Chapter 10

Main Objectives:

(1) Sketch P-xy and T-xy phase envelopes and label bubble, dew, flash and pure component conditions.
(2) Interpret problem statements to choose correct VLE routine to execute.
(3) Execute Bubble, Dew, Flash routines.
(4) Apply Raoult's Law to emission calculations.

General Criteria

$$\mu_i^V = \mu_i^L \quad \text{or} \quad \hat{f}_i^V = \hat{f}_i^L \qquad K_i = y_i/x_i$$

10.3 Raoult's Law $\hat{f}_i^V = y_i P, \quad \hat{f}_i^L = x_i P_i^{sat}$

$$y_i P = x_i P_i^{sat} \qquad K_i = y_i/x_i = P_i^{sat}/P$$

Shortcut K-ratio

$$K_i \approx \frac{P_{c,i} 10^\wedge (\frac{7}{3}(1+\omega)(1-1/T_{r,i}))}{P}$$

Types of Calcs (See Table 10.1)

	know	find	criteria
BP	z=x,T	y,P	$\Sigma y_i = 1 = \Sigma K_i x_i$
DP	z=y,T	x,P	$\Sigma x_i = 1 = \Sigma y_i/K_i$
BT	z=x,P	y,T	$\Sigma y_i = 1 = \Sigma K_i x_i$
DT	z=y,P	x,T	$\Sigma x_i = 1 = \Sigma y_i/K_i$
Flash	z, P, T	x,y	$\Sigma x_i - \Sigma y_i = \Sigma [z_i(1-K_i)/(K_i + (L/F)(1-K_i))] = 0$

BP

$1 = \Sigma K_i x_i$

DP

$1 = \Sigma y_i/K_i$

BT

$1 = \Sigma K_i x_i$

DT

$1 = \Sigma y_i/K_i$

Flash

$\Sigma x_i - \Sigma y_i = 0$

10.5 Emission Modeling (EPA)

terms: headspace

VOC

purge, sweep

noncondensibles

<u>Filling, Charging</u>

$n^i_{head} = P\underline{V}^i_{head}/RT$

$n^f_{head} = P\underline{V}^f_{head}/RT$

$\Delta n_{head} =$

Δn_i

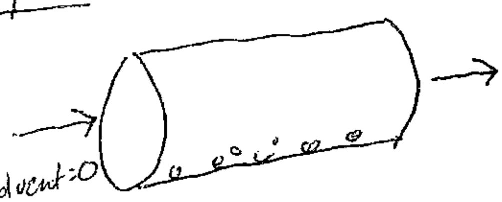
Purge

material balance

$$\frac{dy_i}{y_i} = \frac{-\dot{V}_{sweep}}{V_{tank}}\,dt$$

$$y_i = y_i^i \exp\left[-\frac{\dot{V}_{sweep}\,t}{V_{tank}}\right] \quad (10.3)$$

Dryer



$y_{solvent} = 0$

$y_{nc} = 1$

RL- $y_{solvent} P = x_{solvent} P_{solvent}^{sat}$

$y_{nc} < 1 \qquad y_{nc} = 1 - \sum_{i\,\mathrm{voc}} y_i$

Heating

$$\Delta n_i = \frac{\Delta n_{nc}}{2}\left(\left(\frac{y_i}{y_{nc}}\right)^i + \left(\frac{y_i}{y_{nc}}\right)^f\right)$$

Δn_{nc} given by 10.6

Depressurization

Tank Storage

```
function RaoultBP
% Raoult's law shortcut calculation of bubble pressure.
% vectors for critical properties
% all values set in code, none from workspace.

% The bubble point is BP = x.1*P.1.sat + x.2*P.2.sat + ....
% The value is calculated using vectors multiplication.

% set output - eliminate spaces
format compact

%*******Input section, can be edited**********
%Constants
names ={'ethane', 'propane', 'nbutane', 'ibutane'}
Tc = [305.4 369.8 425.2 408.1]; % critical temps K
Pc = [4.88 4.249 3.797 3.648]; % critical Ps (MPa)
w = [0.099 0.152 0.193 0.177]; % accentric factor

Punits = 'MPa';
Tunits = 'K';

% temperature at which to calculate
T = 320;
% set the composition to use for calculation
x = [0.1 0.9 0. 0.]
%**********End of input section***************

disp(sprintf('T(%s) = %g', Tunits, T))

% create a vector of reduced temperatures and pressures
Tr = T./Tc;

% Create a vector of vapor pressures using shortcut vapor pressure method.
% Note that dot operators are used for element-by-element processing.
Psat = Pc .* (10.^(7/3.*(1+w).*(1-1./Tr)));

% Calculate the bubble pressure.
% Because Psat is a line vector, create a column vector to
% use matrix multiplication. The ' indicates a transpose.
BP = x * Psat';
disp(sprintf('Bubble P(%s) = %g', Punits, BP))

% reset standard output format
format loose
end
```

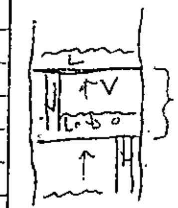
```
function RaoultTxy
% Raoult's Law Txy using Antoine Psat
% vectors for Antoine Coefficients
% all values set in code, none from workspace.

% The routine uses the same methods as RaoultBT, and the background
% of the calculation methods and variables are largely described in
% that function.

% This routine uses
%   element-by-element vector math
%   1-D solver
%   plotting multiple data sets and labelling chart

% Developed by Carl Lira for use with
% Introductory Chemical Engineering Thermodynamics, by
% J. Richard Elliott and Carl T. Lira, Prentice-Hall, 1999.
% See http://www.egr.msu.edu/~lira/thermtxt.htm for links to
% download an original. To be redistributed only in un-modified form,
% and only for educational use. The program developers have no liability
% for the use of the program or results. Copyright 2008, Carl T. Lira.
addpath(genpath('../Psat')); %add path to Antoine constants

%**************Beginning of input section, editable
% id values from rows of ../Psat/AntoineTable.mat
id = [27 31];
%***************End of input section*****************

[names A B C] = AntoineGet(id);
names

% Specify units for Antoine Equation constants.
Tunits = 'C';
Punits = 'mmHg';

%variables
Tguess = 50; %initial guess
% set P basis for bubble T calculation
P = 1000; % same units as Antoine's equation

% Create composition vectors that span the composition range.
x1 = 0:0.1:1;
x2 = 1 - x1;

% Create rows of composition pairs in X by tranposing the
% composition vectors.
X = [x1' x2'];
% count number of rows
Nrows = length(x1);

% find the bubble pressure for each row of X (each pair of compositions)
```

```
for i = 1:Nrows

    options=optimset('Display','iter');   % Option to display output
    [T,fval,status] = fzero(@calcObj,Tguess);  % Call optimizer

    % store the results for later plotting
    Tstore(i) = T;
    % y values are available because a nested function is used.
    Ystore(i,:) = y;

end %for X loop

%print table of results
%'sprintf' is nested in 'disp' to create tighter output format
disp(sprintf('1 = %s, 2 = %s',char(names(1)),char(names(2))))
disp(sprintf('P = %g %s', P, Punits))
disp(sprintf('x1\ty1\tT(%s)',Tunits))
for i = 1:Nrows
    disp(sprintf('%g\t%g\t%g',X(i,1),Ystore(i,1),Tstore(i)))
end

%plot results
% plot x vs. T with a default solid line
plot(X(:,1),Tstore)
% set plot limits for xaxis
xlim([0 1])
hold on % add to plot
% plot y vs. T on same plot with dashed line
plot(Ystore(:,1),Tstore,'--')
% add chart labels
xlabel(['x - y ' char(names(1))])
ylabel(['T(' Tunits ')'])
title([char(names(1)) ' + ' char(names(2)) ' at P= ' num2str(P) ' ' Punits])
hold off
% end of main function

% ------------------------------------------------------------
%beginning of nested objective function(shares vars with main function)
    function obj = calcObj(TCelsius)

        %properties
        % vector of vapor pressures.
        % Use element-by-element vector math
        Psat = 10.^(A - B./(TCelsius+C));  %mm Hg
        % vector of K ratios
        K = Psat/P;
        % use the active row of X to calculate the corresponding y's
        y = X(i,:).*K;
        % objective function
        obj = sum(y) - 1.;
    end
```

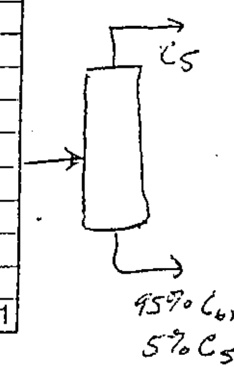
```
% ------------------------------------------------------------

end

% ver 1.1 Added use of ../Psat/AntoineTable 1/19/10 CTL
```

ChE321 - Class Example

Output from process simulator (Peng-Robinson EOS)						Conditions: Feed to the column is equimolar at saturation T.					
10 stages plus reboiler and total condenser						10 stages. Reflux = 5. Distillate flow 0.3*Feed.					
		T(C)		liquid mole fractions				vapor mole fractions			
	condenser	36.31438		0.996203	0.003782	1.50E-05		0.998733	0.001265	1.74E-06	
	1	36.47099		0.988634	0.011238	0.000128		0.996203	0.003782	1.50E-05	
	2	36.86751		0.969816	0.029264	0.00092		0.989898	0.009992	0.000109	
	3	37.89411		0.92334	0.07046	0.0062		0.974249	0.024983	0.000768	
	4	40.64959		0.812185	0.150739	0.037076		0.935772	0.059084	0.005145	
	5	47.79892		0.588092	0.246799	0.165109		0.844877	0.124631	0.030492	
	6	49.1423		0.546172	0.283835	0.169993		0.816765	0.150136	0.033099	
	7	51.72174		0.470683	0.349775	0.179542		0.759347	0.202007	0.038646	
	8	56.02283		0.35865	0.443224	0.198126		0.65442	0.295359	0.05022	
	9	61.97076		0.230824	0.53225	0.236926		0.496008	0.429291	0.074701	
	10	68.97376		0.121172	0.558403	0.320425		0.312681	0.558218	0.1291	
	reboiler	76.85653		0.049151	0.474617	0.476232		0.154095	0.596705	0.249201	
				n-pentane	n-hexane	n-heptane		n-pentane	n-hexane	n-heptane	
Compare with Raoult's Law for various stages. Example comparisons:											
Using	Shortcut	K-ratio	(Antoine	will give slightly different results)							
Bubble Calculation		T(K)=	341.4224	68.27241			Dew Calculation	T(K)=	313.0416	39.89162	
		P(MPa)=	0.101325					P(MPa)=	0.101325		
		Compare	with stage 10					Compare	with stage 4		
	Specie	n-pentane	n-hexane	n-heptane			Specie	n-pentane	n-hexane	n-heptane	
	Tc(K)	469.7	507.4	540.3			Tc(K)	469.7	507.4	540.3	
	Pc(MPa)	3.369	3.012	2.736			Pc(MPa)	3.369	3.012	2.736	
	w	0.249	0.305	0.349			w	0.249	0.305	0.349	
	K	2.67193	0.983684	0.39618			K	1.15701	0.382461	0.140053	
	x	0.121172	0.558403	0.320425			y	0.935772	0.059084	0.005145	
	y	0.323762	0.549292	0.126946		1	x	0.808785	0.154483	0.036733	1.000001



Vapor and liquid assumed in equilibrium leaving stage



P

10.8 Concepts, the bigger picture

$$dG^L = \underline{V}^L dP - \underline{S}^L dT + \left(\frac{\partial G}{\partial n_1}\right)_{T,P,n_2} dn_1 + \left(\frac{\partial G}{\partial n_2}\right)_{T,P,n_1} dn_2$$

Chemical potential $\mu_i \equiv \left(\frac{\partial G}{\partial n_i}\right)_{T,P,n_{j\neq i}}$

Partial Molar Properties

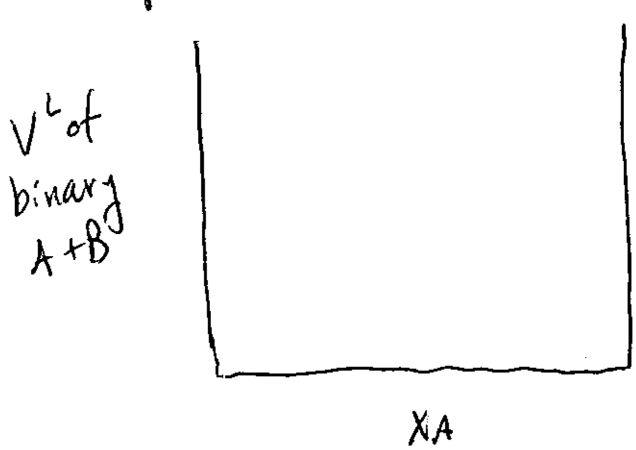
$$\left(\frac{\partial \underline{M}}{\partial n_i}\right)_{T,P,n_{j\neq i}} = \bar{M}_i$$

Results in

$$\underline{M} = n_1 \bar{M}_1 + n_2 \bar{M}_2$$

$$M = x_1 \bar{M}_1 + x_2 \bar{M}_2$$

Graphical Interpretation



Chemical Potential, fugacity in a mixture

Pg 316 shows $\mu_i^V = \mu_i^L$ 10.39

For pure fluid, we used $dG \equiv RT\,d\ln f$

for a mixture component

$$d\mu_i = RT\,d\ln \hat{f}_i \qquad 10.41$$

10.9 Chemical Potential, fugacity for ideal gas

$\bar{U}_i = U_i \qquad U = y_1 U_1 + y_2 U_2$

$\bar{V}_i = V_i \qquad V = y_1 V_1 + y_2 V_2$

$S = y_1 S_1 + y_2 S_2 - R[y_1 \ln y_1 + y_2 \ln y_2]$

$$\mu_i^{ig} = G_i^{ig} + RT \ln y_i \qquad 10.54$$

$$\mu_i^{ig} - \mu_{i,pure}^{ig} = RT \ln \frac{\hat{f}_i^{ig}}{f_{i,pure}^{ig}} = RT \ln y_i$$

$$\hat{f}_i^{ig} = y_i f_{i,pure}^{ig} = y_i P \qquad 10.56$$

10.10 Ideal Solutions

$U = x_1 U_1 + x_2 U_2$

$V = x_1 V_1 + x_2 V_2$

$S = x_1 S_1 + x_2 S_2 - R[x_1 \ln x_1 + x_2 \ln x_2]$

$H = x_1 H_1 + x_2 H_2$

$G = H - TS =$

$$\hat{f}_i^{is} = x_i f_{i,pure} \qquad 10.63$$