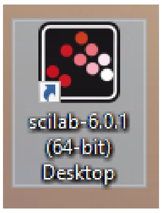
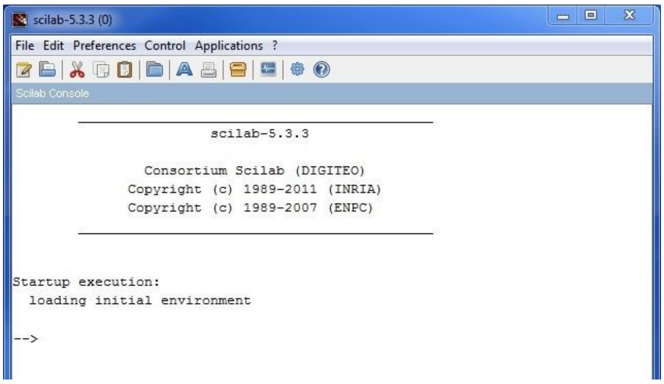
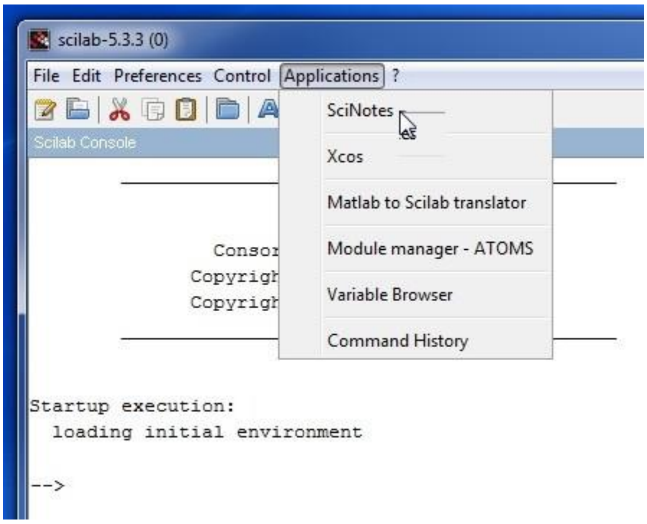
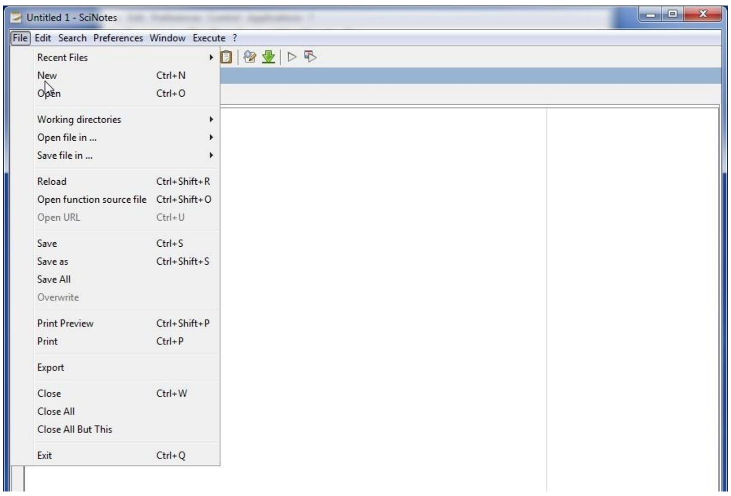
Dynamics of Machines Week 2 – 1st and 2nd Exercises









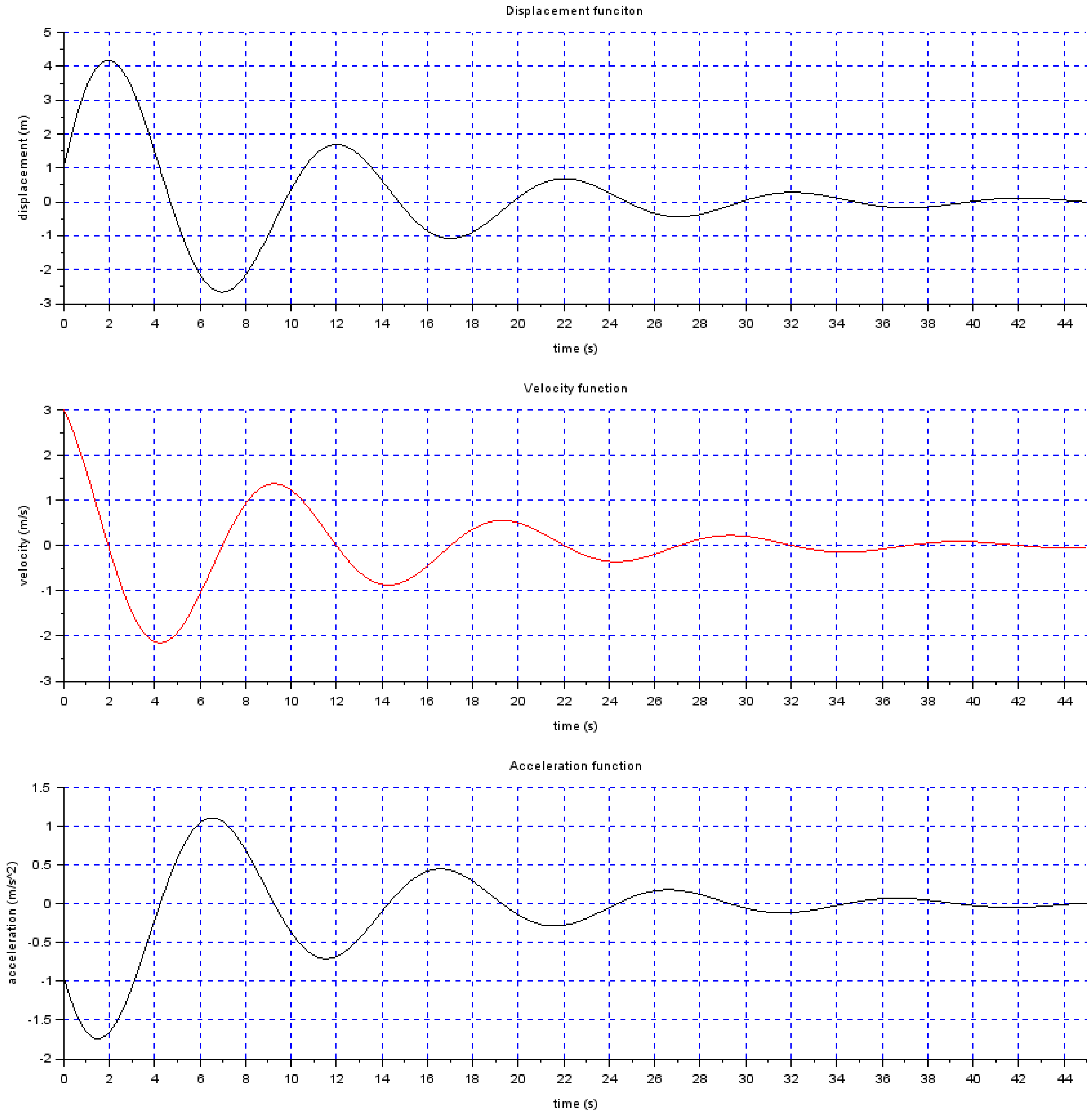
2/1 Exercise

```
// week 2 – 2/1 exercise
// Solve the following second order, homogeneous ordinary differential equation with Runge-Kutta method
        //                              5y"+0.9y'+2y=0
        // Rearrangement:               y"=-(0.9/5)y'-(2/5)y
        // Initial conditions:          t = 0 (s) ➔ y(0)=1 (m) displacement
        //                              t = 0 (s) ➔ y'(0)=3 (m/s) velocity
        //-------------------------------------------------------------------------------------------------
        // Let's define a p column matrix. It has two rows which contains the following data:
        //                      p(1)=y (m) displacement
        //                      p(2)=y' (m/s) velocity
        // Derivative of the p matrix: p' , where p'(1)=y' so the same as p(2) (m/s) velocity
        //              p'(1)=y'=p(2) velocity
        //              p'(2)=y" (m/s^2) acceleration
        //              p'(2)=y"=-(0.9/5)*y'-(2/5)*y
        //              p'(2)=-(0.9/5)*p(2)-(2/5)*p(1)
clear;

function [pdot]=f(t, p)
        // Let's define the elements of p' matrix
pdot(1)=p(2)                            // velocity
pdot(2)=-(0.9/5)*p(2)-(2/5)*p(1)        // rearranged function
endfunction
// Initial Conditions //////////
t0=0;                   // initial time
p0(1)=1;                // initial displacement , y(0)=1 (m)
p0(2)=3;                // initial velocity, y'(0)= 3 (m/s)
// Time interval
t=0:0.002:45;           // from 0s to 45s with 0,002s time increment
// ode command solves the differential equation ---------------------------------------------------------------
p=ode("rk",p0,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 acceleration function we just get the p matrix where p(1,:) contains
        the values of the displacement function and p(2,:) contains the values of the velocity function
        // So we have to calculate the acceleration function with the values of these two functions
a=-(0.9/5)*p(2,:)-(2/5)*p(1,:);         //  acceleration function (m/s^2)
// Plotting results ------------------------------------------------------------------------------------------
subplot(3,1,1)
        //Divide the graphic window into 3x1 matrix of sub-windows with subplot command
plot2d(t,p(1,:),1);
xtitle("Displacement funciton","time (s)","displacement (m)"); // Title of the graph, Label of x axis, Label of y
                                                                  axis
xgrid(2);               // plot grid in the background
subplot(3,1,2)
plot2d(t,p(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



2/2 Exercise

```
// week 2 – 2/2 exercise
        // Solve the following second order, homogeneous ordinary differential equation with Euler method
        //                                         5y"+0.9y'+2y=0 .
        // Rearrangement:                          y"=-(0.9/5)y'-(2/5)y
        // Initial conditions:                     t = 0 (s) ➔  y(0)=1 (m) displacement
        //                                         t = 0 (s) ➔  y'(0)=3 (m/s) velocity
clear;

//  Time interval
dt=0.002;                   // time increment (s)
tmax=45;                    // final time value of the calculation (s)
n=int(tmax/dt);             // number of time steps
//  Initial Conditions ////////////
t0=0;                       // initial time (s)
y0=1;                       // initial displacement y(0)=1 (m)
v0=3;                       // initial velocity, y'(0)= 3 (m/s)
a0=-(0.9/5)*v0-(2/5)*y0;    // initial acceleration, it has to be calculated from initial displacement and initial
                             velocity
t0=dt;
        /////////////////////
t=(1:n);
y=(1:n);
v=(1:n);
a=(1:n);

// 1st elements of the matricies   //////  i=1 loop variable ///////////
t(1)=0;
y(1)=y0;
v(1)=v0;
a(1)=a0;
/////////// for loop /// Numerical Integration ////////
for i=2:n
    t(i)=t0+dt;
    a(i)=-(0.9/5)*v0-(2/5)*y0;
    v(i)=v0+((a(i)+a0)/2)*dt;    // Trapezoidal Rule
    y(i)=y0+((v(i)+v0)/2)*dt;   // Trapezoidal Rule
        /// Variable value exchange
    t0=t(i);
    y0=y(i);
    v0=v(i);
    a0=a(i);
end
// Plotting results //////////////////////
subplot(3,1,1)
plot2d(t,y(1,:),1);
xtitle("Displacement function","t(s)","displacement (m)");
xgrid(2);
subplot(3,1,2)
plot2d(t,v(1,:),5);
xtitle("Velocity function","t(s)","velocity (m/s)");
```

```
xgrid(2);
subplot(3,1,3)
plot2d(t,a(1,:),1);
xtitle("Acceleration function","t(s)","acceleration (m/s^2)");
xgrid(2);
```

>>>>>> Save the script.

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