Programming Series

“Material Balance Simulation of Binary Mixture in Two Phase Separator”

Just using python!

Naufal Hadi

Overview

Two-phase separators (also known as flash separator, flashing vessel/tank, and knock-out drum) are one of the most common types of process equipment used in the industry including oil refineries, chemical plants, refrigeration systems, natural gas and petrochemical processing plants. Two phase separators handles two-phase fluids. One is gaseous phase and the other is liquid phase. As its name suggests, it is used for separating gas and liquid in wet gas stream, or more generally the gas/liquid stream.

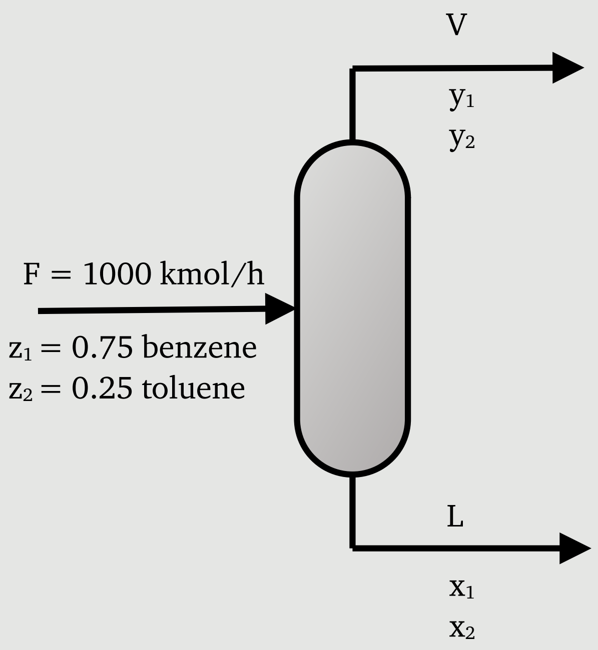
Horizontal two phase separator

Material balance on two phase separator can be achieved by applying thermodynamic. It’s including equation of state and vapor liquid equilibrium. This calculation are easily solved using process simulation software like Hysys, Aveva, DWSIM and Matlab. But in this case I will perform the calculation only using python, a multi purpose programming language. So let’s take a look a problem.

The Problem >>>

The Problems

A stream at 1000 kmol/h consisting mixture of benzene and toluene are separated using two phase separator. The stream contains 0.75 (mole fraction) of benzene (z_1). The separator operating condition is 87 ^{0}C and atmospheric pressure (1.01 bar). Using thermodynamic properties of both component and Peng-Robinson as equation of state, define :
a. molar fraction of both component in vapor phase (y_1 and y_2)
b. molar fraction of both component in liquid phase (x_1 and x_2)
c. vapor flows at top separator (V)
d. liquid flows at bottom separator (L)



Solutions >>>

WARNING !

Freeze at this page and try to solve on your own

Just kidding, ☺
let's slide to next page

Solutions >>>

Solution

The solution are done by deriving material balance and combining with vapor liquid equilibrium.

from material balance :

$$F = L + V \quad \Rightarrow \quad L = F - V$$

for i component gives :

$$Fz_i = Lx_i + Vy_i$$

because $L = F - V$:

$$Fz_i = Fx_i - Vx_i + Vy_i$$

solving for y_i :

$$y_i = \frac{Fz_i + Vx_i - Fx_i}{V}$$

$$y_i = \frac{F}{V} z_i + \left(1 - \frac{F}{V}\right) x_i \qquad \text{and} \quad \frac{F}{V} = \varphi$$

$$\boxed{y_i = \frac{(\varphi - 1)}{\varphi} x_i + \frac{1}{\varphi} z_i} \qquad (1)$$

From vapor liquid equilibrium :

$$K_i = \frac{y_i}{x_i}$$

and : $$\Sigma x_i = \Sigma y_i = 1 \quad \Rightarrow \quad \Sigma x_i - \Sigma y_i = 0$$

now divide equation (I) with x_i respectively gives :

$$K_i = \frac{(\varphi - 1)}{\varphi} + \frac{1}{\varphi} \frac{z_i}{x_i}$$

rearranging :

$$K_i = 1 - \frac{1}{\varphi} + \frac{z_i}{x_i} \frac{1}{\varphi}$$

$$0 = (1 - K_i) + \left(\frac{z_i}{x_i} - 1\right) \frac{1}{\varphi}$$

multiply both side with φ gives :

$$0=(1-K_i)\varphi+(\frac{z_i}{x_i}-1)$$

rearranging and apply $K_i=\frac{y_i}{x_i}$

$$\frac{z_i}{y_i}K_i=1+\varphi(K_i-1)$$

solving for y_i :

$$y_i=\frac{z_i K_i}{1+\varphi(K_i-1)}$$

by dividing both side with K_i, gives x_i :

$$x_i=\frac{z_i}{1+\varphi(K_i-1)} \qquad \text{with} \quad x_i=\frac{y_i}{K_i}$$

recalling $\Sigma x_i - \Sigma y_i = 0$:

$$\Sigma\frac{z_i K_i}{1+\varphi(K_i-1)}-\Sigma\frac{z_i}{1+\varphi(K_i-1)}=0$$

yield :

$$\boxed{\Sigma\frac{z_i(K_i-1)}{1+\varphi(K_i-1)}=0} \qquad (2)$$

The equation (2) also called as Rachford-Rice Equation and used to solve material balance with gives good convergence stability. Now we are going to solve this equation by python code using it's library. The library are called *phasepy*, which is python package for calculation of physical properties of phases at thermodynamic equilibrium. For installation and documentation you can visit their website.

The Code >>>

The Code

To easily understand how the code are working, I suggest using Jupyter Notebook, a scientific python editor. It has friendly interface which return every line of codes that we write. Now launch Jupyter Notebook.

Import package

```
[1]: # import necessary library
     import numpy as np
     from phasepy import component, mixture, preos
     from phasepy.equilibrium import flash
```

Now set given data, operating condition, and thermodynamic properties of components

```
[2]: # given data
     F = 1000 # molar flow. kmol/h
     T = 87. + 273.15   # vessel temperature, converted to K
     P = 1.01 # vessel pressure, bar
     Z = np.array([0.75, 0.25]) # overall molar fraction component
```

```
[3]: # define component thermodynamical properties
     benzene = component(name='benzene', Tc=562.2, Pc=48.98, Zc=0.271, Vc=259.0,
                          w=0.210,
                          Ant=[13.7819, 2726.81, 217.572],
                          GC={'CH=C':6})

     toluene = component(name='toluene', Tc=591.8, Pc=41.06, Zc=0.264, Vc=316,
                        w=0.262,
                        Ant=[13.9320, 3056.96, -217.625],
                        GC={'CH=C':6, 'CH3':1})
```

Define equation of state and mixing rules for mixture

```
[4]: # setting up eos
     mix = mixture(benzene, toluene) #mix given component
     mix.unifac() # using dortmund modified unifac mixing rule
     eos = preos(mix, 'mhv_unifac') # peng robinson equation of state
```

Now our initial setting are complete, let's set random value for initial guess

```
[5]: # initial guess
     x0 = np.array([0.4, 0.6]) # liquid molar fraction to guess
     y0 = np.array([0.2, 0.8]) # vapor molar fraction to guess
```

And start calculation using `flash` *command*

```
[6]: # start calculation
     sep = flash(x0, y0, 'VL', Z, T, P, eos) # solver for flash calculation
```

```
[7]: sep
```

`Sep` *stand for separation, which is new variable to store calculation result. The* `flash` *command will return array that contains molar vapor fraction (y_1 and y_2), liquid vapor fraction (x_1 and x_2) and overall liquid fraction (x). Now let's check the results.*

```
[7]: (array([0.81063167, 0.18936833]),
      array([0.64528823, 0.35471177]),
      0.3667014050359152)
```

In order to facilitate further calculations, we must separate one by one to independent variable. To do that we just index the array

```
[8]: y1 = (sep[0])[0] # benzene vapor fraction, indexing from tuple
     y2 = (sep[0])[1] # toluene vapor fraction, indexing from tuple
     x1 = (sep[1])[0] # benzene liquid fraction, indexing from tuple
     x2 = (sep[1])[1] # benzene liquid fraction, indexing from tuple
```

```
[9]: y1, y2, x1, x2
```

```
[9]: (0.8106316699864328,
      0.18936833001356734,
      0.6452882348270927,
      0.35471176517290726)
```

Now y_1, y_2, y_x, x_2, are independent variable that contains molar fraction in liquid and vapor phase for both component. Next we have calculate the flow of vapor phase at top separator and liquid phase at bottom separator. This is done by simply applying linear algebra solve it using matrix.

from material balance of each component gives :

$$Fz_1 = Lx_1 + Vy_1 \qquad \textit{(benzene component)}$$

$$Fz_2 = Lx_2 + Vy_2 \qquad \textit{(toluene component)}$$

we can rearrange it in matrix form :

$$\begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \end{pmatrix} \begin{pmatrix} L \\ V \end{pmatrix} = \begin{pmatrix} Fz_1 \\ Fz_2 \end{pmatrix}$$

Matrix generation in jupyter notebook

```
[10]: # compute vapor flow and liquid flow using matrix
      import scipy.linalg # import new library
      A = np.array ([[y1, x1],[y2, x2]])
      B = np.array ([F*Z[0], F*Z[1]])
```

```
[11]: print (A)
      print (B)
```

```
[10]:  [[0.81063167 0.64528823]
        [0.18936833 0.35471177]]
       [750. 250.]
```

In order to solve this matrix, we have to import another package that handle linear algebra system

```
[12]: # now solve the matrix
      C = scipy.linalg.solve(A,B) # solver for linear algebra
```

And final results that contains vapor and liquid flow

```
[13]: print (C)
```

```
[13]: [633.29859496 366.70140504]
```

Complete Code >>>

Complete Code

separator.py

```python
"""
Created on Sun Feb 14 2021 17:00:15
@author: naufalovich
"""

# This program perform calculating Bubble/ Dew point of binary mixture
# using phasepy libray and peng-robinson equation of state
# import necessary library
import numpy as np
from phasepy import component, mixture, preos
from phasepy.equilibrium import flash
import scipy.linalg

# given data
F = 1000 # molar flow. kmol/h
T = 85. + 273.15  # vessel temperature, converted to K
P = 1.01 # vessel pressure, bar
Z = np.array([0.75, 0.25]) # overall molar fraction component

# define component thermodynamical properties
benzene = component(name='benzene', Tc=562.2, Pc=48.98, Zc=0.271, Vc=259.0, w=0.210,
                        Ant=[13.7819, 2726.81, 217.572],
                        GC={'CH=C':6})
toluene = component(name='toluene', Tc=591.8, Pc=41.06, Zc=0.264, Vc=316, w=0.262,
                      Ant=[13.9320, 3056.96, -217.625],
                      GC={'CH=C':6, 'CH3':1})

# setting up eos
mix = mixture(benzene, toluene) #mix given component
mix.unifac() # using dortmund modified unifac mixing rule
eos = preos(mix, 'mhv_unifac') # peng robinson equation of state

# initial guess
x0 = np.array([0.4, 0.6]) # liquid molar fraction to guess
y0 = np.array([0.2, 0.8]) # vapor molar fraction to guess

# start calculation
sep = flash(x0, y0, 'VL', Z, T, P, eos) # phase compositions, vapor phase fraction
y1 = (sep[0])[0] # benzene vapor fraction, indexing from tuple
y2 = (sep[0])[1] # toluene vapor fraction
x1 = (sep[1])[0] # benzene liquid fraction
x2 = (sep[1])[1] # benzene liquid fraction

# compute vapor flow and liquid flow using matrix
A = np.array ([[y1, x1],[y2, x2]])
B = np.array ([F*Z[0], F*Z[1]])

# now solve the matrix
C = scipy.linalg.solve(A,B) # solver for linear algebra

# print all solution
print ("-----------------------------------------------")
print ("Vapor Flow (V) :", C[0], "kmol/h")
print ("     benzene fraction in vapor (y1) :", y1)
print ("     toluene fraction in vapor (y2) :", y2)
print ("-----------------------------------------------")
print ("Liquid Flow (L) :", C[1], "kmol/h")
print ("     benzene fraction in liquid (x1) :", x1)
print ("     toluene fraction in liquid (x2) :", x2)
```

This code are available in my github page. You can download or pull the code for free. To run the code just simply type in terminal :

```
python separator.py
```

Then we get :

```
Out

   -----------------------------------------------
   Vapor Flow (V) : 122.04308131931882 kmol/h
        benzene fraction in vapor (y1) : 0.8642111622668507
        toluene fraction in vapor (y2) : 0.13578883773314926
   -----------------------------------------------
   Liquid Flow (L) : 877.9569186806812 kmol/h
        benzene fraction in liquid (x1) : 0.7341237287758344
        toluene fraction in liquid (x2) : 0.2658762712241656
```

Thanks for your time !

Enjoy this post and feel free to like and comment.

End