# Group 4

## Contributions

| | |
|---|---|
| Varun Singh | Writing code for vapor-liquid equilibrium curve.<br>Plotting in McCabe-Thiele Method.<br>Minimum No. of trays code. |
| Vrahant Nagoria | Providing various parameters like Antoine coefficients, etc.<br>Writing code for Rectifying and Stripping Section.<br>Minimum Reflux Ratio Code. |

## Objective

The goal of this project is to study about the McCabe thiele method and apply the methods learnt to a system composed of ethanol-water mixture and simulate it using the DWSIM software.
We are also required to plot the vapor liquid equilibrium curve for the same mixture.

## Codes and Algorithm

### For Vapor-Liquid equilibrium

1. Input Parameters like antoine coefficients and Van laar constants.
2. Input pressure.
3. Calculate the activity coefficient.
4. Solve the modified Raoult's law equation using fsolve
5. Plot the equilibrium curve

```
%% binary system ethanol water
AntCon  = [8.11220 1592.864 226.184;7.96681  1668.210 228.000];
isPlot = true;
vLconst = [1.6798 0.9227];
P       = input('enter pressure'); %pressure [Pa]
hold on;
set(line([0  1],[0  1]),'Color',[0 1 0]);
[xA,yA] = txy_diagram (AntCon,vLconst,P,isPlot);
function [xA,yA] = txy_diagram (AntCon,vLconst,P,isPlot)
global yA_arr
A12 = vLconst(1);
A21 = vLconst(2);
%mole fractions
xA = 0:0.01:1;
xB = 1-xA;
%activity coefficients
gA = [0,
exp(A12*(A21.*(1-xA(2:end-1))./(A12*xA(2:end-1)+A21*(1-xA(2:end-1)))).^2),
1];
```

```
gB = [1,
exp(A21*(A12.*(xA(2:end-1))./(A21*(1-xA(2:end-1))+A12*(xA(2:end-1)))).^2),
0];
xA = xA(:);
xB = xB(:);
gA = gA(:);
gB = gB(:);
%boiling point of pure compound
tA = AntCon(1,2)/(AntCon(1,1)-log10(P))-AntCon(1,3);
tB = AntCon(2,2)/(AntCon(2,1)-log10(P))-AntCon(2,3);
x0   = linspace(tB,tA,length(xA));
spec = optimset('Display','notify');
Temp = fsolve(@Fun,x0,spec,AntCon,xA,xB,gA,gB,P);
yA   = compute_yA(Temp,AntCon,xA,gA,P);
yA_arr = yA;
if isPlot
   plot(xA,yA)
   xlabel('liquid mole fraction of ethanol')
   ylabel('vapor mole fraction of ethanol')
   title('x,y diagram')
   xlim([0 1])
end
%fprintf("%f\n",Temp);
end
function F = Fun (t,AC,xA,xB,gA,gB,P)
%nonlinear system of equations to solve
pA0 = antoineEq(AC(1,:),t);
pB0 = antoineEq(AC(2,:),t);
pA0 = pA0(:);
pB0 = pB0(:);
fA = 1;
fB = 1;
pA = pA0.*xA.*gA./fA;
pB = pB0.*xB.*gB./fB;
F  = pA+pB-P;
end
function F = compute_yA (t,AC,xA,gA,P)
% computes yA for xA
pA0 = antoineEq(AC(1,:),t);
pA0 = pA0(:);
F   = pA0/P.*xA.*gA;
end
function F = antoineEq (ACons,t)
%Antoine equation
t = t(:);
F = 10.^(ACons(1)-(ACons(2)./(ACons(3)+t)));
end
```

## For McCabe-Thiele Method

1. Input the Distillate, bottoms, feed and Reflux ratio.
2. Solve for the Rectifying Section.
3. Solve for the Stripping Section.

4. Find intersection of feed and operating lines.

```
global yA_arr

xe = 0:0.01:1;
ye = yA_arr;
% up, Bottom and Feed mole fractions are equal to 85 % mol, 5 % mol
% and 30 % mol respectively.
xu = input('up mole fraction');
xb = input('bottom mole fraction');
zf = input('feed mole fraction');
% xu=0.85;
% xb=0.05;
% zf=0.3;
% Reflux ratio is equal to 1.51435
R = input('Reflux ratio');
% R=1.51435;
% Feed is a Two-phase mixture with feed quality equal to 0.85
q = input('feed quality');
% q=0.85;
% Computing the intersection of feed line and operating lines
yi=(zf+xu*q/R)/(1+q/R);
xi=(-(q-1)*(1-R/(R+1))*xu-zf)/((q-1)*R/(R+1)-q);
hold on;
axis([0 1 0 1]);
% plotting operating and feed lines and equilibrium curve
plot(xe,ye,'r');
set(line([0  1],[0  1]),'Color',[0 1 0]);
set(line([xu xi],[xu yi]),'Color',[1 0 1]);
set(line([zf xi],[zf yi]),'Color',[1 0 1]);
set(line([xb xi],[xb yi]),'Color',[1 0 1]);
%Stripping Section
c=0;
i=1;
xp(1)=xb;
yp(1)=xb;
while(xp(i)<xi)

   pp = spline(xe,ye);
   yp(i+1) = ppval(pp,xp(i));
   set(line([xp(i) xp(i)],[yp(i) yp(i+1)]),'color',[0 0 1]);
   c=c+1;
   xp(i+1)=(yp(i+1)-xb)*((xi-xb)/(yi-xb))+xb;
   if(xp(i+1)<xi)
       set(line([xp(i) xp(i+1)],[yp(i+1) yp(i+1)]),'color',[0 0 1]);
       c=c+1;
   end
           i=i+1;

end
%Rectifying Section
xp(i)=(yp(i)-xu)*((xi-xu)/(yi-xu))+xu;
set(line([xp(i-1) xp(i)],[yp(i) yp(i)]),'color',[0 0 1]);
```

```
while(xp(i)>xi && xp(i)<xu)

   pp = spline(xe,ye);
   yp(i+1) = ppval(pp,xp(i));
   set(line([xp(i) xp(i)],[yp(i) yp(i+1)]),'color',[0 0 1]);
   c=c+1;
   xp(i+1)=(yp(i+1)-xu)*((xi-xu)/(yi-xu))+xu;
   set(line([xp(i) xp(i+1)],[yp(i+1) yp(i+1)]),'color',[0 0 1]);

   i=i+1;

end
hold off;
disp("Number of trays")
disp(c-1)
```

## Minimum Reflux Ratio

```
global yA_arr
xe = 0:0.01:1;
ye = yA_arr;
% up, Bottom and Feed mole fractions are equal to 85 % mol, 5 % mol
% and 30 % mol respectively.
xu = input('up mole fraction');
xb = input('bottom mole fraction');
zf = input('feed mole fraction');
R = input('Reflux ratio');
% Reflux ratio is equal to 1.51435
% Feed is a Two-phase mixture with feed quality equal to 0.85
q = input('feed quality');
% Computing the intersection of feed line and operating lines
pp = spline(xe,ye);
yi = ppval(pp,zf);
hold on;
axis([0 1 0 1]);
% plotting operating and feed lines and equilibrium curve
plot(xe,ye,'r');
set(line([0  1],[0  1]),'Color',[0 1 0]);
set(line([xu zf],[xu yi]),'Color',[1 0 1]);
set(line([zf zf],[zf yi]),'Color',[1 0 1]);
% Slope for min. reflux ratio
slope_min_rr = ((xu-yi)/(xu-zf));
disp("minimum reflux ratio")
disp(slope_min_rr/(1-slope_min_rr))
```

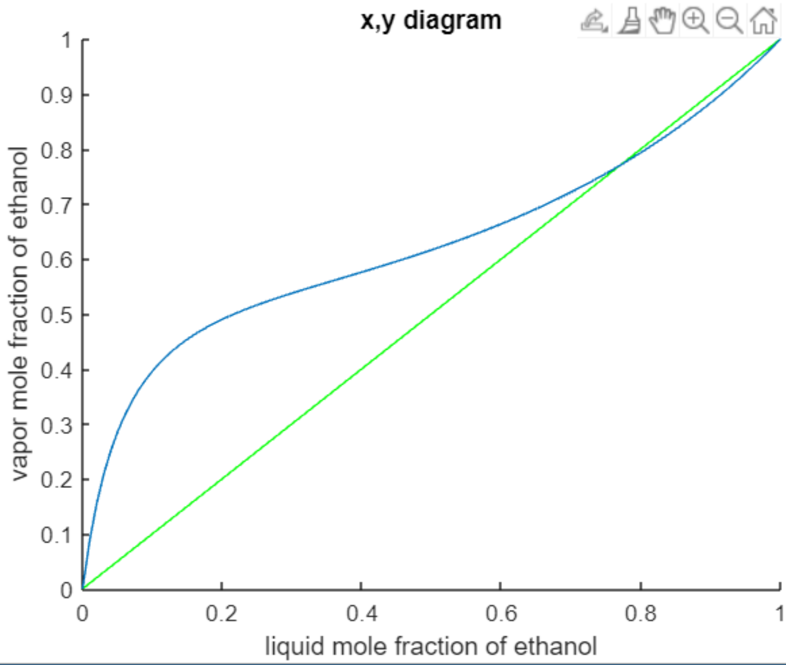
## For Minimum number of Trays

```
global yA_arr
% Equilibrium curve computation
xe = 0:0.01:1;
ye = yA_arr;
% up, Bottom mole fractions are equal to 85 % mol, 5 % mol.
xu = input('up mole fraction');
xb = input('bottom mole fraction');
```

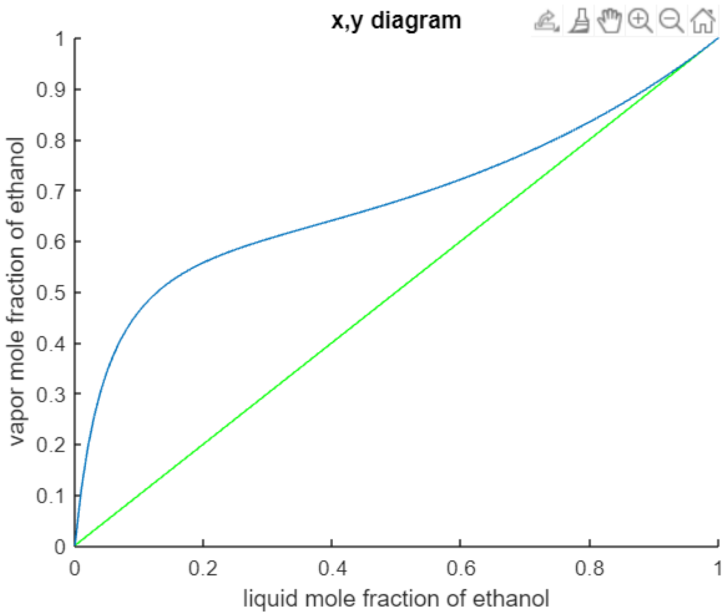
```
% Computing the intersection of feed line and operating lines
hold on;
axis([0 1 0 1]);
% plotting equilibrium curve
plot(xe,ye,'r');
set(line([0  1],[0  1]),'Color',[0 1 0]);
% Plotting trays (min. trays) required
c = 0;
i = 1;
xp(1) = xb;
yp(1) = xb;
while(xp(i)<xu)

   pp = spline(xe,ye);
   yp(i+1) = ppval(pp,xp(i));
   set(line([xp(i) xp(i)],[yp(i) yp(i+1)]),'color',[0 0 1]);
   c = c+1;
   xp(i+1)=yp(i+1);
   set(line([xp(i) xp(i+1)],[yp(i+1) yp(i+1)]),'color',[0 0 1]);

   i = i + 1;

end
hold off;
disp("minimum number of trays")
disp(c)
```

## Important results

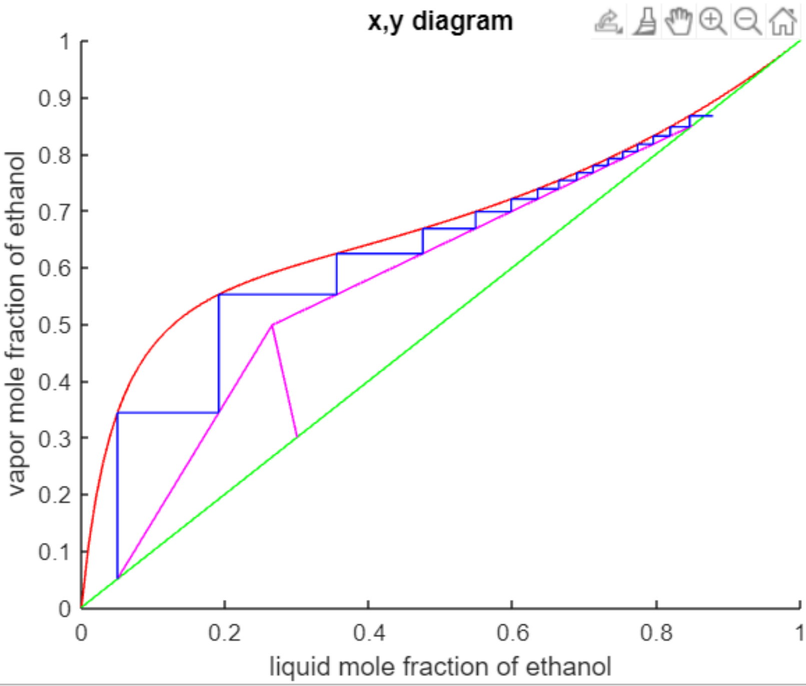
For High Pressures



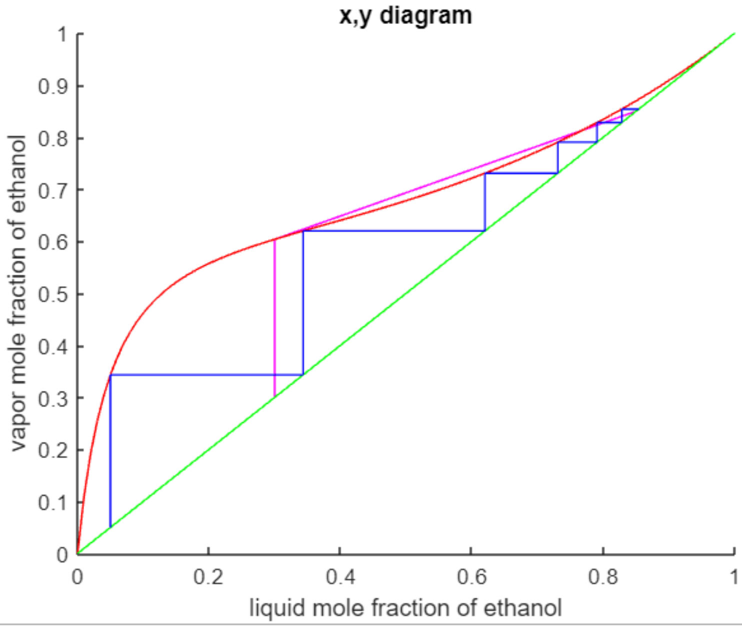

For Low Pressures




McCabe-Thiele Method



For Minimum reflux ratio and minimum number of trays




For P = 50Pa
Number of trays = 16
Minimum number of trays = 6
Minimum reflux ratio = 0.8108