Table of Contents

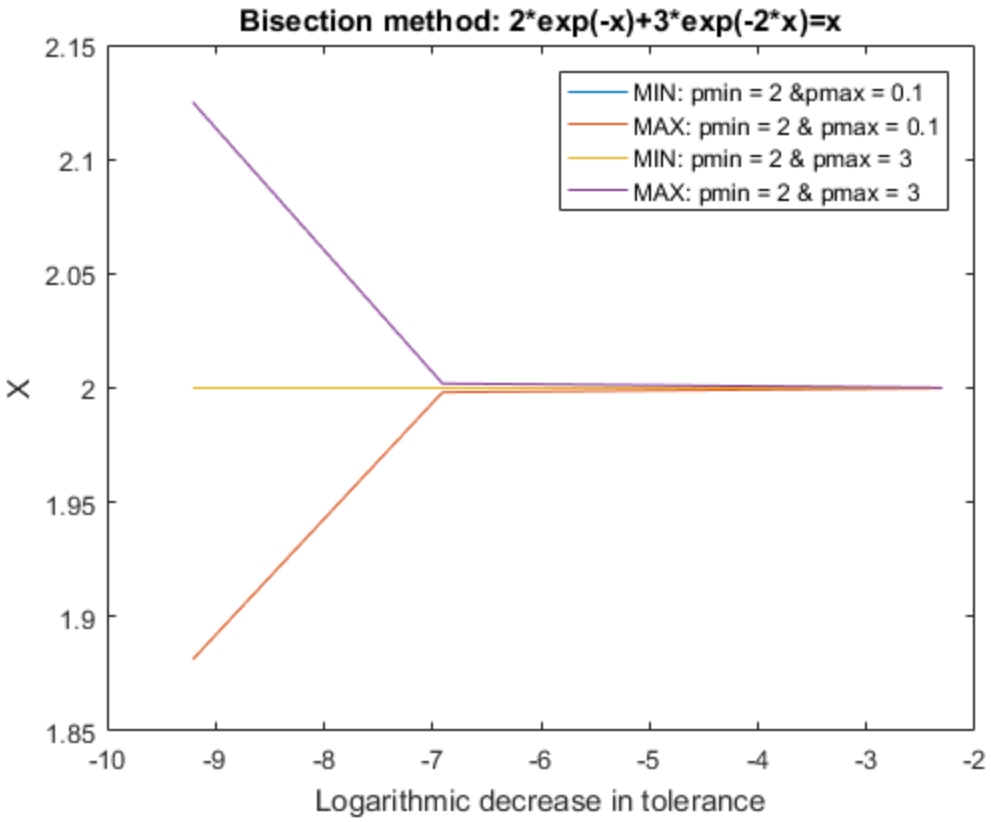
540.305 Problem Set 4: 1
1 Bisection method for solving non-linear problems 1
2 Using fzero to solve non-linear problems 2
3 Solving systems of non-linear problems using fsolve 3
4 Simulating balls moving in a box and colliding 4

540.305 Problem Set 4:

```
clear
clc
```

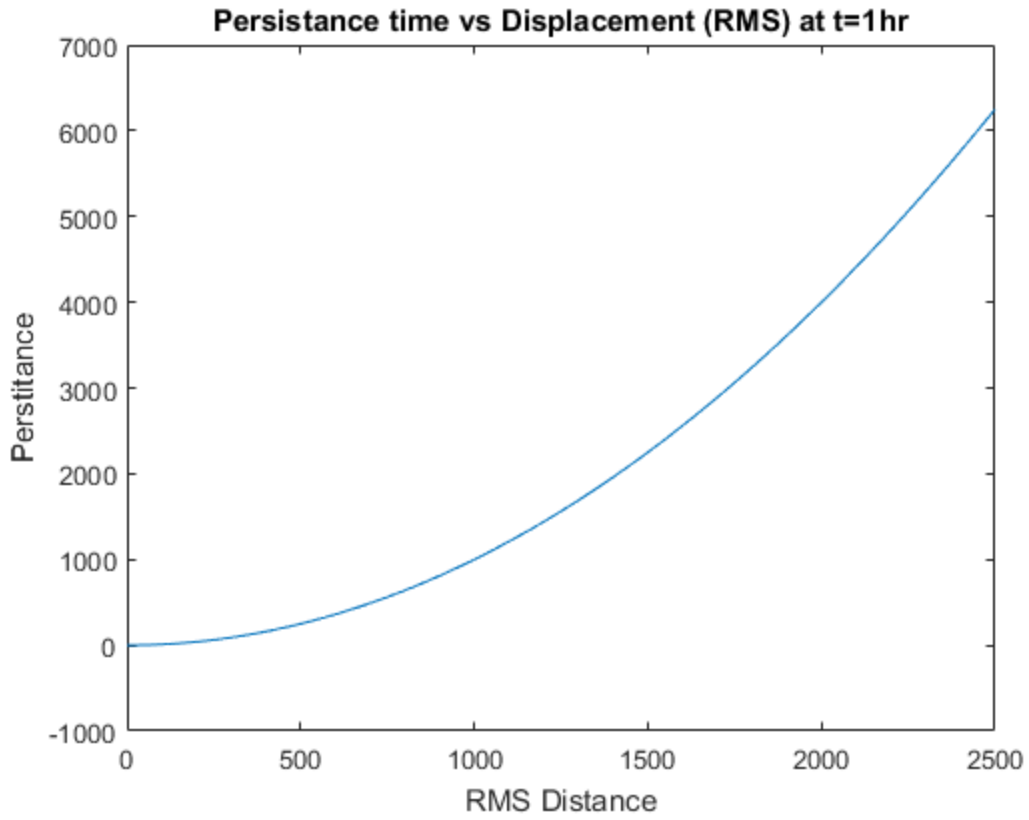
1 Bisection method for solving non-linear problems

```
all = zeros(4,3);
tol = [0.1 0.001 0.0001];
for t = 1:3
    [all(1,t), all(2,t)]=bisection_find_zero(@(x) 2*exp(-
x)+3*exp(-2*x)==x,2,0.1,tol(t));
    [all(3,t), all(4,t)]=bisection_find_zero(@(x) 2*exp(-
x)+3*exp(-2*x)==x,2,3,tol(t));
end
figure
plot(flip(log(tol)),all)
title('Bisection method: 2*exp(-x)+3*exp(-2*x)=x')
xlabel('Logarithmic decrease in tolerance')
ylabel('X')
legend('MIN: pmin = 2 & pmax = 0.1',...
    'MAX: pmin = 2 & pmax = 0.1',...
    'MIN: pmin = 2 & pmax = 3',...
    'MAX: pmin = 2 & pmax = 3')
```



2 Using fzero to solve non-linear problems

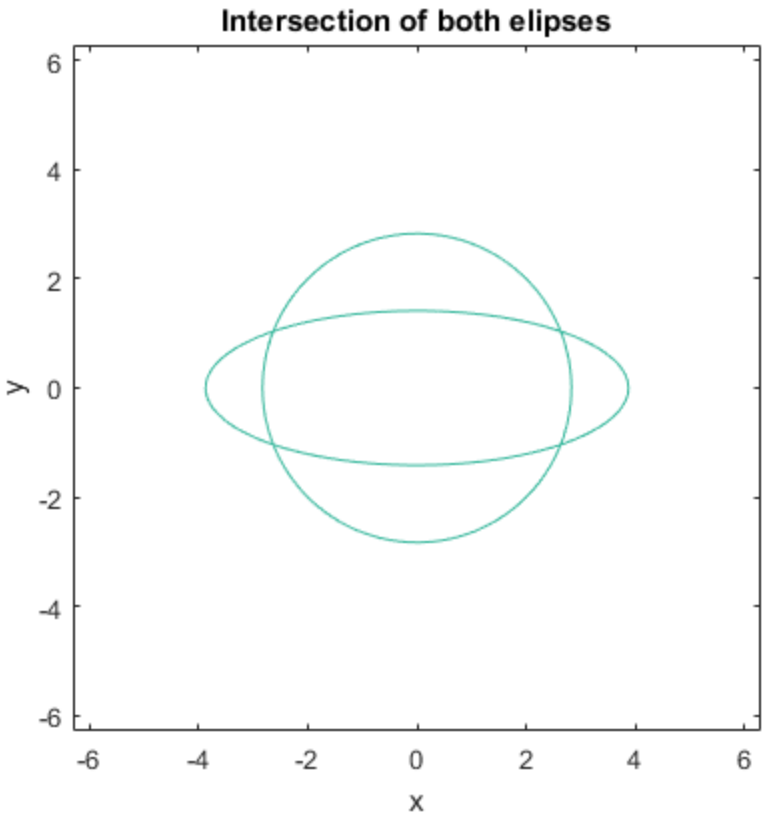
```
sol=zeros(1,2501);
for d = 0:2500
    sol(d+1)=fzero(@(P) d^2==sqrt(2*(P-P^2*(1-exp(-1/P)))),d^2/1000);
end
figure
plot(0:2500,sol)
title('Persistance time vs Displacement (RMS) at t=1hr')
xlabel('RMS Distance')
ylabel('Perstitance')
```



3 Solving systems of non-linear problems using fsolve

```
sys = @(x) [
    x(1)^2+x(2)^2-8;...
    x(1)^2/15+x(2)^2/2-1 ...
    ];
sol=fsolve(sys,[1; 1],optimset('Display','off'));
sol2=fsolve(sys,[-1; 1],optimset('Display','off'));
figure
ezplot('x^2+y^2=8'); axis equal; hold on
ezplot('x^2/15+y^2/2=1')
title('Intersection of both elipses')
disp('Part 3: Both solutions are correct, the functions cross at 4
 points.')
```

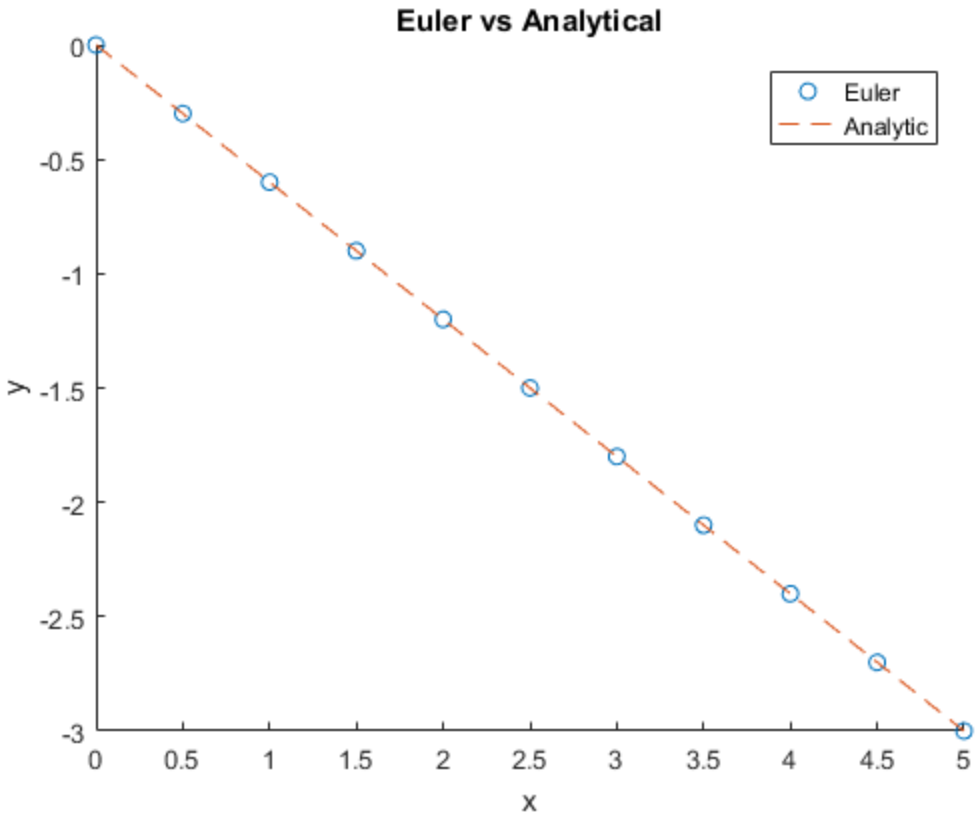
```
Part 3: Both solutions are correct, the functions cross at 4 points.
```

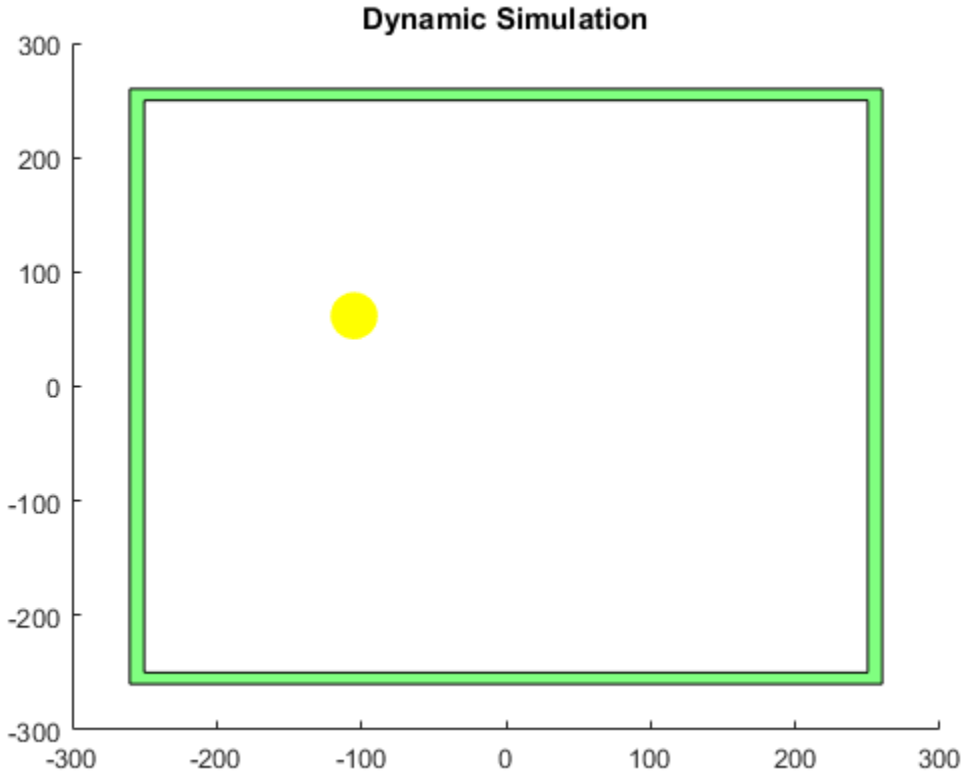


4 Simulating balls moving in a box and colliding

```
h=0.1; % step's size
N=10; % number of steps
y(1)=0;
x(1)=0;
for n=1:N
    y(n+1)= y(n)-h*3;
    x(n+1)=x(n)+5*h;
end
figure
hold on
plot(x,y,'o')
x2=0:0.5:5;
plot(x2,-3*x2/5,'--')
title('Euler vs Analytical')
xlabel('x')
ylabel('y')
legend('Euler','Analytic')
% Simulation

h=0.1; % step's size
N=5000; % number of steps
```

```
y2(1)=0;
x2(1)=0;
v=[5,-3];
[wallx,wally]=deal(250);
r=10;

for n=1:N
    if at_vertical_wall(x2(n),y2(n),v(1),v(2),wallx,h,r)
%         function [contact] =
 at_vertical_wall(x,y,vx,vy,wallx,tstep,r)
%         contact = x+vx*tstep+r >= wallx || x+vx*tstep-r <= -wallx ;
%         end
        v=vertical_wall_velocities(v(1),v(2));
%         function [vx2,vy2]=vertical_wall_velocities(vx1,vy1)
%         [vx2,vy2]=deal([vx1,vy1].*[-1,1]);
%         end
    end
    if at_horizontal_wall(x2(n),y2(n),v(1),v(2),wally,h,r)
%         function [contact] =
 at_horizontal_wall(x,y,vx,vy,wally,tstep,r)
%         contact = y+vy*tstep+r >= wally || y+vy*tstep-r <= -wally;
%         end
        v=horizontal_wall_velocities(v(1),v(2));
%         function [vx2,vy2]=horizontal_wall_velocities(vx1,vy1)
%         [vx2,vy2]=deal([vx1,vy1].*[1,-1]);
%         end
    end
    y2(n+1)= y2(n)+h*v(2);
    x2(n+1)=x2(n)+h*v(1);
end

figure
hold on
for t = 1:10:length(x2)
   rectangle('Position',[-wallx-10 -wally-10 520 520],'FaceColor',[0.5
 1 0.5])
   rectangle('Position',[-wallx -wally 500 500],'FaceColor',[1 1 1])
   scatter(x2(t),y2(t),pi*r^2,[1 1 0],'filled')
   drawnow
end
xlim([-300 300])
ylim([-300 300])
title('Dynamic Simulation')
```


Euler vs Analytical
Euler
Analytic
0
-0.5
-1
-1.5
-2
-2.5
-3
y
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
x


Dynamic Simulation
300
200
100
0
-100
-200
-300
-300
-200
-100
0
100
200
300

Published with MATLAB® R2016a