Modelling chemical reaction networks using neural differential equations

A thesis presented for the Master's degree in Data Science

Chiara Thöni
s1018813

Supervisor:
Dr Tal Kachman

Second reader:
Dr Yuliya Shapovalova

June 27, 2024

Radboud University

Abstract

Nature is ever-changing and always evolving dynamically, displaying chemical transformations in chemistry and material changes in physics. Many of these changes can be prescribed mathematically by sets of differential equations describing the aforementioned dynamical changes. This work focuses on chemical reaction network theory, where ordinary differential equations are used to model the temporal change of chemical species concentration. The functional form of these ordinary differential equations systems is derived from an empirical model of the chemical reaction network. However, the empirical model can be incomplete, for example, when there are hidden interactions between the chemical species, not easily or explicitly quantifiable. As a consequence, there can be large discrepancies between the experimental results and the predictions made by the solution of the ordinary differential equations. Elucidating such methodologies, shedding light on hidden insights, and AI-driven insightful understanding is the goal of this research.

In this thesis, I aim to account for such discrepancies by combining the theoretical vector field with a neural network in the form of a neural ordinary differential equation. With this approach, the neural network acts as a correction term, capturing the dynamics that are not included in the theoretical model. I test the predictive performance of the neural ordinary differential equations on synthetic as well as experimental datasets. When provided with a complete vector field, the model can fit all datasets, even in noisy settings, including open systems. Furthermore, the method can find missing reactions within all aperiodic chemical reaction networks considered. The neural network output, which illustrates the magnitude and temporal signature of the differences between the theoretical model and the experimental data, can be used to reason about the structure of the chemical reaction network.

Contents

Acronyms

CRN	chemical reaction network. 1–3, 8, 9, 11–13, 16, 17, 27, 33, 34, 36, 38
CRNN	chemical reaction neural network. 11
DE	differential equation. 5, 6
IVP	initial value problem. 5, 7, 13, 17
LSTM	long short-term memory. vi, 13, 14, 23–25, 34–38
MLP	multilayer perceptron. 13, 14, 21–25, 27, 36, 37
MSE	mean squared error. 19
ODE	ordinary differential equation. vi, 1, 3, 5, 7–14, 16, 17, 21–28, 33, 35, 36, 38, 58–67
PES	potential energy surface. 8
PINN	physics-informed neural network. 7, 8, 11, 12
SINDy	sparse identification of nonlinear dynamics. 11, 12
UDE	universal differential equation. 12

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

This document has been written with two goals in mind. First, I highlight the flexibility of neural ordinary differential equations (ODEs) in chemical reaction network (CRN) modelling on real-world datasets. In particular, I minimize the differences between the model predictions and the experimental measurements by combining a vector field based on chemical theory with a neural network. Second, I demonstrate the insights provided by the model, where the separate predictions of the neural network can be used to reason about the structure of the CRN.

1.1.1 Problem definition

ODEs are widely used to model dynamical systems. Within the chemical domain, ODEs are used to describe the real-world behaviour of CRNs. To do so, the system of ODEs is derived by combining empirical knowledge regarding the structure of the CRN with the law of mass-action [1, 2]. The resulting system of differential equations is used to predict the concentrations of the chemical species over time. The parameters of the ODEs are tuned to maximize the similarity between the model predictions and experimental measurements [3, 4].

Despite the tuning process, it is not guaranteed that the system of ODEs can perfectly generate their predictions of the experimental data because of confounding effects, such as measurement noise, incomplete modelling, missing data and issues with the experimental design [5]. Furthermore, as the ODEs directly follow from the chemical-informed design of a CRN, the differential equations cannot describe any dynamical changes not included in the theoretical model. This limits the model's performance when the CRN is misspecified, for example, if the model does not include all interactions between the species involved [6]. In addition, the modelling performance can deteriorate if the assumptions for the application of the law of mass action are violated, which could be the case if the real-world chemical system is open or not well-mixed [7].

1.1.2 Contributions

In this work, I create and implement a method to correct for the discrepancies described above using neural ODEs: a combination of dynamical systems modelling and deep learning [8]. Neural ODEs are inspired by the advances in deep residual learning and normalizing flows, which build complex transformations by applying a sequence of smaller transformations [9, 10]. These smaller transformations are represented by a neural network, a mathematical model that can be trained to approximate any function [11, 12, p. 233]. In the limit, the composition of small neural transformations behaves like a differential equation. Yet, in contrast to a traditional differential equation with a predefined vector field, the neural ODE takes form during the training process to minimize the differences between the modelled and experimental data [13].

A note on the scale Because the neural ODEs are used to learn a smaller transformation that is applied in a sequence, the models can be very small. The models that are used in this work have a number of trainable parameters in the order of tens or hundreds. This is very different from the large-scale trend that is fuelled by the latest advances in deep learning, where the number of parameters goes towards the order of billions [14].

Comparison with related work The possibility of solving chemical kinetics using neural ODEs has been explored before. Yet, the approaches introduced in previous work require that the number of reactions is known a priori [15]. Alternatively, the method focuses on the thermodynamical state of the reactions instead of the concentrations of the species [16]. In contrast, this work aims to identify the differences between the modelled and experimental concentrations without making any assumptions about the reactions within the CRN. To this end, we take inspiration from Rackauckas et al. and augment the existing theoretical vector field with a neural network component [17]. This way, the neural network acts as a correction term: it captures the trends within the experimental data that are not explained by the theoretical model [13]. The proposed method, which is summarized in Figure 1.1, does not require that the concentrations of all species involved in the CRN are measured.

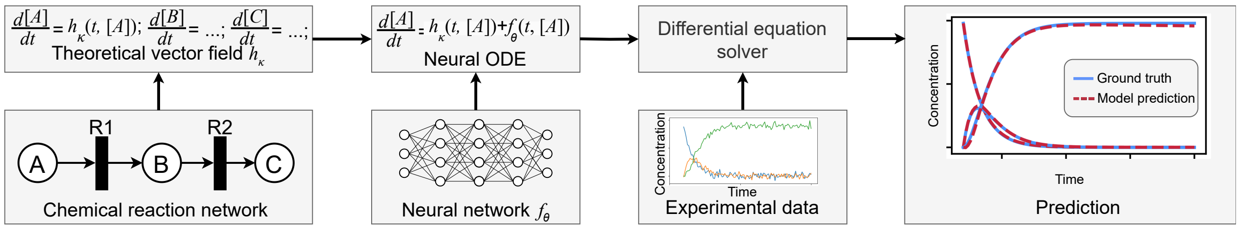

$\frac{d[A]}{dt} = h_\kappa(t, [A]); \frac{d[B]}{dt} = \ldots; \frac{d[C]}{dt} = \ldots;$
Theoretical vector field h_κ
$\frac{d[A]}{dt} = h_\kappa(t, [A]) + f_\theta(t, [A])$
Neural ODE
Differential equation solver
R1
R2
A
B
C
Chemical reaction network
Neural network f_θ
Concentration
Time
Experimental data
Concentration
Ground truth
Model prediction
Time
Prediction

Figure 1.1: **A summary of the model that is used to predict the concentrations of the chemical species over time.** The model combines the theoretical vector field h_κ and a neural network f_θ in a neural ODE to minimize the differences between the model predictions and the experimental data.

The dynamics of the species that are missing from the experimental data are predicted using just the theoretical vector field. Next to minimizing the difference between the theoretical predictions and the experimental data, the proposed model provides an insight into the connectivity of the CRN. The contribution of the network can be calculated separately for each chemical species that is measured. This contribution illustrates the magnitude and temporal signature of the differences between the theoretical model and the experimental data.

Challenges While neural ODEs form a promising approach for solving chemical dynamics, it is challenging to fit the models on stiff systems of differential equations [18]. Regions of strongly increasing or decreasing dynamics, which often occur within chemical kinetics, cause numerical instabilities for explicit ODE solvers [19]. While it is common to counter the stiffness of the dynamics by using implicit solvers such as Kværnø's $^5/_4$ method [20], I also change the temporal scale of the data to reduce the total number of steps that are taken during the solving process. Furthermore, the models were found to frequently get stuck within local minima during training. To counter this phenomenon, the models are trained with a cyclic learning rate scheduler.

Experiments To test the methods, I assess the modelling performance by applying the method to various CRNs. I first consider three synthetic datasets of small reaction networks. The controlled Secondly, I consider two experimental datasets that have been collected in Harmsel et al. This work proposes a chemical oscillator that is designed using small organic molecules. The chemical reaction network consists of 5 reactions and involves 7 species. I use the data from the single-pulse experiment as well as the measured series of sustained oscillations. The model performance is tested by providing it with the theoretical vector field described in [21]. Furthermore, by adding a small influx to the data or deliberately leaving out reactions from the CRN, I also test the model's performance on open systems and incomplete vector fields.

Relevance The results found with the experiments described above offer two insights. Firstly, the neural network contributions can be used to verify the correctness of the theoretical model under noisy measurements. Secondly, the contributions can be used to account for and uncover any dynamics that are not included in the theoretical model. In future work, these contributions may be translated into concrete adjustments to the theoretical vector field by using symbolic regression [22]. In addition, the neural ODE can be extended to calibrate the system of ODEs, rather than improving an existing, calibrated set of differential equations.

1.2 Thesis outline

While writing this thesis, I assume that the reader is familiar with the basics of (recurrent) neural networks. Familiarity with solving differential equations is helpful, but not required.

This topic will be introduced in Section 2.1. Furthermore, Section 2.2 provides a brief introduction to chemical network theory to facilitate readers who do not have a chemical background. Chapter 3 aims to highlight the similarities and differences between the proposed work and existing methods. The chapters that follow focus on just the proposed work by describing the Methods, Results, Discussion and Conclusion.

Code The code used to perform the experiments is available on GitHub.

`https://github.com/KachmanLab/ChemicalReactionNetworks`

The repository also contains the scripts that have been used to generate the figures that are included in this thesis.

Chapter 2

Background

2.1 Differential equations

2.1.1 Definition

Differential equations (DEs) describe the derivative of an unknown function ϕ with respect to one or more variables [23, p. 990]. DEs are commonly used in natural sciences to express a relation between quantities. The latter claim also holds in this work, where the relation of interest is described by the change in concentration of a chemical species over time. As the chemical concentrations are merely derived with respect to the time t, the DEs of interest can be referred to as ODEs: the subset of DEs that are derived with respect to a single variable [23, p. 991].

A first order ODE, which has the generic form shown in Equation 2.1 represents a slope field.

$$\frac{dy}{dx} = f(x, y) \tag{2.1}$$

That is, for each point (x, y) within the domain of f, the slope of the function y is described with $f(x, y)$ [23, p. 999]. Yet, just knowing Equation 2.1 is not sufficient to learn something about the shape of y. In particular, the solution to the differential equation depends on its initial value $y_0 = y(x_0)$. As a consequence, to model y, the so-called initial value problem (IVP) should be solved.

2.1.2 The initial value problem

The initial value problem consists of two components, a differential equation ${}^{dy}/_{dx}$ and a prescribed value of the solution to the DE at a particular point y_0 (the initial point) [23, p. 152]. While the solution to the IVP equals $y = \int f(x, y)\, dx$, systems of differential equations are often solved using numerical methods due to the unavailability of the antiderivative of f [23, p. 152]. As a consequence, the solution to the IVP is often approximated by discretizing the derivative. The simplest method to do so is the Euler discretization, presented in Equation 2.2 [24]. In the equation, n indicates the current state, $n + 1$ the next state and Δx the size of the state change with respect to x.

$$\frac{dy}{dx}(x_n) \approx \frac{y(x_{n+1}) - y(x_n)}{\Delta x} \tag{2.2}$$

Using this discretization, the vector field from Equation 2.1 can be rewritten to the definition provided in Equation 2.3. Using this equation, solving a differential equation can be interpreted as the repeated application of the product between the stepsize Δx and the slope f at x_n. This definition bears similarities with the learning process of residual deep learning architectures (Equation 2.4), which will be discussed in more detail in Section 2.1.4.

$$y(x_{n+1}) = y(x_n) + \Delta x \cdot f(x_n, y(x_n)) \tag{2.3}$$

2.1.3 Numerical methods for solving differential equations

According to Equation 2.3, the solution to the DE is formed by repeatedly incrementing an intermediate solution by the product of the slope at x_n and the stepsize Δx. Therefore, the magnitude of the step size is determinative of the final solution. A step size that is too large results in a suboptimal solution, with an error that accumulates at each step within the solver process. At the same time, a stepsize that is too small increases the computational complexity of the process. The number of steps needed to calculate the solution $N = |x - x_0|/\Delta x$. While a decrease of Δx is expected to lower the error of Euler's method proportionally to $N(\Delta x)^2 = |x - x_0| \cdot \Delta x$ [23, p. 1003], it leads to an increase in N. As a result, various numerical methods for solving DEs have been designed that aim to minimize the error without decreasing the step size. The family of Runge-Kutta methods, which are further improvements on the Euler method, form an example [25, 26]. Within the Runge-Kutta methods, one can make a distinction between implicit and explicit methods. Explicit methods, like the Euler discretization introduced before, calculate the state of the solution at time $n + 1$ given the DE applied to the current state at time n. In contrast, implicit methods find the solution y by solving a DE that involves both the current and the next state. Even though implicit numerical methods are harder to solve, they are more stable than explicit methods [13]. This is an advantage when dealing with the stiff systems that are described in the next section.

Stiffness and numerical stability Solving a system of differential equations in a stepwise fashion as described above introduces a limitation: the quality the solution of the DE depends on the interplay between the stepsize and the slope (see Equation 2.3). Whereas the previous section highlighted the influence of the stepsize, I now focus on the influence of slope. By definition, the magnitude of the slope is dependent on the location x_n. As a consequence, a system of differential equations may contain regions of strongly increasing or decreasing dynamics. The large differences between these regions introduce numerical instabilities into the solving process that can be countered by using implicit solvers with adaptive stepsizes [13]. Stiffness is a very important notion for this thesis as it is an often-occurring phenomenon within chemical kinetics [19]. As a consequence, the choice of solver and temporal scaling of the data plays a large role in the Methods.

2.1.4 Neural ordinary differential equations

Thus far, this section has focused on a system of differential equations with a predefined vector field $f(x, y)$. Yet, when considering a neural ODE, the vector field $f(x, y)$ is represented by a neural network f_θ with parameters θ [27, p. 16]. These neural ODEs, which are introduced by Chen et al. in [8], can be seen as the continuous limits of residual networks. To illustrate the previous claim, consider the residual learning process where each residual block of the network builds towards the sequence of transformations shown in Equation 2.4.

$$y_{t+1} = y_t + f_\theta(t, y_t) \tag{2.4}$$

In the equation, $t \in \{0, \ldots, T\}$ represents the index of the hidden state y, $f_\theta(t, \cdot)$ the t-th residual block of the network and θ the network parameters [28]. In the limit —if more residual blocks are added and smaller steps t are taken—the update step from Equation 2.4 goes to a differential equation that is specified by the neural network f_θ:

$$\frac{dy}{dt}(t) = f_\theta(t, y(t)) \tag{2.5}$$

The neural ODE from Equation 2.5 can mimic the behaviour of a residual neural network by taking an initial point $y(0)$ and applying the neural ODE for a time T to obtain the hidden state of the network h_T that could act as regression output or precede the final classification layer [27, p. 16].

$$h_T = y(0) + \int_0^T f_\theta(t, y(t))\, dt \tag{2.6}$$

Neural ODEs are more flexible than predefined ODEs due to the possibility to train the neural network f_θ. The neural ODE training is based on the solution to the system of differential equations. The partially trained network f_θ is used in combination with the initial point to solve the IVP. The gradients for the parameter updates can be calculated by backpropagating the error of the solution to the IVP with respect to the "ground truth" solution [13].

2.1.5 Universal differential equations

While the neural ODE presented in Equation 2.5 forms a flexible vector field, it disregards the domain knowledge regarding the problem at hand. To embed physical knowledge into the learning process of the neural ODE, Rackauckas et al. propose to combine the neural network output with a predefined vector field h [17]. The resulting vector field is presented in Equation 2.7.

$$\frac{dy}{dt}(t_{n+1}) = h(t, y(t_n)) + f_\theta(t, y(t_n)) \tag{2.7}$$

In the new neural ODE definition, the network can act as a correction term and is able to account for the trends within the data that cannot be explained by the theoretical model alone [27, p. 25]. Even though this approach seems related to physics-informed neural networks (PINNs), there is a fundamental difference between the two methods: Neural ODEs use a neural network to specify the ODE. In contrast, a PINN is used to find a solution to a predefined ODE under constraints based on domain knowledge.

2.2 Chemical reaction networks

The terms "theoretical vector field" or "vector field based on chemical theory" are frequently used within this thesis. This section describes the derivation of such vector fields, by giving the the recipe for constructing a vector field based on a simple CRN.

2.2.1 Species, complexes & reactions

Formally, a CRN is a triple $\{\mathcal{S}, \mathcal{C}, \mathcal{R}\}$ of species ($\mathcal{S}$), complexes ($\mathcal{C}$) and reactions ($\mathcal{R}$) that describes the relationships between (chemical) reactions [7]. To define the three sets above, consider the reaction that describes the oxidation of hydrogen:

$$2\ H_2 + O_2 \rightarrow 2\ H_2O \tag{R_1}$$

In this process, two units of hydrogen (H) combine with a single unit of oxygen (O_2) to form two units of water (H_2O). The three different compounds that take part in the reaction are referred to as the species, $\mathcal{S} = \{H_2,\ O_2,\ H_2O\}$ with $|\mathcal{S}| = n_s$. The species are preceded by the non-negative stoichiometry coefficients a (a vector of respectively 2, 1 and 2). In R_1, the species on the left-hand side of the arrow are considered the reactants, whereas the species on the right-hand side are considered the *products*. Using these terms, a complex i can be defined as an additive combination of species and coefficients $\sum_{j=0}^{n_s} a_{ij} S_j$, such that it corresponds to either the reactants or products of a reaction [7]. Thus, the complexes of R_1 are $\mathcal{C} = \{2\ H_2 + O_2,\ 2\ H_2O\}$ with $|\mathcal{C}| = n_c$. The graphical representation of the example CRN that consists out of $\mathcal{R} = \{R_1\}$ is shown in Figure 2.1.

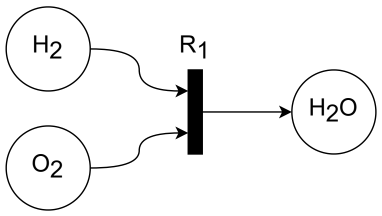


Figure 2.1: The graphical representation of the example CRN that describes the oxidation of hydrogen.

A model of a CRN can be constructed in both a manual and a data-driven fashion. The manual construction of networks, which is based on heuristics and chemical theory, is limited by the size of the networks and thus the effort that is required to formalize the model. Alternatively, CRNs can be constructed by analyzing the potential energy surface of the reactions or by using predefined rules, templates and filters [6]. These approaches come with drawbacks. As the use of templates limits the ability to discover new reaction mechanisms in particular, most CRNs that have been formalized using the methods above are incomplete [6]. This incompleteness can be advantageous for large-scale networks (the incomplete CRNs can be seen as the pruned versions of the complete networks if merely irrelevant reactions are left out) [6]. However, the absence of reactions can also lead to problems when using the CRN for modelling purposes, as explained in the next section.

2.2.2 Dynamical modelling and mass-action kinetics

Using a CRN and the relations between species that it describes, one can build a dynamical model to determine the concentration of the compounds involved [7, 6]. More specifically, if the mixture of compounds is well-stirred and the number of compounds is sufficiently high, one can assume that the probability of the reactions $\mathcal{R}$ to occur is distributed according to an exponential probability distribution [7]. As such, the rate r_i of each reaction $R_i \in \mathcal{R}$ can be described according to the mass-action kinetics shown in Equation 2.8 [1]:

$$r_i(\boldsymbol{x}) = k_i \prod_{j=1}^{n_s} x_j^{a_{ij}} \tag{2.8}$$

where k_i is the reaction rate coefficient, which can be obtained by experiment or simulation [6]. x_j is the concentration of species j and a_{ij} is the stoichiometric coefficient of species j in reaction R_i [7]. Equation 2.8 can also be interpreted as the propensity function of reaction R_i [29].

Using the complexes $\mathcal{C}$, one can determine the state change of each reaction. That is, if a reaction R_i can be described by the complexes $Ci \to C'_i$, where for R_1 it holds that $C_i = 2\ H_2 + O_2$ and $C'_i = 2\ H_2O$, the state change can be summarized by $a_i \to a'_i$. Here, a_i is the vector of the stoichiometry of the complexes grouped by the species, i.e. for R_1, $a_i = (2, 1, 0)$ and $a'_i = (0, 0, 2)$ [7]. Using the state change, the rate of the change in concentration that is brought about by R_j is defined as the state change times the rate r_i.

$$\frac{dx}{dt} = (a'_i - a_i) \cdot k_i \prod_{j=1}^{n_s} x_j^{a_{ij}} \tag{2.9}$$

For CRNs with multiple reactions, the rate of change in the total concentration of a species is provided by the sum over all reactions $R_i \in \mathcal{R}$. Thus, returning to R_1 a final time, if the reaction rate is assumed to be k_1, then the rate of the change in concentration $[\cdot]$ of the three species can be described as a set of ODEs (Equation 2.10). In this work, sets of ODEs that are based on chemical theory will be referred to as h_κ. In h_κ, the subscript κ refers to the reaction rates, the parameters of the vector field. As the construction of the system of ODEs depends on the theoretical model of the CRN, an incomplete theoretical model will be translated into an incomplete vector field.

$$\begin{aligned}\frac{d[O_2]}{dt} &= -k_1[O_2] \\ \frac{d[H_2]}{dt} &= -2k_1[H_2]^2 \\ \frac{d[H_2O]}{dt} &= 2k_1[H_2]^2[O_2]\end{aligned} \tag{2.10}$$

Dynamic model calibration The system of ODEs that is used to describe the change in concentration over time is a function of the reaction rates k. These reaction rates are unknown and not measurable. Instead, they can be estimated using optimization methods such as local search and stochastic global optimization metaheuristics [3, 30]. Here, the model is tasked to minimize the distance between the predictions made with the vector field and the experimentally measured data.

Open and closed systems The change in concentrations as described in Equation 2.10 is based on the law of conservation of mass. Regardless of the change in concentrations, the total mass of the species within the system remains constant. The law of the conservation of mass is based on the assumption that the system is closed: there is no exchange of species between the system and its surroundings. If there is a flux of mass into or out of the system, the system is considered open [7]. In the latter case, the vector field based on the law of mass action is a suboptimal predictor of the concentration of the species over time.

2.2.3 Opportunities for neural ordinary differential equations

The limitations of the dynamical models that have been introduced in the previous section form an opportunity for neural ODEs. In particular, the neural component of the neural ODE can be trained to account for the incompleteness of the vector field. Furthermore, the neural ODE is also expected to pick up any indications towards an open system, which can be translated to the addition or subtraction of a constant concentration per Δt. The exact change in concentration that is added by the neural network can be assessed by calculating by only applying f_θ to the concentrations at each timepoint t. The opportunities for neural ODEs can be leveraged using other machine learning techniques as well. A summary of the proposals that are based on a slightly different approach is provided in the next chapter.

Chapter 3

Related work

The combination of dynamical systems modelling and machine learning has been considered in previous work. The related work can be divided into three categories, focussing on neural ODEs, PINNs or sparse identification of nonlinear dynamics (SINDy). In contrast to the thesis, most work does not actively focus on detecting hidden insights. Rather, it focuses on accounting for stiff equations and calibrating predefined vector fields.

3.1 Neural ordinary differential equations

The possibility of solving chemical kinetics using neural ODEs has already been explored by Ji and Deng [15]. Their proposed chemical reaction neural network (CRNN) can be trained to calibrate the model of a CRN. However, to do so, the number of reactions within the CRN needs to be known a priori. Using this information, the authors propose an architecture with a single hidden layer. Because the number of hidden neurons corresponds to the number of reactions in the CRN, the neural network yields weights and biases that are physically interpretable as stochiometric coefficients and reaction rates. By linking the neural network architecture to the theoretical model, CRNN can not account for any dynamics not covered by the theoretical model [15]. Owoyele and Pal [16] consider a slightly different objective function and use ChemNODE, a more elaborate type of neural ODE, to solve thermochemical kinetics. ChemNODE is not based on a theoretical vector field, making the model more flexible when estimating missing dynamics. Instead, the model trains a unique neural network for each species to estimate its thermochemical state. As a result, the output and architecture of ChemNODE cannot be used directly to analyze the structure of the CRN. Still, the results found in this work suggest that neural ODEs can fit the stiff dynamics of chemical kinetics.

3.2 Physics-informed neural networks

Various authors propose to model chemical dynamics using PINNs. However, just like the neural ODEs, the PINNs are challenged by the stiffness that comes with the chemical kinetics [19]. Ji et al. [31] account for this stiffness by embedding the theoretical vector field as soft constraints within the loss function of the PINN [32]. Although this method allows for the accurate prediction of the chemical kinetics within stiff systems, the PINNs can not compensate for any misspecifications within the theoretical vector field due to their inflexible definition. Podina, Eastman, and Kohandel [33] alleviate the need for a correctly defined vector field within the PINNs by introducing the universal PINN (UPINN): a combination of a neural ODE with a PINN loss. Here, the authors also use a UDE to approximate unknown components of the theoretical vector field. Next, AI Feynman, a physics-inspired method for symbolic regression, is used to translate the neural network output into a concrete vector field [34]. While UPINNs outperform UDEs for data with high noise levels or sparsity, the performance of UPINNs on chemical systems or stiff systems, in general, has not been investigated [33]. Lastly, the authors state that the estimations of the hidden reactions are not unique given the trajectory of training data. To contrast the approach used in this thesis with the work on (U)PINNs, I aim to investigate the opportunities for machine learning in the context of CRNs in a simpler form. By focussing on just the ODEs, I build upon a basis that does not embed any assumptions about the dynamics of the chemical reaction into the loss function, which makes it easier to train the models.

3.3 Sparse identification of nonlinear dynamics

By making use of the information contained in the theoretical model of the CRN, one can perform regression to create a symbolic expression of an unknown differential equation [35, 36, 27]. This approach has been extended by Hoffmann, Fröhner, and Noé [37] with the introduction of reactive SINDy, which is used to estimate reaction rate equations from concentrational data. In contrast to the majority of models that have been introduced before, reactive SINDy can account for missing dynamics within the theoretical vector field by making a new combination of ansatz functions that each represent a single reaction. As a consequence, reactive SINDy does not make any assumptions about the CRN. On the flip side, the model does not use all the available domain knowledge, for example by combining the ansatz functions that are already included in the theoretical model.

Chapter 4

Methods

The methods of this thesis are twofold. First, I consider the experiments that are performed using four synthetic datasets. These analyses serve as a proof of concept before applying the neural ODE to the real-world data collected in [21]. Before describing both datasets, I introduce the general approach used to model the chemical concentrations over time.

4.1 Chemical reaction networks and neural ODEs

As introduced in the Background, the concentration of chemical species can be modelled by solving the IVP constrained by a system of ODEs. Recall that, when vector fields based on CRNs are considered, then h represents the vector field based on the mass-action dynamics of the CRN and κ the vector of reaction rate coefficients $\boldsymbol{k}$. To predict the concentration of the chemical species over time, I solve the IVP. More specifically, if $\boldsymbol{y_0}$ describes the initial concentrations of the chemical species, then the concentrations over time are described by a function y that is a solution to the neural ODE and satisfies $\boldsymbol{y(t_0)} = \boldsymbol{y_0}$. The system of ODE that is used in this work is formed by the addition of the theoretical vector field h_κ and the neural network f_θ, as shown in Equation 4.1.

$$\frac{dy}{dt}(t_{n+1}) = h_\kappa(t, y(t_n)) + f_\theta(t, y(t_n)) \tag{4.1}$$

4.1.1 Neural network architectures

The neural network architectures used in this work are variants on a multilayer perceptron (MLP) and a long short-term memory (LSTM) network. The number of inputs and outputs of both architectures is equal to the number of species that are included in the data. The models are implemented using JAX [38, version 0.4.14] and Equinox [39, version 0.11.1].

Selecting the optimal architecture I test a collection of neural network specifications to compose the final architectures for the MLP and LSTM. In particular, I optimize the network architectures for a single problem domain, the synthetic missing dynamics dataset, using Weights and Biases [40]. I explore the effect of using different network sizes (1, 2, 4 and 8 hidden layers of 16, 32 or 64 neurons), optimizers (Adabelief, stochastic gradient descent) and activation functions (ReLU, leaky ReLU, GeLU and sigmoid). I only used the synthetic missing reactions dataset to tune the architectures and assess their generalizability. The selected architectures are relatively small: the MLP consists of two hidden layers with a width of 64 neurons each and the LSTM is formed with a LSTM cell with a 32-dimensional hidden state followed by a linear layer of 32 neurons. The MLP uses sigmoid activations while the LSTM uses an identity activation before the application of the linear layer. Both networks use the identity function as the final activation to leave their output unconstrained. At each step within the equation-solving process, the neural network is presented with the measurements at a single time point instead of the complete time series. As the LSTM was found to outperform the MLP during the experiments with synthetic data, I only use the LSTM when modelling the experimental data.

Attention layers Within the neural ODE, I apply an attention layer to the temporal dimension of the data. To do so, the solving procedure should be changed so that the neural ODE has access to the entire data point of shape (T, n_s), where T is the total number of time points. To prevent the model from looking into the future, this would not be possible in the real world either, I use a lower-diagonal attention mask, resulting in causal attention. Figure 4.1 provides an example of this mask. The mask has a shape of (T, T) as self-attention is considered.

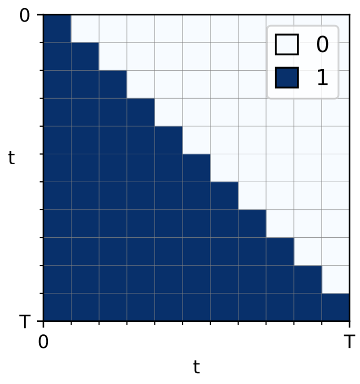


Figure 4.1: A visualization of the attention mask.

I only apply attention to the LSTM-based neural ODEs as their recurrence leverages the presentation of multiple time points.

4.1.2 Neural ODE training and inference

Using Equation 4.1 and the initial concentrations of each species, I train f_θ during the equation solve to match the experimental data at the measurement times t (Algorithm 1). In Algorithm 1, t_s refers to the time regulating integration process of the solver. Even though the solver steps t_s are not guaranteed to match t, the predicted concentrations can be saved at each time t during the process to make the final prediction. The network parameters θ are updated with respect to a mean squared error (MSE) loss for the species that have been measured. The data provided by Harmsel et al. consist of the mean and standard deviation calculated over two experimental measurements. I generate artificial training, validation and test data by drawing from a normal distribution that is parameterized with these statistics. The neural network has been trained on $N = 1000$ train samples using an Adabelief optimizer with a learning rate of 6×10^{-3} ([41, 38], version 0.1.7). I did not apply dropout as this was found to cause instabilities during the differential equation solve, resulting in many rejected solver steps. For the oscillating data I first train the neural ODE on the first 10% of each time series to prevent getting stuck in a local minimum [13].

I calculate the neural network contribution ${}^{dy_{f_\theta}}/_{dt}$ given the prediction of the complete differential equation ${}^{dy}/_{dt}$, as shown in Equation 4.2. The approach used to calculate the neural network contribution is presented in Algorithm 2.

$$\frac{d\hat{\boldsymbol{y}}_{f_\theta}}{dt}(t_{n+1}) = f_\theta(t_n, \hat{\boldsymbol{y}}(t_n)) \tag{4.2}$$

Training and inference have been done on the CPU of my computer, which is an Intel Core i7-10750H CPU @ 2.60 GHz.

Algorithm 1 Neural ODE training

```
Input
  $N_{epochs}$: the number of epochs
  $T$: the time of the last measurement
  $\mathcal{D}$: the dataset of shape $(N, T, n_s)$.
  $f_\theta$: the neural network with parameters $\theta$
  $h_{\boldsymbol{\kappa}}$: the theoretical vector field
Output
  The trained model $f_\theta$

  for epoch in $N_{epochs}$ do
    for $\boldsymbol{x}, \boldsymbol{y}$ in $\mathcal{D}$ do
      $t_s \leftarrow t_0$
      while $t_s < T$ do
        if $t_s$ is $t_0$ then
          $\hat{\boldsymbol{y}}_{t_s} \leftarrow \boldsymbol{y}_0$
        end if
        $\Delta t \leftarrow$ ODEsolver$(\hat{\boldsymbol{y}}_{t_s}, t_s)$
        $\hat{\boldsymbol{y}}_{t_s+\Delta t} \leftarrow h_{\boldsymbol{\kappa}}(\hat{\boldsymbol{y}}_{t_s},\ \Delta t) + f_\theta(\hat{\boldsymbol{y}}_{t_s},\ \Delta t)$
        $t_s \leftarrow t_s + \Delta t$
      end while
      $\mathcal{L} = \text{MSE}\left(\hat{\boldsymbol{y}}_{t_0:T}, \boldsymbol{y}\right)$
      Update $\theta$ by taking a gradient descent step on $\mathcal{L}$
    end for
  end for
```

Algorithm 2 Neural network contribution

```
Input
  $\hat{\boldsymbol{y}}$: the concentrations predicted by the neural ODE.
  $f_\theta$: the neural network with parameters $\theta$
Output
  $c_{f_\theta}$: the neural network contribution

  for $\hat{\boldsymbol{y}}_t$ in $\hat{\boldsymbol{y}}$ do
      $c_{f_\theta,t} \leftarrow f_\theta(\hat{\boldsymbol{y}}_t)$
  end for
```

4.2 Experiments with synthetic data

To generate the synthetic datasets, I construct four different chemical reaction networks: The bottleneck reaction, the (decaying) Brusselator, the open system and the missing reactions network. The reaction networks differ in the number of species and reactions that are involved. The reactions are briefly described in the next section. The details regarding the synthetic data generation, including the individual reactions, vector field specifications and parameter choices, can be found in Appendix B.

4.2.1 Synthetic dataset design

The bottleneck reaction network, consisting of three reactions (Appendix B.1.1), is chosen for its simplicity. The small number of interactions between the species makes the inspection of the neural network contribution relatively simple. As such, the bottleneck reaction is used to verify whether the neural ODE works as expected.

To test the model performance on a complex network, I selected the Brusselator [42]. The Brusselator is a well-known example of a network containing oscillating concentrations (Section B.1.2, [42]). The stability of the Brusselator is controlled by regulating the concentrations of species X and Y. I model a Brusselator with stable oscillations by setting $X = 1$ and $Y = 3$. A Brusselator with decaying oscillations is created by $X = 1$ and $Y = 1.7$. Fitting an oscillating concentration with noisy measurements is challenging as the noise may prevent the network from learning the oscillating behaviour. As a consequence, I consider it important to test these chemical reaction networks (CRNs).

The open system (Appendix B.1.3) and missing reactions networks (Appendix B.1.4) are designed to test the capability of the neural ODE of modelling dynamics that are not included in the theoretical vector field. Both chemical reaction networks are not overly complex. Just like with the bottleneck reaction, this facilitates the interpretation of the neural network contribution. Yet, the data produced with the open system reaction network includes an influx of $\frac{d[B]}{dt} = 0.2$ that is not included in the theoretical model. Alternatively, the generative vector field of the missing reactions network has an additional reaction $C \rightarrow D$ that is reflected in the data, but not in the theoretical model. I expect to retrieve the elements that are "missing" from the modelled vector field in the contribution of the neural network. It may be harder to produce an adequate fit on the missing reactions CRN than on the open system CRN as the former misses a variable, while the latter misses a constant value.

A note on vector fields When actively modelling a discrepancy between the generated data and the modelled vector field h_κ of the neural ODE, which is the case for the open system and missing reactions CRNs, I use a *generative* and a *modelled* vector field. In particular, the data are always generated by solving the IVP using the generative vector field. The predefined vector field h_κ of the neural ODE is described by the modelled vector field. If the generative and modelled vector fields are the same, meaning that the neural ODE has full information regarding the CRN, then I only provide the single, general vector field.

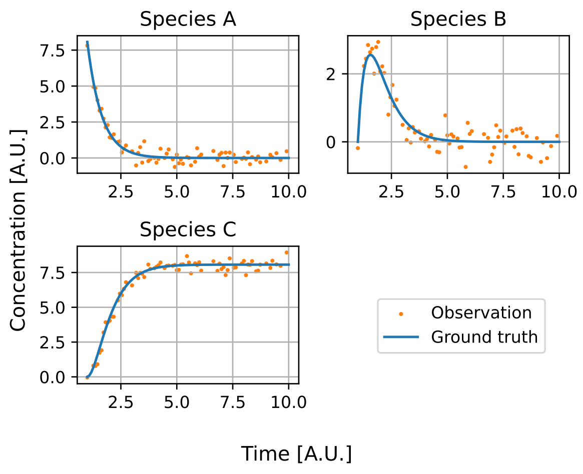
4.2.2 Modelling noisy measurements

Measurement noise The data are generated with two types of noise to mimic experimental measurements. To model noisy fluctuations around the measured concentrations I draw noise from a Gaussian with zero mean and standard deviation σ and a Poisson distribution with rate λ. Gaussian noise is a common form of noise in natural sciences. A Poisson process is commonly used to model shot noise, a common factor in instrumental analysis [43]. The initial concentrations y_0 are drawn from a Gaussian distribution with mean μ_{y_0} and standard deviation σ_{y_0}. The hyperparameters σ, λ, μ_{y_0} and σ_{y_0} depend on the vector field that is considered.

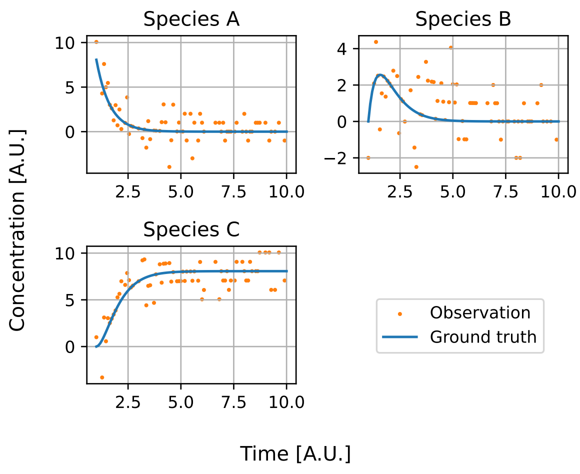
Temporal noise Not all T generated data points are provided to the neural ODE to model noisy measurement timings. Instead, the data points are sampled in a uniform manner with a rate $T_{sample} < T$.

Reaction rate noise The reaction rates $\boldsymbol{k}$ of each CRN are sampled from a normal distribution with mean μ_k and a standard deviation of 0.1. The mean of the reaction rates is set to $\mu_{k_1}, \mu_{k_2} = 1.5, 2$ for the bottleneck, open system and missing reactions CRNs. For the Brusselator reactions, all means μ_{k_i} for $i \in [1, 2, 3, 4]$ are set to 1.

The data are generated from time t_{min} until time t_{max}. The chemical specification, vector field and modelling parameters are provided for each dataset. A summary of the parameters used to generate the data can be found in Table B.1. This generative approach results in a set that has the shape $(N_{datapoints}, T_{sample}, n_s)$. An example data point is shown in Figure 4.2, which shows an example time series of the Bottleneck reaction network. Appendix B.2 provides the other datasets and noise types.



(a)



(b)

Figure 4.2: An example of synthetic data generated with the Bottleneck reaction network under Gaussian (a) and Poisson (b) noise. The noisy observations are plotted in orange, while the ground truth, as described by the chemical dynamics, is shown in blue. The three species involved in the reaction network are plotted separately.

4.3 Experments with real-world data

For the experimental data, I consider two experimental datasets that have been collected by [21]. This work proposes a chemical oscillator that is designed using small organic molecules. The chemical reaction network consists of four reactions and involves 7 species. The reactions include the autocatalytic Fmoc-piperidine (2) deprotection via dibenzofulvene (6); the N-methylpiperidine (5)-catalysed Fmoc-piperidine (2) deprotection; the fast inhibition via acetylation by p-nitrophenyl acetate (3); and the slow inhibition by phenyl acetate (4) that converts piperidine (1) to N-acetyl piperidine [21, p. 7]. this vector field is shown below [21]. The reaction rates $[k_{tr}, k_{ac}, k_{inh1}, k_{inh2}]$ are set to $[2.02 \times 10^{-2}, 6.02 \times 10^{-1}, 2.22 \times 10^{2}, 2.75 \times 10^{-3}]M^{-1}s^{-1}$. sv is the space velocity, which is estimated to be 1×10^{-4}. The change in N-methylpiperidine (5) is not included in the vector field as the concentrations of this species stay constant over time. The experimental system is open, where a controlled concentration of Fmoc-piperidine (2), p-nitrophenyl acetate (3) and phenyl acetate (4) enters the system. These concentrations, indicated with $[\cdot]_{in}$, are equal to $[0.1, 0.03, 1.8]M^{-1}s^{-1}$ for both datasets.

$$\begin{aligned}
\frac{d[1]}{dt} &= k_{tr}[5][2] + k_{ac}[1][2] - k_{inh1}[1][3] - k_{inh2}[1][4] - sv[1] \\
\frac{d[2]}{dt} &= -k_{tr}[5][2] - k_{ac}[1][2] + sv\left([2]_{in} - 2\right) \\
\frac{d[3]}{dt} &= -k_{inh1}[1][3] + sv([3]_{in} - [3]) \\
\frac{d[4]}{dt} &= -k_{inh2}[1][4] + sv([4]_{in} - [4]) \\
\frac{d[6]}{dt} &= k_{tr}[5][2] + k_{ac}[1][2] - sv[6] \\
\frac{d[7]}{dt} &= k_{inh1}[1][3] + k_{inh2}[1][4] - sv[7]
\end{aligned} \tag{4.3}$$

The stepsize dt is governed by the differential equation solver. I use Kværnø's $5/4$ method to account for the stiffness of the ODE ([20, 13], version 0.4.1). Furthermore, I bin the temporal measurements into 100 bins and change the temporal scale of the experimental data from seconds to hours (single-pulse) or days (oscillations) to reduce the total number of steps that are made by the solver.

The oscillating dataset that has been made available describes the absorbance rates instead of the concentrations. As the theoretical vector field considers concentrations, the data are scaled manually to match the order of magnitude described by the theoretical vector field.[1]

4.4 Modelling open systems

When modelling the experimental data, I assume that the data have been measured in a closed system. To test the capacity of the neural ODE to model open systems, I change the original vector field by subtracting 0.02M dibenzofulvene at each period dt. The adapted definition of the change in dibenzofulvene is shown in Equation 4.4. The constant, artificial reduction within the theoretical vector field yields a modelling problem that is equivalent to a constant influx of 0.02M dibenzofulvene in the data.

$$\frac{d[6]}{dt} = k_{tr}[5][2] + k_{ac}[1][2] - sv[6] - 0.02 \tag{4.4}$$

4.5 The identification of missing reactions

I use a similar modelling approach for the identification of missing reactions. However, instead of subtracting a constant concentration for each temporal interval, I remove a part

[1]The relationship between the absorption, concentration and path length is described by the Beer-Lambert law: $A = \varepsilon Cl$. Here, A is the absorbance, ε the extinction coefficient C the concentration and l the length of the light path. However, as ε and l were unavailable, the data have been shifted and scaled based on a minimization of the MSE between the data and predicted concentrations.

of the theoretical vector field. In particular, I modify the change in N-acetyl piperidine to the definition presented in Equation 4.5. This modification partially removes the pathway of the second inhibition reaction of the enzymatic oscillator. The rest of the theoretical vector field remains unaltered.

$$\frac{d[7]}{dt} = k_{inh1}[1][3] - sv[7] \tag{4.5}$$

Chapter 5

Results

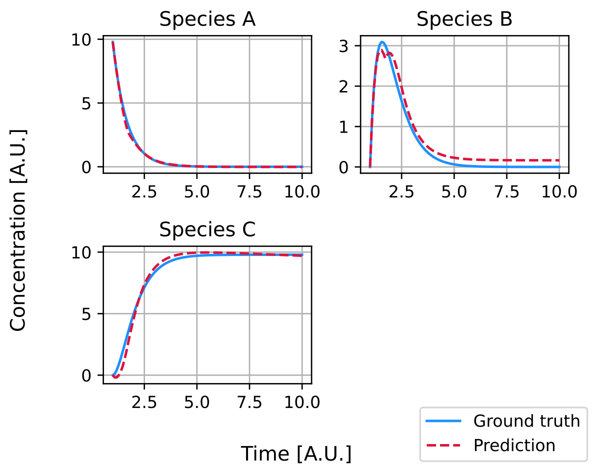
In this chapter, I present the results found with the experiments presented in the Methods. The results are discussed with the aim of the thesis in mind: I consider both the quality of the neural ODE fit and the insights provided by the separate neural network contribution. Furthermore, I will relate the observations made to chemical phenomena, such as the conservation of mass.

5.1 Synthetic data

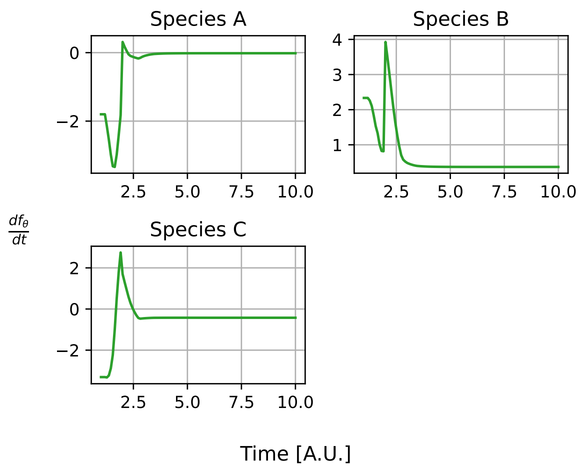
As the neural ODEs learn a good fit on nearly all synthetic datasets, I limit myself to presenting the most interesting results in this section: I highlight unexpected aspects of the predictions and focus on the cases where the modelling performance breaks down. In particular, I consider the performance of the MLP and LSTM on the reaction networks under Poisson noise, as this type of noise turned out to be the most difficult to model. Due to the good performance of the models on the synthetic data under Gaussian noise, I moved these results to the appendix to prevent this section from becoming too large. In particular, the predictions of the neural ODEs on the synthetic data are presented in Appendix B.3, where the results are grouped based on the modelled measurement noise, which can either come from a Gaussian (Appendix B.3.1) or Poisson distribution (Appendix B.3.2).

5.1.1 Bottleneck reaction network

The predictions of the MLP (Figure 5.1) and LSTM (Figure 5.2) are shown below. Even though both models provide a very accurate fit (Subfigures (a)), the neural network contributions of both models differ slightly. For species B and C, the contribution of the MLP is a lot larger during the initial stages ($t < 2.5$) of the reaction than the contribution of the LSTM. Towards the end of the reaction, the contribution of both networks has converged to 0.

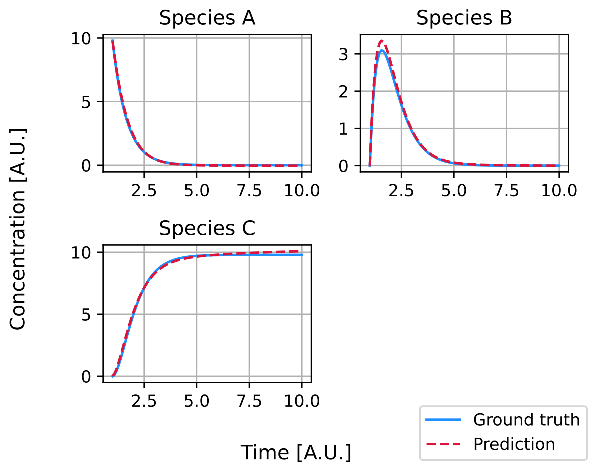


(a)

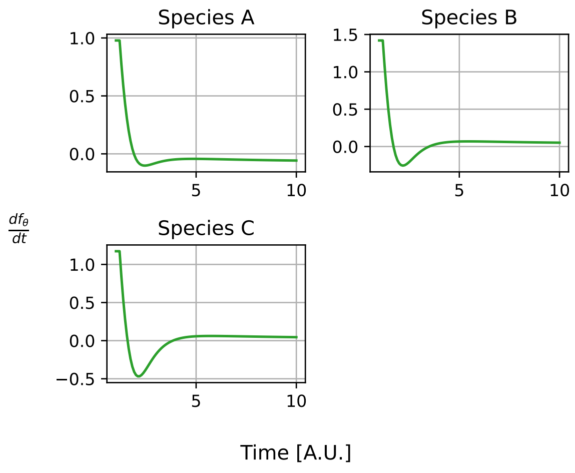


(b)

Figure 5.1: The modelling performance (a) and neural network contribution (b) of the neural ODE using a MLP on the four species within the bottleneck reaction network.



(a)

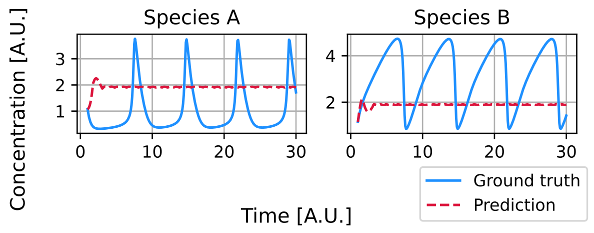


(b)

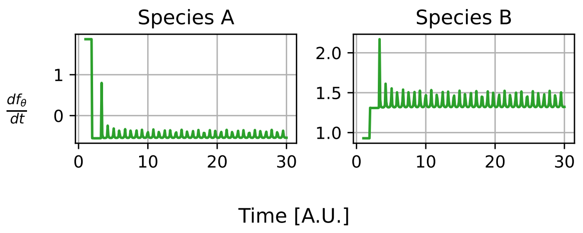
Figure 5.2: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM perceptron on the four species within the bottleneck reaction network.

5.1.2 Brusselator

The fit on the Brusselator, presented in Figures 5.3 and 5.4, is very interesting. The neural ODE using the MLP is not able to adequately fit the data. The MLP itself outputs a high-frequency spike train (Figure 5.3b). The fit based on the LSTM is a lot better, but not perfect. In particular, the neural network underestimates the amplitude of the oscillations by outputting a contribution that is too small (Figure 5.4b). Notice that the contribution of the LSTM of species A resembles the concentrations of species B flipped horizontally, and vice versa. This suggests that the contributions of the neural network are paramount for predicting the phase of the oscillations. In addition, this coupling indicates that the output of the neural ODE aims to adhere to the law of mass preservation.

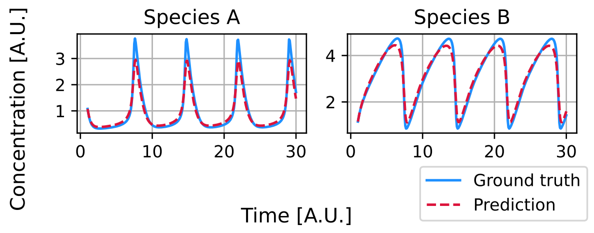


(a)

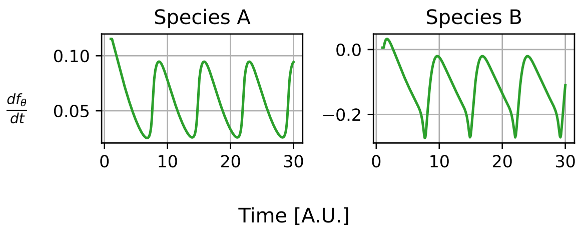


(b)

Figure 5.3: The modelling performance (a) and neural network contribution (b) of the neural ODE using a MLP on the four species within the Brusselator reaction network.



(a)

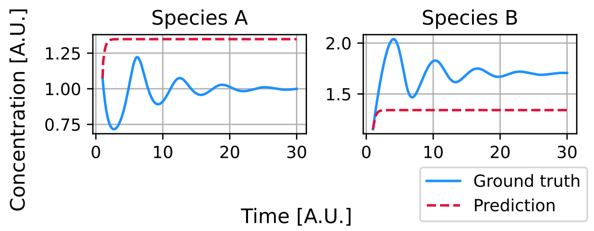


(b)

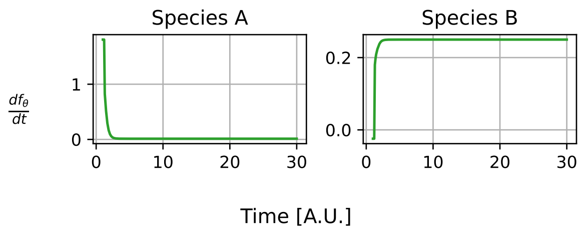
Figure 5.4: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the Brusselator reaction network.

5.1.3 Decaying Brusselator

The results for the decaying Brusselator are comparable to ones found for the Brusselator: the MLP-based neural ODE (Figure 5.5) is not able to fit the data, while the LSTM-based neural ODE (Figure 5.6) underestimates the amplitude of the oscillation. Again, the neural network contribution of the LSTM for one species resembles the flipped concentration of the other species, which has to do with the prediction of the correct phase.

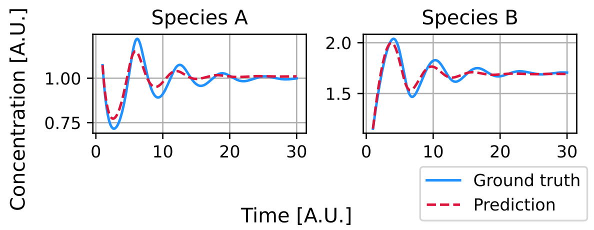


(a)

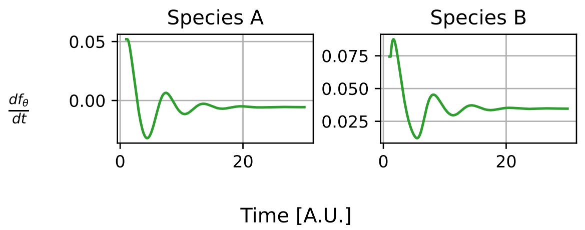


(b)

Figure 5.5: The modelling performance (a) and neural network contribution (b) of the neural ODE using a MLP on the four species within the decaying Brusselator reaction network.



(a)

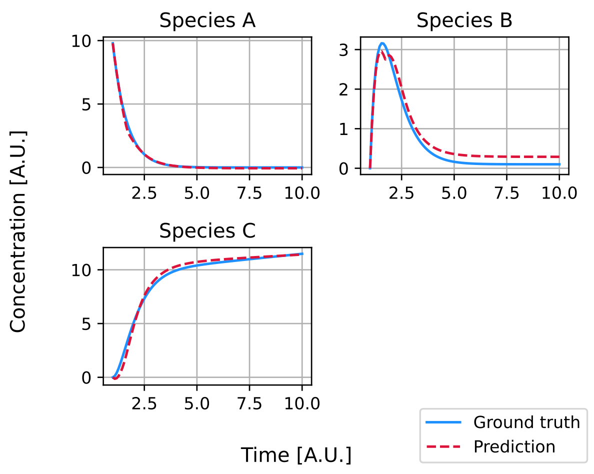


(b)

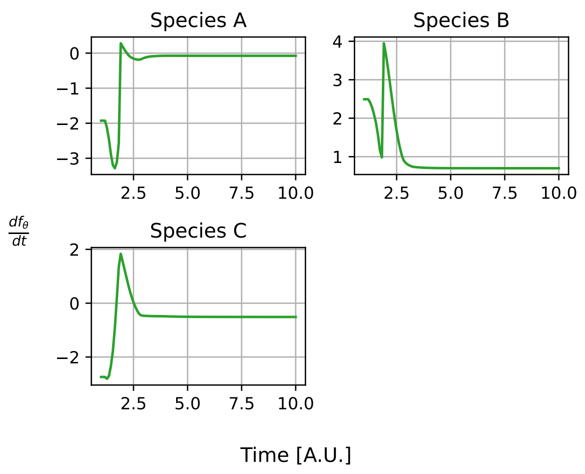
Figure 5.6: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the decaying Brusselator reaction network.

5.1.4 Open system

For the open system it is interesting to see whether the neural network has found the influx of $\frac{d[B]}{dt} = 0.2$. The presence of Poisson noise slightly worsens the fit for the neural ODE based on the MLP, as shown in Figure 5.7a. The neural network contributions of the MLP converge towards 0 for species A. However, for species B the contribution seems towards 0.7. This overestimation is countered by a negative contribution of species C. The contributions of the LSTM converge more clearly 0 for species C (Figure 5.8b). However, the network contribution displays a downward trend for all the species, instead of converging at 0. This downward trend indicates that, under the influence of Poisson noise, the model has found a local optimum that does not yield an output of 0 when the reactions are over. Furthermore, the contributions display a large magnitude at the start of the reaction, which is comparable to the results found for the bottleneck reaction. The predictions of the LSTM-based neural ODE are more accurate than the ones from the MLP-based neural ODE.

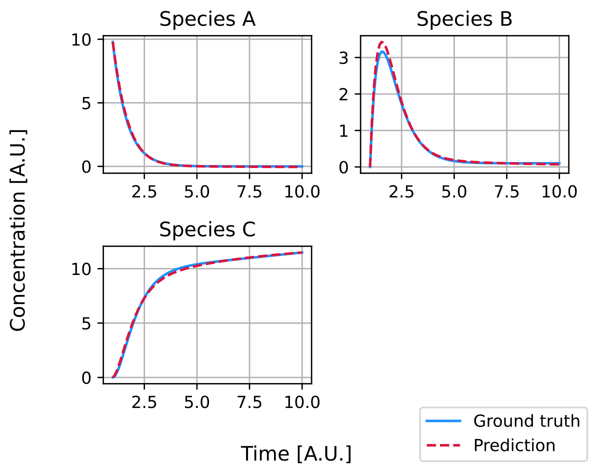


(a)

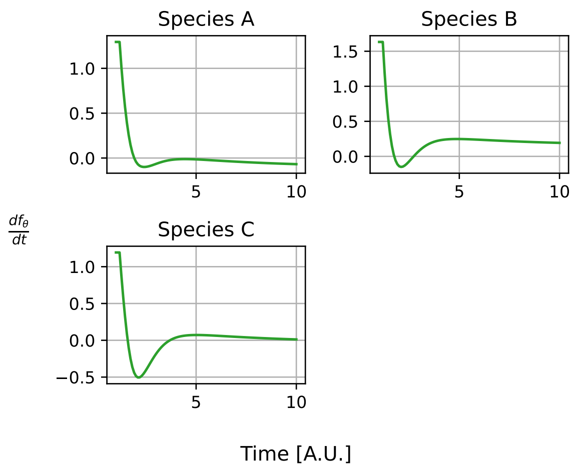


(b)

Figure 5.7: The modelling performance (a) and neural network contribution (b) of the neural ODE using a MLP on the four species within the open system reaction network.



(a)

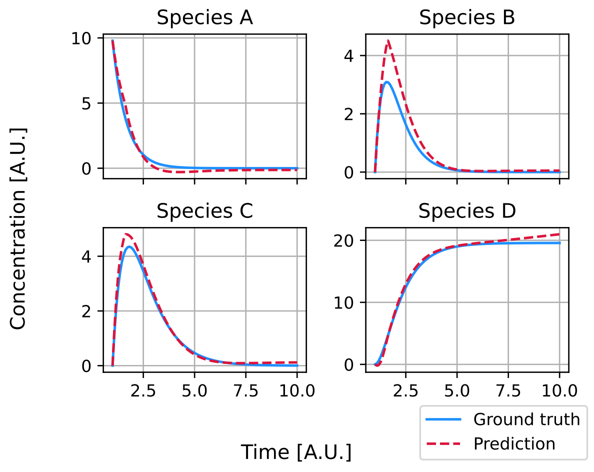


(b)

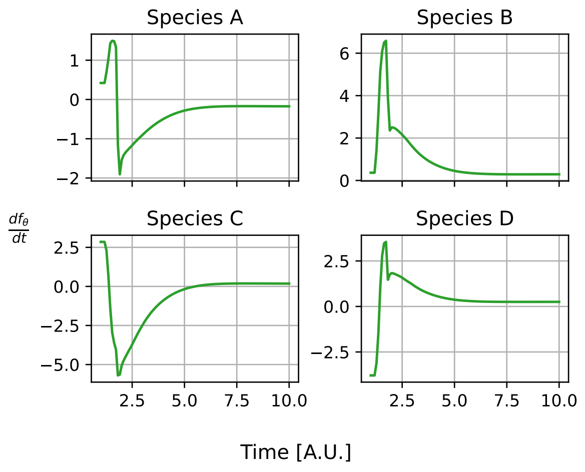
Figure 5.8: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the open system reaction network.

5.1.5 Missing reactions

When considering the CRN with missing reactions, it can be seen from Figure 5.9b and 5.10b that the contribution of the neural network is large for the species C and D, which are involved in the missing reaction $C \to D$. However, both models also seem to compensate for the missing reaction by increasing the concentration of species B while decreasing the concentration of species A. The increase of B implicitly leads to an increase of D due to the presence of the reaction $B \to D$. The artificial increase of B results in a slight deviation in the model prediction in Figures 5.9a and 5.10a.

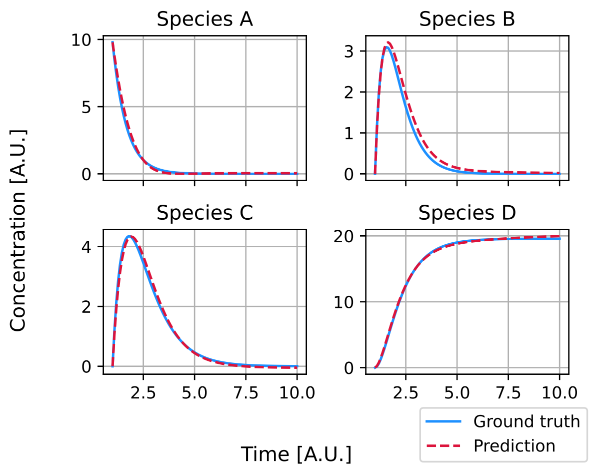


(a)

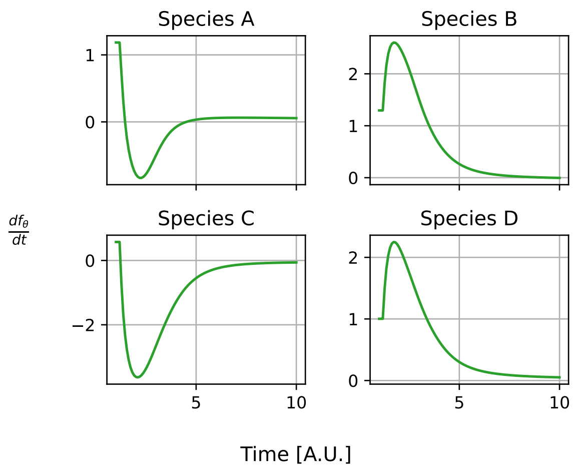


(b)

Figure 5.9: The modelling performance (a) and neural network contribution (b) of the neural ODE using a MLP on the four species within the missing reactions reaction network.



(a)

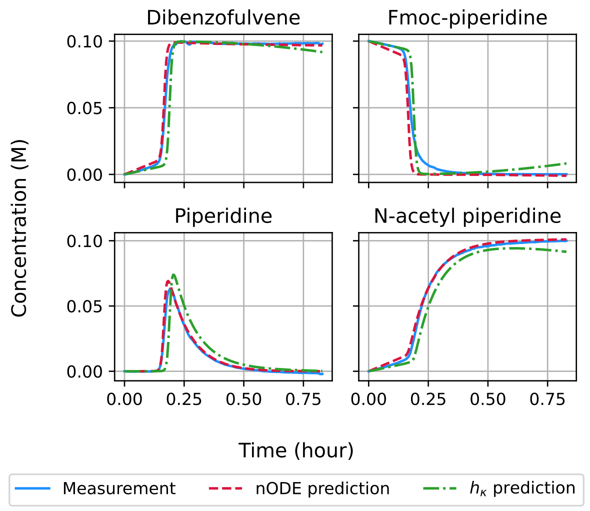


(b)

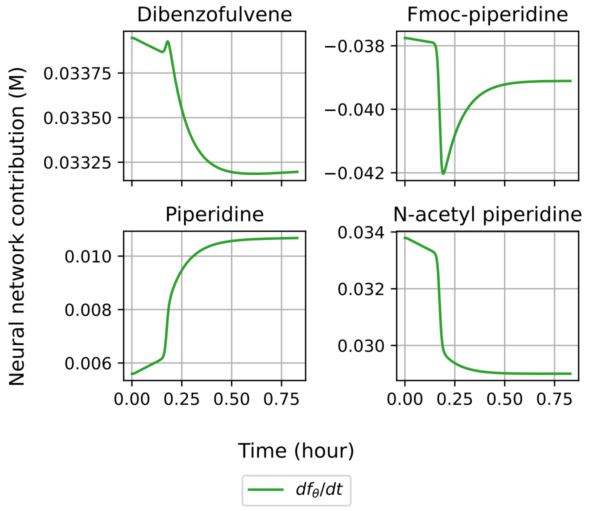
Figure 5.10: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the missing reactions reaction network.

5.2 Experimental data

When considering the experimental data, I test the neural ODE performance with three different variants of h_κ. Firstly, I consider the “standard” neural ODE, where h_κ is based on [21]. Figure 5.11 illustrates the results for the single-pulse experiment based on this model. The experimental measurements (solid blue), neural ODE predictions (dashed red) and predictions based on h_κ (dashdotted green) for the species Fmoc-piperidine (2), dibenzofulvene (5) piperidine (1) and N-acetyl piperidine (7) are shown in Figure 5.11a. The separate contribution of the neural network f_θ at each timepoint t, ${}^{df_\theta}/_{dt}$, is provided in Figure 5.11b. Figure 5.11a shows that the neural ODE provides a better fit than h_κ. This relative improvement can be attributed to the positive neural network contributions for the species dibenzofulvene (5) and piperidine and N-acetyl piperidine (7), and the negative contribution for Fmoc-piperidine (2, Figure 5.11b). The neural network contributions change over time. The rate of their change is the strongest at the start of the reaction and seems to converge towards the end of the reaction.



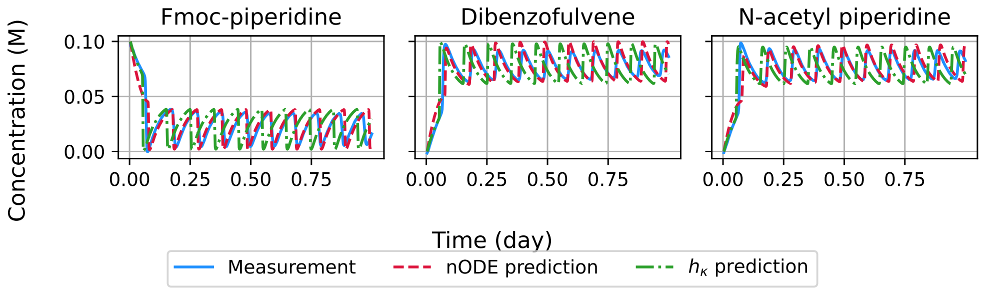
(a)



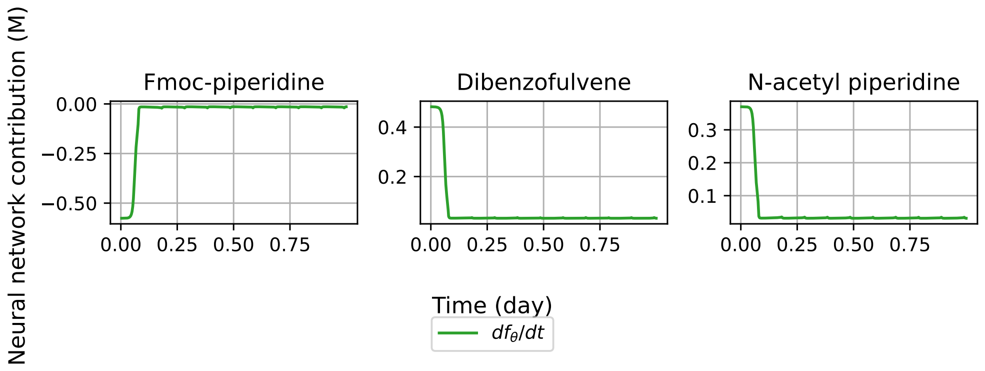
(b)

Figure 5.11: The predictive performance of the neural ODE on the single-pulse data. (a), the experimental measurements (solid blue), predictions from the neural ODE (dashed red) and h_κ (dashdotted green). The four panes illustrate the molar concentrations for the species dibenzofulvene, Fmoc-piperidine, piperidine and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

Figure 5.12 describes the experimental measurements, neural ODE predictions and predictions based on h_κ for the oscillating data. The contributions of the neural network are the strongest at the start of the reaction and converge to 0 after the initial pulse. The predictions by the neural ODE and h_κ have a comparable amplitude and frequency, but the offsets of the neural ODE predictions match better with the measurements. The neural ODE prediction and measurements diverge towards the end of the reaction.



(a)



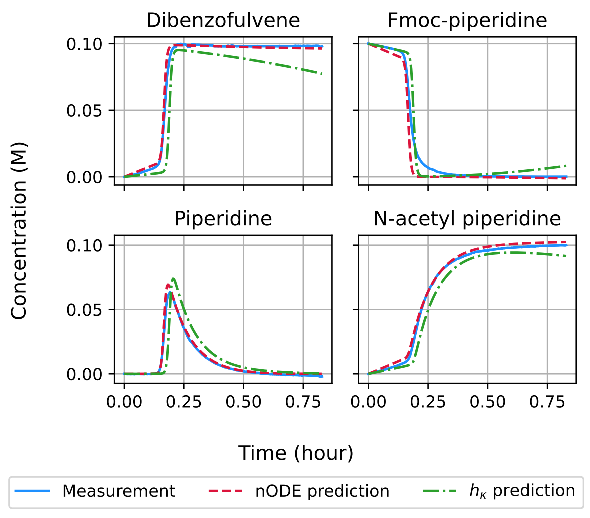
(b)

Figure 5.12: The predictive performance of the neural ODE on the oscillating data. (a), the experimental measurements (solid blue), predictions from the neural ODE (dashed red) and h_κ (dashdotted green). The four panes illustrate the molar concentrations for the species dibenzofulvene, Fmoc-piperidine, piperidine and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

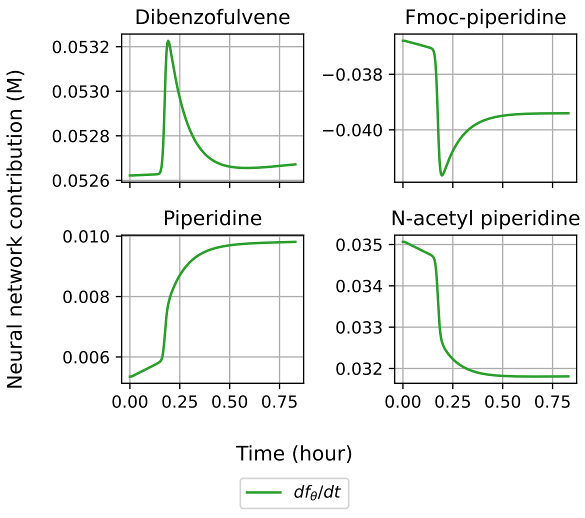
5.2.1 Open systems

Despite the small differences between the experimental measurements and the predictions with h_κ, I assume that the vector field provided in [21] is complete and that the data have been collected within a well-mixed, closed system. To test the predictive performance of the neural ODE on a system that violates the assumptions of the law of mass action, I artificially model an open system. To do so, I adapt the theoretical vector field by subtracting a fixed quantity of a species at each timestep dt. This efflux in the vector field is assumed to have a comparable effect to the influx of the same species within the experimental system. As a proof of concept, I subtract 20 mM Dibenzofulvene at each dt. The predictions of the neural ODE that correspond to this experimental paradigm are presented in Figure 5.13. Figure 5.13a demonstrates that the neural ODE is able to account for the misspecification within the theoretical vector field, while the performance of just h_κ has deteriorated significantly. When comparing the neural network contribution from Figure 5.13 to the contribution in

Figure 5.11, it can be seen that ${}^{df_\theta}/_{dt}$ as increased with approximately 0.02M. This increase matches the concentration that has been subtracted from the vector field.



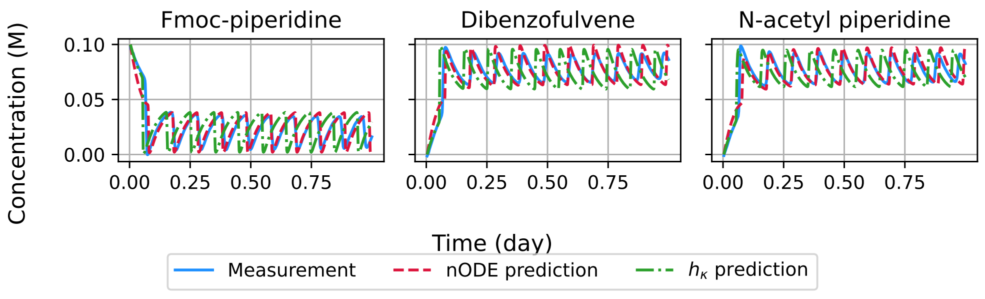
(a)



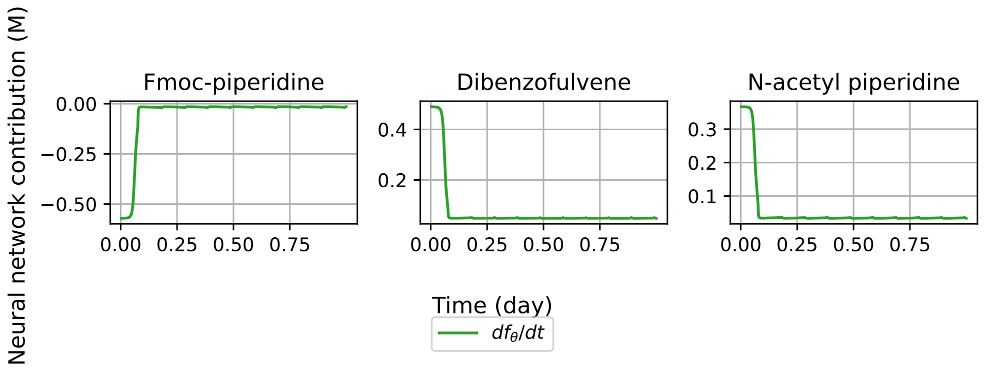
(b)

Figure 5.13: The predictive performance of the neural ODE on the open system. (a), the experimental measurements (solid blue), predictions from the neural ODE (dashed red) and h_κ (dashdotted green). (b), the neural network contribution.

Figure 5.14 describes the experimental measurements, neural ODE predictions and predictions based on h_κ for the oscillating data on the open system. Again, the neural network predictions are the largest during the initial phase of the reaction. Both predictions overestimate the amplitude of the oscillation. The prediction of the neural ODE forms the best match with the measurements with respect to the offset of the oscillations.



(a)

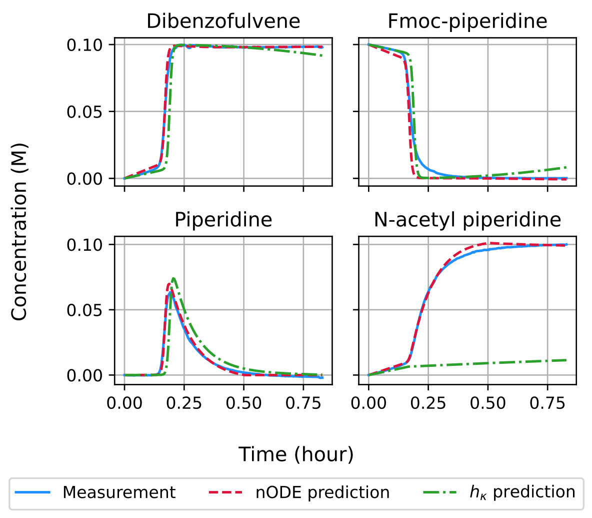


(b)

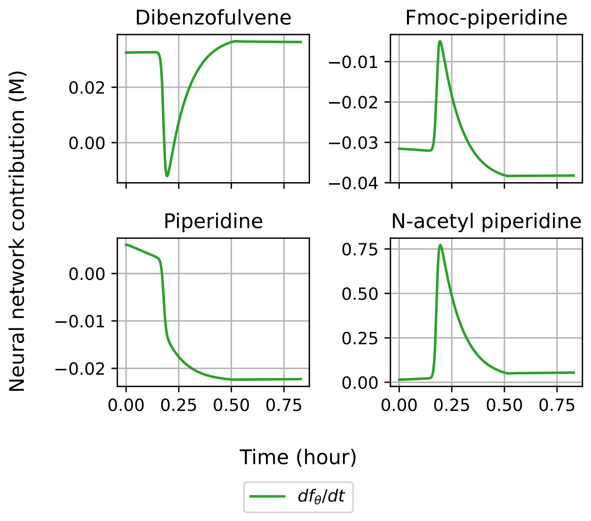
Figure 5.14: The predictive performance of the neural ODE on the oscillating data. (a), the experimental measurements (solid blue), predictions from the neural ODE (dashed red) and h_κ (dashdotted green). The four panes illustrate the molar concentrations for the species dibenzofulvene, Fmoc-piperidine, piperidine and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

5.2.2 Missing reactions

In this section, I assess the resilience of the neural ODE against incomplete vector fields, where the CRN contains more reactions than presumed in theory. To this end, I remove the slow inhibition pathway via phenyl acetate (4), which negatively affects the creation of N-acetyl piperidine (6). This modification differs from the creation of an open system as I am omitting a variable instead of subtracting a constant value. The results are shown in Figure 5.15. Even though the neural ODE fit is not perfect (Figure 5.15a), the missing variable is largely accounted for by an increase in the neural network contribution in Figure 5.15b compared to Figure 5.11b.



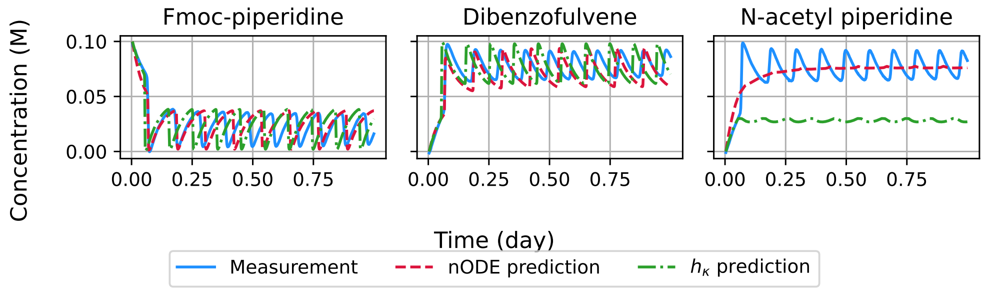
(a)



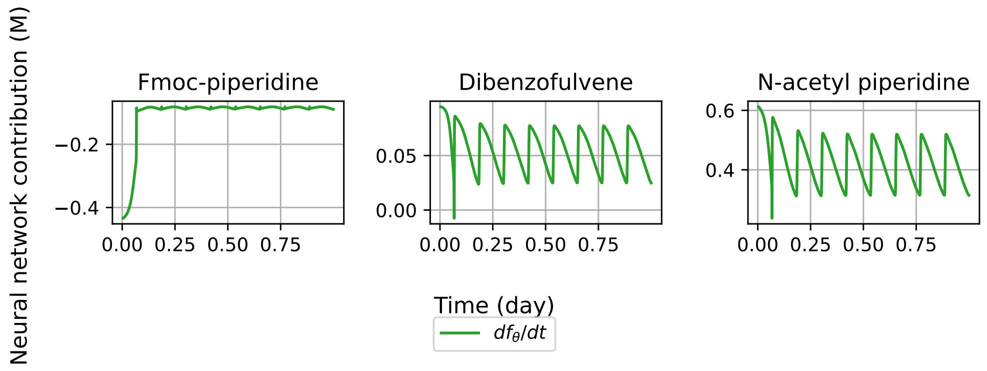
(b)

Figure 5.15: The predictive performance of the neural ODE with a missing reaction. (a), the experimental measurements (solid blue), predictions from the neural ODE (dashed red) and h_κ (dashdotted green). (b), the neural network contribution.

The neural ODE predictions made on the oscillating CRN have a lower quality than the single-pulse predictions. Figure 5.16a shows that the prediction for N-acetyl piperidine does not capture the oscillating behaviour displayed by the measurements. The neural network contribution, presented in Figure Figure 5.16b, does display oscillating behaviour. However, the contribution is too small to counter the misspecified theoretical part of the neural ODE.



(a)



(b)

Figure 5.16: The predictive performance of the neural ODE with a missing reaction. (a), the experimental measurements (solid blue), predictions from the neural ODE (dashed red) and h_κ (dashdotted green). (b), the neural network contribution.

5.3 Attention weights

To demonstrate the applicability of the attention layer within the LSTM, I apply an attention layer to the synthetic decaying Brusselator CRN. Figure 5.17 visualizes the attention weights. The attention weights resemble the oscillations of the Brusselator, for the magnitude of the weights also oscillates, converging towards the average of 0.003375. The effect of the attention weights depends on what species is considered, as the oscillations of the two species are each other's conjugates. If the large weights co-occur with the peaks of the oscillations, the influence of these time points is amplified. In contrast, if the large weights correspond to the valleys while the small weights match the peaks, the values at each timepoint contribute equally to the attention. As a consequence, the attention layer allows the model to focus on specific time points for particular species. This property can be beneficial when many species are involved in the CRN that do not play an equally important role in the reactions of interest.

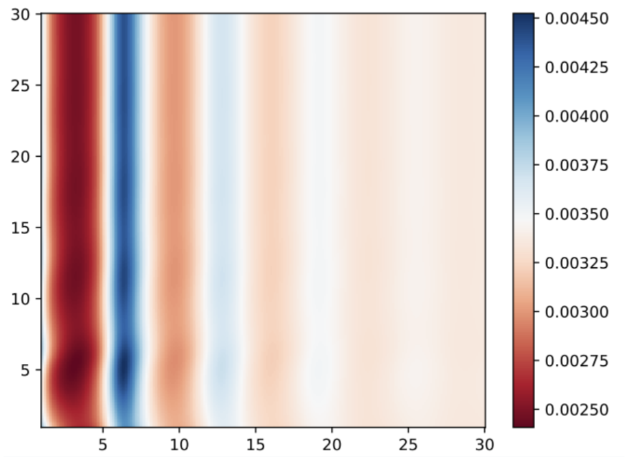


Figure 5.17: The training attention weights of the neural ODE using a LSTM network with temporal attention.

Chapter 6

Discussion

The results presented in the previous chapter show that a neural ODE can learn a good representation for most of the datasets considered. The quality of the predictions depends on the network architecture chosen as f_θ. In particular, the results of the synthetic data experiments indicate that the MLP does not perform well when the data are modelled with shot (Poisson) noise. The neural ODE based on the LSTM could fit all datasets but the experimentally measured oscillations combined with an incomplete vector field. In this section, I discuss the possible causes of the observations listed above. In addition, I present the limitations of this work.

6.1 A comparison of neural network architectures

Before diving into the specifics of both neural network architectures, I consider the differences between the noise distributions used. I focus on the Brusselator network, as the difference in performance between the MLP and LSTM was the clearest for this CRN. The noise that is modelled according to a Gaussian distribution Figure 4.2a is a lot smaller than the noise that is modelled with the Poisson distribution Figure 4.2b. As a consequence, there may be two reasons that explain the difference in performance. First, I suspect that the difference between the model performances could be caused by overfitting. This would be in accordance with the sizes of the two networks, where the MLP with two hidden layers of 64 neurons is more complex than the LSTM with a hidden state and fully connected layer with a width of 32. The memory component of the LSTM is not likely to make a large difference between the models, as the models are applied to a single timepoint per solver step. Second, the models might have gotten stuck in a local optimum during training. The difference in performance is then rooted in the architectural differences between the MLP and the LSTM cell. In particular, the gated structure of the LSTM may have been beneficial during training.

6.2 Modelling oscillating data with missing reactions

The predictions on the oscillating experimental measurements made with a neural ODE with an incomplete vector field (Figure 5.16) were the only results that did not fit the test data. In contrast, the neural ODE was able to account for the missing reactions when the single-pulse data is considered (Figure 5.15). Naturally, it is not guaranteed that the predictions presented in Figure 5.15 are made by a model that has reached the global optimum. This observation has been made in [33] as well. Still, the predictions based on a local optimum seem adequate, as the single pulse data is not overly complex. The oscillating data, however, comes from a more complex distribution. According to Harmsel et al. [21], it is not trivial to design a synthetic chemical oscillator. This assumption of complexity may hold as well for recreating such a vector field based on experimental measurements. In other words, it can be difficult to find the exact correction to the vector field that induces oscillating behaviour. As such, I hypothesize that, for the oscillating data, the local optima are a lot further from the global optimum than for the single-pulse data. As such, when the model converges to a local optimum, the predictions are fairly bad compared to the single-pulse variant. This problem can possibly be solved by experimenting with various schedulers and optimizers.

6.3 Limitations

A limitation of the method is that the neural network contribution is hard to interpret. The contribution represents a combination of all inconsistencies found between the data and the predictions by h_κ. As a consequence, it can be challenging to reason about, or train an algorithm on, the neural network contributions. In addition, there is not always a unique solution that provides an adequate fit. This is also highlighted by the comparison of the results found by the MLP and LSTM for the synthetic data (Section 5.1). As a consequence, it is even harder to reason about the contributions of the neural network.

Chapter 7

Conclusion

The behaviour of chemical species during a reaction is highly dynamic and can be modelled with a system of differential equations. The quality of this system is affected by the theoretical descriptions of the CRN, which can be incomplete. To account for the differences between the experimental data and the theoretical model, I combine dynamical systems modelling and deep learning in the form of neural ODEs. I test the modelling performance of the neural ODEs with standard CRNs, artificially introduced open systems and incomplete vector fields.

The results found underline the predictive qualities of LSTM-based neural ODEs: the models provide an excellent fit for almost all single-pulse and oscillating concentrations under the noisy conditions as presented above. In addition, the separate LSTM output sheds light on the magnitude, species concerned and temporal signature of the discrepancies between the theoretical model and the data. These insights can be translated into concrete changes to the theoretical model with the application of symbolic regression [22, 34]. As such, the neural network output can ultimately be used to reason about the structure of the CRN. Yet, it can be hard to interpret the contributions made by the neural network as its output aggregates all noise sources within the data.

In future work, this research can be extended by using neural ODEs to calibrate the theoretical vector field. For example, by changing the interaction between the neural network and the theoretical vector field, the network can be trained to learn the reaction rates while compensating for theoretical shortcomings.

Bibliography

[1] C. M. Guldberg and P. Waage. “Ueber die chemische Affinität. § 1. Einleitung”. In: *Journal für Praktische Chemie* 19.1 (1879), pp. 69–114. DOI: `https://doi.org/10.1002/prac.18790190111`.

[2] Péter Érdi and János Tóth. *Mathematical models of chemical reactions: theory and applications of deterministic and stochastic models*. Manchester University Press, 1989.

[3] Alejandro F Villaverde et al. “A protocol for dynamic model calibration”. In: *Briefings in Bioinformatics* 23.1 (Oct. 2021), bbab387. ISSN: 1477-4054. DOI: `10.1093/bib/bbab387`. eprint: `https://academic.oup.com/bib/article-pdf/23/1/bbab387/42230172/bbab387.pdf`. URL: `https://doi.org/10.1093/bib/bbab387`.

[4] Polina Lakrisenko et al. “Efficient computation of adjoint sensitivities at steady-state in ODE models of biochemical reaction networks”. In: *PLOS Computational Biology* 19.1 (2023), e1010783.

[5] Khuloud Jaqaman and Gaudenz Danuser. “Linking data to models: Data regression”. In: *Nature reviews. Molecular cell biology* 7 (Dec. 2006), pp. 813–9. DOI: `10.1038/nrm2030`.

[6] Mingjian Wen et al. “Chemical reaction networks and opportunities for machine learning”. In: *Nature Computational Science* 3.1 (Jan. 2023), pp. 12–24. DOI: `10.1038/s43588-022-00369-z`.

[7] David Angeli. “A Tutorial on Chemical Reaction Network Dynamics”. In: *European Journal of Control - EUR J CONTROL* 15 (May 2009), pp. 398–406. DOI: `10.3166/ejc.15.398-406`.

[8] Ricky TQ Chen et al. “Neural ordinary differential equations”. In: *Advances in neural information processing systems* 31 (June 2018). DOI: `10.48550/arXiv.1806.07366`.

[9] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: `1512.03385 [cs.CV]`.

[10] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. “Density estimation using real nvp”. In: *arXiv preprint arXiv:1605.08803* (2016).

[11] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural Networks* 2.5 (1989), pp. 359–366. ISSN: 0893-6080. DOI: `https://doi.org/10.1016/0893-6080(89)90020-8`.

[12] Christopher M Bishop. *Pattern recognition and machine learning*. Springer New York, NY, 2006.

[13] Patrick Kidger et al. *Neural Controlled Differential Equations for Irregular Time Series*. 2020. arXiv: `2005.08926 [cs.LG]`.

[14] Jaime Sevilla et al. “Compute Trends Across Three Eras of Machine Learning”. In: *2022 International Joint Conference on Neural Networks (IJCNN)*. 2022, pp. 1–8. DOI: `10.1109/IJCNN55064.2022.9891914`.

[15] Weiqi Ji and Sili Deng. “Autonomous discovery of unknown reaction pathways from data by chemical reaction neural network”. In: *The Journal of Physical Chemistry A* 125.4 (2021), pp. 1082–1092.

[16] Opeoluwa Owoyele and Pinaki Pal. “ChemNODE: A neural ordinary differential equations framework for efficient chemical kinetic solvers”. In: *Energy and AI* 7 (2022), p. 100118. ISSN: 2666-5468. DOI: `https://doi.org/10.1016/j.egyai.2021.100118`.

[17] Christopher Rackauckas et al. “Universal differential equations for scientific machine learning”. In: *arXiv preprint arXiv:2001.04385* (2020). DOI: `10.48550/arXiv.2001.04385`.

[18] Charles Francis Curtiss and Joseph O Hirschfelder. “Integration of stiff equations”. In: *Proceedings of the national academy of sciences* 38.3 (1952), pp. 235–243.

[19] Colin J. Aro. “CHEMSODE: a stiff ODE solver for the equations of chemical kinetics”. In: *Computer Physics Communications* 97.3 (1996), pp. 304–314. ISSN: 0010-4655. DOI: `10.1016/0010-4655(96)00071-9`.

[20] Anne Kværnø. “Singly diagonally implicit Runge–Kutta methods with an explicit first stage”. In: *BIT Numerical Mathematics* 44.3 (2004), pp. 489–502.

[21] Matthijs ter Harmsel et al. “A catalytically active oscillator made from small organic molecules”. In: *Nature* 621 (2023), pp. 87–93. DOI: `10.1038/s41586-023-06310-2`.

[22] Michael Schmidt and Hod Lipson. “Distilling Free-Form Natural Laws from Experimental Data”. In: *Science* 324.5923 (2009), pp. 81–85. DOI: `10.1126/science.1165893`.

[23] Robert A. Adams and Christopher Essex. *Calculus: a complete course*. 8th ed. Ontario: Pearson, 2013. ISBN: 978-0-32-178107-9.

[24] Yiping Lu et al. “Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 3276–3285.

[25] C. Runge. “Ueber die numerische Auflösung von Differentialgleichungen”. In: *Mathematische Annalen* 46 (1895), pp. 167–178. URL: `http://eudml.org/doc/157756`.

[26] W. Kutta. “Beitrag zur Naherungsweisen Integration Totaler Differentialgleichungen”. In: *Zeitschrift für Mathematik und Physik* 46 (1901), pp. 435–453.

[27] Patrick Kidger. “On Neural Differential Equations”. PhD thesis. University of Oxford, 2021.

[28] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: `10.1109/CVPR.2016.90`.

[29] Qian Yang, Carlos A Sing-Long, and Evan J Reed. “Learning reduced kinetic Monte Carlo models of complex chemistry from molecular dynamics”. In: *Chemical science* 8.8 (2017), pp. 5781–5796. DOI: `10.1039/C7SC01052D`.

[30] Andreas Raue et al. “Data2Dynamics: a modeling environment tailored to parameter estimation in dynamical systems”. In: *Bioinformatics* 31.21 (2015), pp. 3558–3560.

[31] Weiqi Ji et al. "Stiff-PINN: Physics-Informed Neural Network for Stiff Chemical Kinetics". In: *The Journal of Physical Chemistry A* 125.36 (2021), pp. 8098–8106. DOI: `10.1021/acs.jpca.1c05102`.

[32] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational physics* 378 (2019), pp. 686–707. DOI: `10.1016/j.jcp.2018.10.045`.

[33] Lena Podina, Brydon Eastman, and Mohammad Kohandel. "Universal Physics-Informed Neural Networks: Symbolic Differential Operator Discovery with Sparse Data". In: *Proceedings of the 40th International Conference on Machine Learning*. ICML'23. Honolulu, Hawaii, USA: JMLR.org, 2023.

[34] Silviu-Marian Udrescu and Max Tegmark. "AI Feynman: A physics-inspired method for symbolic regression". In: *Science Advances* 6.16 (2020), eaay2631.

[35] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. "Discovering governing equations from data by sparse identification of nonlinear dynamical systems". In: *Proceedings of the national academy of sciences* 113.15 (2016), pp. 3932–3937.

[36] Stéphane d'Ascoli et al. *ODEFormer: Symbolic Regression of Dynamical Systems with Transformers*. 2023. arXiv: `2310.05573 [cs.LG]`.

[37] Moritz Hoffmann, Christoph Fröhner, and Frank Noé. "Reactive SINDy: Discovering governing reactions from concentration data". In: *The Journal of chemical physics* 150.2 (Jan. 2019). DOI: `10.1063/1.5066099`.

[38] Igor Babuschkin et al. *The DeepMind JAX Ecosystem*. 2020. URL: `http://github.com/deepmind`.

[39] Patrick Kidger and Cristian Garcia. "Equinox: neural networks in JAX via callable PyTrees and filtered transformations". In: *Differentiable Programming workshop at Neural Information Processing Systems 2021* (2021).

[40] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Software available from wandb.com. 2020. URL: `https://www.wandb.com/`.

[41] Juntang Zhuang et al. *AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients*. 2020. arXiv: `2010.07468 [cs.LG]`.

[42] Ilya Prigogine. "Time, structure, and fluctuations". In: *Science* 201.4358 (1978), pp. 777–785.

[43] M. Farooq Wahab and Arved E. Reising. "The Ultimate Limit in Measurements by Instrumental Analysis: An Interesting Account of Schroteffekt and Shot Noise". In: *Journal of Chemical Education* 95.9 (2018), pp. 1668–1671. DOI: `10.1021/acs.jchemed.8b00202`.

Reflection

This is it: the process of writing my thesis has almost ended and I may present and defend my work this coming Friday. During one of our meetings, Tal compared the defence to a victory lap. Granted, I am nervous. Nonetheless, I am extremely excited to see what everyone thinks about the work Tal and I have done together.

Paradoxically, everything described in this document started about a year ago with a discussion on quantum machine learning. I was looking for a topic for my thesis in data science back at that time and I approached Tal for a discussion on his main research interests. Because I had no background in physics whatsoever, we quickly agreed this was not the best topic for me. Fortunately, Tal combined his broad interests with a great judgement of character and suggested this topic to me. The field of application, chemical reaction networks, immediately sparked my interest. Yet, I was unfamiliar with neural ordinary differential equations, or to quote Dr Patrick Kidger's amazing PhD thesis [13, p. 16]: "What is a neural differential equation anyway?"

Well, I have had plenty of opportunities to figure that out. I am elated with the experience I have gained while working on this thesis. I learned about differential equations, solvers, stiff equations and chemical dynamics. In addition, I have learned to work with the Jax family, which is a great addition to my experience with deep learning libraries. Tal's invitation to think critically about machine learning practices highlighted the similarities between machine learning and natural sciences (including physics, with his help of course). I deeply appreciate these implicit connections between different fields and I hope to unravel and understand more of these myself in the future as well. Next to the project itself, I learned more about the bigger picture of doing research as well. Doing research is not just about coding and writing, it also includes discussing, presenting and most important of all, being curious and having fun.

Bad reflections Was the whole project as idyllic as described above? By all means, no! Though, that's not a problem at all. In fact, by making mistakes I've created quite a bit of wisdom. I still have a lot to learn regarding communication and asking for help in particular. Truth be told, I enjoy programming challenges, and don't mind "being stuck" on a piece of code for a few hours.[1] It typically invites me to use different strategies than I have used

[1] That is, if there's no deadline very close by.

before, which is a great learning experience. However, I should also learn to learn from someone else's expertise. If there's one thing that I've learned, then it is that I have to *write everything down*. Or, in more academic terms, document everything. I have the tendency to clean notes, code or documentation only once in a while. It's way better to keep everything up to date and pushed to git. The latter also provides some peace of mind when your backpack is totally soaked by the Dutch downpour, and your laptop is still inside... I think that it would help if I split the main project into smaller subprojects of about a week. Then, at the end of the week, I can spend some time writing everything down in a structured way, which makes all the work more easily retrievable.

26-05-2024

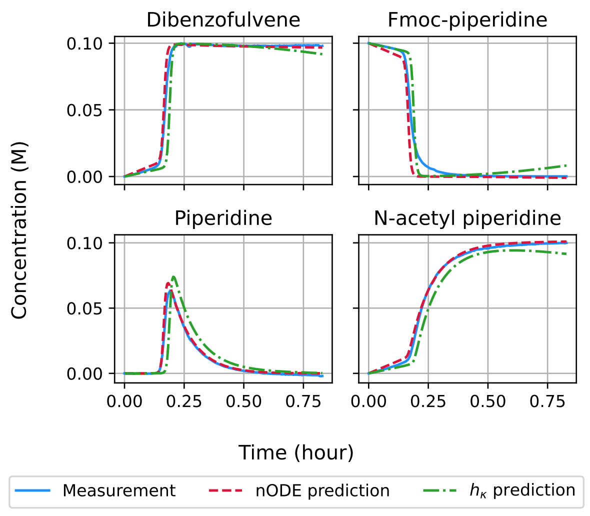
Appendix A

Supplementary figures

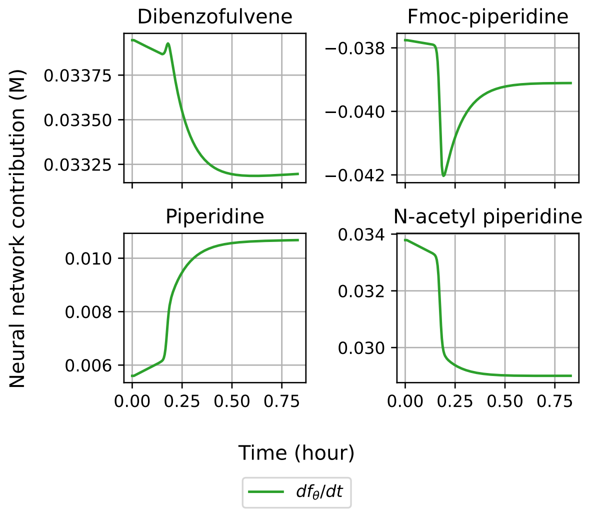
This chapter presents all results that have been found when considering the experimental data, summarizing the standard fit, open system and missing reaction experiments with the single-pulse (Appendix A.1) and oscillating data (Appendix A.2). The chapter serves as a reference without providing an in-depth description of all figures. Instead, all conclusions drawn from these results are provided in the main document.

A.1 Single-pulse experiment

A.1.1 Standard fit



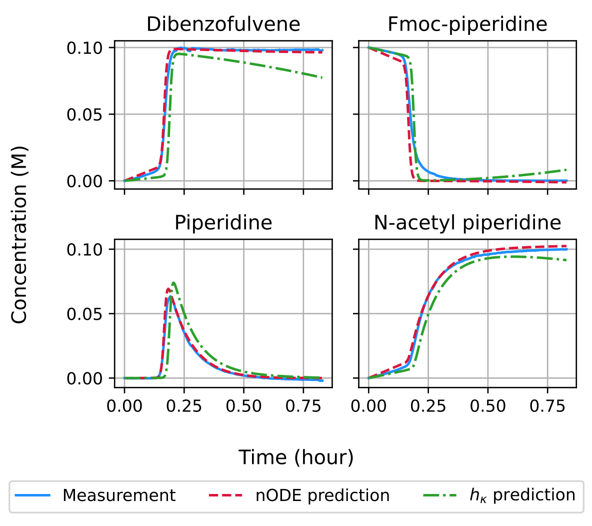
(a)



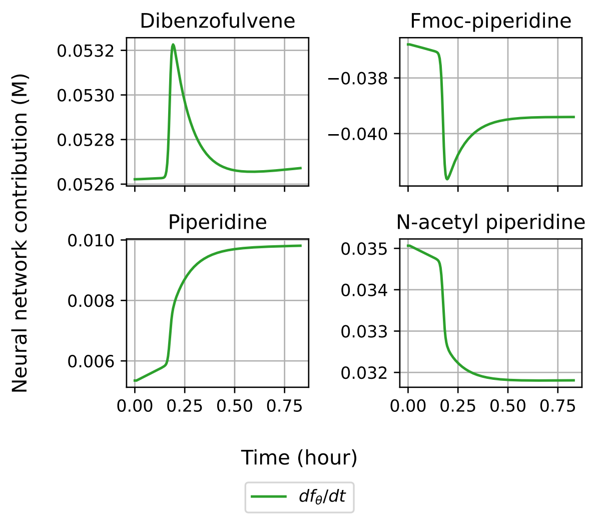
(b)

Figure A.1: The predictive performance of the neural ODE on the single-pulse data. (a), the experimental measurements, predictions (solid blue) from the neural ODE (dashed red) and h_κ (dashdotted green). The four panes illustrate the molar concentrations for the species dibenzofulvene, Fmoc-piperidine, piperidine and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

A.1.2 Open system



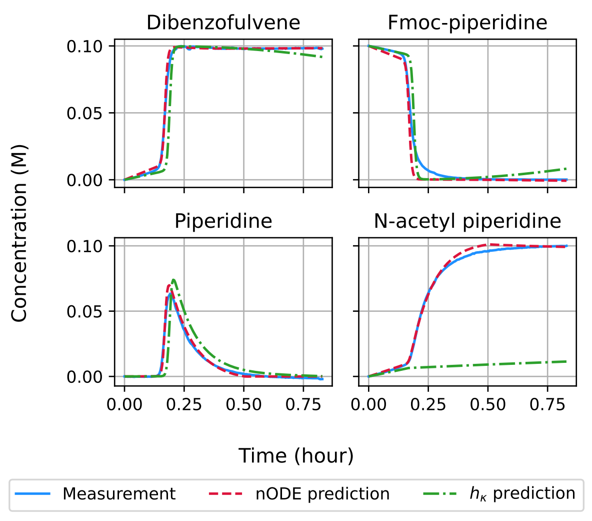
(a)



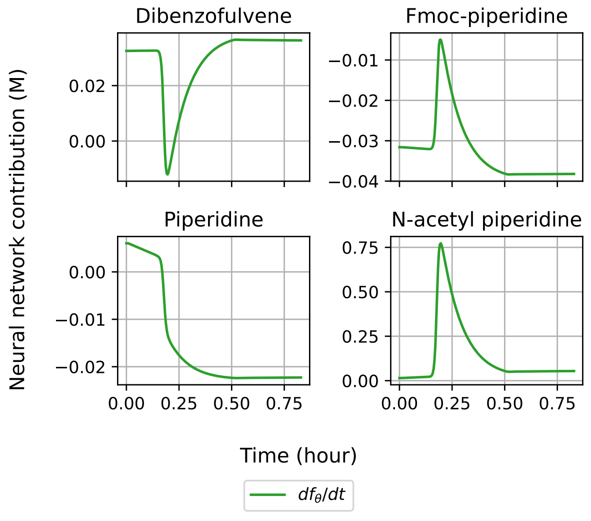
(b)

Figure A.2: The predictive performance of the neural ODE on the single-pulse data in the open system. (a), the experimental measurements, predictions (solid blue) from the neural ODE (dashed red) and h_κ (dashdotted green). The four panes illustrate the molar concentrations for the species dibenzofulvene, Fmoc-piperidine, piperidine and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

A.1.3 Missing reactions



(a)

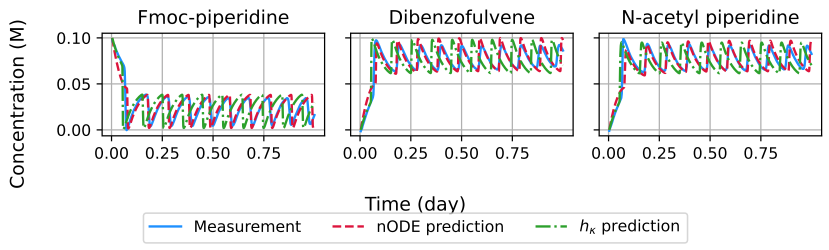


(b)

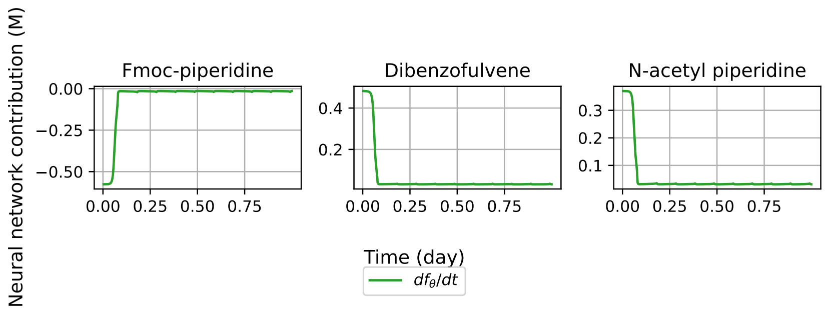
Figure A.3: The predictive performance of the neural ODE with an incomplete vector field on the single-pulse data in the open system. (a), the experimental measurements, predictions (solid blue) from the neural ODE (dashed red) and h_κ (dashdotted green). The four panes illustrate the molar concentrations for the species dibenzofulvene, Fmoc-piperidine, piperidine and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

A.2 Sustained oscillations

A.2.1 Standard fit



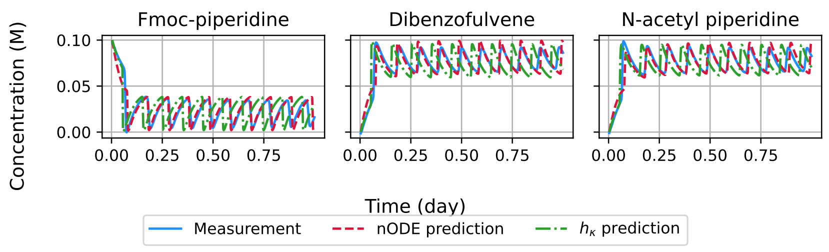
(a)



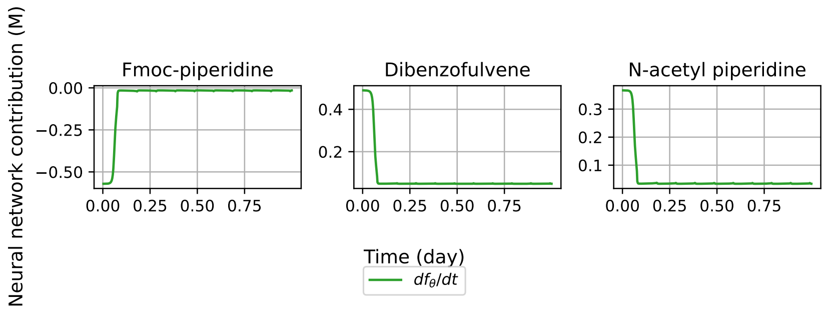
(b)

Figure A.4: The predictive performance of the neural ODE on the oscillating data. (a), the experimental measurements, predictions (solid blue) from the neural ODE (dashed red) and h_κ (dashdotted green). The three panes illustrate the molar concentrations for the species Fmoc-piperidine, dibenzofulvene and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

A.2.2 Open system



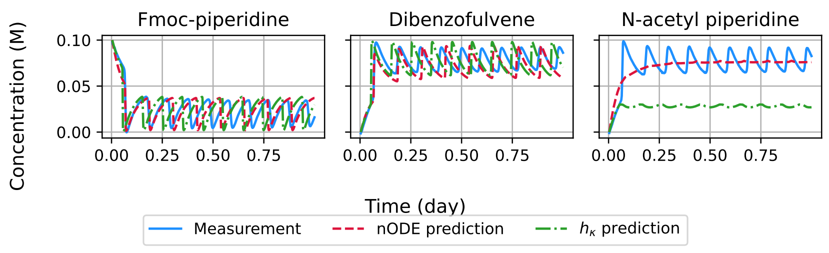
(a)



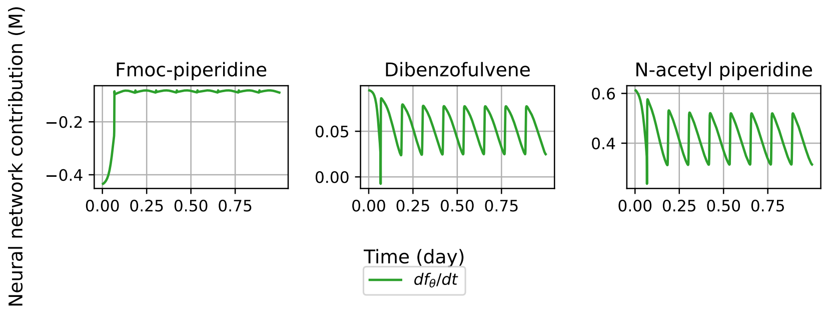
(b)

Figure A.5: The predictive performance of the neural ODE on the oscillating data in the open system. (a), the experimental measurements, predictions (solid blue) from the neural ODE (dashed red) and h_κ (dashdotted green). The three panes illustrate the molar concentrations for the species Fmoc-piperidine, dibenzofulvene and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

A.2.3 Missing reactions



(a)



(b)

Figure A.6: The predictive performance of the neural ODE on the oscillating data of the missing reactions dataset. (a), the experimental measurements, predictions (solid blue) from the neural ODE (dashed red) and h_κ (dashdotted green). The three panes illustrate the molar concentrations for the species Fmoc-piperidine, dibenzofulvene and N-acetyl piperidine. (b), the neural network contribution for the four measured species.

Appendix B

Experiments with synthetic data

The neural ODE has been trained on synthetic data to assess the performance of the model in a controlled setting. The performance has been tested on the four chemical reaction networks that are described below. For some chemical reaction networks, we use a "generative" and a "modelled" vector field to actively model a discrepancy between the generated data and the modelled vector field h_κ of the neural ODE. The datasets are generated with Gaussian and Poisson noise around the measurements. Furthermore, not all generated data points are provided to the neural ODE to model noisy measurement timings. The parameters that regulate the noise are described in Appendix B.1. A visualization of the noisy training data is provided in Appendix B.2. The modelling performance of the neural ODE is shown in Appendix B.3.

B.1 Chemical reaction networks

This section lists the parameters used to generate the synthetic datasets. For each dataset, the chemical specification, vector field and modelling parameters are provided. A description of the parameters can be found in Table B.1.

Abbreviation	Description
t_{min}, t_{max}	The first and last recorded time point.
T	The total number of recorded time points.
T_{sample}	The fraction of time points in the training data.
σ	The standard deviation of the Gaussian distribution used to mimic noisy observations.
λ	The rate of the Poisson distribution used to mimic noisy observations.
μ_{y_0}	The mean of the Gaussian distribution used to sample the initial concentrations.
σ_{y_0}	The standard deviation of the Gaussian distribution used to sample the initial concentrations.
μ_k	The mean of the Gaussian distribution used to sample the reaction rates $\boldsymbol{k}$.

Table B.1: The description of the parameters used to generate the synthetic datasets.

B.1.1 Bottleneck reaction network

The bottleneck reaction network consists of three species and is described below. The parameters that are used to generate the data are presented in Table B.2.

Specification.

$$A \xrightarrow{k_1} B \qquad (R_1)$$
$$B \xrightarrow{k_2} C \qquad (R_2)$$

Vector field.

$$\frac{d[A]}{dt} = -k_1[A]$$
$$\frac{d[B]}{dt} = k_1[A] - k_2[B]$$
$$\frac{d[C]}{dt} = k_2[B]$$

Parameters.

t_{min}	t_{max}	T	T_{sample}	σ	λ	σ_{y0}
1	10	100	0.7	0.1	1	0.1

Table B.2: The parameters used to generate the data that correspond to the bottleneck reaction network.

B.1.2 Brusselator

The Brusselator consists of 5 species and is described below [42]. The vector field only describes species A and B, as species X and Y are provided in excess, making their concentrations constant. For stable oscillations $X = 1$ and $Y = 3$. A Brusselator with decaying oscillations is created by $X = 1$ and $Y = 1.7$. The parameters that are used to generate the data are presented in Table B.3.

Specification.

$$X \xrightarrow{k_1} A \qquad (R_1)$$
$$2A + B \xrightarrow{k_2} 3A \qquad (R_2)$$
$$Y + A \xrightarrow{k_3} B + D \qquad (R_3)$$
$$A \xrightarrow{k_4} E \qquad (R_4)$$

Vector field.

$$\frac{d[A]}{dt} = k_1[X] + k_2[A]^2[B] - k_3[Y][A]$$
$$\frac{d[B]}{dt} = -k_2[A]^2[B] + k_3[Y][A]$$

Parameters.

t_{min}	t_{max}	T	T_{sample}	σ	λ	σ_{y0}
1	30	300	0.7	0.01	0.1	0.01

Table B.3: The parameters used to generate the data that correspond to the Brusselator reaction network.

B.1.3 Open system

The open system consists of three species and is described below. The reaction network includes an influx of 0.2B per dt, which is not known to the model. This is reflected in the difference between the generative and the modelled vector field. The parameters that are used to generate the data are presented in Table B.4.

Specification.

$$A \xrightarrow{k_1} B \quad (R_1)$$
$$\varnothing \xrightarrow{0.2} B \quad (R_2)$$
$$B \xrightarrow{k_2} C \quad (R_3)$$

Generative vector field.

$$\frac{d[A]}{dt} = -k_1[A]$$
$$\frac{d[B]}{dt} = k_1[A] - k_2[B] + 0.2$$
$$\frac{d[C]}{dt} = k_2[B]$$

Modelled vector field.

$$\frac{d[A]}{dt} = -k_1[A]$$
$$\frac{d[B]}{dt} = k_1[A] - k_2[B]$$
$$\frac{d[C]}{dt} = k_2[B]$$

Parameters.

t_{min}	t_{max}	T	T_{sample}	σ	λ	σ_{y_0}
1	10	100	0.7	0.1	1	0.1

Table B.4: The parameters used to generate the data that correspond to the open system reaction network.

B.1.4 Missing reactions

The missing reactions network consists of four species and is described below. The reaction $C \to D$ is not known to the model. This is reflected in the difference between the generative and the modelled vector field. The parameters that are used to generate the data are presented in Table B.5.

Specification.

$$A \xrightarrow{k_1} B + C \quad (R_1)$$
$$B \xrightarrow{k_2} D \quad (R_2)$$
$$C \xrightarrow{k_3} D \quad (R_3)$$

Generative vector field.

$$\frac{d[A]}{dt} = -k_1[A]$$
$$\frac{d[B]}{dt} = k_1[A] - k_2[B]$$
$$\frac{d[C]}{dt} = k_1[A] - k_3[C]$$
$$\frac{d[D]}{dt} = k_2[B] + k_3[C]$$

Modelled vector field.

$$\frac{d[A]}{dt} = -k_1[A]$$
$$\frac{d[B]}{dt} = k_1[A] - k_2[B]$$
$$\frac{d[C]}{dt} = k_1[A]$$
$$\frac{d[D]}{dt} = k_2[B]$$

Parameters.

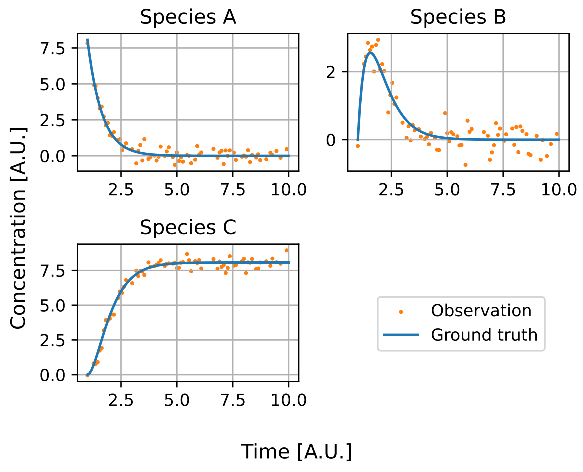
t_{min}	t_{max}	T	T_{sample}	σ	λ	σ_{y_0}
1	10	100	0.7	0.1	1	0.1

Table B.5: The parameters used to generate the data that correspond to the missing reactions reaction network.

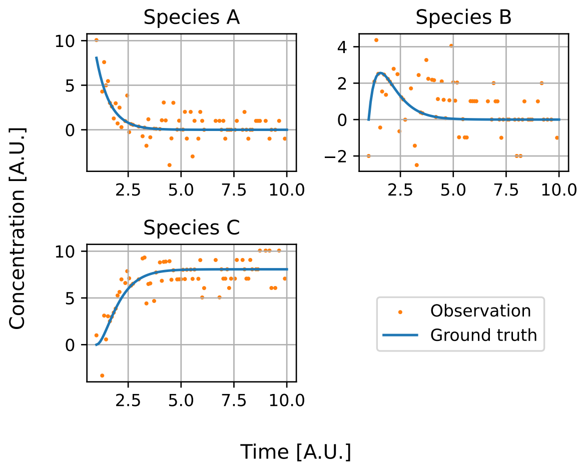
B.2 Visualization of the synthetic training data

This section provides examples of the synthetic training data generated with the vector fields from the previous chapter. The data are polluted with Gaussian (subplots (a)) and Poisson (subplots (b)) noise. The noisy data points are plotted in orange, while the noiseless ground truth is shown in blue.

B.2.1 Bottleneck reaction network



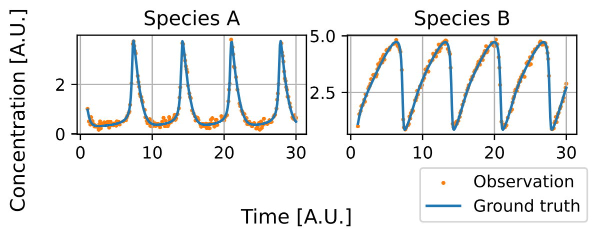
(a)



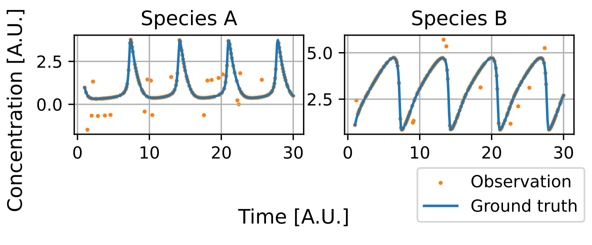
(b)

Figure B.1: The modelled ground truth (blue) and the noisy observations (orange) generated with the bottleneck reaction network. The observations represent a single data point. The measurement noise has been generated using (a) a Gaussian distribution with $\sigma = 0.1$ and (b) a Poisson distribution with $\lambda = 1$. The sampling rate is $T_{sample} = 0.7$.

B.2.2 Brusselator



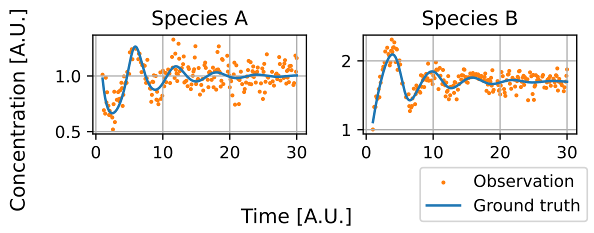
(a)



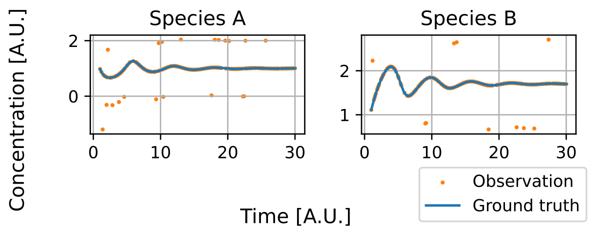
(b)

Figure B.2: The modelled ground truth (blue) and the noisy observations (orange) generated with the Brusselator reaction network. The observations represent a single data point. The measurement noise has been generated using (a) a Gaussian distribution with $\sigma = 0.01$ and (b) a Poisson distribution with $\lambda = 0.1$. The sampling rate is $T_{sample} = 0.7$.

B.2.3 Decaying Brusselator



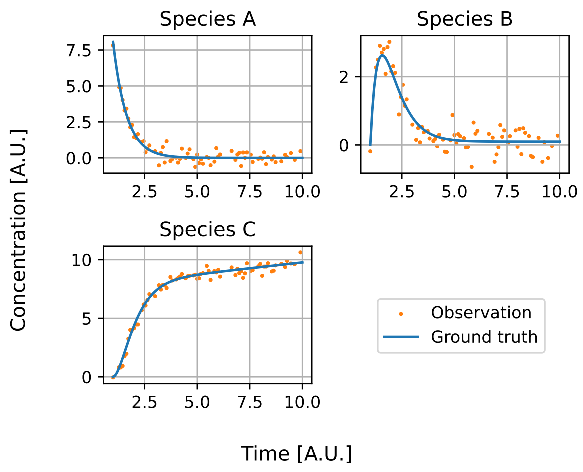
(a)



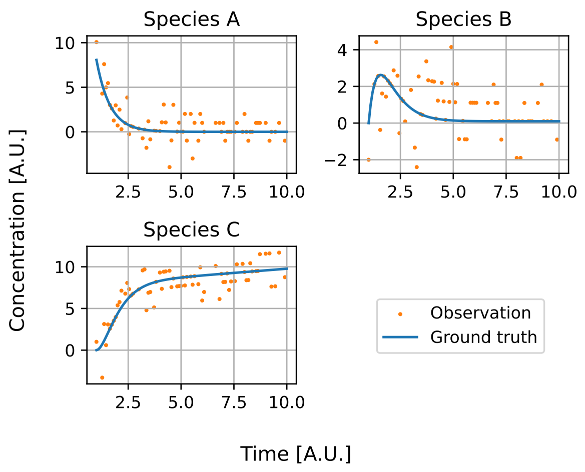
(b)

Figure B.3: The modelled ground truth (blue) and the noisy observations (orange) generated with the decaying Brusselator reaction network. The observations represent a single data point. The measurement noise has been generated using (a) a Gaussian distribution with $\sigma = 0.01$ and (b) a Poisson distribution with $\lambda = 0.1$. The sampling rate is $T_{sample} = 0.7$.

B.2.4 Open system



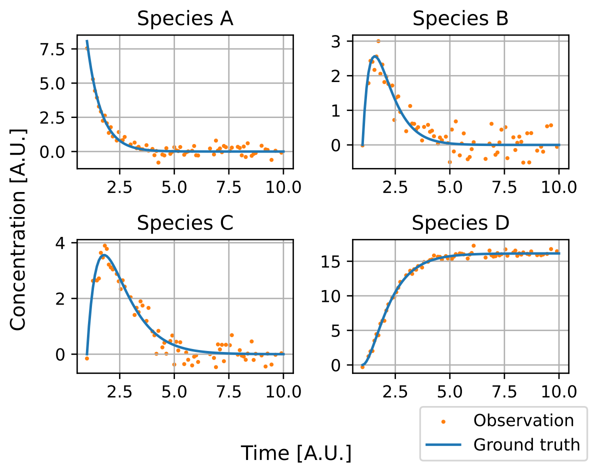
(a)



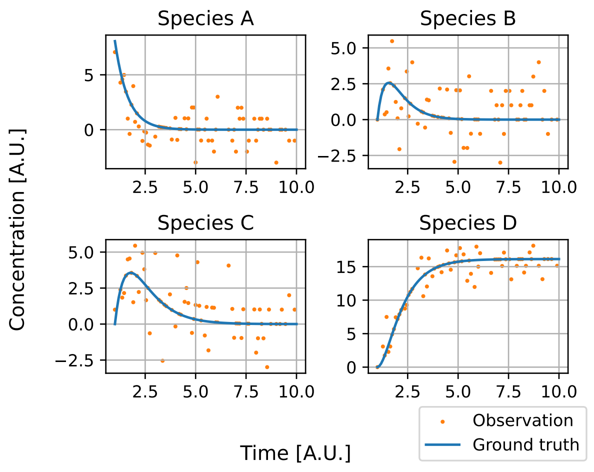
(b)

Figure B.4: The modelled ground truth (blue) and the noisy observations (orange) generated with the open system reaction network. The observations represent a single data point. The measurement noise has been generated using (a) a Gaussian distribution with $\sigma = 0.1$ and (b) a Poisson distribution with $\lambda = 1$. The sampling rate is $T_{sample} = 0.7$.

B.2.5 Missing reactions



(a)



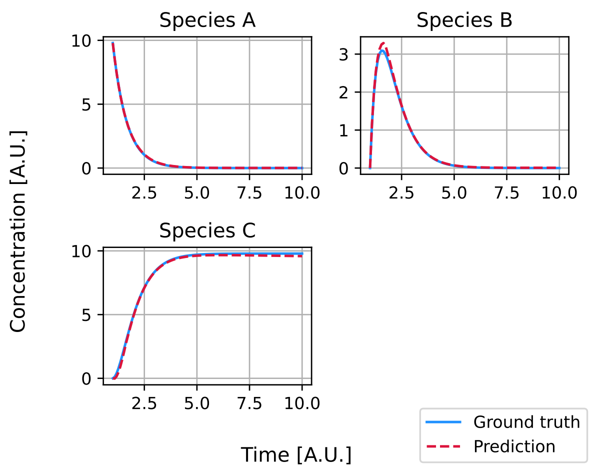
(b)

Figure B.5: The modelled ground truth (blue) and the noisy observations (orange) generated with the missing reactions reaction network. The observations represent a single data point. The measurement noise has been generated using (a) a Gaussian distribution with $\sigma = 0.1$ and (b) a Poisson distribution with $\lambda = 1$. The sampling rate is $T_{sample} = 0.7$.

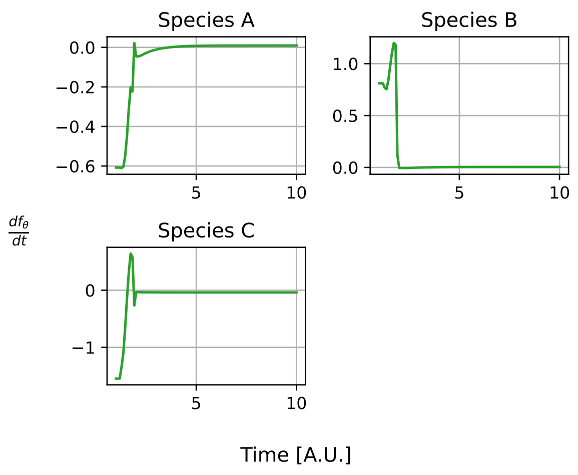
B.3 Neural ODE modelling performance

In this section, the modelling performances of the different neural ODEs are presented. The results are grouped based on the modelled measurement noise, which can either come from a Gaussian (Section B.3.1) or Poisson distribution (Section B.3.2). We test the performance of a neural ODE that is based on a multilayer perceptron (MLP) and a neural ODE that is based on a long short-term memory (LSTM) network. While the neural ODE with the MLP vector field is able to model the synthetic data under Gaussian noise, Appendix B.3.2 shows that the performance deteriorates under Poisson noise for the Brusselator CRNs. The neural ODE with the LSTM provides a better fit in these cases.

B.3.1 Gaussian noise



(a)



(b)

Figure B.6: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the bottleneck reaction network.

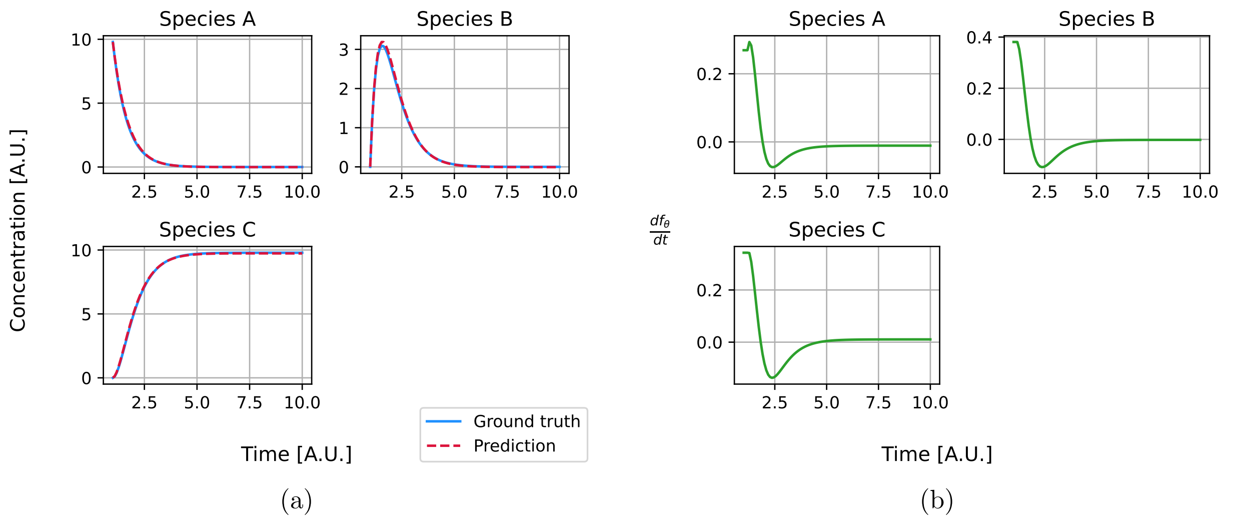


Figure B.7: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the bottleneck reaction network.

Bottleneck reaction network

Brusselator

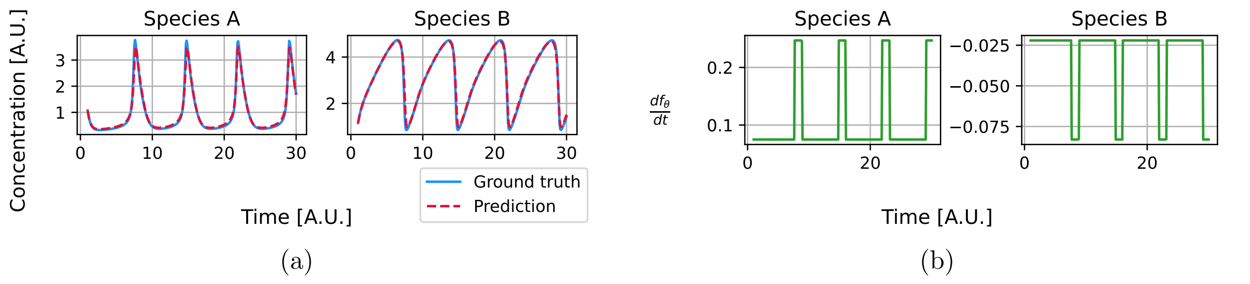
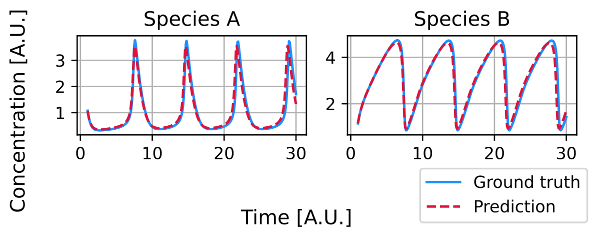
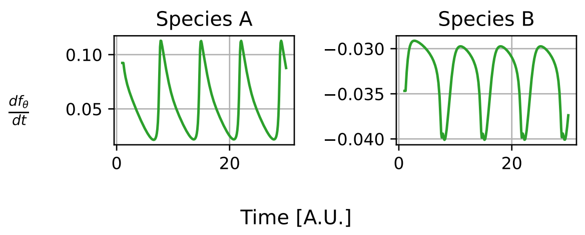


Figure B.8: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the Brusselator reaction network.



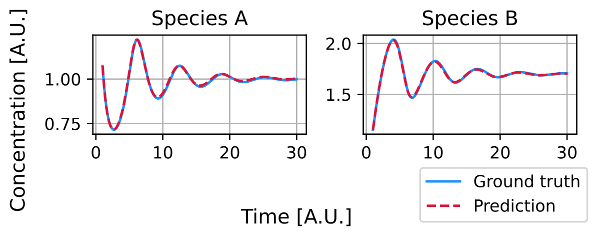
(a)



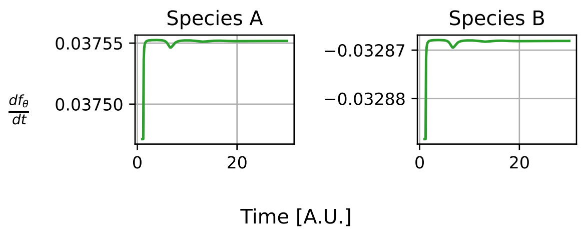
(b)

Figure B.9: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the Brusselator reaction network.

Decaying Brusselator

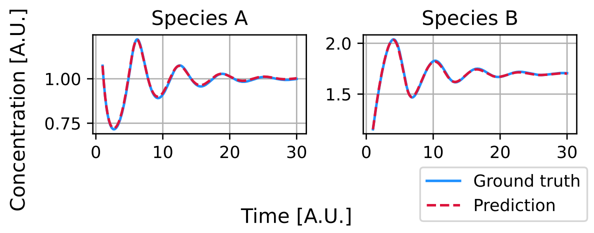


(a)

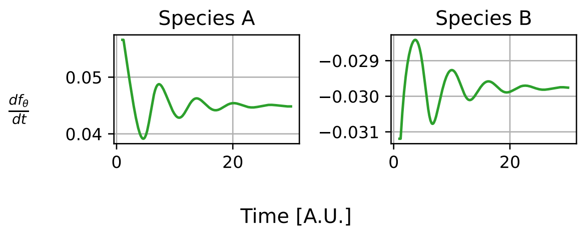


(b)

Figure B.10: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the decaying Brusselator reaction network.



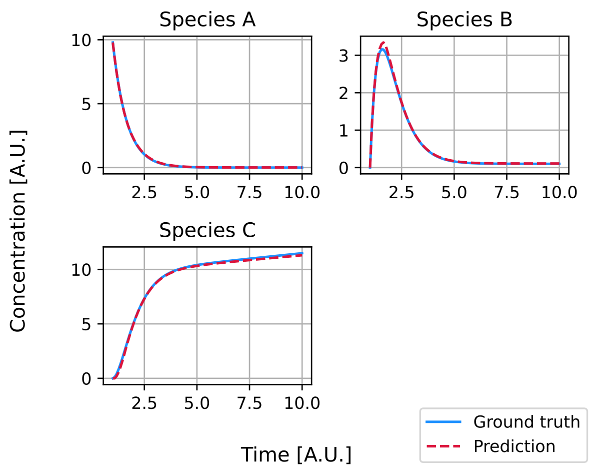
(a)



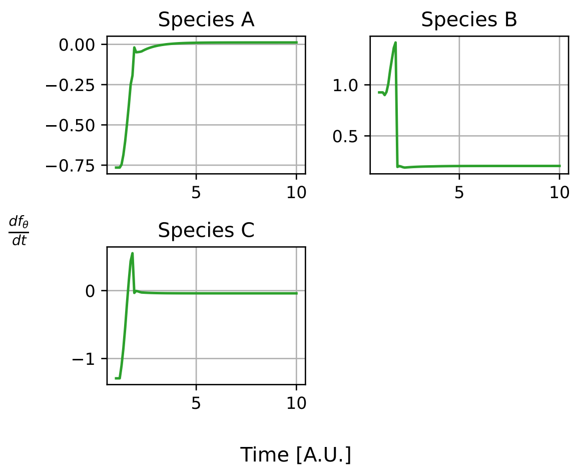
(b)

Figure B.11: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the decaying Brusselator reaction network.

Open system

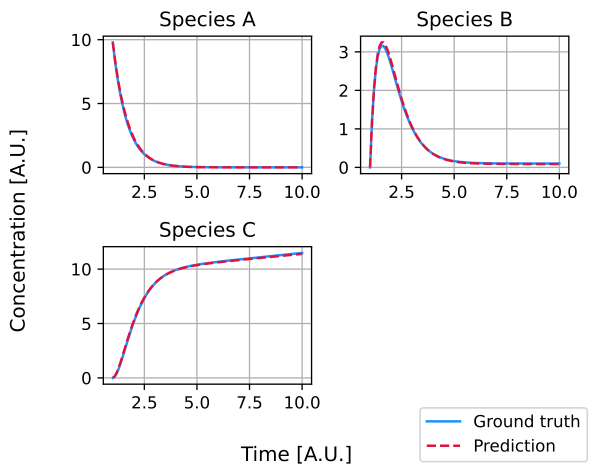


(a)

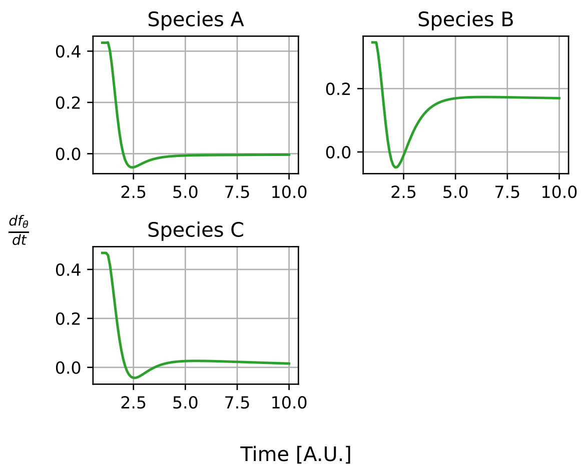


(b)

Figure B.12: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the open system reaction network.



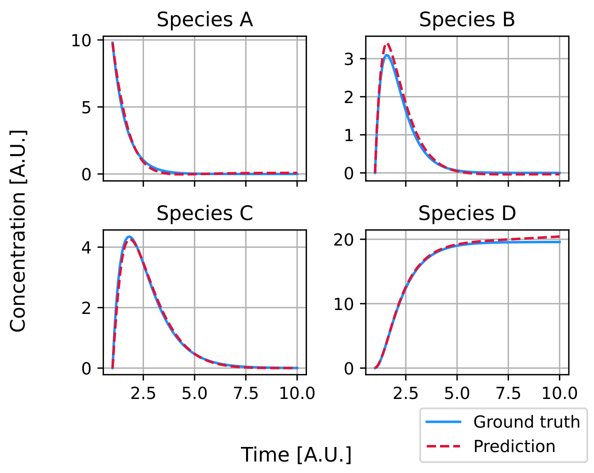
(a)



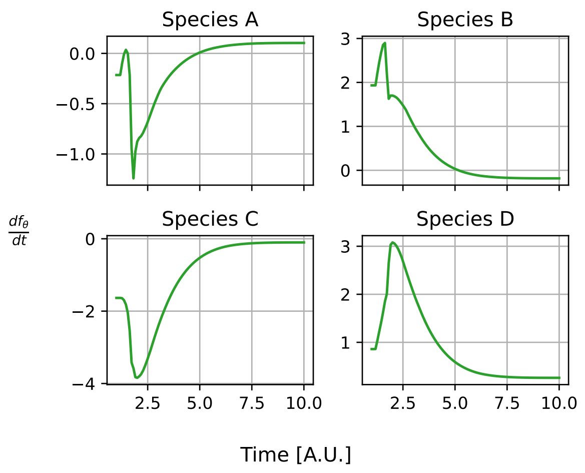
(b)

Figure B.13: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the open system reaction network.

Missing reactions

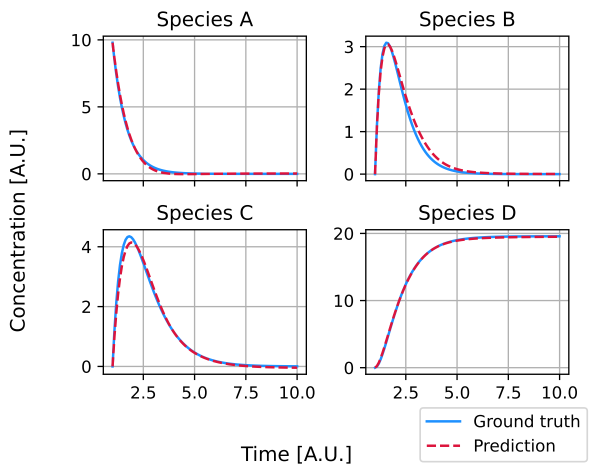


(a)

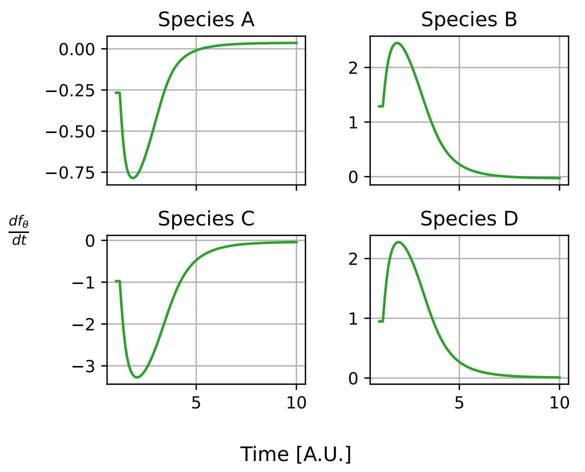


(b)

Figure B.14: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the missing reactions reaction network.



(a)



(b)

Figure B.15: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the missing reactions reaction network.

B.3.2 Poisson noise

Bottleneck reaction network

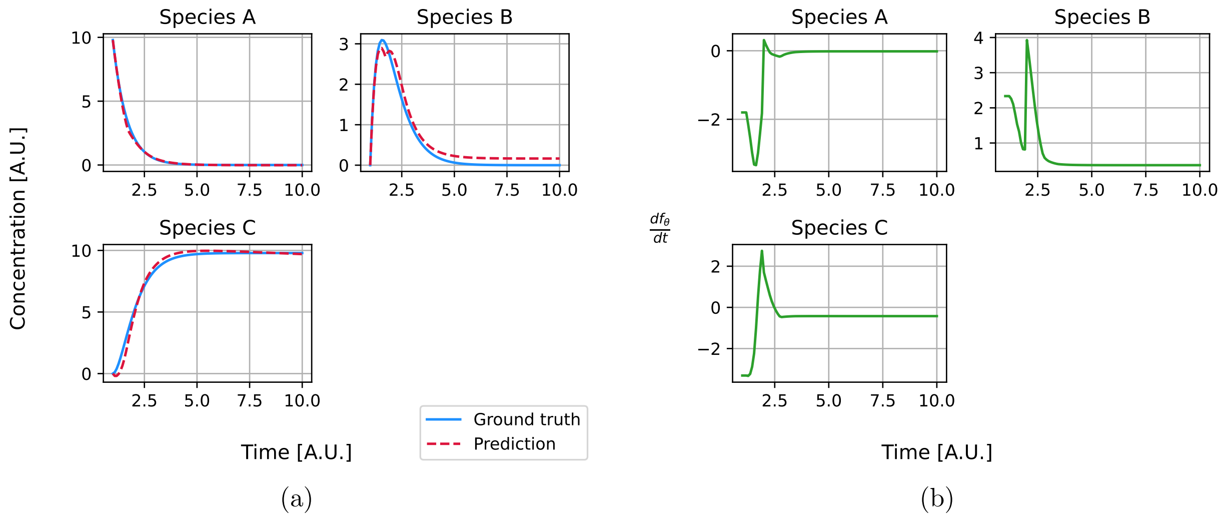


Figure B.16: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the bottleneck reaction network.

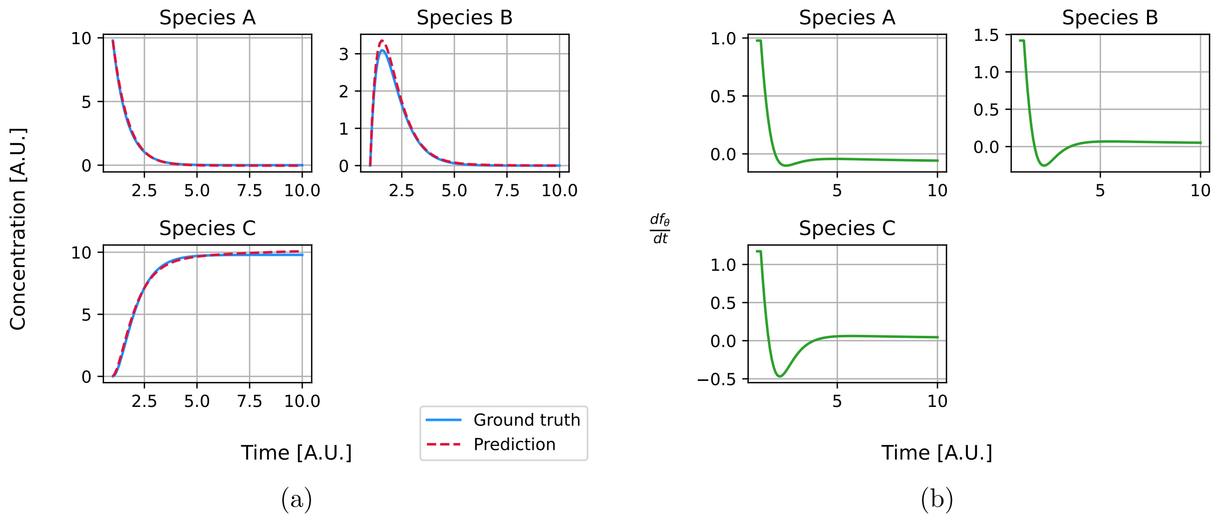
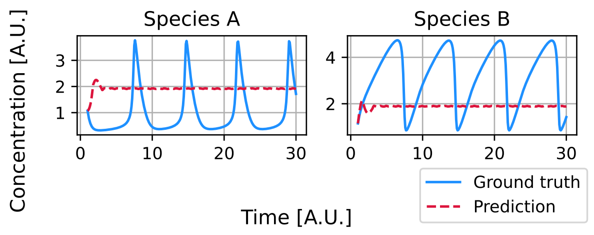
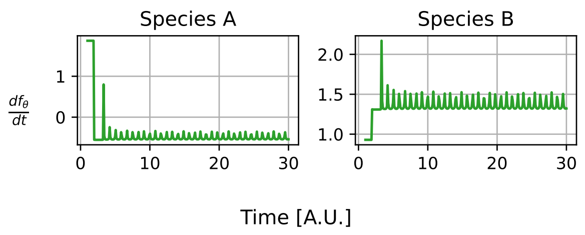


Figure B.17: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM perceptron on the four species within the bottleneck reaction network.

Brusselator

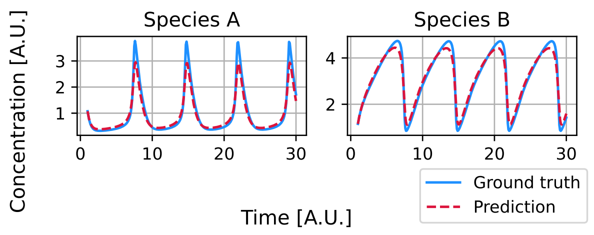


(a)

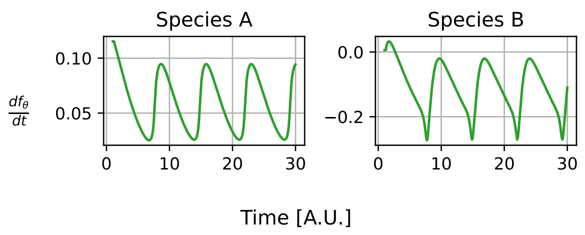


(b)

Figure B.18: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the Brusselator reaction network.



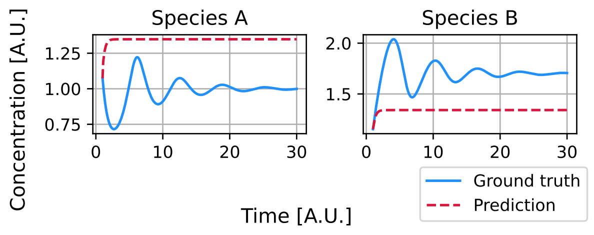
(a)



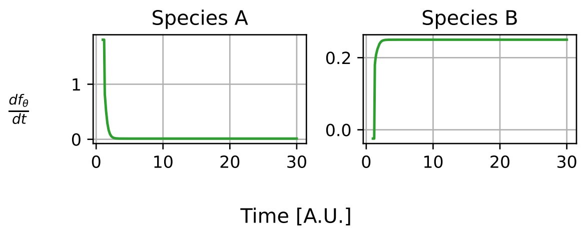
(b)

Figure B.19: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the Brusselator reaction network.

Decaying Brusselator

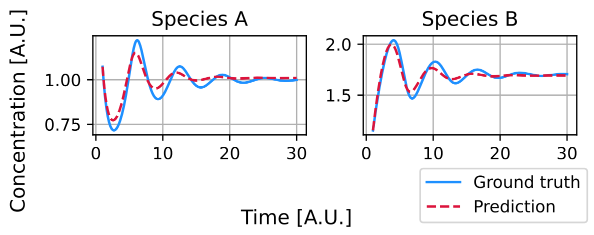


(a)

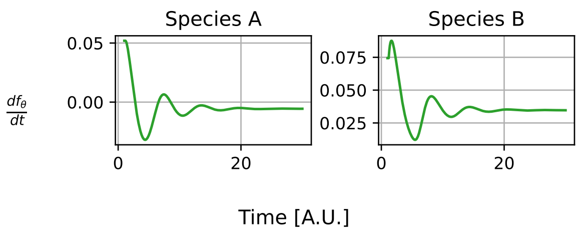


(b)

Figure B.20: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the decaying Brusselator reaction network.



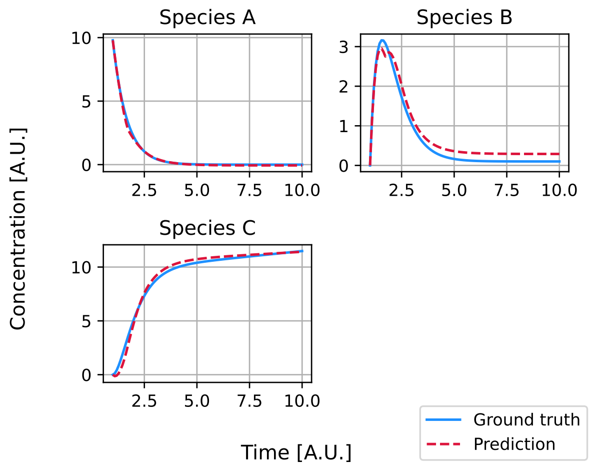
(a)



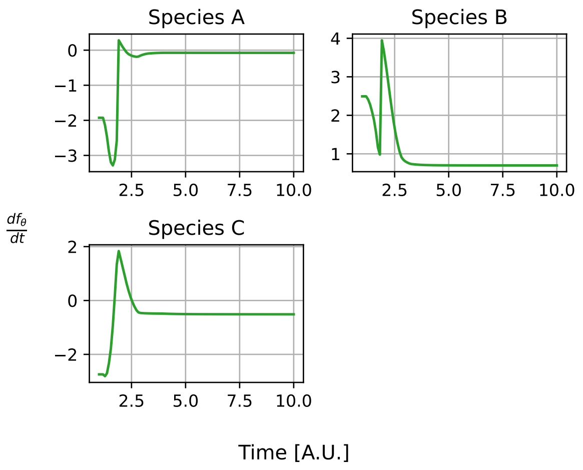
(b)

Figure B.21: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the decaying Brusselator reaction network.

Open system

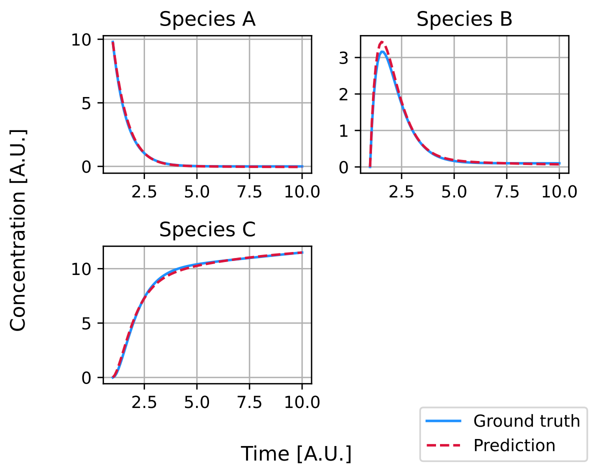


(a)

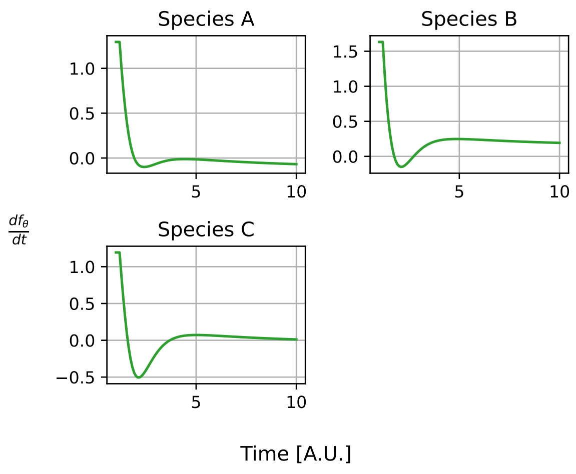


(b)

Figure B.22: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the open system reaction network.



(a)



(b)

Figure B.23: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the open system reaction network.

Missing reactions

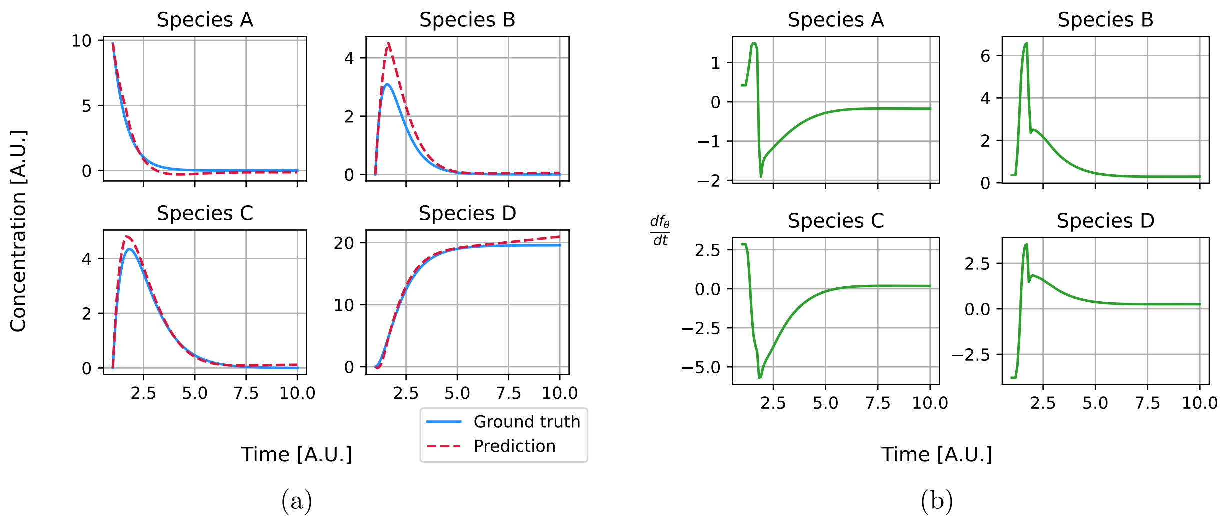


Figure B.24: The modelling performance (a) and neural network contribution (b) of the neural ODE using a multilayer perceptron on the four species within the missing reactions reaction network.

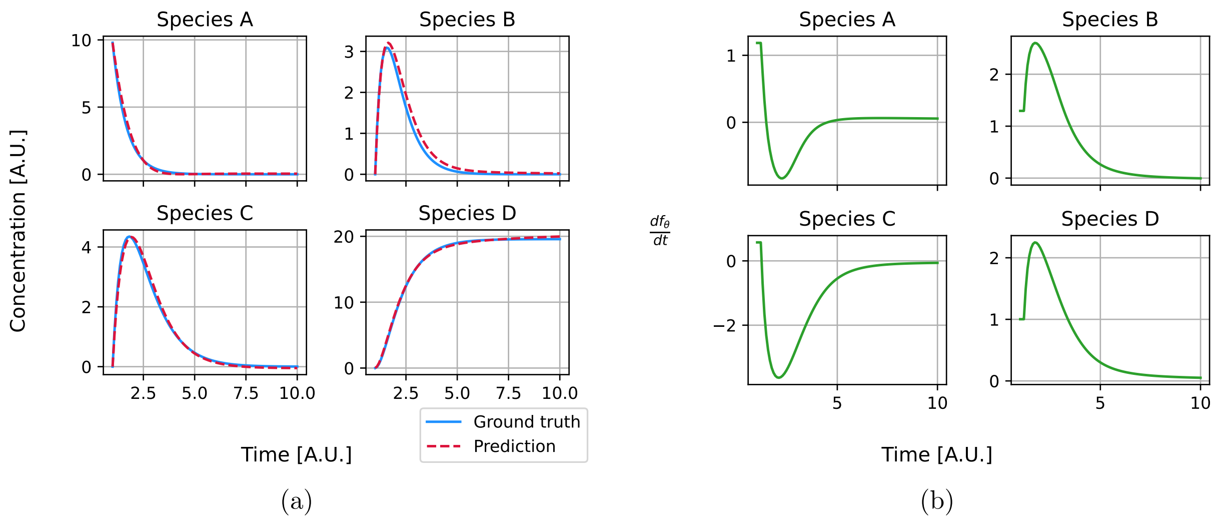


Figure B.25: The modelling performance (a) and neural network contribution (b) of the neural ODE using a LSTM on the four species within the missing reactions reaction network.