MATLAB Programs for the Kaldor Business Cycle Model by Elmer G. Wiens

```
function [] = kaldor_business_cycle( )
%Written by Elmer G. Wiens June 2019
%Set alpha = to desired adjustment coefficient
%Run kaldor_business_cycle at command line
clear

alpha = 8; % 6 4;

figure(1);
hold on;

nxax = 0;pxax = 8; nyax = 0;pyax = 4 + 0.3;
axis([nxax pxax+1 nyax pyax])

plot(zeros(100),linspace(nyax,pyax), 'LineWidth',2)
plot(linspace(nxax,pxax+1),zeros(100), 'LineWidth',2)


title({'Kaldor Business Cycle Model'}, 'FontSize',15)
ylabel('k', 'FontSize',17)
xlabel('y', 'FontSize',17)

kaldor_bc_phase(alpha )

x0 = [1 3.5; 4 0.5; 4.1 1.5];

for i = 1:3
    y0(1) = x0(i,1);
    y0(2) = x0(i,2);
    h = 0.1;
    tsup = 25;
    time = 1;
    [t,y] = RungeKutta(tsup,h,y0,'RK_func', time, alpha);
    if (i < 3)
        plot(y(:,1), y(:,2), 'b', 'LineWidth',3);
    elseif alpha > 6
        plot(y(:,1), y(:,2), 'g', 'LineWidth',3);
    end
end

return;
end

function [  ] = kaldor_bc_phase(alpha )
%Written by Elmer G. Wiens June 2019
syms y k

s = .2; d = .2; b = .5; i1 = 1; yd = 5; yk = 4; ti = 1.8; r = 1;
ts = .6;

inv = r*tanh(b*(y - yk)) + ti - i1*k;
```

```
sav = s * (y-yd)* k + ts;

dep = d * k;

dy = alpha*(inv - sav);

k1 = (r*tanh(b*(y - yk)) + ti - ts)/(i1 + s * (y-yd));

dk = inv - dep ;

k2 = (r*tanh(b*(y - yk)) +  ti) / (i1 +  d);

kdiff = k1 - k2;

yt = double(solve(kdiff, 'y'));
kt1 = double(subs(k1, y, yt));
kt2 = double(subs(k2, y, yt));


figure(1);
hold on;

pxax = 8;

w = linspace(.1, pxax, 100);

for i=1:length(w)
    q1(i) = double(subs(k1, y, w(i)));
    q2(i) = double(subs(k2, y, w(i)));
end

plot(w,q1, 'r', 'LineWidth',3);
plot(w,q2, 'r', 'LineWidth',3);

return
end


function [t,w] = RungeKutta(tsup,h,y0,func, time, alpha)
%Written by Elmer G. Wiens June 2019
%RungeKutta method to solve y' = f(t,y), y(0) = y0
%[t,y] = sysRungeKutta(tsup,h,yo,func)
%t = (t1,t2,...,tsup) -> time vector
%y = (y1,y2,y3,...,yn) -> solution vector
%0 = left endpoint of integration
%tsup = right endpoint of integration
%h = step size
%func = f(t,y) as in yk' = fk(t,y)-> f = (f1,f2,...,fn)
%y0 = vector of initial value -> y(0) = y0
maxw = 15;  %maximum size of a w vector component

N = tsup/h;
if(mod(tsup,h) > 0)
```

```
   N = N + 1;
end
t(1) = 0;
m =length(y0);
for j=1:m
   w(1,j) = y0(j);
end
number = 0;
i = 1; h = h * time; tsup = tsup * time;
while(i <= N & number == 0)
   t(i+1) = t(i) + h;
   if(abs(t(i+1)) >= abs(tsup))
      h = abs(tsup - t(i))*time;
      t(i+1) = tsup;
   end

   [f] = feval(func,t(i),w(i,:), alpha);
   for j = 1:m
      K1(j) = h * f(j);
   end

   z = w(i,:) + .5 * K1;
   [f] = feval(func,t(i)+ h/2,z, alpha);
   for j = 1:m
      K2(j) = h * f(j);
   end

   z = w(i,:) + .5 * K2;
   [f] = feval(func,t(i)+ h/2,z, alpha);
   for j = 1:m
      K3(j) = h * f(j);
   end

   z = w(i,:) +  K3;
   [f] = feval(func,t(i)+ h,z, alpha);
   for j = 1:m
      K4(j) = h * f(j);
   end

   w(i+1,:) = w(i,:) + (1/6) * (K1 + 2 * K2 + 2 * K3 +  K4);

   i = i+1;
end

function [yp] = RK_func(t,y, alpha)
%Written by Elmer G. Wiens June 2019
s = .2; d = .2; b = .5; i1 = 1; yd = 5; yk = 4; ti = 1.8; r = 1;ts = .6;
yp(1) = alpha * (tanh(b*(y(1) - yk)) + ti  - i1* y(2)  - s * (y(1)-yd)* y(2)
- ts);
yp(2) = tanh(b*(y(1) - yk)) + ti - i1*y(2) - d * y(2);

return
```