Mécanique & Vibrations
Intégration en temps des équations du mouvement (séance 3)

Patrick ROZYCKI, Pascal COSSON, Laurent GORNET

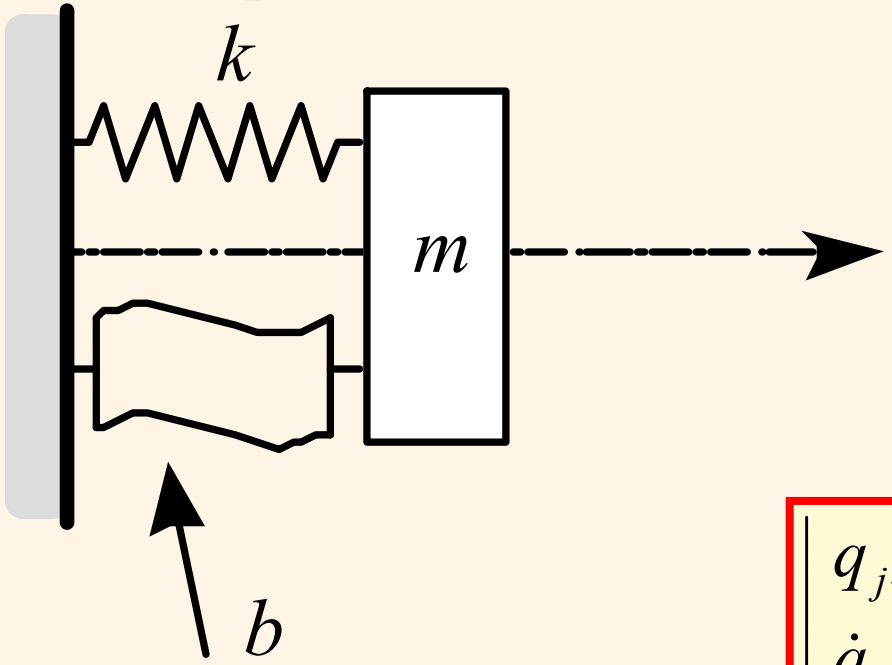
Ecole Centrale de Nantes

Cours n°3
Dynamique des structures

Euler implicite

RUNGE KUTTA, NEWMARK

✓ remarque : cas de l'oscillateur amorti



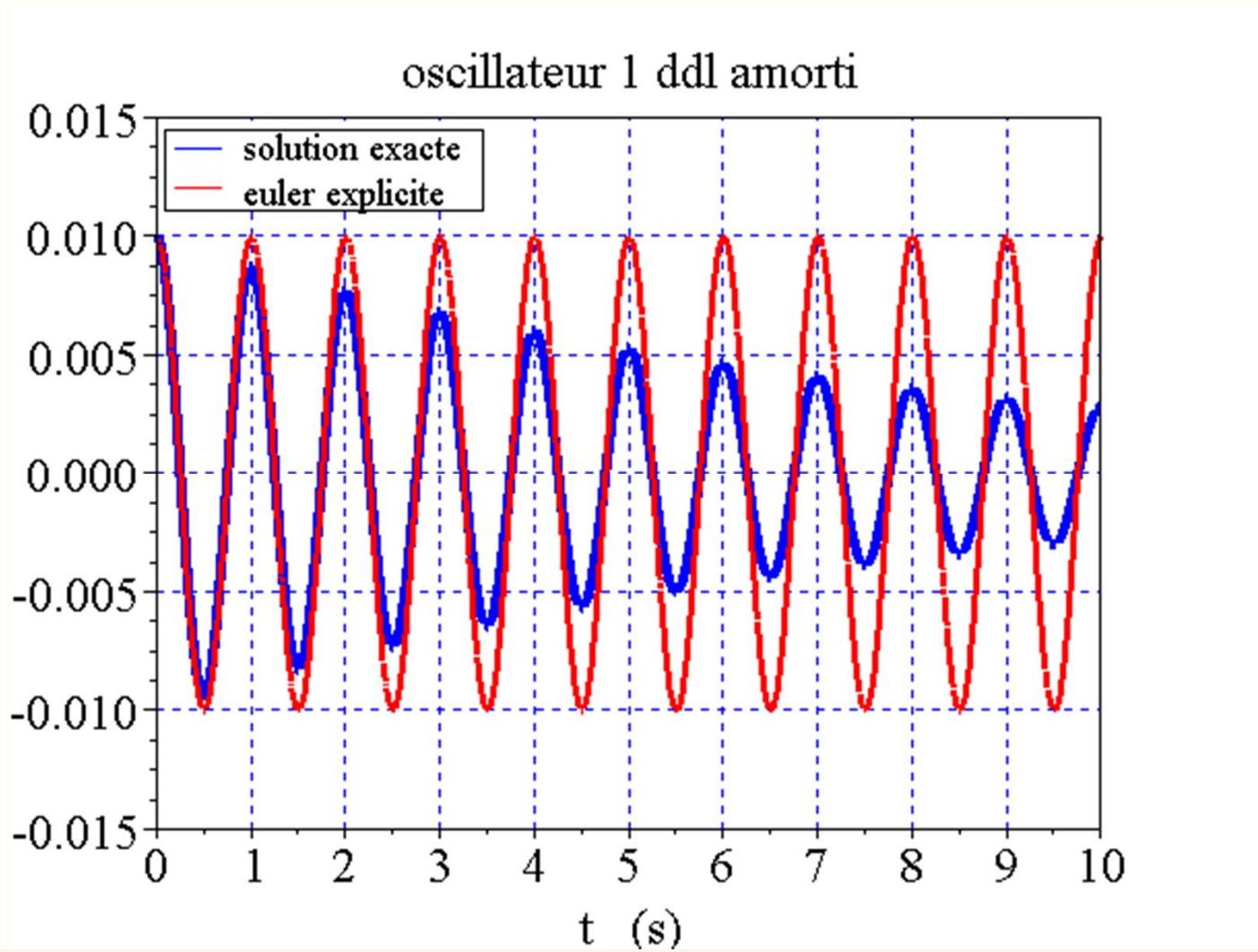
$$\ddot{q}(t) + 2\varepsilon\,\omega_0 \dot{q}(t) + \omega_0^{\,2} q(t) = 0$$

$$\omega_0^{\,2} = \frac{k}{m} \qquad b = 2\,\varepsilon\,\omega_0\,m$$

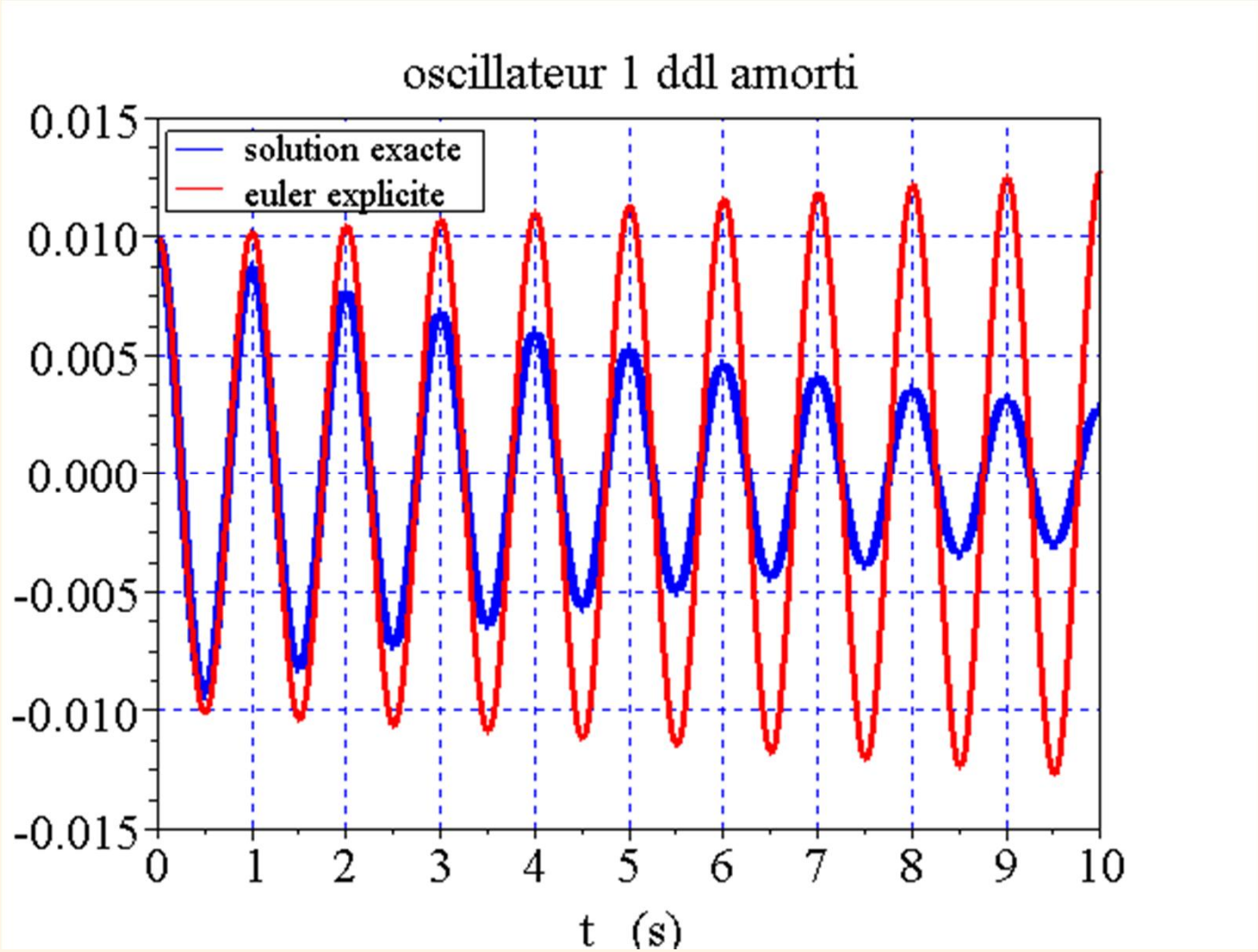
$$\begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{bmatrix} 1 & \Delta t \\ -\omega_0^{\,2}\,\Delta t & 1 - 2\varepsilon\,\omega_0\,\Delta t \end{bmatrix} \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix}$$

$$0 < \varepsilon < 1 \quad \Rightarrow \quad \lambda = 1 - \varepsilon\,\omega_0\,\Delta t \pm i\,\omega_0\,\Delta t \left[1 - \varepsilon^2\right]^{1/2}$$

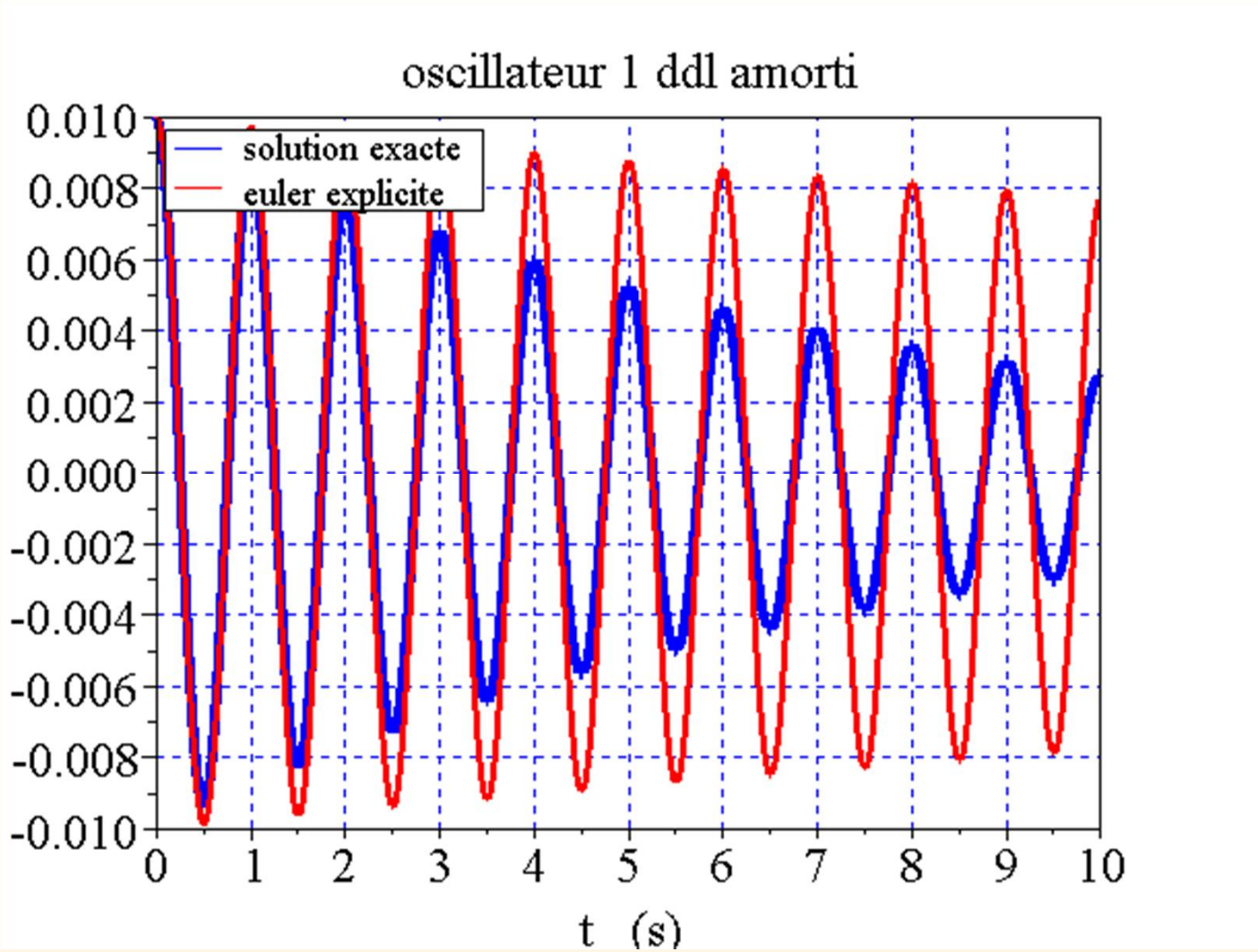
$$|\lambda| < 1 \quad \Rightarrow \quad \Delta t < \frac{2\varepsilon}{\omega_0}$$



$$\Delta t = \Delta t_c = \frac{2\varepsilon}{\omega_0}$$



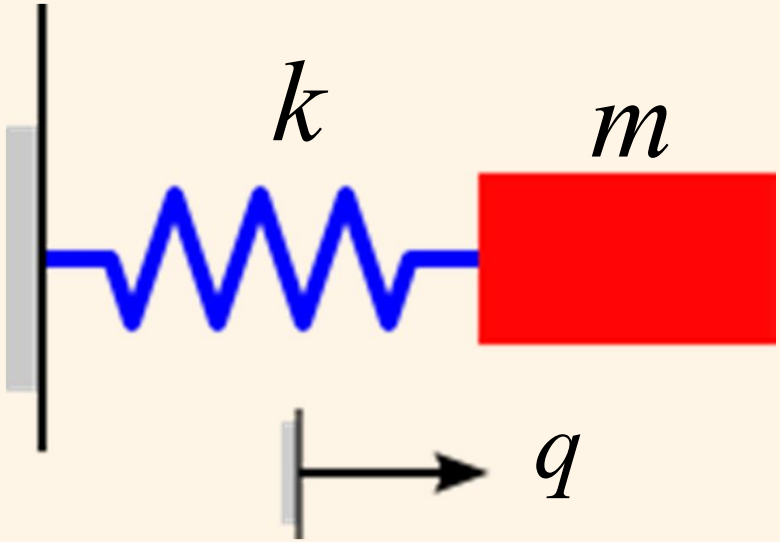
$$\Delta t = 1.2 \times \Delta t_c = 1.2 \times \frac{2\varepsilon}{\omega_0}$$



$$\Delta t = 0.8 \times \Delta t_c = 0.8 \times \frac{2\varepsilon}{\omega_0}$$

Exemple utilisé pour présenter les schémas d'intégration

➢ oscillateur linéaire



$$\ddot{q} + \frac{k}{m} q = 0 \quad \text{soit} \quad \boxed{\ddot{q} + \omega_0^{\,2} q = 0} \quad \text{avec} \quad \omega_0^{\,2} = \frac{k}{m}$$

❖ à l'instant t_j $\quad q_j = q(t_j) \quad \dot{q}_j = \dot{q}(t_j) \quad \ddot{q}_j = \ddot{q}(t_j)$

➢ Schéma d'EULER implicite

$$\begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix} + \Delta t \begin{vmatrix} \dot{q}_{j+1} \\ \ddot{q}_{j+1} \end{vmatrix}$$

Avec équation : $\ddot{q} + \omega_0^{\,2} q = 0$

- à l'instant t_j q_j $\dot{q}_j$ $\ddot{q}_j$ connus
- à l'instant t_{j+1} q_{j+1} $\dot{q}_{j+1}$ $\ddot{q}_{j+1}$ à déterminer

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

$$\begin{vmatrix} q_j \\ \dot{q}_j \\ \ddot{q}_j \end{vmatrix} \longrightarrow \begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \\ \ddot{q}_{j+1} \end{vmatrix}$$

➢ Schéma d'EULER implicite

$$\begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix} + \Delta t \begin{vmatrix} \dot{q}_{j+1} \\ \ddot{q}_{j+1} \end{vmatrix}$$

$$\ddot{q}_{j+1} + \omega_0^{\,2} q_{j+1} = 0 \quad \Longrightarrow \quad \ddot{q}_{j+1} = -\omega_0^{\,2} q_{j+1}$$

$$\begin{bmatrix} 1 & -\Delta t \\ +\omega_0^{\,2}\Delta t & 1 \end{bmatrix} \begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix} \quad \Longrightarrow \quad \begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \underbrace{\frac{1}{1+\omega_0^{\,2}\Delta t^2}\begin{bmatrix} 1 & \Delta t \\ -\omega_0^{\,2}\Delta t & 1 \end{bmatrix}} \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix}$$

$[A]$ matrice d'amplification

$$\lambda_1 = \frac{1 - i\omega_0 \Delta t}{1+\omega_0^{\,2}\Delta t^2} \qquad \lambda_2 = \frac{1 + i\omega_0 \Delta t}{1+\omega_0^{\,2}\Delta t^2}$$

$$|\lambda_i| < 1 \qquad i = 1,2$$

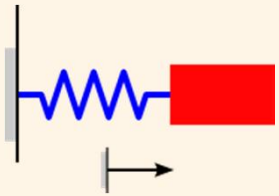
schéma inconditionnellement stable

➢ Stabilité du Schéma d'EULER implicite avec Matlab Symbolique

```
clear all; close all; clc;
 dt1= sym('dt1','real');
 w0= sym('w0','real');

A = [1  ,-dt1 ,0 ; 0 ,1 ,-dt1; w0 * w0,  0 ,  1]
B = [1 ,0 ,0 ; 0 ,1, 0; 0, 0 ,0]
C= A\B
% identique C =  inv(A) * B
% Vecteurs et valeurs propres

[z,d] = eig(C) ; d ;
simplify(d)
mo=abs(d)
% module <1 ? Oui schéma inconditionnellement stable
eval(mo)
```

- **méthode 1 – Euler implicite sans matrice d'amplification**

$$\begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix} + \Delta t \begin{vmatrix} \dot{q}_{j+1} \\ \ddot{q}_{j+1} \end{vmatrix}$$



$$\ddot{q}_{j+1} = -\omega_0^{\,2} q_{j+1}$$

Ecrire les équations – Euler implicite

Programmer en Matlab

Matlab Classique

Matlab vectoriel

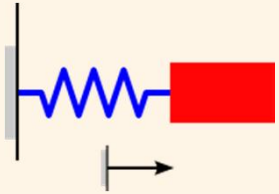
- **méthode 2 – Euler implicite avec matrice d'amplification**

Programmer en Matlab

- **méthode 1 – Euler implicite sans matrice d'amplification**

```
t2 = (0:dt2:T0)'
np2    = size(t2,1)
q2     = zeros(np2,1)
dq2    = zeros(np2,1) ;
energ2 = zeros(np2,1);
q2(1)  = q0
dq2(1) = dq0

for inc = 2 : np2
  q2(inc)  = (q2(inc-1) + dt2 * dq2(inc-1))/(1 + w0c * dt2 * dt2)
  ddqc     = -w0c * q2(inc)
  dq2(inc) = dq2(inc-1) + dt2 * ddqc
end
energ2 = 0.5*(dq2 .* dq2 + w0c * (q2.^2))
                                  plot(t2,q2,'b-','Linewidth',3)
```

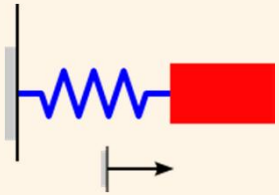
$$\begin{vmatrix} q_{j+1} \\ \dot{q}_{j+1} \end{vmatrix} = \begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix} + \Delta t \begin{vmatrix} \dot{q}_{j+1} \\ \ddot{q}_{j+1} \end{vmatrix}$$



$$\ddot{q}_{j+1} = -\omega_0^{\,2} q_{j+1}$$

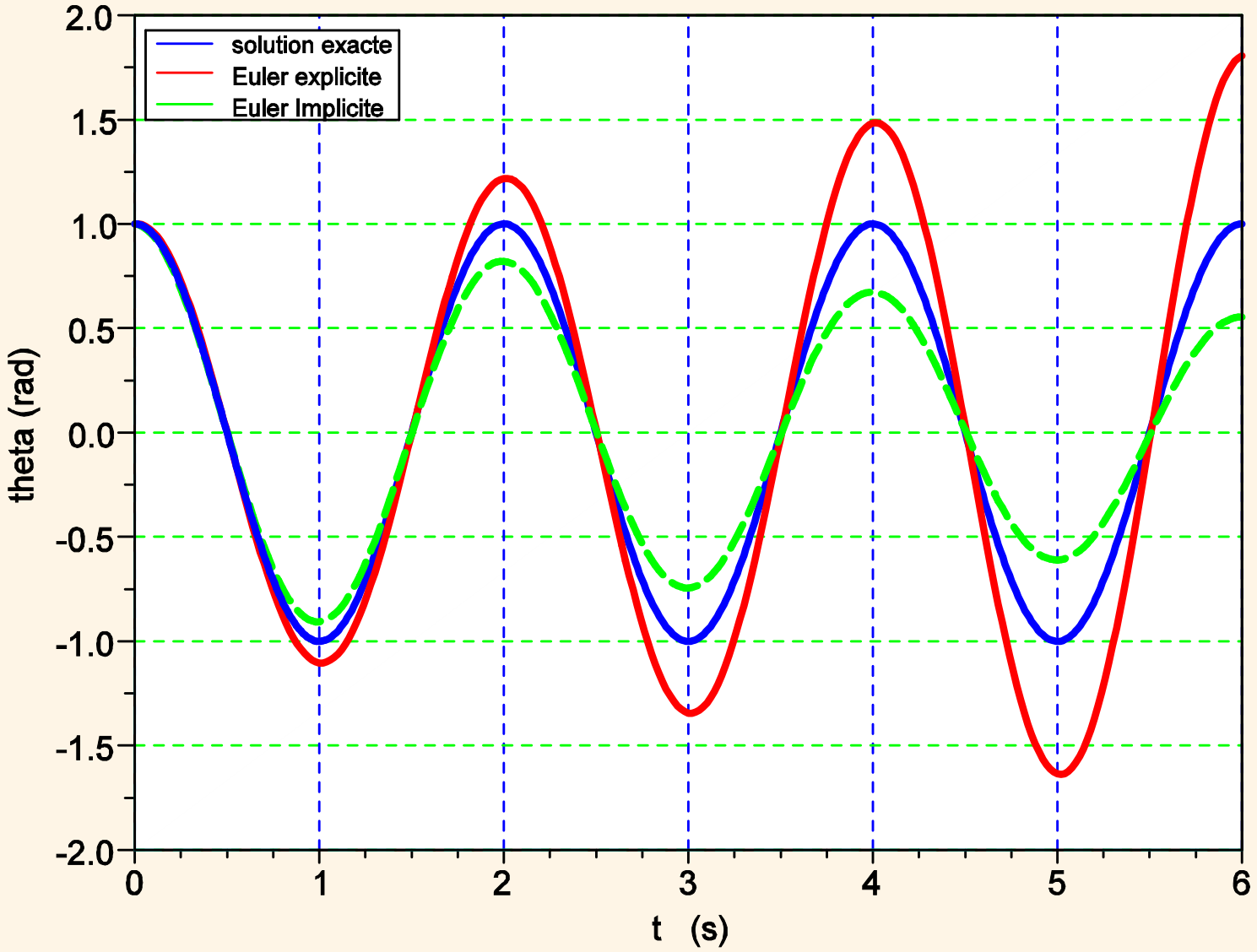
- **méthode 2 – Euler implicite avec matrice d'amplification**

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



```
q = [q0;dq0]
q2b = zeros(np2,1)
dq2b = zeros(np2,1)

q2b(1) = q0
dq2b(1) = dq0

A = [1 , dt2 ; -w0c * dt2  , 1]
A = A / (1 + w0c * dt2 * dt2)
```

```
for inc = 2 : np2
    q   = A * q
    q2b(inc) = q(1)
    dq2b(inc) = q(2)
end;

energ2m = 0.5*(dq2b  .* dq2b
+ w0c * (q2b  .^2))

   plot(t2,q2b,'b-','Linewidth',3)
```


solution exacte
Euler explicite
Euler Implicite
theta (rad)
t (s)
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
0
1
2
3
4
5
6

$\Delta t = 20$ ms

➢ Critères d'évaluation d'un schéma d'intégration

✓ consistance

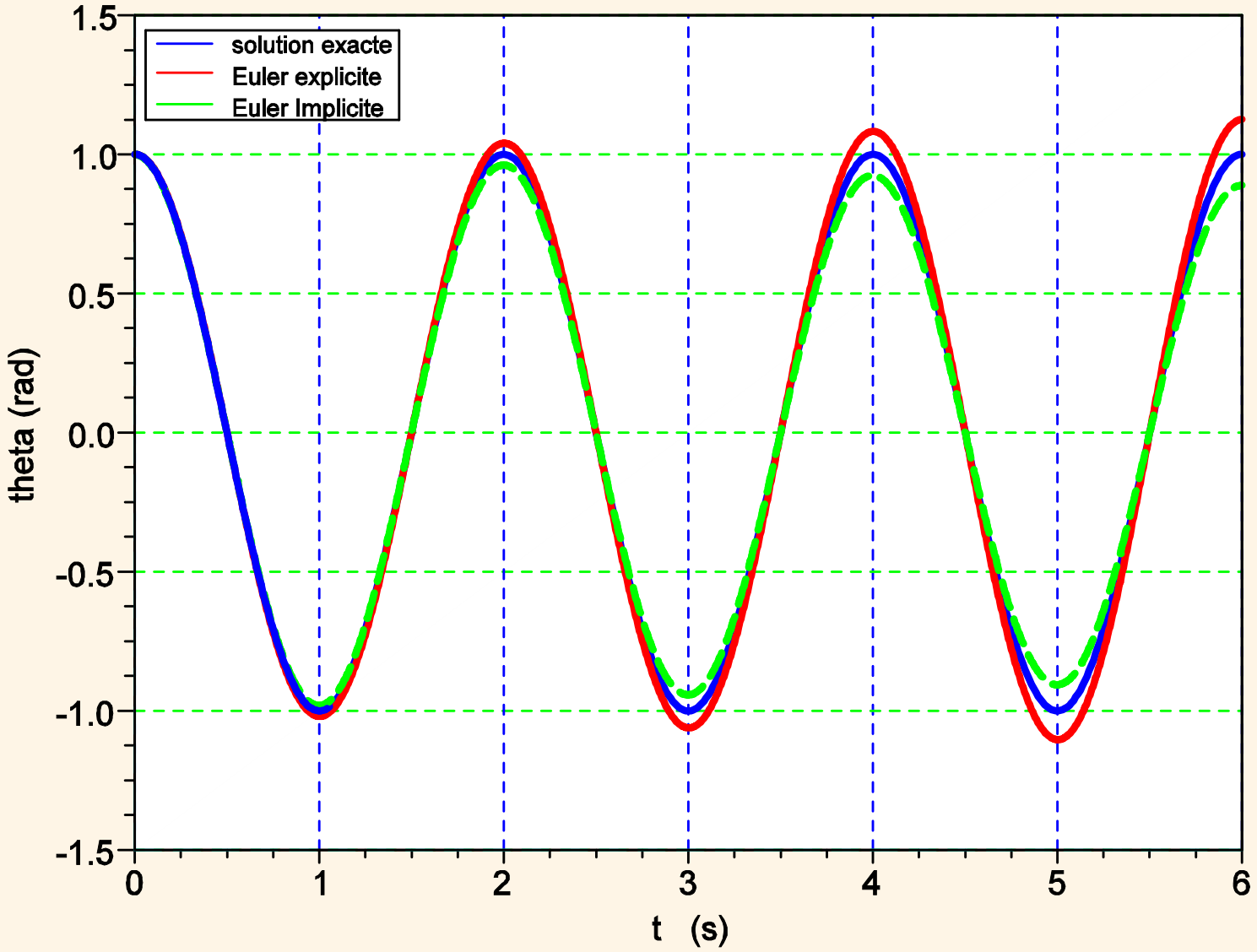
$$\dot{q}(t_j) = \lim_{\Delta t \to 0} \dot{q}_j \qquad \ddot{q}(t_j) = \lim_{\Delta t \to 0} \ddot{q}_j$$

✓ stabilité

Le module des valeurs propres de la matrice d'amplification est inférieur à 1

✓ précision

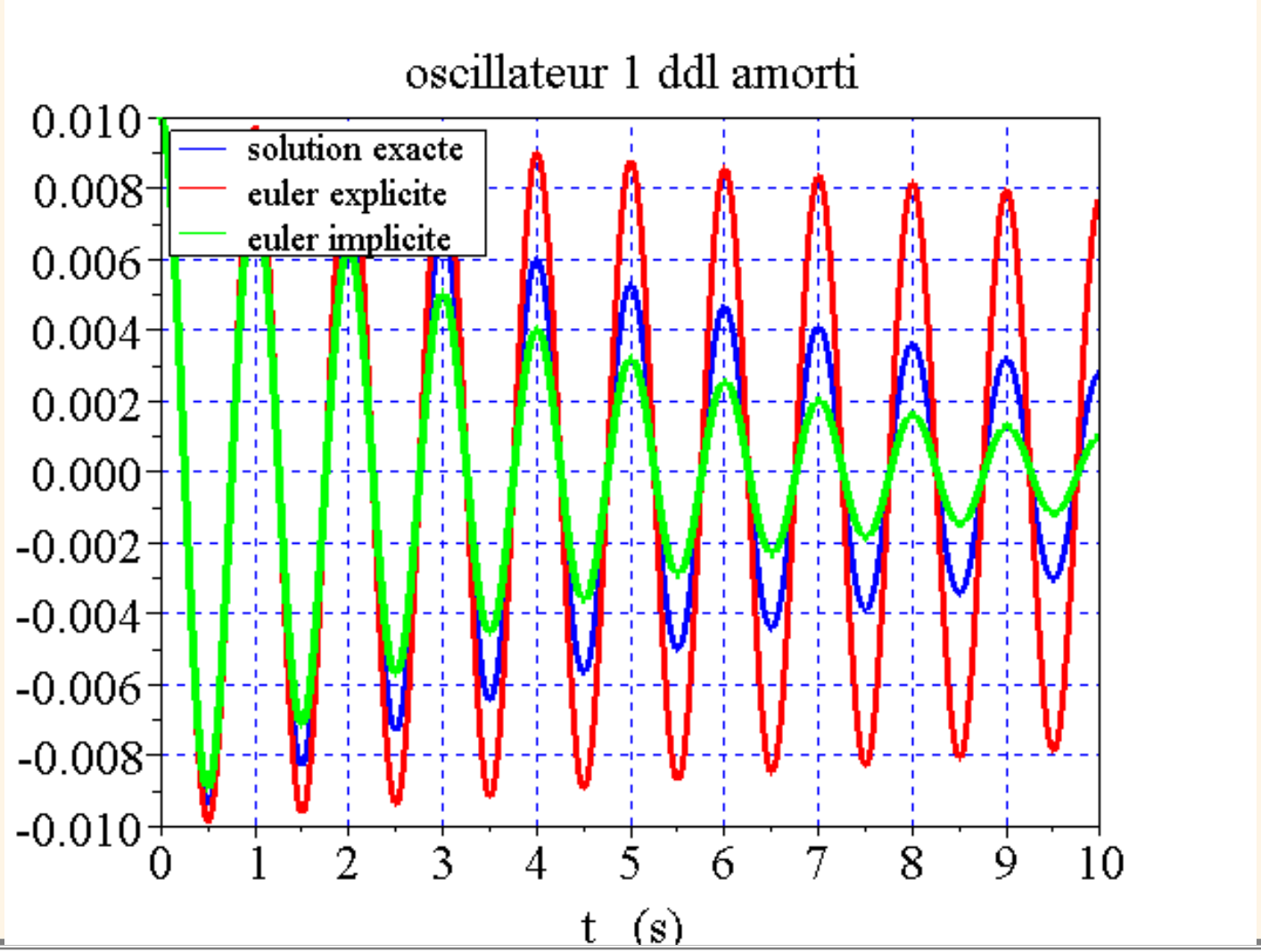
- ❑ solution exacte $\theta(t_j) = \theta_0 \cos(\omega_0 t_j + \varphi)$
- ❑ précision
  - en période
  - en amplitude



$$\Delta t = 4 \text{ ms}$$

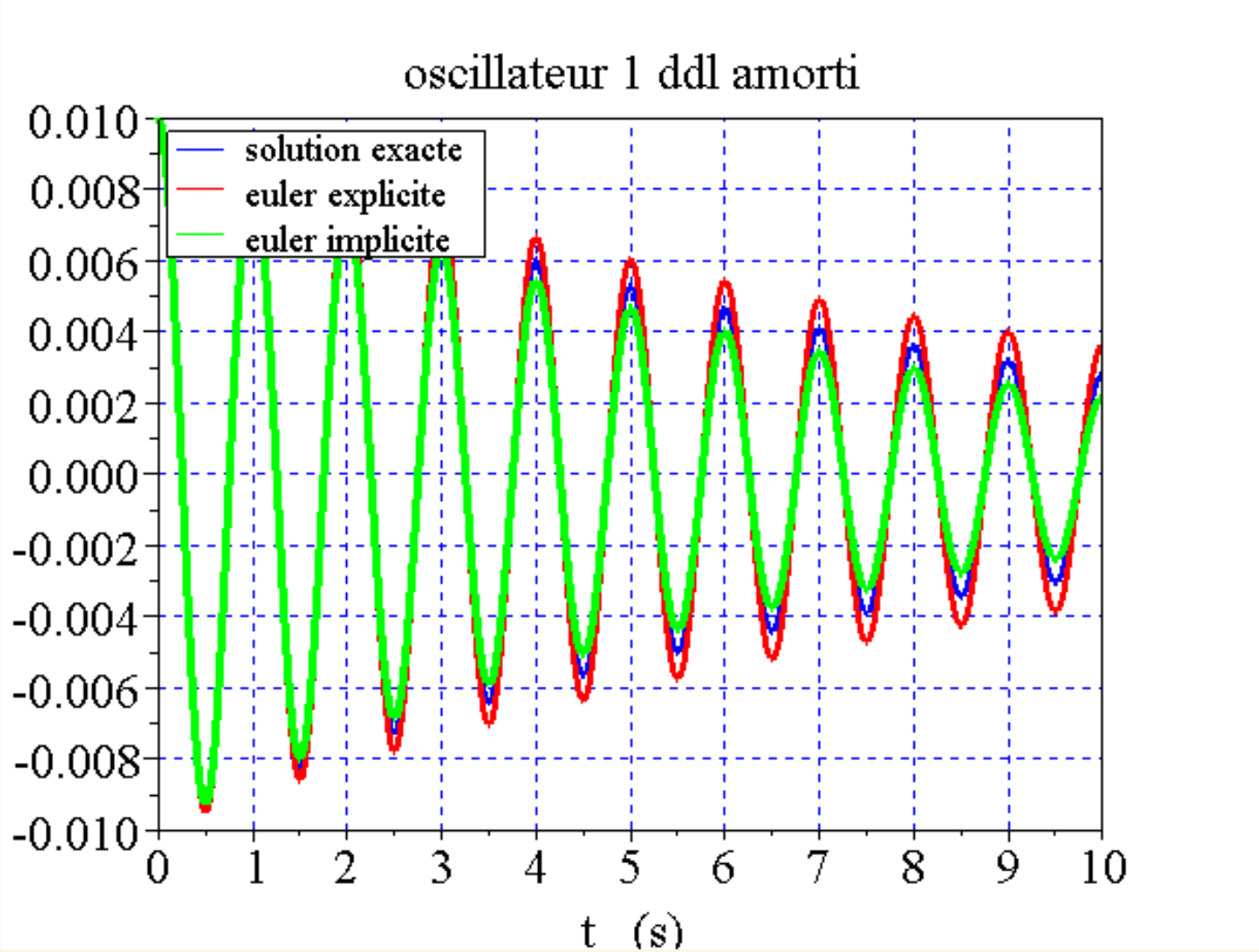
✓ oscillateur amorti

$$\Delta t = 0.8 \times \frac{2\varepsilon}{\omega_0}$$



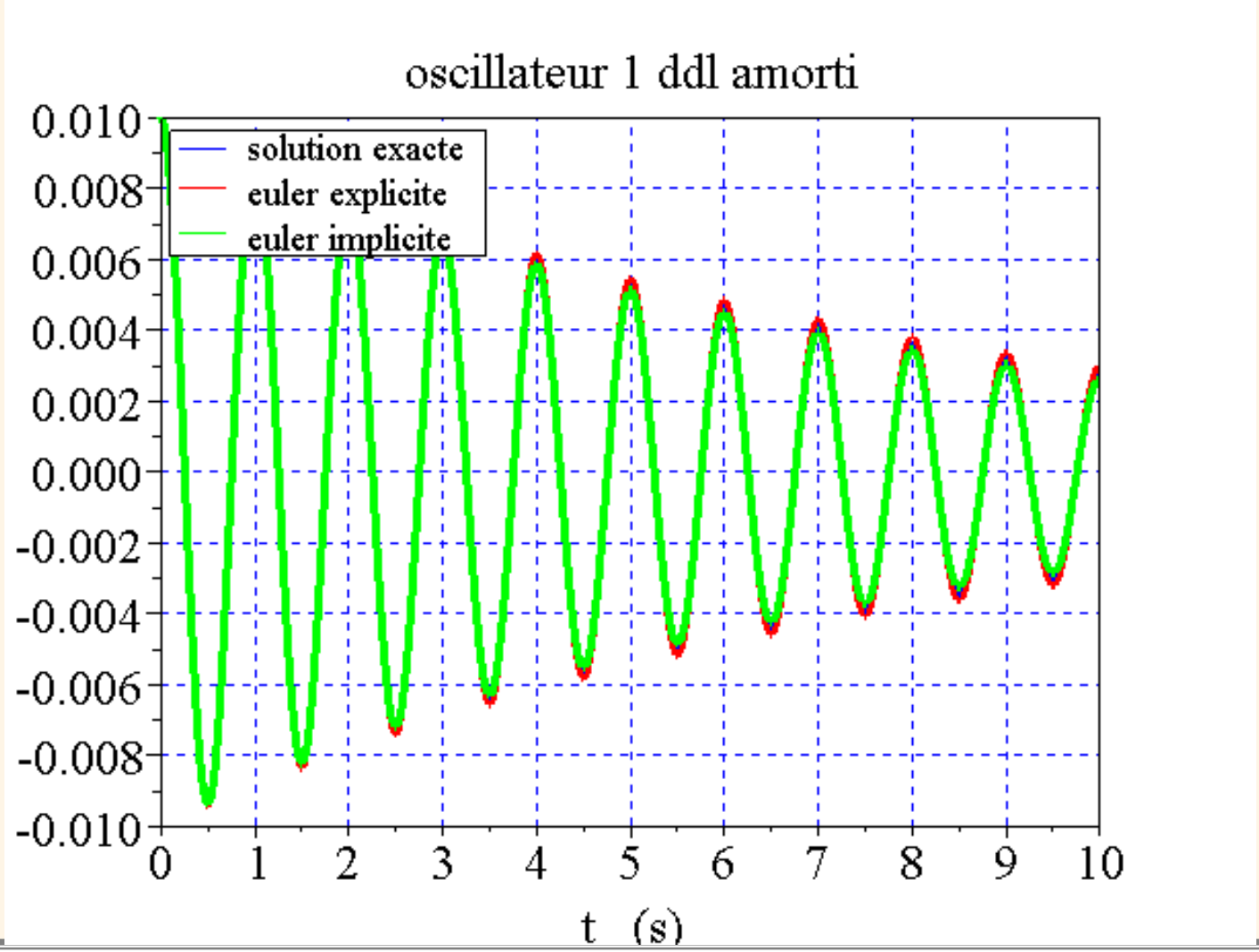
✓ oscillateur amorti

$$\Delta t = 0.2 \times \frac{2\varepsilon}{\omega_0}$$



✓ oscillateur amorti

$$\Delta t = 0.05 \times \frac{2\varepsilon}{\omega_0}$$



✓ remarque sur la stabilité – propagation des erreurs numériques (1/1)

$$[A]=\begin{bmatrix} 1 & 1 \\ -i\omega_0 & i\omega_0 \end{bmatrix}\begin{bmatrix} 1-i\omega_0\,\Delta t & \\ & 1+i\omega_0\,\Delta t \end{bmatrix}\begin{bmatrix} i\omega_0 & -1 \\ i\omega_0 & 1 \end{bmatrix}\frac{1}{2i\,\omega_0}$$

EULER explicite

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

soit $$\begin{vmatrix} q_{j+p} \\ \dot{q}_{j+p} \end{vmatrix} = \begin{bmatrix} 1 & 1 \\ -i\omega_0 & i\omega_0 \end{bmatrix}\begin{bmatrix} (1-i\omega_0\,\Delta t)^p & \\ & (1+i\omega_0\,\Delta t)^p \end{bmatrix}\begin{bmatrix} i\omega_0 & -1 \\ i\omega_0 & 1 \end{bmatrix}\frac{1}{2i\omega_0}\times\begin{vmatrix} q_j \\ \dot{q}_j \end{vmatrix}$$

✓ remarque sur la stabilité – propagation des erreurs numériques (1/2)

$$[A] = [Z_1 \quad \cdots \quad Z_n] \times \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \times [Z_1 \quad \cdots \quad Z_n]^{-1}$$

$$[A] = [Z] \times [D] \times [Z]^{-1}$$

$$U_{j+p} = [A]^p U_j = \left([Z] \times [D]^p \times [Z]^{-1} \right) U_j$$

✓ remarque sur la stabilité – propagation des erreurs numériques (2/2)

$$\left| \begin{matrix} q_j \\ \dot{q}_j \end{matrix} \right|^* = \left| \begin{matrix} q_j + \Delta q_j \\ \dot{q}_j + \Delta \dot{q}_j \end{matrix} \right| \quad \Rightarrow \quad \left| \begin{matrix} q_{j+p} \\ \dot{q}_{j+p} \end{matrix} \right|^* = \left| \begin{matrix} q_{j+p} + \Delta q_{j+p} \\ \dot{q}_{j+p} + \Delta \dot{q}_{j+p} \end{matrix} \right|$$

$$\left| \begin{matrix} q_{j+p} \\ \dot{q}_{j+p} \end{matrix} \right|^* = \begin{bmatrix} 1 & \Delta t \\ -\omega_0^{\,2} \Delta t & 1 \end{bmatrix}^p \left| \begin{matrix} q_j \\ \dot{q}_j \end{matrix} \right|^*$$

$$\left| \begin{matrix} \Delta q_{j+p} \\ \Delta \dot{q}_{j+p} \end{matrix} \right| = \begin{bmatrix} 1 & 1 \\ -i\omega_0 & i\omega_0 \end{bmatrix} \begin{bmatrix} (1 - i\omega_0 \Delta t)^p & \\ & (1 + i\omega_0 \Delta t)^p \end{bmatrix} \begin{bmatrix} i\omega_0 & -1 \\ i\omega_0 & 1 \end{bmatrix} \frac{1}{2i\omega_0} \times \left| \begin{matrix} \Delta q_j \\ \Delta \dot{q}_j \end{matrix} \right|$$

$$\left\| \left| \begin{matrix} \Delta q_{j+p} \\ \Delta \dot{q}_{j+p} \end{matrix} \right| \right\|_\infty \rightarrow +\infty$$

Quand les valeurs propres de la matrice d'amplification sont de module supérieur à 1, les erreurs numériques se propagent.

✓ remarque sur la stabilité – propagation des erreurs numériques (2/2)

$$U_j^* = U_j + \Delta U_j \quad \Rightarrow \quad U_{j+p}^* = U_{j+p} + \Delta U_{j+p}$$

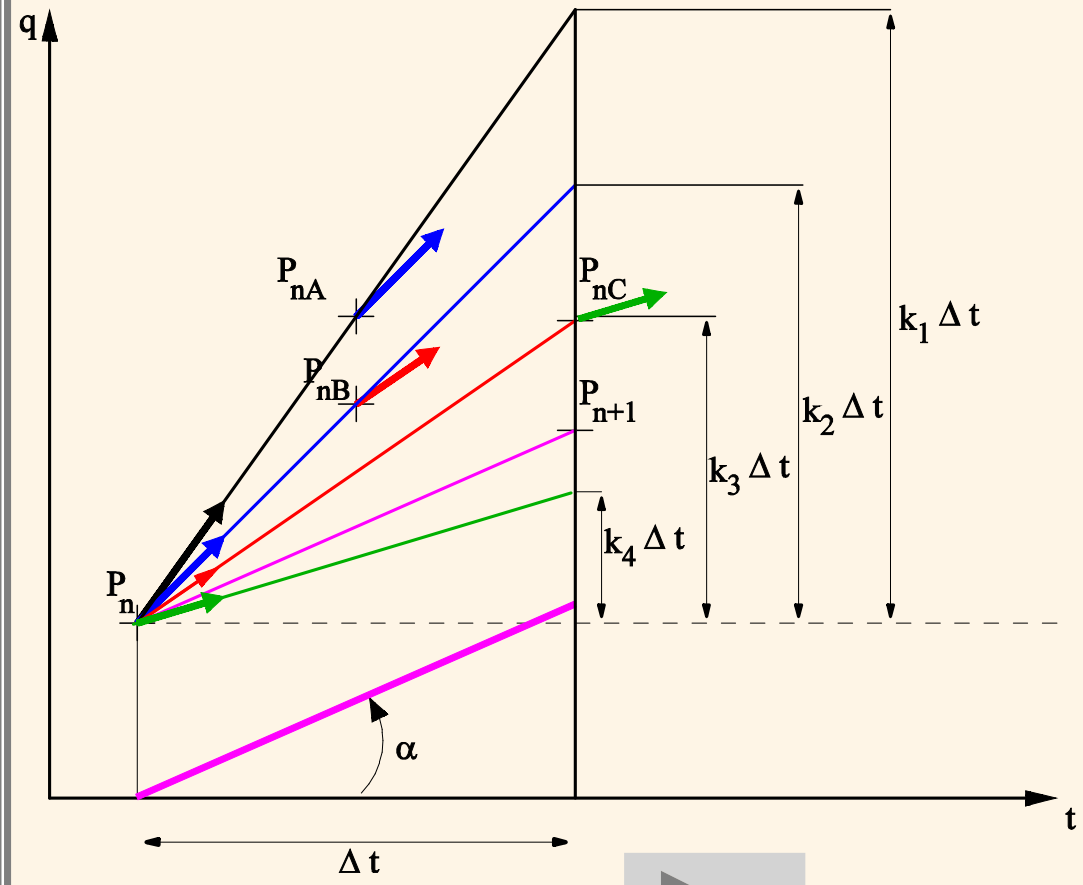
$$\Delta U_{j+p} = \left([Z][D]^p [Z]^{-1} \right) \Delta U_j$$

$$\| \Delta U_{j+p} \|_\infty \to +\infty$$

Quand les valeurs propres de la matrice d'amplification sont de module supérieur à 1, les erreurs numériques se propagent.

➢Schéma d'intégration de RUNGE KUTTA

$$\left\{\dot{U}(t)\right\} = \left\{F\left(U(t), t\right)\right\}$$



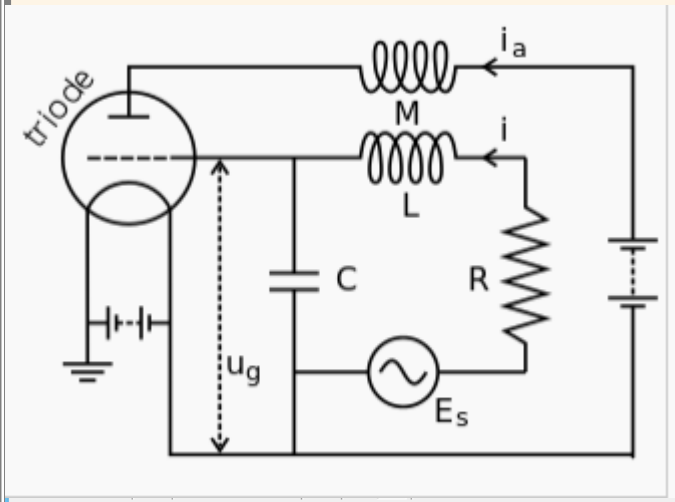
$$U_{n+1} = U_n + K \times \Delta t$$

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

$$\begin{cases} k_1 = F\left(U_n, t_n\right) \\ k_2 = F\left(U_n + k_1 \dfrac{\Delta t}{2}, t_n + \dfrac{\Delta t}{2}\right) \\ k_3 = F\left(U_n + k_2 \dfrac{\Delta t}{2}, t_n + \dfrac{\Delta t}{2}\right) \\ k_4 = F\left(U_n + k_3 \Delta t, t_n + \Delta t\right) \end{cases}$$

Balthasar Van Der Pol

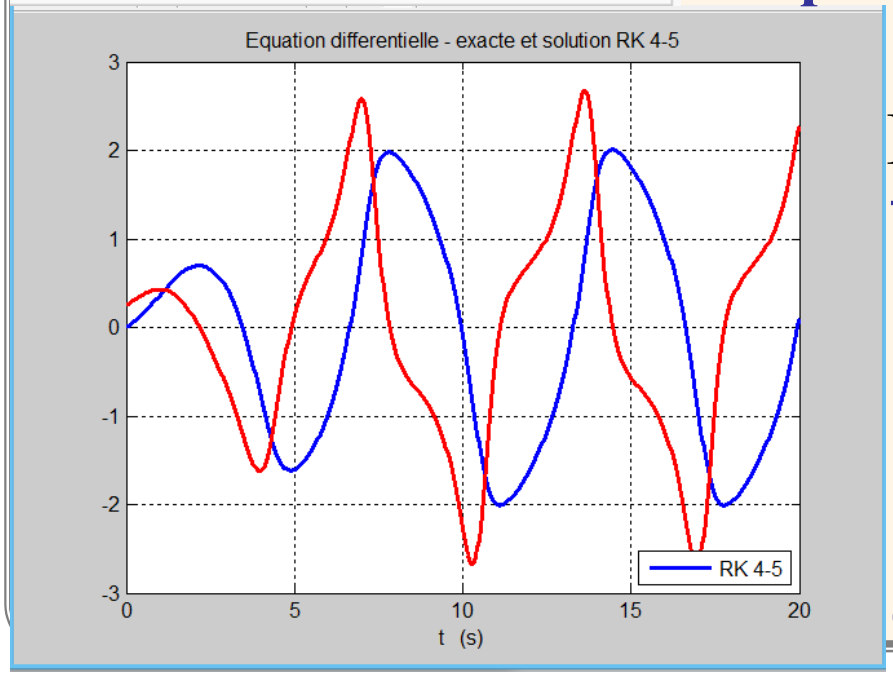
1927, Philips



$$\frac{\partial^2 x}{\partial t^2} + (x^2 - 1)\frac{\partial x}{\partial t} + x = 0$$

$$t_0 = 0 \Rightarrow x(t_0) = 0 \qquad \dot{x}(t_0) = 0.25$$

Ecrire le système du premier ordre pour algorithme de Runge Kutta



Fichier cal_f.m

```
function [ dUc ] =  cal_f(Uc,tc,w0c)
    dUc = zeros(2,1)   ;

    dUc(1)= Uc(2);
    dUc(2) = (1-Uc(1)^2)* Uc(2)-Uc(1);
end
```

➢RUNGE KUTTA

Système différentiel 1ème ordre

2 inconnues

```
dt6 = 0.04
t6 = (0:dt6:20)'
np6 = size(t6,1)

q0 = 0. ;
dq0 =0.25 ;

q6     = zeros(np6,1)
dq6    = zeros(np6,1)

q6(1)  =  q0
dq6(1) = dq0
qj     = [q0 ; dq0]
```

```
for inc = 2 : np6
      tc  = t6(inc-1)
      xc  = qj
      k1 = cal_fc(xc, tc,0)
      xc  = qj + k1 * dt6 / 2
      k2 = cal_fc(xc, tc + dt6 / 2 ,0)
      xc  = qj + k2 * dt6 / 2
      k3 = cal_fc(xc, tc + dt6 / 2 ,0)
      xc  = qj + k3 * dt6
      k4 = cal_fc(xc, tc + dt6 ,0)
      dq  = (k1 + 2 *k2 + 2 * k3 + k4) / 6
      qj   = qj + dq * dt6
      q6(inc)  = qj(1)
      dq6(inc) = qj(2)
end
```

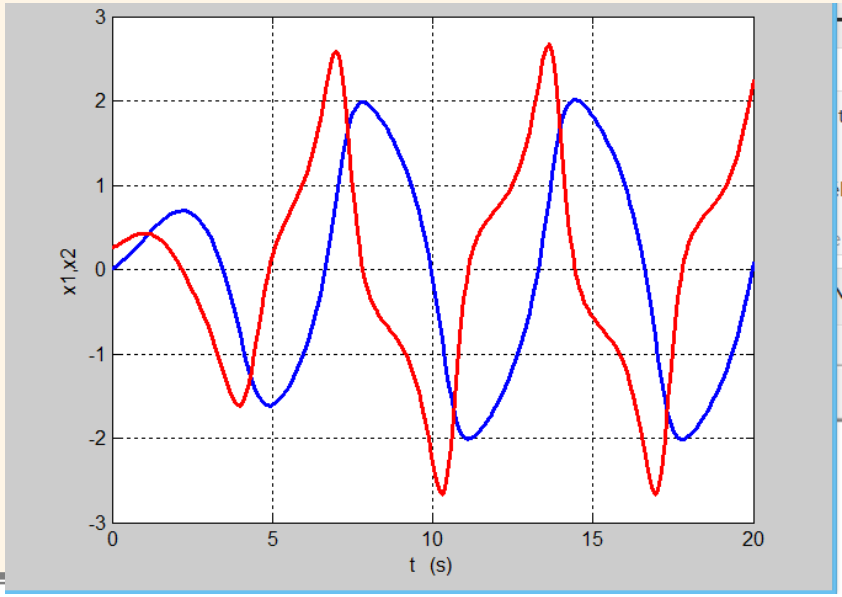
Equation Van Der Pol

```
t0 = 0;
tfinal = 20;
x0 =[0  0.25]';
```

Solutions avec : $x(t_0) = 0 \qquad \dot{x}(t_0) = 0.25$

MatLab : ode45, ode23, ode113, ode15s

```
[t,x] = ode45('vdp1', [t0 tfinal],x0);
plot(t,x(:,1),'-b',t,x(:,2),'-r', 'Linewidth',2)
```



Fichier vdp1.m

```
function dx = vdp1(t,x)
dx(1)= x(2);
dx(2) = (1-x(1)^2)* x(2)-x(1);
dx = dx'
% dx, vecteur colonne
```

Travaux dirigés : résolution numérique des équations du mouvement

Résumé du cours séance 3 :

Les méthodes qui vont transformer l'équation différentielle du second ordre en un système équations différentielles du premier ordre

Oscillateur linéaire à un degré de liberté

Schéma Euler implicite

Résultats de l'oscillateur linéaire à un degré de liberté amorti

Le schéma de Runge Kutta adapté à un système différentiel du 1er ordre

Travaux dirigés à faire :

Ouvrir le fichier : MecaNum_practice2019def.pdf

Exercice 1 : Exercice étude d'un oscillateur linéaire à un degré de liberté

Faire les questions avec Euler implicite et Runge Kutta

Exercice 2 : Traiter l'exercice étude d'un oscillateur linéaire amorti à un degré de liberté

L'objectif est de programmer avec Matlab ou Scilab sur votre PC :

La solution analytique , la solution par Euler Explicite

La solution avec Euler implicite et la solution avec Runge Kutta