Инженерная школа природных ресурсов

Направление подготовки Химическая технология

Отделение химической инженерии

**PYTHON ДЛЯ ЗАДАЧ ХИМИЧЕСКОЙ ТЕХНОЛОГИИ**

**Отчет по лабораторной работе № 3**

**Введение в библиотеки NumPy, SciPy и Matplotlib**

**(продолжение)**

Выполнил студент гр. 2ДМ24 Грива Д.В.

(Подпись)

21 \_\_декабря\_\_\_\_ 2023 г.

Отчет принят:

Преподаватель

доцент ОХИ ИШПР, к.т.н. В.А. Чузлов

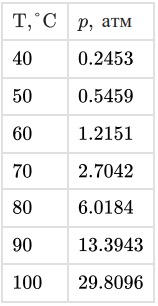
(Подпись)

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

Томск 2023 г.

**Задание 1**

Дана зависимость давления паров вещества от температуры:



Определить значения давления паров при T ∈ [40; 100] с шагом 5 °C, используя:

Кубический сплайн;

Одну из аппроксимирующих функций: проверить линейную, степенную и экспоненциальную аппроксимирующие функции, выбрать наиболее подходящую (по значению суммарной ошибки) и провести расчеты с использованием данной функции.

**Программная реализация:**

import numpy as np

from scipy.interpolate import CubicSpline

from scipy.optimize import least\_squares

import matplotlib.pyplot as plt

def linear(x: float | np.ndarray, params: tuple[float, float]) -> float | np.ndarray:

a0, a1 = params

return a0 + a1 \* x

def power(x: float | np.ndarray, params: tuple[float, float]) -> float | np.ndarray:

a, b = params

return a \* x \*\* b

def exponent(x: float | np.ndarray, params: tuple[float, float]) -> float | np.ndarray:

a, b = params

return a \* np.exp(b \* x)

def residuals(params: tuple[float, float], x: np.ndarray, y: np.ndarray, func: callable) -> np.ndarray:

return y - func(x, params)

# Data preparation

temperatures = np.array([40, 50, 60, 70, 80, 90, 100])

pressures = np.array([0.2453, 0.5459, 1.2151, 2.7042, 6.0184, 13.3943, 29.8096])

interpolation\_temperatures = np.arange(40, 101, 5)

x0 = (0.01, 0.01)

results\_linear = least\_squares(residuals, x0=x0, args=(temperatures, pressures, linear))

results\_power = least\_squares(residuals, x0=x0, args=(temperatures, pressures, power))

results\_exponent = least\_squares(residuals, x0=x0, args=(temperatures, pressures, exponent))

cubic\_spline = CubicSpline(temperatures, pressures)

cubic\_spline\_pressures = cubic\_spline(interpolation\_temperatures)

linear\_pressures = linear(interpolation\_temperatures, results\_linear.x)

power\_pressures = power(interpolation\_temperatures, results\_power.x)

exponent\_pressures = exponent(interpolation\_temperatures, results\_exponent.x)

cubic\_spline\_model = CubicSpline(temperatures, pressures)

plt.figure(figsize=(12, 8))

plt.plot(temperatures, pressures, 'o', label='Исходные данные')

plt.plot(interpolation\_temperatures, linear\_pressures, label='Линейная аппроксимация')

plt.plot(interpolation\_temperatures, cubic\_spline\_pressures, label='Кубический сплайн')

plt.plot(interpolation\_temperatures, power\_pressures, label='Cтепенная аппроксимация')

plt.plot(interpolation\_temperatures, exponent\_pressures, label='Экспоненциальная аппроксимация')

plt.xlabel('Температура (°C)')

plt.ylabel('Давление паров (atm)')

plt.title('Оценка давления пара с помощью различных методов')

plt.legend()

plt.grid(True)

plt.show()

**Ответ**:

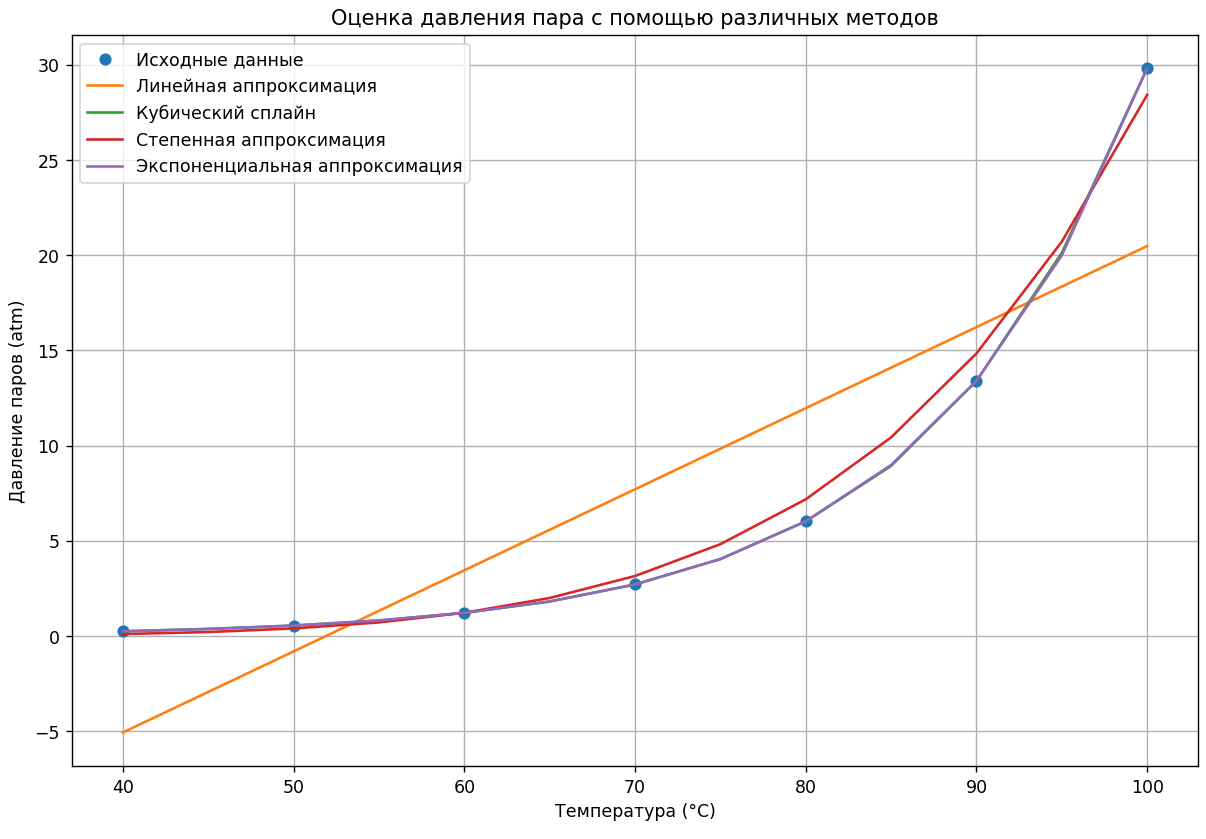


Рисунок 1 – Оценка давления пара с помощью различных методов

Кубическая сплайн-интерполяция представляет собой гладкую кривую, проходящую через все исходные точки данных.

Этот метод известен своей гладкостью и способностью точно моделировать нелинейные данные. Он создает кусочно-непрерывную кривую, что особенно полезно для отражения нелинейной зависимости между температурой и давлением паров.

Линейное приближение изображается в виде прямой линии.

Эта модель предполагает постоянную скорость изменения давления пара в зависимости от температуры. Это самая простая форма аппроксимации, но она может не совсем точно отражать нелинейную природу данных.

Cтепенная аппроксимация выглядит как кривая линия, потенциально более подходящая к точкам данных по сравнению с линейной моделью, но менее гладкая, чем кубический сплайн.

Cтепенная аппроксимация вводит нелинейную зависимость между температурой и давлением пара. Она сложнее линейной модели и может более эффективно моделировать нелинейные тенденции.

Экспоненциальная аппроксимация представлена кривой, которая сначала медленно растет, а затем более резко увеличивается.

Эта модель подходит для данных, в которых скорость изменения возрастает экспоненциально. Она хорошо подходит для данных, полученных при высоких температурах, но может расходиться при более низких температурах.

Кубический сплайн отлично подходит для плавной и точной интерполяции, если известно, что точки данных следуют нелинейному графику.

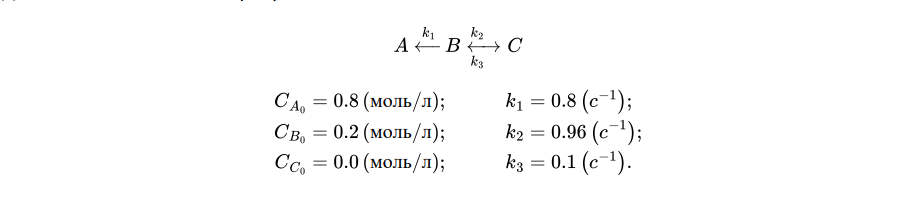
Линейная модель, несмотря на свою простоту, может не подойти для этого набора данных из-за своей нелинейной природы.

Cтепенная аппроксимация обеспечивает баланс между простотой линейной модели и сложностью кубического сплайна.

Экспоненциальная аппроксимация может быть более подходящей для данных, которые показывают экспоненциальный рост, но может быть менее точной при более низких температурах.

**Задание 2**

Дана схема химических превращений:



Решите систему дифференциальных уравнений изменения концентраций веществ во времени при помощи функции scipy.integrate.solve\_ivp() на отрезке [0; 5] с шагом h = 0.1. По результатам расчетов постройте зависимость C (t) для каждого компонента при помощи библиотеки matplotlib.

**Программная реализация:**

from scipy.integrate import solve\_ivp

import numpy as np

import matplotlib.pyplot as plt

def chemical\_system(t, concentrations, k1, k2, k3):

A, B, C = concentrations

dAdt = -k1 \* B + k3 \* C

dBdt = k1 \* B - k2 \* B + k3 \* C

dCdt = k2 \* B - k3 \* C

return [dAdt, dBdt, dCdt]

C\_A\_0 = 0.8

C\_B\_0 = 0.2

C\_C\_0 = 0.0

initial\_concentrations = [C\_A\_0, C\_B\_0, C\_C\_0]

k1 = 0.8

k2 = 0.96

k3 = 0.1

time = (0, 5)

t\_eval = np.arange(0, 5.1, 0.1)

solution = solve\_ivp(chemical\_system, time, initial\_concentrations, args=(k1, k2, k3), t\_eval=t\_eval)

plt.figure(figsize=(10, 6))

plt.plot(solution.t, solution.y[0], label='Концентрация A [A]')

plt.plot(solution.t, solution.y[1], label='Концентрация B [B]')

plt.plot(solution.t, solution.y[2], label='Концентрация C [C]')

plt.xlabel('Время (с)')

plt.ylabel('Концентрация (моль/л)')

plt.title('Изменения концентрации со временем для A, B, C')

plt.legend()

plt.grid(True)

plt.show()

**Ответ**:

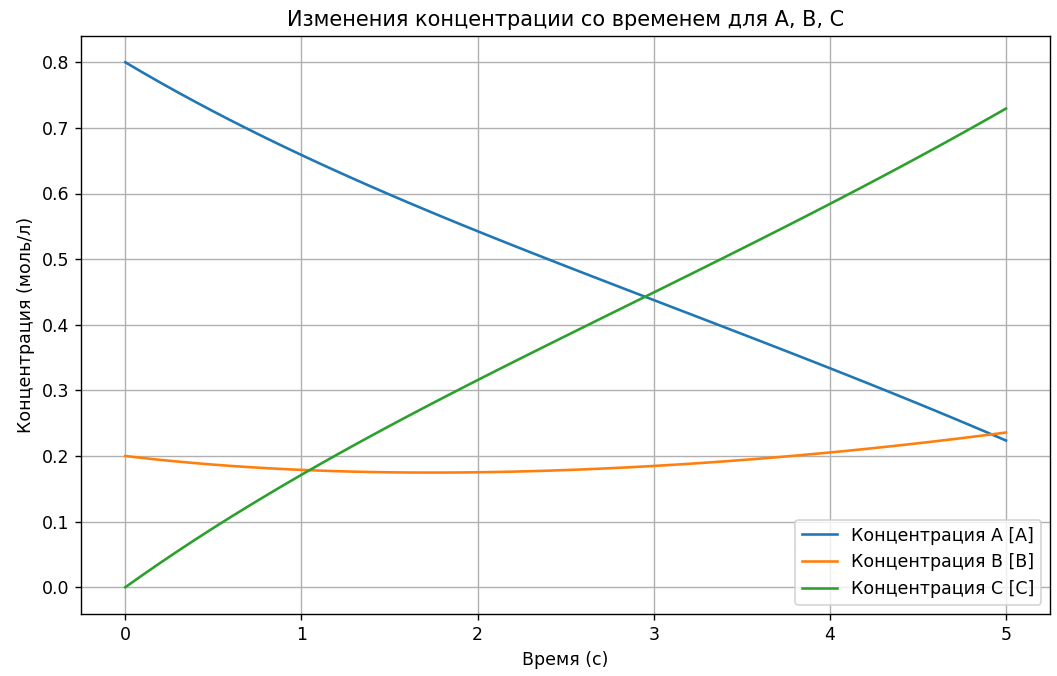


Рисунок 2 – Изменение концентрации со временем

На графике показаны динамические изменения концентраций химических видов A, B и C с течением времени. Концентрация вида A постоянно растет благодаря его образованию из B. Концентрация B сначала немного повышается, но затем снижается, поскольку он расходуется на образование A и C. Концентрация C постоянно растет, поскольку он образуется из B. Такое поведение подчеркивает взаимодействие химических реакций, в которых B выступает в качестве ключевого промежуточного продукта, и подчеркивает влияние скорости реакции на эволюцию системы к динамическому равновесию.

**Задание 3**

:

Используя функцию scipy.integrate.quad() для вычисления значения энтропии воды при ее нагревании от 400 до 500 K по формуле:

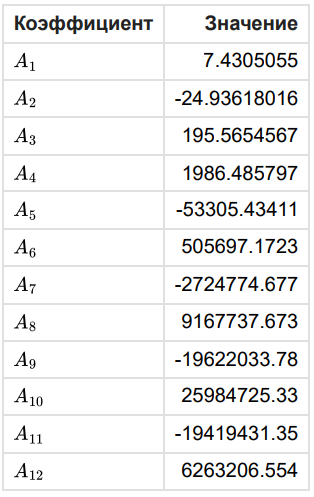
где T - температура, К; η = 3 - количество молей;

- теплоемкость, Дж/(моль К);

R - универсальная газовая постоянная;

= 647,126 - критическая температура, К.

Коэффициенты полинома A(1) − A(12):



**Программная реализация:**

from scipy.integrate import quad

def heat\_capacity(T, coefficients, Tc, R):

tau = 1 - T / Tc

Cv = R \* sum([coeff \* tau\*\*(j - 1) for j, coeff in enumerate(coefficients)])

return Cv

# Given data

coefficients = [7.4305055, -24.93618016, 195.5654567, 1986.485797, -53305.43411,

505697.1723, -2724774.677, 9167737.673, -19622033.78, 25984725.33,

-19419431.35, 6263206.554]

R = 8.314

Tc = 647.126

eta = 3

def integrand(T):

return heat\_capacity(T, coefficients, Tc, R) / T

Delta\_S, error = quad(integrand, 400, 500)

Delta\_S\_total = eta \* Delta\_S

print(Delta\_S\_total, error)

**Ответ**:

Рассчитанное изменение энтропии 136,59 Дж/К при нагревании 3 молей воды с 400 до 500 К является количественным выражением этого рассеивания энергии. Оно отражает увеличение беспорядка или случайности на молекулярном уровне по мере нагревания воды.