```
%Sneha Kabaria, srk97
%Take Home Final - ChemE 5440
%Problem 2

%------------------------------------------------------------------------
%Part B - Plot nullclines
%------------------------------------------------------------------------

%Define Variables
alpha = 10;
syms u v n

%Define equations
f(u,v,n) = alpha/(1+v^n)-u;
g(u,v,n) = alpha/(1+u^n)-v;

%Define space to plot
u_plot = linspace(-10,10,200);
u_plot_small = linspace(-20,20,200);
u_plot_pos = linspace(0.0001,10,200);
v_plot = linspace(-10,10,200);

%Plot nullclines when n = 1;
n=1;
f_nullclines_n1(u) = solve(f(u,v,n)==0,[v])
g_nullclines_n1(u) = solve(g(u,v,n)==0,[v])
figure(1);
%Plot f nullclines
subplot(2,3,1); plot(u_plot,f_nullclines_n1(u_plot),'-k')
xlabel("u"); ylabel("v"); title("f nullcline, n=1");
%Plot g nullclines
subplot(2,3,2); plot(u_plot,g_nullclines_n1(u_plot),'-r')
xlabel("u"); ylabel("v"); title("g nullcline, n=1");
%Plot together
subplot(2,3,3);plot(u_plot,f_nullclines_n1(u_plot),'-k',u_plot,g_nullclines_n1↙
(u_plot),'-r');
xlabel("u"); ylabel("v"); title("nullclines, n=1");
axis([-5 5 -35 35])
xL = xlim;
yL = ylim;
line([0 0], yL);  %y-axis
line(xL, [0 0]);  %x-axis
legend("f nullcline", "g nullcline","x-axis","y-axis")
%Plot nullclines when n = 2;
n=2;
f_nullclines_n2(u) = solve(f(u,v,n)==0,[v])
g_nullclines_n2(u) = solve(g(u,v,n)==0,[v])
f_nullcline_n2_1(u) = (-(u - 10)/u)^(1/2);
f_nullcline_n2_2(u) = -(-(u - 10)/u)^(1/2);
figure(1);
```

```
%Plot f nullclines
subplot(2,3,4); plot(u_plot_pos,f_nullcline_n2_1(u_plot_pos),'-k',u_plot_pos,↙
f_nullcline_n2_2(u_plot_pos),'--k')
xlabel("u"); ylabel("v"); title("f nullcline, n=2");
legend("f nullcline 1", "f nullcline 2")
%Plot g nullclines
subplot(2,3,5); plot(u_plot_small,g_nullclines_n2(u_plot_small),'-r')
xlabel("u"); ylabel("v"); title("g nullcline, n=2");
%Plot together
subplot(2,3,6);
plot(u_plot_pos,f_nullcline_n2_1(u_plot_pos),'-k',u_plot_pos,f_nullcline_n2_2↙
(u_plot_pos),'--k',u_plot_small,g_nullclines_n2(u_plot_small),'-r')
xlabel("u"); ylabel("v"); title("nullclines, n=2");
axis([-5 15 -20 20])
xL = xlim;
yL = ylim;
line([0 0], yL);  %y-axis
line(xL, [0 0]);  %x-axis
legend("f nullcline 1","f nullcline 2", "g nullcline","x-axis","y-axis")

%Find Intersections in each case
%n =1
u_inter_n1 = double(solve(f_nullclines_n1(u) == g_nullclines_n1(u),u))
u_inter_n2_1 = double(solve(f_nullcline_n2_1(u) == g_nullclines_n2(u),u))
u_inter_n2_2 = double(solve(f_nullcline_n2_2(u) == g_nullclines_n2(u),u))


%--------------------------------------------------------------------------
%Part C - Plot StreamPlot
%--------------------------------------------------------------------------

%Define Functions with different n
%From Part A:
%f(u,v,n) = alpha/(1+v^n)-u;
%g(u,v,n) = alpha/(1+u^n)-v;
%n = 1
f_n1(u,v) = f(u,v,1);
g_n1(u,v) = g(u,v,1);
%n = 2
g_n2(u,v) = f(u,v,2);
f_n2(u,v) = f(u,v,2);

%Calculate the Steady-States
%n=1
SS_n1 = solve(f_n1(u,v)==0, g_n1(u,v)==0,[u,v]);
u_SS_n1 = double(SS_n1.u)
v_SS_n1 = double(SS_n1.u)
%n=2
SS_n2 = solve(f_n2(u,v)==0, g_n2(u,v)==0,[u,v]);
u_SS_n2 = SS_n2.u
```

```
v_SS_n2 = SS_n2.v
%Also Calculate v's from previous u's found from nullclines
%only for n=2
u_inter_n2_1
v_inter_n2_1 = double(f_nullcline_n2_1(u_inter_n2_1))


%StreamPlot made using Wolfram Alpha Online

%--------------------------------------------------------------------------
%Part D - Construct Jacobian
%--------------------------------------------------------------------------

%Work done by hand to find the Jacobian, typed in the write-up

%--------------------------------------------------------------------------
%Part E - Find the Eigenvalues at the center SS for n=1 and n=2
%--------------------------------------------------------------------------

%n = 1
alpha = 10;
nd1 = 1; %n = 1
ss_nd1 = 2.7; %u_S = 1
[trJ1, detJ1, e_nd1_1, e_nd2_2] = eigen(ss_nd1,nd1,alpha)

%n=2
alpha = 10;
nd2 = 2; %n = 1
ss_nd2 = 2; %u_S = 1
[trJ2, detJ2, e_nd2_1, e_nd2_2] = eigen(ss_nd2,nd2,alpha)

%--------------------------------------------------------------------------
%FUNCTIONS for Problem 2
%--------------------------------------------------------------------------

%-----------Functions for Part D (not used)

%Find the determinant function
function detJ = det_fun(ss,n,alpha)

    detJ = 1 - alpha^2 * n^2 * ss^(2*n - 2) / ((ss^n + 1)^4);

end

%-----------Functions for Part E

%Eigenvalue Functions
function [trJ,detJ,eigen1,eigen2] = eigen(ss,n,alpha)

    %ss = u @ steady state ; n = cooperativity ; alpha is alpha
```

```
    trJ = -2; %Constant, solved in part D, tr(J)

    detJ = 1 - alpha^2 * n^2 * ss^(2*n - 2) / ((ss^n + 1)^4); %det(J)

    eigen1 = (trJ + sqrt(trJ^2 - 4*detJ))/2; %first eigenvalue

    eigen2 = (trJ - sqrt(trJ^2 - 4*detJ))/2; %second eigenvalue

end
```