Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Российский химико-технологический университет имени Д.И. Менделеева»

Факультет цифровых технологий и химического инжиниринга

Кафедра информационных компьютерных технологий

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3**

**ПО КУРСУ**

**«ЦИФРОВОЕ МОДЕЛИРОВАНИЕ ФИЗИКО-ХИМИЧЕСКИХ СИСТЕМ»:**

**«Моделирование бинарных парожидкостных систем с использованием модели Вильсона»**

Ведущий преподаватель

Ст. преподаватель Скичко Е.А.

**СТУДЕНТ группы КС-26** Золотухин А.А.

**Москва**

**2024**

# Задание

1. Используя приведенные данные (табл. 1), подобрать константы в модели Вильсона и построить y-x и P-x диаграмму.
   1. Исследуемая система (*2 вариант*): Ацетон + Хлороформ.

Таблица 1

| Вариант 2 | | |
| --- | --- | --- |
| Данные VLE, T | 298.15K | |
| X1 | Y1 | P, bar |
| 0.100 | 0.055 | 0.2420 |
| 0.300 | 0.275 | 0.2216 |
| 0.500 | 0.589 | 0.2287 |
| 0.700 | 0.780 | 0.2546 |
| 0.900 | 0.951 | 0.2887 |
| Параметры | | |
|  | 1 | 2 |
| A | 16.65 | 15.97 |
| B | 2940.46 | 2696.79 |
| C | -35.93 | -46.16 |
| Tc, K | 508.10 | 536.40 |
| Pc, bar | 47.01 | 54.72 |
| w | 0.31 | 0.22 |

Теоретическое обоснование решения.

**Модель Вильсона[[1]](#footnote-0)** хорошо описывает парожидкостное равновесие VLE (Vapour-Liquid Equilibrium) сильно неидеальных систем со специфическим взаимодействием между молекулами компонентов.

Концентрация *i-ого* компонента () в паровой фазе определяется уравнением:

| , |  |
| --- | --- |

где - коэффициент активности *i-ого* компонента,

- концентрация *i-ого* компонента в жидкости,

- давление паров *i-ого* компонента, *Па*,

- общее давление в системе, *Па*.

При этом для компонентов *1* и *2*:

|  |  |
| --- | --- |

где *,*

*.*

Здесь и - молярные объемы чистых жидких компонентов *1* и *2*,

и - параметр, определяющий разность энергии взаимодействия молекул *1-го* и *2-го* компонентов друг с другом и между собой, рассчитывается по экспериментальным данным бинарной смеси.

Находим давления паров чистых жидкостей и при *T* по **уравнению Антуана**:

| *,* |  |
| --- | --- |

где *T* - термодинамическая температура, *K*,

*A1, B1, C1, A2, B2, C2* - константы, специфичные для вещества, которые зависят от конкретного рассматриваемого вещества.

Предполагается, что имеется несколько экспериментальных точек для смеси при температуре *T*. Таких точек пять, т.е. пяти значениям *x* соответствуют пять экспериментальных равновесных значений *y* и *P*. Для каждой точки следует рассчитать и из *(1)*, *(4)* и *(5)* по уравнениям :

|  |  |
| --- | --- |

Для каждой точки рассчитывают избыточную мольную энергию Гиббса (*gE*):

| , |  |
| --- | --- |

где *R* - универсальная газовая постоянная, *Дж/(моль \* K)*.

Избыточная энергия Гиббса по Вильсону:

|  |  |
| --- | --- |

Далее нужно подобрать константы уравнения *(9)* таким образом, чтобы минимизировать расхождение между значениями , рассчитанным по уравнению и определенным по экспериментальным данным.

Затем, используя *(6)* и *(7)*, нахожу и при произвольно выбранных значениях *x1* в диапазоне от *x1* = 0 до *x1* = 1.

В конце, для каждого выбранного значения *x1* нахожу соответствующие значения *y1* и *P* путем решения уравнений *(6)* и *(7)*, используя соотношения массового баланса и *.* По полученным результатам нужно построить диаграммы *y – x* и *P – x*.

Код.

import numpy as np

import matplotlib.pyplot as plt

T = 298.15

R\_temp = 8.31

X\_1 = [0.100, 0.300, 0.500, 0.700, 0.900]

Y\_1 = [0.055, 0.275, 0.589, 0.780, 0.951]

P\_bar = [0.2420, 0.2216, 0.2287, 0.2546, 0.2887]

P = [i \* 1e5 for i in P\_bar]

R = 1.987

A\_1, A\_2 = 16.65, 15.97

B\_1, B\_2 = 2940.46, 2696.79

C\_1, C\_2 = -35.93, -46.16

Tc\_1, Tc\_2 = 508.10, 536.40

Pc\_1, Pc\_2 = 47.01, 54.72

w\_1, w\_2 = 0.31, 0.22

def gamma\_calc(X\_1, X\_2, Y\_1, Y\_2, P\_vp\_1, P\_vp\_2):

gamma\_1 = []

gamma\_2 = []

for i in range(len(X\_1)):

gamma\_1.append((Y\_1[i] \* P[i]) / (X\_1[i] \* P\_vp\_1))

gamma\_2.append((Y\_2[i] \* P[i]) / (X\_2[i] \* P\_vp\_2))

return (gamma\_1, gamma\_2)

def gibbs\_energy\_calc(X\_1, X\_2, gamma\_1, gamma\_2):

g\_e = []

for i in range(0, len(X\_1)):

g\_e.append(R \* T \* (X\_1[i] \* np.log(gamma\_1[i]) + X\_2[i] \* np.log(gamma\_2[i])))

return g\_e

def gamma\_Wilson\_calc(X\_1, X\_2, lambda\_12, lambda\_21):

gamma\_1 = []

gamma\_2 = []

for i in range(len(X\_1)):

gamma\_1.append(np.exp(-1 \* np.log(X\_1[i] + lambda\_12 \* X\_2[i]) + X\_2[i] \* ((lambda\_12 / (X\_1[i] + lambda\_12 \* X\_2[i])) - (lambda\_21 / (X\_2[i] + lambda\_21 \* X\_1[i])))))

gamma\_2.append(np.exp(-1 \* np.log(X\_2[i] + lambda\_21 \* X\_1[i]) - X\_1[i] \* ((lambda\_12 / (X\_1[i] + lambda\_12 \* X\_2[i])) - (lambda\_21 / (X\_2[i] + lambda\_21 \* X\_1[i])))))

return (gamma\_1, gamma\_2)

def gibbs\_energy\_Wilson\_calc(X\_1, X\_2, lambda\_12, lambda\_21):

g\_e = []

for i in range(0, len(X\_1)):

g\_e.append(R \* T \* (-X\_1[i] \* np.log(X\_1[i] + lambda\_12 \* X\_2[i]) - X\_2[i] \* np.log(X\_2[i] + lambda\_21 \* X\_1[i])))

return g\_e

def main():

X\_2 = []

for i in range(0, len(X\_1)):

X\_2.append(1 - X\_1[i])

Y\_2 = []

for i in range(0, len(Y\_1)):

Y\_2.append(1 - Y\_1[i])

P\_vp\_1 = np.exp(A\_1 - B\_1 / (T + C\_1))

P\_vp\_2 = np.exp(A\_2 - B\_2 / (T + C\_2))

gamma\_1, gamma\_2 = gamma\_calc(X\_1, X\_2, Y\_1, Y\_2, P\_vp\_1, P\_vp\_2)

g\_e = gibbs\_energy\_calc(X\_1, X\_2, gamma\_1, gamma\_2)

X\_1\_W = np.arange(0, 1.01, 0.01)

X\_2\_W = []

for i in range(0, len(X\_1\_W)):

X\_2\_W.append(1 - X\_1\_W[i])

X\_1\_W = np.arange(0, 1.01, 0.01)

X\_2\_W = []

for i in range(0, len(X\_1\_W)):

X\_2\_W.append(1 - X\_1\_W[i])

alpha\_12, alpha\_21 = 116, -506

Z\_RA\_1 = 0.29056 - 0.08775 \* w\_1

Z\_RA\_2 = 0.29056 - 0.08775 \* w\_2

Tr\_1 = T / Tc\_1

Tr\_2 = T / Tc\_2

V\_L\_1 = (R\_temp \* Tc\_1 / Pc\_1) \* Z\_RA\_1 \*\* (1 + (1 - Tr\_1) \*\* (2 / 7))

V\_L\_2 = (R\_temp \* Tc\_2 / Pc\_2) \* Z\_RA\_2 \*\* (1 + (1 - Tr\_2) \*\* (2 / 7))

lambda\_12 = V\_L\_2 / V\_L\_1 \* np.exp(-1 \* alpha\_12 / (R \* T))

lambda\_21 = V\_L\_1 / V\_L\_2 \* np.exp(-1 \* alpha\_21 / (R \* T))

gamma\_1\_W, gamma\_2\_W = gamma\_Wilson\_calc(X\_1\_W, X\_2\_W, lambda\_12, lambda\_21)

Y\_1\_W = [0]

for i in range(1, len(X\_1\_W)):

Y\_1\_W.append(1 / ((gamma\_2\_W[i] \* X\_2\_W[i] \* P\_vp\_2) / (gamma\_1\_W[i] \* X\_1\_W[i] \* P\_vp\_1) + 1))

Y\_2\_W = []

for i in range(0, len(Y\_1\_W)):

Y\_2\_W.append(1 - Y\_1\_W[i])

plt.figure

plt.grid(True)

plt.title("Ацетон (y-x)")

plt.plot(X\_1\_W, Y\_1\_W, color='blue')

plt.scatter(X\_1, Y\_1, color='orange')

plt.xlabel("X")

plt.ylabel("Y")

plt.show()

plt.figure

plt.grid(True)

plt.title("Хлороформ (y-x)")

plt.plot(X\_2\_W, Y\_2\_W, color='blue')

plt.scatter(X\_2, Y\_2, color='orange')

plt.xlabel("X")

plt.ylabel("Y")

plt.show()

P\_1\_W = []

for i in range(0, len(X\_1\_W)):

P\_1\_W.append(gamma\_1\_W[i] \* X\_1\_W[i] \* P\_vp\_1)

P\_2\_W = []

for i in range(0, len(X\_2\_W)):

P\_2\_W.append(gamma\_2\_W[i] \* X\_2\_W[i] \* P\_vp\_2)

P\_W = []

for i in range(0, len(P\_1\_W)):

P\_W.append((P\_1\_W[i] + P\_2\_W[i]) / 750.1)

plt.figure

plt.grid(True)

plt.title("Ацетон-Хлороформ (P-y-x)")

plt.plot(X\_1\_W, P\_W, color="blue")

plt.plot(Y\_1\_W, P\_W, color="orange")

plt.scatter(X\_1, P\_bar, color='orange')

plt.scatter(Y\_1, P\_bar, color='orange')

plt.xlabel("X, Y")

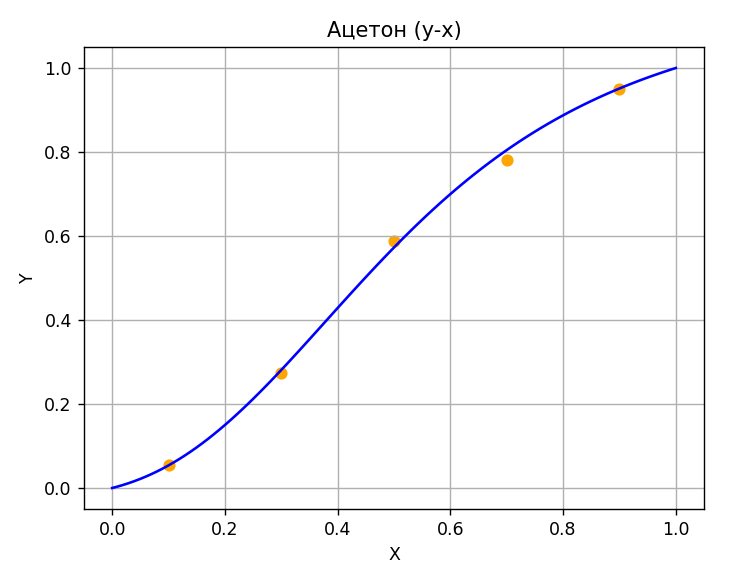
plt.ylabel("Давление (P), бар")

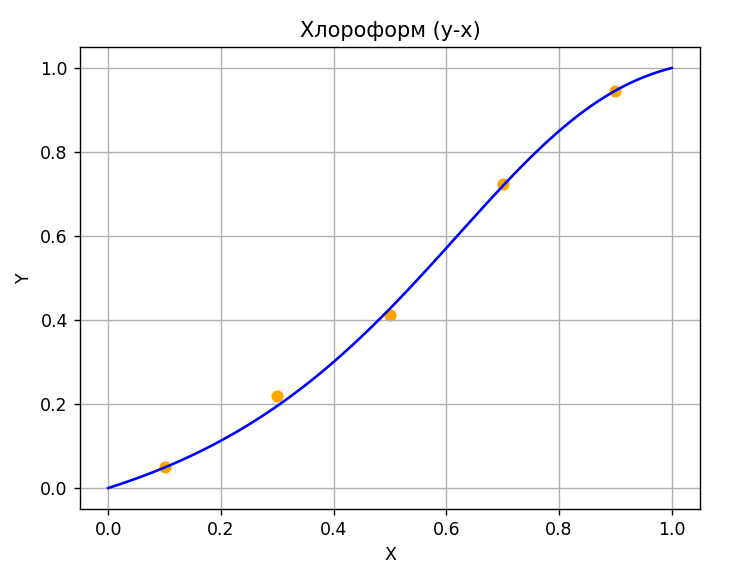
plt.show()

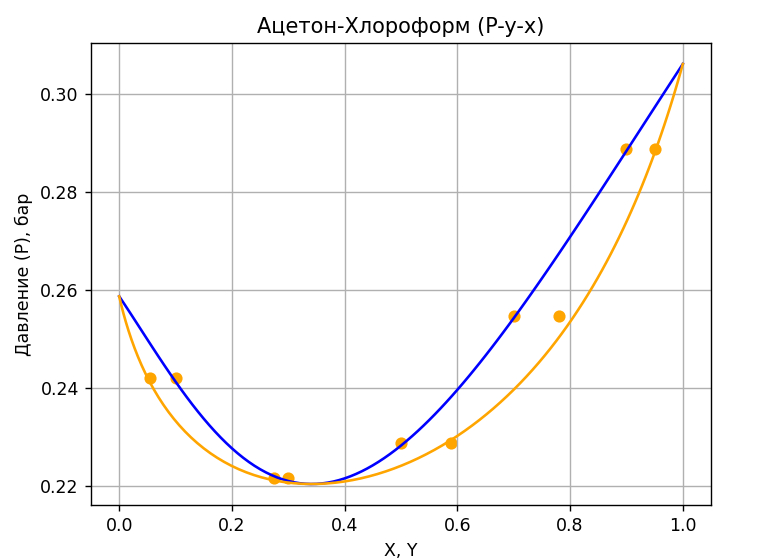
g\_e\_W = gibbs\_energy\_Wilson\_calc(X\_1\_W, X\_2\_W, lambda\_12, lambda\_21)

main()

Результаты расчетов.







1. первая модель локального состава, применяемая для описания уравнения состояния жидкостей. [↑](#footnote-ref-0)