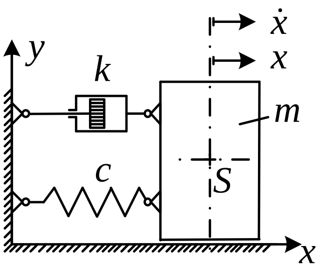
Széchenyi István University
GIVK

Department of
Applied Mechanics

Dynamics of Machines Week 9 – –1st and 2nd Exercises

Free Vibration of a Single Degree of Freedom System



Equation of motion:

$$m\ddot{x} = F = F_{damp}(\dot{x}) + F_{spring}(x)$$

$$m\ddot{x} = F = F_{damp}(\dot{x}) + F_{spring\_1}(x) + F_{spring\_2}(x^2)$$

$$m\ddot{x} = (-k\dot{x}) + \left(-\frac{1}{c}x\right) + \left(-\frac{1}{d}x^2\right)$$

$$\ddot{x} = \frac{1}{m}\left(-k\dot{x} - \frac{1}{c}x - \frac{1}{d}x^2\right) \quad (\text{rearrangement})$$

Data: $m = 2\ kg$, $c = 5\cdot 10^{-2}\ m/N$,

$d = -5,02\cdot 10^{-4}\ m^2/N$, $k = 2\ Ns/m$

Initial Condtitions: $x_0 = 0,01\ m$, $\dot{x}_0 = 0\ m/s$

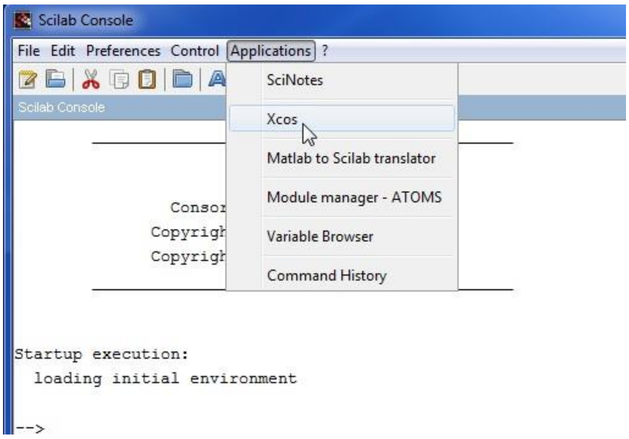
Spring force: $F_{rugó}(x) = \left(-\frac{1}{c}x - \frac{1}{d}x^2\right)$, Damping force: $F_{csill}(\dot{x}) = (-k\dot{x})$

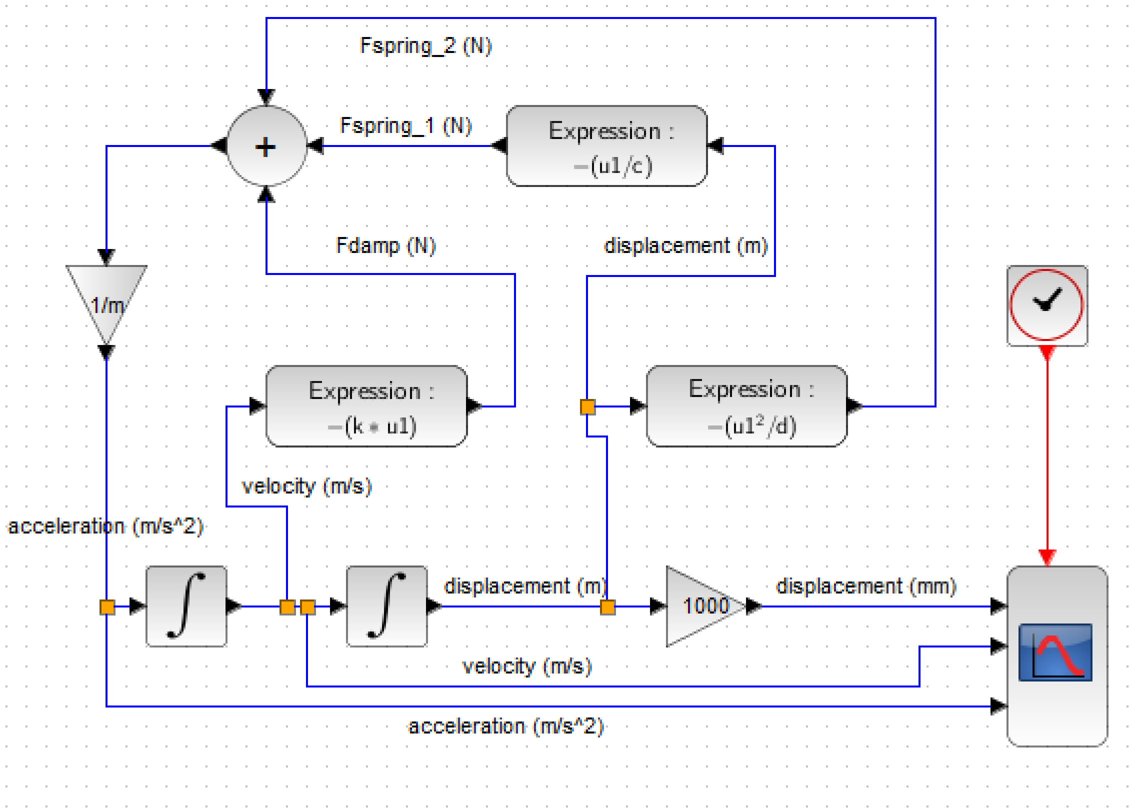
Calculate and plot the displacement, velocity and acceleration functions of the mass in the first 10s. There are two initial conditions in $t = t_0 = 0\ s$ the initial displacement is $x_0 = 0,01\ m$ and the initial velocity is $\dot{x}_0 = 0\ m/s$.

Friction is totally neglected.

Length of one time step: $\Delta t = 0,001\ (s)$

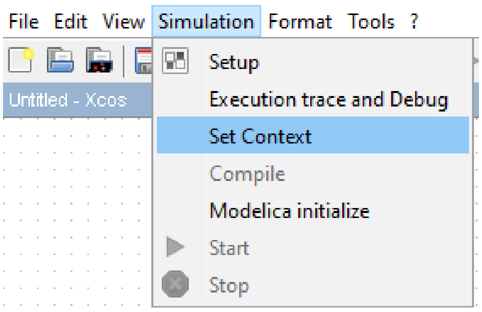
9/1 Exercise – Free Vibration of a Single Degree of Freedom System (xcos)

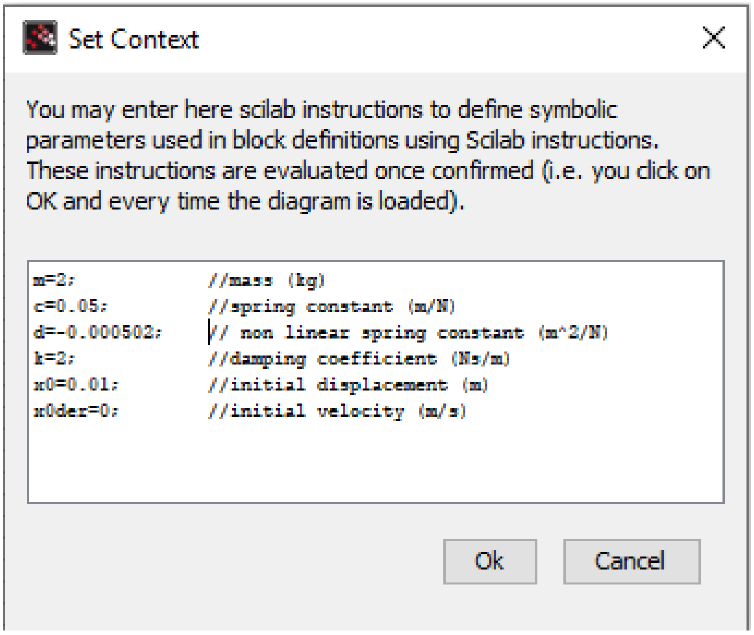


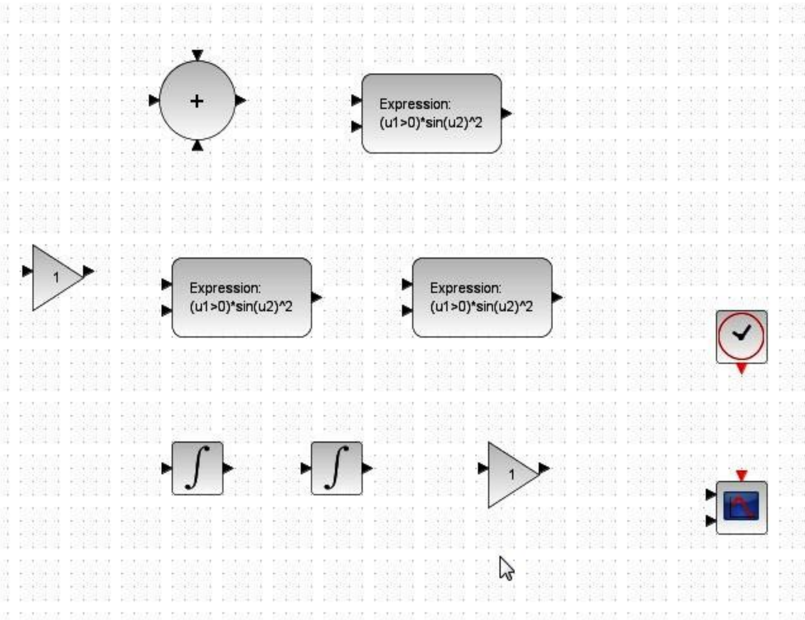


Set Context:

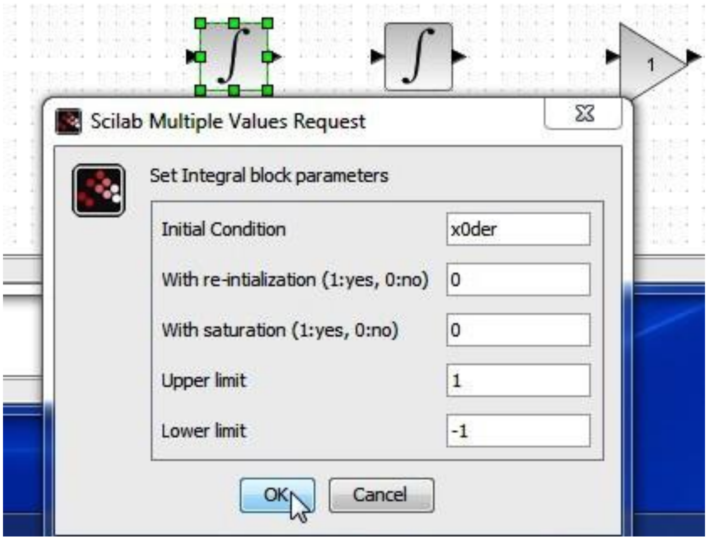
```
m=2;        //mass (kg)
c=0.05;     //spring constant (m/N)
d=-0.000502; //non linear spring constant (m^2/N)
k=2;        //damping coefficient (Ns/m)
x0=0.01;    //initial displacement (m)
x0der=0;    //initial velocity (m/s)
```



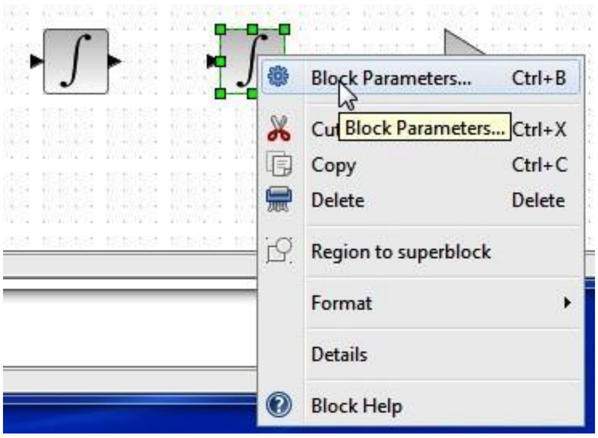


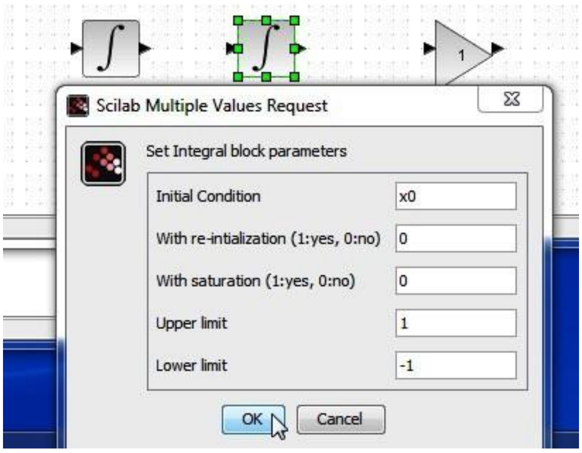


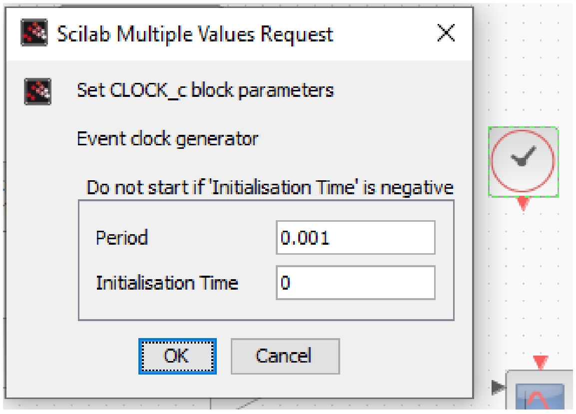
Initial velocity: x0der (m/s)



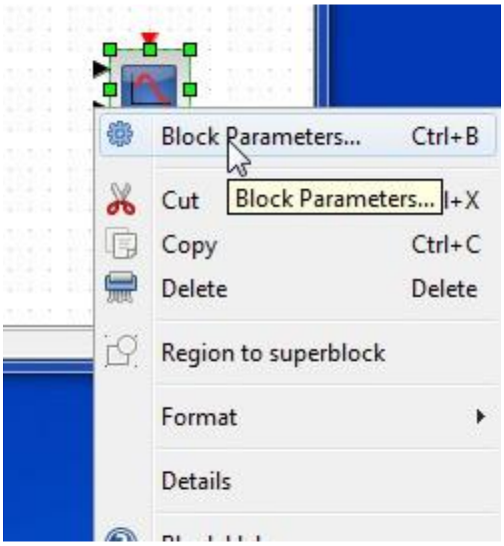
Initial displacement: x0 (m)



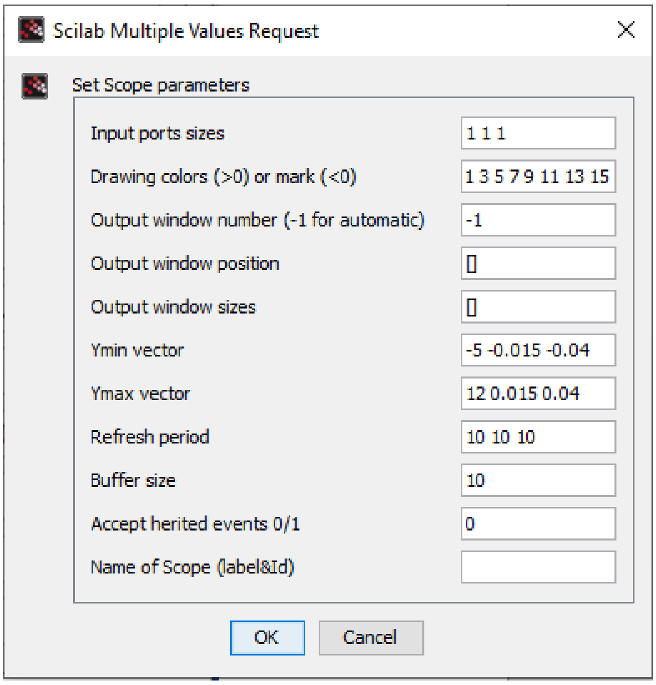




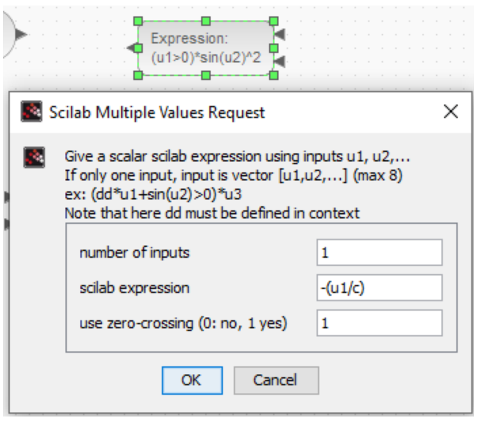
Scilab Multiple Values Request
Set CLOCK_c block parameters
Event clock generator
Do not start if 'Initialisation Time' is negative
Period
0.001
Initialisation Time
0
OK
Cancel



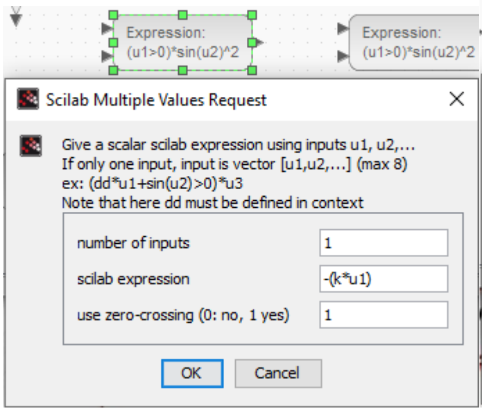
Block Parameters... Ctrl+B
Cut
Block Parameters...
Copy Ctrl+C
Delete Delete
Region to superblock
Format
Details



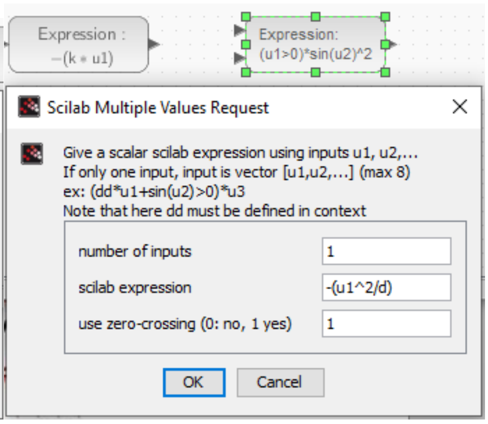
Scilab Multiple Values Request
Set Scope parameters
Input ports sizes
1 1 1
Drawing colors (>0) or mark (<0)
1 3 5 7 9 11 13 15
Output window number (-1 for automatic)
-1
Output window position
[]
Output window sizes
[]
Ymin vector
-5 -0.015 -0.04
Ymax vector
12 0.015 0.04
Refresh period
10 10 10
Buffer size
10
Accept herited events 0/1
0
Name of Scope (label&Id)
OK
Cancel



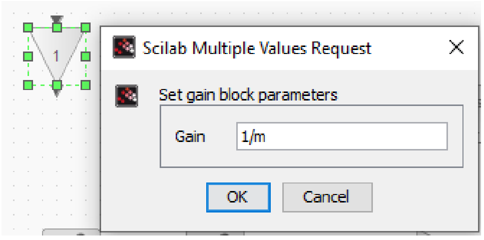
Expression:
(u1>0)*sin(u2)^2
Scilab Multiple Values Request
Give a scalar scilab expression using inputs u1, u2,...
If only one input, input is vector [u1,u2,...] (max 8)
ex: (dd*u1+sin(u2)>0)*u3
Note that here dd must be defined in context
number of inputs
1
scilab expression
-(u1/c)
use zero-crossing (0: no, 1 yes)
1
OK
Cancel

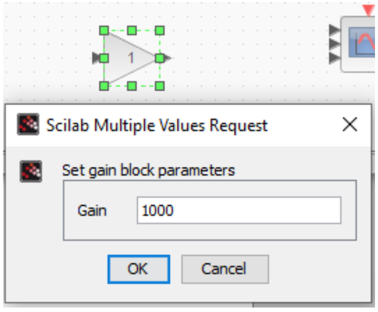


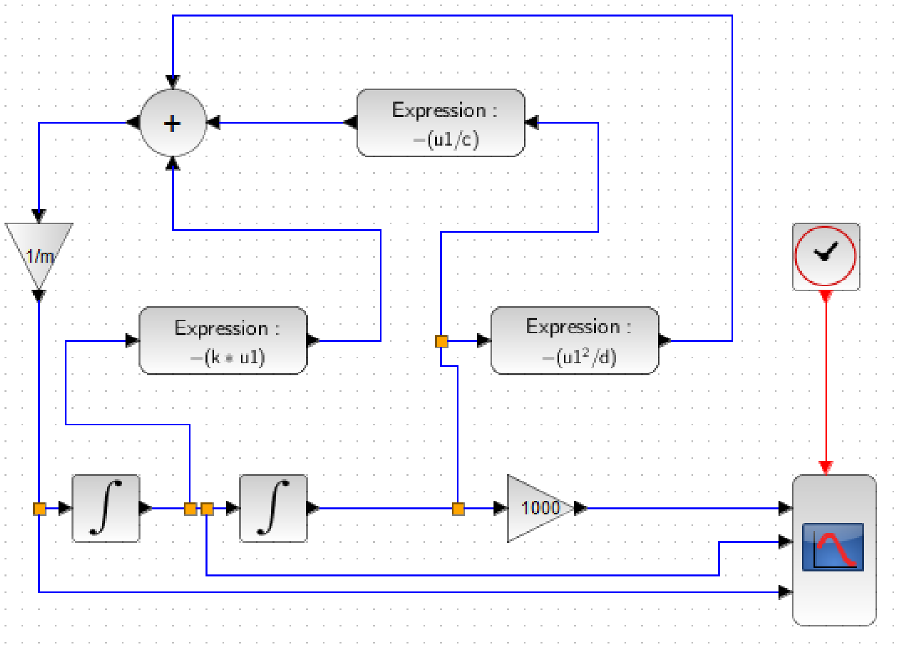
Expression:
(u1>0)*sin(u2)^2
Expression:
(u1>0)*sin(u2)^2
Scilab Multiple Values Request
Give a scalar scilab expression using inputs u1, u2,...
If only one input, input is vector [u1,u2,...] (max 8)
ex: (dd*u1+sin(u2)>0)*u3
Note that here dd must be defined in context
number of inputs
1
scilab expression
-(k*u1)
use zero-crossing (0: no, 1 yes)
1
OK
Cancel



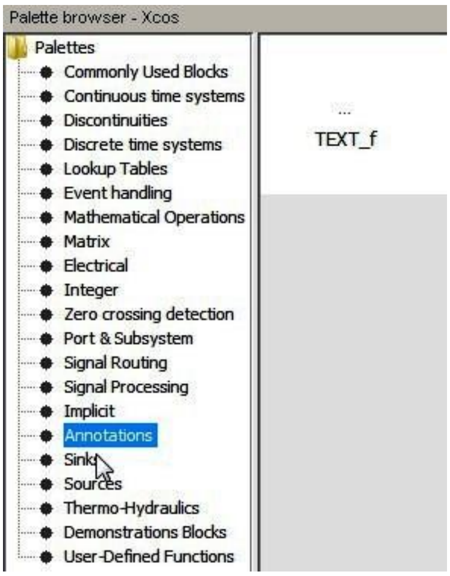
Expression :
−(k ∗ u1)
Expression:
(u1>0)*sin(u2)^2
Scilab Multiple Values Request
Give a scalar scilab expression using inputs u1, u2,...
If only one input, input is vector [u1,u2,...] (max 8)
ex: (dd*u1+sin(u2)>0)*u3
Note that here dd must be defined in context
number of inputs
1
scilab expression
-(u1^2/d)
use zero-crossing (0: no, 1 yes)
1
OK
Cancel

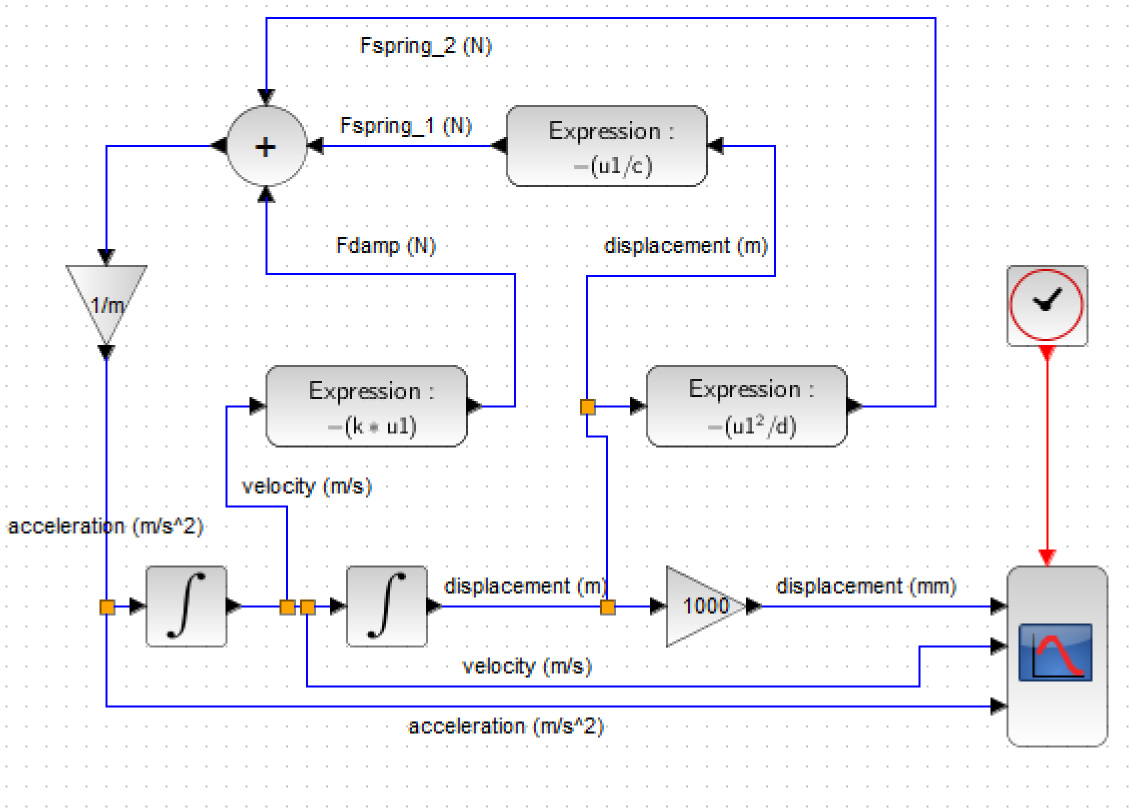




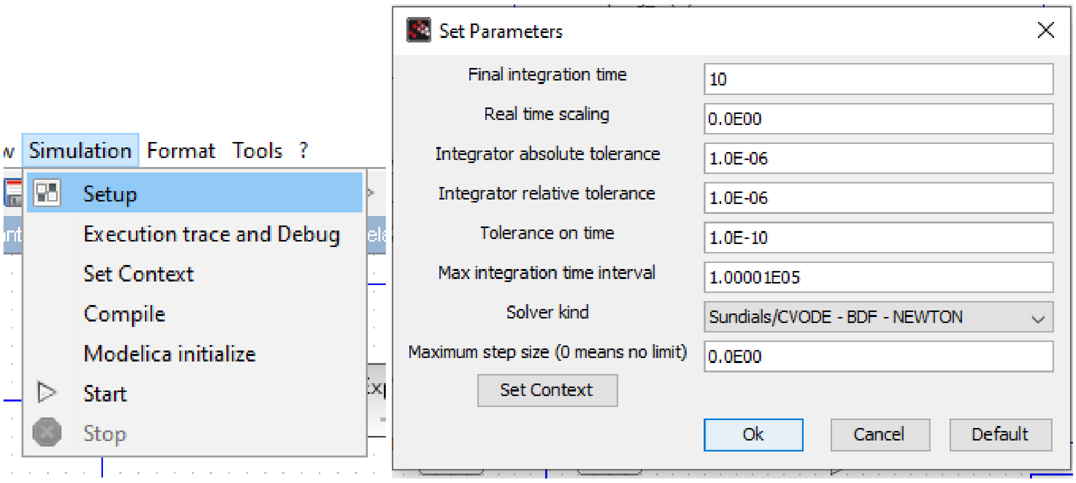


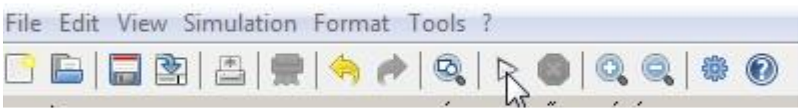
Make annotations:

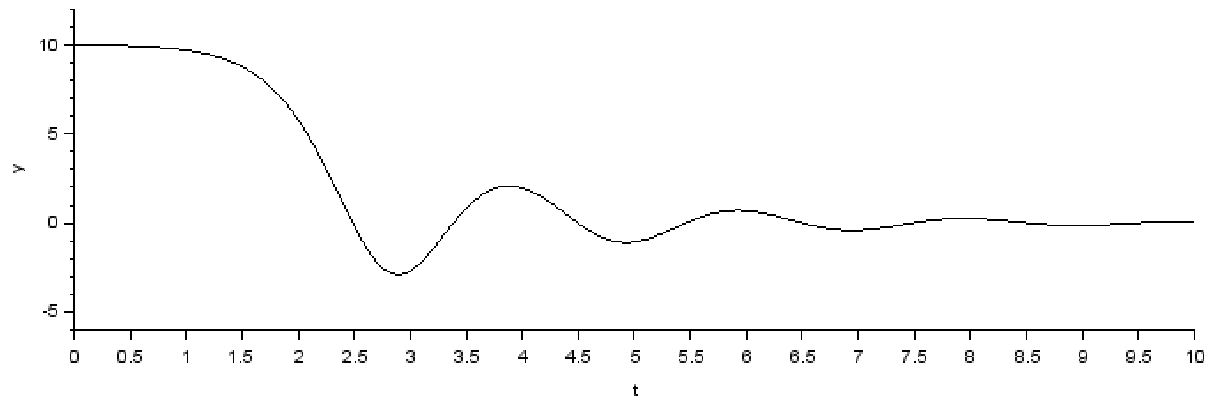


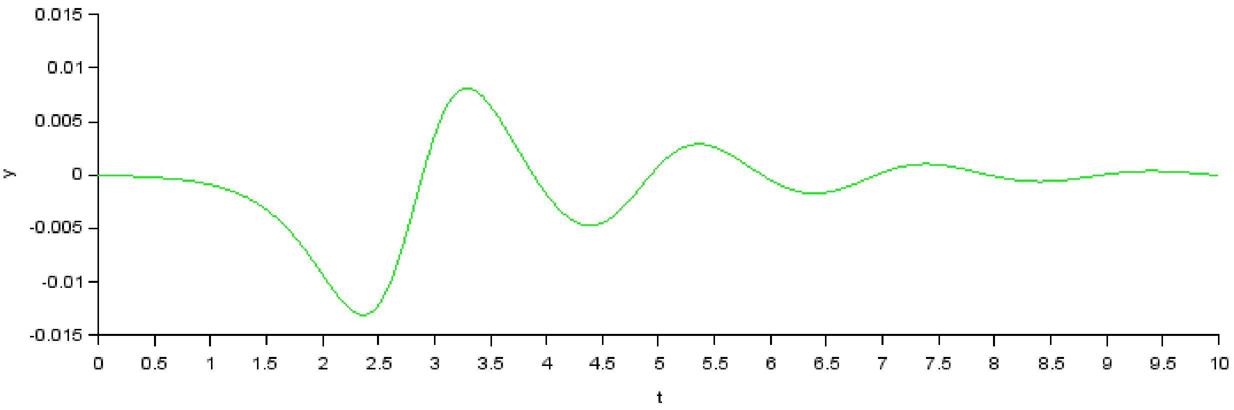


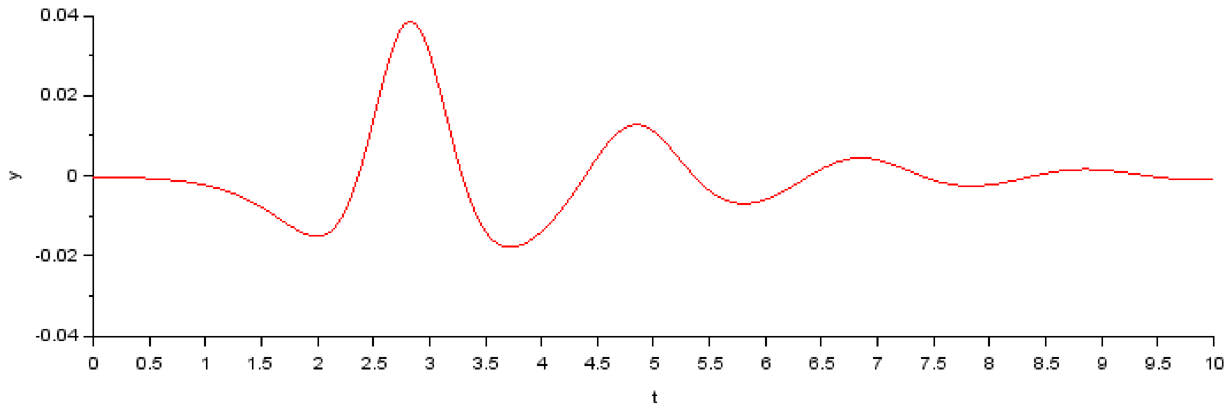
Final integration time: 10 (s)



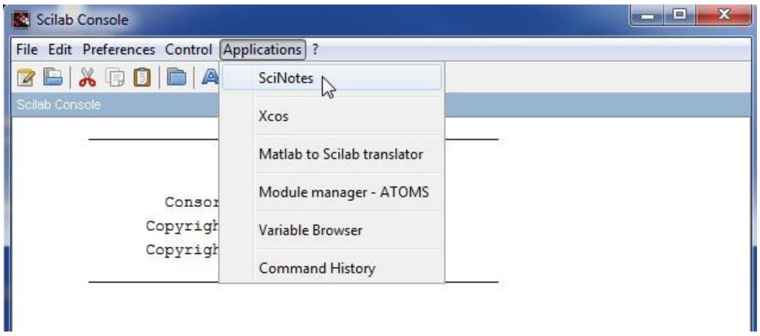



10
5
0
-5
y
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5 8 8.5 9 9.5 10
t


0.015
0.01
0.005
0
-0.005
-0.01
-0.015
y
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5 8 8.5 9 9.5 10
t


0.04
0.02
0
-0.02
-0.04
y
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5 5.5 6 6.5 7 7.5 8 8.5 9 9.5 10
t

9/2 Exercise – Free Vibration of a Single Degree of Freedom System (SciNotes)




```
// week 9 – 2nd exercise
          // Solve the following second order differential equation with Runge-Kutta method
          // Runge-Kutta módszerrel:  mx''=Fdamp(x')+Fspring(x).
          // Rearrangement:              x''=(1/m)*(Fdamp(x')+Fspring(x))
          // Damping force (N):         Fdamp(x')= -k*x'
          // Spring force (N):           Fspring(x)=-(x/c)-(x^2/d)
          //-----------------------------------------------------------------------------------------------
          // Initial Conditions:  t=0 (s)  >>>  x(0)=0.01 (m) initial displacement
          //                              t=0 (s)  >>>  x'(0)=0 (m/s) initial velocity
          //-----------------------------------------------------------------------------------------------
          // Let's define an x column matrix. It has two rows which contains the following data:
          //                          x(1)=x (m) displacement
          //                          x(2)=x' (m/s) velocity
          // Derivative of the x matrix: x' , where x'(1)=x' so the same as x(2) (m/s) velocity
          //                          x'(1)=x'' so x(1) (m/s) velocity,
          //                          x'(2)=x'' (m/s^2) acceleration,
          //                          x'(2)=x''=-kx'-(x/c)-(x^2/d)
          //                          x'(2)=x''=-k*x(2)-(x(1)/c)-((x(1)^2)/d)
clear;
// Variables -----------------------------------------------------------------------------------------
m=2;                 // kg
c=5*10^-2;           // spring constant (m/N)
d=-5.02*10^-4;       // (m^2/N)
k=2;                 // damping coefficient (Ns/m)
// Initial Conditions ////
x0(1)=0.01;          // initial displacement (m)
x0(2)=0;             // initial velocity (m/s)
t0=0;                // initial time (s)
// Time interval
t=0:0.001:10.0;      // from 0s to 10s with 0,001s time increment
// Calculation ---------------------------------------------------------------------------------------
function [xdot]=f(t, x)
          // Let's define the element of x' matrix
          xdot(1)=x(2);                                              // velocity
          xdot(2)=(1/m)*((-k*x(2,:))-(x(1,:)/c)-(x(1,:)^2/d));        // rearranged function
endfunction
//use ode command to solve the differential equation ------------------------------------------------------
x=ode("rk",x0,t0,t,f); // rk - means Runge-Kutta method, p0 – initial values matrix, t0 – initial time, t – time
                         interval, f – righ hand side of the differential equation
        ///////////////////////  Caution !!!  ///////////////////
```

```
        // With this method we can not get the angular acceleration function we just get the p matrix where
        p(1,:) contains the values of the angle function and p(2,:) contains the values of the angular velocity
        function
        // So we have to calculate the angular acceleration function with the values of these two functions
a(1,:)=(1/m)*((-k*x(2,:))-(x(1,:)/c)-(x(1,:)^2/d));  // acceleration (m/s^2)
// Plotting-----------------------------------------------------------------------------------------------------
subplot(3,1,1)
plot2d(t,x(1,:)*1000,1)
xtitle("Displacement function","time (s)","displacement (mm)")
xgrid(2);
         ////////////
subplot(3,1,2)
plot2d(t,x(2,:),5)
xtitle("Velocity function","time (s)","velocity (m/s)")
xgrid(2);
         ////////////
subplot(3,1,3)
plot2d(t,a(1,:),1);
xtitle("Acceleration function","time (s)","acceleration (m/s^2)");
xgrid(2);
```

>>>>>> Save the script.

Execute: Execute >>>> .... file with no echo