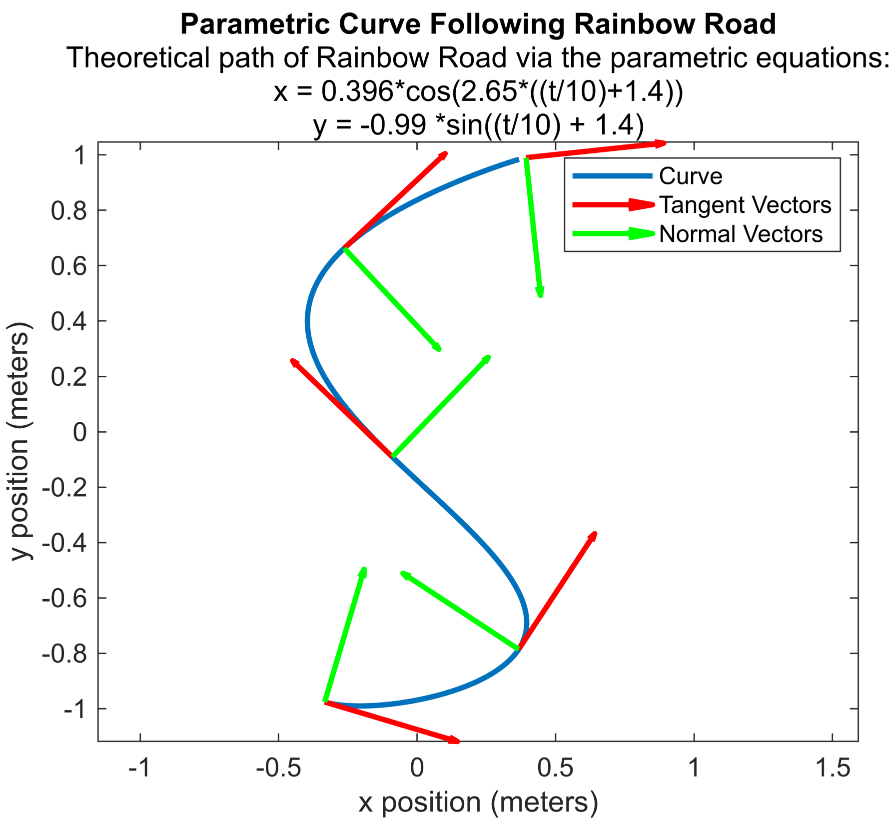
QEA 2 Rainbow Road Project Code

By Joseph Vazhaeparampil & Kuhu Jayaswal

Exercise 6.1

```
clear all; clc;
clear output;

syms t;
x = 0.396*cos(2.65*((t/10)+1.4));
y = -0.99 *sin((t/10) + 1.4);

figure;
plot(double(subs(x,t, linspace(0, 3.2 * 10, 300))), double(subs(y,t, linspace(0,
3.2 * 10, 300))), 'LineWidth', 2);
axis equal;
xlabel('x position (meters)');
ylabel('y position (meters)');
title('Parametric Curve Following Rainbow Road');
subtitle({'Theoretical path of Rainbow Road via the parametric equations:','x =
0.396*cos(2.65*((t/10)+1.4))', 'y = -0.99 *sin((t/10) + 1.4)'})
hold on;

dx = diff(x, t);
dy = diff(y, t);
t_vals = linspace(0, 3.3 * 10, 5);

speed = sqrt(dx.^2 + dy.^2);

dx_norm = dx/speed;
dy_norm = dy/speed;

x_sample = double(subs(x, t, t_vals));
y_sample = double(subs(y, t, t_vals));

dx_sample = double(subs(dx_norm, t, t_vals));
dy_sample = double(subs(dy_norm, t, t_vals));

qx = diff(dx_norm, t);
qy = diff(dy_norm, t);

q_norm = sqrt(qx^2 + qy^2);
Nx = qx/q_norm;
Ny = qy/q_norm;

Nx_sample = double(subs(Nx, t, t_vals));
Ny_sample = double(subs(Ny, t, t_vals));
```

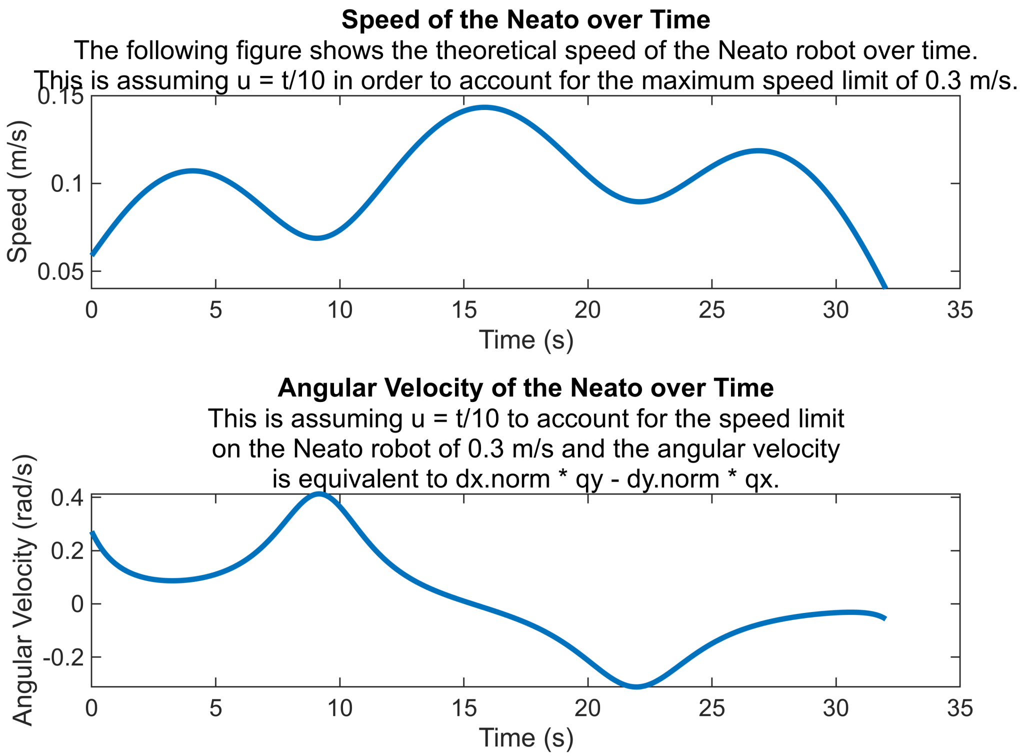
```
for i = 1:length(t_vals)
% tangent vectors
quiver(x_sample(i), y_sample(i), dx_sample(i), dy_sample(i), 0.5, 'r', 'LineWidth',
2);
% normal vectors
quiver(x_sample(i), y_sample(i), Nx_sample(i), Ny_sample(i), 0.5, 'g', 'LineWidth',
2);
end
legend('Curve', 'Tangent Vectors', 'Normal Vectors');
```



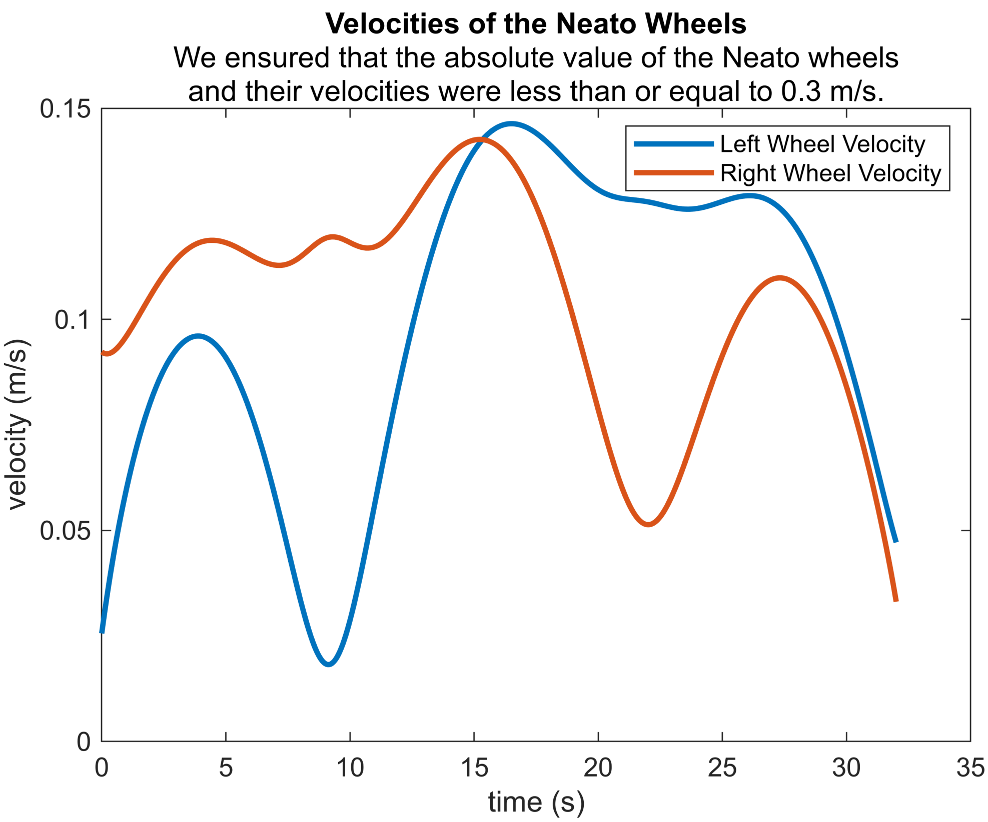
Exercise 6.2

```
d = 0.245; % wheelbase (meters)
u = (linspace(0, 32, 1000));
w = (dx_norm * qy - dy_norm * qx);

% compute left and right wheel velocities
v_l = speed - (d/2)*w;
v_r = speed + (d/2)*w;

figure;
subplot(2,1,1);
plot(u, double(subs(speed, u)), LineWidth=2);
xlabel('Time (s)');
ylabel('Speed (m/s)');
```

```
title('Speed of the Neato over Time');
subtitle({'The following figure shows the theoretical speed of the Neato robot over
time.', 'This is assuming u = t/10 in order to account for the maximum speed limit
of 0.3 m/s.'})
subplot(2,1,2);
plot(u, double(subs(w, u)), LineWidth=2);
xlabel('Time (s)');
ylabel('Angular Velocity (rad/s)');
title('Angular Velocity of the Neato over Time')
subtitle({'This is assuming u = t/10 to account for the speed limit',  'on the
Neato robot of 0.3 m/s and the angular velocity', 'is equivalent to dx.norm * qy -
dy.norm * qx.'})
```




```
figure;
plot(u, double(subs(v_l, u)), LineWidth=2);
hold on;
plot(u, double(subs(v_r, u)), LineWidth=2);
xlabel('time (s)');
ylabel('velocity (m/s)')
title('Velocities of the Neato Wheels')
legend('Left Wheel Velocity', 'Right Wheel Velocity')
subtitle({'We ensured that the absolute value of the Neato wheels', 'and their
velocities were less than or equal to 0.3 m/s.'})
```



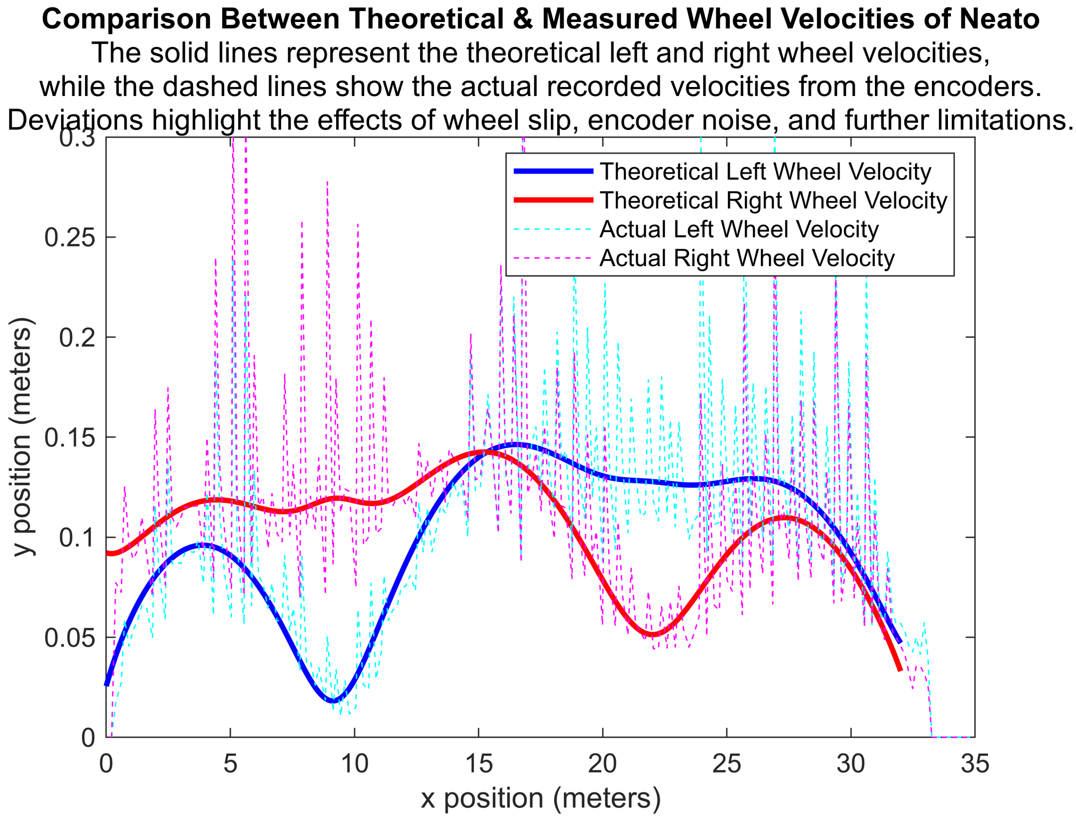

```
%Exercise 6.3
neatov3.connect('192.168.16.135');
```

```
Connecting to the Neato.
Testing connection.
Connection successful.
```

```
t_range = 0:0.1:32;

%evaluate vl(t) and vr(t) for t = t_range
vl_range = double(subs(v_l,t_range));
vr_range = double(subs(v_r,t_range));

%initialize recorded data
recorded_data = [];

%set the timer to zero
tic;
iter = 0;
%loop until 60 seconds have passed
    while toc<=35

%set t_in to the amount of time that has elapsed
t_in = toc;

vl_out = interp1(t_range,vl_range,t_in);
```

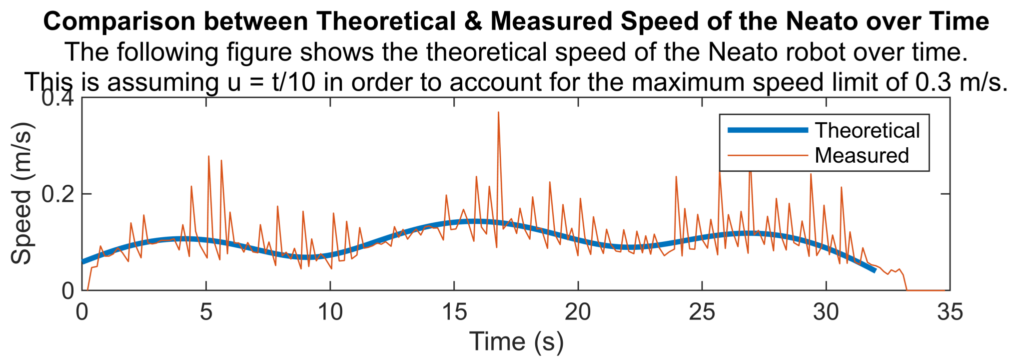
```
vr_out = interp1(t_range,vr_range,t_in);

%command the left and right wheel velocities (in meters/second)
neatov3.setVelocities(vl_out, vr_out);

sensors = neatov3.receive();

%if there is data to collect
    if length(sensors.encoders)>=2
        iter = iter + 1;
%left encoder measurement
xl = sensors.encoders(1);

%right encoder measurement
xr = sensors.encoders(2);

%log data in matrix
recorded_data(iter,:)=[t_in,xl,xr,vl_out,vr_out];
    end
end

%set the wheel velocities to zero
neatov3.setVelocities(0,0);

%by waiting until you get sensor data
sensors = neatov3.receive();
```

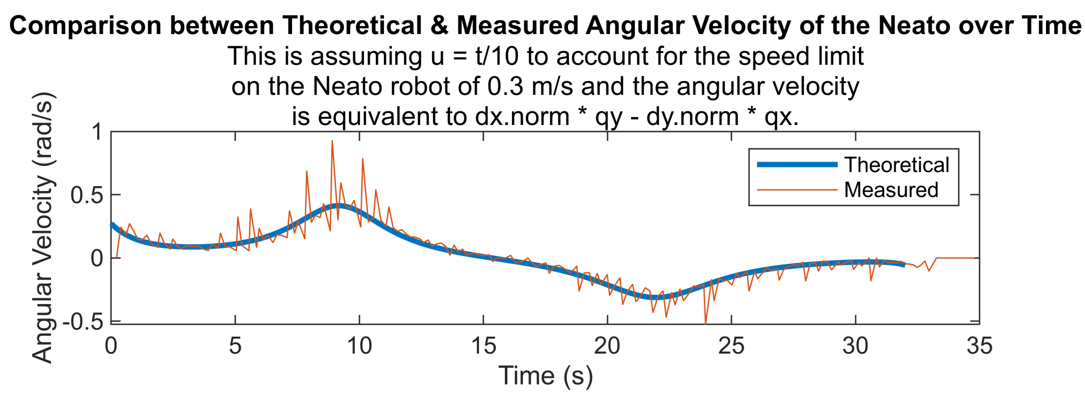
```
% Exercise 6.4
v_l_measured = diff(recorded_data(:, 2))./diff(recorded_data(:, 1));
v_r_measured = diff(recorded_data(:, 3))./diff(recorded_data(:, 1));

figure;
plot(u, double(subs(v_l, u)), LineWidth=2, Color='b');
hold on;
plot(u, double(subs(v_r, u)), LineWidth=2, Color='r');
plot(recorded_data(2:end, 1), v_l_measured, 'c--')
plot(recorded_data(2:end, 1), v_r_measured, 'm--')
ylim([0 .3])
xlabel('x position (meters)');
ylabel('y position (meters)');
title('Comparison Between Theoretical & Measured Wheel Velocities of Neato');
legend('Theoretical Left Wheel Velocity', 'Theoretical Right Wheel Velocity',
'Actual Left Wheel Velocity', 'Actual Right Wheel Velocity');
subtitle({'The solid lines represent the theoretical left and right wheel
velocities,', 'while the dashed lines show the actual recorded velocities from the
encoders.', 'Deviations highlight the effects of wheel slip, encoder noise, and
further limitations.'})
```




```
x_0 = -0.33373; %initial x_pos
y_0 = -0.9756; %initial y_pos

theta_0 = 0.01683/0.05649;
t_step = diff(recorded_data(:,1));

v_l_measured = diff(recorded_data(:,2))./t_step;
v_r_measured = diff(recorded_data(:,3))./t_step;
v_measured = (v_l_measured + v_r_measured )./2;
omega_measured = (v_l_measured  - v_r_measured )./d;

%Linear Speed and Angular Velocity Comparison Plots
figure;
subplot(2,1,1);
plot(u, double(subs(speed, u)), LineWidth=2);
hold on;
plot(recorded_data(2:end,1), v_measured);
hold off;
legend('Theoretical', 'Measured')

xlabel('Time (s)');
ylabel('Speed (m/s)');
title('Comparison between Theoretical & Measured Speed of the Neato over Time');
```

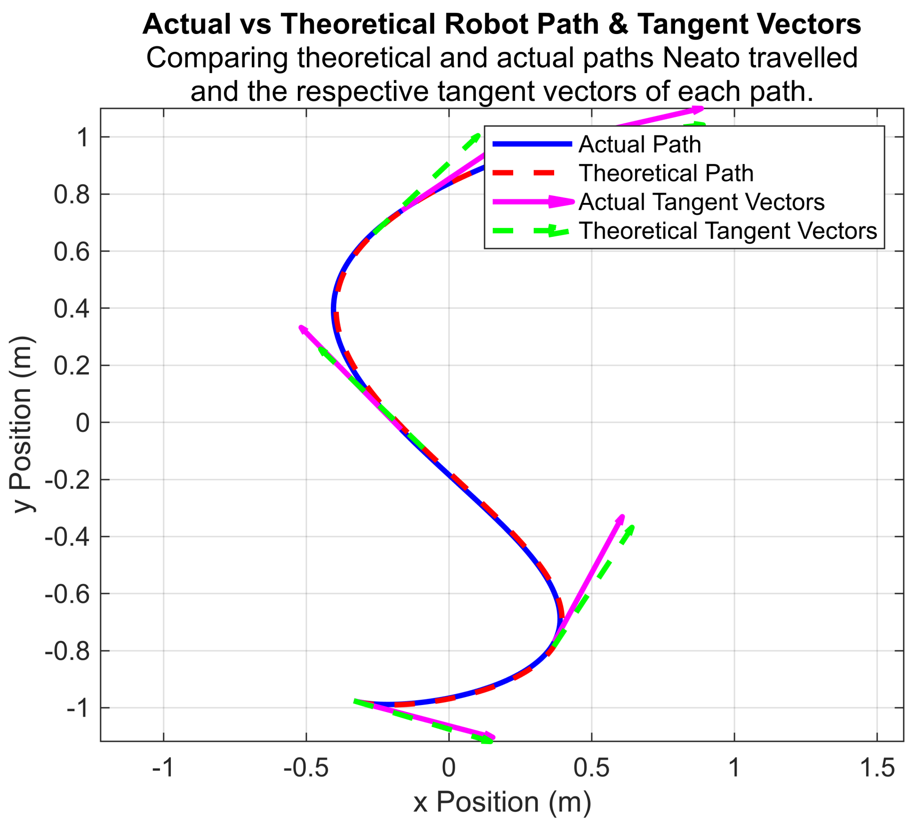
```
subtitle({'The following figure shows the theoretical speed of the Neato robot over
time.', 'This is assuming u = t/10 in order to account for the maximum speed limit
of 0.3 m/s.'})
subplot(2,1,2);
plot(u, double(subs(w, u)), LineWidth=2);
hold on;
plot(recorded_data(2:end,1), -omega_measured);
legend('Theoretical', 'Measured');
xlabel('Time (s)');
ylabel('Angular Velocity (rad/s)');
title('Comparison between Theoretical & Measured Angular Velocity of the Neato over
Time')
subtitle({'This is assuming u = t/10 to account for the speed limit',  'on the
Neato robot of 0.3 m/s and the angular velocity', 'is equivalent to dx.norm * qy -
dy.norm * qx.'})
```






```
theta_measured =  theta_0 + cumsum(omega_measured .* t_step);
dx_measured = v_measured .* cos(theta_measured) .* t_step;
dy_measured = v_measured .* -sin(theta_measured) .* t_step;

x_measured = (x_0 + cumtrapz(dx_measured));
y_measured = y_0 + cumtrapz(dy_measured);

figure;
t2_vals = [2 50 100 150 190];
```

```
p1 = plot(x_measured, y_measured, 'b', 'LineWidth', 2, 'DisplayName', 'Actual
Path');
hold on;
p2 = plot(double(subs(x,t, linspace(0, 3.2 * 10, 300))), double(subs(y,t,
linspace(0, 3.2 * 10, 300))), 'r--', 'LineWidth', 2, 'DisplayName', 'Theoretical
Path');
xlabel('x Position (m)');
ylabel('y Position (m)');
axis equal;
title('Actual vs Theoretical Robot Path & Tangent Vectors');
grid on;

speed_measured = sqrt(dx_measured.^2 + dy_measured.^2);
dx_unit_measured = dx_measured ./ speed_measured;
dy_unit_measured = dy_measured ./ speed_measured;


for i = 1:length(t_vals)
t1 = quiver(x_measured(t2_vals(i)), y_measured(t2_vals(i)),
dx_unit_measured(t2_vals(i)), dy_unit_measured(t2_vals(i)), 0.5, 'm', 'LineWidth',
2, 'DisplayName', 'Actual Tangent Vectors');
t2 = quiver(x_sample(i), y_sample(i), dx_sample(i), dy_sample(i), 0.5, 'g--',
'LineWidth', 2, 'DisplayName', 'Theoretical Tangent Vectors');
end
legend([p1 p2 t1 t2]);
subtitle({'Comparing theoretical and actual paths Neato travelled', 'and the
respective tangent vectors of each path.'})
```


Actual vs Theoretical Robot Path & Tangent Vectors
Comparing theoretical and actual paths Neato travelled
and the respective tangent vectors of each path.
Actual Path
Theoretical Path
Actual Tangent Vectors
Theoretical Tangent Vectors
y Position (m)
x Position (m)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1
1.5