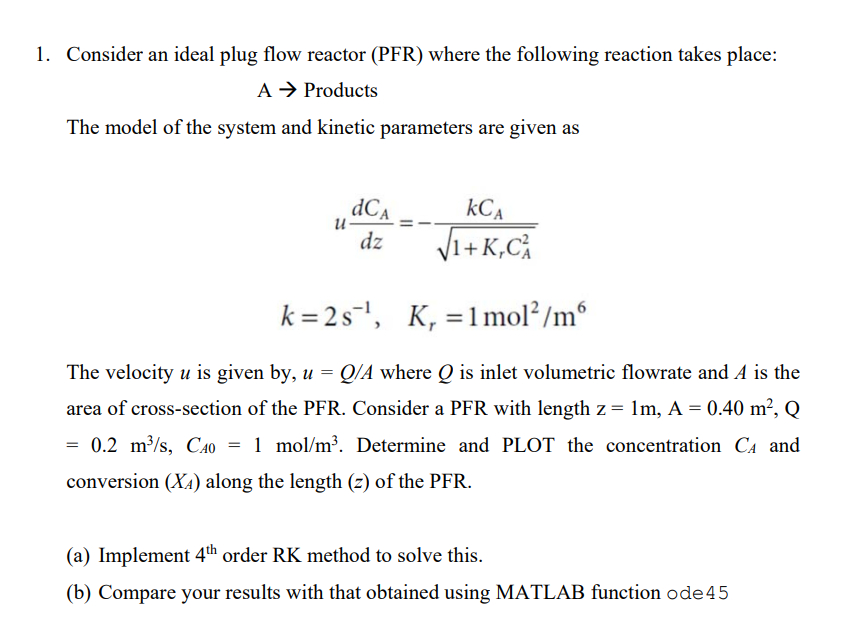
**CH49019: CAPE Laboratory Autumn 2021**

**Assignment-1**

**S. Varshith Reddy**

**18CH10053**

****

**Code:**

%4th order Runge-Kutta Method

Q=0.2;% Parameters

A=0.4;

u=Q/A;

Ca0=1;

k=2;

kr=1;

z=1;

f=@(Ca\_RK) -(k/u)\*Ca\_RK/(sqrt(1+kr\*Ca\_RK\*Ca\_RK)); % Function to calculate the dCdz

Xa\_RK=@(Ca\_RK) (Ca0-Ca\_RK)/Ca0; % Function to calculate conversion

h=0.1;%Step Size

Ca\_RK=zeros([z/h + 1,1]);

Ca\_RK(1)=1;% Concentration at z=0

for i=1:size(Ca\_RK,1)-1 %4th order RK method

k1=h\*f(Ca\_RK(i));

k2=h\*f(Ca\_RK(i)+0.5\*k1);

k3=h\*f(Ca\_RK(i)+0.5\*k2);

k4=h\*f(Ca\_RK(i)+k3);

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

end

Xa\_RK=Xa\_RK(Ca\_RK);%Calculating conversion along the length

plot([0:h:z],Ca\_RK);

xlabel("z");ylabel("Concentration");

title("Concentration along the length of the PFR using 4th order RK method");

plot([0:h:z],Xa\_RK);

xlabel("z");ylabel("Conversion");

title("Conversion along the length of the PFR using 4th order RK method");

%ode45 solver

f=@(z,Ca\_ode45) -(k/u)\*Ca\_ode45/(sqrt(1+kr\*Ca\_ode45\*Ca\_ode45));

Xa\_ode45=@(Ca\_ode45) (Ca0-Ca\_RK)/Ca0;

[Z Ca\_ode45]=ode45(f,[0:h:z],Ca0);

Xa\_ode45=Xa\_ode45(Ca\_ode45);

plot([0:h:z],Ca\_ode45);

xlabel("z");ylabel("Concentration");

title("Concentration along the length of the PFR using ode45 solver");

plot([0:h:z],Xa\_ode45);

xlabel("z");ylabel("Conversion");

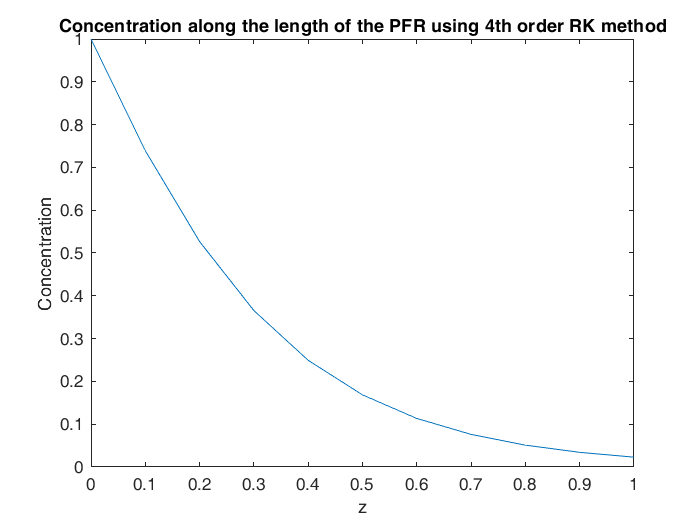
title("Conversion along the length of the PFR using ode45 solver");

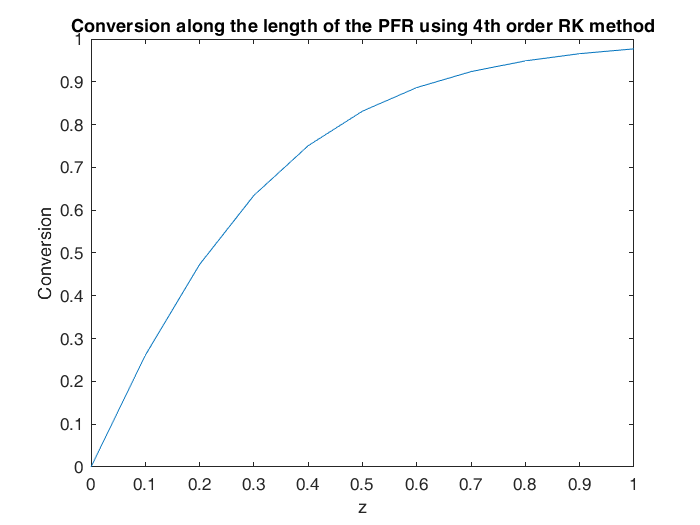
**Results:**

For h=0.1

4th order Runge Kutta Method:

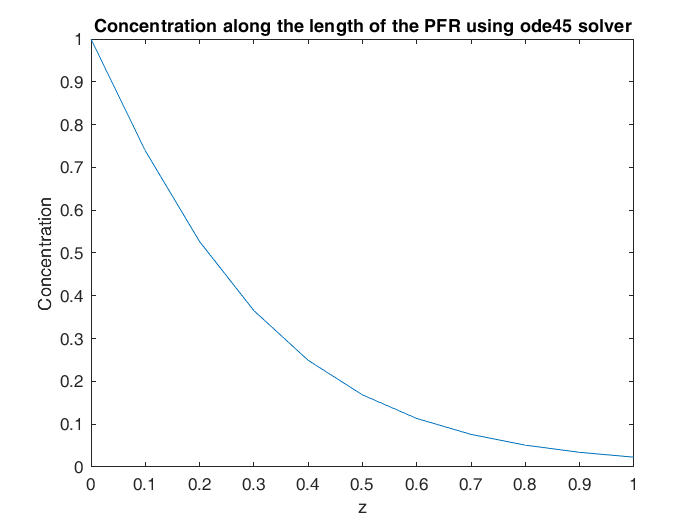
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ca** | 1 | 0.7389 | 0.5267 | 0.3654 | 0.2493 | 0.1685 | 0.1134 | 0.0762 | 0.0511 | 0.0343 | 0.0230 |
| **XA** | 0 | 0.2611 | 0.4733 | 0.6346 | 0.7507 | 0.8315 | 0.8866 | 0.9238 | 0.9489 | 0.9657 | 0.9770 |
| **z** | 0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |

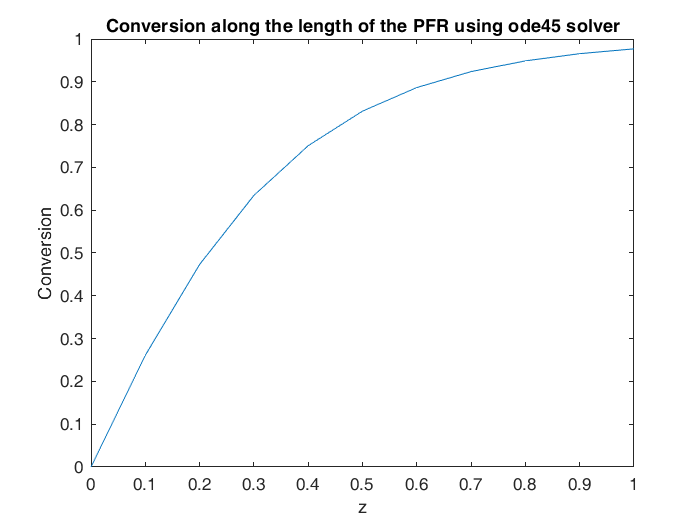




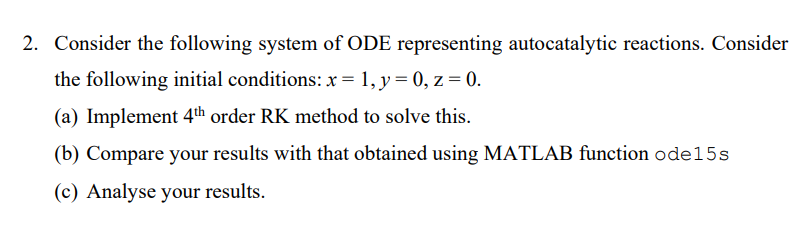
ode45 solver:

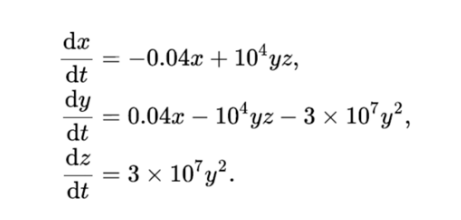
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ca** | 1 | 0.7389 | 0.5267 | 0.3654 | 0.2492 | 0.1685 | 0.1134 | 0.0761 | 0.0511 | 0.0342 | 0.0230 |
| **XA** | 0 | 0.2611 | 0.4733 | 0.6346 | 0.7507 | 0.8315 | 0.8866 | 0.9238 | 0.9489 | 0.9657 | 0.9770 |
| **z** | 0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |





**Conclusions:**

****

****

**Code:**

%4th order Runge-Kutta Method

clear

format long

for h=[0.1 0.01 0.001 0.0001 0.00001]%Loop to check for minimum order of the step length for proper solution

y\_RK=zeros([1/h + 1,3]); %Initializing the matrix

y\_RK(1,:)=[1 0 0]; %Initial Condition

for i=1:(1/h) % 4th order RK method

a=h\*auto\_cat\_reaction\_Rk(y\_RK(i,:));

b=h\*auto\_cat\_reaction\_Rk(y\_RK(i,:)+(0.5\*a));

c=h\*auto\_cat\_reaction\_Rk(y\_RK(i,:)+(0.5\*b));

d=h\*auto\_cat\_reaction\_Rk(y\_RK(i,:)+(c));

a1=(a+2.\*b+2.\*c+d);

y\_RK(i+1,:)=y\_RK(i,:)+a1/6;

if ~(isnan(y\_RK(i+1,:))) % checks if any of the values are NAN which means step size should be further refined

continue;

else

break;

end

end

if i==1000 %Breaks if no NAN values are found in the matrix

break

end

end

%ode15s solver

[t\_ode45, y\_ode15s]=ode15s(@auto\_cat\_reaction,[0:0.1:1],[1 0 0]);%ode solver

x=y\_ode15s(:,1);%values of x from t=0 to t=1

y=y\_ode15s(:,2);%values of y from t=0 to t=1

z=y\_ode15s(:,3);%values of z from t=0 to t=1

plot([0:h:1],y\_RK,[0:0.1:1],y\_ode15s);legend("RK-x","RK-y","RK-z","ode15s-x","ode15s-y","ode15s-z");

plot([0:h:1],y\_RK(:,1),[0:0.1:1],x);xlabel("t");ylabel("x");legend("RK","ode15s");

plot([0:h:1],y\_RK(:,2),[0:0.1:1],y);xlabel("t");ylabel("y");legend("RK","ode15s");

plot([0:h:1],y\_RK(:,3),[0:0.1:1],z);xlabel("t");ylabel("z");legend("RK","ode15s");

function dydt=auto\_cat\_reaction(~,y)

dydt=[-0.04\*y(1)+10000\*y(2)\*y(3);

0.04\*y(1)-10000\*y(2)\*y(3)-3e7\*y(2)\*y(2);

3e7\*y(2)\*y(2);];

end

function dydt=auto\_cat\_reaction\_Rk(y)

dydt=[-0.04\*y(1)+10000\*y(2)\*y(3) 0.04\*y(1)-10000\*y(2)\*y(3)-3e7\*y(2)\*y(2) 3e7\*y(2)\*y(2);];

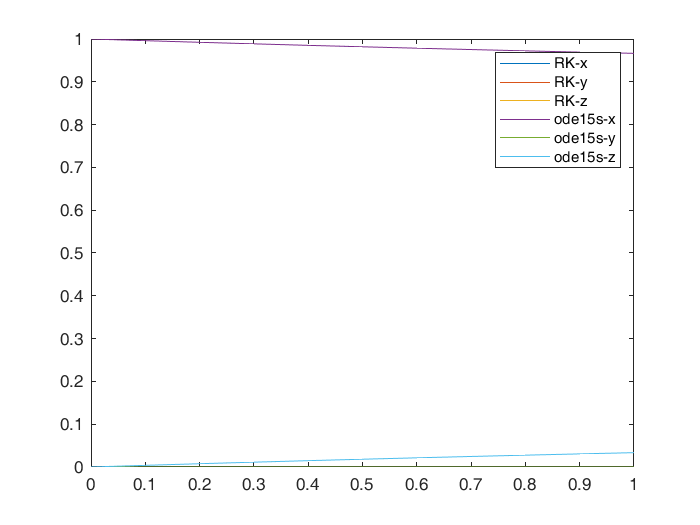
end

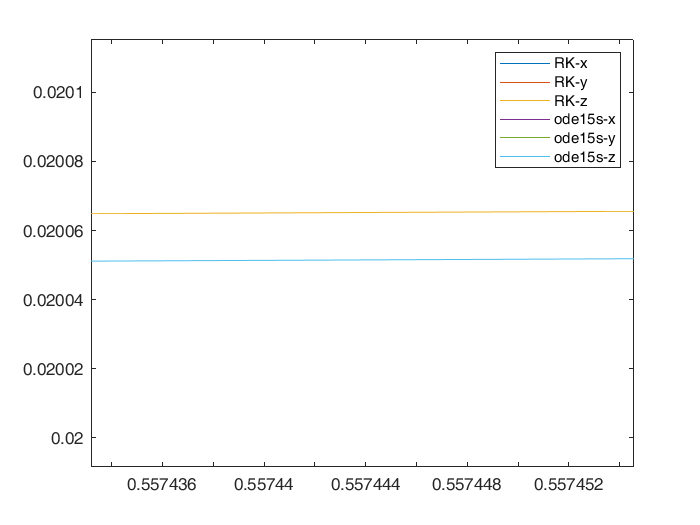
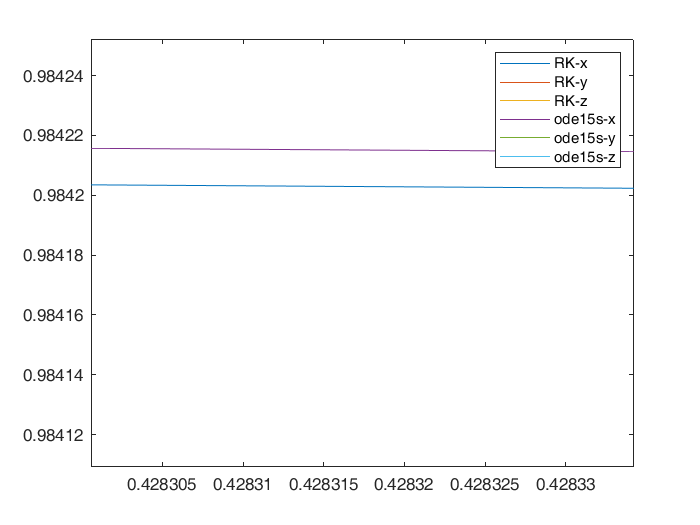
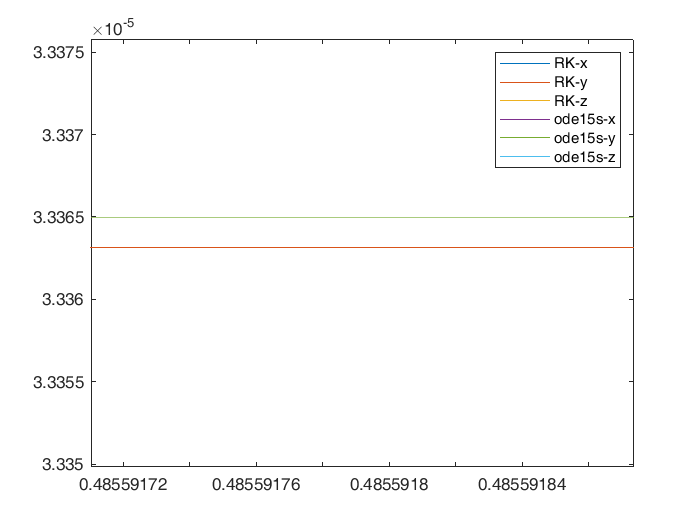
**4th order Runge Kutta Method:**

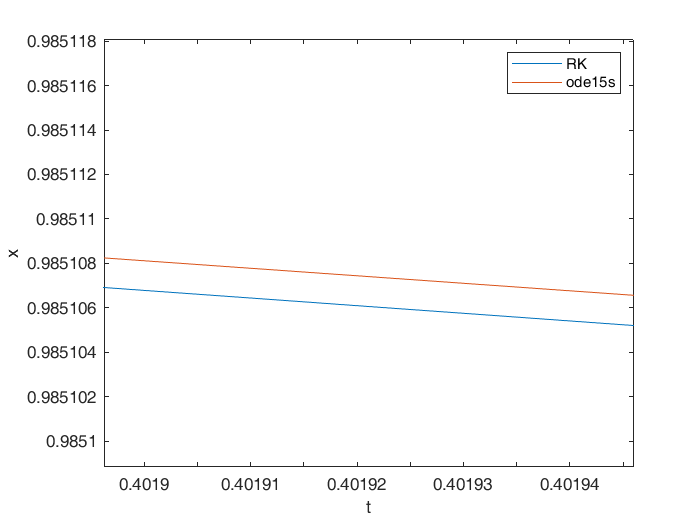
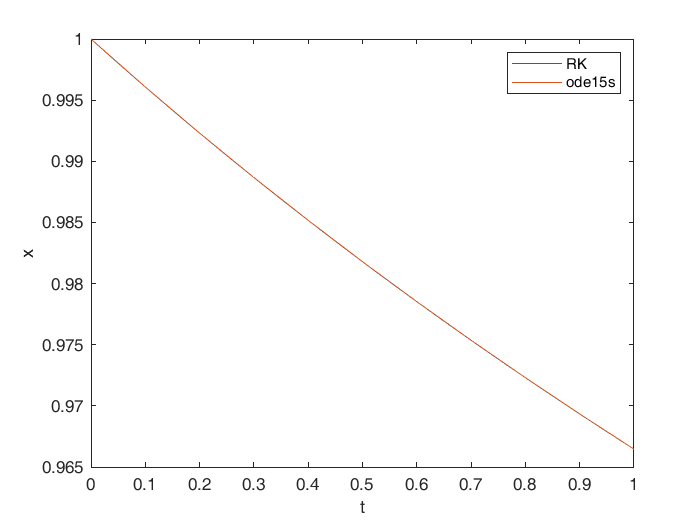
|  |  |  |  |
| --- | --- | --- | --- |
| **t** | **x** | **y** | **z** |
| 0 | 1 | 0 | 0 |
| 0.1 | 0.996077747 | 3.58044E-05 | 0.003886449 |
| 0.2 | 0.992305945 | 3.5123E-05 | 0.007658932 |
| 0.3 | 0.988673939 | 3.44772E-05 | 0.011291584 |
| 0.4 | 0.985172113 | 3.3864E-05 | 0.014794023 |
| 0.5 | 0.981791773 | 3.32809E-05 | 0.018174946 |
| 0.6 | 0.978525033 | 3.27258E-05 | 0.021442241 |
| 0.7 | 0.975364721 | 3.21965E-05 | 0.024603082 |
| 0.8 | 0.9723043 | 3.16912E-05 | 0.027664008 |
| 0.9 | 0.969337796 | 3.12083E-05 | 0.030630995 |
| 1 | 0.966459737 | 3.07463E-05 | 0.033509517 |

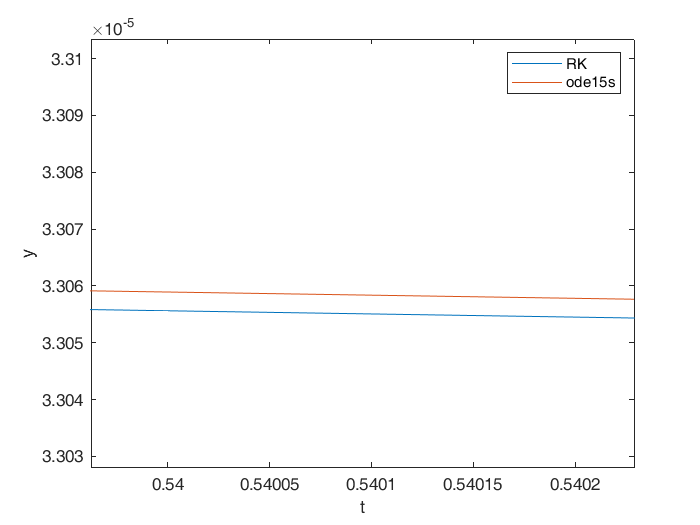
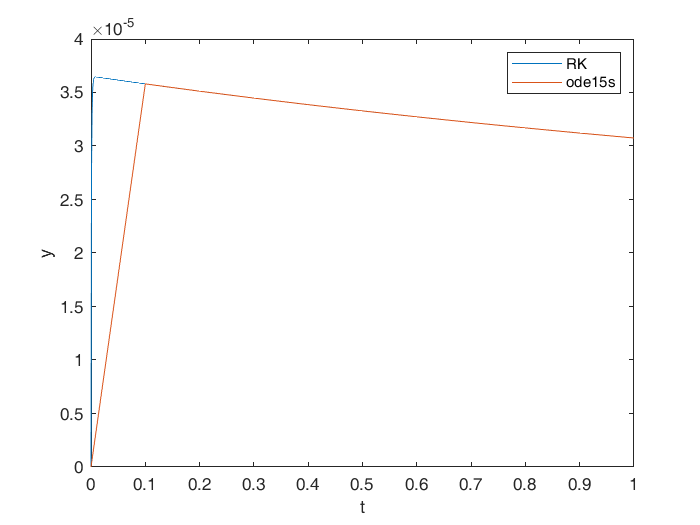
**Ode15s method:**

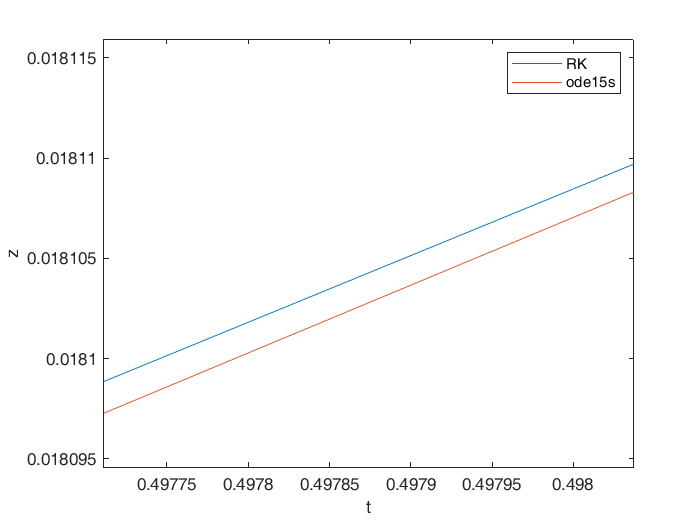
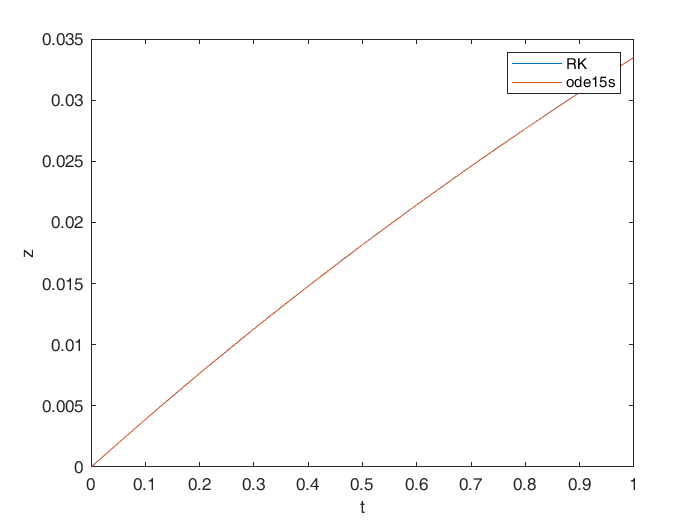
|  |  |  |  |
| --- | --- | --- | --- |
| **t** | **x** | **y** | **z** |
| 0 | 1 | 0 | 0 |
| 0.1 | 0.996077649 | 3.58042E-05 | 0.003886546 |
| 0.2 | 0.992305933 | 3.51229E-05 | 0.007658944 |
| 0.3 | 0.988674063 | 3.44773E-05 | 0.011291459 |
| 0.4 | 0.985172341 | 3.3864E-05 | 0.014793795 |
| 0.5 | 0.981792055 | 3.3281E-05 | 0.018174664 |
| 0.6 | 0.978525354 | 3.27258E-05 | 0.02144192 |
| 0.7 | 0.975365077 | 3.21966E-05 | 0.024602726 |
| 0.8 | 0.972304689 | 3.16913E-05 | 0.02766362 |
| 0.9 | 0.969338212 | 3.12084E-05 | 0.03063058 |
| 1 | 0.966460178 | 3.07463E-05 | 0.033509075 |



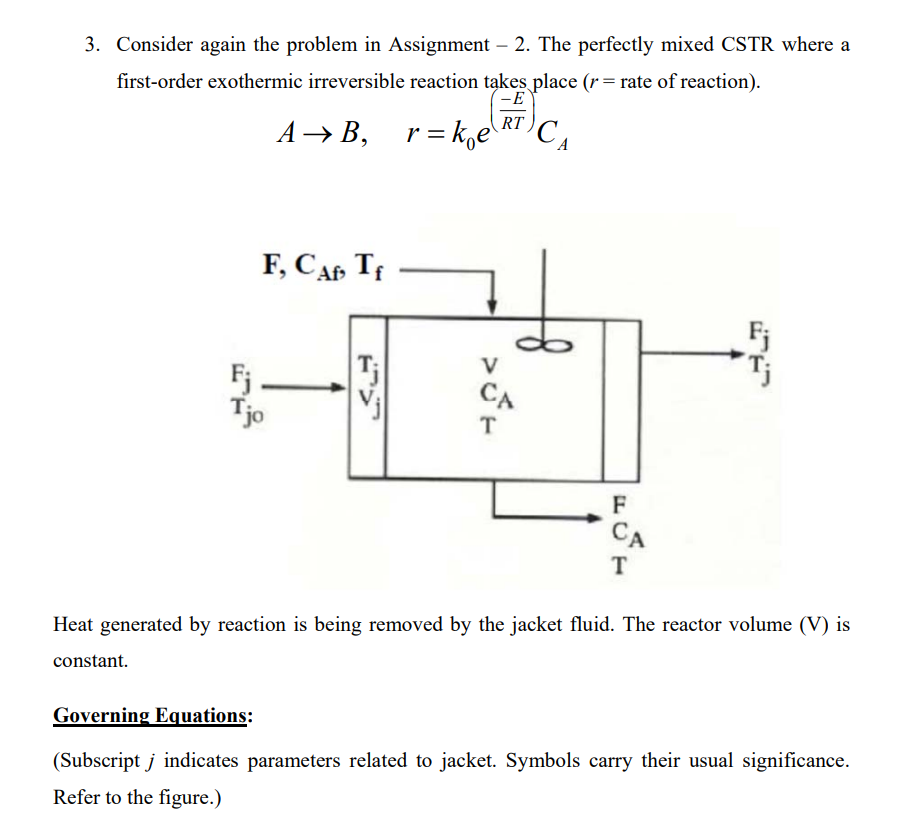


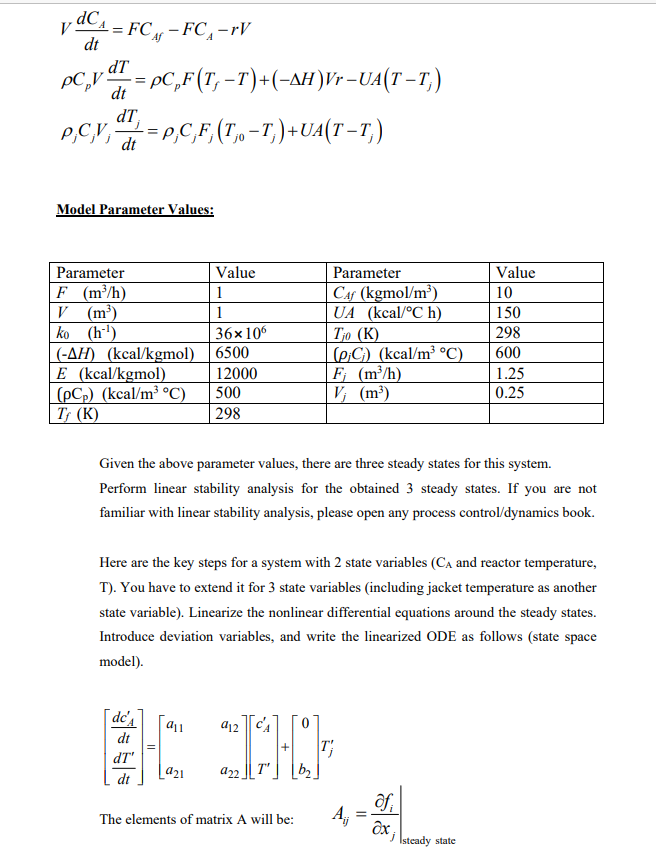


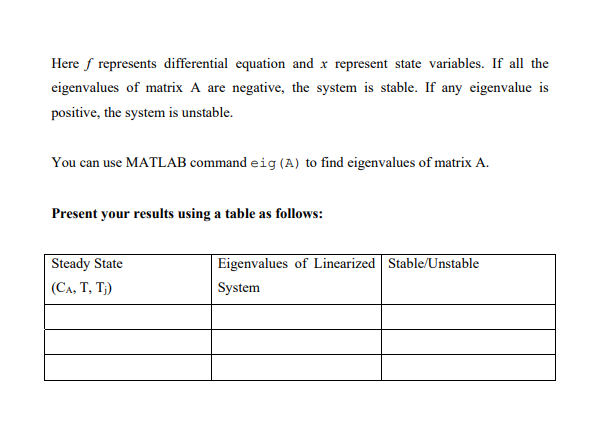




**Conclusions:**

****

****

****

**Code:**

%Stability Analysis

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]); %Initializing a matrix to store solutions for each inital guess

for i=1:20 %Solving for steady states

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

z=[];

z(1)=0+rand\*(10); %Taking random values as inital guess for all variables

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

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

a(i,1:3)=fsolve(@cstrsteady,z,options); %Usinf fsolve to solve 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 uique 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

%Creating 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);

A=zeros([3 3 3]);

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

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

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

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

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

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

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

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

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

A(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(A(:,:,i));

end

Eig

for i=1:3 %Checking if the steady states are stable or not

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)

%Parameters

R=1.987;

F=1;

Vj=0.25;

V=1;

k0=36\*1e6;

H=-6500;

E=12000;

rhoCp=500;

Tf=298;

Caf=10;

UA=150;

Tj0=298;

rhojCj=600;

Fj=1.25;

Cs=x(1);

Ts=x(2);

Tjs=x(3);

r = k0 \* exp(-E/(R\*Ts))\*Cs;%Rate of the reactions

f(1)=F\*Caf-F\*Cs-r\*V;%Declaring the non linear equations

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

f(3)=rhojCj\*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) | Stable |
| (6.1650, 337.8844, 304.6474) | (0.6650, -0.911, -6.0387) | Not Stable |
| (8.9686, 308.7270, 299.7878) | (-0.5765, -0.9355, -6.0534) | Stable |

**Conclusions:**