# cheg325 SIS 10.1-1

AUTHOR
k.wodehouse

```
import numpy as np
import pandas as pd
from scipy.optimize import fsolve

df = pd.read_csv('101 data.txt', sep=',', index_col=0)
df['x'] /= 100
df
```

| | x | A | B |
|---|---|---|---|
| **Component** | | | |
| Ethane | 0.05 | 817.08 | 4.402229 |
| Propane | 0.10 | 1051.38 | 4.517190 |
| n-Butane | 0.40 | 1267.56 | 4.617679 |
| 2-Methylpropane | 0.45 | 1183.44 | 4.474013 |

where $A, B$ are parameters for this equation

$$\log_{10}(P^{vap}) = -\frac{A}{T} + B$$

## (a)

for finding our bubble point at 5 bar we know $P_{total}$ and $x_i$, so this is our “bubble T”. we solve this iteratively by choosing temperatures and then calculating the total pressure and seeing if it matches our given $P_{total}$ of 5 bar.

```
components = list(df.index)
Ptotal = 5.0

def P_calculated(T):
    P_i = 0
    for component in components:
        A, B, x = df.loc[component]['A'], df.loc[component]['B'], df.loc[component]['x']
        P_i += (10**(B - A/T)) * x

    return abs(P_i - Ptotal)

dew_temp = fsolve(P_calculated, 500)[0]
print(f'dew temp: {dew_temp:.2f} ºK')
```

```
dew temp: 293.66 ºK
```

## (b)

for the dew point we can just start with an initial guess thats super low and let the fsolve work its magic.

```
components = list(df.index)
Ptotal = 5.0

def P_calculated(T):
    P1 = 0
    for component in components:
        A, B, y = df.loc[component]['A'], df.loc[component]['B'], df.loc[component]['x']
        pvap = (10**(B - A/T))
        P1 += y / pvap

    return abs((1/P1) - Ptotal)

dew_temp = fsolve(P_calculated, 500)[0]
print(f'dew temp: {dew_temp:.2f} ºK')
```

```
dew temp: 314.23 ºK
```

## (c)

first, we calculate $V$ from knowing that the sum of $\{y_i\}$ must be one.

then, we use that $V$ to calculate the vapor and liquid mole fractions.

```
components = list(df.index)
Ptotal = 5.0
T = 273.15 + 30.0

def flash_calculation(V):
    sum_thingy = 0
    for component in components:
        A, B, z_iF = df.loc[component]['A'], df.loc[component]['B'], df.loc[component]['x
        pvap = (10**(B - A/T))
        ki = pvap / Ptotal
        sum_thingy += (z_iF * ki)/(1 + V*(ki - 1))
    return abs(1 - sum_thingy)

V_solved = fsolve(flash_calculation, 0.5)[0]
print(f'V: {V_solved:.4f}\n')

def flash_calculation(V):
    print('----- vapor mole fractions -----')
    for component in components:
        A, B, z_iF = df.loc[component]['A'], df.loc[component]['B'], df.loc[component]['x
        pvap = (10**(B - A/T))
        ki = pvap / Ptotal
        yi = (z_iF * ki)/(1 + V*(ki - 1))
        print(f'{component}: {yi:.4f}')

    print('\n----- liquid mole fractions -----')
    for component in components:
        A, B, z_iF = df.loc[component]['A'], df.loc[component]['B'], df.loc[component]['x
        pvap = (10**(B - A/T))
        ki = pvap / Ptotal
        yi = (z_iF * ki)/(1 + V*(ki - 1))
        xi = yi / ki
        print(f'{component}: {xi:.4f}')

flash_calculation(V_solved)
```

```
V: 0.1334

----- vapor mole fractions -----
Ethane: 0.2289
Propane: 0.1921
n-Butane: 0.2326
2-Methylpropane: 0.3464

----- liquid mole fractions -----
Ethane: 0.0225
Propane: 0.0858
n-Butane: 0.4258
2-Methylpropane: 0.4659
```

## (d)

for our adiabatic process we will need our balance equations on top of the raoults law. let’s assume the process is a continuous flow adiabatic expansion (flashbacks to cheg231 joule thompson stuff)

mass balance

$$\frac{dm}{dt} = \dot{m}_{\text{in}} - \dot{m}_{\text{out}} = 0 \qquad \text{(since steady state)}$$

energy balance (applying knowledge from mass bal) (ignoring KE, PE)

$$\frac{d}{dt}[U] = 0 = \dot{Q} + \dot{W}_s + \dot{W}_{pv} + \dot{m}\left(\hat{H}_{in} - \hat{H}_{out}\right) = 0$$

$$0 = \hat{H}_{in} - \hat{H}_{out} \qquad \text{(isenthalpic!)}$$

and we still can use our raults law for each component

$$x_i P_i^{\text{vap}} = y_i P_{\text{total}}$$

if you were to solve this as before you’d just need more iteration with another equation and another unknown variable.

```
# this is filler text!
```