Quick dusting

“I hear and I forget. I see and I remember. I do and I understand.”

–Confucius

ODE 1ST ORDER FORMULATION

- Every nth order ODE can be expressed as a system of 1st order ODEs

$$\frac{d^2y}{dt^2} + 2\gamma\omega\frac{dy}{dt} + \omega^2 y = 0$$

$$\Leftrightarrow \begin{cases} \dfrac{dy}{dt} = z \\ \dfrac{dz}{dt} = -2\gamma\omega\, z - \omega^2 y \end{cases}$$

$$\Leftrightarrow \frac{d}{dt}\begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -2\gamma\omega \end{bmatrix} \cdot \begin{bmatrix} y \\ z \end{bmatrix}$$

A second order ODE

⇔ first order system of the form

$$\frac{d}{dt}\vec{X} = \vec{f}(\vec{X}, t)$$

⇔ Matrix formulation (**only** Linear ODE)

How to MATLAB this

```
% Parameters
w = 1;
g = 0.1;

% X = [y;z], w = omega, g = gamma
f = @(t,X) [X(2);-w^2*X(1)-2*w*g*X(2)]

% Initial_conditions
X0 = [1;0];
time_int = [0,100];

% Solving ODE:
% ode45(function handle, time interval, initial condition)
[t,X] = ode45(f,time_int,X0);

plot(t,X);
xlabel('t'), ylabel('y,z'), legend('displacement', 'velocity')
```

$$\frac{d^2y}{dt^2} + 2\gamma\omega\frac{dy}{dt} + \omega^2 y = 0$$

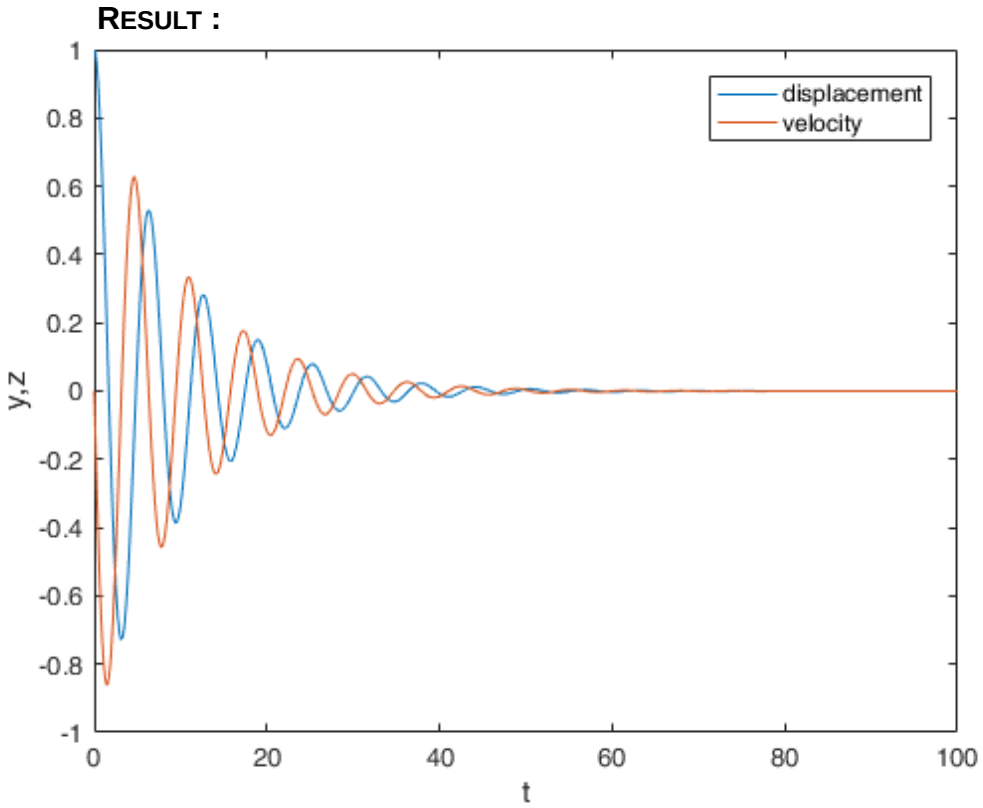
$$\Leftrightarrow \begin{cases} \dfrac{dy}{dt} = z \\ \dfrac{dz}{dt} = -2\gamma\omega\, z - \omega^2 y \end{cases}$$

$$\Leftrightarrow \frac{d}{dt}\begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -2\gamma\omega \end{bmatrix} \cdot \begin{bmatrix} y \\ z \end{bmatrix}$$

How to MATLAB this

```
% Parameters
w = 1;
g = 0.1;

% X = [y;z], w = omega, g = gamma
f = @(t,X) [X(2);-w^2*X(1)-2*w*g*X(2)]

% Initial_conditions
X0 = [1;0];
time_int = [0,100];

% Solving ODE:
% ode45(function handle, time interval, initial condition)
[t,X] = ode45(f,time_int,X0);

plot(t,X);
xlabel('t'), ylabel('y,z'), legend('displacement', 'velocity')
```

Result :



How to make functions in MATLAB

There are two types of functions

Anonymous functions

```
% Your code
X = [10 ; 2];
t = 5

% Your function
f = @(t,X) [X(2);-X(1)*X(2)]

% Call your function
Y = f(t,X);
disp(Y)
>>>          2
           -20
```

Named functions

```
%% in file my_f.m
function y = my_f(t,X)
y = [X(2);-X(1)*X(2)]
end
```

```
%% in file my_script.m
% Your code
X = [10 ; 2];
t = 5
% Call your function
Y = my_f(t,X);
disp(Y)
>>>          2
           -20
```

You need to write two separate files for this syntax:

- `my_f.m` (function definition)
- `my_script.m` (function call)

HANDS ON!

```
% Parameters
w = 1;
g = 0.1;

% X = [y;z], w = omega, g = gamma
f = @(t,X) [X(2);-w^2*X(1)-2*w*g*X(2)]

% Initial_conditions
X0 = [1;0];
time_int = [0,100];

% Solving ODE:
% ode45(function handle, time interval, initial condition)
[t,X] = ode45(f,time_int,X0);

plot(t,X);
xlabel('t'), ylabel('y,z'), legend('displacement', 'velocity')
```