CH5150: Project Report

Adil Salfi
CS20BTECH11031

October 7, 2024

Problem Statement

This report addresses the problem of finding the maximum concentration of auto catalyst B given the rate reactions by parameterizing rate reaction constants and using scipy ode solver followed by scipy optimize.

1 Introduction

We are given an autocatalytic Isothermal batch reaction whose rate reactions are given by

$$\frac{dC_A}{dt} = -k_1 C_A C_B$$

$$\frac{dC_B}{dt} = k_1 C_B C_A - k_2 C_B$$

$$\frac{dC_C}{dt} = -k_2 C_B$$

We need to find optimal k_1 and k_2 values for which C_B is maximized and the time at which maximum value of C_B is observed.

2 Methods

Here, the methods used in solving the problem are mentioned. The basic outline of the solution is:

- We first load the experimental data provided using `numpy` and `pandas`. Only C_B and t are stored as they are of prime importance

- We then use an ODE solver to calculate values of C_B as function of time t. This is achieved by using methods from `scipy.integrate`
- We then use `scipy.optimize` methods to optimize $C_B(t)$ calculated from the ODE solver

2.1 ODE solving

Three functions are defined to assist in the ODE solving

1. $\mathbf{rate(t, y, k_1, k_2)} \rightarrow \mathbf{\frac{dA}{dt}, \frac{dB}{dt}, \frac{dC}{dt}}$:

   The function evaluates reaction rates given A, B, C concentrations as y and given some intial k_1, k_2 values

2. $\mathbf{solve\_system(t, k_1, k_2)} \rightarrow \mathbf{sol}$

   This function solves for C_B for values of k_1, k_2 values and from 0 to $t[-1]$ limit range

3. $\mathbf{B\_t\_model\_pred(t, k_1, k_2)} \rightarrow \mathbf{B\_conc}$

   This is a simple placeholder function which invokes solve_system internally and returns C_B for various time intervals given by parameter t

Finally, C_B is returned as a function of $t, k1, k2$.

2.2 Optimization

- We first used `scipy.optimize.minimize` but this simply led to the minimizer assigning $k_2 = 0$
- To prevent this, experimental data had to be encoded to guide the minimizer, therefore we used `scipy.optimize.least_squares`

To this end we use the following functions

1. **residuals(params, t_data, B_data)** $\rightarrow$ **B_pred - B_data**

   This function returns the difference between predicted C_B and the observed C_B

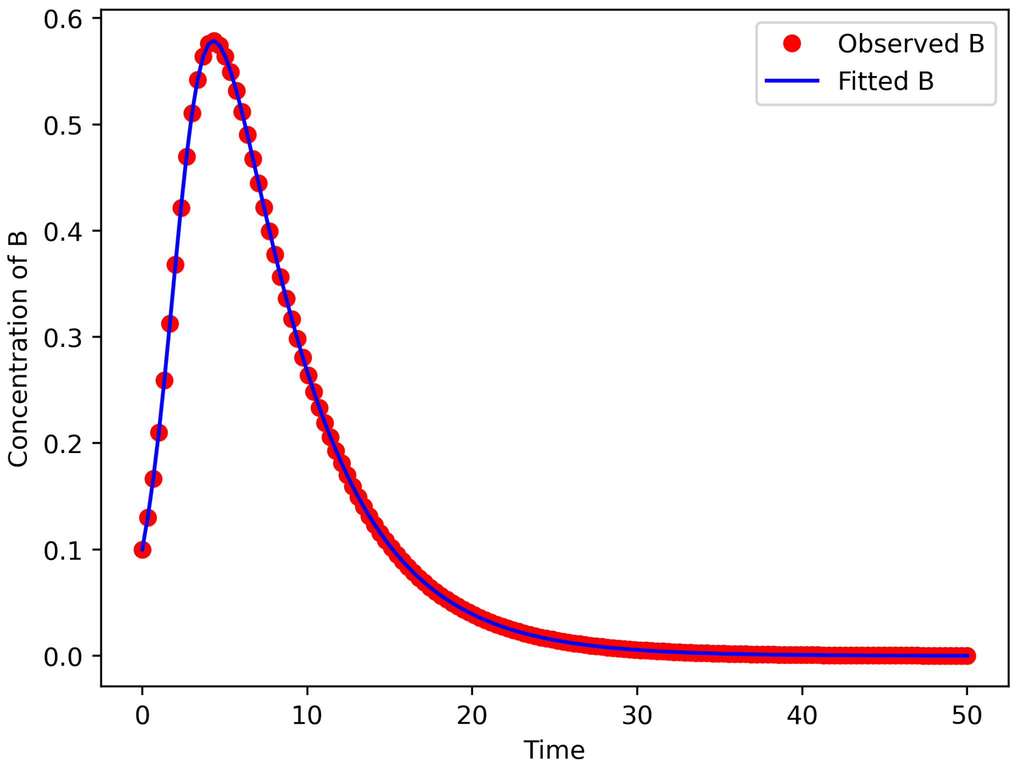
2. **fit(initial_guess) → result**

   This function fits the experimental data and returns the fitted C_B prediction function using `scipy.optimize.least_squares` which internally minimizes the residual function

   Therefore internally a minimizer is being used

3 Results

We were able to accurately fit the C_B function using `scipy.optimize.least_sqaures` by sucessfully minimizing the residual values. Below is the figure



After running the code the optimal values were

$$k_1 = 1, k_2 = 0.2, max(C_B) = 0.5787 \text{ and } t = 4.3624 \text{ seconds}$$

4 Future Work

Following are the ideas that could have been implemented but were infeasible due to time constraints

1. To prevent $k_2 = 0$ from occuring we can customize the objective function by minimizing a penalty term which adds a high penalty if $k_2 < 1e - 6$
2. To prevent overfitting from occuring we could add a regularization term like $\lambda(k_1^2 + k_2^2)$
3. We could use fewer data points to prevent overfitting, this is a naive but effective solution
4. We could use cross validation by splitting data into training and validation sets. This solution is feasible when the dataset is large

References

[1] SciPy Community. fmin. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.fmin.html`, 2024. Accessed: 2024-10-07.

[2] SciPy Community. least_squares. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html#scipy.optimize.least_squares`, 2024. Accessed: 2024-10-07.

[3] SciPy Community. minimize. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html`, 2024. Accessed: 2024-10-07.

[4] SciPy Community. Ode solver. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html`, 2024. Accessed: 2024-10-07.

[2] [4] [1] [3]