function [ y ] = myErf( x0,xfinal, N, y0 )

h = (xfinal-x0)/N;

xn = y0;

y = xn;

for i=1:N

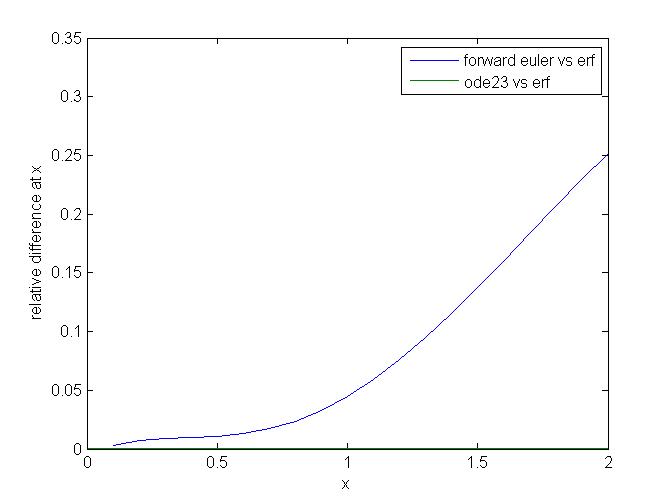
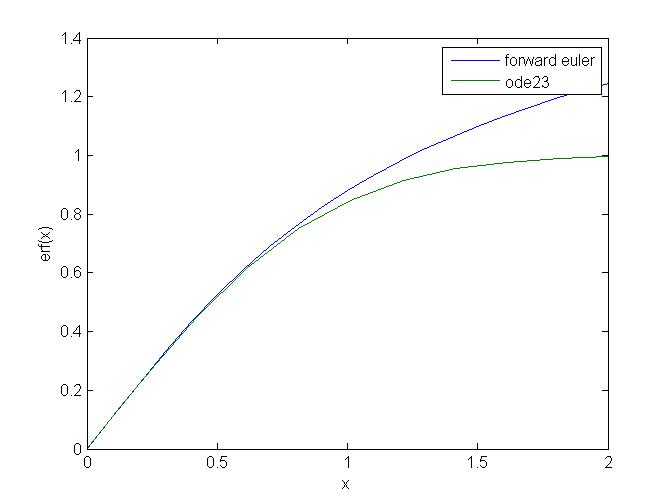
xn1 = xn + (2/sqrt(pi)\*exp(-xn^2))\*h;

y = [y xn1];

xn = xn1;

end

end



function myShoot(init\_pos,init\_theta,init\_vel)

%

% Initial position and velocity

%

x0=init\_pos(1);

y0=init\_pos(2);

theta0=init\_theta(1);

v0=init\_vel(1);

%

% Set ode options and call ode 45

%

options=odeset('Events',@events,'Refine',10,'MaxStep',0.01);

[t,z]=ode45(@motion\_ode,[0:0.05:5],[x0;y0;theta0;v0],options);

%

% After the calculation, plot/animate the trajectory of the triangle

%

figure(1);

hold on;

n=length(t);

tri\_h=plot(z(1,1),z(1,2),'r>','linewidth',10,'MarkerFaceColor','r');

set(tri\_h,'erasemode','normal');

for s=2:n,

set(tri\_h,'XData',z(s,1),'YData',z(s,2));

if s==2,

dot\_h=plot(z(1,1),z(1,2),'.','color',[1,0.7529,0.7961],'linewidth',4,'MarkerFaceColor','g');

else

dx=[get(dot\_h,'XData') z(s-1,1)];

dy=[get(dot\_h,'YData') z(s-1,2)];

set(dot\_h,'XData',dx,'YData',dy);

end

drawnow;

pause(0.05);

end

%

% Dynamics function for the triangle: Complete the missing parts

%

function dz = motion\_ode(t,z)

dz(1,1) = z(4)\*cos(z(3));

dz(2,1) = z(4)\*sin(z(3));

dz(3,1) = -9.81/z(4)\*cos(z(3));

dz(4,1) = -(((0.72\*1.29\*0.005)/2)\*((z(4)\*cos(z(3)))^2+(z(4)\*sin(z(3)))^2))/0.2-9.81\*sin(z(3));

end

%

% Event function: Complete the missing parts

%

function [values,halt,direction]=events(t,z);

values(1) = z(2);

halt(1) = 1;

direction(1) = -1;

values(2) = 2 - z(1);

halt(2) = 1;

direction(2) = -1;

values(3) = (abs(z(1) - 1.5) + abs(z(2) - 0.05)) - 0.05;

halt(3) = 1;

direction(3) = -1;

end

end

