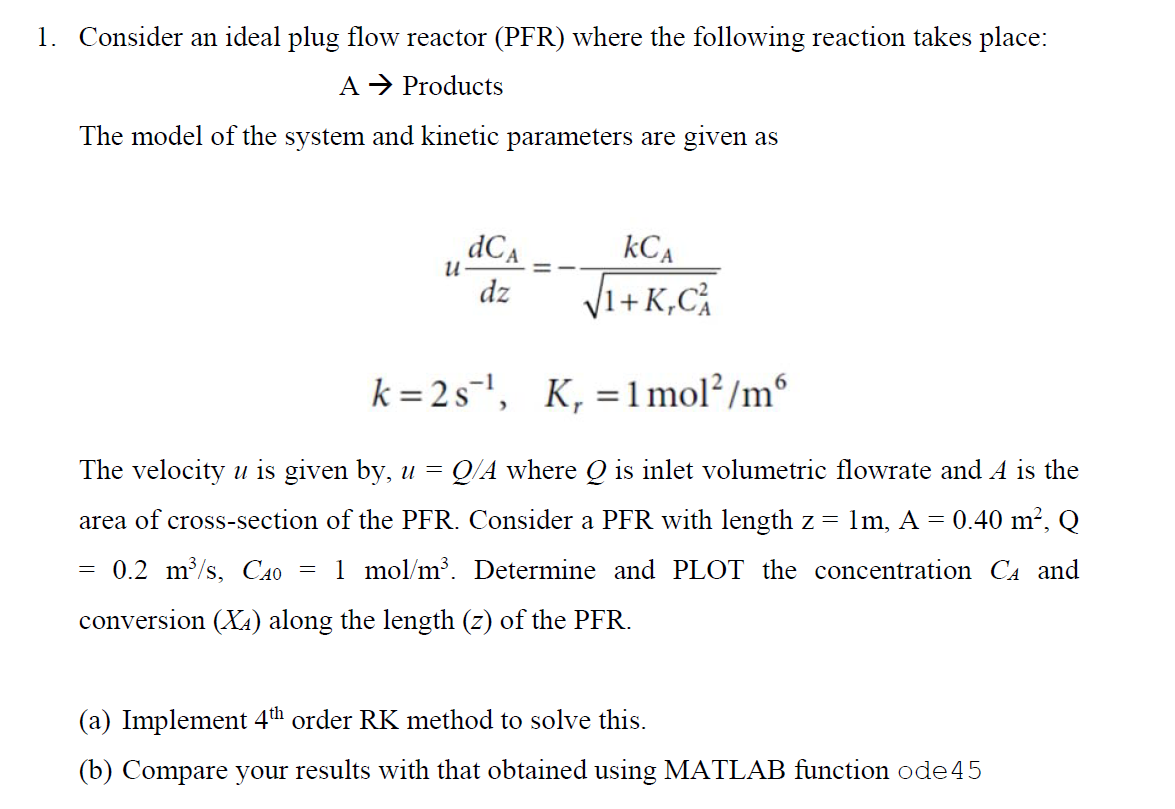
**CAPE Laboratory Assignment-3**

Instructor: Dr Debashis Sarkar

*18CH10071, Anshuman Agrawal*

**Q.1**

**Problem Statement**



**MATLAB Code**

**4th order Runge Kutta method**

**(divided the length span in 101 points)**

global u k Kr

k = 2; % s^-1

Kr = 1; % mol^2 m^-6

z = 1; % length of PFR

A = 0.4; % m^2

Q = 0.2; % m^3 s^-1

CA0 = 1; % mol m^-3

u = Q/A; % velocity

h = 0.01;

l = 0:h:z;

CA(1) = CA0;

for i=1:(z/h)

k1 = fun(CA(i));

k2 = fun(CA(i)+(h/2)\*k1);

k3 = fun(CA(i)+(h/2)\*k2);

k4 = fun(CA(i)+h\*k3);

CA(i+1) = CA(i) + (h/6)\*(k1+2\*k2+2\*k3+k4);

end

XA = (CA0 - CA)/CA0;

plot(l,CA,'-o','LineWidth',2);

hold on;

plot(l, XA,'-x','LineWidth',2);

legend('CA','XA');

function s = fun(CA)

global u k Kr

s = -(1/u)\*k\*CA/sqrt(1+Kr\*(CA^2));

end

***ode45* method**

**(length span division is default)**

global u k Kr

k = 2; % s^-1

Kr = 1; % mol^2 m^-6

z = 1; % length of PFR

A = 0.4; % m^2

Q = 0.2; % m^3 s^-1

CA0 = 1; % mol m^-3

u = Q/A; % velocity

l = [0 z];

[length,CA\_sol] = ode45(@(l,CA) -(1/u)\*k\*CA/sqrt(1+Kr\*(CA^2)),l,CA0);

XA = (CA0 - CA\_sol)/CA0;

plot(length,CA\_sol,'-o','LineWidth',2);

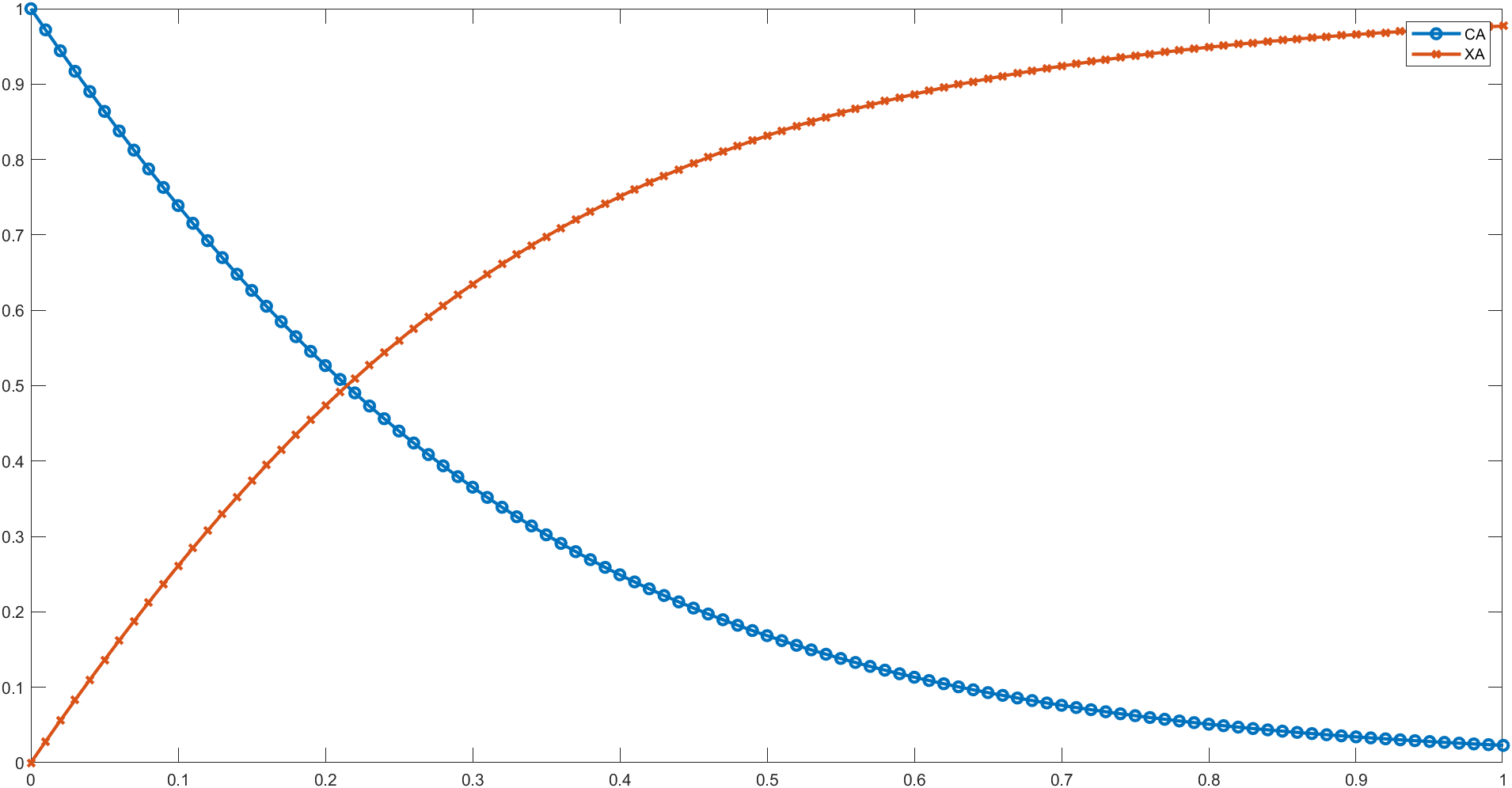
hold on;

plot(length, XA,'-x','LineWidth',2);

legend('CA','XA');

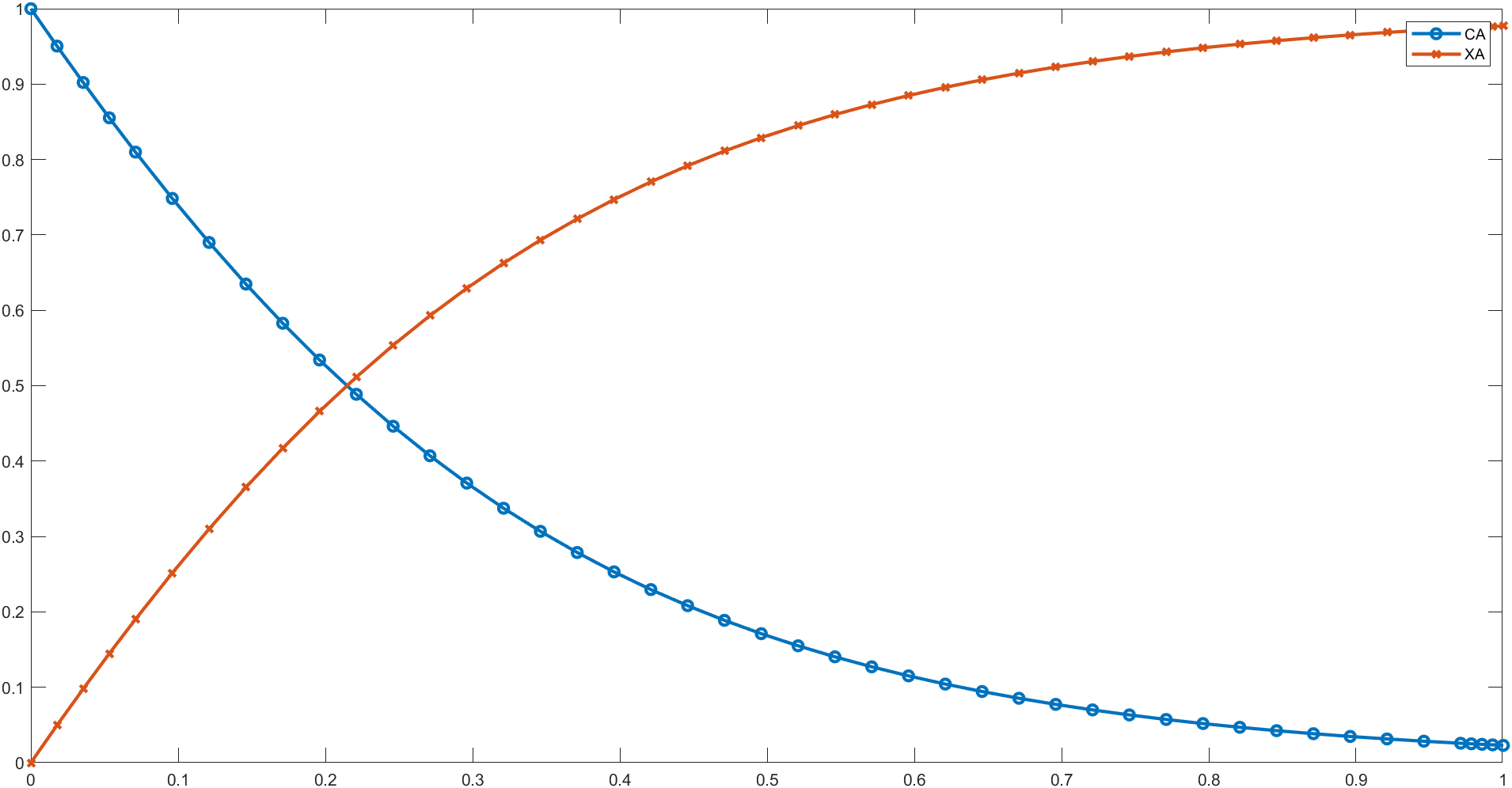
**Results**

|  |  |  |
| --- | --- | --- |
| **Length (z)** | **CA** | **XA** |
| 0 | 1 | 0 |
| 0.0100000000000000 | 0.971917607284530 | 0.0280823927154701 |
| 0.0200000000000000 | 0.944246419995474 | 0.0557535800045265 |
| 0.0300000000000000 | 0.916997436245865 | 0.0830025637541348 |
| 0.0400000000000000 | 0.890181370933788 | 0.109818629066212 |
| 0.0500000000000000 | 0.863808591438300 | 0.136191408561700 |
| 0.0600000000000000 | 0.837889052348865 | 0.162110947651135 |
| 0.0700000000000000 | 0.812432230089295 | 0.187567769910705 |
| 0.0800000000000000 | 0.787447058368295 | 0.212552941631705 |
| 0.0900000000000000 | 0.762941865438657 | 0.237058134561343 |
| 0.100000000000000 | 0.738924314172167 | 0.261075685827833 |
| 0.110000000000000 | 0.715401345954232 | 0.284598654045768 |
| 0.120000000000000 | 0.692379129369242 | 0.307620870630759 |
| 0.130000000000000 | 0.669863014584233 | 0.330136985415767 |
| 0.140000000000000 | 0.647857494245361 | 0.352142505754639 |
| 0.150000000000000 | 0.626366171581560 | 0.373633828418440 |
| 0.160000000000000 | 0.605391736266426 | 0.394608263733574 |
| 0.170000000000000 | 0.584935948427994 | 0.415064051572006 |
| 0.180000000000000 | 0.564999631022737 | 0.435000368977263 |
| 0.190000000000000 | 0.545582670611612 | 0.454417329388389 |
| 0.200000000000000 | 0.526684026399109 | 0.473315973600891 |
| 0.210000000000000 | 0.508301747227783 | 0.491698252772217 |
| 0.220000000000000 | 0.490432996066629 | 0.509567003933371 |
| 0.230000000000000 | 0.473074081397061 | 0.526925918602939 |
| 0.240000000000000 | 0.456220494789070 | 0.543779505210930 |
| 0.250000000000000 | 0.439866953874825 | 0.560133046125175 |
| 0.260000000000000 | 0.424007449868937 | 0.575992550131063 |
| 0.270000000000000 | 0.408635298753421 | 0.591364701246579 |
| 0.280000000000000 | 0.393743195239999 | 0.606256804760001 |
| 0.290000000000000 | 0.379323268640321 | 0.620676731359679 |
| 0.300000000000000 | 0.365367139812924 | 0.634632860187076 |
| 0.310000000000000 | 0.351865978410875 | 0.648134021589125 |
| 0.320000000000000 | 0.338810559722111 | 0.661189440277889 |
| 0.330000000000000 | 0.326191320471886 | 0.673808679528114 |
| 0.340000000000000 | 0.313998413039748 | 0.686001586960252 |
| 0.350000000000000 | 0.302221757628794 | 0.697778242371206 |
| 0.360000000000000 | 0.290851092009854 | 0.709148907990146 |
| 0.370000000000000 | 0.279876018545160 | 0.720123981454840 |
| 0.380000000000000 | 0.269286048273213 | 0.730713951726787 |
| 0.390000000000000 | 0.259070641907566 | 0.740929358092434 |
| 0.400000000000000 | 0.249219247666050 | 0.750780752333950 |
| 0.410000000000000 | 0.239721335903206 | 0.760278664096794 |
| 0.420000000000000 | 0.230566430567063 | 0.769433569432937 |
| 0.430000000000000 | 0.221744137542065 | 0.778255862457935 |
| 0.440000000000000 | 0.213244169973243 | 0.786755830026757 |
| 0.450000000000000 | 0.205056370693092 | 0.794943629306908 |
| 0.460000000000000 | 0.197170731892744 | 0.802829268107256 |
| 0.470000000000000 | 0.189577412193453 | 0.810422587806547 |
| 0.480000000000000 | 0.182266751283998 | 0.817733248716002 |
| 0.490000000000000 | 0.175229282294845 | 0.824770717705155 |
| 0.500000000000000 | 0.168455742081690 | 0.831544257918310 |
| 0.510000000000000 | 0.161937079589738 | 0.838062920410262 |
| 0.520000000000000 | 0.155664462466452 | 0.844335537533548 |
| 0.530000000000000 | 0.149629282085070 | 0.850370717914930 |
| 0.540000000000000 | 0.143823157134238 | 0.856176842865763 |
| 0.550000000000000 | 0.138237935921213 | 0.861762064078787 |
| 0.560000000000000 | 0.132865697527500 | 0.867134302472500 |
| 0.570000000000000 | 0.127698751946725 | 0.872301248053275 |
| 0.580000000000000 | 0.122729639325389 | 0.877270360674611 |
| 0.590000000000000 | 0.117951128417959 | 0.882048871582041 |
| 0.600000000000000 | 0.113356214358697 | 0.886643785641303 |
| 0.610000000000000 | 0.108938115843920 | 0.891061884156080 |
| 0.620000000000000 | 0.104690271809936 | 0.895309728190065 |
| 0.630000000000000 | 0.100606337683945 | 0.899393662316055 |
| 0.640000000000000 | 0.0966801812776703 | 0.903319818722330 |
| 0.650000000000000 | 0.0929058783864113 | 0.907094121613589 |
| 0.660000000000000 | 0.0892777081496815 | 0.910722291850319 |
| 0.670000000000000 | 0.0857901482235074 | 0.914209851776493 |
| 0.680000000000000 | 0.0824378698088823 | 0.917562130191118 |
| 0.690000000000000 | 0.0792157325757441 | 0.920784267424256 |
| 0.700000000000000 | 0.0761187795171693 | 0.923881220482831 |
| 0.710000000000000 | 0.0731422317642193 | 0.926857768235781 |
| 0.720000000000000 | 0.0702814833880186 | 0.929718516611981 |
| 0.730000000000000 | 0.0675320962121598 | 0.932467903787840 |
| 0.740000000000000 | 0.0648897946553905 | 0.935110205344610 |
| 0.750000000000000 | 0.0623504606217211 | 0.937649539378279 |
| 0.760000000000000 | 0.0599101284525659 | 0.940089871547434 |
| 0.770000000000000 | 0.0575649799532766 | 0.942435020046723 |
| 0.780000000000000 | 0.0553113395044226 | 0.944688660495577 |
| 0.790000000000000 | 0.0531456692663861 | 0.946854330733614 |
| 0.800000000000000 | 0.0510645644842639 | 0.948935435515736 |
| 0.810000000000000 | 0.0490647488986730 | 0.950935251101327 |
| 0.820000000000000 | 0.0471430702668295 | 0.952856929733171 |
| 0.830000000000000 | 0.0452964959971904 | 0.954703504002810 |
| 0.840000000000000 | 0.0435221089000092 | 0.956477891099991 |
| 0.850000000000000 | 0.0418171030553278 | 0.958182896944672 |
| 0.860000000000000 | 0.0401787797992187 | 0.959821220200781 |
| 0.870000000000000 | 0.0386045438284684 | 0.961395456171532 |
| 0.880000000000000 | 0.0370918994233636 | 0.962908100576636 |
| 0.890000000000000 | 0.0356384467877855 | 0.964361553212215 |
| 0.900000000000000 | 0.0342418785054278 | 0.965758121494572 |
| 0.910000000000000 | 0.0328999761106292 | 0.967100023889371 |
| 0.920000000000000 | 0.0316106067720316 | 0.968389393227968 |
| 0.930000000000000 | 0.0303717200870520 | 0.969628279912948 |
| 0.940000000000000 | 0.0291813449849650 | 0.970818655015035 |
| 0.950000000000000 | 0.0280375867362457 | 0.971962413263754 |
| 0.960000000000000 | 0.0269386240657004 | 0.973061375934300 |
| 0.970000000000000 | 0.0258827063668235 | 0.974117293633177 |
| 0.980000000000000 | 0.0248681510147526 | 0.975131848985247 |
| 0.990000000000000 | 0.0238933407751439 | 0.976106659224856 |
| 1 | 0.0229567213062686 | 0.977043278693731 |



***Plot using 4th order RK method***

|  |  |  |
| --- | --- | --- |
| **Length (z)** | **CA** | **XA** |
| 0 | 1 | 0 |
| 0.0177617192929090 | 0.950403608750618 | 0.0495963912493823 |
| 0.0355234385858181 | 0.902131509512916 | 0.0978684904870839 |
| 0.0532851578787271 | 0.855243212286957 | 0.144756787713043 |
| 0.0710468771716361 | 0.809794339239428 | 0.190205660760572 |
| 0.0960468771716361 | 0.748359464386972 | 0.251640535613028 |
| 0.121046877171636 | 0.689997686931994 | 0.310002313068006 |
| 0.146046877171636 | 0.634800259563848 | 0.365199740436152 |
| 0.171046877171636 | 0.582824502330490 | 0.417175497669510 |
| 0.196046877171636 | 0.534092811967884 | 0.465907188032116 |
| 0.221046877171636 | 0.488590992364694 | 0.511409007635306 |
| 0.246046877171636 | 0.446271455946184 | 0.553728544053816 |
| 0.271046877171636 | 0.407053856643349 | 0.592946143356651 |
| 0.296046877171636 | 0.370828902247703 | 0.629171097752297 |
| 0.321046877171636 | 0.337467865973380 | 0.662532134026620 |
| 0.346046877171636 | 0.306826967393335 | 0.693173032606665 |
| 0.371046877171636 | 0.278749474585508 | 0.721250525414492 |
| 0.396046877171636 | 0.253070160828099 | 0.746929839171902 |
| 0.421046877171636 | 0.229626045831646 | 0.770373954168354 |
| 0.446046877171636 | 0.208255547726260 | 0.791744452273740 |
| 0.471046877171636 | 0.188799076675269 | 0.811200923324731 |
| 0.496046877171636 | 0.171101877199258 | 0.828898122800742 |
| 0.521046877171636 | 0.155020591059849 | 0.844979408940151 |
| 0.546046877171636 | 0.140419441194940 | 0.859580558805060 |
| 0.571046877171636 | 0.127169614393238 | 0.872830385606762 |
| 0.596046877171636 | 0.115150793270969 | 0.884849206729031 |
| 0.621046877171636 | 0.104254472327842 | 0.895745527672158 |
| 0.646046877171636 | 0.0943799168824444 | 0.905620083117556 |
| 0.671046877171636 | 0.0854332426754375 | 0.914566757324563 |
| 0.696046877171636 | 0.0773282549762214 | 0.922671745023779 |
| 0.721046877171636 | 0.0699881386657237 | 0.930011861334276 |
| 0.746046877171636 | 0.0633422134676719 | 0.936657786532328 |
| 0.771046877171636 | 0.0573251281047494 | 0.942674871895251 |
| 0.796046877171636 | 0.0518773532958500 | 0.948122646704150 |
| 0.821046877171636 | 0.0469461184495158 | 0.953053881550484 |
| 0.846046877171636 | 0.0424830614957135 | 0.957516938504286 |
| 0.871046877171636 | 0.0384436262625342 | 0.961556373737466 |
| 0.896046877171636 | 0.0347873700762389 | 0.965212629923761 |
| 0.921046877171636 | 0.0314785271393216 | 0.968521472860678 |
| 0.946046877171636 | 0.0284843784471991 | 0.971515621552801 |
| 0.971046877171636 | 0.0257748282344645 | 0.974225171765536 |
| 0.978285157878727 | 0.0250394999086548 | 0.974960500091345 |
| 0.985523438585818 | 0.0243251370455199 | 0.975674862954480 |
| 0.992761719292909 | 0.0236311431152354 | 0.976368856884765 |
| 1 | 0.0229569380053415 | 0.977043061994659 |



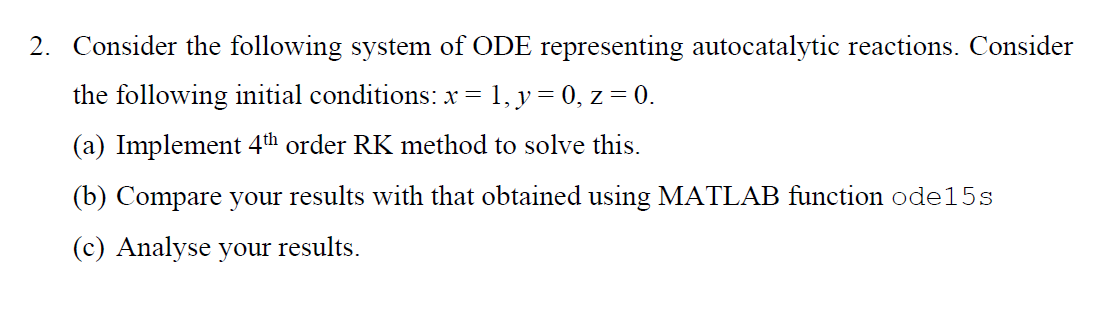
***Plot using ode45 method***

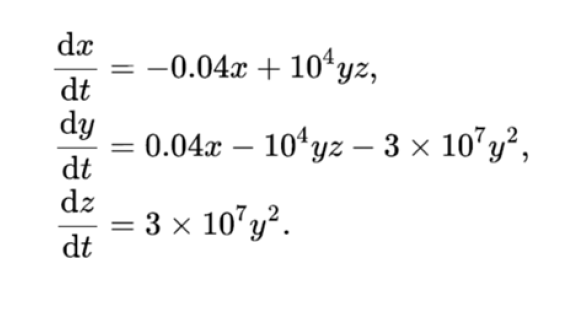
**Conclusion**

Both methods took nearly the same time for computation because even the *ode45* algorithm involves the Runge-Kutta method. However, the conventional Runge-Kutta code occupied more memory because the length was spanned over a greater number of points. Therefore, more data points were considered in an effort to increase accuracy. Both the methods gave nearly same output values with slight deviation in higher decimal places which is not very significant. The conventional Runge-Kutta code might have lost some precision while having round-off errors in the computation procedure had we used the same number of points as *ode45*.

**Q.2**

**Problem Statement**





**MATLAB Code**

**Multivariable 4th order Runge-Kutta method**

**(code with h = 0.01)**

clear all;

x(1) = 1;

y(1) = 0;

z(1) = 0;

t = 0:0.01:100;

for i=1:10000

k1 = func(t(i),x(i),y(i),z(i));

k2 = func(t(i)+0.005,x(i)+0.005\*k1(1),y(i)+0.005\*k1(2),z(i)+0.005\*k1(3));

k3 = func(t(i)+0.005,x(i)+0.005\*k2(1),y(i)+0.005\*k2(2),z(i)+0.005\*k2(3));

k4 = func(t(i)+0.01,x(i)+0.01\*k3(1),y(i)+0.01\*k3(2),z(i)+0.01\*k3(3));

x(i+1) = x(i) + (1/6)\*(k1(1)+2\*k2(1)+2\*k3(1)+k4(1));

y(i+1) = y(i) + (1/6)\*(k1(2)+2\*k2(2)+2\*k3(2)+k4(2));

z(i+1) = z(i) + (1/6)\*(k1(3)+2\*k2(3)+2\*k3(3)+k4(3));

end

t = t';

x = x';

y = y';

z = z';

plot(t,x,'-x','LineWidth',2);hold on;

plot(t,y,'-o','LineWidth',2);hold on;

plot(t,z,'-.','LineWidth',2);

legend('x','y','z');

function dXdt = func(t,x,y,z)

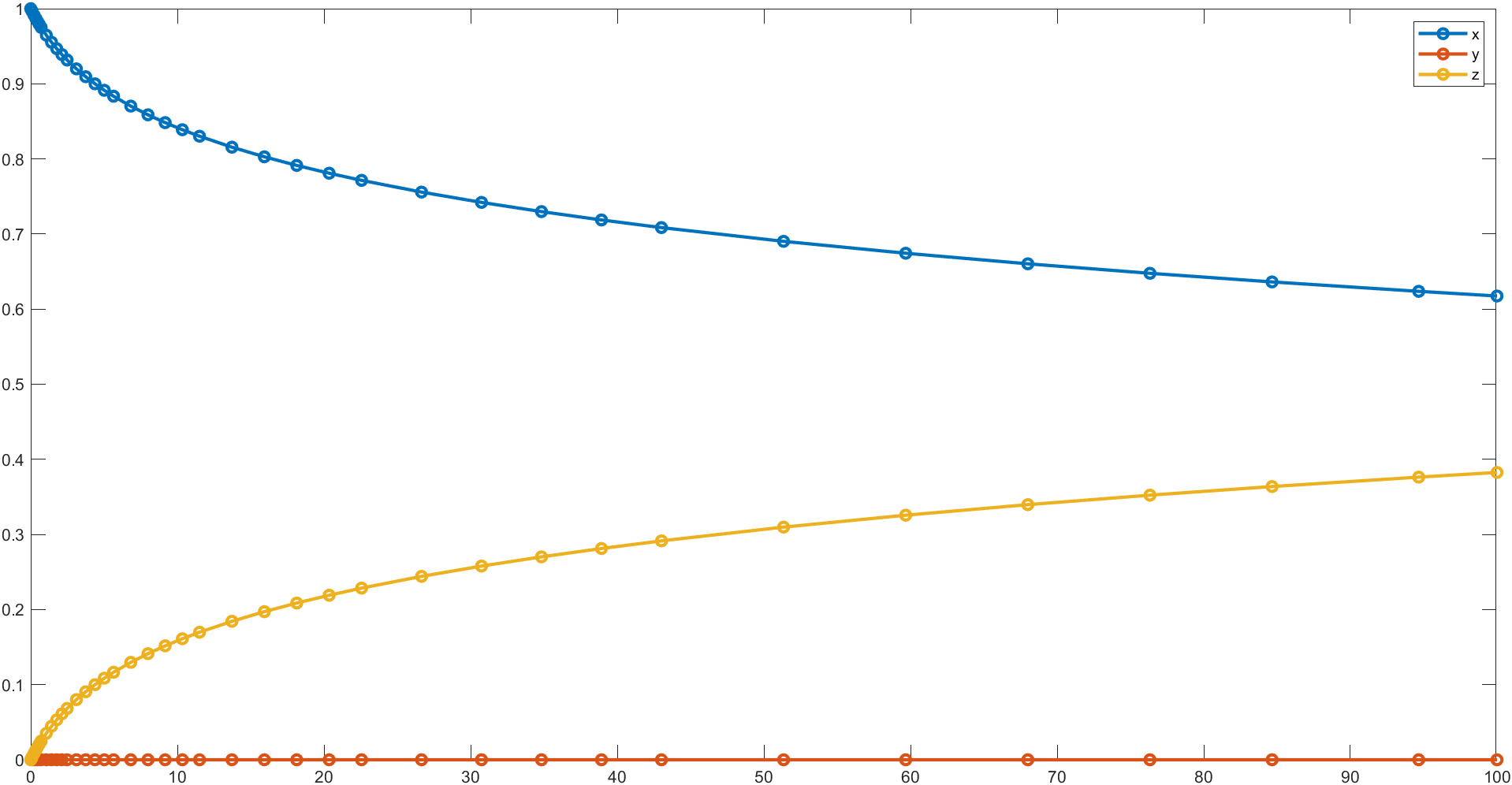
dXdt = [-0.04\*x+(10^4)\*y\*z;

0.04\*x-(10^4)\*y\*z-(3\*(10^7))\*(y^2);

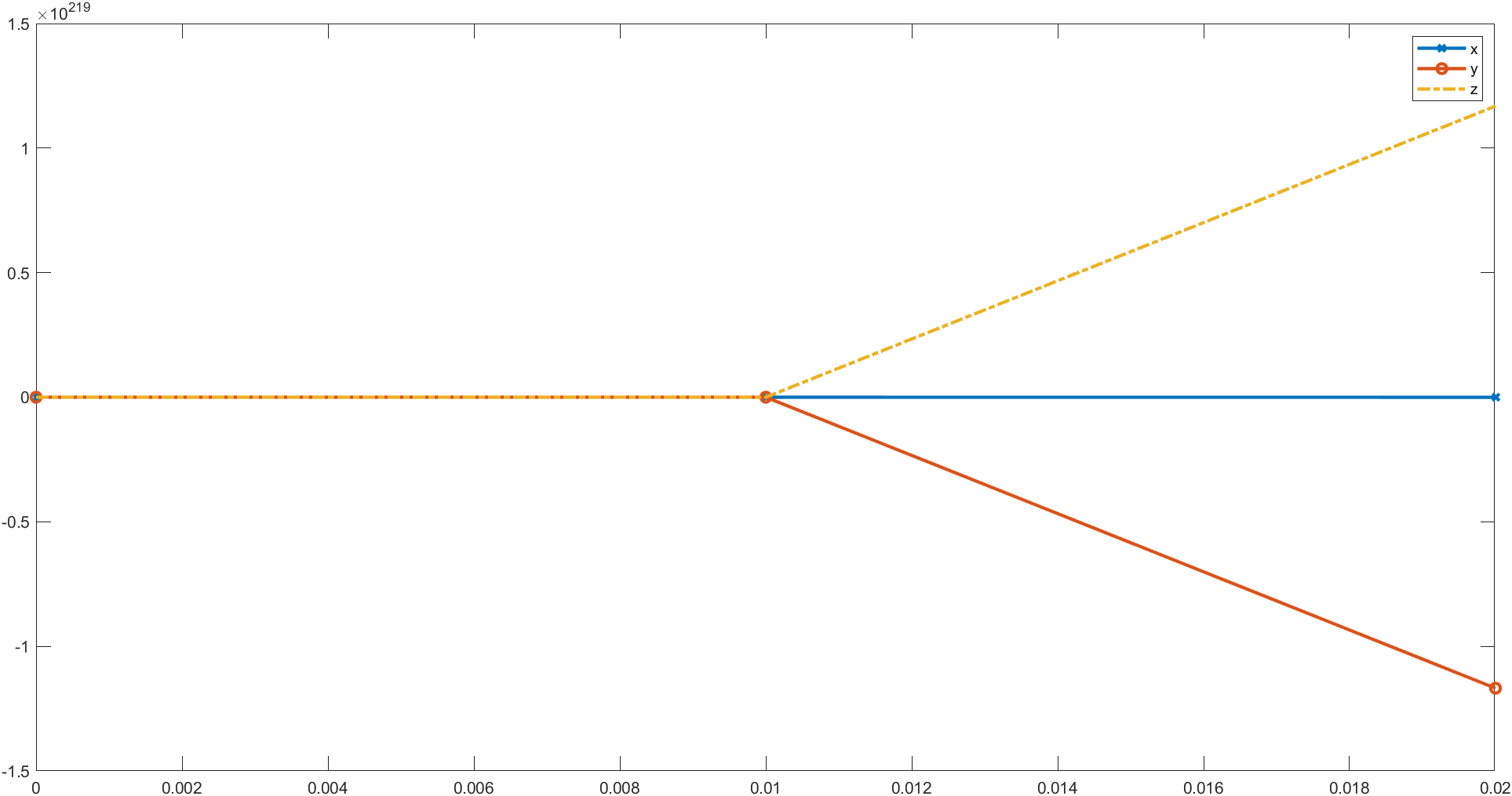
(3\*(10^7))\*(y^2)];

end

**Results**



***Plot generated with ode15s method***



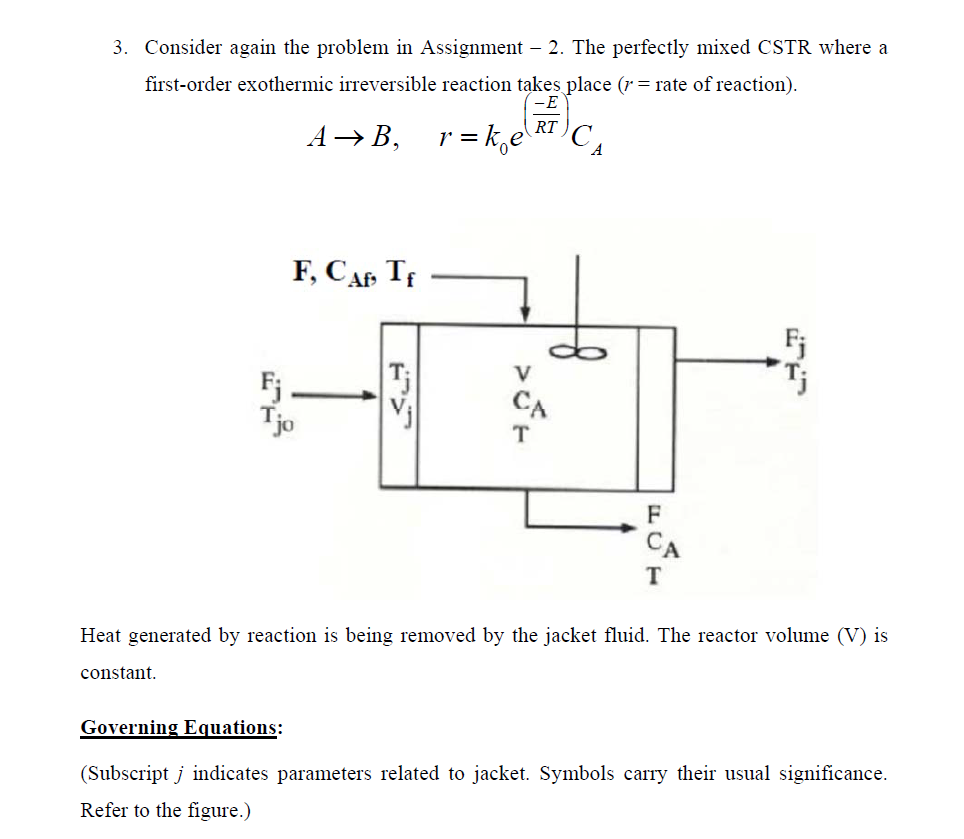
***Plot generated with multivariable 4th order RK method***

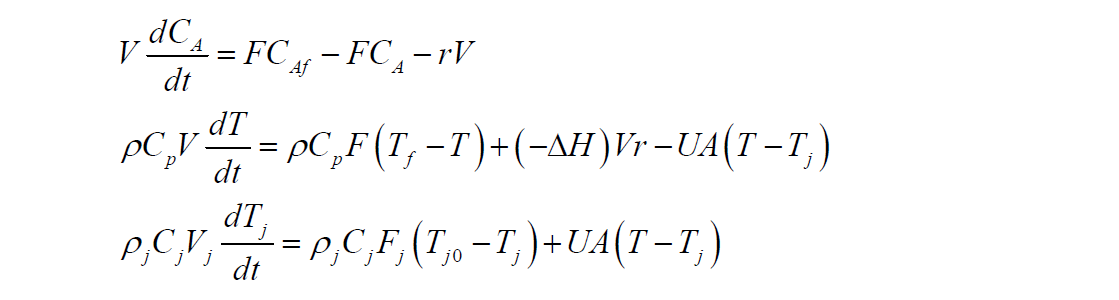
**Conclusion**

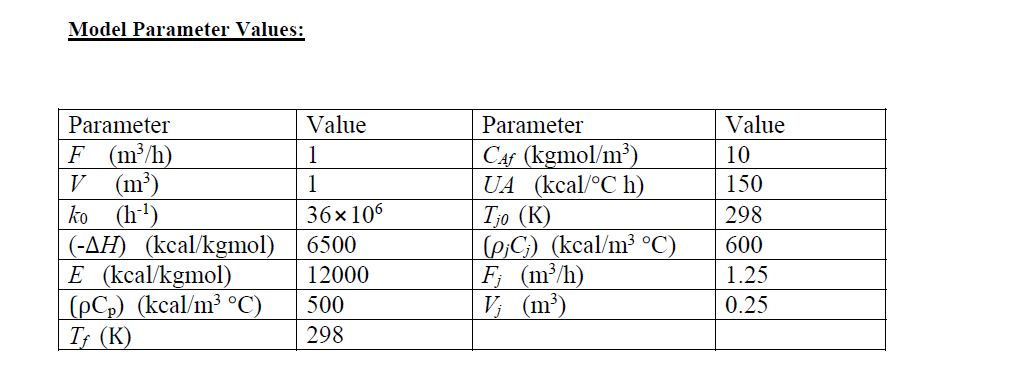
As evident from the plot obtained with the 4th order Runge-Kutta Method, the given system of differential equations is a stiff system. It means that for this method, it is highly unstable and doesn’t give desired results. Usually, one solution to such problems is by lowering the step size to a very small number but here the stiffness persists even at h values as low as 0.01. The system was tested with h = 1, h = 0.1, and h =0.01 but returned stiff results every time. However, ode15s uses a multistep solver in its algorithm which can override the stiffness of a system and return consistent results. Both methods take nearly the same computational time but Runge-Kutta method will require more amount of memory because of an unusual step size which will lead to a very large solution data set.

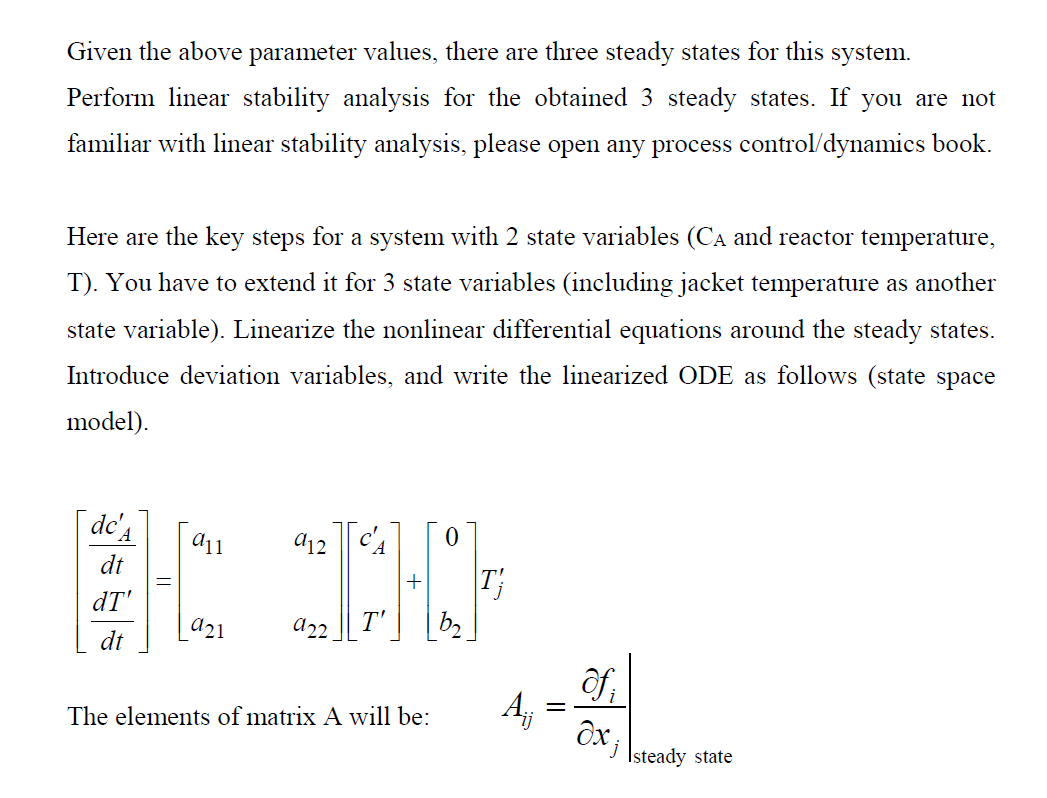
**Q.3**

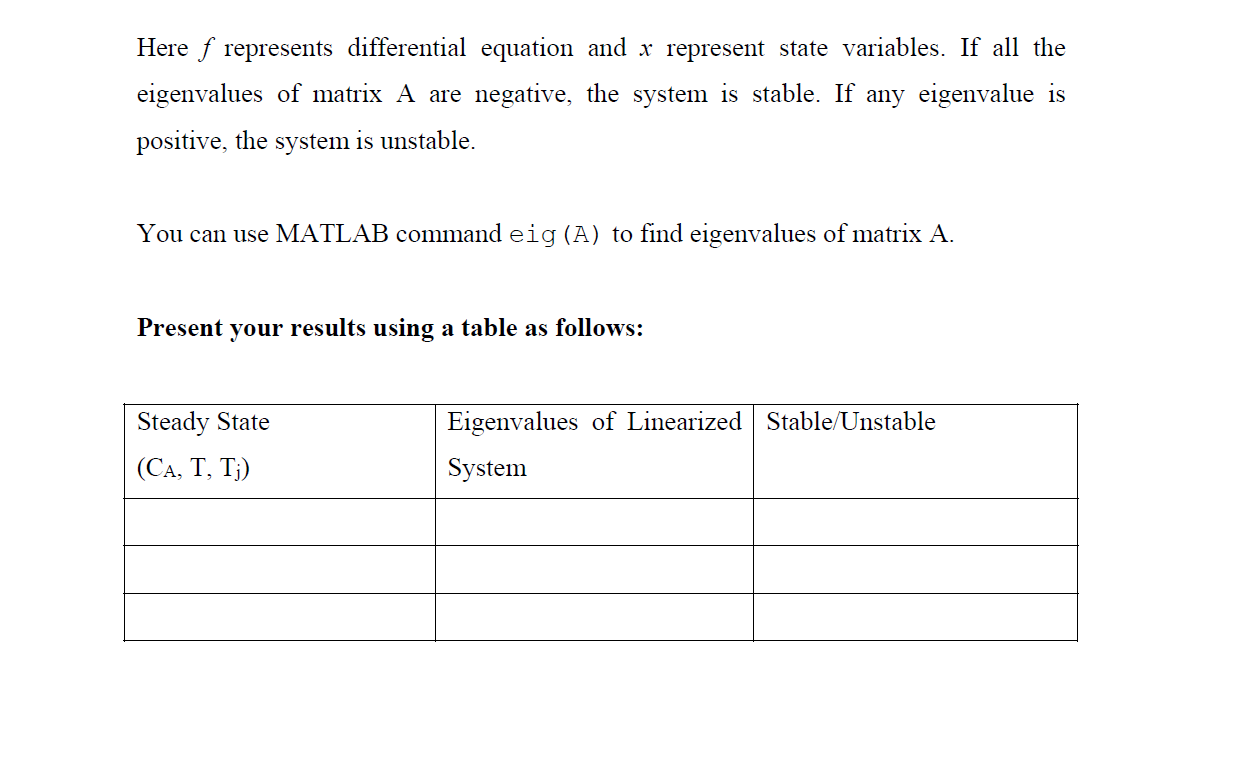
**Problem Statement**











**MATLAB Code**

clear all;

R=1.987;

F=1;

Vj=0.25;

V=1;

k0=36000000;

H=6500;

E=12000;

rhoCp=500;

Tf=298;

Caf=10;

UA=150;

Tj0=298;

rhojCj=600;

Fj=1.25;

A=zeros([20,3]);

for i=1:20 %Finding steady states

options=optimoptions('fsolve','Display','off'); %Supressing the output

z=[];

z(1)=0+rand\*(10); %Intial Guesses

z(2)=298+rand\*(100);

z(3)=298+rand\*(100);

A(i,1:3)=fsolve(@cstrsteady,z,options); %Solving the function

end

A=round(A,6);%Rounding the variables to 6 decimal places

Cs=A(:,1);

Ts=A(:,2);

Tjs=A(:,3);

Cs=unique(Cs,'stable'); %Finding only unique values and stores them in the order that is found in sol.

Ts=unique(Ts,'stable');

Tjs=unique(Tjs,'stable');

A=([Cs Ts Tjs])

syms Ca T Tj

%Forming the equations

dCdt=(F/V)\*(Caf-Ca)-k0\*exp(-E/(R\*T))\*Ca;

dTdt=(rhoCp\*F\*(Tf-T)+H\*V\*k0\*exp(-E/(R\*T))\*Ca-UA\*(T-Tj))/(rhoCp\*V);

dTjdt=(Fj/Vj)\*(Tj0-Tj)+(UA/(rhojCj\*Vj))\*(T-Tj);

Cas=A(:,1);

Ts=A(:,2);

Tjs=A(:,3);

C=zeros([3 3 3]);

for i=1:3 %Finding the value of derivative at steady state

C(1,1,i) = double(subs(diff(dCdt,Ca),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(1,2,i) = double(subs(diff(dCdt,T),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(1,3,i) = double(subs(diff(dCdt,Tj),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(2,1,i) = double(subs(diff(dTdt,Ca),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(2,2,i) = double(subs(diff(dTdt,T),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(2,3,i) = double(subs(diff(dTdt,Tj),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(3,1,i) = double(subs(diff(dTjdt,Ca),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(3,2,i) = double(subs(diff(dTjdt,T),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

C(3,3,i) = double(subs(diff(dTjdt,Tj),[Ca T Tj],[Cas(i) Ts(i) Tjs(i)]));

end

Eig=zeros([3,3]);

for i=1:3

Eig(i,:)=eig(C(:,:,i));

end

Eig

for i=1:3 %Checking for stability of steady states

if real(Eig(i,:))<0

fprintf("The steady state [%f,%f,%f] is stable\n",A(i,1),A(i,2),A(i,3));

else

fprintf("The steady state [%f,%f,%f] is not stable\n",A(i,1),A(i,2),A(i,3));

end

end

function f=cstrsteady(x)

R=1.987;

F=1;

Vj=0.25;

V=1;

k0=36\*1e6;

H=-6500;

E=12000;

Rho\_Cp=500;

Tf=298;

CAf=10;

UA=150;

Tj0=298;

Rhoj\_Cj=600;

Fj=1.25;

Cs=x(1);

Ts=x(2);

Tjs=x(3);

r = k0 \* exp(-E/(R\*Ts))\*Cs;

f(1)=F\*CAf-F\*Cs-r\*V;

f(2)=Rho\_Cp\*F\*(Tf-Ts)-H\*V\*r-UA\*(Ts-Tjs);

f(3)=Rhoj\_Cj\*Fj\*(Tj0-Tjs)+UA\*(Ts-Tjs);

end

**Results**

|  |  |  |
| --- | --- | --- |
| **Steady State (Ca, T, Tj)** | **Eigenvalues of Linearized System** | **Stable/Unstable** |
| (1.4094, 387.3423, 312.8904) | (-1.9597+0.74i, -1.9597-0.74i, -59805) | Asymptotically Stable |
| (6.1650, 337.8844, 304.6474) | (0.6650, -0.911, -6.0387) | Unstable |
| (8.9686, 308.7270, 299.7878) | (-0.5765, -0.9355, -6.0534) | Asymptotically Stable |

**Conclusion**

This stability test is to distinguish between the numerous critical points that exist for a system. According to our tests performed above, the 2nd steady state obtained came out to be unstable with the first eigen value being greater than the other 2 and the other 2 being < 0. This implies that such a critical point is geometrically a saddle point. Such tests can be used to classify each critical point and understand their geometrical characteristics too. The last steady state has all eigen values < 0 and therefore, gives an asymptotically stable system and so does the first steady state. The only difference being that the nature of the third steady state’s critical points is that of a sink whereas, the first one is an inward spiral.