Retrouver l'équation du mouvement du pendule simple avec les équations de Lagrange

On a l'équation de Lagrange dans le cours

Lagrangien $L = E_c - E_p$

$$\boxed{\frac{d}{dt}\left[\frac{\partial L}{\partial \dot{x}_i}\right] - \frac{\partial L}{\partial x_i} = Q_i} \qquad \left(\frac{\partial E_p}{\partial \dot{x}_i} = 0\right)$$

Pour le pendule simple, on a trouvé

$$\begin{aligned} E_c(m / \mathcal{R}_0) &= \frac{m\, a^2\, \dot{\theta}^2}{2} \\ E_p(m / \mathcal{R}_0) &= -mg\, a\, \cos\theta \, + \, \mathrm{Cte} \end{aligned}$$

Donc, on a

$$L = E_c - E_p = \frac{ma^2\dot{\theta}^2}{2} + mga\cos\theta - Cte$$

On a Q=0 car il n'y a pas de force extérieure.

Donc

$ma^2\ddot{\theta} + mga\sin\theta = 0$ C'est l'équation de Lagrange

MecaNum_practice2019def.pdf

Oscillateur conservatif linéaire à un degré de liberté

1.1

$$\ddot{q} + \omega_0^2 q = 0$$

q=A*sin(w0*t)+B*cos(w0*t)

w0=2*pi q0=1 q0’=0 T0=3

A=0 B=1

q=cos(2*pi*t)

```
clear all;
w0=2*pi;
q0=1;
dq0=0;
T0=3;
n=300;
dt=T0/n;
%1.1
subplot(7,2,1)
plot(1:n,cos(2*pi*(1:n)*dt))
E=2*pi^2*ones(1,n);
subplot(7,2,2)
plot(1:n,E)
```

1.2

$$E^{\star} = \frac{1}{2}\left(\dot{q}^{\,2} + \omega_0^2\, q^2\right)$$

$E^*=1/2*(4*pi^2*sin^2(2*pi*t)+4*pi^2*cos^2(2*pi*t))=2*pi^2$

C’est une constante

2.1

On a $\ddot{q} + \omega_0^2 q = 0$ et $\begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix} + \Delta t \times \begin{vmatrix} \dot{q}_j \\ \ddot{q}_j \end{vmatrix}$

Par EULER explicite, $q_{i+1} = q_i + h \times f(t_i, q_i)$

Donc, $\begin{cases} \dot{q}_j = \dot{q}_j \\ \ddot{q}_j = -\omega_0^2 q_j \end{cases}$

Donc, $\begin{pmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{pmatrix} = \begin{pmatrix} q_j \\ \dot{q}_j \end{pmatrix} + \Delta t \times \begin{pmatrix} \dot{q}_j \\ -\omega_0^2 q_j \end{pmatrix}$

Donc, on a $\begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{bmatrix} 1 & \Delta t \\ -\omega_0^2 \Delta t & 1 \end{bmatrix} \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix}$

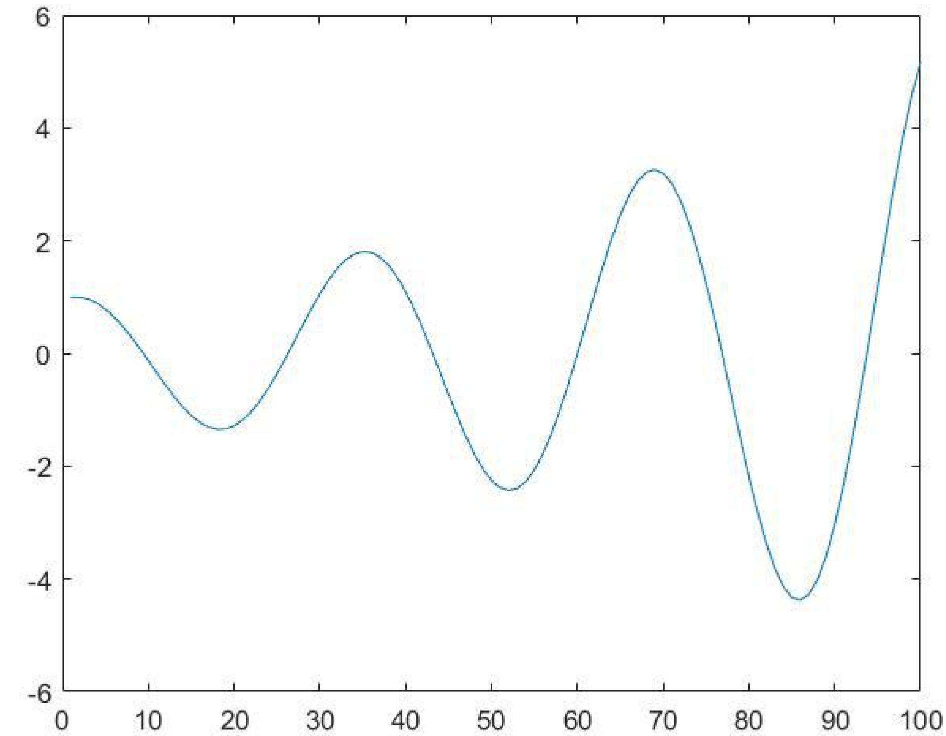
2.2

Méthode 2:

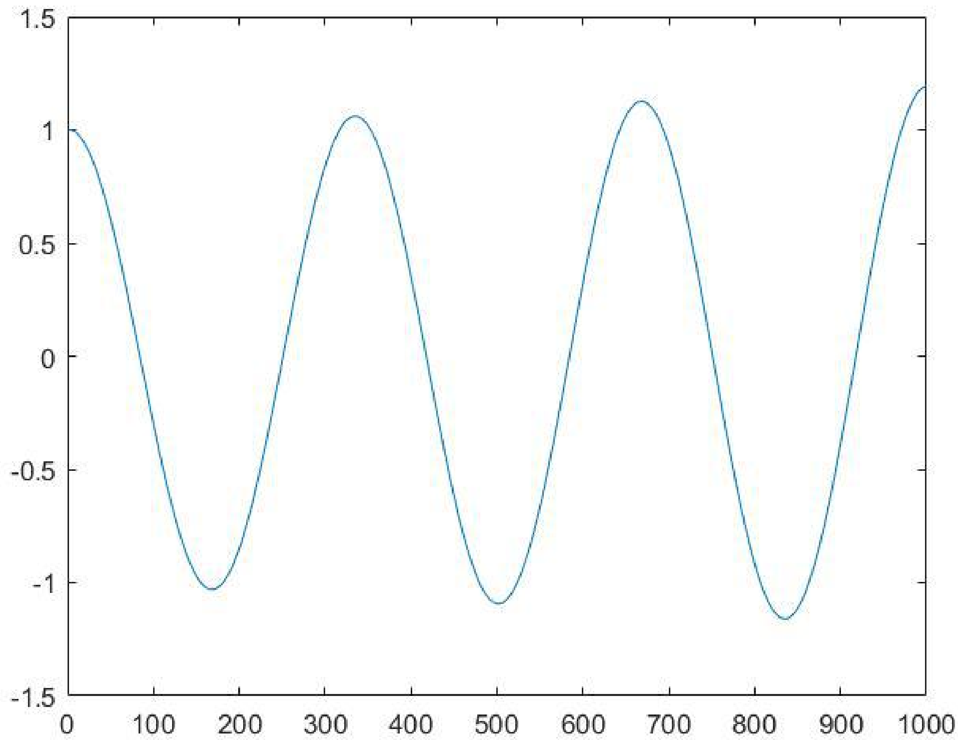
```
clear all;
w0=2*pi;
q0=1;
dq0=0;
T0=3;
n=300;
dt=T0/n;
%1.1
subplot(7,2,1)
plot(1:n,cos(2*pi*(1:n)*dt))
E=2*pi^2*ones(1,n);
subplot(7,2,2)
plot(1:n,E)

%2.2
A=[1 dt;-w0^2*dt 1];
[X1,A1]=eig(A);
U(:,1)=[q0;dq0];
E1(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U(:,i+1)=A*U(:,i);
    E1(i+1)=1/2*((U(2,i))^2+w0^2*(U(1,i))^2);
end

subplot(7,2,3)
plot(1:n,U(1,:))
subplot(7,2,4)
plot(1:n,E1)
```

2.3

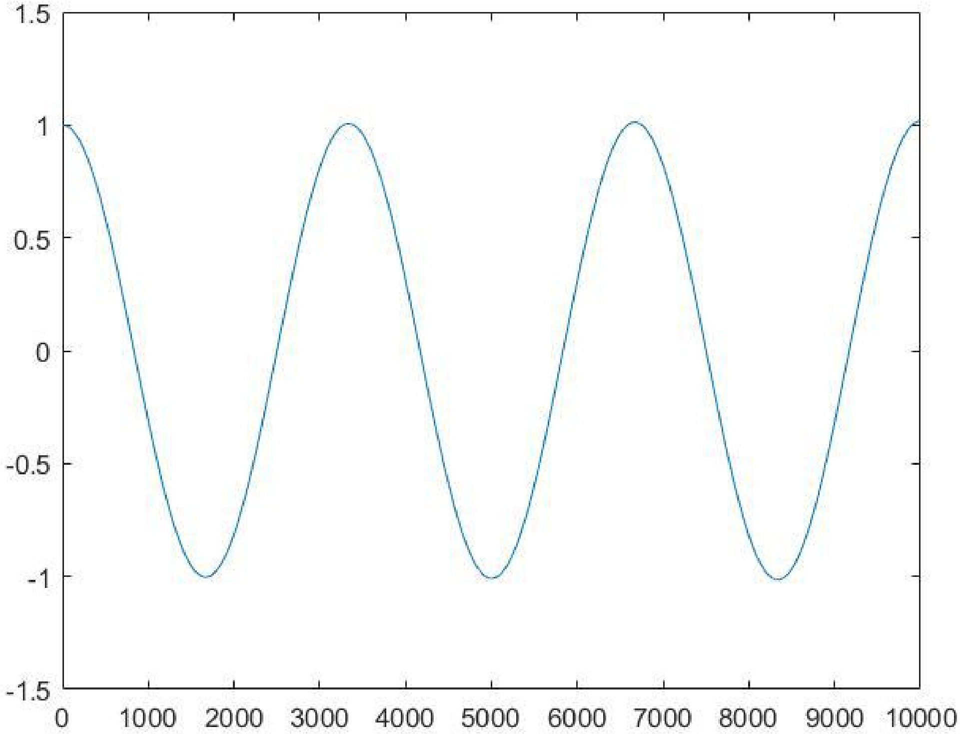
n=100



n=1000



n=10000



On peut voir que quand n devient grand c'est-à-dire que dt devient petit, la divergence devient lente.

2.4

E1
1x300 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.8171	19.8954	19.9739	20.0528	20.1319	20.2114	20.2912	20.3713

E1
1x300 double

	295	296	297	298	299	300	301	302	303	304
14	62.6177	62.8649	63.1130	63.3622	63.6123	63.8635				

On a celle à partir de la solution exacte : E*=2*pi= 19.7392

En comparant les valeurs, celle de logiciel augemente avec le temp, c'est-à-dire avec la divergence.

Donc, un plus dt petit possible est plus correcte.

2.5

$$A=\begin{pmatrix} 1 & \Delta t \\ -\omega_0^2\Delta t & 1 \end{pmatrix}$$

$$\det(\lambda I-A)=\begin{vmatrix} \lambda-1 & -\Delta t \\ \omega_0^2\Delta t & \lambda-1 \end{vmatrix}=\lambda^2-2\lambda+1+\omega^2\Delta t^2$$

$$\lambda=\frac{2\pm i\omega\Delta t}{2}$$

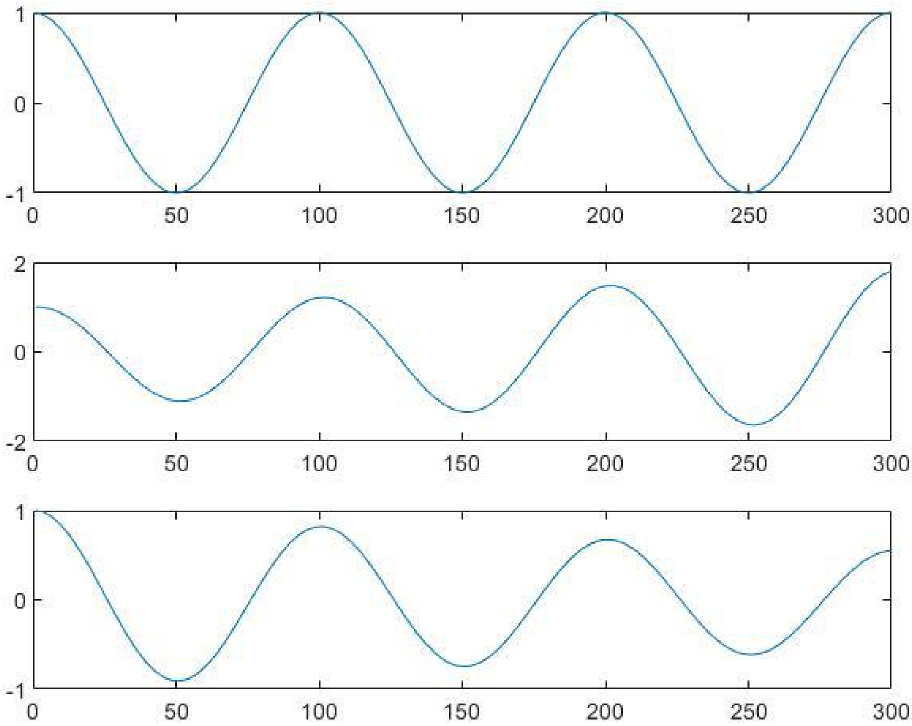
$$|\lambda|>1$$

donc, il est toujours instable.

3.1

```
clear all;
w0=2*pi;
q0=1;
dq0=0;
T0=3;
n=300;
dt=T0/n;
%1.1
subplot(7,2,1)
plot(1:n,cos(2*pi*(1:n)*dt))
E=2*pi^2*ones(1,n);
subplot(7,2,2)
plot(1:n,E)

%2.2
A=[1 dt;-w0^2*dt 1];
[X1,A1]=eig(A);
U(:,1)=[q0;dq0];
E1(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U(:,i+1)=A*U(:,i);
    E1(i+1)=1/2*((U(2,i))^2+w0^2*(U(1,i))^2);
end

subplot(7,2,3)
plot(1:n,U(1,:))
subplot(7,2,4)
plot(1:n,E1)

%3.1
B=inv([1 -dt;w0^2*dt 1]);
[X2,B1]=eig(B);
U2(:,1)=[q0;dq0];
E2(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U2(:,i+1)=B*U2(:,i);
    E2(i+1)=1/2*((U2(2,i))^2+w0^2*(U2(1,i))^2);
end
subplot(7,2,5)
plot(1:n,U2(1,:))
subplot(7,2,6)
plot(1:n,E2)
```

3.2

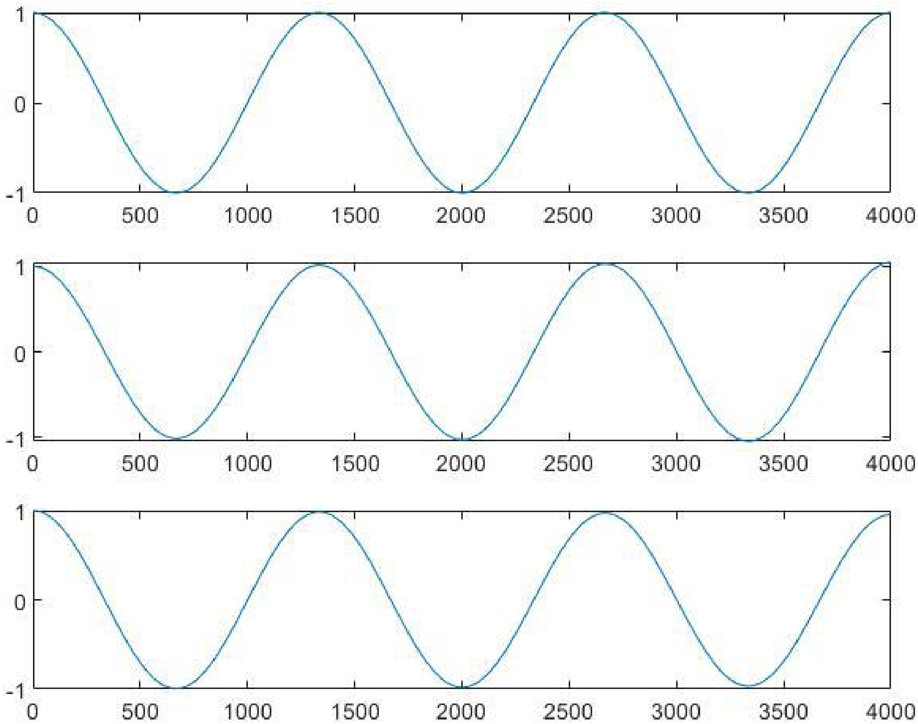


La première est la soluion exacte, la deuxième est EULER explicite et la troisème est EULER implicite. Comme on voit, la solution exacte est la plus bonne, EULER explicite diverge et EULER implicite converge.

3.3

n=4000

C’est-à-dire que dt=0.00075



3.4

On a celle à partir de la solution exacte : E*=2*pi= 19.7392

EULER explicite

E1

1x300 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.8171	19.8954	19.9739	20.0528	20.1319	20.2114	20.2912	20.3713

E1

1x300 double

	295	296	297	298	299	300	301	302	303	304
14	62.6177	62.8649	63.1130	63.3622	63.6123	63.8635				

EULER implicite

X1 E2

1x300 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.6616	19.5843	19.5073	19.4306	19.3541	19.2780	19.2022	19.1267

X1 E2

1x300 double

1	292	293	294	295	296	297	298	299	300	301
.3213	6.2965	6.2717	6.2470	6.2225	6.1980	6.1736	6.1494	6.1252	6.1011	

Comme on voit que, la valeur de E* de EULER implicite diminue.

Le plus petit dt est, le plus précis la simulation est.

3.5

-0.0000 - 0.1572i

0.9876 + 0.0000i

Les modules sont tous inférieure que 1, donc, il est stable.

4.1

$$\ddot{q} + \omega_0^2 q = 0$$

$$\begin{cases} \dot{q} = q \\ \ddot{q} = -\omega_0^2 q \end{cases}$$

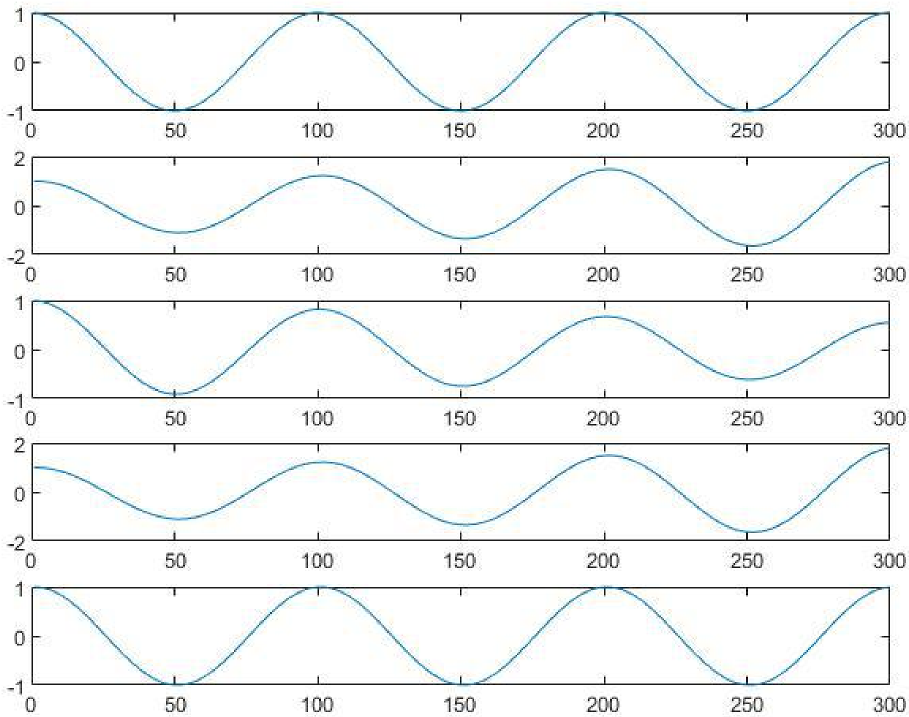
$$\begin{pmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{pmatrix} \begin{pmatrix} q_j \\ \dot{q}_j \end{pmatrix} \Delta t + \begin{pmatrix} q_j \\ \dot{q}_j \end{pmatrix}$$

4.2

```
clear all;
w0=2*pi;
q0=1;
dq0=0;
T0=3;
n=300;
dt=T0/n;
%1.1
subplot(7,2,1)
plot(1:n,cos(2*pi*(1:n)*dt))
E=2*pi^2*ones(1,n);
subplot(7,2,2)
plot(1:n,E)

%2.2
A=[1 dt;-w0^2*dt 1];
[X1,A1]=eig(A);
U(:,1)=[q0;dq0];
E1(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U(:,i+1)=A*U(:,i);
    E1(i+1)=1/2*((U(2,i))^2+w0^2*(U(1,i))^2);
end

subplot(7,2,3)
plot(1:n,U(1,:))
subplot(7,2,4)
plot(1:n,E1)

%3.1
B=inv([1 -dt;w0^2*dt 1]);
[X2,B1]=eig(B);
U2(:,1)=[q0;dq0];
E2(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U2(:,i+1)=B*U2(:,i);
    E2(i+1)=1/2*((U2(2,i))^2+w0^2*(U2(1,i))^2);
end
subplot(7,2,5)
plot(1:n,U2(1,:))
subplot(7,2,6)
plot(1:n,E2)
```

```
%4.2
%ordre 1
C=[0 1;-w0^2 0];
U3(:,1)=[q0;dq0];
E3(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U3(:,i+1)=U3(:,i)+C*U3(:,i)*dt;
    E3(i+1)=1/2*((U3(2,i))^2+w0^2*(U3(1,i))^2);
end
subplot(7,2,7)
plot(1:n,U3(1,:))
subplot(7,2,8)
plot(1:n,E3)

%ordre 4
U4(:,1)=[q0;dq0];
E4(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    k1=C*U4(:,i);
    k2=C*(U4(:,i)+1/2*k1*dt);
    k3=C*(U4(:,i)+1/2*k2*dt);
    k4=C*(U4(:,i)+k3*dt);
    U4(:,i+1)=U4(:,i)+(k1+2*k2+2*k3+k4)/6*dt;
    E4(i+1)=1/2*((U4(2,i))^2+w0^2*(U4(1,i))^2);
end
subplot(7,2,9)
plot(1:n,U4(1,:))
subplot(7,2,10)
plot(1:n,E4)
```

4.3



On peut voir que la méthode de RUNGE KUTTA de ordre 4 est plus précis par rapport à la solution exacte.

4.4

EULER explicite

E1

1x300 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.8171	19.8954	19.9739	20.0528	20.1319	20.2114	20.2912	20.3713

E1

1x300 double

	295	296	297	298	299	300	301	302	303	304
14	62.6177	62.8649	63.1130	63.3622	63.6123	63.8635				

EULER implicite

X1 E2

1x300 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.6616	19.5843	19.5073	19.4306	19.3541	19.2780	19.2022	19.1267

X1 E2

1x300 double

)1	292	293	294	295	296	297	298	299	300	301
.3213	6.2965	6.2717	6.2470	6.2225	6.1980	6.1736	6.1494	6.1252	6.1011	

RUNGE-KUTTA

E4

1x299 double

	1	2	3	4	5	6	7	8	9	10
1	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392

E4

1x299 double

	294	295	296	297	298	299	300	301	302	303
92	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392				

On peut voir que la quantité E* de RUNGE KUTTA d’ordre 4 est plus précis par rapport à celle exacte, ni converge ni diverge.

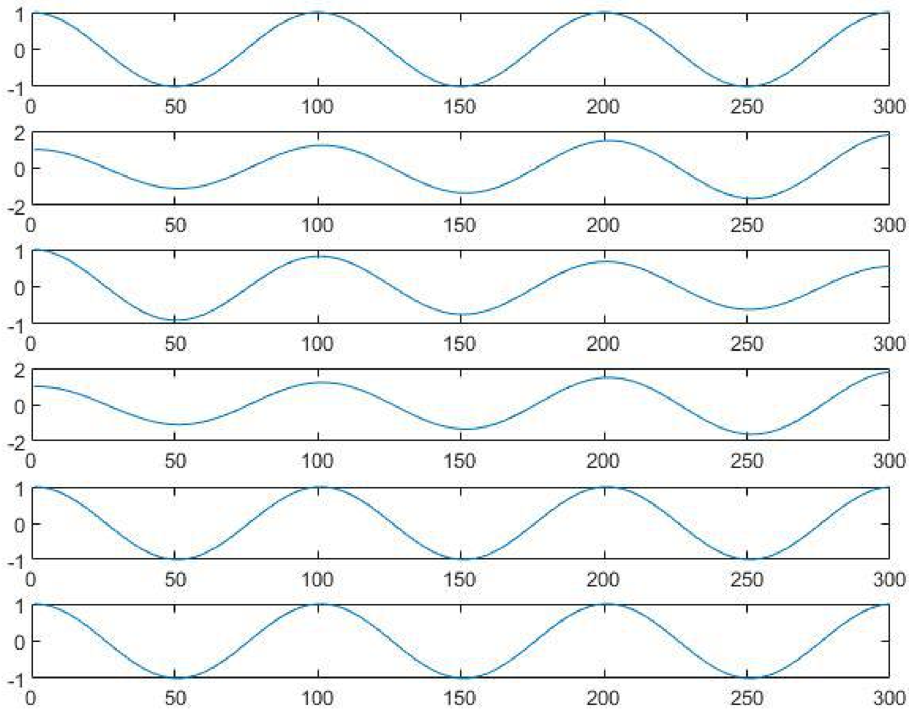
5.1.1

```
clear all;
w0=2*pi;
q0=1;
dq0=0;
T0=3;
n=300;
dt=T0/n;
%1.1
subplot(7,2,1)
plot(1:n,cos(2*pi*(1:n)*dt))
E=2*pi^2*ones(1,n);
subplot(7,2,2)
plot(1:n,E)

%2.2
A=[1 dt;-w0^2*dt 1];
[X1,A1]=eig(A);
U(:,1)=[q0;dq0];
E1(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U(:,i+1)=A*U(:,i);
    E1(i+1)=1/2*((U(2,i))^2+w0^2*(U(1,i))^2);
end

subplot(7,2,3)
plot(1:n,U(1,:))
subplot(7,2,4)
plot(1:n,E1)

%3.1
B=inv([1 -dt;w0^2*dt 1]);
[X2,B1]=eig(B);
U2(:,1)=[q0;dq0];
E2(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U2(:,i+1)=B*U2(:,i);
    E2(i+1)=1/2*((U2(2,i))^2+w0^2*(U2(1,i))^2);
end
subplot(7,2,5)
plot(1:n,U2(1,:))
subplot(7,2,6)
plot(1:n,E2)
```

```
%4.2
%ordre 1
C=[0 1;-w0^2 0];
U3(:,1)=[q0;dq0];
E3(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U3(:,i+1)=U3(:,i)+C*U3(:,i)*dt;
    E3(i+1)=1/2*((U3(2,i))^2+w0^2*(U3(1,i))^2);
end
subplot(7,2,7)
plot(1:n,U3(1,:))
subplot(7,2,8)
plot(1:n,E3)

%ordre 4
U4(:,1)=[q0;dq0];
E4(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    k1=C*U4(:,i);
    k2=C*(U4(:,i)+1/2*k1*dt);
    k3=C*(U4(:,i)+1/2*k2*dt);
    k4=C*(U4(:,i)+k3*dt);
    U4(:,i+1)=U4(:,i)+(k1+2*k2+2*k3+k4)/6*dt;
    E4(i+1)=1/2*((U4(2,i))^2+w0^2*(U4(1,i))^2);
end
subplot(7,2,9)
plot(1:n,U4(1,:))
subplot(7,2,10)
plot(1:n,E4)

%5.1
y=0.5;
b=0.25;
U5(:,1)=[q0;dq0];
E5(1)=1/2*((dq0)^2+w0^2*(q0)^2);
D=[1+b*dt^2*w0^2 0;y*dt*w0^2 1];
EE=[1-(0.5-b)*dt^2*w0^2 dt;-(1-y)*dt*w0^2 1];
F=inv(D)*EE;
[X5,F1]=eig(F);
for i=1:n-1
    U5(:,i+1)=F*U5(:,i);
    E5(i+1)=1/2*((U5(2,i))^2+w0^2*(U5(1,i))^2);
end
subplot(7,2,11)
```

```
plot(1:n,U5(1,:))
subplot(7,2,12)
plot(1:n,E5)
```

5.1.2



La méthode de NEWMARK est autant précise que la méthode de RUNGE KUTTA.

5.1.3

EULER explicite

E1
1x300 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.8171	19.8954	19.9739	20.0528	20.1319	20.2114	20.2912	20.3713

E1
1x300 double

	295	296	297	298	299	300	301	302	303	304
14	62.6177	62.8649	63.1130	63.3622	63.6123	63.8635				

EULER implicite

X1 E2
1x300 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.6616	19.5843	19.5073	19.4306	19.3541	19.2780	19.2022	19.1267

X1 E2
1x300 double

)1	292	293	294	295	296	297	298	299	300	301
.3213	6.2965	6.2717	6.2470	6.2225	6.1980	6.1736	6.1494	6.1252	6.1011	

RUNGE-KUTTA

E4
1x299 double

	1	2	3	4	5	6	7	8	9	10
1	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392

E4
1x299 double

	294	295	296	297	298	299	300	301	302	303
92	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392				

NEWMARK

E5
1x299 double

1	2	3	4	5	6	7	8	9	10
19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392

E5
1x299 double

	294	295	296	297	298	299	300	301	302	303
92	19.7392	19.7392	19.7392	19.7392	19.7392	19.7392				

On peut voir que la valeur est exactement laquelle de la solution exacte.

5.1.4

-0.000000000000000 - 0.157176725477590i

0.987570492151392 + 0.000000000000000i

Il est stable car les modules sont tous inférieures que 1

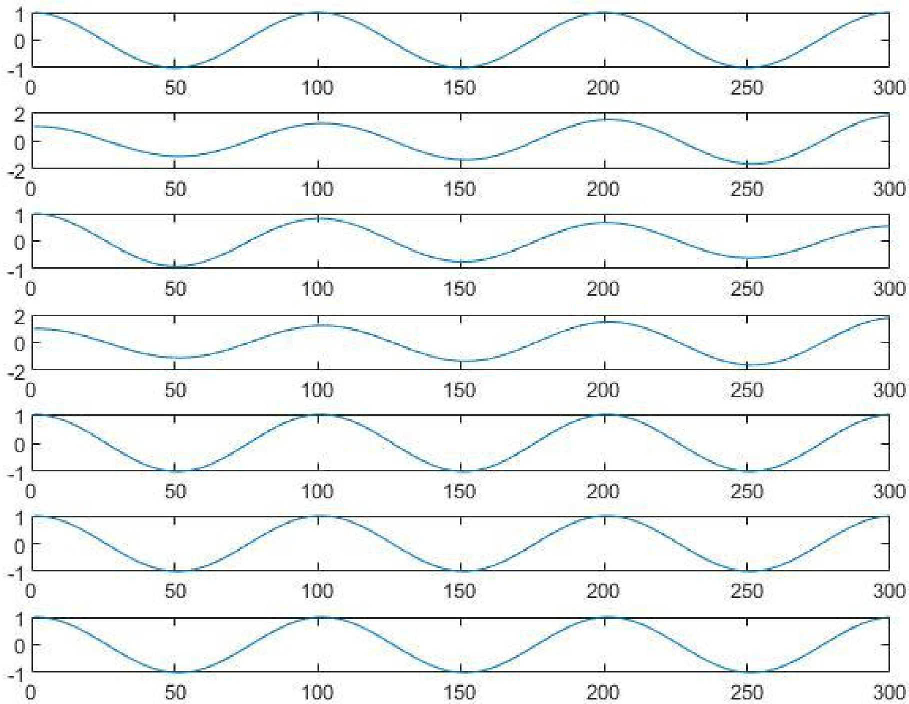
5.2.1

```
clear all;
w0=2*pi;
q0=1;
dq0=0;
T0=3;
n=300;
dt=T0/n;
%1.1
subplot(7,2,1)
plot(1:n,cos(2*pi*(1:n)*dt))
E=2*pi^2*ones(1,n);
subplot(7,2,2)
plot(1:n,E)

%2.2
A=[1 dt;-w0^2*dt 1];
[X1,A1]=eig(A);
U(:,1)=[q0;dq0];
E1(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U(:,i+1)=A*U(:,i);
    E1(i+1)=1/2*((U(2,i))^2+w0^2*(U(1,i))^2);
end

subplot(7,2,3)
plot(1:n,U(1,:))
subplot(7,2,4)
plot(1:n,E1)

%3.1
B=inv([1 -dt;w0^2*dt 1]);
[X2,B1]=eig(B);
U2(:,1)=[q0;dq0];
E2(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U2(:,i+1)=B*U2(:,i);
    E2(i+1)=1/2*((U2(2,i))^2+w0^2*(U2(1,i))^2);
end
subplot(7,2,5)
plot(1:n,U2(1,:))
subplot(7,2,6)
plot(1:n,E2)
```

```
%4.2
%ordre 1
C=[0 1;-w0^2 0];
U3(:,1)=[q0;dq0];
E3(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    U3(:,i+1)=U3(:,i)+C*U3(:,i)*dt;
    E3(i+1)=1/2*((U3(2,i))^2+w0^2*(U3(1,i))^2);
end
subplot(7,2,7)
plot(1:n,U3(1,:))
subplot(7,2,8)
plot(1:n,E3)

%ordre 4
U4(:,1)=[q0;dq0];
E4(1)=1/2*((dq0)^2+w0^2*(q0)^2);
for i=1:n-1
    k1=C*U4(:,i);
    k2=C*(U4(:,i)+1/2*k1*dt);
    k3=C*(U4(:,i)+1/2*k2*dt);
    k4=C*(U4(:,i)+k3*dt);
    U4(:,i+1)=U4(:,i)+(k1+2*k2+2*k3+k4)/6*dt;
    E4(i+1)=1/2*((U4(2,i))^2+w0^2*(U4(1,i))^2);
end
subplot(7,2,9)
plot(1:n,U4(1,:))
subplot(7,2,10)
plot(1:n,E4)

%5.1
y=0.5;
b=0.25;
U5(:,1)=[q0;dq0];
E5(1)=1/2*((dq0)^2+w0^2*(q0)^2);
D=[1+b*dt^2*w0^2 0;y*dt*w0^2 1];
EE=[1-(0.5-b)*dt^2*w0^2 dt;-(1-y)*dt*w0^2 1];
F=inv(D)*EE;
[X5,F1]=eig(F);
for i=1:n-1
    U5(:,i+1)=F*U5(:,i);
    E5(i+1)=1/2*((U5(2,i))^2+w0^2*(U5(1,i))^2);
end
subplot(7,2,11)
```

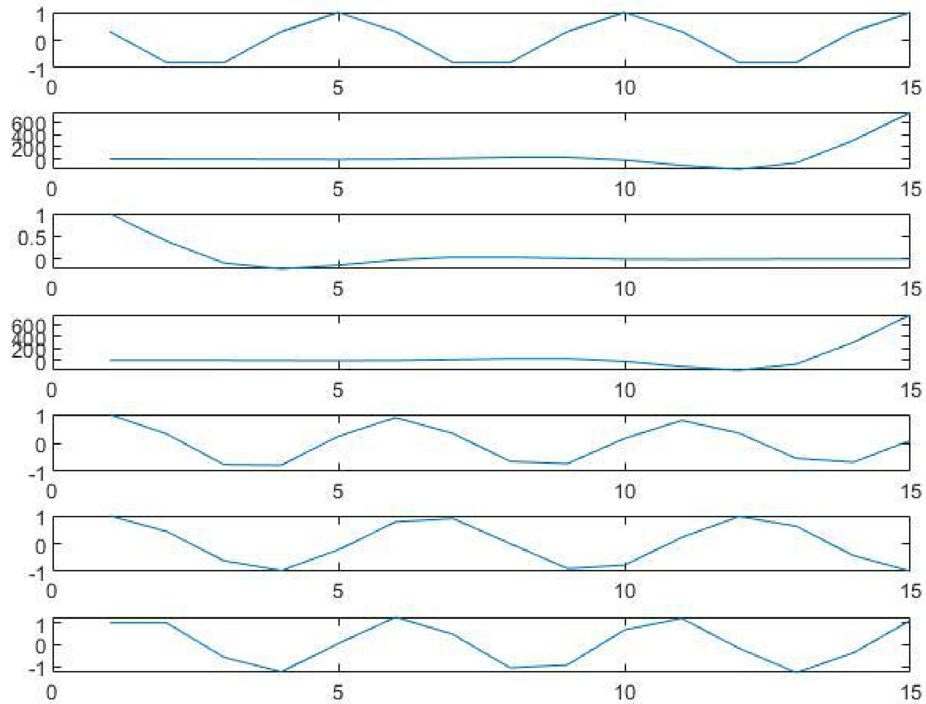
```
plot(1:n,U5(1,:))
subplot(7,2,12)
plot(1:n,E5)

%5.2
y2=0.5;
b2=0;
U6(:,1)=[q0;dq0;0];
E6(1)=1/2*((dq0)^2+w0^2*(q0)^2);
G=[1 0 0;w0^2 0 1;0 1 -dt/2];
H=[1 dt dt^2/2;0 0 0;0 1 dt/2];
[X6,G6]=eig(inv(G)*H);

for i=1:n-1
   U6(1,i+1)=U6(1,i)+dt*U6(2,i)+dt^2/2*U6(3,i);
   U6(3,i+1)=-w0^2*U6(1,i+1);
   U6(2,i+1)=U6(2,i)+dt/2*(U6(3,i)+U6(3,i+1));
   E6(i+1)=1/2*((U6(2,i))^2+w0^2*(U6(1,i))^2);
end

subplot(7,2,13)
plot(1:n,U6(1,:))
subplot(7,2,14)
plot(1:n,E6)
```

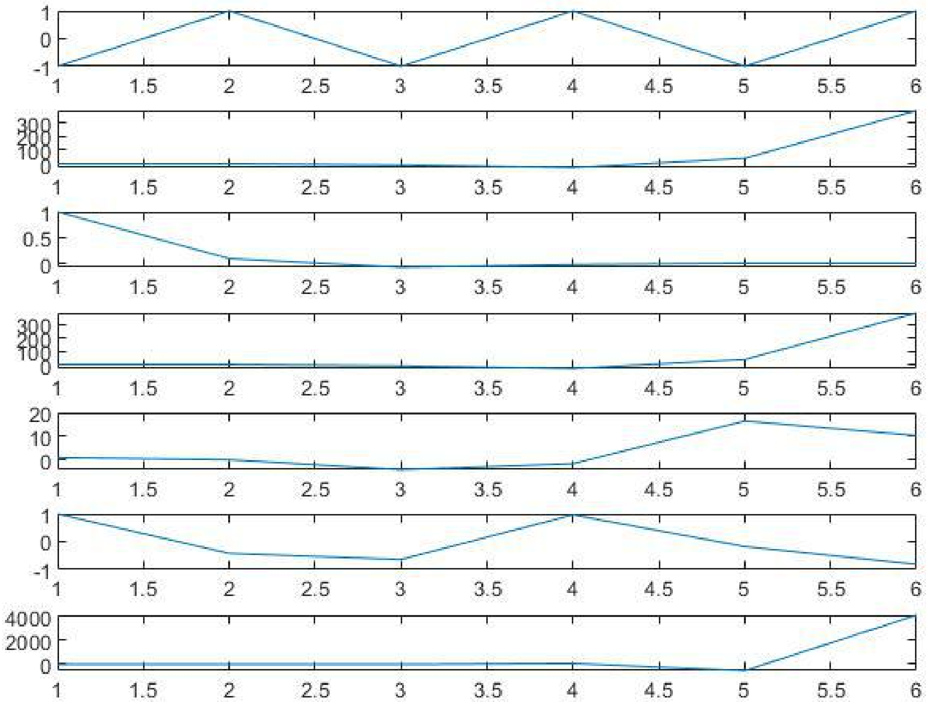
5.2.2



C’est plus précis.

5.2.3





Il y a plus de l'oscillation quand b devient petit.

5.2.4

Les valeurs propres :

2.523478242613594e-17 + 0.000000000000000e+00i

0.998026079119782 + 0.062800839140846i

0.998026079119782 - 0.062800839140846i

ε =0

et $\Delta t_c = \dfrac{2\varepsilon}{\omega_0}$

donc $\Delta t_c = 0$ et $\alpha = 0$

dans le logiciel on trouve que

n=37

dt=0.0811

a=0.2547

il converge critiquement, mais c'est pas précis.