delta t^2}\begin{bmatrix} 1 & \Delta t \\ -\omega_0^2\Delta t & 1 \end{bmatrix}$, on peut trouver facilement les 2 valeurs propres de cette matrice, qui sont $\lambda = \frac{1}{1+\omega_0^2\Delta t^2}(1 \pm \omega_0\Delta t \cdot i)$. Le module de ces 2 valeurs propres est inférieur à 1, ce qui explique la stabilité du schéma d'EULER implicite.

4. RUNGE KUTTA

4.1 En transformant l'équation du mouvement, on obtient un système des équations différentielles du premier ordre.

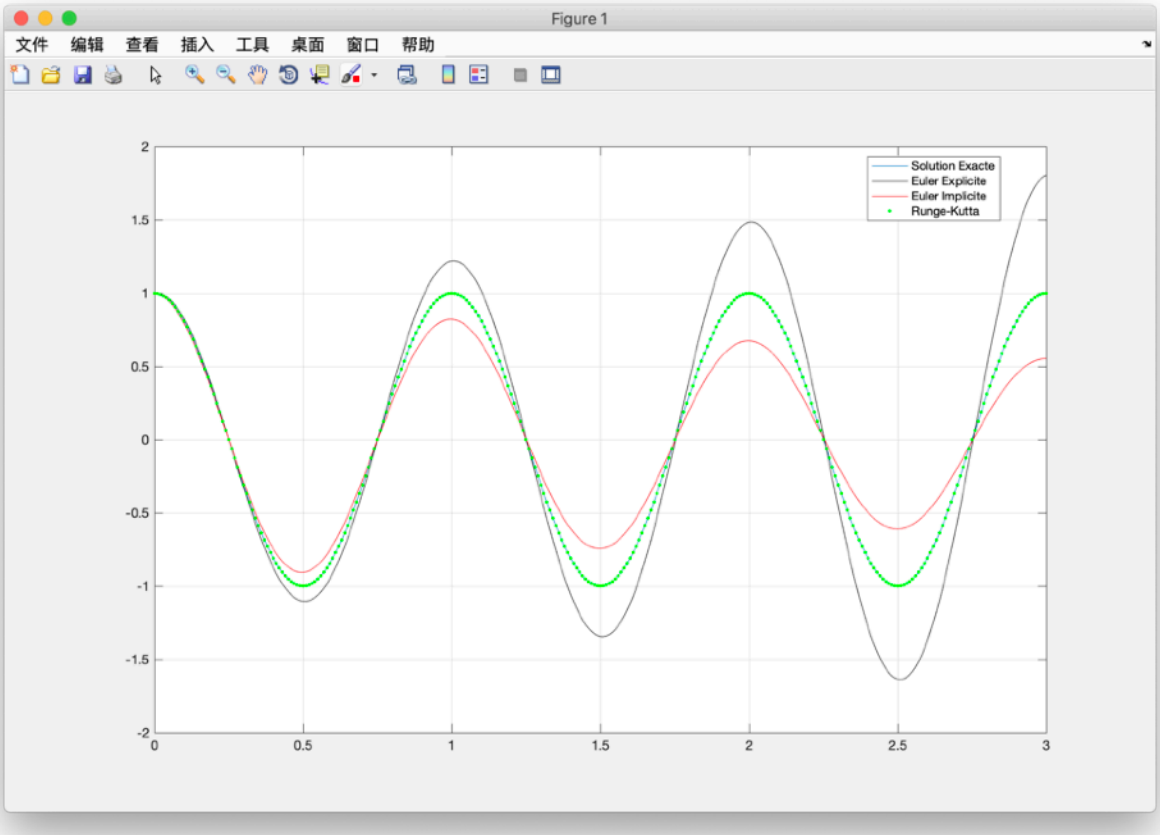
$$\begin{cases} \frac{dy}{dx} = z \\ \frac{dz}{dx} = -\omega_0^2 y \\ y(0) = 1, \quad z(0) = 0 \end{cases}$$

4.2 Programmation sur Matlab

RUNGEKUTTA.m

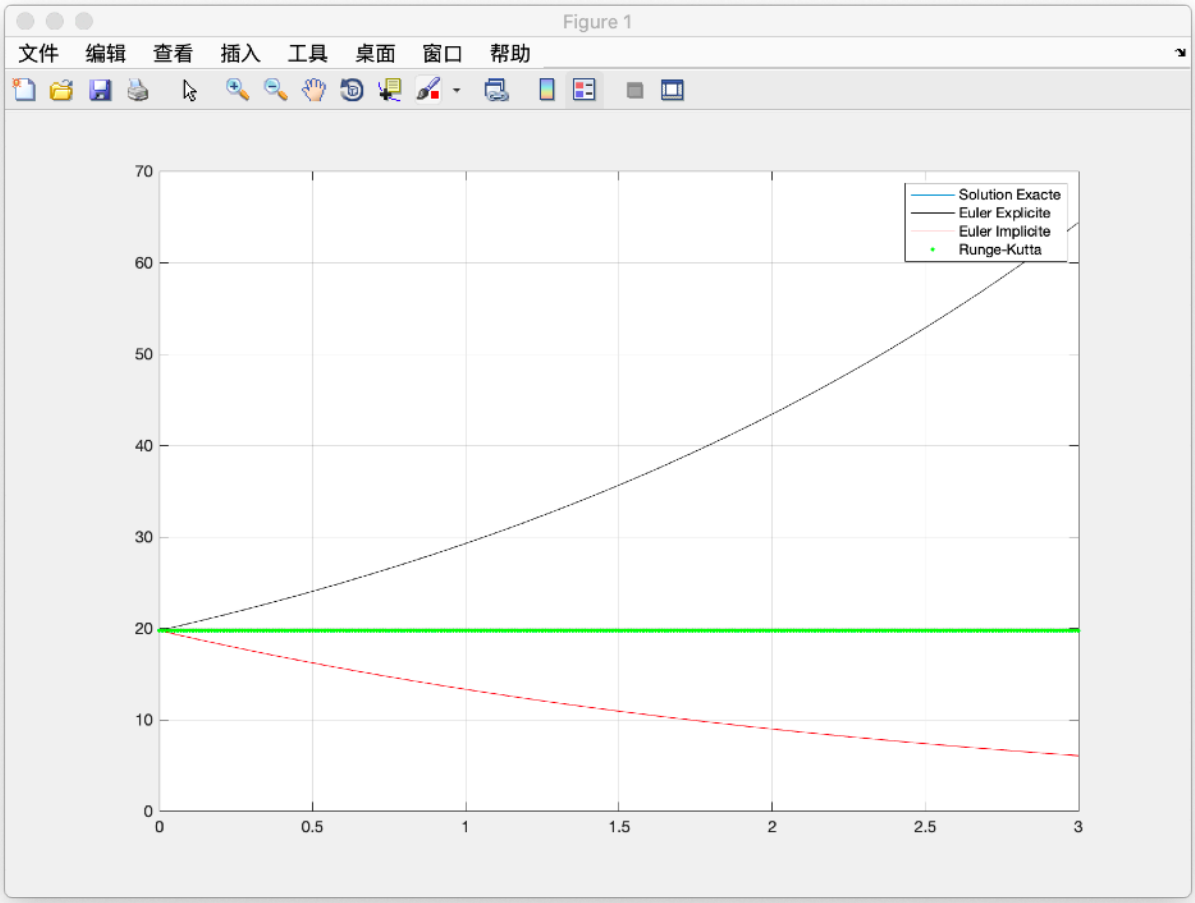
```
omega0 = 2 * pi;
T0 = 3;
h = 0.01;

x = 0:h:T0;
y(1) = 1;
z(1) = 0;

for n = 1:length(x) - 1
    k1 = z(n);
    l1 = - omega0^2 * y(n);
    k2 = z(n) + 0.5 * h * l1;
    l2 = - omega0^2 * (y(n) + 0.5 * h * k1);
    k3 = z(n) + 0.5 * h * l2;
    l3 = - omega0^2 * (y(n) + 0.5 * h * k2);
    k4 = z(n) + h *l3;
    l4 = - omega0^2 * (y(n) + h * k3);

    y(n+1) = y(n) + 1/6 * h * (k1 + 2 * k2 + 2 * k3 + k4);
    z(n+1) = z(n) + 1/6 * h * (l1 + 2 * l2 + 2 * l3 + l4);
end

t = x';
q = y';
plot(t,q);
```

4.3 Comparaison entre les différentes méthodes. On prendra $\Delta t = 0.01s$



Dans ce graphique, on peut voir facilement que la solution obtenue par RUNGE-KUTTA a une très bonne précision, elle est très proche de la solution exacte. De plus, la solution obtenue par EULER explicite diverge. En même temps, la solution obtenue par EULER implicite introduit un amortissement.

4.4 Comparaison de E^* entre les différentes méthodes. On prendra $\Delta t = 0.01s$



Dans ce graphique, on peut voir facilement que la solution obtenue par RUNGE-KUTTA a une très bonne précision. La solution obtenue par EULER explicite diverge très vite. De plus, la solution obtenue par EULER implicite converge vers 0.

Exercice 2: Oscillateur linéaire amorti à un degré de liberté

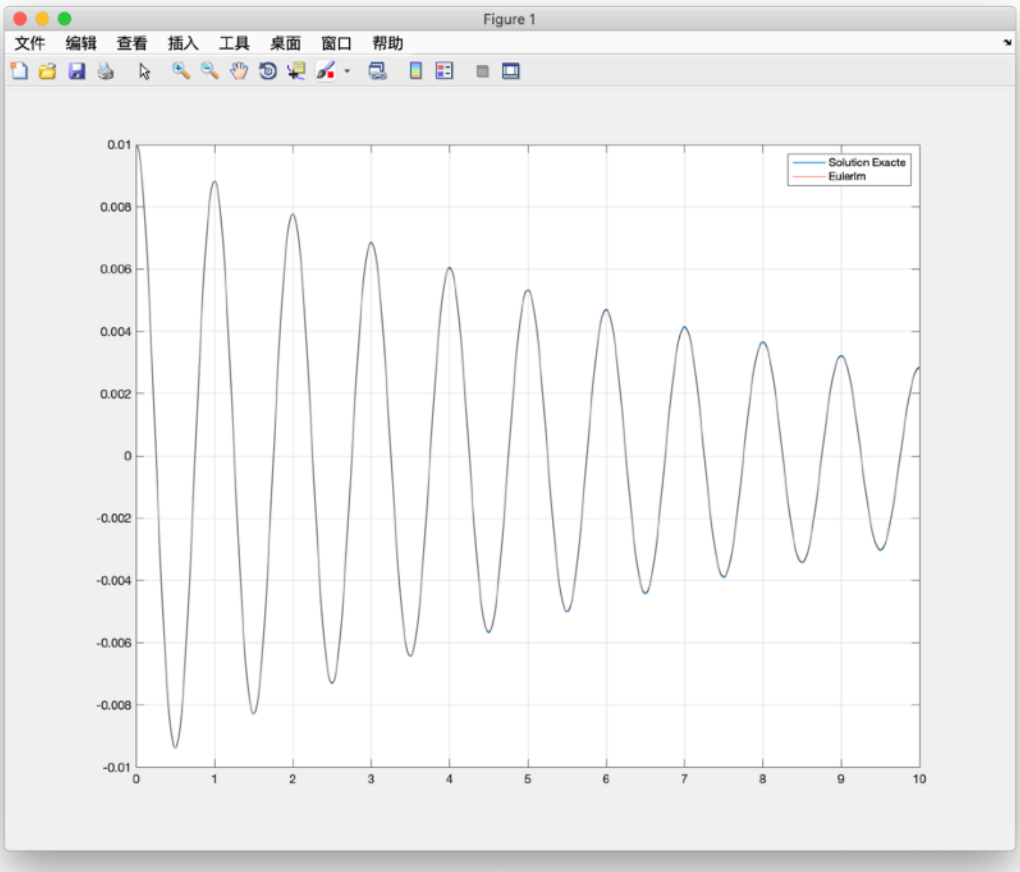
2. EULER implicite

```
omega0 = 2 * pi;
epsilon = 0.02;
T0 = 1;

h = 0.01 * 2 * epsilon / omega0;

t = 0:h:(10*T0);
y1(1) = 0.01;
y2(1) = 0;

for n = 2:length(t)
    y2(n) = (y2(n-1) - h * omega0^2 * y1(n-1))/(1 + 2 * epsilon * omega0 * h + h^2 * omega0^2);
    y1(n) = y1(n-1) + h * y2(n);
end

t = t';
x = y1';
```

Premièrement, on programme la résolution sur Matlab. Et ensuite, on essaie de différents pas de temps pour obtenir une bonne précision. On prendra

$$\Delta t = 0.01 * \frac{2\epsilon}{\omega_0}$$

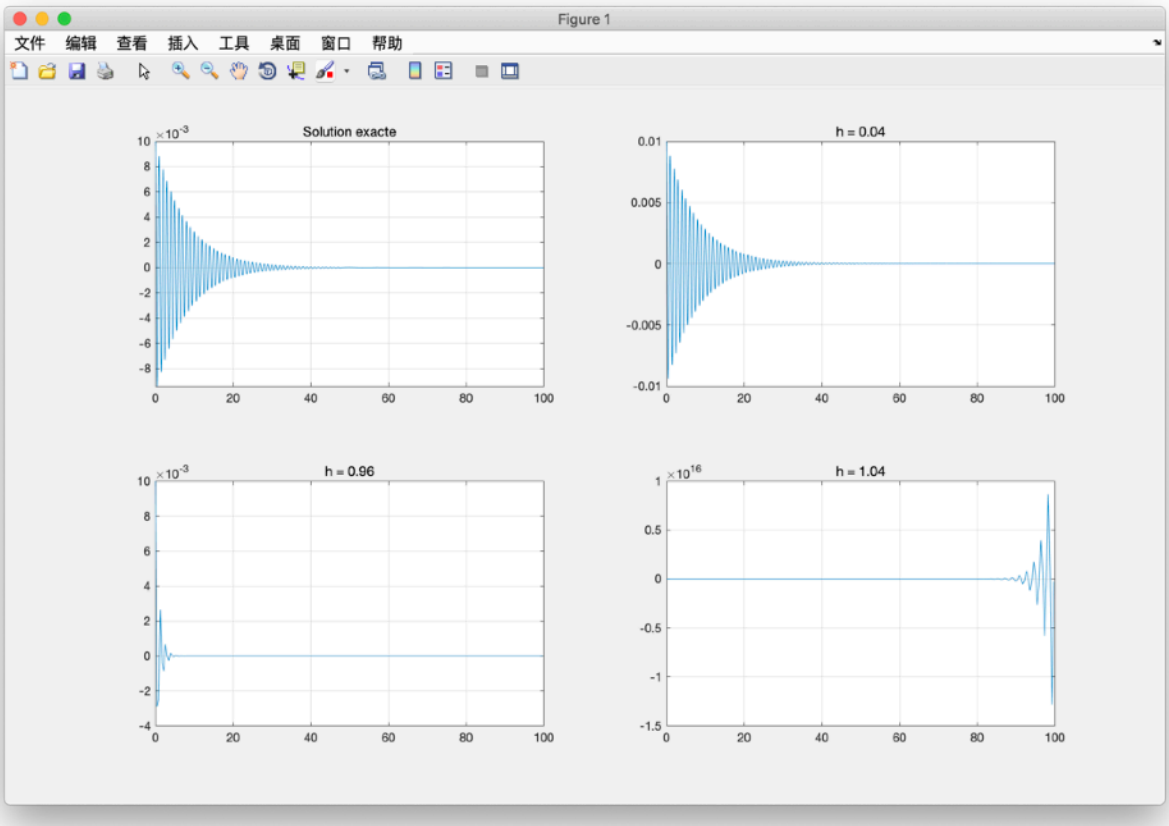
Et on peut obtenir une précision suffisante.



3. RUNGE-KUTTA

3.1 Dans ce graphique, on peut trouver facilement la différence.

- Quand $h = 0.04$, la solution reste stable, et elle a une bonne précision.
- Quand $h = 0.96$, la solution reste stable, mais la précision est très mauvaise.
- Quand $h = 1.04$, la solution n'est plus stable.



3.2 Pas de temps critique.

On a testé de différents pas de temps en changeant h, on trouve le pas de temps critique $\Delta t_c = 0.01 \times \frac{2\sqrt{2}}{\omega_0}$.