Solving Langmuir Adsorption Model as a Neural ODE

Ali Shahbaaz Khan

May 2, 2025

1 Introduction

Langmuir adsorption model was initially derived to study the effects of adsorption of gases onto solid surfaces. The model assumed that the surface is homogenous and that the gas and surface are in equilibrium. Here, we can see how this model can be used for antibiotic adsorption to study how antibiotic molecules attach to activated carbon and biochar which are commonly used for wastewater treatment. Furthermore, we can also use it to study antibiotic-bacteria interaction. Langmuir adsorption dynamics can be stated as an ordinary differential equation :

$$\frac{dq}{dt} = k_a C(Q_m - q) - k_d q$$

where q is the amount of antibiotic adsorbed per unit mass of the membrane, C is the antibiotic concentration in the solution, Qm is the maximum adsorption capacity, k_a is the adsorption rate constant, and k_d is the desorption rate constant.

In this project, we are going to explore how neuralODEs can be used to solve Langmuir Adsorption ODE.

2 Methodology

2.1 Neural ODE

In predictive modeling, the objective is to learn a mapping $\mathbb{R} : X \to Y$. We can say there is a complex function that can represent this transformation. We know that a neural network can be used as a function approximation and, with a significantly large number of hidden layers, we can approximate complex functions. To define such a network, the number of discrete layers has to be stated and each layer has its free parameters. This leads to costly computing. With neural ODE, the hidden layer is modeled as continuous dynamics. We can state the problem as :

$$\frac{dh(t)}{dt} = f(h(t), t, \theta)$$

where θ are the free parameters. We can say that f is a neural network and θ are the parameters of that neural network. Given an input layer, which we can assume to be the initial condition $h(t_0)$ of the equation, our objective is to find $h(T)$ such that $h(T) = y$(y being the prediction for given x). We can find this value by solving:

$$h(T) = y = \int_{t_0}^{T} f(h(t), t, \theta)\, dt$$

We will now solve this equation analytically using Julia.

3 Results

3.1 Numerical Integration

We numerically solve the Langmuir equation using Tsitouras solver($tsit5()$ in Julia) to generate the training data. We use $q_0 = 0.6$ and $(k_a, k_d, q_m, C) = (2.0, 0.2, 1.0, 2.0)$ as initial condition and parameters. We further use a $tspan$ of $0, 5$ and a dataset size of 30.

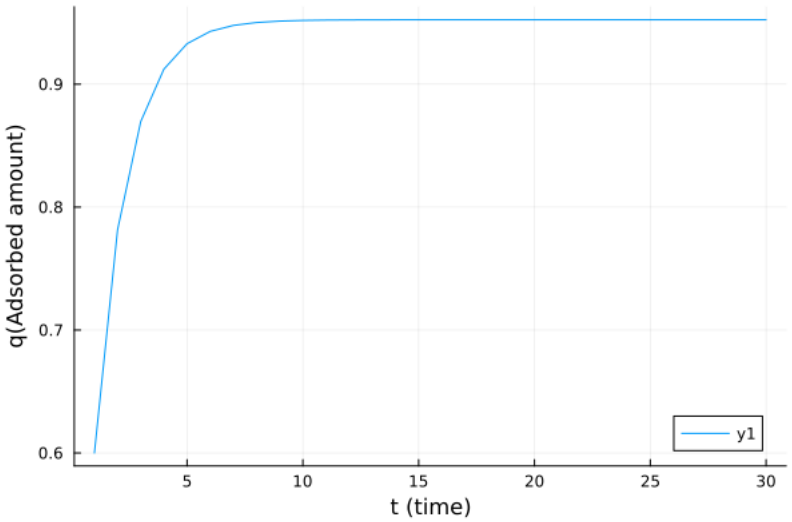

Figure 1: Underlying data of Langmuir Adsorption Model

Using this data as our training data we proceed to train a neural ODE with two optimizers: ADAM with a learning rate of 0.01 and Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm or BFGS. We use two optimizers because ADAM is good for initial loss reduction, but it cannot converge to the solution, therefore, we use BFGS to capitalize on the minimization to fit the model. The difference between ADAM-only optimization and ADAM+BFGS optimization can be seen in the figures below.

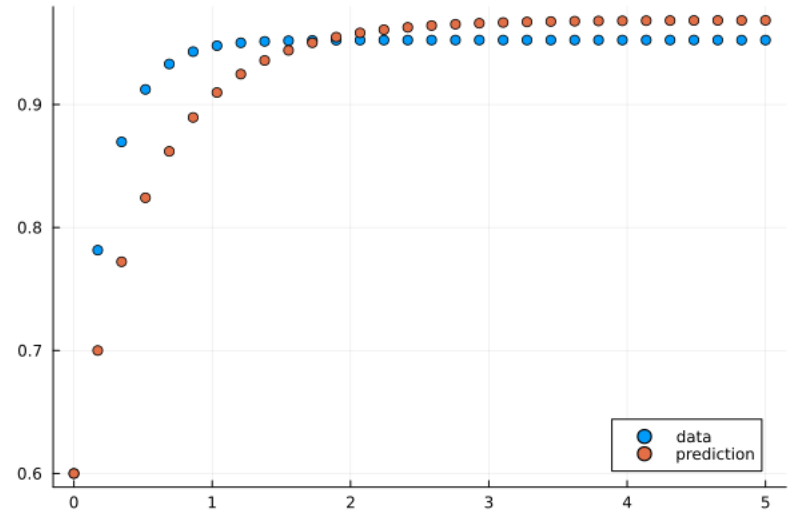

Figure 2: Model fit with ADAM

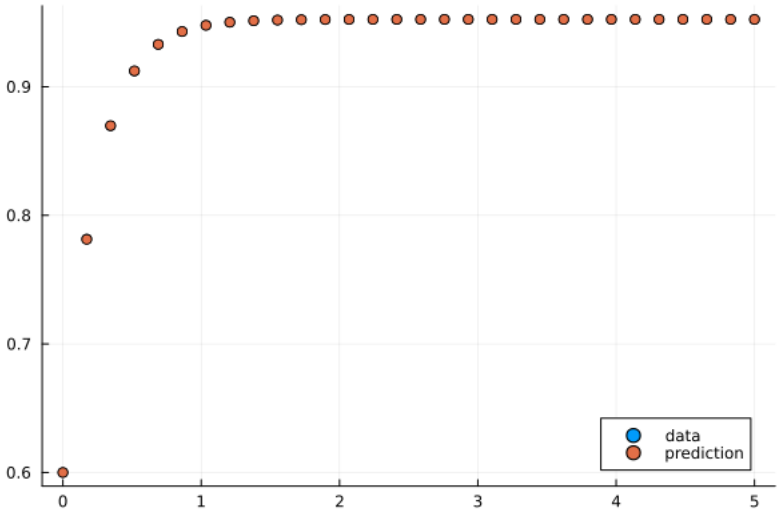

Figure 3: Model Fit with ADAM+BFGS

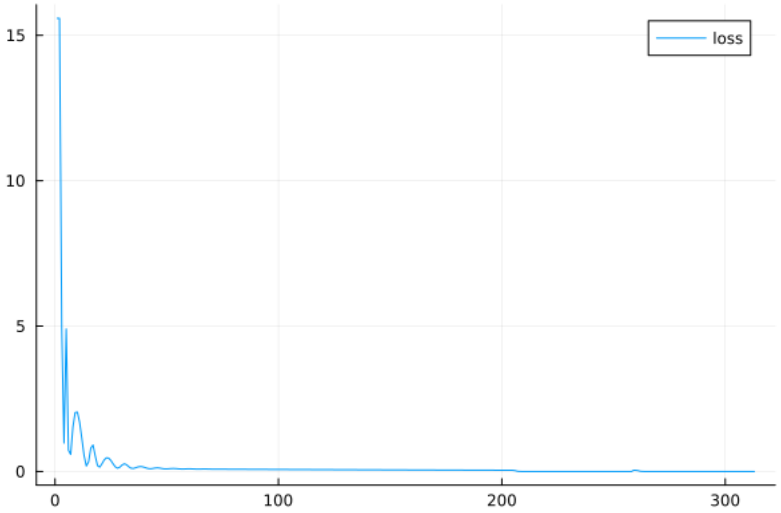

Figure 4: loss curve

4 Conclusion

The project successfully implemented the Langmuir adsorption model using a neuralODE. We replace the Langmuir ODE with a neural network and learn the dynamics from the generated data. Traditionally, an ODE needs to be explicitly written down to solve it numerically. However, physical systems have constraints that need to be considered, making the system more complex to solve. In contrast, neural ODE offers a flexible approach in which we do not require an explicit model. Systems that are chaotic, non-linear, and sensitive to subtle change can be solved as a neural network since they can capture these changes implicitly. This project highlights the trade-off between the approaches: Traditional methods are interpretable and simple, whereas, NeuralODE provides a data-driven method for complex scenarios.