AI-Driven Constrained Mathematical Models of Biological Systems

A thesis report
submitted in partial fulfillment of the requirements
for the award of the degree of

M.Tech. in Computational Biology

by

Alka Singh (MT23224)

Under the supervision of

Dr. Sriram K.

Department of Computational Biology,
Indraprastha Institute of Information Technology Delhi
Okhla Industrial Estate, Phase III, New Delhi, India – 110020
July, 2025

Certificate

This is to certify that the thesis entitled "AI-Driven Constrained Mathematical Models of Biological Systems" **submitted** by Alka Singh to the Indraprastha Institute of Information Technology, Delhi, for the award of the degree of **Master of Technology**, is an original research work carried out by her under my supervision. In my opinion, the thesis **meets the required standards and fulfills** all regulations related to this degree

The results presented in this thesis have not been submitted, in whole or in part, to any other university or institute for the award of any degree or diploma.

Date: 29 JULY, 2025

Dr. Sriram K.
Department of Computational Biology,
Indraprastha Institute of Information
Technology Delhi, Okhla Industrial Estate-
Phase III New Delhi, India – 110020

Acknowledgements

*I would like to express my heartfelt gratitude to **Dr. Sriram K.** for his invaluable guidance and support throughout the course of my M.Tech thesis. His expertise, encouragement, and thoughtful feedback have played a crucial role in shaping the direction and quality of my work. I am especially grateful for the freedom he provided to explore ideas independently, make mistakes, and learn from them. His mentorship not only helped me navigate complex problems but also instilled a sense of confidence and intellectual curiosity that will continue to guide me in the future.*

*I would also like to acknowledge **Harika G L**, a Ph.D. student under Dr. Sriram K., for the insightful discussions that contributed significantly to this work.*

*My sincere thanks go to my classmate **Shaikh Shuhail**, whose steady support in understanding biological concepts was invaluable in bridging my background as a non-biologist. I am equally grateful to **Prasanna Kumar S.** for his assistance with technical implementation at various stages of the thesis.*

*Additionally, I would also like to acknowledge the use of online resources, including **YouTube lectures**, various digital content platforms, and **OpenAI's ChatGPT**, which aided my understanding of interdisciplinary topics. I utilized OpenAI's ChatGPT specifically for assistance in creating, formatting, and refining tables and figures included in this thesis.*

*Finally, I am deeply thankful to **my family** for their unwavering love, patience, and encouragement, which have been a constant source of strength throughout this journey.*

Sincerely,

Alka Singh
(MT23224)

Abstract

Biological systems are inherently complex, governed by nonlinear dynamics, feedback loops, multi-scale interactions, and hidden regulatory mechanisms. Accurately modeling such systems is essential for understanding processes like cellular growth, differentiation, signaling, and immune response. Traditional systems biology approaches rely on prior mechanistic knowledge to construct models using ordinary differential equations (ODEs). While powerful in well-characterized scenarios, these models are often difficult to apply in data-rich but poorly understood systems, as they require manual specification of the governing equations and assume complete knowledge of interactions. To overcome these limitations, we propose a hybrid data-driven framework that integrates classical dynamical systems theory with modern sparse regression techniques. Our approach begins with Takens' embedding to reconstruct the system's dynamics from time-series data using delay coordinates. We then apply Sparse Identification of Nonlinear Dynamical Systems (SINDy) to infer minimal and interpretable ODEs that govern the observed dynamics. Recognizing the prevalence of rational nonlinearities in biological systems, such as Michaelis–Menten and Hill kinetics, we extend our method using Implicit SINDy (SINDy-PI), which can identify models involving algebraic constraints and rational terms. We validate our framework on synthetic time-series data generated from a range of well-known biological models, including the FitzHugh–Nagumo, Goodwin, Oregonator, glycolytic oscillators, mass-action systems, Michaelis–Menten kinetics, and microbial growth dynamics. The results demonstrate accurate recovery of the underlying dynamical structure, phase space features, and network topologies. To assess real-world applicability, we apply the framework to two biologically relevant systems: (1) the eukaryotic cell cycle and (2) tumor–immune interactions relevant to cancer immunotherapy. From synthetic data emulating experimental observations, we successfully rediscover governing equations that predict system behavior and reveal key regulatory motifs. Sensitivity analysis and bifurcation techniques further confirm that the inferred models exhibit known transitions and critical points consistent with biological expectations. In summary, our method bridges the gap between mechanistic modeling and data-driven inference. It offers a unified, interpretable, and scalable approach to uncovering meaningful biological dynamics from time-series data. This work advances the field of systems biology by enabling dynamic modeling in settings with limited mechanistic knowledge, ultimately supporting improved understanding, hypothesis generation, and therapeutic design.

Keywords: Systems biology, Nonlinear dynamics, Data-driven modeling, ODEs, Takens' embedding, Sparse regression, SINDy, SINDy-PI, eukaryotic cell cycle, Tumor–immune interaction, Sensitivity analysis, Bifurcation analysis.

Contents

List of Figures

List of Tables

Abbreviations

ODE	**O**rdinary **D**ifferential **E**quation
ACF	**A**uto- **C**orrelation **F**unction
FNN	**F**alse **N**earest **N**eighbor
SVD	**S**ingular **V**alue **D**ecomposition
PCA	**P**rincipal **C**omponent **A**nalysis
SINDy	**S**parse **I**dentification of **N**onlinear **Dy**namics
SINDy-PI	**S**parse **I**dentification of **N**onlinear **Dy**namics - **P**arallel **I**mplicit
STLSQ	**S**equentially **T**hresholded **L**east **SQ**uares
LASSO	**L**east **A**bsolute **S**hrinkage and **S**election **O**perator
SR3	**S**parse **R**elaxed **R**egularized **R**egression
MSE	**M**ean **S**quared **E**rror
AIC	**A**kaike **I**nformation **C**riterion
BIC	**B**ayesian **I**nformation **C**riterion
CAR-T	**C**himeric **A**ntigen **R**eceptor **T** cell
FDA	**F**ood and **D**rug **A**dministration
XPPAUT	**X** Windows **P**hase **P**lane + **AUT**O
COPASI	**CO**mplex **PA**thway **SI**mulator

Part I

Chapter 1

Introduction

1.1 Motivation and Background

Biological systems are inherently complex and dynamic in nature. They consist of networks with many interacting components, including genes, proteins, and metabolites. These networks govern many essential processes for life, including cellular growth, differentiation, immunological response, and many more. In order to understand how such networks regulate cellular behavior and respond to any internal or external stimuli, we require mathematical modeling [2].

Mathematical modeling offers a robust framework for representing, evaluating, and predicting biological behavior. It allows researchers to identify hidden mechanisms, test hypotheses, and investigate emergent dynamic behaviors like oscillations, multistability, and chaos that are often difficult to quantify directly through experiments [3].

Traditionally, these models were constructed using first principles, where the functional forms were derived from biological knowledge, validated through controlled experiments, and typically put into the form of ordinary differential equations (ODEs). These mechanistic approaches, grounded in the **principle of parsimony (Occam's razor)** [2], have resulted in many foundational models in systems biology. Although the traditional method has provided significant insights in several biological situations, it fails to capture the full complexity of living systems.

Also, in contrast to physics, which is grounded in well-defined fundamental principles, biology often involves complex structures, variability, and incomplete knowledge. Also, recent advancements in high-throughput technologies such as RNA sequencing, proteomics, single-cell analysis, and live-cell imaging have revolutionized experimental biology. These technologies generate large-scale, high-resolution temporal and spatial datasets, providing great insights into biological processes. However, they also offer a significant challenge: **how can one manually extract and formulate the underlying patterns from such kind of high-dimensional, noisy, and nonlinear data?** [4, 5]

Furthermore, the functional complexity of biological systems arises from the complex structure of their regulatory and metabolic networks. **These networks exhibit distinct patterns,**

including feedback loops, repeated interaction motifs, and sparse connectivity, rather than being randomly connected. The structured design of such networks is essential for maintaining proper cellular activity. Disruptions in these networks, whether due to mutations or environmental stresses, can result in illnesses such as cancer, metabolic syndromes, and autoimmune disorders [6, 7]. Understanding how these networks behave over time and how their dynamics respond to perturbations, is not only theoretically important but also essential for the development of improved therapies, diagnostics, and therapeutic interventions. This highlights the essential need for accurate models that can capture not only the topology of these networks but also their time-dependent dynamics [8]

Determining such models directly from data using classical modeling approaches typically requires researchers to guess the right equations from a set of possible equations and then refine them through trial and error. This is time-consuming and often not feasible in situations where systems become more complex and nonlinear. As a result, the growing complexity and data abundance in modern biology create new opportunities for model building and require automated ways of modeling [4, 5].

Among the emerging techniques, **the Sparse Identification of Nonlinear Dynamics (SINDy) framework is a novel approach that integrates nonlinear function libraries with sparse regression to extract governing equations from data** [5, 9]. Unlike black-box machine learning models that lack interpretability, SINDy produces structurally simple, white-box models that are mathematically tractable and capable of capturing biologically meaningful dynamics.

This thesis lies at the intersection of conventional mathematical modeling and modern data-driven discovery of ODE-based biological models. It examines how techniques like SINDy and its variants may be used to build interpretable models of biological networks directly from experimental time-series data. By integrating sparse regression approaches with domain-specific information, we aim to bridge the gap between abstract dynamical theory and the realistic modelling of biological networks.

1.2 Mathematical Tools for Analyzing Dynamical Systems

While data-driven frameworks like **SINDy** enable us to **recover dynamical models** directly from the biological data, the resultant systems are **often expressed as coupled nonlinear ordinary differential equations (ODEs)**, which describe how biological components evolve over time. However, finding equations is only the initial step. **To understand and make inferences from these models, we must examine their qualitative behavior**. This requires several mathematical tools capable of revealing properties such as steady states, oscillations, multistability, and chaotic dynamics [10].

This section examines the essential mathematical tools required for the analysis of ODE-based models. This includes regulatory functions that explain the interactions among molecular components, steady-state analysis for identifying equilibrium conditions, Jacobian-based

methods for determining local stability, and bifurcation theory to analyze qualitative shifts in dynamics in response to parameter variations.

1.2.1 Regulation

Regulation refers to the control of one biological component by another through certain molecular interactions. It is essential for determining many events, like the timing and rate of gene expression, protein synthesis, and the production or degradation of metabolites [11].

Regulation can be broadly classified as

- **Positive regulation (activation)**, where a component promotes the activity of another. For example, a transcription factor may enhance the transcription of a particular gene, resulting in increased mRNA and protein synthesis [11].
- **Negative regulation (inhibition)**, where a component suppresses the activity of another. A typical example is end-product inhibition, when a final product of a metabolic pathway inhibits the activity of an upstream enzyme, hence preventing excessive synthesis [11].

These regulatory influences can be transient or permanent, and their collective interplay gives rise to feedback loops, which can give rise to phenomena such as switch-like transitions or oscillations, which will be discussed in the subsequent sections.

Regulatory Functions

Regulatory effects are incorporated into models using some mathematical expressions termed as regulatory functions. These functions characterize the relationship between input (concentration of regulator) and output (rate of production of target). The **Michaelis–Menten equation**, and the **Hill function** provide biologically plausible frameworks for incorporating enzymatic and regulatory mechanisms.

(1) **Michaelis–Menten equation:** The Michaelis–Menten equation is commonly used to model enzymatic reactions. It describes how the rate of a reaction increases with substrate concentration and saturates at high levels. The equation is typically written as:

$$\boxed{v := v(S) = \frac{v_{\max} S}{K_M + S}} \tag{1.2.1}$$

where:

- S is the substrate concentration,
- $v_{\max}$ is the maximum reaction rate,
- K_M is the Michaelis constant (substrate concentration at which the rate is half of $v_{\max}$).

(2) **The Hill Function:** While Michaelis–Menten kinetics describes how reaction rates saturate at high substrate concentrations, the Hill function (for activation) extends it by incorporating co-operativity. In particular, it captures the phenomenon of ultrasensitivity, where the system shows a relatively flat response for small input levels, followed by a sharp transition when a certain threshold is reached (**Figure 1.1**). This behavior resembles a biological switch. Typically, two types of Hill functions are utilized when modeling a biological problem:

$$\boxed{H^{+}(\mathrm{x}, K, n) := \frac{x^n}{K^n + x^n} \quad \text{(Activation)}} \tag{1.2.2}$$

$$\boxed{H^{-}(\mathrm{x}, K, n) := \frac{K^n}{K^n + x^n} \quad \text{(Inhibition)}} \tag{1.2.3}$$

where x is the regulator concentration, K is the half-maximal value, and n is the Hill coefficient controlling cooperativity [12].

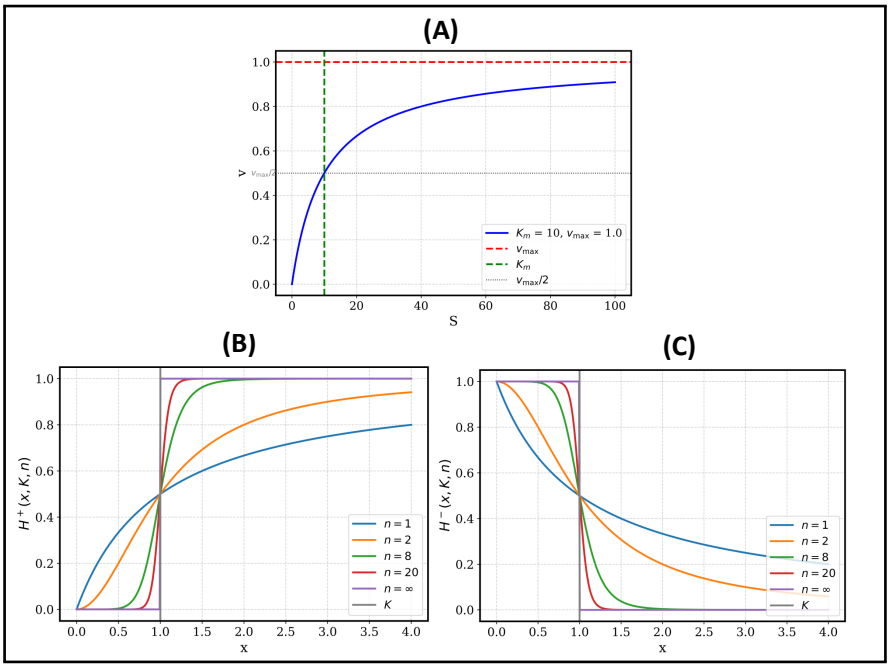

Figure 1.1. *Response curves for enzyme kinetics and regulatory Hill functions. (A) represents the classic Michaelis–Menten kinetics, which shows a hyperbolic relationship between substrate concentration S and reaction velocity v. The curve rises rapidly at low substrate concentrations and gradually saturates as it approaches the maximum velocity $v_{\max}$. The dashed vertical line (green) corresponds to the Michaelis constant K_m, where the reaction rate reaches half of $v_{\max}$, shown in red. (B) and (C) represent the response of Hill functions for activation (H^+) **(S– shaped)** and inhibition (H^-) **(Z– shaped)** respectively. For $n = 1$, the response is graded (Michaelian). As n increases, the response becomes increasingly switch-like, resembling a step function in the limit $n \to \infty$. The threshold parameter K determines the concentration at which the effect is half-maximal.*

1.2.2 Differential Equation Modeling in Biological Systems

Mathematical models based on differential equations are essential tools for representing the dynamic behavior of biological systems. Once the regulatory interactions between components have been determined, ordinary differential equations (ODEs) may be used to describe the temporal evolution of a system. These equations determine how the concentration of each component varies over time due to the combination of synthesis, degradation, and regulatory interactions.

Consider a biological system consisting of n interacting components (species) say $x_1, x_2, \ldots, x_n$. (e.g., concentrations of proteins, mRNA, or metabolites), and let $\mathbf{k}$ denote a set of parameters (e.g., rate constants, thresholds). The time evolution of the system may then be represented by the following set of **autonomous ordinary differential equations** (ODEs) [10]:

$$\frac{dx_i}{dt} = F_i(\mathbf{x}, \mathbf{k}), \quad i = 1, 2, \ldots, n. \tag{1.2.4}$$

where

- $\mathbf{x} = (x_1, x_2, ...x_n)$, and $\mathbf{k}$ is a vector of non-negative parameters.
- $F_i : \mathbb{R}^n \to \mathbb{R}$ can be a combination of linear and nonlinear terms that describes the net effect of regulating factors due to other variables on x_i.

1.2.3 Steady States and Stability Analysis

A steady state represents a condition in which the system exhibits no net change with respect to time. Mathematically, a point $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$ is called a **steady state** (or dynamic equilibrium point) for the system given by **Eq.**(1.2.4) if

$$F_i(\mathbf{x}^*, \mathbf{k}) = 0, \quad \text{for all } i = 1, 2, \ldots, n. \tag{1.2.5}$$

It is important to note that the steady-state concentrations $x_i^*, \quad i = 1, 2, ...n$ are dependent on the parameters $k_\alpha \in \mathbf{k}$ and must remain non-negative for all i, as negative concentrations have no physical meaning.

A steady state $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$ is said to be *stable* if small perturbations around it decrease over time, bringing the system back to the steady state.

Jacobian Matrix and Linear Stability Analysis

To determine the stability of a steady state, the system of ODEs given by **Eq.**(1.2.4) is linearized in a neighborhood of the steady state, resulting in the following linearized equation:

$$\frac{dx_i}{dt} \approx \sum_j a_{ij} x_j \tag{1.2.6}$$

where

$$a_{ij} = \left.\frac{\partial F_i}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}^*}.$$

are the partial derivatives calculated at the steady state. The system (1.2.4) and (1.2.6) exhibit similar dynamics near the steady states [13]. The partial derivatives in (1.2.6) constitute a matrix $J = [a_{ij}]_{n\times n}$, known as the Jacobian matrix or, sometimes, the community matrix in ecological literature. Its interpretation has been provided in **Box 1.2.3** [14].

Interpretation of the Jacobian Matrix J

The elements a_{ij} of the Jacobian matrix measure the influence of component j on the dynamics of component i near a steady state.

Diagonal Elements:

1. $a_{ii} = 0$: Species i does not influence its own rate of change to the first order.
2. $a_{ii} \neq 0$: Species i exerts self-regulation.

Off-Diagonal Elements:

1. $a_{ij} > 0$: An increase in x_j increases dx_i/dt (activation).
2. $a_{ij} < 0$: An increase in x_j decreases dx_i/dt (inhibition).
3. $a_{ij} = 0$: No interaction from j to i.

According to the theory of linear differential equations, the solutions to the linearized system (1.2.4) can be expressed as a superposition of exponential terms of the form, $e^{\lambda_j t}$ where $\{\lambda_j\}$ denotes the set of eigenvalues of the Jacobian matrix, which determines the stability of the steady state [15]:

- If all eigenvalues have negative real parts, perturbations decrease, and the matrix J is referred to as (qualitatively) stable.
- If either eigenvalue has a positive real part, the corresponding perturbations will amplify with time, making the matrix J (qualitatively) unstable [15].

1.2.4 From Local Stability to Bifurcation Analysis

While the Jacobian provides local information on the stability of stable states, it does not account for how system behavior changes when parameters are varied. A small change in parameters, such as the feedback strength or reaction rates, may cause a huge shift in the dynamics of biological systems. This motivates the study of **bifurcations**.

A bifurcation is a qualitative change in the behavior of a dynamical system as a parameter

crosses a critical value. The parameter value at which this change occurs is referred to as a **bifurcation point**.

A plot of steady states versus a parameter (keeping others fixed) is called a bifurcation diagram. This diagram highlights the parameter regions where the model exhibits biologically relevant behaviors [16, 17]. **Table 1.1** provides the summary of different types of bifurcations along with their characteristics.

Table 1.1. *Common bifurcations and their biological significance.*

Bifurcation Type	**Biological Interpretation**
Saddle-Node Bifurcation	Two fixed points of opposite stability collide and annihilate each other (or are created) as a parameter is varied. This is often observed in bistable gene switches [18, 19].
Transcritical Bifurcation	Two fixed points exchange their stability when they intersect as a parameter changes. Activation of a gene that can be both on or off depending on conditions [11, 16].
Pitchfork Bifurcation	A symmetric equilibrium gives rise to multiple asymmetric branches as a parameter changes. Often used for modeling symmetry breaking in cell fate decisions [16, 17].
Hopf Bifurcation	A fixed point loses stability, and a stable or unstable limit cycle emerges. This transition explains the onset of biological oscillations, such as cell cycle oscillations [18, 20].
Saddle-Node Infinite Period (SNIPER) Bifurcation	A saddle and a stable node collapse on a closed orbit as a parameter is varied. Often used to explain slow, size-dependent triggering of cell cycle transitions [18, 19]

Bifurcation analysis is often performed using numerical continuation software such as `XPPAUT` [21] or `MATCONT` [22], particularly for nonlinear and high-dimensional systems.

1.2.5 Feedback Loops and Feedback Analysis

A feedback loop is **a closed chain of regulatory interactions** in which a component eventually influences its own activity, either directly or indirectly. The existence of these loops is crucial in determining the qualitative behavior of biological systems. Depending on their configuration (e.g., positive, negative, or mixed), feedback loops give rise to a diverse range of dynamic phenomena, shown in **Table 1.2** [11, 15, 23].

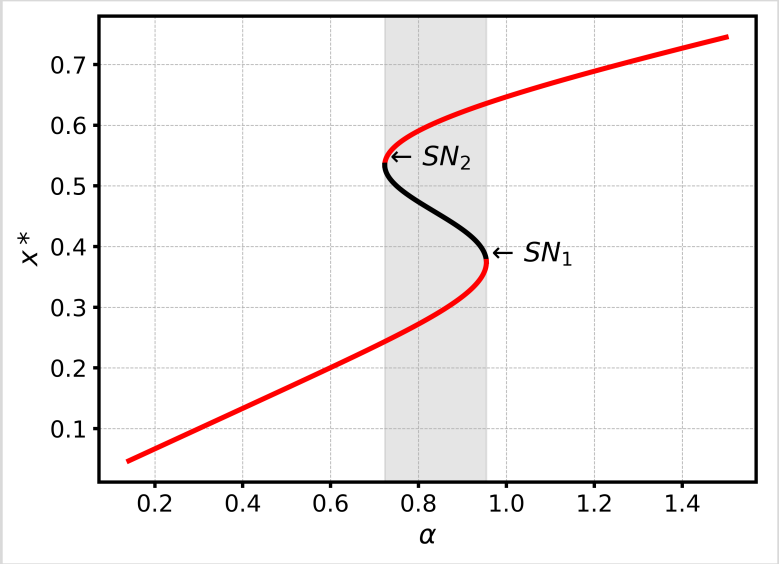

Figure 1.2. *An illustration of a bifurcation diagram. A codimension-one bifurcation diagram showing how the steady-state x* of a dynamic variable x changes with respect to the parameter α. The red branches correspond to stable steady states, while the black branch indicates unstable steady states. The two turning (fold) points SN_1 and SN_2 correspond to saddle-node bifurcations* ***(Table1.1)****. The shaded region between these two represents the region of bistability where a parameter gives rise to one unstable and two stable steady states. For any given value of the parameter in this region, the system can reside in either a high or low stable state, depending on its history.*

Table 1.2. *Fundamental dynamical behaviors in biological systems arising from feedback regulation. Each behavior reflects a distinct class of system dynamics driven by specific regulatory motifs.*

Dynamical Behavior	**Role of Feedback Loops**
Homeostasis	Requires at least one negative feedback loop (necessary) to maintain internal stability by responding to external perturbations [11].
Limit Cycle Oscillations	Requires a negative feedback loop of length $\geq$ (necessary). Along with this, delays and possibly nonlinearity in the regulation functions can make the system sufficient to exhibit sustained oscillations [11].
Multistability	Requires at least one *positive feedback loop* (necessary). Additionally, if the loop is *functional* under appropriate parameter regimes, then the system can exhibit multistability. [11].
Chaos	Requires both nonlinearity and at least one negative feedback loop, often in high-dimensional systems. Chaos is not guaranteed unless certain sensitive dependence conditions are met [11, 24].

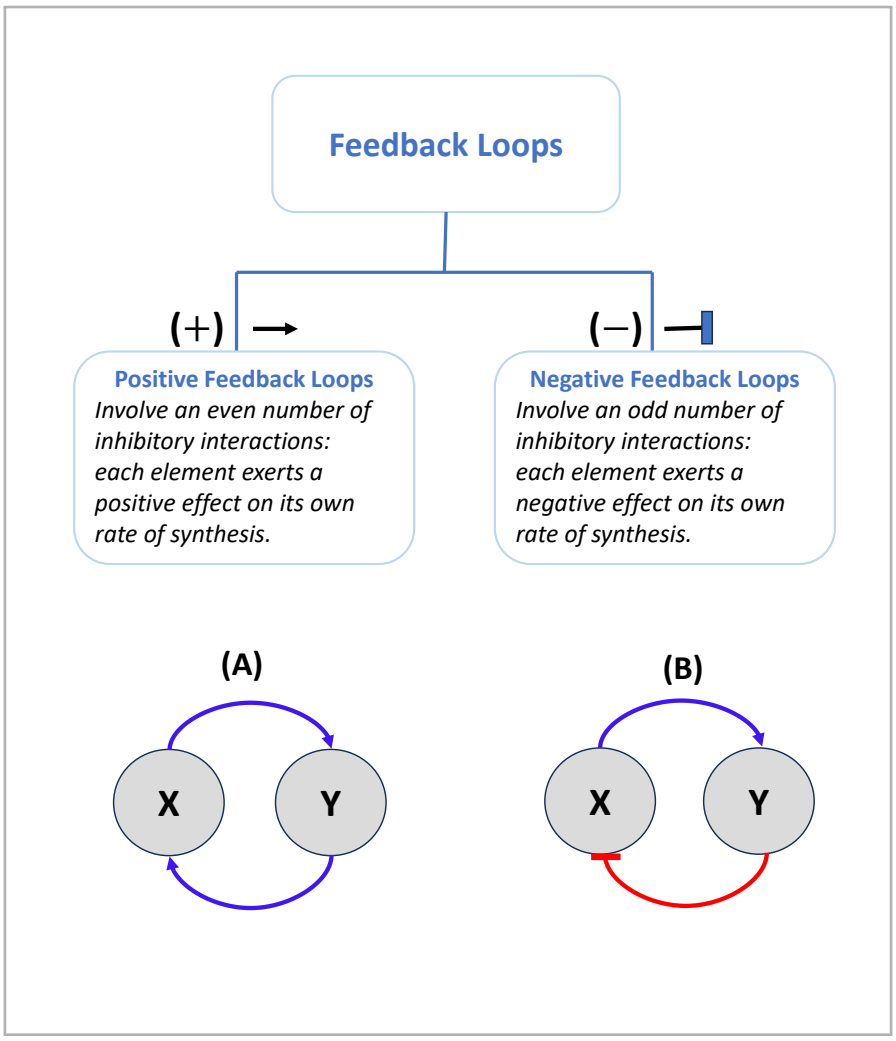

Figure 1.3. *Classification of feedback loops. (A) A simple positive feedback loop. (B) A simple negative feedback loop. Blue and red arrows represent activation and inhibition, respectively.*

Mathematically, **a feedback loop near a steady state is represented by an interaction graph, often derived from the sign-pattern of the Jacobian,** where

- **Nodes** represent components (state variables) of the system.
- **Edges** represent interactions between these components, such as activation $(+)$ or inhibition $(-)$. **Arrows** $(\rightarrow)$ are used to indicate activation, while **blunt ends** $(\dashv)$ are used to represent inhibition [23].

Identification of Local Feedback Loops using Jacobian

Any set of nonzero elements in the Jacobian matrix **J** defined by **Eq.**(1.2.6) that constitutes **a circular permutation of indices** $i, j, k, \ldots$ **defines a feedback loop**. Moreover, its sign is determined by the product of the signs of these terms.

- If the Jacobian/ interaction graph contains a positive loop, the system may exhibit multistability.
- If it contains a negative loop of length ≥ 2, the system may exhibit sustained oscillations.

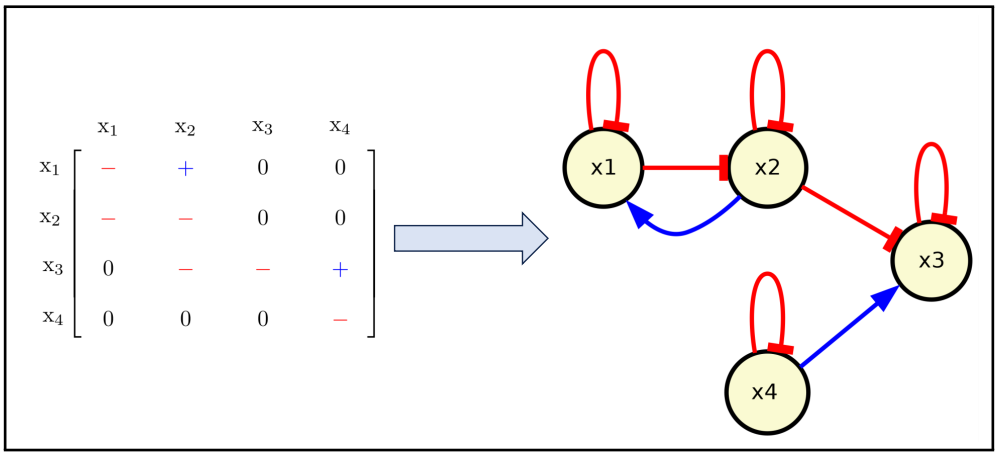

Figure 1.4. *Local interaction network using signs of Jacobian*

1.2.6 Sensitivity Analysis

Sensitivity analysis is a fundamental tool used in dynamic modeling to assess the effect of changes in model parameters on the behavior of a system. In particular, it evaluates how small perturbations in parameter values (cause) affect key outputs (effect). In this way, it helps in determining which parameters are most critical to model outcomes. This is particularly relevant in biological systems, where accurate values of parameters may be difficult to measure or are subject to inherent biological variability [25].

In this thesis, we have used COPASI (COmplex PAthway SImulator) [26] to perform local sensitivity analysis (in chapters 8 and 9) under steady-state conditions to determine which parameters have the greatest influence on the steady-state levels of the system variables. This step can reveal parameters (leverage points) whose perturbation can cause major qualitative shifts in system dynamics, such as transition of states, loss of homeostasis, or emergence of oscillations. Identification of such parameters can guide experimental design or therapeutic targeting in biological systems. The analysis has been carried out using the configuration illustrated in **Figure 1.5**.

1.3 Statement of The Thesis Problem

Classical and mechanistic approaches for modeling biological systems often start with the identification of network motifs, assigning them feedback structures based on biological theories, and then formalizing them into mathematical models. These models are typically represented as differential equations whose parameters are fitted through experiments. The

resultant models, then, are used to simulate system behavior and extract insights for making predictions about the system's behaviors.

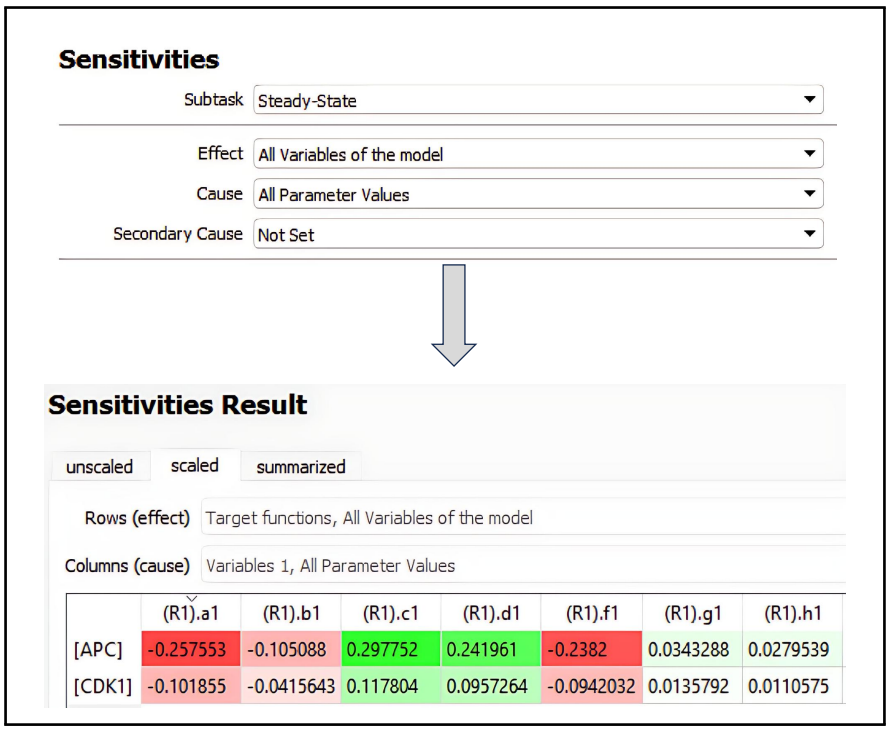

Figure 1.5. *An illustration of a local sensitivity analysis performed in COPASI. Each output species is monitored at steady state, and all kinetic parameters in the model are perturbed. In this setting, when we run sensitivity, COPASI calculates a 2D sensitivity matrix S, where each entry $S_{ij} = \frac{\partial x_i}{\partial p_j}$ measures the change in steady-state concentration x_i (rows) in response to a small change in parameter p_j (columns). The lower panel shows scaled sensitivities computed as Scaled $S_{ij} = \frac{\partial x_i}{\partial p_j} \cdot \frac{p_j}{x_i}$. The scaled form is dimensionless, facilitating comparison across parameters with different units or magnitudes.*

This thesis aims to address the inverse problem: using time-series data to identify the governing differential equations and then deduce the dynamic behaviors, including steady states and regulatory interactions. To accomplish this, we develop a reproducible pipeline that integrates phase-space reconstruction, sparse model identification, and symbolic analysis to extract interpretable equations from time series data. This pipeline is designed to overcome several challenges inherent in data-driven modeling of biological systems:

- **Handling hidden variables:** In many biological systems, not all variables are measurable. To solve this, the pipeline leverages phase-space reconstruction methods based on Takens' embedding theorem, which allow for the reconstruction of the whole system dynamics from single-variable time-series data. The PyTISEAN tool is used to evaluate the reconstruction against known examples.
- **Discovering governing equations:** The pipeline uses the Sparse Identification of Nonlinear Dynamics (SINDy) framework and its variant, Implicit SINDy (SINDy-PI) to find

interpretable ODE models using `PySINDy` Python package.

- **Biological interpretation:** The inferred models are examined to determine steady states, interaction graphs, bifurcation diagrams, and sensitivity profiles. This is accomplished through symbolic computation and numerical simulation. Specifically, we used `SymPy` for any symbolic compuation and Jacobian-based analysis; `Graphviz` and drawing the feedback loops; `XPPAUT` [21] to generate bifurcation diagrams, and COPASI (COmplex PAthway SImulator) [26] to perform local sensitivity analysis.

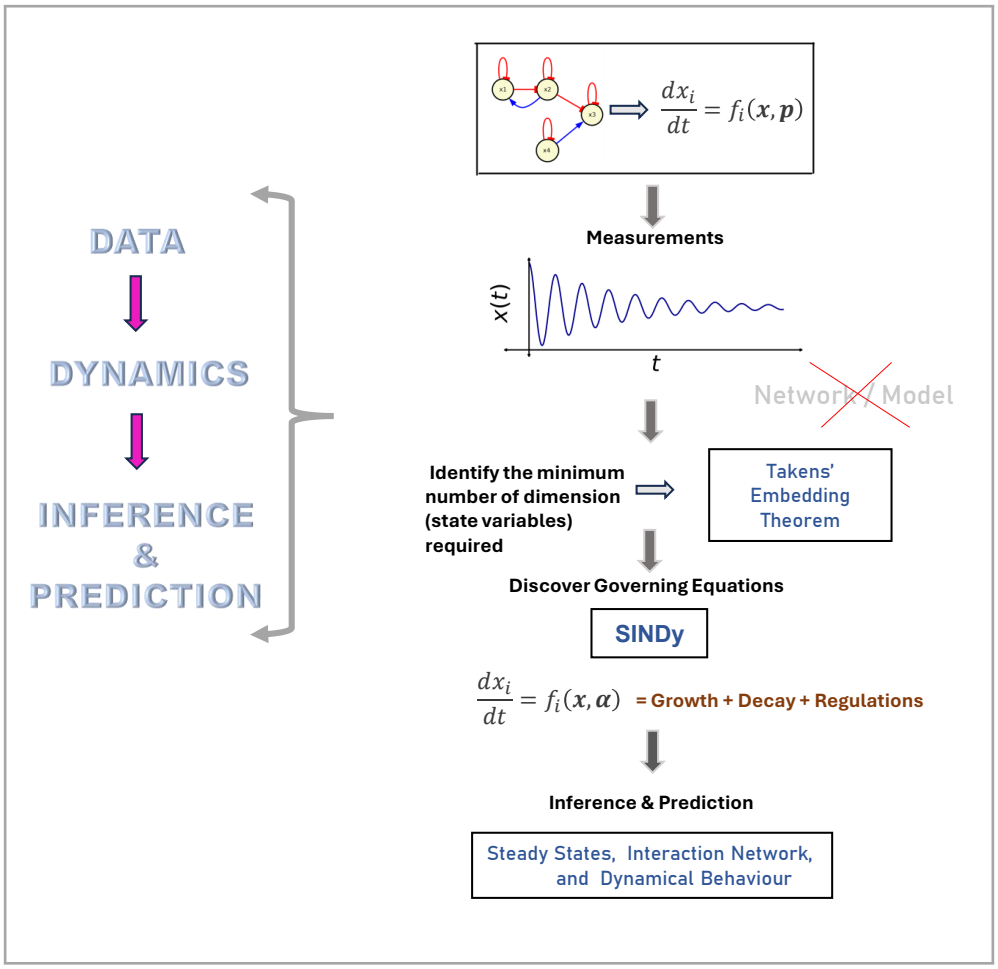

Figure 1.6. *Schematic of the thesis problem and proposed solution framework.*

1.4 Thesis Organization

The thesis is divided into two parts: the first part focuses on the methodology development and implementation (Takens' embedding theorem, SINDy, SINDy-PI). The second part deals with the application of these techniques to biological systems like the cell division cycle and cancer immunotherapy. A schematic overview is provided in **Figure 1.7**.

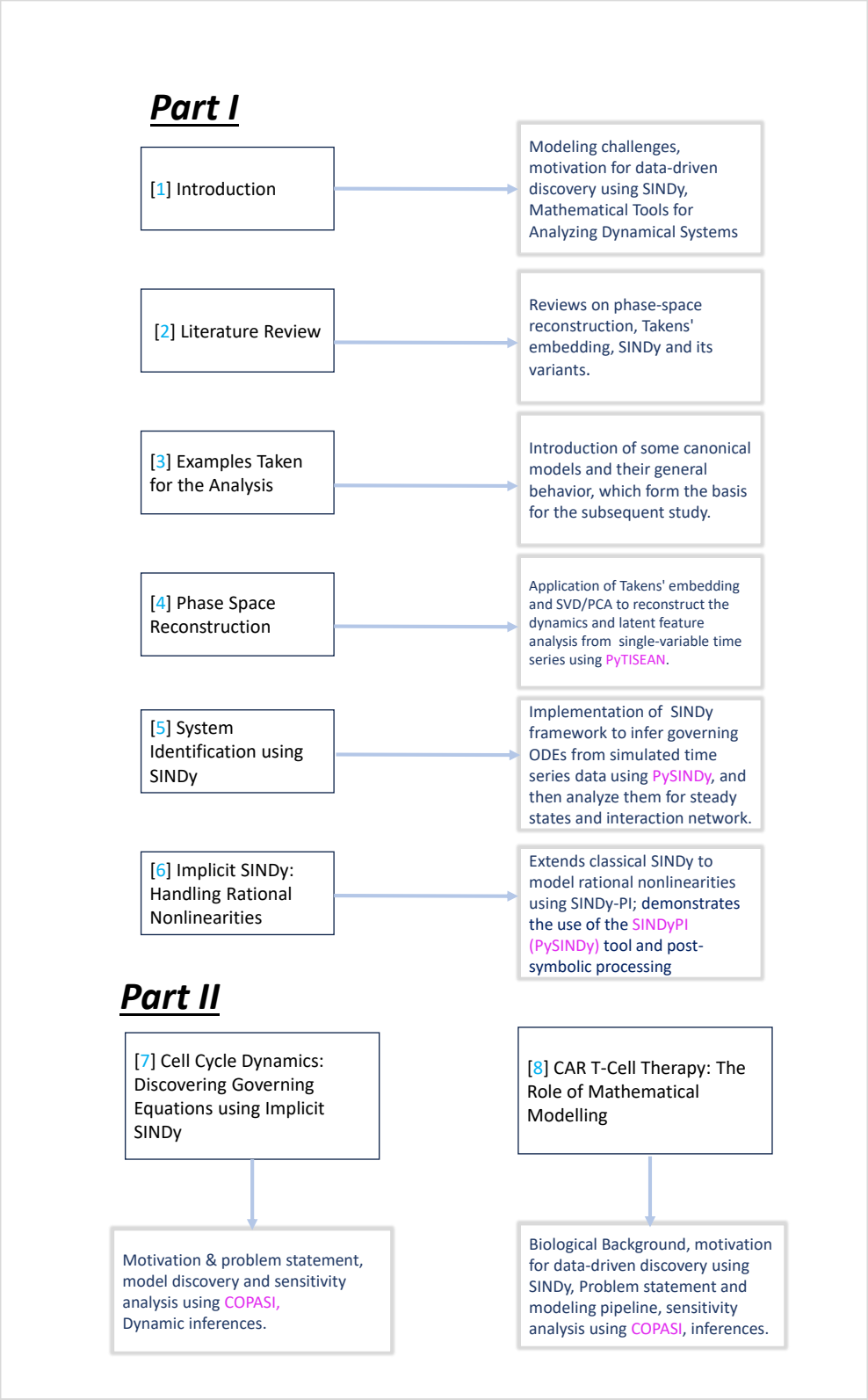

Figure 1.7. *Overall structure of the thesis.*

Chapter 2

Literature Review

2.1 Introduction

Modeling biological systems is crucial for understanding cellular growth, their response to external or internal signals, and their internal regulation [11]. Traditionally, researchers have relied on mechanistic models, usually formulated as sets of ordinary differential equations (ODEs), to describe known biochemical interactions. Classical models such as the FitzHugh-Nagumo model for excitability [27, 28], the Goodwin oscillator for gene regulation [29, 30], and mass-action-based models for reaction kinetics [31–33] have revealed interesting dynamic phenomena like bistability, oscillations, and feedback control. However, these models rely on a detailed knowledge of the system, like which molecules interact, what is the strength and sequence of their interactions. In several biological systems, such a level of mechanistic detail is often incomplete, noisy, or inaccessible. Consequently, even established models may not properly represent the underlying complexity of cellular activity.

Recent advancements in biological technologies, such as RNA sequencing, proteomics, single-cell analysis, and live-cell imaging have enhanced the accessibility of time-resolved biological data. Although these datasets provide remarkable insight into cellular function, they also increase the difficulty of modeling tasks, particularly due to noise, partial observability, and high dimensionality [4, 5]. To address this, two complementary approaches have evolved. The first approach addresses imperfect measurements and identifying the number of dimensions by reconstructing the system's underlying dynamics utilizing techniques such as delay embedding and dimensionality reduction [34, 35]. The second approach employs data-driven modeling, namely sparse regression approaches such as SINDy, to automatically discover governing equations without requiring precise mechanistic assumptions [9].

Together, these approaches form a powerful foundation for modern data-driven modeling in biology. This chapter reviews techniques for reconstructing system dynamics from incomplete measurements and determining governing equations from data, emphasizing how combining them facilitates the development of interpretable models that may reveal underlying dynamics.

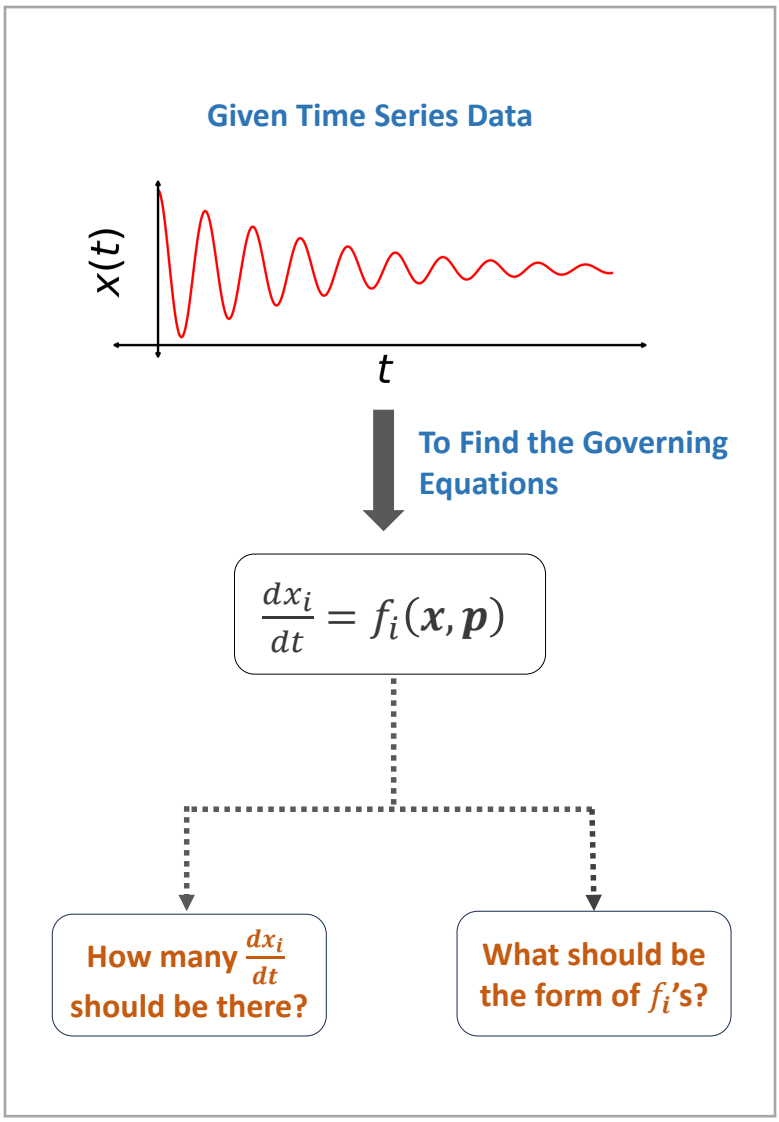

Figure 2.1. *Schematic overview of the data-driven model discovery process. Given time-series data $x(t)$, the goal is to infer a system of governing equations of the form $\frac{dx_i}{dt} = f_i(\mathbf{x}, \mathbf{p})$. Two key challenges are: determining the appropriate number of state variables (dimensionality), and identifying the functional form of each right-hand-side expression f_i.*

2.2 State Space Reconstruction and Latent Feature Analysis

A frequent problem in experimental biology is that only a subset of the variables influencing a dynamical system can be measured directly [5]. Additionally, many data-driven approaches require an appropriate dimension, and therefore, the minimal number of relevant state variables is necessary for accurate and interpretable model identification. Takens' embedding theorem serves as the theoretical foundation for this [34], which helps in recovering the underlying dynamics by reconstructing the system's state (phase) space from a scalar observable. It states that, under generic conditions, the time evolution of an n-dimensional continuous dynamical

system can be accurately represented in a delay-coordinate space derived from any sufficiently lengthy scalar time series [34, 36].

In practice, one creates vectors

$$\mathbf{x}(t) = \big[x(t),\ x(t-\tau),\ x(t-2\tau), \ldots, x(t-(m-1)\tau)\big]$$

whose components are copies of the observable $x(t)$, that have been shifted in time for the reconstruction. If the embedding delay τ and the embedding dimension m are chosen correctly, the reconstructed system preserves the important topological and dynamical properties of the original system.

However, successful phase space reconstruction depends on choosing the correct embedding parameters, time delay (τ) and embedding dimension (m). τ is often estimated using the autocorrelation function (ACF) [37], and m is determined using methods like false nearest neighbors (FNN) [38].

These standard methods assume a single, constant delay (τ) for all dimensions, which may be suboptimal for systems with multiple time scales, such as fast-slow systems, and may fail to capture cases where different state variables evolve at different rates. Additionally, autocorrelation and FNN depend on statistical heuristics that do not always generalize well to strongly nonlinear or chaotic systems. The FNN algorithm requires defining a threshold to classify points as false neighbors. Selecting this threshold is subjective and can affect the final embedding dimension [39].

Despite its utility, delay coordinate embedding does not inherently prioritize the **most relevant features and/or correct embedding dimension of the system**. To reduce dimensionality and extract dominant structures, techniques like Singular Value Decomposition (SVD) and Principal Component Analysis (PCA) have been widely applied to the embedded data obtained from the delay coordinate embedding [35, 37].

2.3 Data-Driven Discovery: SINDy

As discussed earlier, the classical way of modeling is often based on first-principles derivation, where we write equations from known biochemical reactions, and such mechanistic details are typically inadequate, noisy, or experimentally inaccessible [5]. This constraint, the growing complexity, and data abundance have motivated a shift toward data-driven approaches that can discover the dynamic equations directly from experimental observations.

Early efforts in this area utilized symbolic regression techniques, including evolutionary algorithms and genetic programming. These approaches seek to derive closed-form equations from data without requiring prior assumptions on the model structure. While promising in theory, they are computationally expensive and frequently prone to overfitting, particularly when the data is noisy or high-dimensional.

Brunton *et al.* [9] presented a more scalable and interpretable alternative known as the

Sparse Identification of Nonlinear Dynamics (SINDy) framework. SINDy assumes that several dynamical systems are regulated by a sparse set of dominant interactions. This assumption is valid for several biological networks when only a limited number of variables affect the system at a particular time [9, 40]. Also, it requires access to the time series of all relevant dynamical variables in order to identify the underlying governing equations. The number of time series required can be determined using time-delay embedding, which helps identify the minimum number of state variables needed to reconstruct the system's dynamics. SINDy formulates the model discovery problem as a sparse regression problem, where time derivatives $\dot{x}_k$ for each k, are expressed as a linear combination of candidate nonlinear functions in a predefined library $\Theta(\mathbf{x})$:

$$\dot{x}_k = \Theta(\mathbf{x})\,\boldsymbol{\xi}_k, \qquad \text{where } \boldsymbol{\xi}_k \text{ is sparse vector of coefficients for } k^{th} \text{ equation..}$$

Methods like sequential thresholding, ridge [41] and LASSO [42] are utilized to eliminate redundant terms, yielding simple and interpretable models that capture the system's fundamental dynamics.

2.4 Implicit-SINDy and SINDy-PI

Despite its utility, conventional SINDy experiences challenges in biological situations. Many biological processes are regulated by rational functions (e.g., Michaelis-Menten or Hill kinetics), which are difficult to capture using only polynomial libraries.

To address this limitation, Mangan *et al.* [8] introduced implicit-SINDy. Rather than formulating an explicit equation,

$$\dot{x}_k = f(\mathbf{x}),$$

the implicit formulation aims to identify an implicit relationship in the derivative

$$F\big(\mathbf{x}, \dot{x}_k\big) = 0,$$

where F is represented as a sparse linear combination of candidate functions that rely on both the state vector $\mathbf{x}$ and derivative $\dot{x}_k$ for each k. After formulating the problem as an implicit equation, the implicit-SINDy requires an implicit feature library $\Theta(\mathbf{x}, \dot{x}_k)$ which solves the regression problem:

$$\Theta(\mathbf{x}, \dot{x}_k)\,\boldsymbol{\xi}_k = 0$$

Although implicit-SINDy successfully recovers the rational dynamics, the implicit formulation has significant drawbacks. It often results in a large and ill-conditioned null space, particularly when the data is noisy. This makes the sparse solutions unreliable or dense and prevents the model from generalizing.

To address the limitations of the implicit formulation, Kaheman *et al.* [43] developed SINDy-PI (Parallel Implicit), which reformulates the implicit-SINDy problem into a series of

explicit sparse regression problems that may be solved in parallel. By selecting one candidate term in $\Theta(\mathbf{x}, \dot{x}_k)$ as a fixed target and expressing it as a linear combination of all other terms, the method transforms the problem into:

$$\theta_j(\mathbf{x}, \dot{x}_k) = \sum_{i \neq j} \theta_i(\mathbf{x}, \dot{x}_k)\, \xi_i^{(j)},$$

which is then solved using sparse regression. This strategy not only avoids null-space computations but also improves numerical stability, ensures interpretability, and facilitates recovery of biologically meaningful models.

Massonis *et al.* extended this procedure by introducing further improvements which integrate symbolic simplification and identifiability analysis into the SINDy-PI framework. Their methodology transforms identified models into interpretable formats such as Hill or Michaelis–Menten kinetics, and confirms structural identifiability by eliminating non-unique representations. This guarantees that the retrieved models are both sparse and predictive, as well as biologically significant [44].

2.5 Summary

The literature shows a distinct transition from traditional, handcrafted models to data-driven, sparse model discovery methods that are better aligned with the structure and constraints of modern biological data. Through the integration of phase space reconstruction methods, dimension identification, and both explicit and implicit sparse regression, researchers may now derive governing equations from high-dimensional, noisy datasets with limited observability.

An excellent example of this methodology can be seen in the study of Brummer *et al.* [45], who examined CAR T-cell treatment using time-lapse microscopy data of glioblastoma–immune interactions. Initially, they utilized delay embedding and singular value decomposition to reconstruct a clean, low-dimensional phase space from noisy tumor trajectories. After that, by applying SINDy to the latent variables, they discovered dynamics that captured functional responses, including single-versus double-CAR binding and Allee-like tumor proliferation. This example shows how delay coordinate embedding and sparse model discovery together can be used to discover interpretable biological mechanisms while also providing practical insights for treatment design.

This thesis builds upon these methods by applying SINDy and its implicit variants to canonical biological models, aiming to evaluate their robustness, accuracy, and ability to reveal underlying regulatory interactions.

Chapter 3

Standard Test Cases Taken for the Analysis

This chapter presents a collection of canonical models that form the basis for the subsequent study in this thesis. These examples have been taken from the study conducted by Prokop and Gelens [5]. These examples illustrate a variety of systems spanning physical, chemical, and biological domains and have been selected to assess the applicability and constraints of the Sparse Identification of Nonlinear Dynamics (SINDy) framework.

The main goal of this chapter is to show the governing equations and general behavior (such as time series, phase diagrams, and codimension-one bifurcation diagrams) of each model. These serve as a foundation for the more extensive studies that follow in the upcoming chapters, where we use time-delay embedding techniques, construct SINDy for model discovery, and draw the interaction networks for further investigations. Each example is presented below along with its brief introduction, governing equations, and qualitative dynamics. All code, simulation files, and visualization scripts used for this chapter are available at https://github.com/Alka-CBhub/Chapter-3.

3.1 Example-1: FitzHugh–Nagumo Model

The FitzHugh–Nagumo (FHN) model was first introduced in the 1960s to describe the oscillatory spiking behavior of neurons [27]. Since then, it has been applied to a variety of biological phenomena, including early embryonic cell cycle oscillations [46] and cardiac excitation dynamics [28]. This model captures oscillatory behavior in a **two-dimensional** framework and is moderately complex, featuring a **cubic nonlinearity** to model excitation and recovery dynamics. The following form of the FHN model is used, as taken from [5]:

$$\frac{dx}{dt} = -x^3 + \gamma x^2 + \delta x - y, \tag{3.1.1}$$

$$\frac{dy}{dt} = \varepsilon(x - \beta y + \alpha), \tag{3.1.2}$$

where the variables x and y are dimensionless state variables. The parameters $\alpha, \beta, \gamma, \delta, \varepsilon$ are the kinetic parameters that govern the system's dynamics (**Figure 3.1**).

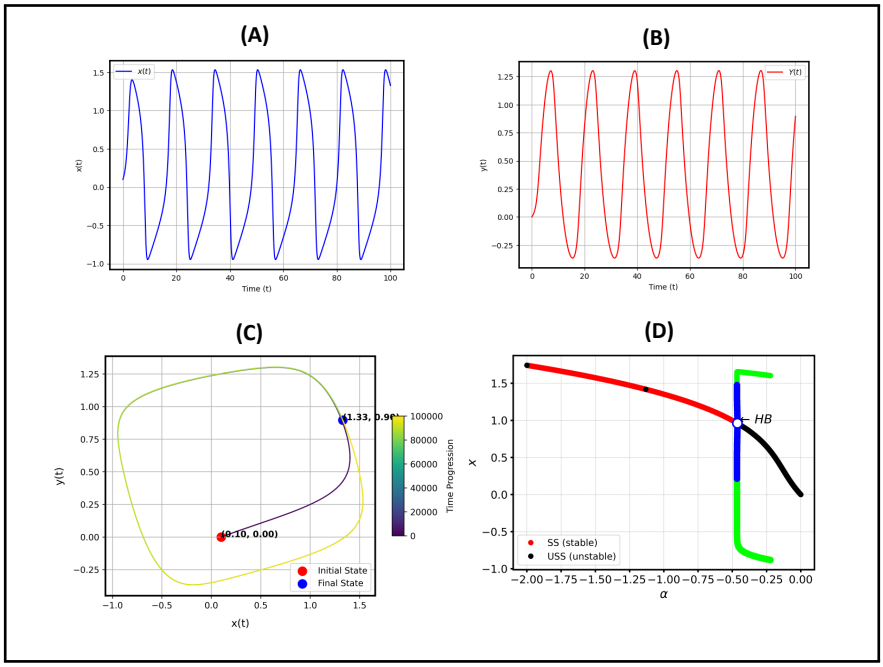

Figure 3.1. *Time series, phase diagram, and codimension-one bifurcation diagram of the FitzHugh-Nagumo model. (A), (B) Time series of the FitzHugh-Nagumo model for the variables x and y, each starting with x = 0.1 and y = 0.0, respectively. The values of the parameters to simulate the system are taken from the* ***Table*** (3.2)*. (C) Phase diagram of the FitzHugh–Nagumo model. The system transitions from the initial state (red circle) to a stable limit cycle, showing the sustained oscillations in x and y. (D) A plot of steady-state x as a function of bifurcation parameter* α*. HB is Hopf bifurcation, and SS and USS are stable and unstable steady states. A Hopf bifurcations occur at* $\alpha \approx -0.463$*. Near* -0.463*, the stable fixed point (red curve) becomes unstable (black curve), and a limit cycle emerges around it. This type of bifurcation is a subcritical Hopf, as the periodic orbit surrounds unstable steady states. The green and blue points correspond to maximum and minimum values attained by a stable and unstable limit cycle, respectively.*

3.1.1 Interaction Network using Jacobian

The model exhibits a unique steady state at $(0, 0)$. To visualize the regulatory structure near this steady state, we constructed the interaction network based on the sign structure of the Jacobian matrix, shown in **Figure 3.2**.

The model contains a positive feedback loop (via self-activation of x), a negative feedback loop (via self-inhibition of y) and a negative feedback loop (via $x \to y \dashv x$). Such negative feedback loops are a necessary condition for sustained *oscillations*, while positive feedback enables *excitability* [23]. In this system, the time-scale separation between x (fast) and y (slow) introduces the delay required to destabilize the fixed point, making oscillatory behavior possible under current parameter regimes.

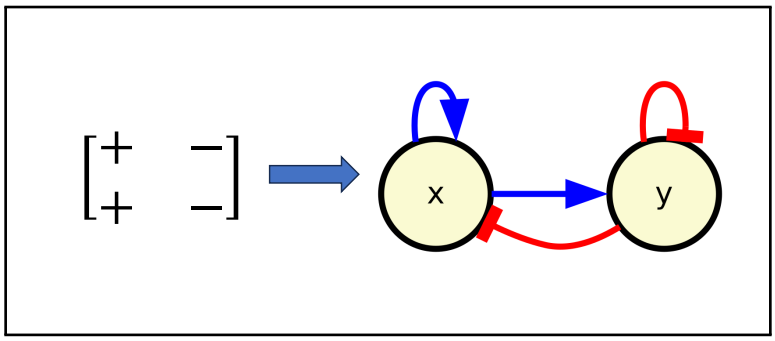

Figure 3.2. *Interaction network derived from the Jacobian near the steady state of the FitzHugh–Nagumo model. The interaction network is constructed based on the sign structure of the Jacobian matrix evaluated at the steady state* $(x^*, y^*) = (0, 0)$*. The sign pattern indicates the following regulatory influences:* x *exhibits positive auto-regulation,* y *negatively regulates* x*,* x *activates* y*, and* y *exhibits negative auto-regulation. Blue arrows represent activation and red arrows represent inhibition. This feedback architecture includes both one-node loops (a positive self-loop on* x *and a negative self-loop on* y*) and a two-node negative feedback loop (*$x \to y \dashv x$*).*

3.2 Example-2: The Goodwin Model

The Goodwin model, proposed by B. Goodwin in the 1960s, describes biological processes including circadian rhythms and the phosphorylation/dephosphorylation of transcription factors [29, 30]. Unlike the FitzHugh–Nagumo model, the Goodwin model exhibits greater structural complexity. It captures **nonlinear interactions via the Hill function** $H(w)$. We consider the following variant of the Goodwin model, as used in [5]:

$$\frac{dx}{dt} = aH(z) - dx, \tag{3.2.1}$$

$$\frac{dy}{dt} = bx - ey, \tag{3.2.2}$$

$$\frac{dz}{dt} = cy - fz, \tag{3.2.3}$$

where the Hill function is defined as

$$H(w) = \frac{K^n}{K^n + w^n}, \quad \text{for any arbitrary variable } w. \tag{3.2.4}$$

The variables x, y, and z represent dynamical variables specific to the system. For example, in a circadian rhythm context, they may correspond to clock mRNA, clock protein, and a transcriptional repressor, respectively. The parameters a, b, c, d, e, f are rate constants governing production and degradation processes (**Figure 3.3**).

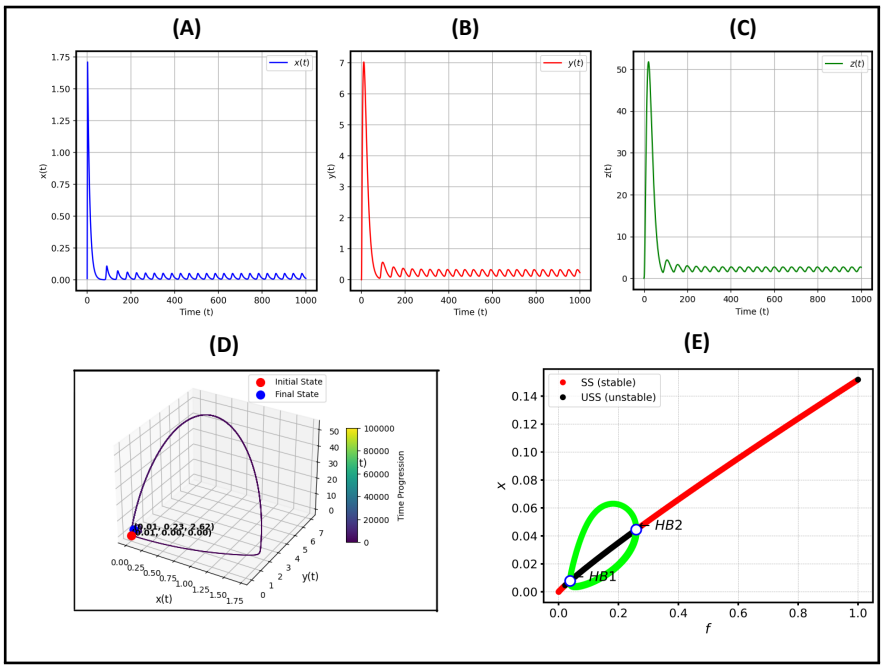

Figure 3.3. *Time series, phase diagram, and codimension-one bifurcation diagram of the Goodwin model. (A), (B), and (C) Time series of the Goodwin model for the variables x, y, and z, each starting with x = 0.01, y = 0.0, and z = 0.0, respectively. The values of the parameters to simulate the system are taken from the* ***Table*** (3.2)*. (C) Phase diagram of the Goodwin model. The system transitions from the initial state (red circle) to the final state through a spiral path, illustrating the damped oscillations in x and y. (D) A plot of steady-state x as a function of the bifurcation parameter f. HB1 and HB2 are Hopf bifurcations, and SS, USS are stable and unstable steady states. Two Hopf bifurcations occur near $f = 0.04$ and $f = 0.26$. Near $f = 0.04$, a stable steady state (red curve) loses its stability (black curve) and gives rise to a stable limit cycle (green curve), while near $f = 0.26$ the stable limit cycle disappears, and the steady state becomes stable again (red curve). This is the case of supercritical Hopf, as the periodic orbit surrounds unstable steady states. The green points correspond to maximum and minimum values attained by a stable limit cycle oscillation.*

3.2.1 Interaction Network using Jacobian

This model also exhibits a unique steady state at $(0.0187,\ 0.187,\ 1.87)$. We constructed the interaction network near this state based on the sign pattern of the Jacobian, as shown in **Figure 3.4**.

The Jacobian reveals three negative self-regulatory loops (on x, y, and z) and a three-node closed negative feedback loop: $x \to y \to z \dashv x$. Thomas and Thieffry's framework suggests that this configuration is a necessary condition for the emergence of sustained oscillations [23]. However, in the current parameter regime, the system exhibits damped oscillations and eventually converges to a unique stable state. This suggests that, while the feedback structure allows for oscillations, additional factors, such as time-scale separation and parameter values, are crucial for realizing them in practice.

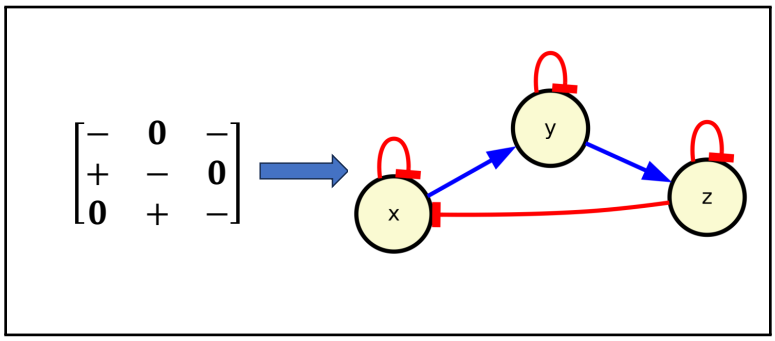

Figure 3.4. *Interaction network derived from the Jacobian near the steady state of the Goodwin model. The interaction network is based on the sign structure of the Jacobian matrix evaluated at the steady state* $(x^*, y^*, z^*) = (0.0187,\ 0.187,\ 1.87)$. *All three components* x, y, z *exhibit negative auto-regulation. Additionally,* z *inhibits* x, *while* x *activates* y, *and* y *activates* z. *Blue arrows represent activation and red arrows represent inhibition. This feedback architecture forms one-node negative feedback loops and a three-node closed negative feedback loop* ($x \to y \to z \dashv x$).

3.3 Example-3: Mass-Action Model

Mass-action models [32, 33] are among the most widely used frameworks for describing the dynamics of biochemical systems. They offer a relatively simple yet **high-dimensional** representation of molecular interactions, based on the law of mass action, which states that reaction rates are proportional to the concentrations of the interacting species. As a result, these models **typically involve linear or quadratic terms**.

The model considered here, developed by Hopkins et al. [31], describes the oscillatory behavior of the early embryonic cell cycle in *Xenopus laevis* using mass-action kinetics. The governing equations for this model are [5]:

$$\frac{du}{dt} = k_{p,a}(A_T - u)v - k_{d,a}u(P_T - y), \tag{3.3.1}$$

$$\frac{dv}{dt} = k_s - b_{\text{deg}}uv, \tag{3.3.2}$$

$$\frac{dw}{dt} = k_{p,g}(G_T - w)v - k_{d,g}(P_T - y)w, \tag{3.3.3}$$

$$\frac{dx}{dt} = -k_{p,e}xw + k_{\text{cat}}y, \tag{3.3.4}$$

$$\frac{dy}{dt} = k_{\text{ass}}(E_T - x - y)(P_T - y) - k_{\text{diss}}y - k_{\text{cat}}y. \tag{3.3.5}$$

Here, u, v, w, x, and y, represent molecular concentrations or active/inactive forms of regulatory components. The system exhibits oscillatory behavior for some specific choices of kinetic parameters (**Figure 3.5**).

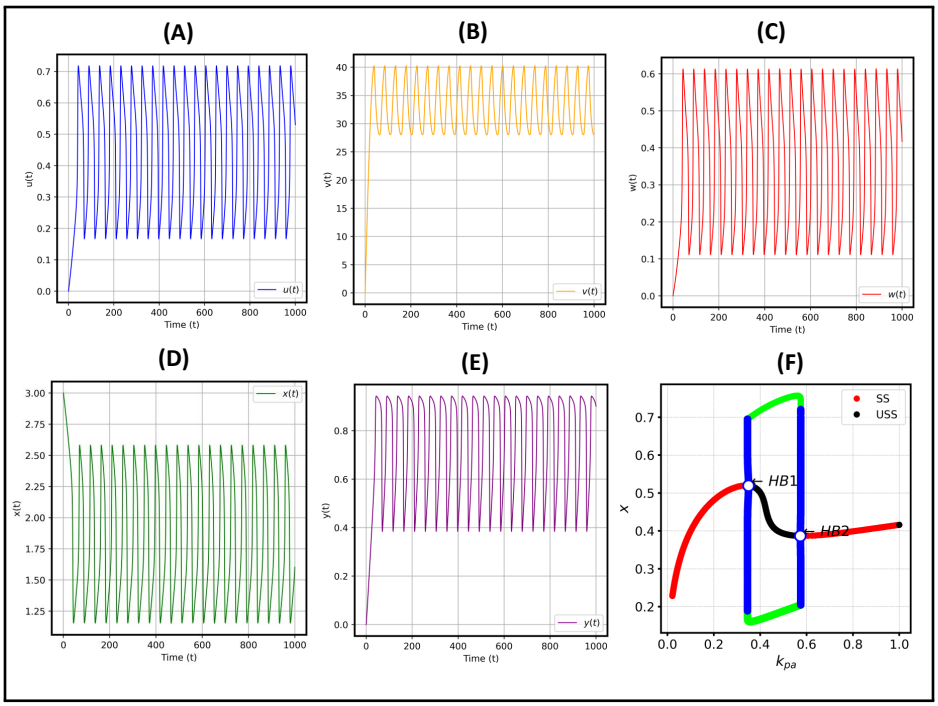

Figure 3.5. *Time series and codimension-one bifurcation diagram of the mass action model. (A)-(E) Time series of the mass action model for the variables u, v w, y, and z, each starting with u = 0, v = 0, w = 0, y = 3, z = 0. The values of the parameters to simulate the system are taken form the* ***Table*** (3.2)*. (F) A plot of steady state x as a function of bifurcation parameter* k_{pa}*. HB1 and HB2 are Hopf bifurcations, and SS and USS are stable and unstable steady states. Two Hopf bifurcations occur near* $k_{pa} = 0.35$ *and* $k_{pa} = 0.57$*. Near* $k_{pa} = 0.35$*, a stable steady state (red curve) loses its stability (black curve) and gives rise to limit cycles, while near* $f = 0.57$ *the limit cycle disappears, and the steady state becomes stable again (red curve). This is the case of subcritical Hopf, as the periodic orbit surrounds unstable steady states. The green and blue points correspond to maximum and minimum values attained by stable and unstable limit cycles respectively.*

3.3.1 Interaction Network using Jacobian

The steady state for this model occurs at $(0.501,\ 29.9,\ 0.386,\ 1.71,\ 0.881)$. The interaction network near this steady state is shown in **Figure 3.6**. From the structure of the Jacobian, we can see that all variables, u, v, w, x, and y exhibit negative self-regulation. Multiple cross-regulatory interactions are also present. The variable u is activated by both v and y, and v is inhibited by u. Further, w is activated by both v and y, while x is inhibited by w but activated by y. At last y is inhibited by x.

This network reveals multiple nested regulatory motifs. All variables u, v, w, x, y exhibiting negative self-regulation near the steady state are represented by red self-loops. They might contribute to local stability. Two-node delayed negative feedback loops are observed in the interactions $v \to u \dashv v$ and $x \to y \dashv x$, each forming a minimal inhibitory circuit. A three-element positive feedback circuit is present via the interaction $w \dashv x \dashv y \to w$. Additionally, a five-element closed loop spanning the full system is also present, $u \dashv v \to w \dashv x \dashv y \to u$.

This may cause a larger delay in the network. Overall, these motifs satisfy the necessary structural conditions for the emergence of sustained oscillations [23].

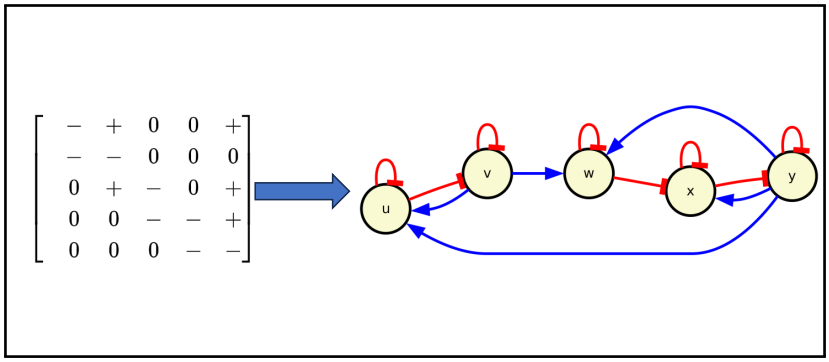

Figure 3.6. *Interaction network derived from the Jacobian near the steady state for the mass action model. The network is based on the sign structure of the Jacobian matrix evaluated at the steady state* $(u^*, v^*, w^*, x^*, y^*) = (0.501,\ 29.9,\ 0.386,\ 1.71,\ 0.881)$. *All components* u, v, w, x, *and* y *exhibit self-inhibition. Cross-regulatory interactions include activation of* u *by* v *and* y*; inhibition of* v *by* u*; activation of* w *by* v *and* y*; inhibition of* x *by* w*; activation of* x *by* y*; and inhibition of* y *by* x. *Blue arrows represent activation and red arrows represent inhibition. This feedback architecture forms multiple nested regulatory motifs. Negative self-loops on all variables, two-node delayed negative feedback loops,* $v \to u \dashv v$ *and* $x \to y \dashv x$, *a three-node positive feedback loop arises through the interaction* $w \dashv x \dashv y \to w$ *and a five-node negative loop* $u \dashv v \to w \dashv x \dashv y \to u$.

3.4 Example-4: The Oregonator Model

The Belousov–Zhabotinsky (BZ) reaction is a classical chemical oscillator. Even though it's not biological, its oscillating behavior is similar to rhythms seen in systems like cardiac tissue [47]. The system of differential equations describing this reaction is known as the Oregonator model, proposed by Field, Körös, and Noyes [48], describes its dynamics using a simplified set of reactions. The model involves three key components: bromous acid $[HBrO_2]$, bromide ions $[Br^-]$, and the cerium ion ratio $[Ce^{4+}]/[Ce^{3+}]$. The governing equations are:

$$\epsilon \frac{dx}{dt} = q\,a\,y - x\,y + a\,x - x^2, \tag{3.4.1}$$

$$\gamma \frac{dy}{dt} = -q\,a\,y - x\,y + f\,b\,z, \tag{3.4.2}$$

$$\frac{dz}{d\tau} = a\,x - b\,z, \tag{3.4.3}$$

Here, x, y, and z denote dimensionless concentrations, and parameters ϵ, γ, q, f control the system's time scales and stability (**Figure 3.7**).

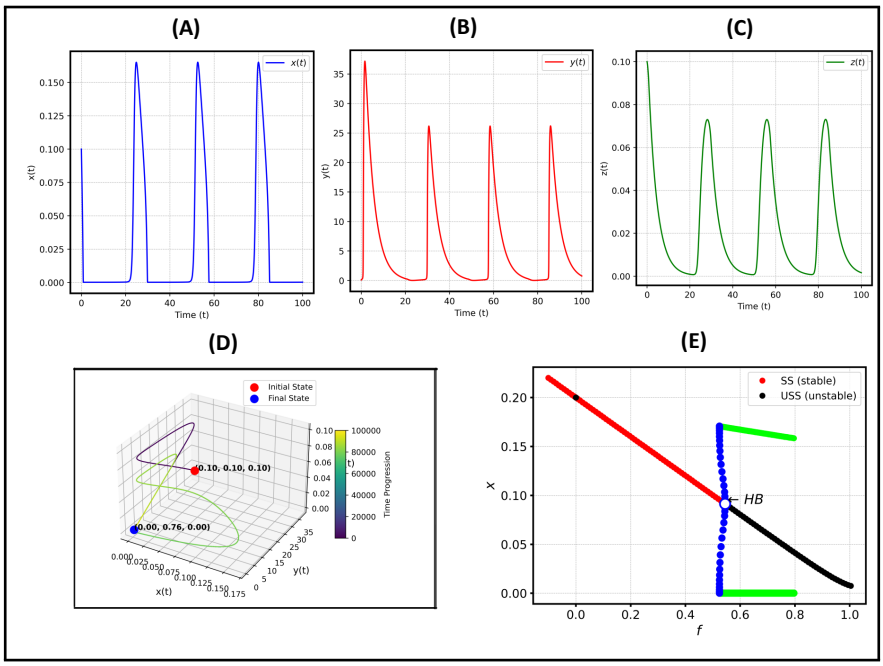

Figure 3.7. *Time series, phase diagram, and codimension-one bifurcation diagram of the Oregonator model. (A)-(C) Time series of the Oregonator model for the variables x, y, and z, starting with x = 0.1, y = 0.1, z = 0.1. The values of the parameters to simulate the system are taken form the* ***Table*** (3.2)*. (D) Phase diagram of the Oregonator model. The system transitions from the initial state (red circle) and converges to a limit cycle. (E) A plot of steady-state x as a function of bifurcation parameter f. HB is Hopf bifurcations, SS and USS are stable and unstable steady states. A Hopf bifurcations occur near $f = 0.54$. Near $f = 0.54$, a stable steady state (red curve) loses its stability (black curve) and gives rise to a limit cycle. This is the case of subcritical Hopf, as the periodic orbit surrounds unstable steady states. The green and blue points correspond to maximum and minimum values attained by a stable and unstable limit cycles, respectively.*

3.4.1 Interaction Network using Jacobian

This model exhibits two distinct steady states, $(\,0,\ 0,\ 0\,)$ and $(\,0.0706,\ 0.130,\ 0.0565\,)$ respectively. The regulatory structure corresponding to each equilibrium is shown in **Figure 3.8**.

Near $(0, 0, 0)$, the Jacobian reveals three self-regulatory loops on (x, y, and z and a three-element positive feedback loop formed entirely from positive interactions: $x \to z \to y \to x$. This architecture supports *multistability* [23].

In contrast, near the second steady state, $(0.0706,\ 0.130,\ 0.0565)$, all variables exhibit negative self-regulation, indicating local damping. The cross-regulatory structure includes mutual inhibition between x and y, as well as activation from x to z and from z to y. These interactions give rise to a three-element loop $x \to z \to y \dashv x$, which constitutes a *negative feedback loop*. This architecture aligns with the possibility of oscillations, particularly when delays or suitable time-scale separation are present.

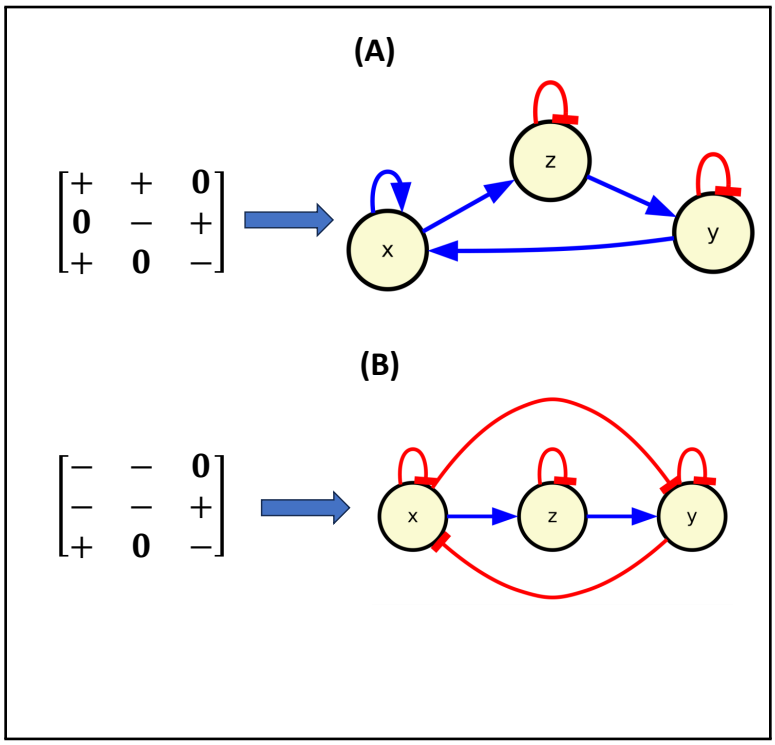

Figure 3.8. *Interaction networks derived from the Jacobian near steady states for the Oregonator model. (A), (B) represent the networks based on the sign structure of the Jacobian evaluated at steady states* $(0, 0, 0)$ *and* $(0.0706,\ 0.130,\ 0.0565)$ *respectively. (A) Near the steady state* $(x^*, y^*, z^*) = (0, 0, 0)$*, the system exhibits a positive self-loop on* x*, negative self-loops on* y *and* z*, and a fully positive three-node feedback loop* $x \to z \to y \to x$*. (B) Near steady state* $(0.0706,\ 0.130,\ 0.0565)$*, all variables show self-inhibition. The cross-regulation involving* $x \to z$*,* $z \to y$*, and* $y \dashv x$*, forming a delayed negative feedback loop* $x \to z \to y \dashv x$*. Blue arrows denote activation and red bars denote inhibition.*

3.5 Example-5: Glycolytic Oscillation in Yeast

Glycolysis is a metabolic process that breaks down glucose into smaller molecules to produce energy. One of its primary roles is to make adenosine triphosphate (ATP), especially when there isn't enough oxygen. Glycolytic oscillations have been seen in cell-free extracts or concentrated suspensions of non-proliferating *Saccharomyces cerevisiae* (yeast) under glucose starvation.

The interaction between adenosine diphosphate (ADP) and ATP is the main force behind these oscillations, and they constitute the foundation of the famous Selkov model [49]. The equations governing the dynamics are [50]:

$$\frac{dx}{dt} = a + x^2 y - x, \tag{3.5.1}$$

$$\frac{dy}{dt} = b - x^2 y. \tag{3.5.2}$$

where x and y are dimensionless variables that may represent ADP and ATP concentration, respectively (**Figure 3.9**).

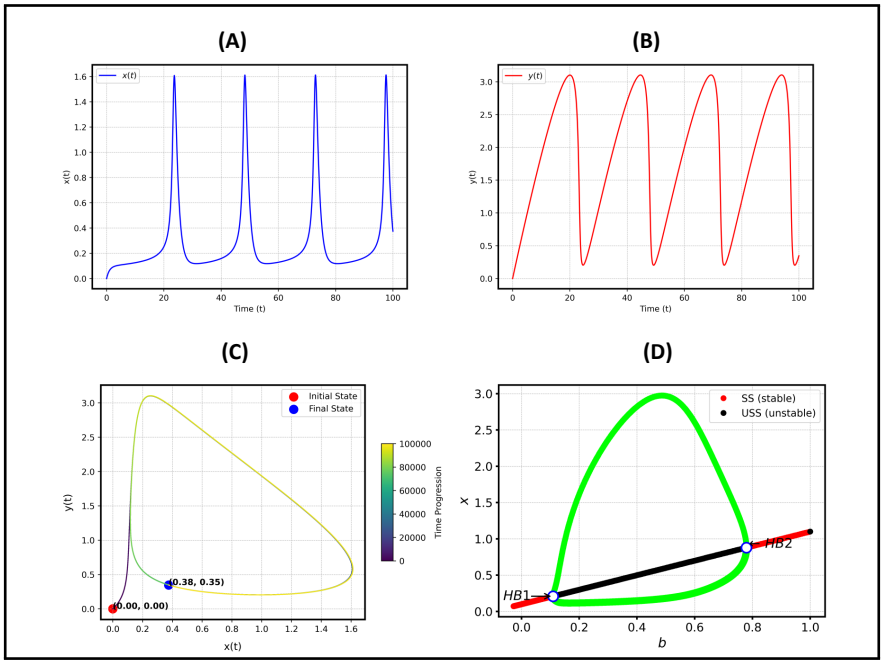

Figure 3.9. *Time series, phase diagram, and codimension-one bifurcation diagram of the Goodwin model. (A), (B) Time series of the glycolytic oscillation for the variables x, and y, starting with x = 0.0, y = 0.0. The values of the parameters to simulate the system are taken from* ***Table*** (3.2)*. (C) Phase diagram of glycolytic oscillation. The system transitions from the initial state (red circle) to a stable limit cycle. (D) A plot of steady state x as a function of bifurcation parameter b. HB1 and HB2 are Hopf bifurcations, and SS and USS are stable and unstable steady states. Two Hopf bifurcations occur near* $b = 0.11$ *and* $b = 0.78$*. Near* $f = 0.11$*, a stable steady state (red curve) loses its stability (black curve) and gives rise to a stable limit cycle (green curve) while near* $f = 0.78$ *the stable limit cycle disappears, and the steady state becomes stable again (red curve). This is the case of supercritical Hopf, as the periodic orbit surrounds unstable steady states. The green points correspond to maximum and minimum values attained by a stable limit cycles.*

3.5.1 Interaction Network using Jacobian

The system exhibits a single steady state at $(x^*, y^*) = (0.300,\ 2.22)$. The interaction network near this state is shown in **Figure 3.10**. The structure of the Jacobian shows the presence of a positive self-loop on x, and a negative self-loop on y. The cross-regulatory interactions involve the inhibition of y via x and activation of x via y. These interactions form a two-node feedback circuit $x \dashv y \rightarrow x$, which constitutes a *negative feedback loop*. The similar kind of architecture has been seen in the FitzHugh-Nagumo model.

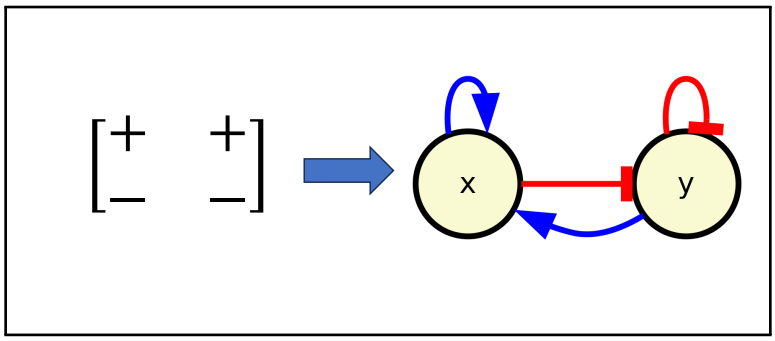

Figure 3.10. *Interaction network derived from the Jacobian near the steady state for the glycolytic oscillation model. The interaction network is based on the sign structure of the Jacobian matrix evaluated at the steady state* $(x^*, y^*) = (0.300,\ 2.22)$*. The sign pattern indicates the following regulatory influences:* x *exhibits positive regulation via* y *and itself.* y *exhibits negative auto-regulation as well inhibited by* x*. Blue arrows represent activation and red arrows represent inhibition. This feedback architecture includes both one-node loops (a positive self-loop on* x *and a negative self-loop on* y*) and a two-node negative feedback loop* $(x \to y \dashv x)$*.*

In this chapter, we focused on understanding the dynamics of a biological regulatory circuit using some standard examples of nonlinear ODE models. These models capture essential features such as activation, inhibition, and feedback, which are fundamental to many biological networks. Through numerical simulations and phase portrait visualization, we analyzed their dynamical behavior. Also, for each of the models, we identified steady states and explored the local interaction network using Jacobian. We also used XPPAUT [21] to perform bifurcation analysis and analyzed how a system's behavior changes with respect to certain key parameters. Overall, this chapter demonstrated how mathematical and computational tools can uncover rich dynamical behavior in biological networks.

Table 3.1. *Summary of the Models*

No.	Model	# Variables	# Parameters
1	FitzHugh–Nagumo Model	2	5
2	Goodwin Model	3	8
3	Mass Action Models	5	14
4	Oregonator Model for B–Z Reaction	3	6
5	Glycolytic Oscillation in Yeast	2	2

Table 3.2. *Values of parameters for each model*

No.	Model	Parameters
1	FitzHugh-Nagumo Model	$\alpha = 0$, $\beta = 0.5$, $\gamma = 1$, $\delta = 1$, $\varepsilon = 0.3$.
2	Goodwin Model	$a = 1$, $b = 1$, $c = 1$, $d = 0.1$, $e = 0.1$, $f = 0.1$, $n = 10$, $K = 1$.
3	Mass Action Models	$k_{p,a} = 0.4$, $k_{d,a} = 100$, $k_{p,g} = 0.05$, $k_{diss} = 1$, $k_{d,g} = 20$, $k_{p,e} = 6$, $k_{cat} = 4.5$, $k_{ass} = 100$, $k_s = 1.5$, $b_{deg} = 0.1$, $A_T = 1$, $P_T = 1$, $G_T = 1$, $E_T = 3$.
4	Oregonator Model for B-Z Reaction	$\epsilon = 0.067$, $\gamma = 8.89 \times 10^{-5}$, $q = 8 \times 10^{-4}$, $a = 0.2$, $b = 0.25$, $f = 0.65$.
5	Glycolytic Oscillation in Yeast	$a = 0.1$, $b = 0.2$

Chapter 4

Phase Space Reconstruction

Understanding and analyzing complicated dynamical systems presents a number of challenges. **Often, we do not know all of the important state variables that represent the system**. Even when these variables are known, measuring them can be challenging, and even if we can, the act of measuring itself can disturb the natural dynamics of the system under investigation.

Assume that $\mathbf{s}(t) \in \mathbb{R}^n$ denotes the whole set of variables that characterize a system's state at time t. Also let $h : \mathbb{R}^n \to \mathbb{R}^m$, where $m < n$, represent a measurement function, which is basically a mapping done by a device or sensor to acquire the observable data. During any trial of the measurement process, we always observe $\mathbf{x}(t) \in \mathbb{R}^m$, a reduced and simplified picture of the entire state $\mathbf{s}(t)$, instead of observing the full $\mathbf{s}(t)$ **because measuring all the state variables is usually infeasible**. For simplicity, we concentrate on the scenario when $m = 1$, indicating a single scalar time series [51].

The key challenge, therefore, is to ascertain all internal state variables of the system based solely on the available time series, $x(t)$. A proficient method for addressing this challenge is **reconstructing the state space using delay coordinate embedding**.

State space reconstruction is a fundamental method in the theory of nonlinear time-series analysis. This methodology, first introduced by Packard et al. [36] and subsequently formalized by Takens [34], enables the approximation of the whole dynamics of the system with just the observed time series $x(t)$. **Reconstructed dynamics** are not always the same as the system's internal dynamics, but they **are topologically equivalent to the real dynamics if the reconstruction is done properly**. This topological equivalence guarantees that the system's many essential features, which are invariant under smooth transformations, are preserved. As a result, insights derived from the reconstructed dynamics can often be applied directly to the original system.

4.1 Delay Coordinate Embedding

Delay coordinate embedding, often known as time delay embedding, is the conventional method for state space reconstruction, initially proposed by Packard et al. [36]. The key idea of delay

coordinate embedding is to graph the data against its delayed version. In other terms, it involves transforming a scalar time series $x(t)$ into a higher-dimensional space by generating the delay vectors, $\mathbf{r}(t)$ as:

$$\mathbf{r}(t) = [x(t), x(t-\tau), x(t-2\tau), \ldots, x(t-(m-1)\tau)]$$

where τ represents the delay lag and m denotes the embedding dimension. If N denotes the length of the time series $x(t)$, beginning at $t = 1$, then the total number of embedding vectors will be $N - (m-1)\tau$, and the dimensions of the embedded data matrix will be $[N - (m-1)\tau] \times m$.

For delay coordinate embedding to function properly, the observed time series data $x(t)$ must meet many criteria:

(1) The data points $x(t_i)$ must be evenly spaced. Real-world data acquisition systems rarely achieve such type of time series. Therefore, interpolation is frequently employed as a solution. However, interpolated data combines real and estimated dynamics, introducing potential inaccuracies [52].

(2) The measurement function h that generates the time series $x(t)$ must be a smooth and generic function of the state space of the system. This condition may fail under certain circumstances, such as when an event counter in the data acquisition system exceeds its limit. Testing whether the measurement function meets this theoretical criterion might be difficult. Practical solutions to this problem include changing the sample frequency or measuring a new quantity, followed by reanalyzing the data. Consistent findings across modifications indicate reliability of the measuring procedure [52].

A Comment on the Choice of the Embedding Parameters

The parameters τ and m are collectively referred to as embedding parameters. According to Takens' theorem [34], **embedding is theoretically valid for any $\tau > 0$, provided that m is sufficiently large and data are measured with infinite precision**. However, these ideal conditions are unattainable in practice due to noise, finite data lengths, and computational limitations. The selection of τ and m significantly impacts the embedding quality, as **different combinations unfold the dynamics to varying extents**. If τ is **too small**, the components of $\mathbf{r}(t)$ becomes **strongly correlated**. As a result, reconstructed trajectories align closely with the diagonal of the embedding space. As τ grows, the dynamics diverge more from this diagonal. Conversely, as τ increases, the dynamics progressively unfold away from this diagonal. In practical applications, the value of τ must be chosen carefully to guarantee that the embedding is sufficiently unfolded and prevents overlapping trajectories.

If d represents the actual dimension of the underlying system, original embedding theorems require that $m > 2d$ ensure topological equivalence between the reconstructed and true dynamics. Sauer et al. [53] presented a less stringent requirement, $m > 2d_{cap}$, where

d_{cap} denotes the capacity dimension of the attractor. Unfortunately, calculating d or d_{cap} is challenging without any prior knowledge about the system.

4.2 Methods for Estimating the Embedding Parameters

4.2.1 Estimating the Time Delay

State space reconstruction begins with the estimation of the time delay (τ), a critical parameter that significantly impacts the selection of the embedding dimension (m). Literature has several techniques for identifying an optimal τ. These strategies generally provide optimal results for specific systems and average results for others. The autocorrelation function and mutual information techniques stand out among these approaches. This thesis uses the autocorrelation function approach to determine the best time delay [37].

Autocorrelation Function Method

The autocorrelation function technique is based on the idea that **each delayed component in the reconstructed state space must yield new, independent information regarding the system's underlying dynamics**. As a result, the reconstructed space accurately preserves the original system's key dynamics. Mathematically, the autocorrelation function $(ACF(\tau))$ at lag τ is defined as

$$ACF(\tau) = \frac{\sum_{t=1}^{N-\tau} (x_t - \bar{x})(x_{t+\tau} - \bar{x})}{\sum_{t=1}^{N} (x_t - \bar{x})^2}, \tag{4.2.1}$$

To estimate the time delay τ using this method, the $ACF(\tau)$ is plotted as a function of τ. The smallest τ for which $ACF(\tau)$ decays to $1/e$ is chosen as a time delay.

4.2.2 Estimating the Embedding Dimension

After determining the time delay τ, the next step is to estimate the embedding dimension m, which reflects the number of delayed copies of the time series needed to reconstruct the system's dynamics. This thesis utilizes the method of false nearest neighbors (FNN) to estimate m [38].

The Method of False Neighbors

This method is based on the idea that when a deterministic system is embedded into a higher-dimensional space, it should unfold the attractor smoothly. This means **that points close together in a lower dimension** $(m = k)$ **should remain close when embedded into a higher dimension** $(m = k + 1)$**. If they become significantly separated, it indicates the dynamics were not sufficiently unfolded with dimension** $(m = k)$**. These points are considered as false nearest neighbors.** [37]. The procedure involves the following steps:

1. Given a time series $\{x_1, x_2, \ldots, x_N\}$; $x_i = x(i)$, and a chosen time delay τ, form vectors in m dimensional embedded space:

$$p_i = [x_i, x_{i+\tau}, x_{i+2\tau}, \ldots, x_{i+(m-1)\tau}].$$

   where p_i represents the state of the system at time i.

2. For each vector p_i, identify its **nearest neighbor** p_j in the current m- dimesnional space, such that for a given small positive threshold ϵ

$$||p_i - p_j|| < \epsilon,$$

3. Now, increase the embedding dimension to $m + 1$, by appending next delayed coordinate to each vector as

$$p_i' = [x_i, x_{i+\tau}, x_{i+2\tau}, \ldots, x_{i+m\tau}].$$

   and

$$p_j' = [x_j, x_{j+\tau}, x_{j+2\tau}, \ldots, x_{j+m\tau}].$$

4. Compute the ratio of the distance between the added coordinates to the original distance between p_i and p_j:

$$R_i = \frac{|x_{i+m\tau} - x_{j+m\tau}|}{||p_i - p_j||}.$$

   If R_i exceeds a predefined threshold R_{tr}, then p_j is marked as a false nearest neighbor ofp_i.

5. Repeat the above process for all points in the dataset and calculate the proportion of false nearest neighbors.

6. Increase m gradually and observe the proportion of false nearest neighbors. The appropriate embedding dimension m is determined as the smallest m at which the proportion approaches zero (indicating that most points are correctly close in higher dimensions and the dynamics are accurately unfolded).

4.3 Latent Variable Analysis

Latent variables are hidden structures or processes within a dataset that are not directly visible yet significantly influence the observed measurements. Many factors in high-dimensional biological data, such as gene expression profiles or time series, are interrelated and contained in a manifold of lower dimensions. While Takens' embedding allows us to reconstruct dynamics from a single scalar measurement, it doesn't automatically rank or prioritize the most significant directions of variability.

Techniques such as *Singular Value Decomposition (SVD)* and *Principal Component Analysis (PCA)* are employed to detect dominant patterns and minimize redundancy. These approaches identify orthogonal axes that identify the main directions capturing the variation in the data, hence allowing dimensionality reduction, noise filtration, and enhanced interpretability [35].

4.3.1 Singular Value Decomposition (SVD)

Singular Value Decomposition is a matrix factorization method that represents a real matrix $X \in \mathbb{R}^{m \times n}$ as the product of three "simple" matrices:

$$X = USV^\top$$

where:

- U is an $m \times m$ orthogonal matrix.
- V is an $n \times n$ orthogonal matrix.
- S is an $m \times n$ diagonal matrix with nonnegative entries s_i, arranged in descending order along the diagonal.

The columns of U are referred to as the **left singular vectors** of X and constitute an orthonormal basis for the column space of X with $SV^\top$ as coefficients. The columns of V are referred to as the **right singular vectors** and constitute an orthonormal basis for the row space of X with US as coefficients. The elements of S are the **singular values** of X.

The matrix X may be expressed as

$$X = \sum_{k=1}^{r} s_k u_k v_k^\top$$

where $r = \text{rank}(X)$. Each term captures a latent structure, whereas the singular values denote their relative significance.

4.3.2 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) aims to identify a collection of orthogonal directions (principal components) that maximize the variance within the data. It is predominantly utilized for feature extraction and dimensionality reduction.

For a data matrix $X \in \mathbb{R}^{m \times n}$, with each column centered at its mean, PCA requires the calculation of the covariance matrix:

$$\text{Cov}(X) = \frac{1}{m-1} X^\top X$$

and calculating its eigenvalues and eigenvectors. The eigenvectors define the principle directions, whereas the associated eigenvalues reflect the explained variance.

- The first principal component captures the direction of maximum variance.
- Subsequent components are orthogonal and capture decreasing amounts of variance.

Hence, PCA transforms the data into a new basis where each axis corresponds to a principal component.

4.3.3 Relationship Between SVD and PCA

A clear mathematical link exists between SVD and PCA, particularly when PCA is derived from the covariance matrix. If the data matrix X is column-centered, the covariance matrix is proportional to $X^\top X$. Performing Singular Value Decomposition (SVD) on X, gives

$$X = USV^\top$$

This indicates that,

$$X^\top X = VS^2V^\top$$

Therefore,

- The columns of V (right singular vectors) constitute the **principal components**.
- The squared singular values s_k^2 are the **eigenvalues** of the covariance matrix and relate to the variances along the main components.
- The product $XV = US$ gives the **principal component scores**—the coordinates of the data in the new principal component basis.

Alternatively, Conversely, if the rows of X are centered, then $XX^\top$ represents the covariance matrix, and the left singular vectors U correspond to the principal components of the variables (e.g., gene profiles).
This relationship indicates that PCA is essentially a specific instance of SVD applied to a centered matrix, where

$$\text{PCA} = \text{SVD of centered } X$$

Utilizing SVD or PCA enables the identification of significant latent dimensions in complex biological datasets, the denoising of observations, and the extraction of interpretable low-dimensional representations [35, 37].

4.4 Tools for Implementation

There are a number of techniques that may be used for nonlinear time series analysis and to estimate embedding parameters. Some well-known software packages are Teaspoon [54], Giotto-tda [55], and TISEAN [56].

In this thesis, we use TISEAN [56], a well-established software package **specifically designed for the analysis of time series data of nonlinear deterministic dynamical systems**. TISEAN offers a wide range of tools for phase space reconstruction, embedding parameter estimation, and various time series applications [37].

To enable integration with a Python-based process, we utilized its Python wrapper, known as PyTISEAN. **PyTISEAN provides** a sophisticated interface to TISEAN's C-based algorithms, facilitating **the implementation of TISEAN in the Python environment**. The tools used to find the embedding parameters and reconstruct the time series are

- `mutual` (mutual information) and `autocorr` (autocorrelation) for estimating the **optimal time delay** τ,
- `false_nearest` for identifying the **minimum embedding dimension** d by minimizing false neighbors,
- `delay` for reconstructing the phase space after finding the embedding parameters.

4.5 Application

The techniques discussed in this chapter can be used to determine the minimum number of dimensions (state variables) required before proceeding with any model identification method. To demonstrate the practical application of these techniques, we apply the methods described in this chapter to the examples introduced in **Chapter 3**. All codes and Jupyter notebooks corresponding to this chapter are available at `https://github.com/Alka-CBhub/Chapter-4`.

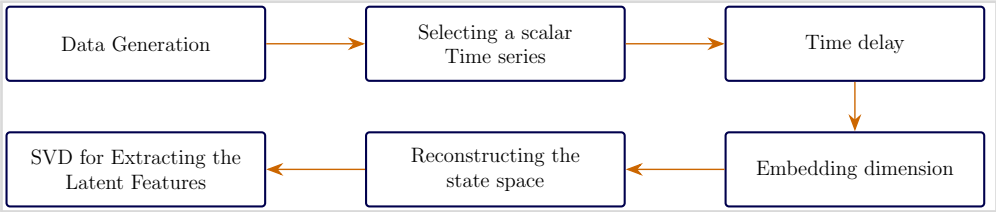

Figure 4.1. *Schematic overview of the State space reconstruction*

4.5.1 Example-1: FitzHugh–Nagumo Model

The FitzHugh–Nagumo (FHN) model, introduced in **Section 3.1**, is a **two-dimensional dynamical system** having cubic nonlinearity. In this section, we reconstruct its phase space using delay embedding.

We begin by generating the time series data by integrating the ODEs, using the parameter values listed in **Table 3.2** and simulation settings from **Table 4.1**. From the resulting trajectory, we extract the $x(t)$ variable and use it as the **observable scalar time series**.

Table 4.1. *Simulation settings for the FitzHugh-Nagumo model.*

Setting	IC	t_0	t_f	Δt
Value	[0.1, 0.0]	0	100	0.001

To estimate the embedding parameters, we employ the `autocor` and `false_nearest` functions from the PyTISEAN package. **Figure 4.2** displays the autocorrelation function (ACF) and the false nearest neighbors (FNN) outputs. The optimal time delay τ is selected as the first lag where the ACF drops below the threshold $1/e$. In this case, the selected delay is $\tau = \mathbf{2762}$ **steps**, which corresponds to $\mathbf{2.762}$ **seconds** given the sampling interval of 0.001.

Using this delay, the embedding dimension d is determined from the FNN method as the smallest dimension for which the percentage of false neighbors effectively drops to zero. This gives $\mathbf{d = 2}$.

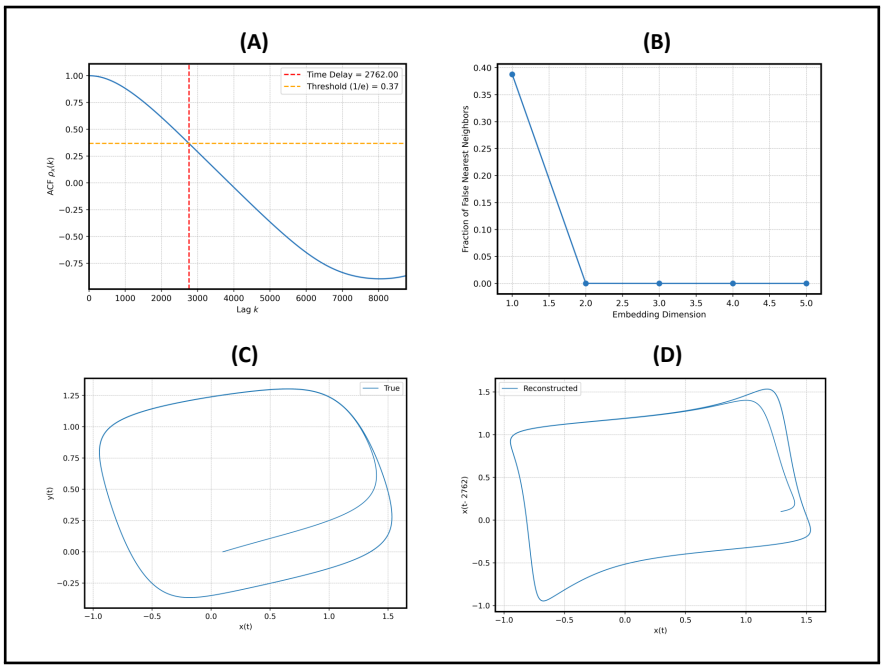

Figure 4.2. *Reconstruction parameters for the FitzHugh-Nagumo model using PyTISEAN. (A) Graphical output of the* `autcor` *function for the one dimensional time series* $x(t)$ *from the FitzHugh-Nagumo Model. The default threshold value* $(1/e)$ *is shown as the horizontal line in the plot. The optimal time delay* τ *is selected as the lag* $\tau = 2762$ *(time steps) where the ACF drops below the threshold* $1/e$ *for the first time. (B) Graphical output of the* `false_nearest` *function for the one dimensional time series from the FitzHugh-Nagumo Model. The function is called with the parameters* `M = 1,5`*, where 1 and 5 are the minimum and maximum allowed dimensions to test,* `d = 2762`*, and* `f = 3`*. The graph shows an immediate drop-off of the percentage of false-nearest neighbors to 0 near d=2, indicating that no additional embedding is necessary. (C) and (D) represent the original and reconstructed phase space for the FitzHugh-Nagumo Model. The time series for the reconstructed space is obtained by using* `delay`*. The phase diagram shows that they are topologically equivalent.*

With the estimated parameters $(\tau, d) = (2762, 2)$, we reconstruct the phase space using the `delay` function, which provides the embedded data whose columns are $[x(t), x(t-\tau)]$, starting from, $t_0 = 2.762$. A comparison of the original $[x(t), y(t)]$ and reconstructed $[x(t), x(t-\tau)]$ state spaces confirms that they are topologically equivalent, preserving the essential structure of the system dynamics.

To extract the latent feature from the reconstructed space, we perform SVD on the embedded data. The lower panel in **Figure 4.3** shows the first two latent components obtained after applying SVD to the reconstructed data matrix. Since SVD involves rotation and scaling operations, the latent features obtained from SVD require further processing like denoising and rescaling to get the biologically relevant features.

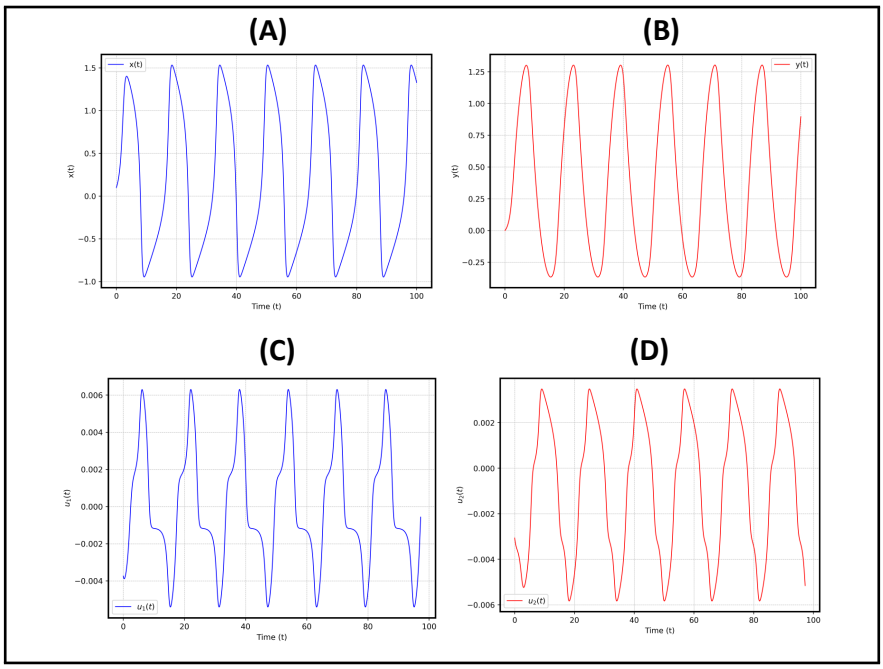

Figure 4.3. *Latent feature extraction using Singular Value Decomposition (SVD) on the delay-embedded time series from the FitzHugh–Nagumo model. In the upper panel, (A) and (B) show the original time series for the variables $x(t)$ and $y(t)$, respectively. In the lower panel, (C) and (D) display the first two latent components obtained after applying SVD to the data obtained from delay coordinate embedding. These components capture the dominant temporal patterns and preserve the core structure of the oscillations.*

4.5.2 Example-2: Goodwin Model

Next, we reconstruct the phase space of the Goodwin model, introduced in **Section 3.2**. It is a **three-dimensional nonlinear oscillator** used to describe biological rhythms such as circadian clocks and transcriptional regulation.

We begin by numerically integrating the system with the parameters listed in **Table 3.2** and simulation settings given in **Table 4.2**. Among the three resulting time series, we choose $x(t)$ as the observable input for delay coordinate embedding.

Table 4.2. *Simulation settings for the Goodwin model.*

Setting	IC	t_0	t_f	Δt
Value	[0.01, 0, 0]	0	1000	0.01

To estimate the delay, we apply the `autcor` function to the $x(t)$ series. The autocorrelation gives $\tau = \mathbf{1017}$ **steps**, which correspond to $\mathbf{10.17}$ **seconds** given the sampling interval of 0.01. Using this value of τ, we apply the `false_nearest` function to determine the minimum embedding dimension. The output reveals a rapid drop in the percentage of false neighbors, reaching zero at $\mathbf{d = 3}$, which matches the known dimension of the underlying

system (**Figure 4.4**).

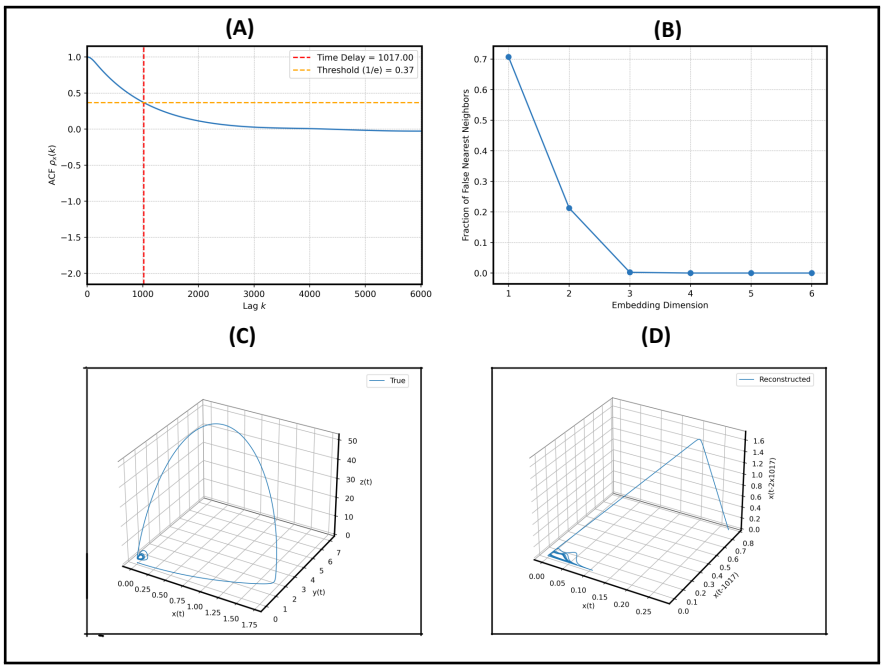

Figure 4.4. *Reconstruction parameters for the Goodwin model using PyTISEAN. (A) Graphical output of the* `autcor` *function for the one dimensional time series from the Goodwin Model. The default threshold value* $(1/e)$ *is shown as the horizontal line in the plot. The optimal delay is selected at lag* $\tau = 1017$ *time steps. (B) Graphical output of the* `false_nearest` *function for the one dimensional time series from the Goodwin model. The function is called with the parameters* `M = 1, 6`*, which allow to test a range of dimensions from 1 to 6,* `d = 1017`*, and* `f = 2`*. The graph shows an immediate drop-off of near* $d = 3$*, indicating that no additional embedding is necessary. (C) and (D) represent the original and reconstructed phase space for the Goodwin model. The time series for the reconstructed space is obtained by using* `delay`*. Both phase space are topologically identical.*

Using the estimated embedding parameters $(\tau, d) = (1017, 3)$, we perform delay coordinate embedding to reconstruct the full trajectories $[x(t), x(t-\tau), x(t-2\tau)]$, which start from time $t_0 = 2 \times 10.17$ seconds. Comparison of the original phase space $[x(t), y(t), z(t)]$ and the reconstructed space is shown in **Figure 4.4**.

We further extract the latent feature from the embedded data using SVD. The lower panel in **Figure 4.5** shows the first three latent components obtained after applying SVD to the reconstructed data matrix. The features capture the essential dynamics of the original data but need some preprocessing for further use.

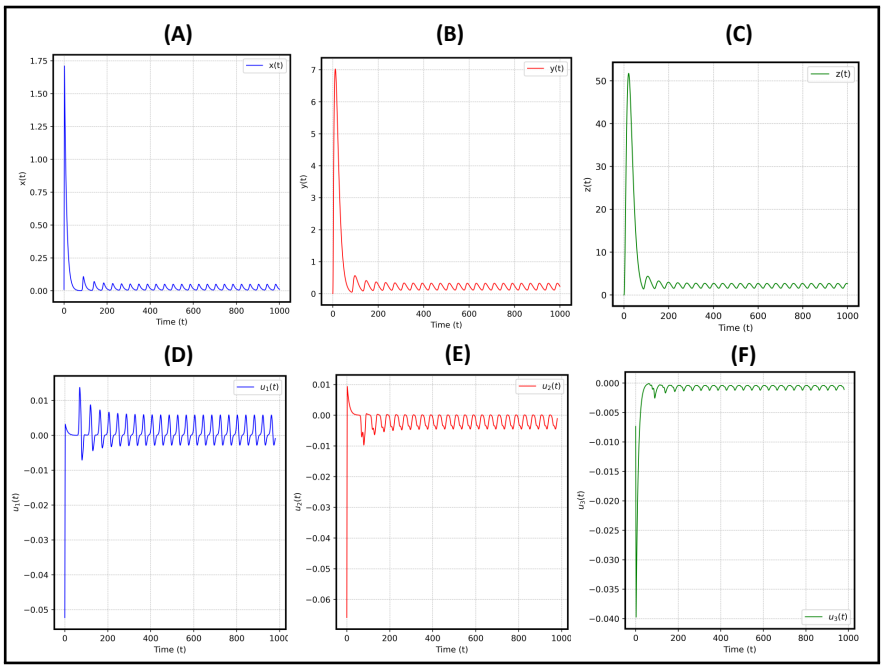

Figure 4.5. *Latent feature extraction using Singular Value Decomposition (SVD) on the delay-embedded time series from the Goodwin model. In the upper panel, (A)-(C) show the original time series for the variables x, y, and z of the Goodwin model. In the lower panel (D)-(F) display the first three latent components obtained after applying SVD to the data obtained from delay coordinate embedding. The components capture the temporal patterns observed in the original time series.*

4.5.3 Example-3: Mass-Action Model

The mass–action cell-cycle model introduced in **Section 3.3** is a **five-dimensional model** whose oscillatory dynamics emerge from linear and bilinear terms. We reconstruct this system using delay embedding based on the scalar time series $x(t)$.

The trajectories are obtained by integrating the model for the parameters listed in **Table 3.2** and simulation settings given in **Table 4.3**

Table 4.3. *Simulation settings for the Mass-Action model.*

Setting	IC	t_0	t_f	Δt
Value	[0, 0, 0, 3, 0]	0	1000	0.01

The optimal time delay in this case is obtained at $\tau = \mathbf{843}$ **steps** that correspond to $\mathbf{8.43}$ **seconds** for sampling interval of 0.01

Using this delay, we apply the false nearest neighbors (FNN) method to determine the minimal embedding dimension. The FNN output shows a sharp drop in the percentage of false neighbors from over 36% at $d = 1$ to nearly 0% at $d = 2$, after which the curve flattens. This suggests the embedding dimension $\mathbf{d = 2}$ (**Figure 4.6**.

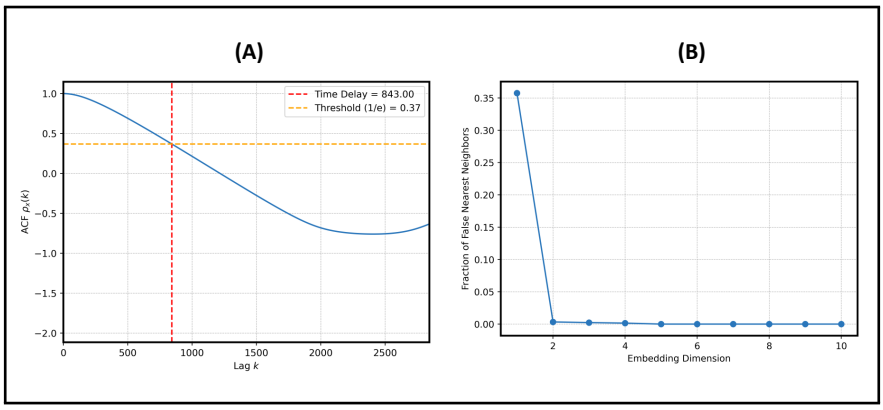

Figure 4.6. *Reconstruction parameters for the Mass-Action Model using PyTISEAN. (A) Graphical output of the* `autcor` *function for the one dimensional time series from the Mass-Action Model. The default threshold value* $(1/e)$ *is shown as the horizontal line in the plot. The optimal delay is selected at lag* $\tau = 843$ *time steps. (B) Graphical output of the* `false_nearest` *function for the one dimensional time series from the Mass-Action Model. The function is called with the parameters* `M = 1,10`*, testing the dimension range from 1-10,* `d = 843`*, and* `f = 1.12`*. The graph shows an immediate drop-off of the percentage of false-nearest neighbors at* $d = 2$*, after no significant false neighbors are found.*

Since the original system is 5-dimensional, we use singular value decomposition (SVD) on the delay-embedded matrix with dimension $d = 5$ to learn more about the inherent dimensionality of the reconstructed system. The singular values are $\{317.66,\ 95.90,\ 82.17,\ 27.80,\ 26.76\}$, which shows that the **first two or three components account for most of the variance**. This indicates that **the system lies on a two- to three-dimensional manifold, even if the entire model is five-dimensional**.

Overall, these findings show that the mass-action cell cycle model, while algebraically high-dimensional, has low-dimensional oscillatory behavior.

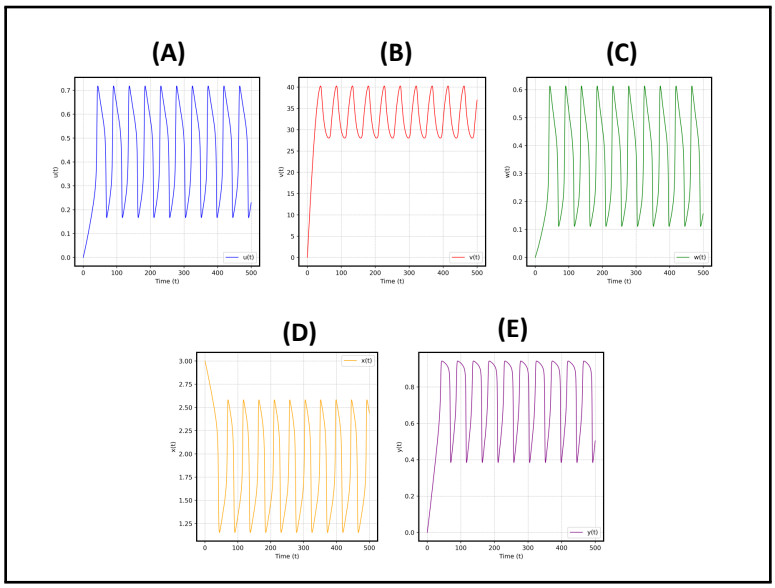

Figure 4.7. *Original time series for the mass action model for the variables u, v, w, x and y respectively.*

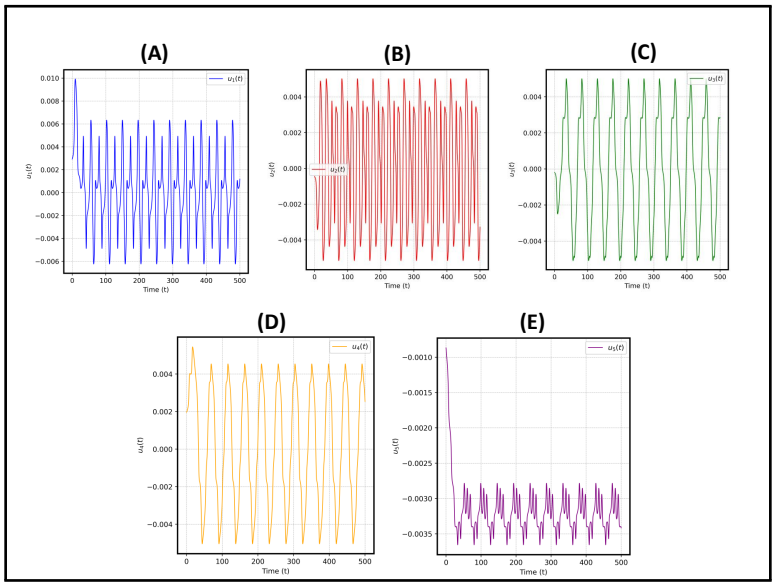

Figure 4.8. *Latent feature extraction using Singular Value Decomposition (SVD) on the delay-embedded time series for the Mass Action Model. (A)–(E) present the first five left singular vectors $u_1(t), \ldots, u_5(t)$ derived after applying Singular Value Decomposition (SVD) to the delay-embedded matrix of the Mass-Action model. The first two panels (A) and (B) clearly represent the dominant oscillatory pattern of the system. The next two panels (C), (D) also exhibit regular oscillations and are visually similar, indicating that they provide further structure in the dynamics. However, because the fourth component contributes far less to overall variance, it is likely to provide only small details or replicate identical aspects collected by earlier components. The fifth panel (E) has no clear pattern and is most likely dominated by noise.*

4.5.4 Example-4: The Oregonator Model

We now apply delay embedding to reconstruct the phase space of the Oregonator model, introduced in **Section 3.4**. The Oregonator is a **three-dimensional nonlinear oscillator** governed by a fast-slow timescale separation. We simulate the system using the parameters in **Table 3.2** and the settings listed in **Table 4.4**. Among the three variables generated, we choose $x(t)$ as the scalar observable to reconstruct the system.

Table 4.4. *Simulation settings for the Oregonator model.*

Setting	IC	t_0	t_f	Δt
Value	[0.1, 0.1, 0.1]	0	100	0.001

We apply the `autcor` function to the time series $x(t)$ which gives $\tau = \mathbf{2993}$ **steps**, which corresponds to $\mathbf{2.993}$ **seconds** based on the sampling interval. With this delay, we determine

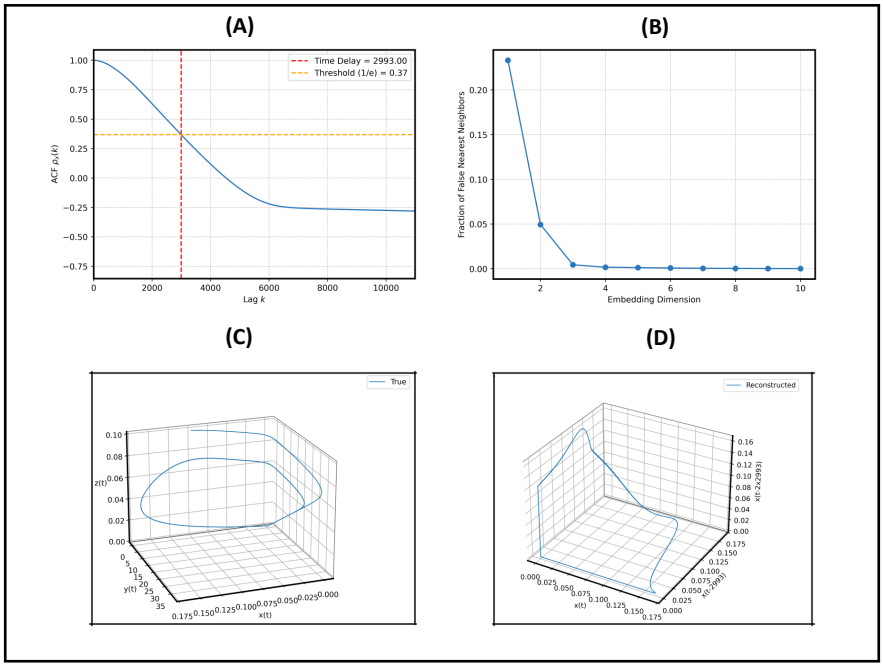

Figure 4.9. *Reconstruction parameters for the Oregonator model using PyTISEAN. (A) Graphical output of the* `autcor` *function for the one dimensional time series from the Oregonator model. The default threshold value* $(1/e)$ *is shown as the horizontal line in the plot. The optimal time delay is selected at lag* $\tau = 2993$ *time steps. (B) Graphical output of the* `false_nearest` *function for the one dimensional time series from the Oregonator model. The function is called with the parameters* `M = 1,10`, `d = 2993`, *and* `f = 2`. *The graph shows an immediate drop-off at* $d = 3$, *indicating that no further embedding dimension is necessary. (C) and (D) show the original and reconstructed phase spaces respectively. The reconstructed trajectory is obtained using* `delay`, *and it clearly preserves the topological features of the original system.*

the embedding dimension using `false_nearest`. The output shows a sharp drop to

zero at $\mathbf{d = 3}$, consistent with the known dimensionality of the system (**Figure 4.9**).

Using the embedding parameters $(\tau, d) = (2993, 3)$, we reconstruct the phase space via delay coordinates $[x(t), x(t-\tau), x(t-2\tau)]$, beginning from $t_0 = 2 \times 2.993$ seconds. A comparison of the original and reconstructed phase space confirms that the essential topological properties are preserved across both the system.

We apply singular value decomposition (SVD) to the embedded data for extracting the latent features. The lower panel in **Figure 4.10** shows the first three singular vectors obtained after applying SVD to the reconstructed space. These latent components show the important time patterns seen in the original system.

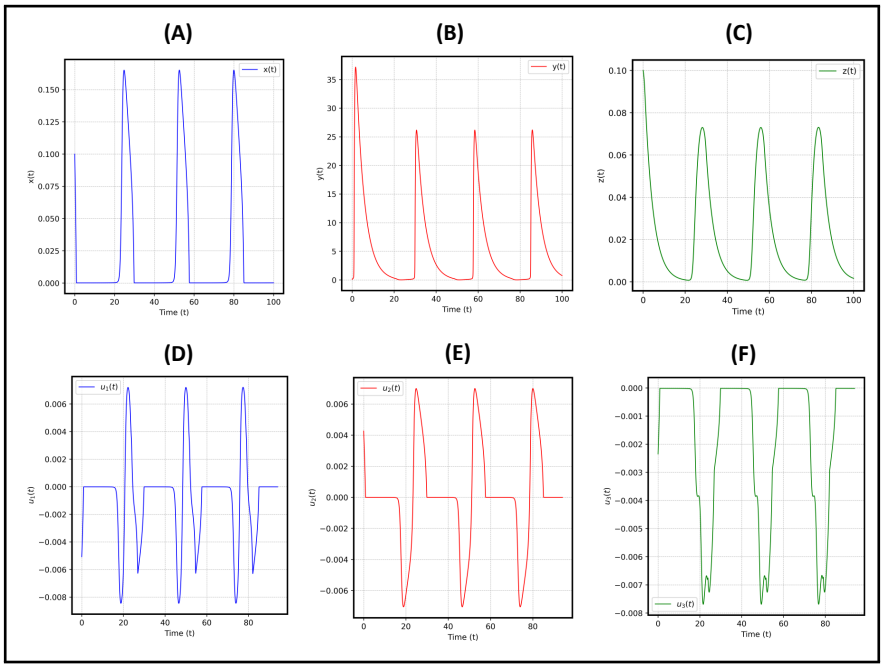

Figure 4.10. *Latent feature extraction using Singular Value Decomposition (SVD) on the delay-embedded time series from the Oregonator model. In the upper panel, (A)–(C) show the original time series for the state variables x, y, and z. In the lower panel, (D)–(F) show the first three latent components obtained by applying SVD to the delay-embedded time series constructed from $x(t)$. The extracted components capture the oscillatory behavior of the system.*

4.5.5 Example-5: Glycolytic Oscillation

As the final example in this section, we consider the phase space reconstruction for the glycolytic oscillator described in **Section 3.5**. This **two-variable nonlinear model** captures metabolic oscillations observed in yeast cells. The time series is generated by numerically integrating the system using the parameters in **Table 3.2** and simulation settings listed in **Table 4.5**. We choose the variable $x(t)$ as the scalar observable used for delay embedding.

Table 4.5. *Simulation settings for the Glycolytic Oscillation model.*

Setting	IC	t_0	t_f	Δt
Value	[0, 0]	0	100	0.001

The `autcor` function when applied to $x(t)$ crosses the $1/e$ line at $\tau = \mathbf{2057}$ **steps**. Therefore, the delay in this case is $\mathbf{2.057}$ **seconds,** for the sampling interval of 0.001. Using the delay, the FNN approach reveals $\mathbf{d = 2}$ as the minimal embedding dimension, beyond which the fraction of false neighbors quickly drops, as illustrated in **Figure 4.11**.

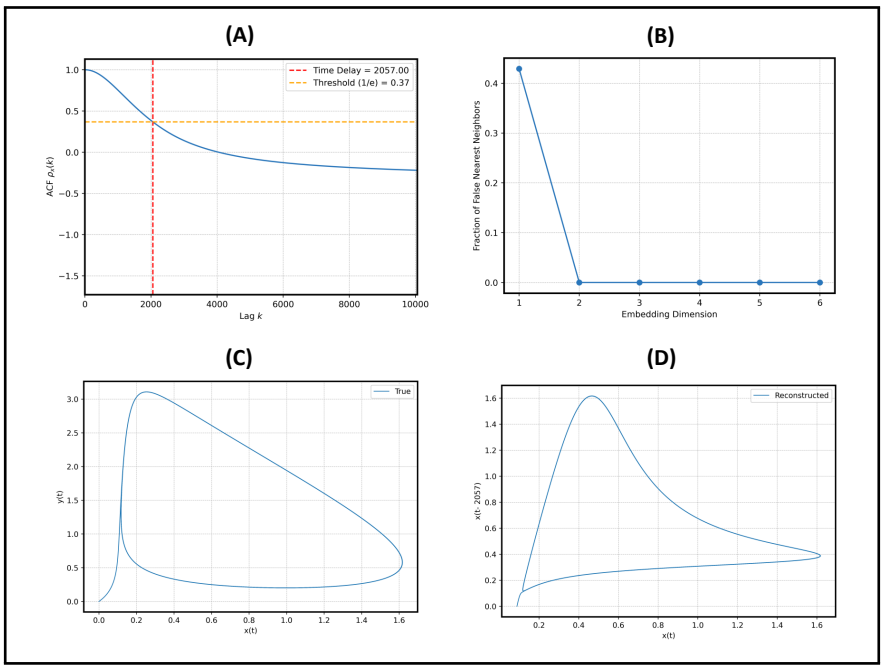

Figure 4.11. *Reconstruction parameters for the Glycolytic Oscillation model using PyTISEAN. (A) Graphical output of the* `autcor` *function for the one dimensional time series from the Glycolytic Oscillation model. The default threshold value* $(1/e)$ *is shown as the horizontal line in the plot. The optimal time delay is selected at lag* $\tau = 2057$*.(B) Graphical output of the* `false_nearest` *function for the one dimensional time series from the Glycolytic Oscillation model. The function was called with the parameters* `M = 1,6`*,* `d = 2057`*, and* `f = 3`*. The graph shows an immediate decline of the percentage of false-nearest neighbors to 0 near* $d = 2$*, indicating that no additional embedding is necessary. (C), (D) represent the original and reconstructed phase space for Glycolytic Oscillation model. The time series for the reconstructed space is obtained by using* `delay`*.*

For the embedding parameters $(\tau, d) = (2057, 2)$, we reconstruct the system dynamics by creating delay vectors $[x(t), x(t-\tau)]$, starting from $t_0 = 2.057$ seconds. The resultant phase diagram preserves the fundamental oscillatory characteristics of the original system, indicating a successful reconstruction.

We further perform singular value decomposition (SVD) on the embedded data to reveal the latent structure in the reconstructed state space. The first two left singular vectors, seen in

Figure 4.12, reflect some of the temporal trends seen in the original system.

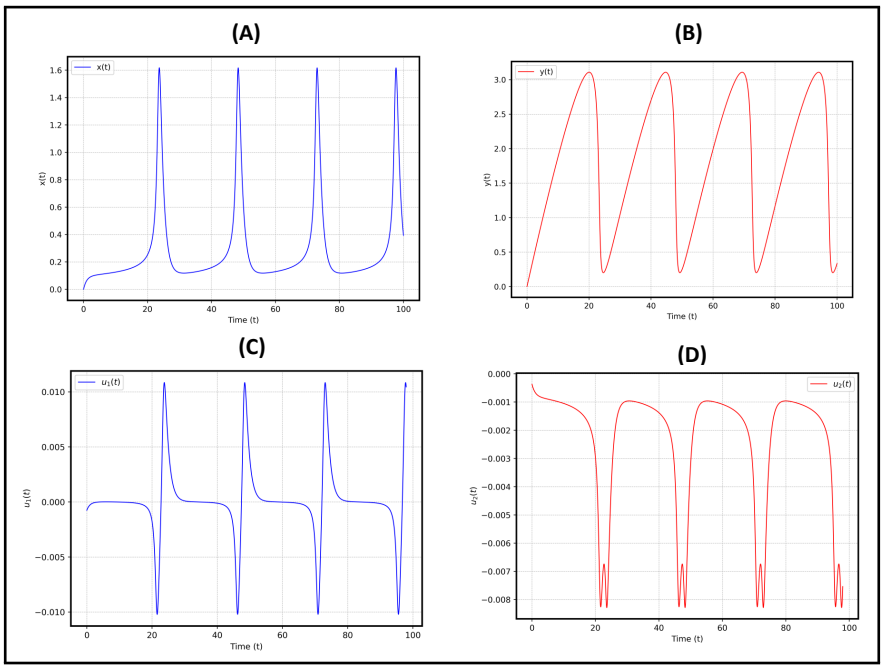

Figure 4.12. *Latent feature extraction using Singular Value Decomposition (SVD) on the delay-embedded time series from the Glycolytic oscillation model. The upper panels (A) and (B) show the original time series for the variables x and y, respectively. The lower panels (C) and (D) display the first two left singular vectors derived from the singular value decomposition (SVD) of the embedded data. These latent components represent the primary oscillatory activity of the system and preserve its fundamental dynamic characteristics.*

Discussion

In this chapter, we applied delay coordinate embedding to five canonical systems, namely FitzHugh–Nagumo, Goodwin, Mass-Action, Oregonator, and Glycolytic Oscillation—to evaluate the efficacy of the PyTISEAN package for phase space reconstruction from scalar time series. In all cases, `autcor` and `false_nearest` provided embedding parameters (τ, d) that effectively unfolded the attractor, preserving the fundamental topological characteristics of the original systems. Moreover, singular value decomposition (SVD) applied to the reconstructed state space consistently revealed the latent structure with the predominant oscillatory behavior.

The current methodology is effective in most aspects, but it has several limitations that became clear during implementation:

- **Uniform delay embedding:** This study employed the conventional method of uniform delay embedding, where delayed vectors are created using a single fixed delay τ. This method, although simple, may not be ideal for systems that evolve across several time scales. A single delay can underresolve either slow or fast dynamics, resulting in one

time-scale to appear as noise in the reconstruction. To overcome this limitation, adaptive or non-uniform embeddings might be used [51].

- **Sensitivity to FNN parameters:** The false-nearest-neighbor (FNN) method requires the selection of a distance ratio threshold f, which is fundamentally subjective and problem-specific. There is no universally accepted value, and various systems may require different thresholds to get meaningful results.
- **Interpretability of SVD components:** The SVD algorithm provides scaled and standardized representations of both the actual data in the first column and estimates of the latent data in the subsequent columns. While the primary components capture prominent dynamics, they do not directly align with the original state variables. Consequently, rescaling or offset correction is required before any further application or biological interpretation.
- For extensive or finely sampled time series, PyTISEAN may surpass internal array restrictions, leading to crashes until the data is down-sampled or the time step is adjusted to a coarser resolution. Furthermore, PyTISEAN has no inherent visualization capabilities, in contrast to alternatives like Teaspoon. Users must depend on other libraries such as `NumPy` and `Matplotlib` for result visualization and further analysis.

Despite these constraints, the reconstructed attractors preserved the topological characteristics, and the latent components consistently captured the principal oscillatory patterns for all the models. Thus, PyTISEAN remains an efficient tool for performing delay embedding of nonlinear deterministic time series. Future research should explore improved embedding methodologies, including non-uniform delays and adaptive feature extraction approaches, for complicated biological data characterized by noise, different time scales, or unknown structure, to provide more accurate and robust reconstructions.

Chapter 5

System Identification using SINDy

Identifying governing equations directly from the data remains a fundamental challenge across many scientific and engineering disciplines. Despite the tremendous rise in data availability, the development of reliable and interpretable models frequently lags behind. This gap is particularly apparent in disciplines like climate science, neurology, ecology, finance, and epidemiology, where the complexity of systems makes it impossible to develop governing equations from fundamental principles [5, 9].

The **S**parse **I**dentification of **N**onlinear **D**ynamics (SINDy) framework addresses this challenge by integrating sparsity-promoting techniques with machine learning to predict fundamental dynamics directly from (possibly noisy) measurement data [9].

The core idea of SINDy is that many physical systems exhibit dynamics driven by a finite number of dominant terms when expressed in an appropriate mathematical basis. This supports the hypothesis that such systems allow for sparse representations within a larger functional space. SINDy utilizes sparse regression to determine the essential terms required for predicting the observed dynamics accurately. This methodology balances interpretability and predictive accuracy, providing simple models that mitigate overfitting [9, 40].

5.1 Sparse Identification of Nonlinear Dynamics (SINDy)

5.1.1 Problem Statement

Consider a nonlinear autonomous system with n state variables $x_1, x_2, \ldots, x_n$, whose dynamics is governed by the equations

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t)\big), \qquad \mathbf{x}(t) \in \mathbb{R}^n, \tag{5.1.1}$$

Here, $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \mathbf{f}\big(\mathbf{x}(t)\big) = (f_1(\mathbf{x}(t)), f_2(\mathbf{x}(t)), ..., f_n(\mathbf{x}(t))^\mathsf{T}$,

$$\dot{x}_k = f_k\big(x_1, x_2, \ldots, x_n\big), \qquad k = 1, 2, \ldots, n. \tag{5.1.2}$$

and for each k, the component function $f_k : \mathbb{R}^n \to \mathbb{R}$ determines the rate of change of the

state variable x_k.

The objective is to build a parsimonious model for each x_k directly from time-series data $\{\mathbf{x}(t_i)\}_{i=1}^m$ using SINDy. Specifically, SINDy looks for a coefficient vector $\xi_\mathbf{k}$ such that

$$\dot{x}_k \approx \Theta(\mathbf{x})\,\xi_k, \quad k = 1, 2, ..., n. \tag{5.1.3}$$

where $\Theta(\mathbf{x})$ represents a library of candidate (nonlinear) functions, also known as a **basis** or a **feature library**, and the vector $\xi_\mathbf{k}$ determines which terms are active in $\Theta(\mathbf{x})$.

5.1.2 Steps involved in SINDy

(1) **Data Collection:** Data is gathered from literature or simulated using established models at several time points $t_1, t_2, \ldots, t_m$ for all state variables. This provides the measurements, $\mathbf{x}(t_i) \in \mathbb{R}^n$ which may be organized as a data matrix

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^\mathsf{T}(t_1) \\ \mathbf{x}^\mathsf{T}(t_2) \\ \vdots \\ \mathbf{x}^\mathsf{T}(t_m) \end{bmatrix} = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \ldots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \ldots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \ldots & x_n(t_m) \end{bmatrix} \in \mathbb{R}^{m \times n}. \tag{5.1.4}$$

The associated time derivatives $\dot{\mathbf{x}}(t_i)$ may be measured or approximated numerically, yielding the derivative matrix

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}^\mathsf{T}(t_1) \\ \dot{\mathbf{x}}^\mathsf{T}(t_2) \\ \vdots \\ \dot{\mathbf{x}}^\mathsf{T}(t_m) \end{bmatrix} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \ldots & \dot{x}_n(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \ldots & \dot{x}_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \ldots & \dot{x}_n(t_m) \end{bmatrix} \in \mathbb{R}^{m \times n}. \tag{5.1.5}$$

(2) **Library Construction:** After acquiring the data, $\mathbf{X}$ and $\dot{\mathbf{X}}$, the library $\boldsymbol{\Theta}(\mathbf{x})$ is constructed as follows:

$$\boldsymbol{\Theta}(\mathbf{x}) = [\mathbf{1}, \mathbf{x}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \ldots, \sin(\mathbf{x}), \cos(\mathbf{x}), \ldots] \in \mathbb{R}^{m \times p}, \tag{5.1.6}$$

Here,

$$\mathbf{x}^{(d)} = \{\, x_1^d,\ x_1^{d-1}x_2,\ \ldots,\ x_n^d \}$$

represents the collection of all monomials of degree d, and p denotes the total number of candidate functions. This, when evaluated at time points, results in the following **feature matrix**:

$$\boldsymbol{\Theta}(\mathbf{X}) = [\mathbf{1}, \mathbf{X}, \mathbf{X}^{(2)}, \mathbf{X}^{(3)}, \ldots, \sin(\mathbf{X}), \cos(\mathbf{X}), \ldots] \in \mathbb{R}^{m \times p}, \tag{5.1.7}$$

(4) **Sparse regression:** The problem in **Eq.** (5.1.3)becomes

$$\dot{\mathbf{X}}_{\mathbf{k}} = \boldsymbol{\Theta}(\mathbf{X})\,\xi_{\mathbf{k}}, \quad k = 1, 2, ..., n \tag{5.1.8}$$

where each $\xi_{\mathbf{k}}$ is found by solving the following sparse linear regression (convex optimization) problem:

$$\xi_k = \arg\min_{\xi'_k} \underbrace{\left\|\dot{X}_k - \boldsymbol{\Theta}(X)\,\xi'_k\right\|_2^2}_{\text{Reconstruction Loss}} + \underbrace{\lambda\, R(\xi'_k)}_{\text{Sparsity Promoting Term}} . \tag{5.1.9}$$

Once the coefficient vectors $\boldsymbol{\xi}_k$ are obtained, substituting them into **Eq.** (5.1.3) yields a closed-form system of differential equations that approximates the original dynamics.

Remarks

- The sparsity on the coefficients is enforced using the regularization term, $R(\hat{\xi})$ which is typically selected from one of the following:
  1. $R(\hat{\xi}) = \lambda\|\hat{\xi}\|_0$ Sequential threshold least squares (STLS) method [9][57].
  2. $R(\hat{\xi}) = \lambda\|\hat{\xi}\|_2$ Ridge regression method [56].
  3. $R(\hat{\xi}) = \lambda\|\hat{\xi}\|_1$ LASSO (**L**east **A**bsolute **S**hrinkage and **S**election **O**perator) method [42].
- The hyperparameter λ, referred to as the regularization parameter, determines the amount of sparsity. A higher λ results in a sparser solution, whereas a lower λ yields a less sparse solution; when $\lambda = 0$, the problem reduces to unconstrained least squares optimization only.
- The original SINDy method employs the sequential threshold Ridge (STRidge) regression technique ($R(\hat{\xi}) = \lambda_1\|\hat{\xi}\|_0 + \lambda_2\|\hat{\xi}\|_2$) to iteratively refine the model by sequentially imposing thresholds on the ridge regression results [9][5].
- LASSO is favored over ridge regression when the objective is sparsity. Ridge regression ($R(\hat{\xi}) = \lambda\|\hat{\xi}\|_2$) penalizes large coefficients to mitigate overfitting without enforcing sparsity, whereas LASSO ($R(\hat{\xi}) = \lambda\|\hat{\xi}\|_1$) employs a $l1$-penalty that not only minimizes overfitting but also reduces certain coefficients to zero, which encourages sparsity in the solution [58] (**Figure 5.1**).

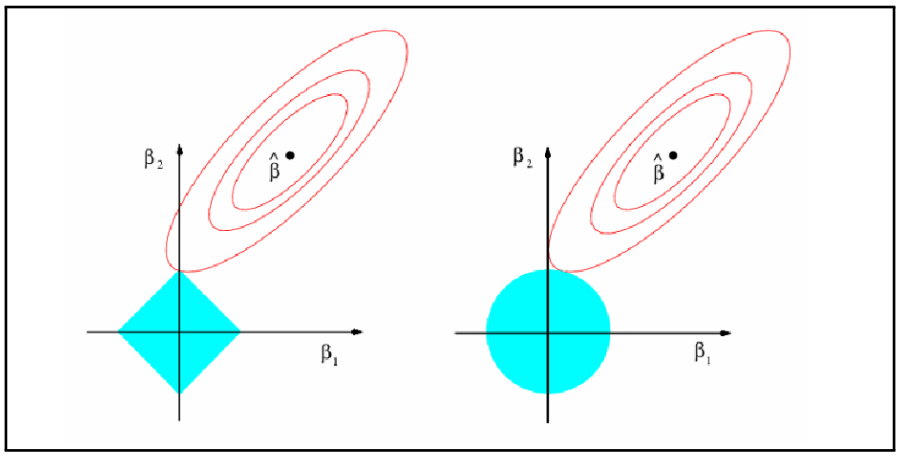

Figure 5.1. *Comparison of Ridge and LASSO regression (adapted from [58]). The left panel illustrates the ℓ_1-penalty used in LASSO (diamond-shaped constraint), which encourages sparse solutions by often intersecting with the feasible region. The right panel shows the ℓ_2-penalty used in Ridge regression (circular constraint), which generally yields non-sparse solutions by shrinking coefficients toward zero but not to exactly zero. Contours (in red) represent levels of the ordinary least squares cost function, and the intersection point $\hat{\beta}$ indicates the optimal solution under each regularization.*

Selection of Terms in the Candidate Library

Library construction is a critical aspect of SINDy. One way is to select the terms in the library based on the prior information about the system. The alternative technique (a practical strategy) is to begin with a small collection of functions, such as low-order polynomials, and gradually expand the library's complexity until a sparse and accurate model is created [9].

Sparse Relaxed Regularized Regression

While LASSO is an efficient method for variable selection, sparsity is sometimes enforced by sparse relaxed regularized regression (SR3). The reason for this is because LASSO ensures both sparsity and accuracy simultaneously via a single optimization problem:

$$\min_{\xi} \frac{1}{2}\|\dot{\mathbf{X}}_{\mathbf{k}} - \Theta(\mathbf{X})\xi\|_2^2 + \lambda\|\xi\|_1, \tag{5.1.10}$$

However, the restriction to the l_1-norm makes LASSO potentially inadequate for problems that may benefit from non-convex penalties (e.g., l_p-norms with $p < 1$) or composite regularizers (e.g., sparsity with smoothness). Also, the penalty imposed by the l_1 norm reduces all coefficients by an amount proportionate to λ, hence, introducing bias, especially for large coefficients. This can make the model less interpretable and affect its predictive performance.

5.1.3 Frequent Terms used in the Candidate Library for Modeling a Biological System: Biochemical Library

When the state variables represent concentrations of biomolecules (e.g., proteins, mRNA, metabolites), the candidate library $\Theta(\mathbf{x})$ must reflect the existing kinetic patterns observed in cell biology. Systems biology suggests that a concise yet comprehensive "biochemical library" often comprises:

1. Linear and first-order degradation

- **Production / basal input:** Bias Term
- **First-order decay:** $-\beta\, x_i$

2. Mass-action interactions (binding, dimerization, catalysis) $k_{ij}\, x_i x_j$.

3. Michaelis-Menten saturation (enzyme kinetics)

$$v_{\max}\, \frac{x_i}{K_M + x_i}.$$

4. Hill-type cooperativity and ultrasensitivity

$$\underbrace{\frac{x^n}{K^n + x^n}}_{\text{activation}}, \qquad \underbrace{\frac{K^n}{K^n + x^n}}_{\text{inhibition}} \quad \text{Typical exponents: } n = 2\text{–}4.$$

5. Cross-regulation terms

$$x_i\, \frac{x_j^n}{K^n + x_j^n}\,, \qquad \frac{x_i^{n_1}}{K_1^{n_1} + x_i^{n_1}}\, \frac{x_j^{n_2}}{K_2^{n_2} + x_j^{n_2}}$$

Incorporating these terms directly into the SINDy feature library speeds up sparse regression efficiency and produces models that are readily interpretable by biologists [10].

Sparse Relaxed Regularized Regression (SR3) addresses these limitations by decoupling the objectives of sparsity and accuracy. It aims to optimize

$$\min_{\xi,\mathbf{w}} \frac{1}{2}\|\dot{\mathbf{X}}_k - \Theta(\mathbf{X})\xi\|_2^2 + \lambda R(\mathbf{w}) + \frac{1}{2\eta}\|\xi - \mathbf{w}\|_2^2, \tag{5.1.11}$$

where $\mathbf{w}$ represents the vector of relaxed coefficients that approximates ξ and η and λ are the hyperparameters. This decoupling eliminates the direct conflict between sparsity and accuracy, resulting in more robust and interpretable solutions [59].

5.2 Tools for Implementation

The Sparse Identification of Nonlinear Dynamics algorithm has been implemented in Python using `PySINDy` package [60, 61]. `PySINDy` is a sparse regression package featuring multiple implementations of the Sparse Identification of Nonlinear Dynamical Systems (SINDy) methodology. It offers a versatile, easy-to-use interface that interacts effortlessly with `NumPy`, `scikit-learn`, and more scientific Python libraries.

Implementing SINDy on experimental data requires many modeling choices, including the method for numerical differentiation, the construction of the candidate-function library, and the selection of a sparse-regression solver. `PySINDy` has integrated support for each of these stages:

1. **Numerical differentiation:** Includes methods like finite-difference schemes and smoothed finite differences for noise-robust derivative estimates.
2. **Feature libraries:** Provides polynomial, Fourier, and fully custom libraries that can be tailored to the underlying biology.
3. **Sparse regression optimizer:** Involves optimization algorithms such as LASSO [42], sequentially thresholded least squares (STLSQ) [9], and SR3 [59], which enable the efficient identification of parsimonious models.

5.3 Application

We use the Sparse Identification of Nonlinear Dynamics (SINDy) framework to analyze the canonical models described in **Chapter 3**. Each model was studied utilizing a unified workflow including simulation, preprocessing, library construction, sparse regression, and validation, as described in **Figure 5.2**. The underlying methodology remained unchanged, but the feature libraries and sparsity parameters were adjusted to each system, as detailed in **Table 5.1**.

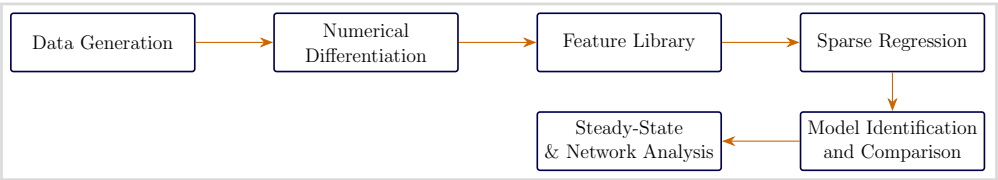

Figure 5.2. *Schematic overview of the SINDy workflow*

Methodology Overview

The overall procedure consists of the following key steps: numerical simulation of the system, preparation of time-series data, construction of a candidate function library, sparse regression for model identification, and validation of the inferred dynamics via networks.

1. **Numerical Simulation and Data Generation:** For each system, time-series data were generated by numerically integrating the original differential equations using known initial conditions and parameter values (**Table 3.2**).

2. **Numerical Differentiation:** A smoothed finite difference scheme was employed to determine time derivatives from the simulated trajectories. This method provides robustness to any numerical noise.

3. **Candidate Libraries:** For all models except the Goodwin model, two types of candidate libraries were constructed:

   - **Prior-informed library:** Includes only those terms that are known to be present in the original model equations.
   - **Polynomial library:** Contains all monomials in the state variables up to degree 3 (since the maximum degree involved in the equations was 3).

   These libraries serve as the feature space for the sparse regression step.

4. **Sparse Regression:** Sparse regression was then carried out using the Sequentially Thresholded Least Squares (STLSQ) technique, which repeatedly refines the solutions by switching between least squares estimation and thresholding. In each iteration, coefficients with magnitudes less than a defined threshold are adjusted to zero, encouraging sparsity in the final model. By default, this approach uses sequentially thresholded Ridge regression to balance model accuracy and regularization.

5. **Model Validation and Interpretation:**

   - **Simulation comparison:** Trajectories of the original and identified models were compared.
   - **Steady-state analysis:** Equilibrium points were computed to verify alignment with the original system.
   - **Jacobian analysis and interaction network construction:** The Jacobian matrix was evaluated at the steady state to uncover the local regulatory interactions. The sign structure of the Jacobian was then visualized as a directed graph, showing excitatory and inhibitory influences.

Table 5.1. *Candidate library and hyperparameter values for different models.*

S.No	**Model**	**Library Terms**	**Hyperparameters** $(Alpha, Threshold)$
1	FitzHugh–Nagumo Model	(I) $[1, x, x^2, x^3, y]$ (II) $[1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3]$	(I) $\alpha = 0.05$, $threshold$ = 0.1 (II) $\alpha = 1.8 \times 10^{-6}$, $threshold = 6.2 \times 10^{-2}$
2	Goodwin Model	(I) $[x, y, z, 1/(z^{10}+1)]$	(I) $\alpha = 1.6 \times 10^{-5}$, $threshold = 1.0 \times 10^{-3}$
3	Mass Action Model	(I) $\begin{bmatrix} 1, u, v, w, x, y, uv, \\ uy, vw, wx, wy, xy, y^2 \end{bmatrix}$ (II) $\begin{bmatrix} 1, u, v, w, x, y, \\ u^2, v^2, w^2, x^2, y^2, \\ uv, uw, ux, uy, vw, \\ vx, vy, wx, wy, xy \end{bmatrix}$	(I) $\alpha = 1.0 \times 10^{-14}$, $threshold$ = 0.025 (II) $\alpha = 6.6 \times 10^{-14}$, $threshold$ = 0.025
4	Oregonator Model	(I) $[x, y, z, xy, x^2]$ (II) $[1, x, y, z, x^2, xy, xz, y^2, yz, z^2]$	(I) $\alpha = 1.1 \times 10^{-4}$, $threshold$ = 0.002 (II) α= 1.0×10^{-10}, $threshold$= 3.0×10^{-4}
5	Glycolytic Oscillation	(I) $[1, x, x^2y]$ (II) $[1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3]$	(I) $\alpha = 0.05$, $threshold$ = 0.1 (II) $\alpha = 1.0 \times 10^{-5}$, $threshold = 1.0 \times 10^{-2}$

5.3.1 Example 1: FitzHugh–Nagumo Model

Recall the FitzHugh–Nagumo (FHN) model, introduced in **Section 3.1**. This is a two-variable system that captures the essential dynamics of excitable systems, including neurons and cardiac tissue. The SINDy pipeline, as outlined in Section 5.3, was employed to infer the governing equations. The simulation parameters and candidate libraries used for model identification are provided below:

Table 5.2. *Settings for simulation of FitzHugh-Nagumo model.*

Setting	IC	t_0	t_f	Δt
Value	[0.1, 0.0]	0	100	0.001

Candidate Library:

Two types of candidate libraries were chosen to train SINDy:

(1) Candidate library consisting of the terms present in the ODEs [use of prior information]:

$$\Theta_1(X) = \left[\ 1,\ x,\ x^2,\ x^3,\ y\ \right]$$

(2) Library consisting of all monomials in x and y up to degree 3 to ensure that all nonlinearities present in the original model:

$$\Theta_2(X) = \left[1,\ x,\ y,\ x^2,\ xy,\ y^2,\ x^3,\ x^2y,\ xy^2,\ y^3\right]$$

Identified Equations:

For both the candidate libraries and for the choice of specific hyperparameters (**Table 5.1**), the SINDy algorithm converged to the same set of equations:

$$\boxed{\begin{aligned}\dot{x} &= 0.999\,x - 1.001\,y + 1.000\,x^2 - 0.999\,x^3,\\ \dot{y} &= 0.300\,x - 0.150\,y.\end{aligned}}$$

These equations closely resemble the original system, exhibiting negligible variations in coefficients **(Table 5.3 and 5.4)**. Also, the time-series simulations of both the original and SINDy models exhibit nearly identical trajectories, further confirming that the inferred system accurately captures the dynamics **(Figure 5.3)**.

Table 5.3. *Comparison of coefficients for the FitzHugh-Nagumo model for library* (5.3.1)

S.No	**Term**	$\dot{x}$		$\dot{y}$	
		True	**Estimated**	**True**	**Estimated**
1	1	0.000	0.000	0.000	0.000
2	x	1.000	0.999	0.300	0.300
3	x^2	1.000	1.000	0.000	0.000
4	x^3	-1.000	-0.999	0.000	0.000
5	y	-1.000	-1.001	-0.150	-0.150

Table 5.4. *Comparison of coefficients for the FitzHugh-Nagumo model for library* (5.3.1)

S.No	**Term**	$\dot{x}$		$\dot{y}$	
		True	**Estimated**	**True**	**Estimated**
1	1	0	0.000	0	0.000
2	x	1	0.999	0.3	0.300
3	y	-1	-1.001	-0.15	-0.150
4	x^2	1	1.000	0	0.000
5	xy	0	0.000	0	0.000
6	y^2	0	0.000	0	0.000
7	x^3	-1	-0.999	0	0.000
8	x^2y	0	0.000	0	0.000
9	xy^2	0	0.000	0	0.000
10	y^3	0	0.000	0	0.000

Steady-State and Network Analysis:

A steady-state analysis of the SINDy-inferred system gave a steady state at $(x^*, y^*) = (0, 0)$, matches with the analytical steady state of the original model. To reveal the underlying regulatory interactions, the Jacobian matrix was computed at this point, and its sign structure was used to construct the local interaction network. The sign structure of the Jacobian and the corresponding interaction network are shown in **Figure 5.3**).

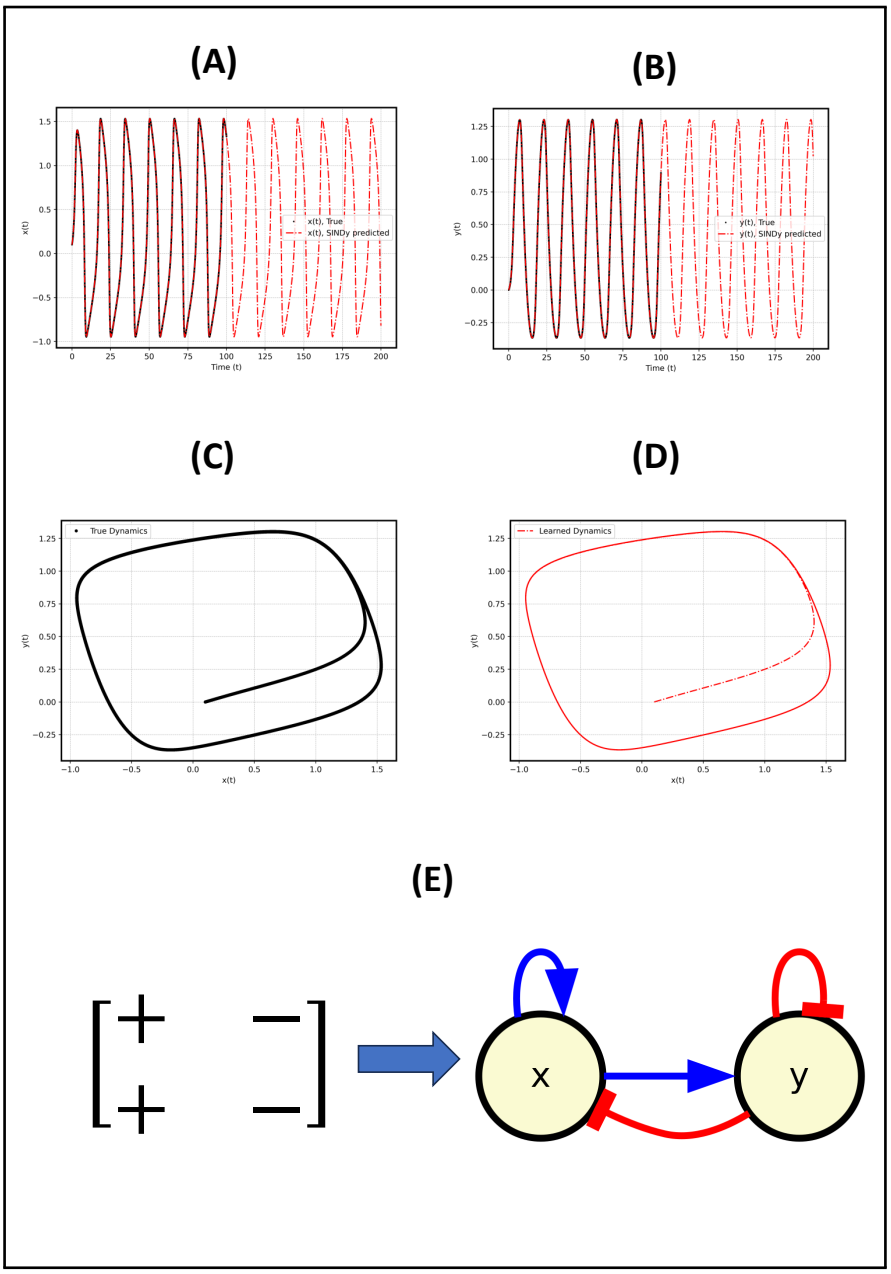

Figure 5.3. *Results from the Sparse Identification of Nonlinear Dynamics (SINDy) method applied to the FitzHugh–Nagumo model. (A) and (B) show the time series of the FitzHugh–Nagumo model for variables x and y, respectively. The black curves represent the true dynamics, while the red curves correspond to the SINDy-simulated trajectories. (C) and (D) display the corresponding phase diagram in the x–y plane for the original and SINDy-learned model, respectively. The similarity between the two diagrams indicates that the learned model effectively captures the underlying system dynamics. (E) depicts the sign structure of the Jacobian matrix evaluated at the steady state $(x^*, y^*) = (0, 0)$, along with the resulting interaction network. Blue arrows indicate positive regulation (activation), and red arrows indicate negative influences (inhibition).*

5.3.2 Example 2: The Goodwin Model

The Goodwin model, introduced in **Section 3.2**, describes a three-variable biological oscillator based on negative feedback with time delays, often used to model circadian rhythms and transcriptional regulation. The SINDy pipeline, as outlined in **Section 5.3**, was employed to infer the governing equations. The simulation parameters and candidate libraries used for model identification are provided below:

Table 5.5. *Settings for simulation of the Goodwin Model.*

Setting	IC	t_0	t_f	Δt
Value	[0.01, 0, 0]	0	1000	0.01

Candidate Library:

In this case, the library used consists of the terms present in the ODEs:

$$\Theta(X) = \left[x,\, y,\, z,\, \frac{1}{z^{10}+1}\right]$$

Identified Equations:

The system of equations inferred by SINDy algorithm is

$$\begin{aligned}
\dot{x} &= -0.100\,x + 1.000\,\frac{1}{z^{10}+1},\\
\dot{y} &= 1.000\,x - 0.100\,y,\\
\dot{z} &= 1.000\,y - 0.100\,z.
\end{aligned}$$

These equations match the original Goodwin model in structure and show negligible numerical deviation in coefficient values, indicating successful model recovery **(Table 5.6)**. Further, the time-series simulations of both the original and SINDy models exhibit nearly identical trajectories **(Figure 5.4)**.

Table 5.6. *Comparison of coefficients for the Goodwin model for library* (5.3.2)

S.No	**Term**	$\dot{x}$		$\dot{y}$		$\dot{z}$	
		True	**Estimated**	**True**	**Estimated**	**True**	**Estimated**
1	x	-0.1	-0.100	1.0	1.000	0	0
2	y	0	0	-0.1	-0.100	1.0	1.000
3	z	0	0	0	0	-0.1	-0.100
4	$\frac{1}{z^{10}+1}$	1.0	1.000	0	0	0	0

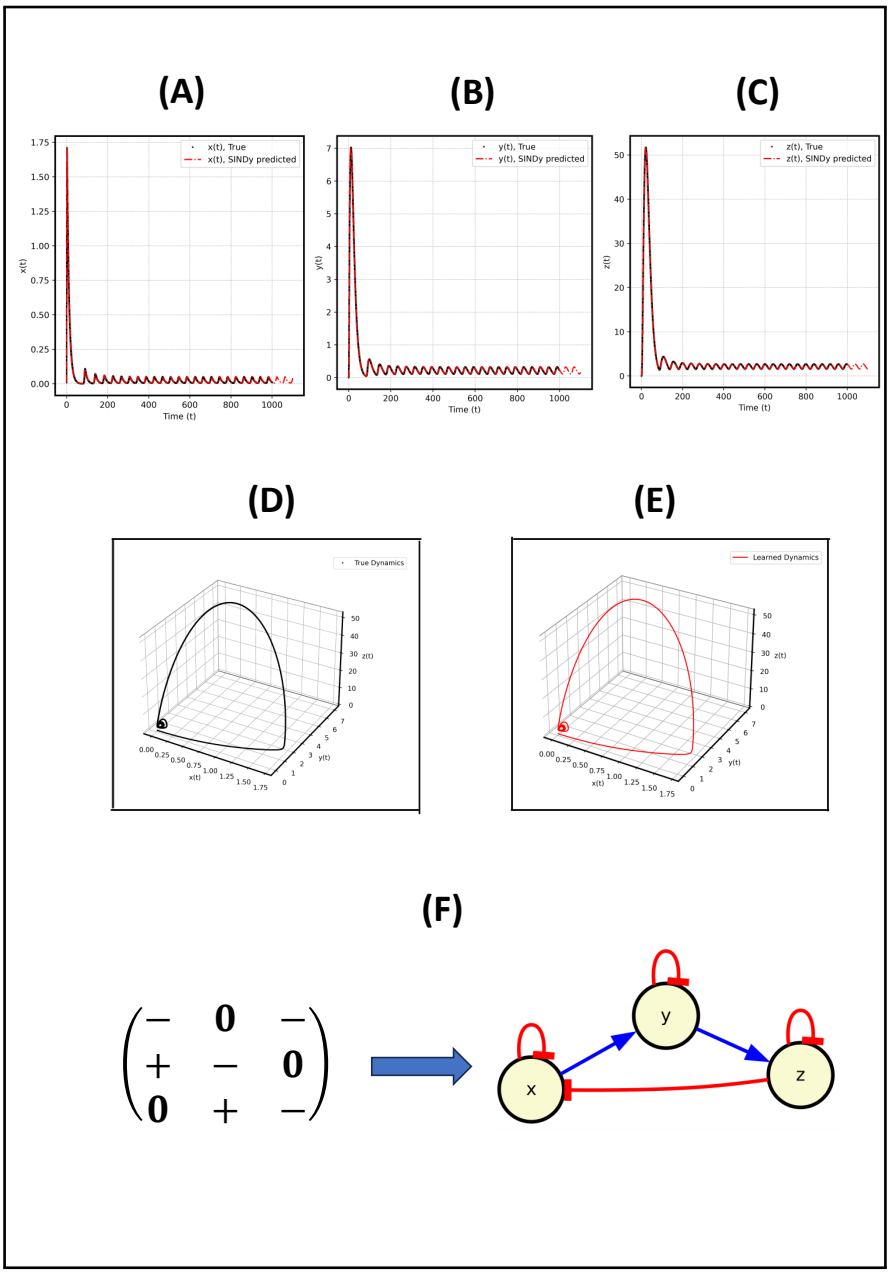

Figure 5.4. *Results from the Sparse Identification of Nonlinear Dynamics (SINDy) method applied to the Goodwin model. (A)-(C) show the time series of the Goodwin model for variables x, y, and z, respectively. The black curves represent the original dynamics, and the red curves correspond to the SINDy-simulated trajectories. (D) and (E) display the 3D phase diagram for the original and SINDy-learned model, respectively. The similarity between the two diagrams indicates that the learned model effectively captures the underlying system dynamics. (F) illustrates the sign structure of the Jacobian matrix evaluated at the steady state $(x^*, y^*, z^*) = (0.0187, 0.187, 1.87)$ and corresponding interaction network. Red arrows indicate negative regulation (inhibition), and blue arrows positive regulation (activation).*

Steady-State and Network Analysis:

Steady-state analysis of the SINDy-learned model revealed a fixed point at approximately $(x^*, y^*, z^*) = (0.0187,\ 0.187,\ 1.87)$, aligning with the analytical steady state of the original model. The local regulatory interactions were obtained using the sign structure of the Jacobian evaluated at the steady state, as shown in **Figure 5.4**.

5.3.3 Example 3: Mass–Action Model

The mass-action model, introduced in **Section 3.3**, describes the early embryonic cell-cycle oscillator using five molecular species. The governing equations were inferred using the SINDy pipeline outlined in **Section 5.3**. Details of the simulation parameters and candidate function libraries are given below:

Table 5.7. *Settings for simulation of Mass action Model.*

Setting	IC	t_0	t_f	Δt
Value	[0,0,0,3 0]	0	100	0.001

Candidate Libraries:

SINDy was performed using the following two libraries:

1. Library including only those terms known to appear in the ODEs:

$$\Theta_1(X) = [\,1,\ u,\ v,\ w,\ x,\ y,\ uv,\ uy,\ vw,\ wx,\ wy,\ xy,\ y^2\,] \tag{5.3.1}$$

2. Library containing all monomials up to the second degree in each variable:

$$\begin{aligned}\Theta_2(X) = [\,1,\ u,\ v,\ w,\ x,\ y,\ u^2,\ v^2,\ w^2,\ x^2,\ y^2,\\ uv,\ uw,\ ux,\ uy,\ vw,\ vx,\ vy,\ wx,\ wy,\ xy\,]\end{aligned} \tag{5.3.2}$$

Identified Equations:

For particular choices of hyperparameters, SINDy identified the same governing equations for both libraries:

$$\begin{aligned}
\dot{u} &= -100.000\,u \ +\ 0.400\,v \ -\ 0.400\,u\,v \ +\ 100.000\,u\,y,\\
\dot{v} &= 1.500\cdot 1 \ -\ 0.100\,u\,v,\\
\dot{w} &= 0.050\,v \ -\ 20.000\,w \ -\ 0.050\,v\,w \ +\ 20.000\,w\,y,\\
\dot{x} &= 4.500\,y \ -\ 6.000\,w\,x,\\
\dot{y} &= 300.000\cdot 1 \ -\ 100.000\,x \ -\ 405.500\,y \ +\ 100.000\,x\,y \ +\ 100.000\,y^2
\end{aligned}$$

These equations match the original Mass Action model and show negligible numerical deviation in coefficient values, as shown in **(Table 5.8 and 5.9)**. Also, the time-series simulations of both the original and SINDy models exhibit nearly identical trajectories, further confirming that the inferred system accurately captures the dynamics **(Figure 5.5)**.

Steady-State and Network Analysis.

Steady-state analysis of the SINDy-learned model revealed a fixed point at approximately $(u^*, v^*, w^*, x^*, y^*) = (0.501, 29.9, 0.386, 1.71, 0.881)$. The sign structure of the Jacobian evaluated at this steady state and the corresponding interaction network is shown in **Figure 5.5**.

Table 5.8. *Comparison of coefficients for the Mass–Action model for library 5.3.1*

S.No	Term	$\dot{u}$		$\dot{v}$		$\dot{w}$		$\dot{x}$		$\dot{y}$	
		True	Est.	True	Est.	True	Est.	True	Est.	True	Est.
1	1	0.000	0.000	1.500	1.500	0.000	0.000	0.000	0.000	300.000	300.003
2	u	-100.000	-100.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	v	0.400	0.400	0.000	0.000	0.050	0.050	0.000	0.000	0.000	0.000
4	w	0.000	0.000	0.000	0.000	-20.000	-20.000	0.000	0.000	0.000	0.000
5	x	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	-100.000	-100.001
6	y	0.000	0.000	0.000	0.000	0.000	0.000	4.500	4.500	-405.500	-405.503
7	uv	-0.400	-0.400	-0.100	-0.100	0.000	0.000	0.000	0.000	0.000	0.000
8	uy	100.000	100.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	vw	0.000	0.000	0.000	0.000	-0.050	-0.050	0.000	0.000	0.000	0.000
10	wx	0.000	0.000	0.000	0.000	0.000	0.000	-6.000	-6.000	0.000	0.000
11	wy	0.000	0.000	0.000	0.000	20.000	20.000	0.000	0.000	0.000	0.000
12	xy	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	100.000	100.001
13	xw	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	y^2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	100.000	100.001

Table 5.9. *Comparison of coefficients for the Mass–Action model for library 5.3.2*

S.No	Term	$\dot{u}$		$\dot{v}$		$\dot{w}$		$\dot{x}$		$\dot{y}$	
		True	**Est.**	**True**	**Est.**	**True**	**Est.**	**True**	**Est.**	**True**	**Est.**
1	1	0.000	0.000	1.500	1.500	0.000	0.000	0.000	0.000	300.000	300.000
2	u	-100.000	-100.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	v	0.400	0.400	0.000	0.000	0.050	0.050	0.000	0.000	0.000	0.000
4	w	0.000	0.000	0.000	0.000	-20.000	-20.000	0.000	0.000	0.000	0.000
5	x	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	-100.000	-100.000
6	y	0.000	0.000	0.000	0.000	0.000	0.000	4.500	4.500	-405.500	-405.500
7	u^2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
8	v^2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
9	w^2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
10	x^2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	100.000	100.000
11	y^2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
12	uv	-0.400	-0.400	-0.100	-0.100	0.000	0.000	0.000	0.000	0.000	0.000
13	uw	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
14	ux	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
15	uy	100.000	100.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
16	vw	0.000	0.000	0.000	0.000	-0.050	-0.050	0.000	0.000	0.000	0.000
17	vx	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
18	vy	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
19	wx	0.000	0.000	0.000	0.000	0.000	0.000	-6.000	-6.000	0.000	0.000
20	wy	0.000	0.000	0.000	0.000	20.000	20.000	0.000	0.000	0.000	0.000
21	xy	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	100.000	100.000

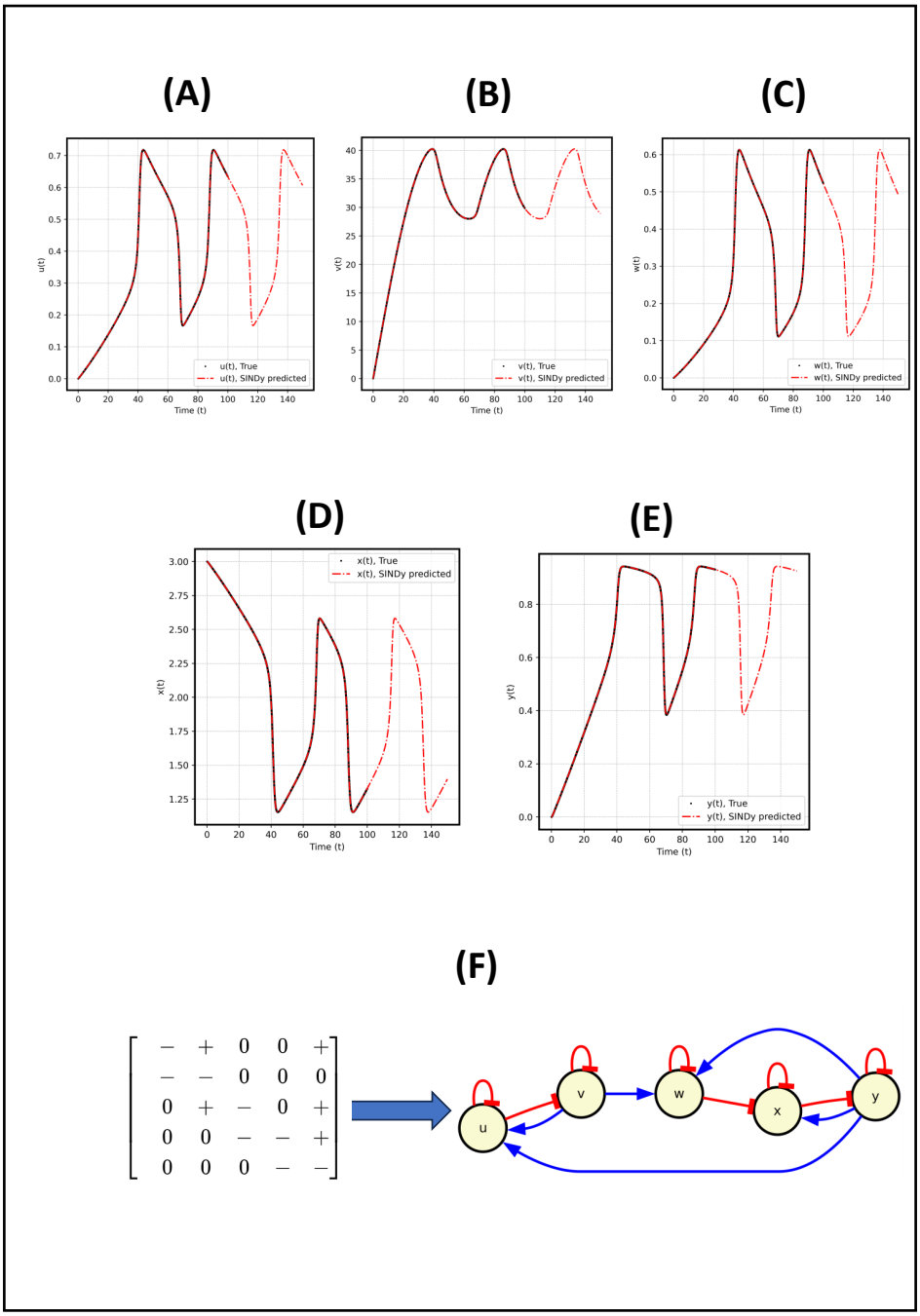

Figure 5.5. *Results from the Sparse Identification of Nonlinear Dynamics method applied to the Mass-Action model. (A–E) show time series trajectories of the Mass-Action model for variables u, v,w, x and y, respectively. The black curves represent the true dynamics, while the red curves correspond to the SINDy-simulated trajectories. (F) the sign structure of the Jacobian matrix evaluated at the steady state $(u^*, v^*, w^*, x^*, y^*) = (0.501, 29.9, 0.386, 1.71, 0.881)$, along with the resulting interaction network. Blue arrows indicate positive regulation (activation), and red arrows indicate negative regulation (inhibition).*

5.3.4 Example 4: Oregonator Model

The Oregonator model, introduced in **Section 3.4**, encapsulates the nonlinear oscillatory dynamics of the Belousov–Zhabotinsky (BZ) chemical reaction using three interacting variables. The simulation parameters and the libraries for implementing the SINDy algorithm are given below:

Table 5.10. *Settings for simulation of the Oregonator Model.*

Setting	IC	t_0	t_f	Δt
Value	[0.1, 0.1, 0.1]	0	1000	0.01

Candidate Libraries:

1. Candidate library containing all the terms present in the original ODEs:

$$\Theta_1(X) = [x,\ y,\ z,\ xy,\ x^2] \tag{5.3.3}$$

2. Candidate library incorporating all monomials upto degree 2:

$$\Theta_2(X) = [1,\ x,\ y,\ z,\ x^2,\ xy,\ xz,\ y^2,\ yz,\ z^2] \tag{5.3.4}$$

Identified Equations:

In this case, SINDy gave a different set of equations corresponding to each library. For the library (Θ_1), the equations identified by the SINDy are

$$\begin{aligned}
\dot{x} &= 2.983\,x + 0.002\,y - 14.928\,xy - 14.909\,x^2,\\
\dot{y} &= -0.031\,x - 1.800\,y + 1827.942\,z - 11249.478\,xy + 0.369\,x^2,\\
\dot{z} &= 0.200\,x - 0.250\,z.
\end{aligned}$$

In contrast, using the library (Θ_2), SINDy identified the following equations:

$$\begin{aligned}
\dot{x} &= 2.983\,x + 0.002\,y - 0.002\,z - 14.903\,x^2 - 14.915\,xy - 0.025\,xz + 0.010\,z^2,\\
\dot{y} &= -0.019 \cdot 1 + 0.361\,x - 1.798\,y + 1828.680\,z - 1.070\,x^2 - 11251.686\,xy\\
&\quad - 3.061\,xz - 0.043\,yz - 2.032\,z^2,\\
\dot{z} &= 0.200\,x - 0.250\,z + 0.001\,x^2 + 0.004\,xz - 0.004\,z^2
\end{aligned}$$

Although both inferred models incorporate additional terms beyond those in the original system, they successfully retain all the true governing terms. Furthermore, the coefficients associated with the common terms closely match the original values. Despite additional

terms, simulations of the inferred models nearly mimic the genuine system trajectories, as demonstrated in **Figure 5.6**.

Table 5.11. *Comparison of coefficients for the Oregonator model for library* (5.3.3)

S.No	Term	$\dot{x}$		$\dot{y}$		$\dot{z}$	
		True	**Est.**	**True**	**Est.**	**True**	**Est.**
1	x	2.985	2.983	0.000	-0.031	0.200	0.200
2	y	0.002	0.002	-1.800	-1.800	0.000	0.000
3	z	0.000	0.000	1827.897	1827.942	-0.250	-0.250
4	xy	-14.925	-14.928	-11248.594	-11249.478	0.000	0.000
5	x^2	-14.925	-14.909	0.000	0.369	0.000	0.000

Table 5.12. *Comparison of coefficients for the Oregonator model for library* (5.3.4)

S.No	Term	$\dot{x}$		$\dot{y}$		$\dot{z}$	
		True	**Est.**	**True**	**Est.**	**True**	**Est.**
1	1	0.000	0.000	0.000	-0.019	0.000	0.000
2	x	2.985	2.983	0.000	0.361	0.200	0.200
3	y	0.002	0.002	-1.800	-1.798	0.000	0.000
4	z	0.000	-0.002	1827.897	1828.680	-0.250	-0.250
5	x^2	-14.925	-14.903	0.000	-1.070	0.000	0.001
6	xy	-14.925	-14.915	-11248.594	-11251.686	0.000	0.000
7	xz	0.000	-0.025	0.000	-3.061	0.000	0.004
8	y^2	0.000	0.000	0.000	0.000	0.000	0.000
9	yz	0.000	0.000	0.000	-0.043	0.000	0.000
10	z^2	0.000	0.010	0.000	-2.032	0.000	-0.004

Steady-State and Network Analysis:

For the model inferred using the library (Θ_1), two steady states were identified, $(x, y, z) = (0.0705,\ 0.130,\ 0.0564)$ and $(0, 0, 0)$. In contrast, the model inferred using library (Θ_2), only the non-trivial steady state $(0.0705, 0.130, 0.0564)$ was recovered. The Jacobian matrix was evaluated at both the steady states. The sign structure of the Jacobian and the corresponding interaction network at both steady states is shown in **Figure 5.6**.

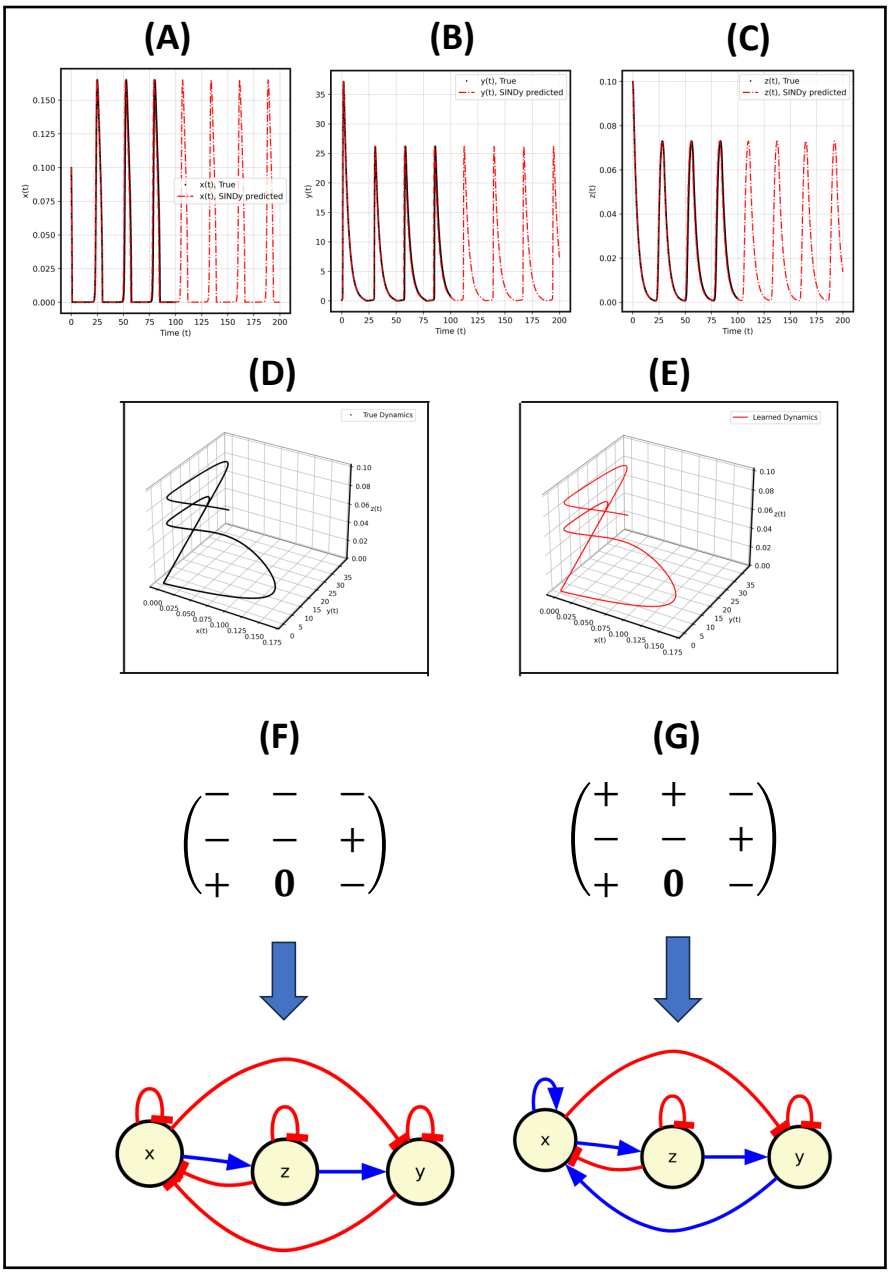

Figure 5.6. *Results from the Sparse Identification of Nonlinear Dynamics method applied to the Oregonator model. (A)-(C) show the time series of the Oregonator model for variables x, y, and z, respectively. The black curves represent the original dynamics, and the red curves correspond to the SINDy-simulated trajectories. (D–E) display the 3D phase diagram for the original and SINDy-learned model, respectively. The similarity between the two diagrams indicates that the learned model effectively captures the underlying system dynamics. (F–G) Jacobian sign patterns and inferred interaction networks: (F) corresponds to the steady state $(0.0705,\ 0.130,\ 0.0564)$, while (G) corresponds to the steady state $(0,\ 0,\ 0)$. The lower panel illustrates the corresponding networks where blue arrows represent activation, and red arrows represent inhibition.*

Although the steady states and time-series trajectories align closely between the original

and inferred models, the interaction network differs from the original model. This can further be analyzed using the entries and their magnitude in the Jacobian. Small but non-zero Jacobian entries introduce weak spurious interactions in the inferred interaction network. These weak terms, while numerically insignificant, can alter the network topology if not thresholded during interpretation. This highlights that accurately reproducing the dynamics does not always guarantee faithful recovery of the underlying interaction structure (**Figure 5.7**).

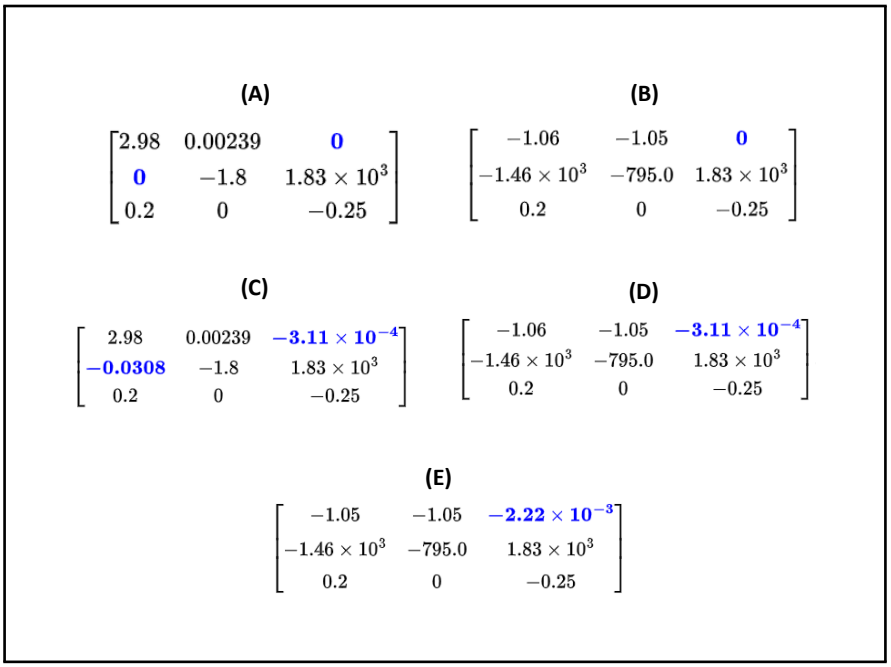

Figure 5.7. *Comparison of the Jacobian evaluated at steady states for the original and SINDy-identified equations for the Oregonator model. The original model and the SINDy-identified model using library 5.3.3 share nearly the same steady states: (1)* $(0, 0, 0)$ *and (2)* $(0.0705, 0.130, 0.0564)$ *respectively. In contrast, the model identified using library 5.3.4 exhibits only steady state (2), due to the presence of a small bias term in the inferred equation for* $\dot{y}$*. (A)–(B) show the Jacobians at steady states (1) and (2) for the original model. (C)–(D) show the Jacobians at the same points for the SINDy-inferred model using library 5.3.3. (E) shows the Jacobian at steady state (2) for the SINDy-inferred model using library 5.3.4. Small but non-zero Jacobian entries (highlighted in blue) introduce weak spurious interactions in the inferred interaction network.*

5.3.5 Example 5: Glycolytic Oscillation Model

The final example is the Selkov model for glycolytic oscillation, described in **Section 3.5**, is a two-variable system that exhibits metabolic oscillations typical of yeast cells. The simulation settings and candidate libraries for performing SINDy are given below:

Table 5.13. *Settings for simulation of Glycolytic Oscillation model.*

Setting	IC	t_0	t_f	Δt
Value	[0.1, 0.0]	0	100	0.001

Candidate Libraries:

1. Candidate library containing all the terms present in the original ODEs:

$$\Theta_1(X) = [1,\ x,\ x^2 y] \tag{5.3.5}$$

2. Candidate library containing all monomials up to degree 3:

$$\Theta_2(X) = [1,\ x,\ y,\ x^2,\ xy,\ y^2,\ x^3,\ x^2 y,\ xy^2,\ y^3] \tag{5.3.6}$$

Identified Equations:

For the choice of hyperparameters in **Table 5.1**, SINDy gave the same models for both the libraries:

$$\begin{aligned} \dot{x} &= 0.100 \cdot 1 \ - \ 1.000\, x \ + \ 1.000\, x^2 y, \\ \dot{y} &= 0.200 \cdot 1 \ - \ 1.000\, x^2 y \end{aligned}$$

The identified equations accurately reproduce the structure of the original model, effectively recovering the true system dynamics **(Table 5.14 and 5.15)**. Moreover, the time-series trajectories produced by the SINDy-inferred models strongly align with those of the actual system **(Figure 5.8)**.

Steady-State and Network Analysis:

A steady-state analysis of the SINDy-inferred system revealed a steady state at $(x^*, y^*) = (0.300,\ 2.220)$, , matching with the analytical steady state of the original model. The Jacobian matrix was analyzed to determine the regulatory interactions near steady state. The sign structure of the Jacobian and the corresponding interaction diagram are illustrated in **Figure 5.8**.

Table 5.14. *Comparison of coefficients for the simple model*

S.No	**Term**	$\dot{x}$		$\dot{y}$	
		True	**Est.**	**True**	**Est.**
1	1	0.1	0.100	0.2	0.200
2	x	-1.0	-1.000	0.0	0.000
3	x^2y	1.0	1.000	-1.0	-1.000

Table 5.15. *Comparison of coefficients for the simple two-variable model*

S.No	**Term**	$\dot{x}$		$\dot{y}$	
		True	**Est.**	**True**	**Est.**
1	1	0.1	0.1	0.2	0.2
2	x	-1.0	-1.0	0.0	0.0
3	y	0.0	0.0	0.0	0.0
4	x^2	0.0	0.0	0.0	0.0
5	xy	0.0	0.0	0.0	0.0
6	y^2	0.0	0.0	0.0	0.0
7	x^3	0.0	0.0	0.0	0.0
8	x^2y	1.0	1.0	-1.0	-1.0
9	xy^2	0.0	0.0	0.0	0.0
10	y^3	0.0	0.0	0.0	0.0

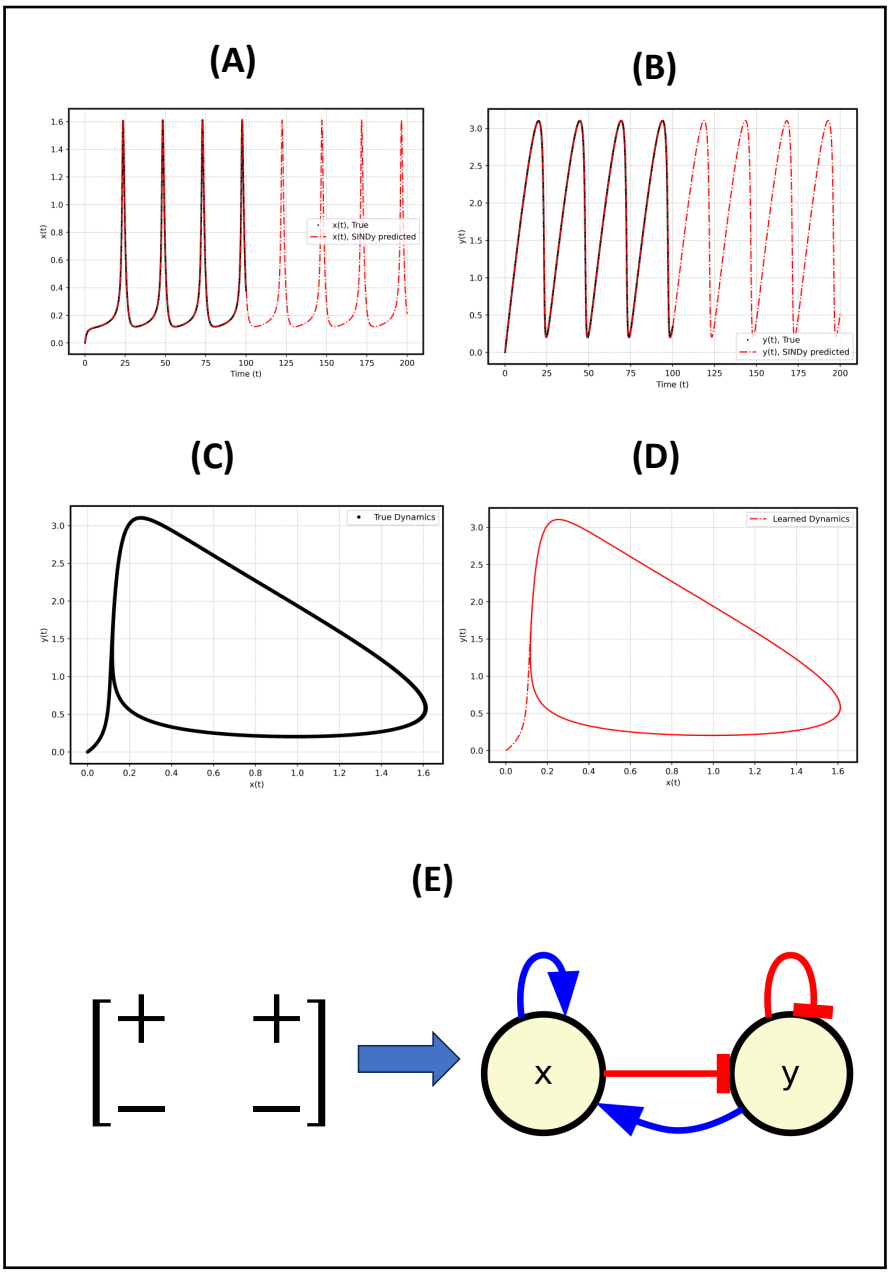

Figure 5.8. *Results from the Sparse Identification of Nonlinear Dynamics method applied to the Glycolytic oscillation (Selkov) model. (A) and (B) show the time series of the FitzHugh–Nagumo model for variables x and y, respectively. The black curves represent the true dynamics, while the red curves correspond to the SINDy-simulated trajectories. (C) and (D) display the corresponding phase diagram in the x–y plane for the original and SINDy-learned model, respectively. The similarity between the two diagrams indicates that the learned model effectively captures the underlying system dynamics. (E) depicts the sign structure of the Jacobian matrix evaluated at the steady state $(x^*, y^*) = (0.30, 2.22)$, along with the resulting interaction network. Blue arrows indicate positive regulation (activation), and red arrows indicate negative influences (inhibition).*

5.4 Discussion

The Sparse Identification of Nonlinear Dynamics (SINDy) framework was evaluated on five examples, namely, FitzHugh–Nagumo (FHN), Goodwin, Mass-Action cell-cycle, the Oregonator, and Glycolytic (Selkov) Oscillation model, to assess its efficacy in discovering interpretable governing equations using simulated time-series data. The entire procedure involved simulating each system, extracting time derivatives from trajectories, creating candidate function libraries, executing sparse regression using STLSQ, and validating the SINDY-learned models using steady-state, Jacobian, and network analysis. The codes and notebooks for all the analysis done in this chapter are available at `https://github.com/Alka-CBhub/Chapter-5`.

All models, excluding the Goodwin oscillator, utilized two types of libraries: (1) a prior-informed library including just terms derived from the original system, and (2) a polynomial library comprising all monomials up to degree 3.

In most cases, **both libraries produced consistent results: the proper terms were recognized, extra terms (if any) had tiny coefficients, and the dynamics of the SINDy-inferred models closely resembled the original trajectories and phase diagram. The steady states estimated from the discovered equations were likewise consistent with the analytical fixed points, and the sign pattern of the Jacobians typically reflected the actual local regulatory interactions**.

However, **the Oregonator model showed an important limitation**. While both inferred models accurately captured the system's dynamics and stable states, the expanded polynomial library incorporated several terms not present in the original equations. These **extra terms, although did not affect the time evolution, appeared as non-zero entries in the Jacobian. Consequently, it affected the deduced interaction network**. This finding suggests that **successful reconstruction of dynamic behavior does not always imply correct model discovery, especially when network edges are chosen only by numerical Jacobian structure and no significance threshold is used on their entries**.

For the **Goodwin model**, we used a single candidate library consisting only of terms explicitly found in the original ODEs. This model **incorporates a Hill-type term,** $\frac{1}{z^8+1}$. The standard SINDy framework symbolically depicts each candidate term, treating its rational representation as an a single indivisible basis function. Thus, the discovered equations preserved this term in its original form, possibly multiplied by a relevant coefficient derived via regression. This shows a fundamental limitation of the standard SINDy approach. The **candidate terms are treated as symbolic black-box terms, with SINDy simply assigning a scalar coefficient to them**. This limits interpretability and generalization, particularly when the actual dynamics incorporate nonlinearities that may be analytically factored or reduced.

In the next chapter, we examine this limitation in more detail and learn an adaptable variation of SINDy— Implicit SINDy—which can more naturally identify and express models incorporating rational functions with improved identifiability.

Overall, our analysis shows that SINDy, when combined with effective library design, may

recover meaningful dynamics across systems of varying complexity. For real-world applications, **it is important to consider not only trajectory accuracy but also the interpretability of inferred structures, particularly in network analysis**.

Chapter 6

Implicit SINDy: Handling Rational Nonlinearities

Sparse Identification of Non-linear Dynamics (SINDy) has become a fundamental approach for identifying governing equations directly from data. When the underlying dynamics can be expressed as low-degree polynomials, a simple library consisting of monomials is generally sufficient to derive models that are both precise and understandable. Unfortunately, several biochemical processes are either mediated by enzymes or include cooperativity. The terms corresponding to these kinetics involve rational nonlinearity and cannot be expressed sparsely in a polynomial basis. Mangan et al. [8], introduced the notion to **generalize the SINDy library to account for rational nonlinear dynamics**. It involves creating the library that consists of both state variables and their derivatives and allow the algorithm to identify the coefficients of an implicit algebraic relation that naturally accommodates rational functions.

Reasons for the Failure of Normal SINDy

When the system involves **enzyme-mediated or cooperative binding phenomena**, the library must have the appropriate rational functions to accurately represent these phenomena. To understand the failure of traditional SINDy, consider the following single-species Michaelis-Menten dynamics:

$$\dot{x} = 0.6 - \frac{1.5x}{0.3 + x}.$$

If some one intends to implement standard SINDy in this scenario, several questions must be addressed:

(1) Can the **rational term** $\frac{x}{K+x}$, be written as **a finite linear combination of various rational functions**? The answer is no, since generic rational nonlinearities are not sparse linear combinations of a finite number of rational functions [8].

(2) In certain cases, a rational function may be represented as an infinite power series and

can be approximated by a finite linear combination of polynomials, subject to a specific degree of inaccuracy. Using this type of approximation in the context of standard SINDy will lead to two additional inquiries:

(2a) What should be the highest degree of the polynomial?

(2b) What should be the error tolerance?

Assume that a polynomial of degree d is used to represent a rational term in n variables in a rare instance. Consequently, the SINDy library should contain the monomial terms. The total number of all these monomials in n variables is determined by

$$N_m = \binom{n+d}{d}.$$

For example:

If $n = 5$ variables and $d = 4$, then:

$$N_m = \binom{5+4}{4} = \binom{9}{4} = 126.$$

As a result, the SINDy library will need to contain N_m terms, and if the degree d is moderate to high, it can be very large for a given number of state variables n.

Due to the possibility of several rational terms and state variables, the process will be computationally costly. Additionally, it may result in a model that does not accurately reflect the dynamics.

Mangan *et al.* [8] proposed the concept of generalizing the SINDy library to accommodate rational nonlinear dynamics by including both state and derivative terms into an implicit feature matrix, which will be explored in the subsequent section.

6.1 Implicit SINDy and SINDy-PI

Suppose that each of the state variables x_k in the equation

$$\dot{x}_k = f_k\big(x_1, x_2, \ldots, x_n\big), \qquad k = 1, 2, \ldots, n. \tag{6.1.1}$$

(6.1.1) is governed by some nonlinear rational function. The above equation then can be expressed as:

$$\dot{x}_k = f_k(\mathbf{x}) \tag{6.1.2}$$

$$= \frac{f_k^N(\mathbf{x})}{f_k^D(\mathbf{x})}, \tag{6.1.3}$$

where $f_k^N(\mathbf{x})$ and $f_k^D(\mathbf{x})$ are polynomials in the state variables $x_1, x_2, ..., x_n$ obtained after simplification.

To model the dynamics represented by (6.1.3), it is rewritten in the **implicit form** by multiplying both sides of the equation by the denominator polynomial:

$$f_k^N(\mathbf{x}) = f_k^D(\mathbf{x})\dot{x}_k, \text{ or} \tag{6.1.4}$$

$$f_k^N(\mathbf{x}) - f_k^D(\mathbf{x})\dot{x}_k = 0 \tag{6.1.5}$$

Thus, the original dynamics (6.1.1) has the form

$$F_k(\mathbf{x}, \dot{x_k}) = 0. \tag{6.1.6}$$

For example: The equation

$$\dot{x} = 0.6 - \frac{1.5x}{0.3 + x}$$

may be expressed in the rational form (6.1.3) as:

$$\dot{x} = \frac{0.18 - 0.3x}{0.3 + x}$$

which further can be written in the implicit form (6.1.6) as:

$$0.18 - 0.3x - 0.3\dot{x} - x\dot{x} = 0$$

The implicit formulation (6.1.6) in $\dot{x}_k \; \forall \; k$, has motivated to **generalize the SINDy library to include both purely state-dependent terms and the terms involving the product of $\dot{x}_k$ with functions of $\mathbf{x}$** [8]:

$$\mathbf{\Theta}_k(\mathbf{x}, \dot{x}_k(t)) = \begin{bmatrix} \mathbf{\Theta_k^N}(\mathbf{x}) & \dot{x}_k(t)\mathbf{\Theta_k^D}(\mathbf{x}) \end{bmatrix} \quad \forall \quad k = 1, 2, ..., n \tag{6.1.7}$$

where, the first term, $\mathbf{\Theta_k^N}(\mathbf{x})$, is the library of numerator monomials in $\mathbf{x}$ and the second term, $\dot{x}_k(t)\mathbf{\Theta_k^D}(\mathbf{x})$, is obtained by multiplying each term in the library of denominator polynomials $\mathbf{\Theta_k^D}(\mathbf{x})$ with the vector $\dot{x}_k(t)$[8].

Remark:

- **Complexity (the number of terms)** of the model **written in the implicit formulation** (6.1.5), will be : Number of monomials in the numerator $\times$ Number of monomials in the denominator.
- In most cases, **the same polynomial degree** can be used for both the numerator and denominator libraries, so that $\boldsymbol{\Theta}_k^{\mathbf{N}}(\mathbf{x}) = \boldsymbol{\Theta}_k^{\mathbf{D}}(\mathbf{x})$. Thus, in this case, the augmented library (6.1.7) is only twice the size of the original polynomial library in $\boldsymbol{\Theta}_k^{\mathbf{N}}(\mathbf{x})$ [8].

Hence, the equation (6.1.6) in terms of the augmented library can be written as:

$$\boldsymbol{\Theta_k}(\mathbf{x}, \dot{x}_k(t))\boldsymbol{\xi}_k = 0, \quad k = 1, 2, ..n \tag{6.1.8}$$

where ξ_k is the coefficient vector for the kth implicit equation. The **Eq.**(6.1.8) terms of the data matrix take the form

$$\boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k)\boldsymbol{\xi}_k = 0, \quad \boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k) = \left[\boldsymbol{\Theta}_k^{\mathbf{N}}(\mathbf{X}) \quad \text{diag}(\dot{x}_k(t))\boldsymbol{\Theta}_k^{\mathbf{D}}(\mathbf{X})\right] \tag{6.1.9}$$

Note that finding **the sparsest non-zero vector $\boldsymbol{\xi}_k$ (as a trivial solution is always there, which will capture no dynamics)** that satisfies (6.1.9) is equivalent to finding the null space of $\boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k)$. This led to solving the **sparse null space problem**:

$$\min \|\boldsymbol{\xi}_k\|_0 \quad \text{subject to} \quad \boldsymbol{\xi}_k \in \mathcal{N} \setminus \{0\} \tag{6.1.10}$$

where:

- $\mathcal{N} = \text{Null Space}(\boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k)$
- $\|\boldsymbol{\xi}_k\|_0$ counts the number of non-zero entries in $\boldsymbol{\xi}_k$, (combinatorial, discontinuous, and non-convex function).

Remark:

- The constraint set, $\mathcal{N} = \text{Null Space}(\boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k)$ is a linear subspace, but the intersection with the objective function $\|\boldsymbol{\xi}_k\|_0$ introduces multiple local minima.
- If p is the number of columns in the implicit feature matrix, then the **(theoretical) dimension** (independent directions) of $\mathcal{N} = p - m$ where $m >> p$.

The problem (6.1.10) constitutes a non-convex optimization problem that is computationally NP-hard: **to determine the best subset of non-zero coefficients, one will have to evaluate**

all the possible combinatorial subsets, which grows exponentially with the number of candidates in the library. To address this sparse null space problem, implicit SINDy employs the Alternating Directions Method (ADM) developed by Qu et al. [62]. This approach identifies the sparsest vector within a subspace of the null space by alternating between two tasks: (1) enforcing sparsity and (2) projecting onto the null space.

6.1.1 Limitations of implicit formulation of SINDy

The implicit formulation of SINDy is solved using the Alternating Directions Method (ADM), which requires null space calculations. If $\boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k)$ has full rank (columns in the feature matrix are linearly independent), the null space $\mathcal{N}$ is properly defined and has the smallest dimension. However, when the data matrix $\mathbf{X}$ is noisy, it perturbs each column of $\boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k)$. As a result, this decreases the **numerical rank** of $\boldsymbol{\Theta_k}(\mathbf{X}, \dot{\mathbf{X}}_k)$, and so increases the dimension of the null space $\mathcal{N}$. This increase in size may not represent the true direction. Due to a richer null space, ADM might get stucked in suboptimal local minima. As a result, the sparse solutions become unreliable or dense, preventing the model from generalizing.

6.1.2 Robust formulation of the Implicit SINDy: SINDy-PI

To address the limitations of the implicit formulation, Kaheman et al. [43] developed SINDy-PI, which reformulates the implicit SINDy problem into a series of explicit sparse regression problems that may be solved in parallel. The key idea of SINDy-PI is that, if a single term $\theta_j(\cdot)$ of the dynamics (6.1.6) is known, then the original implicit equation (6.1.8) can be rewritten by isolating θ_j on the left-hand side and expressing it as a linear combination of all other terms:

$$\theta_j(\mathbf{X}, \dot{\mathbf{X}}_k) \;=\; \sum_{i \neq j} \theta_i(\mathbf{X}, \dot{\mathbf{X}}_k)\, \xi_i^{(j)} \tag{6.1.11}$$

Here, $\theta_j(\mathbf{X}, \dot{\mathbf{X}}_k)$ is the j^{th} column of $\boldsymbol{\Theta}_k(\mathbf{X}, \dot{\mathbf{X}}_k)$ and $\xi^{(j)} \in \mathbb{R}^{p-1}$ for $j = 1, 2, ..., p$. The equation (6.1.11) now is no longer an implicit equation in the derivative, and thus, it can be solved for a sparse coefficient vector $\boldsymbol{\xi}_i^{(j)}$ by minimizing:

$$\underbrace{\left\| \theta_j(\mathbf{X}, \dot{\mathbf{X}}_k) \;-\; \sum_{i \neq j} \theta_i(\mathbf{X}, \dot{\mathbf{X}}_k)\, \xi_i^{(j)} \right\|_2^2}_{\text{Residual Loss}} \;+\; \underbrace{\lambda_j \left\| \boldsymbol{\xi}^{(j)} \right\|_0}_{\text{Sparsity Penalty}} \tag{6.1.12}$$

where λ_j is the penalty for the j^{th} regression problem. This explicit regression problem is solved for each candidate $\theta_j(\cdot)$, $j = 1, 2, ..., p$ in parallel.

In this way SINDy-PI avoids explicit null-space computations, hence enhancing numerical stability and robustness to noise than implicit SINDy [43].

Reformulating SINDy-PI as a Global Constrained Optimization Problem

Each individual sparse regression problem in SINDy-PI (6.1.11) is formulated by selecting one candidate feature θ_j as the left-hand side and expressing it as a sparse linear combination of the remaining $p-1$ terms in the library, hence yielding p regression problems. Kaheman et al. [43] formulated these separate sparse regressions as a single constrained optimization problem by writing the regression equations as a single constrained system of equations:

$$\boldsymbol{\Theta}\big(\mathbf{X}, \dot{\mathbf{X}}_k\big) \;\approx\; \boldsymbol{\Theta}\big(\mathbf{X}, \dot{\mathbf{X}}_k\big)\,\boldsymbol{\Xi} \quad \text{subject to} \quad \Xi_{jj} = 0. \tag{6.1.13}$$

where:

(1) $\boldsymbol{\Theta} = [\,\theta_1 \mid \theta_2 \mid \cdots \mid \theta_p\,] \in \mathbb{R}^{m\times p}$, θ_j is a candidate basis function evaluated over m time points.

(2)
$$\boldsymbol{\Xi} = \left[\,\boldsymbol{\xi}^{(1)} \;\middle|\; \boldsymbol{\xi}^{(2)} \;\middle|\; \cdots \;\middle|\; \boldsymbol{\xi}^{(p)}\,\right] \in \mathbb{R}^{p\times p}, \quad \text{with} \quad \boldsymbol{\xi}^{(j)} \in \mathbb{R}^{p},$$
and $\boldsymbol{\xi}_i^{(j)} = \Xi_{jj} = 0, \quad j = 1, 2, \ldots, p$. These diagonal constraint ensures that no column uses itself for reconstruction.

(3)
$$\boldsymbol{\Theta}\,\Xi \;=\; \left[\,\boldsymbol{\Theta}\,\xi^{(1)} \;\middle|\; \boldsymbol{\Theta}\,\xi^{(2)} \;\middle|\; \cdots \;\middle|\; \boldsymbol{\Theta}\,\xi^{(p)}\right] \;=\; \left[\,\hat{\theta}_1 \;\middle|\; \hat{\theta}_2 \;\middle|\; \cdots \;\middle|\; \hat{\theta}_p\right]$$
where $\hat{\theta}_j = \sum_{i\neq j} \theta_i\,\xi_i^{(j)}$.

so that, the optimization problem becomes:

$$\min_{\boldsymbol{\Xi}} \underbrace{\left\|\Theta(\mathbf{X}, \dot{\mathbf{X}}_k) - \Theta(\mathbf{X}, \dot{\mathbf{X}}_k)\,\boldsymbol{\Xi}\right\|_2^2}_{\text{Residual Loss}} + \underbrace{\beta\,\|\boldsymbol{\Xi}\|_0}_{\text{Sparsity Penalty}} \qquad \text{s.t.} \quad \operatorname{diag}(\boldsymbol{\Xi}) = 0, \tag{6.1.14}$$

where $\beta > 0$ is a sparsity-promoting regularization parameter. This formulation enables all candidate models to be evaluated in a unified optimization framework and solved efficiently in parallel using sequential thresholded least squares [43].

6.2 Tools for Implementation

In this thesis, the SINDy-PI framework is implemented in Python utilizing the PySINDy Python package [60, 61]. This package has specific modules to identify models with implicit dynamics. The implementation involves two key components:

(1) **Feature Library Construction:**

Designing the library of candidate function is the crucial part in while performing implicit SINDy. The library is constructed using the `SINDyPILibrary` class from the pysindy.feature_library.sindy_pi_library module. Unlike standard SINDy, this library separately accepts two sets of user-defined functions:

(a) One involving the function of state variables.

(b) Another for defining the derivative of the state variable.

After defining the functions, the library constructed from `SINDyPILibrary` contains all the terms from (a) and (b) and their tensor product.

Things to consider when constructing the library:

- Separate feature libraries are created for each equation $\dot{x}_k$, as the implicit formulation requires that $\dot{x}_k$ appears only in the k^{th} equation.
- The library must include both:
  - Purely state-dependent terms: $\theta(\mathbf{x})$
  - Implicit terms involving derivatives: $\dot{x}_k \cdot \theta(\mathbf{x})$

(2) **Optimization Framework:**
Once the library has been constructed, the next step is to perform sparse regression. This is done by using the optimizer `SINDyPI`, implemented in the module `pysindy.optimizers.sindy_pi`. The **theoretical formulation** (6.1.14) involves ℓ_0 norm minimization, which is non-convex and computationally hard. PySINDy uses convex relaxations for practical implementation. Specifically, for each $j = 1, \ldots, p$, the optimizer solves:

$$\min_{\boldsymbol{\xi}^{(j)}} \frac{1}{2} \left\| \theta_j - \Theta \boldsymbol{\xi}^{(j)} \right\|_2^2 + \lambda R(\boldsymbol{\xi}^{(j)}) \quad \text{subject to} \quad \boldsymbol{\xi}_j^{(j)} = 0$$

where $R(\cdot)$ is a sparsity-inducing regularization term, such as ℓ_1, ℓ_2, or their weighted variants. Convex optimizations are handled internally using solvers available in `CVXPY`, a convex optimization library.

Remark: In order to control the strength of the regularization PySINDy allows:

- A global threshold parameter `threshold`, applied uniformly across all coefficients, or
- A matrix `thresholds[i, j]`, specifying the threshold for the j^{th} term in the i^{th} equation providing flexibility in handling variables with different scales or noise sensitivities.

Algorithm 1: Parallel Implicit Sparse Identification (SINDy-PI)

Input: Implicit Feature matrix $\Theta \in \mathbb{R}^{m \times p}$;
Sparsity parameter $\lambda > 0$;
Tolerance `tol`, maximum iterations `max_iter`;
Output: Sparse coefficient matrix $\boldsymbol{\Xi} \in \mathbb{R}^{p \times p}$

Initialize $\boldsymbol{\Xi} \leftarrow \mathbf{0}_{p \times p}$;
for $j = 1$ **to** p **do**
  Extract target feature: $\theta^{(j)} \leftarrow \Theta[:, j]$;
  Define optimization variable $\boldsymbol{\xi}^{(j)} \in \mathbb{R}^p$;
  Solve:

$$\min_{\boldsymbol{\xi}^{(j)}} \left\| \theta^{(j)} - \Theta \boldsymbol{\xi}^{(j)} \right\|_2^2 + \lambda \left\| \boldsymbol{\xi}^{(j)} \right\|_1 \quad \text{s.t.} \quad \xi_j^{(j)} = 0$$

  Set $\boldsymbol{\Xi}[:, j] \leftarrow \boldsymbol{\xi}^{(j)}$;
return $\boldsymbol{\Xi}$

6.2.1 Model Selection in SINDy-PI

For a given choice of hyperparameters, the SINDy-PI algorithm returns a collection of candidate models: one for each target term θ_j by solving a series of sparse regression problems in parallel [43]. The goal of model selection is to determine which among these candidates best balances *accuracy* and *parsimony*, that is, to identify implicit models that adequately explain the observed data while remaining as simple as possible.

Let $Y = \boldsymbol{\Theta}\left(\mathbf{X}, \dot{\mathbf{X}}_k\right) \in \mathbb{R}^{m \times p}$ and $\hat{Y} = \boldsymbol{\Theta}\left(\mathbf{X}, \dot{\mathbf{X}}_k\right)\boldsymbol{\Xi} \in \mathbb{R}^{m \times p}$ where Y and $\hat{Y}$ are defined in (6.1.14). Each column in $\hat{Y}$ corresponds to the predicted model obtained by solving an individual regression problem associated with the corresponding implicit feature.

Further, define:

$$\mathbf{y}^{(j)} = Y[:, j], \quad \hat{\mathbf{y}}^{(j)} = \hat{Y}[:, j], \quad \text{and} \quad \mathbf{r}^{(j)} = \mathbf{y}^{(j)} - \hat{\mathbf{y}}^{(j)}, \quad j = 1, 2, \ldots, p$$

where $\mathbf{r}^{(j)}$ is the residual vector for the j^{th} model (column).

To evaluate the quality of each inferred model, we can compute several performance metrics for every model ($j = 1, 2, \ldots, p$:.

(a) **Coefficient of Determination (R^2):**

$$R_j^2 = 1 - \frac{\|\mathbf{r}^{(j)}\|_2^2}{\|\mathbf{y}^{(j)} - \bar{y}^{(j)}\|_2^2}, \quad \text{where} \quad \bar{y}^{(j)} = \frac{1}{m} \sum_{i=1}^{m} y_i^{(j)}.$$

A higher $R_j^2 \in [0, 1]$ indicates better model fit.

(b) **Relative Error:**

$$\text{Rel}_j = \frac{\|\mathbf{r}^{(j)}\|_2}{\|\mathbf{y}^{(j)}\|_2}.$$

This normalized residual helps to compare models regardless of the scale of $\mathbf{y}^{(j)}$.

(c) **Sparsity (Number of Active Terms):**

$$k_j = \|\boldsymbol{\xi}^{(j)}\|_0,$$

where $\boldsymbol{\xi}^{(j)}$ is the coefficient vector for the j^{th} equation. A lower k_j implies a more parsimonious model.

(d) **Akaike Information Criterion (AIC):**

$$\text{AIC}_j = 2k_j + m \ln\left(\frac{\|\mathbf{r}^{(j)}\|_2^2}{m}\right).$$

AIC penalizes the number of active terms to discourage overfitting.

(e) **Bayesian Information Criterion (BIC):**

$$\text{BIC}_j = k_j \ln(m) + m \ln\left(\frac{\|\mathbf{r}^{(j)}\|_2^2}{m}\right).$$

BIC imposes a stronger penalty for model complexity, especially when m is large.

These metrics can be used separately or combined to evaluate model accuracy, sparsity, and complexity. The final model is often chosen based on high R_j^2, low relative error, and lowest AIC and BIC values.

Multiple candidate functions $\theta_j(\cdot)$ may generate accurate and sparse models. These models should be cross-referenced to verify the consistency of selected terms [43].

Remarks:

- When a specific column of Y is constant (e.g., all ones, which may arise when a bias term is included independently in the library), the coefficient of determination R^2 becomes undefined, since the denominator $\|\mathbf{y}^{(j)} - \bar{y}^{(j)}\|_2^2$ evaluates to zero. In such instances, libraries like `scikit-learn` yield $R^2 = 0$, which may wrongly suggest that the associated model is an inadequate match. Consequently, for constant columns, it is advisable to utilize additional metrics.
- When modeling the dynamics of biological systems, it is often required that all expressions in the denominator of rational terms should remain positive to maintain physical and biological validity. Consequently, models exhibiting singularities or including negative-valued denominators are excluded due to their potential to produce biologically implausible or undetermined outcomes in simulation.

6.3 Application

Methodology Overview

This thesis employs the SINDy-PI (Parallel Implicit Sparse Identification of Nonlinear Dynamics) framework to discover rational dynamical models from time-series data. The overall methodology follows a systematic pipeline, as illustrated in Figure 6.1. The workflow integrates simulation, sparse model identification, and interpretability filtering to produce biologically meaningful rational ODEs.

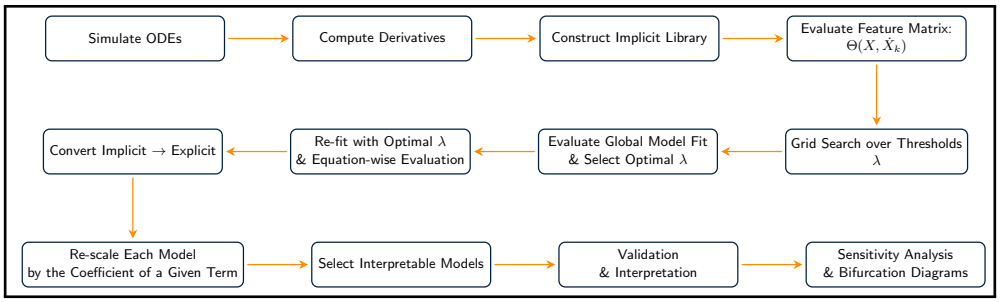

Figure 6.1. *Schematic Overview of the Implicit SINDy (SINDy-PI) workflow*

The key steps in the methodology are outlined below:

1. **Simulate the ODEs:** The given system of ODEs are numerically integrated to generate synthetic time-series data $\mathbf{x}(t)$ for all state variables, based on predefined parameters and initial conditions.

2. **Compute Time Derivatives:** Derivatives $\dot{\mathbf{x}}(t)$ are estimated from the simulated trajectories using finite difference methods.

3. **Construct the Implicit Library:** An implicit feature library $\Theta(x, \dot{x})$ is constructed. This includes both state-dependent monomials and derivative-coupled terms, such as $\dot{x} \cdot \theta(x)$, to enable the discovery of rational expressions.

4. **Evaluate the Feature Matrix:** The constructed library is evaluated on the dataset to obtain matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}})$, which serves as the data matrix for the model discovery process.

5. **Perform Threshold Grid Search:** A grid of sparsity thresholds λ is explored. For each value, SINDy-PI solves a constrained sparse regression problem to identify a set of implicit candidate models 6.3.

6. **Evaluate Global Model Fit and Select λ^*:** Each model is evaluated using global performance metrics such as mean squared error (MSE), relative error, and information criteria (AIC and BIC). The optimal threshold λ^* is selected based on a balance between accuracy and sparsity.

7. **Re-fit and Evaluate Equation-wise Models:** Using the selected thresholdλ^*, SINDy-PI is re-applied to generate a set of implicit equations. Each equation is individually assessed using R^2, relative error, AIC, BIC, and the number of active terms.

8. **Convert Implicit to Explicit Form:** All the implicit equations are algebraically rearranged to solve for the derivative, yielding explicit rational expressions suitable for interpretation.

9. **Re-scale Equations:** Each model is then rescaled by dividing through the coefficient of a selected term from the denominator to normalize the expression and facilitate comparison. If any given term is not present in the denominator, then the expression will get rescaled by the coefficient of the leading term.

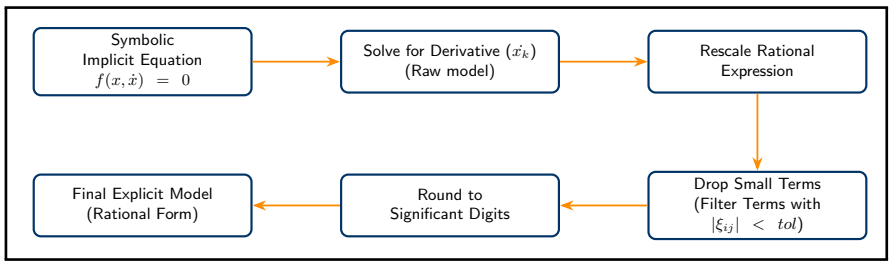

Figure 6.2. *Schematic overview of the implicit to explicit conversion and rescaling.*

10. **Select Interpretable Models:** Models are filtered based on interpretability criteria like absence of singularities, positive-valued denominators, and biological plausibility.

11. **Validation and Interpretation:** The final model is validated by comparing it with the ground-truth system. Additional analysis includes computation of steady states, Jacobian matrices, and network visualization to explain the system's regulatory behavior.

This pipeline enables robust discovery of accurate, interpretable, and biologically relevant rational ODE models, even in the presence of complex nonlinearities.

Steps in Performing Global Threshold Tuning

The **SINDy-PI implementation generates a collection of implicit equations** by solving the optimization problem:

$$\min_{\boldsymbol{\Xi}} \underbrace{\left\|\Theta(\mathbf{X}, \dot{\mathbf{X}}_k) - \Theta(\mathbf{X}, \dot{\mathbf{X}}_k)\,\boldsymbol{\Xi}\right\|_2^2}_{\text{Residual Loss}} + \underbrace{\lambda\, R(\boldsymbol{\Xi})}_{\text{Sparsity Penalty}} \quad \text{s.t.} \quad \operatorname{diag}(\boldsymbol{\Xi}) = 0, \tag{6.3.1}$$

where $\lambda > 0$ is a sparsity-promoting regularization parameter that must be carefully tuned to obtain a model that best balances accuracy and sparsity [43].

We let $Y = \boldsymbol{\Theta}\big(\mathbf{X}, \dot{\mathbf{X}}_k\big) \in \mathbb{R}^{m\times p}$ (implicit feature matrix) and $\hat{Y} = \boldsymbol{\Theta}\big(\mathbf{X}, \dot{\mathbf{X}}_k\big)\boldsymbol{\Xi} \in \mathbb{R}^{m\times p}$ where m is the number of time points and p is the number of candidates in implicit feature library. In this scenario, $k = 1$. Each column of $\hat{Y}$ represents the prediction for the corresponding column of Y. We conduct a grid search by choosing a set of 20 threshold parameters logarithmically spaced between 10^{-6} and 10^{-1} to find the optimal value of threshold, λ. In this process, we fit a SINDy-PI for each value of the threshold λ, resulting in the coefficient matrix $\boldsymbol{\Xi} \in \mathbb{R}^{p\times p}$, and assess the overall efficacy of the models (coefficient matrix $\boldsymbol{\Xi}$) by using the following performance metrics:

- **Mean Squared Error (MSE)** between the predicted and actual feature matrices:

$$\text{MSE} = \frac{1}{n \cdot p} \sum_{i=1}^{n} \sum_{j=1}^{p} (Y_{ij} - \hat{Y}_{ij})^2$$

- **Relative Error**, calculated as the normalized Frobenius norm of the residual matrix:

$$\text{Relative Error} = \frac{\|Y - \hat{Y}\|_F}{\|Y\|_F},$$

- **Akaike Information Criterion (AIC)** and **Bayesian Information Criterion (BIC)**, which penalize model complexity based on the number of active terms k and the total residual sum of squares:

$$\text{AIC} = 2k + m \cdot \ln\left(\frac{\text{RSS}}{m}\right), \quad \text{BIC} = k \cdot \ln(m) + m \cdot \ln\left(\frac{\text{RSS}}{n}\right)$$

where RSS $= \|Y - \hat{Y}\|_F^2, \quad k = \|\boldsymbol{\Xi}\|_0$.

We computed all the metrics and then select the optimal threshold accordingly.

Limitations of Global Threshold Tuning: Given a value of a threshold, SINDy-PI algorithms generate multiple models, one for each candidate term in the implicit library. In order to choose the optimal value of the threshold, we used a uniform global sparsity criterion, in which a uniform threshold is typically imposed for all equations. Though it simplifies the selection process, it brings numerous significant constraints:

1. We can not expect each implicit equation to show the same degree of noise or sensitivity. Therefore, a single threshold may not sufficiently balance sparsity and accuracy across all implicit models.

2. For each state variable, the implicit feature matrix usually combines **state monomials and derivative**-coupled terms. Using a uniform criterion might result in biased selection of models that may favour high-magnitude features and eliminate significant but smaller-magnitude terms.

3. Performance indicators such as AIC or global mean squared error include residuals from all equations, which might mask poor fits in certain models. As a result, the selected threshold may fail to provide equations that capture the true dynamics.

4. Another significant issue with global threshold tuning arises when we use extremely small threshold values, such as $\lambda \leq 10^{-10}$. Although such values may preserve the entire set of candidate terms and occasionally recover the proper model structure, they frequently cause numerical instability in the sparse regression solver. Specifically, as regularization decreases, the underlying linear system becomes ill-conditioned and susceptible to overfitting, causing the solver to fail or provide degenerate solutions with all-zero coefficients. **This problem occurred repeatedly during threshold sweep, with several models failing to converge or producing zero-valued coefficient vectors at extremely low thresholds**.

6.3.1 Example-1: Michaelis-Menten Kinetics

As a first example, we implemented the implicit-SINDy framework on a classical enzyme system regulated by Michaelis-Menten dynamics. This system describes the time evolution of a single substrate concentration x, which is consumed by an enzymatic process. The model takes the well-known form:

$$\boxed{\dot{x} = j_x - \frac{V_{\max} x}{K_m + x},} \tag{6.3.2}$$

where j_x denotes a constant input flux, $V_{\max}$ represents the maximum reaction rate, and K_m is the Michaelis constant which represents the substrate concentration at half-maximal velocity [8].

To test model recovery, we generate synthetic time-series data using the parameters defined in **Table 6.1** and settings in **Table 6.2**.

Table 6.1. *Parameter values for the MM kinetics Model.*

j	V_{max}	K_m
0.6	1.5	0.5

Table 6.2. *Settings for simulation of the MM Kinetics model.*

Setting	IC	t_0	t_f	Δt
Value	0.1	0	500	0.05

After substituting the parameter values, the model 6.3.2 becomes

$$\boxed{\dot{x} = -1.5\frac{x}{(x + 0.3)} + 0.6} \tag{6.3.3}$$

To make the further analysis more clear, we express the model 6.3.3 in its pure rational form :

$$\boxed{\dot{x} = \frac{0.18 - 0.9\,x}{x + 0.3}} \tag{6.3.4}$$

Writing the above equation in implicit form gives

$$\boxed{(0.18 - 0.9\,x) - \dot{x}(x + 0.3) = 0} \tag{6.3.5}$$

Note that, the **Eq.** (6.3.5) is spanned by, $[1,\ x\,] \cup \dot{x} \cdot [1,\ x]$. Therefore we choose the library for x:

$$\Theta(x, \dot{x}) = [1,\ x\,,\ \dot{x},\ x\dot{x}\,]$$

We then conduct a grid search by choosing a set of 20 threshold parameters logarithmically spaced between 10^{-8} and 10^{-1} to find the optimal value of threshold, λ (**Figure 6.3**). We

observe that each metric returns a nearly identical value for the optimal threshold shown in **Table 6.3**.

Equation-wise model evaluation for optimal threshold

We re-fit the SINDy-PI using the optimal value of the threshold $lambda$, which results in a set of 4 implicit equations. We assessed every equation using the same training set in terms of:

- R^2 score
- Relative Error
- AIC and BIC
- Number of active terms (k)

Table **Table 6.4** shows the value of these scores together with the LHS term and the model index. This table ranks every equation based on relative R^2 score along with other scores.

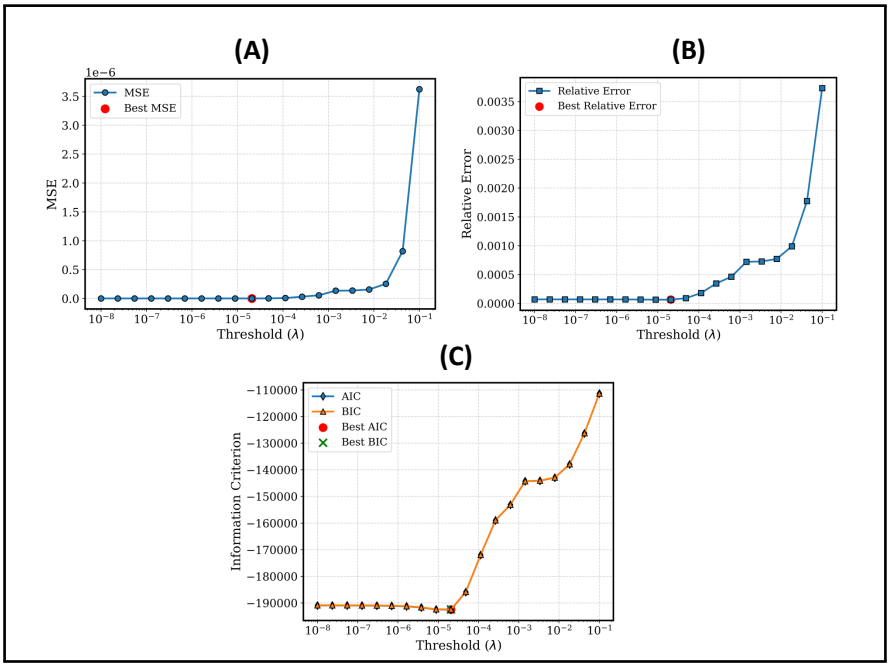

Figure 6.3. *Global diagnostic plots for determining the sparsity threshold for MM Kinetics model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated over a logarithmically spaced grid from* 10^{-8} *to* 10^{-1}*. The implicit feature matrix* $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{10001\times 4}$ *is approximated by* $\hat{Y} = \Theta\Xi$*, and model performance is assessed by residual norms and complexity-penalizing criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 6.3****.*

Table 6.3. *Best thresholds selected by different metrics for MM Kinetics Model.*

Metric	**Best Threshold**
Mean Squared Error	$2.069\,00\times10^{-5}$
Relative Error	$2.069\,00\times10^{-5}$
Akaike Information Criterion (AIC)	$2.069\,00\times10^{-5}$
Bayesian Information Criterion (BIC)	$2.069\,00\times10^{-5}$

All models show a good performance score on the training data. Since a biologically interpretable model must not have singularities or negative terms in the denominator, we reformulate all candidate models to explicit equations for further examination.

Table 6.4. *Equations sorted by lowest Relative Error for* $\lambda = 2.069 \times 10^{-5}$.

Eq	**Feature**	$\mathbf{R^2}$	**Relative Error**	**AIC**	**BIC**
1	x	9.999 80	$5.146\,30\times10^{-5}$	$-2.297\,08$	$-2.296\,86$
0	1	0.000 000	$5.235\,80\times10^{-5}$	$-1.971\,62$	$-1.971\,40$
2	$\dot{x}$	9.999 70	$5.499\,00\times10^{-3}$	$-2.106\,58$	$-2.106\,37$
3	$x\,\dot{x}$	9.979 99	$4.467\,60\times10^{-2}$	$-2.094\,66$	$-2.094\,44$

Candidate models for x:

$$\textbf{Model 0:}\quad \dot{x} = \frac{0.1811 - 0.9055\,x}{1.0\,x + 0.3013}, \quad \text{Extra Terms: 0} \tag{6.3.6}$$

$$\textbf{Model 1:}\quad \dot{x} = \frac{0.1955 - 0.9775\,x}{1.0\,x + 0.3360}, \quad \text{Extra Terms: 0} \tag{6.3.7}$$

$$\textbf{Model 2:}\quad \dot{x} = \frac{0.1912 - 0.9558\,x}{1.0\,x + 0.3258}, \quad \text{Extra Terms: 0} \tag{6.3.8}$$

$$\textbf{Model 3:}\quad \dot{x} = \frac{0.1465 - 0.7326\,x}{1.0\,x + 0.2178}, \quad \text{Extra Terms: 0} \tag{6.3.9}$$

We find that all the possible models described above are both interpretable and exhibit a good performance score. To know the variation of coefficients across all models, we compare the learned coefficients with the ground truth values for each term in the numerator and the denominator by listing the minimum and maximum values of the coefficients across all final candidate models. The comparisons are summarized in **Table 6.5** and **Table 6.6** respectively.

Table 6.5. *Comparison of coefficients for the numerator terms for $\dot{x}$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	x	-0.9	-0.9775	-0.7326
2	1	0.18	0.1465	0.1955

Table 6.6. *Comparison of coefficients for the denominator terms for $\dot{x}$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	x	1.0	1.0	1.0
2	1	0.3	0.2178	0.3360

Besides coefficient comparison, we simulated the derived equations and compared their dynamics with the original model. As seen in **Figure 6.4**, all recovered models accurately replicated the original time series of $x(t)$, with negligible deviation in both transient and steady-state behavior. All identified models converged to an identical steady-state value of $x = 0.2$, matches with the analytical steady state of the original model. Moreover, all produced a consistent interaction network that was again found to align corresponding to the original model shown in **Figure 6.4**).

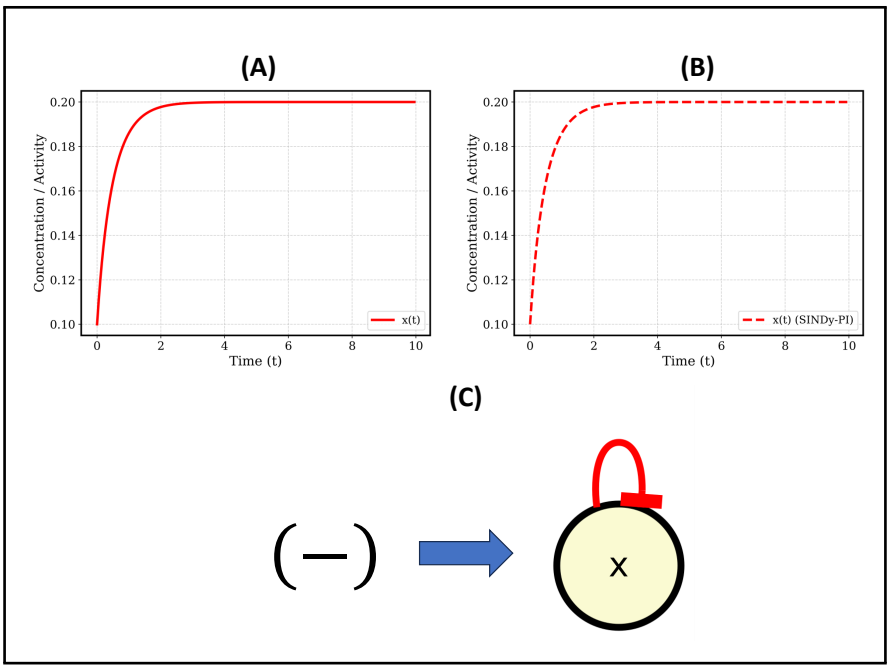

Figure 6.4. *Results from the Implicit SINDy (SINDy-PI) method applied to the MM Kinetics model. (A) and (B) show the original and SINDy simulated time series of the MM kinetics model for the variable x. (C) represents the sign of the Jacobian evaluated at the steady state $x^* = 0.20$, along with the resulting interaction network. The red arrows indicate negative influence (inhibition). All candidate models accurately replicate the original dynamics.*

6.3.2 Example 2: Microbial Growth Model

As a second example, we studied a two-dimensional system that monitors microbial population dynamics in a batch reactor. This system, regulated by Monod-type kinetics, has been used as a case study for model identifiability [44]. It describes the interaction between the microbial biomass (x_1) and a growth-limiting substrate (x_2) via the following set of nonlinear differential equations:

$$\dot{x}_1 = \frac{\mu x_1 x_2}{K_s + x_2} - K_d x_1, \tag{6.3.10}$$

$$\dot{x}_2 = -\frac{\mu x_1 x_2}{\gamma(K_s + x_2)}, \tag{6.3.11}$$

where μ is the maximum specific growth rate, K_s is the saturation constant, K_d is the death rate, and γ represents the yield coefficient, quantifying substrate depletion per unit of biomass produced [44].

Table 6.7. *Parameter values for the Microbial Growth Model.*

μ	K_s	K_d	γ
0.4	1.0	0.05	0.5

Table 6.8. *Settings for simulation of the Microbial Growth Model.*

Setting	IC	t_0	t_f	Δt
Value	$[0.1,\ 0.4]$	0	500	0.05

We simulated this system using the parameters and settings listed in **Table 6.7** and **Table 6.8**, respectively, to generate synthetic data. Also, substituting the parameter values to the system 6.3.11 gives:

$$\boxed{\dot{x}_1 = \frac{0.4\, x_1 x_2}{x_2 + 1.0} - 0.05\, x_1} \tag{6.3.12}$$

$$\boxed{\dot{x}_2 = -\frac{0.4\, x_1 x_2}{0.5\, x_2 + 0.5}} \tag{6.3.13}$$

which further can be written in pure rational form as:

$$\dot{x}_1 = \frac{0.35\, x_1\, x_2 - 0.05\, x_1}{x_2 + 1.0}, \tag{6.3.14}$$

$$\dot{x}_2 = -\frac{0.8\, x_1\, x_2}{x_2 + 1}. \tag{6.3.15}$$

To test implicit SINDy, we build a separate library for both the state variables x_1 and x_2 and use SINDy-PI to recover each equation independently. The selected candidate libraries for

x_1 and x_2 are:

$$\Theta_1(x, \dot{x}_1) = \left[x_1,\ x_2,\ x_1x_2,\ \dot{x}_1,\ \dot{x}_1x_1,\ \dot{x}_1x_2,\ \dot{x}_1x_1x_2 \right],$$

$$\Theta_2(x, \dot{x}_2) = \left[x_2,\ x_1x_2,\ \dot{x}_2,\ \dot{x}_2x_2,\ \dot{x}_2x_1x_2 \right],$$

we perform a grid search independently to identify the optimal value of λ for each equation. The optimal thresholds and equation wise model performance for each x_1 and x_2 are summarized in the subsequent tables.

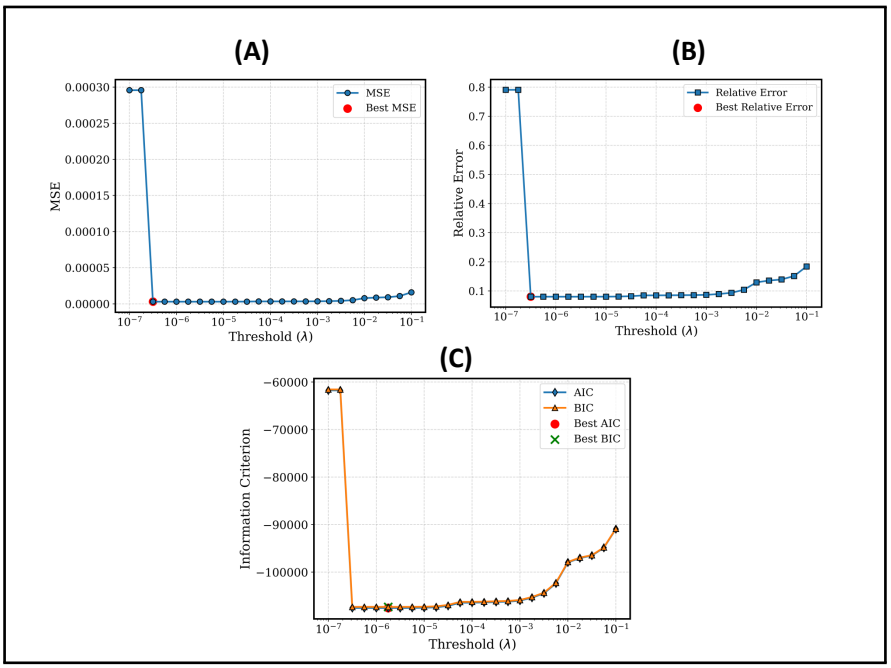

Figure 6.5. *Global diagnostic plots for determining the sparsity threshold for x_1 in the microbial growth model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated on 25 points logarithmically spaced between 10^{-7} to 10^{-1}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{10001\times 7}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and complexity-penalizing criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 6.9****.*

Table 6.9. *Best thresholds selected by different metrics for x_1 in the Microbial Growth Model.*

Metric	**Best Threshold**
Mean Squared Error	$3.162\,00\times10^{-7}$
Relative Error	$3.162\,00\times10^{-7}$
Akaike Information Criterion (AIC)	$1.778\,00\times10^{-6}$
Bayesian Information Criterion (BIC)	$1.778\,00\times10^{-6}$

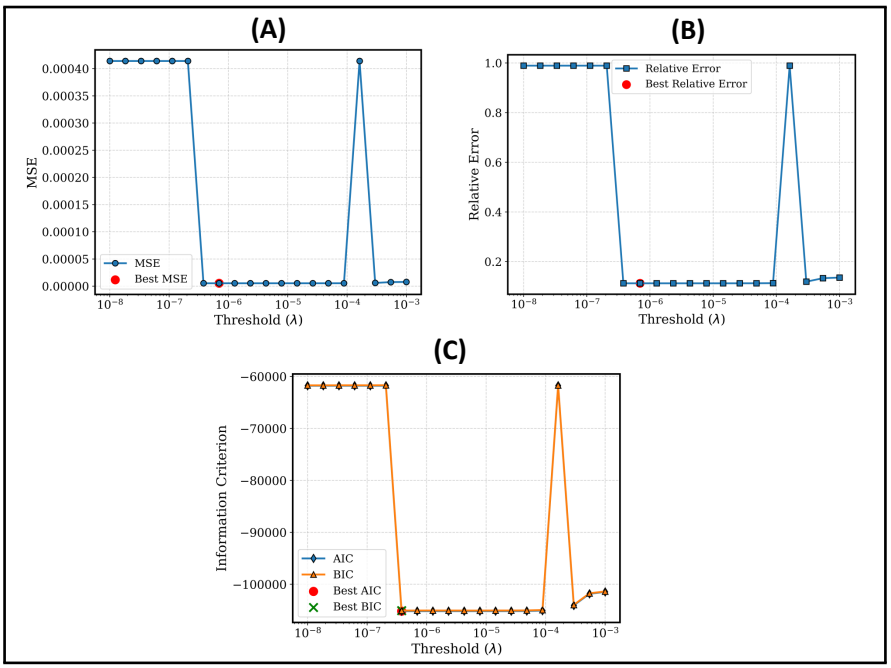

Figure 6.6. *Global diagnostic plots for determining the sparsity threshold for x_2 in the Microbial Growth Model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated on 20 points logarithmically spaced between 10^{-8} to 10^{-3}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_2) \in \mathbb{R}^{10001\times 5}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and complexity-penalizing criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 6.10****.*

Table 6.10. *Best thresholds selected by different metrics for x_2 in the Microbial Growth Model.*

Metric	**Best Threshold**
Mean Squared Error	$6.952\,00\times10^{-7}$
Relative Error	$6.952\,00\times10^{-7}$
Akaike Information Criterion (AIC)	$3.793\,00\times10^{-7}$
Bayesian Information Criterion (BIC)	$3.793\,00\times10^{-7}$

Table 6.11. *Equations for x_1 sorted by lowest Relative Error for $\lambda = 3.162 \times 10^{-7}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
0	x_1	1.000 00	$6.327\,20\times10^{-}$	−3.066	−3.066 06
1	x_1x_2	1.000 00	$5.513\,50\times10^{-5}$	−2.996 19	−2.995 90
3	$\dot{x}_1$	1.000 00	$2.334\,00\times10^{-}$	−3.033	−3.032 71
6	$\dot{x}_1x_2$	9.999 97	$1.647\,10\times10^{-3}$	−2.960 16	−2.959 87
5	$x_1\,\dot{x}_1\,x$	9.952 19	$6.880\,80\times10^{-}$	−2.643	−2.642 80
2	x_2	9.886 29	$1.013\,10\times10^{-1}$	−1.075 91	−1.075 55
4	$x_1\,\dot{x}_1$	9.598 70	$1.996\,10\times10^{-}$	−2.131	−2.130 67

Table 6.12. *Equations for x_2 sorted by lowest Relative Error for $\lambda = 6.952 \times 10^{-7}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
0	x_1x_2	1.000 00	$3.060\,20\times10^{-}$	−3.113	−3.113 75
2	$\dot{x}_2$	1.000 00	$4.366\,90\times10^{-5}$	−3.132 85	−3.132 64
4	$x_2\dot{x}_2$	1.000 00	$2.838\,50\times10^{-}$	−3.021	−3.021 40
3	$x_1x_2\dot{x}_2$	9.955 95	$6.537\,80\times10^{-2}$	−2.353 79	−2.353 57
1	x_2	9.855 28	$1.143\,00\times10^{-}$	−1.051	−1.051 53

Candidate models for x_1:

$$\text{Model 0:}\quad \dot{x}_1 = \frac{0.35\,x_1x_2 - 0.05001\,x_1}{1.0\,x_2 + 1.0} \tag{6.3.16}$$

$$\text{Model 1:}\quad \dot{x}_1 = \frac{0.3507\,x_1\,x_2 - 0.05008\,x_1}{1.0\,x_2 + 1.002} \tag{6.3.17}$$

$$\text{Model 3:}\quad \dot{x}_1 = \frac{0.3516\,x_1\,x_2 - 0.0502\,x_1}{1.0\,x_2 + 1.004} \tag{6.3.18}$$

$$\text{Model 6:}\quad \dot{x}_1 = \frac{0.3476\,x_1\,x_2 - 0.04971\,x_1}{1.0\,x_2 + 0.9937} \tag{6.3.19}$$

Candidate models for x_2:

$$\textbf{Model 0:}\quad \dot{x}_2 = \frac{-0.8005\,x_1x_2}{1.0\,x_2 + 1.0}$$

$$\textbf{Model 2:}\quad \dot{x}_2 = \frac{-0.8005\,x_1x_2}{1.0\,x_2 + 1.001}$$

$$\textbf{Model 4:}\quad \dot{x}_2 = \frac{-0.799\,x_1x_2}{1.0\,x_2 + 0.9988}$$

Table 6.13. *Comparison of coefficients for the numerator terms for $\dot{x}_1$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1\,x_2$	0.3500	0.3476	0.3516
2	x_1	-0.0500	-0.0502	-0.0497

Table 6.14. *Comparison of coefficients for the denominator terms for $\dot{x}_2$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	x_2	1.0000	1.0000	1.0000
2	1	1.0000	0.9937	1.0040

Table 6.15. *Comparison of coefficients for the numerator terms for $\dot{x}_2$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1\,x_2$	-0.8000	-0.8005	-0.7990

Table 6.16. *Comparison of coefficients for the denominator terms for $\dot{x}_2$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	x_2	1.0000	1.0000	1.0000
2	1	1.0000	0.9988	1.0010

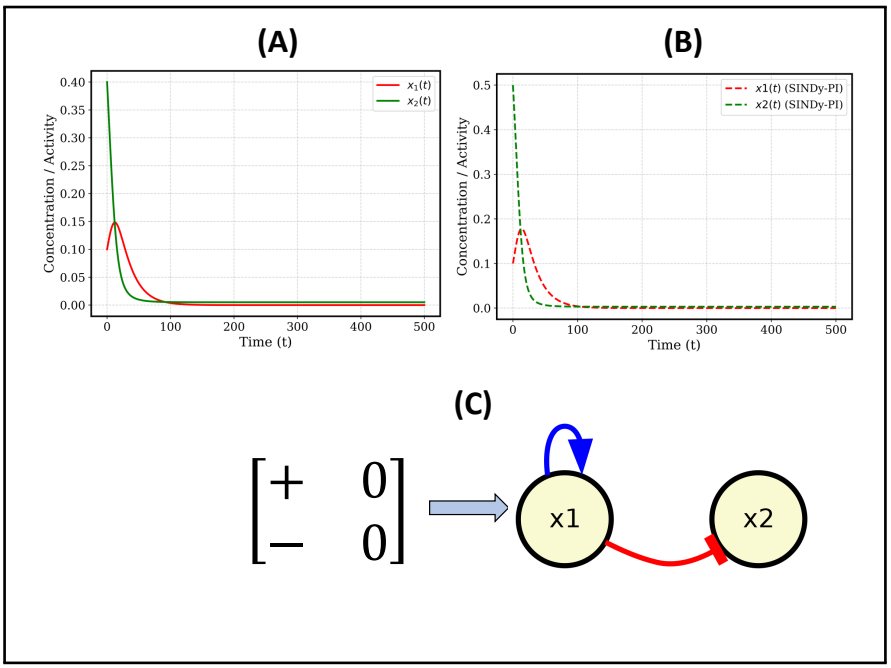

Figure 6.7. *Results from the Implicit SINDy (SINDy-PI) method applied to the Microbial Growth Model. (A) and (B) show the original and SINDy simulated time series of the Microbial Growth Model for the variable x_1 and x_2 respectively. (C) represents the sign structure of the Jacobian matrix evaluated at the steady state $(x_1^*, x_2^*) = (0, 1)$, along with the resulting interaction network. The red and blue arrows indicate negative and positive regulation respectively. Any combination of candidate models for x_1 and x_2 accurately replicate the original dynamics.*

Discussion

In this chapter, we applied the **Implicit SINDy (SINDy-PI)** framework to two examples involving rational nonlinearities: (1) the Michaelis-Menten (MM) enzyme kinetics model (one variable) [8] and (2) a Microbial Growth model (two variables) [44] to illustrate the methodology and implementation approach. Unlike the classical SINDy, which treats each candidate term as an isolated symbolic expression, Implicit SINDy enables us to discover the governing dynamics as an implicit expression in derivatives, thus allowing rational functions to emerge in the dynamics.

The candidate library in both examples was constructed using nonlinear terms present in the original equations to make the analysis simple and validate the tool. The learned models obtained not only matched the behavior of the original equations but were also compact, containing few terms. This shows that **SINDy-PI may effectively reconstruct interpretable rational models without requiring a large basis**. The codes and notebooks are available at `https://github.com/Alka-CBhub/Chapter-6`.

However, an important limitation of the SINDy-PI framework frequently highlighted in the literature [8, 43, 44] is that a single trajectory may not be sufficient to uniquely identify the

model. Also, the coefficients in the rescaled model were dropped below a tolerance, which may be subjective and may affect the number of terms in the model. We were able to recover models back from a single trajectory in our analysis, probably due to the candidate library, which was constructed with prior information. In practical situations, where the actual expressions for nonlinearities are uncertain and data may be noisy or partial, the concept of multiple trajectories may become necessary to ensure model identifiability and interpretability.

Part II

Chapter 7

Cell Cycle Dynamics: Discovering Governing Equations using Implicit SINDy

7.1 Motivation and Problem Statement

Biological systems often exhibit rhythmic behavior, as seen in many processes such as cardiac dynamics, circadian clocks, and cell division. In many eukaryotic cells, the cell division cycle progresses through a strictly regulated sequence of events, namely cell growth, DNA replication, and mitotic division, which are basically governed by transcriptional feedback and checkpoint controls [63]. However, in the early embryonic cell cycles of *Xenopus laevis*, the dynamics are fundamentally different. These cycles oscillate autonomously with a period of ≈ 25 minutes and are driven entirely by post-translational protein-level interactions. The lack of transcription and the rapid transition between S-phase and mitosis make this system a minimal yet powerful model for understanding the design principles of biochemical oscillators [20, 64, 65].

To understand the design principles of this autonomous biochemical oscillator, Ferrell *et al.* [20] studied this system by developing a series of differential equation models. Their objective was to figure out how **protein-level feedback interactions** cause sustained oscillations in CDK1 activity. Their modeling framework demonstrated that the interaction between a negative feedback loop (where CDK1 activates APC, leading to CDK1 inactivation) and a positive feedback loop (where CDK1 promotes its own activation by stimulating Cdc25 and inhibiting Wee1) suffices to produce both bistability and robust oscillatory dynamics [15, 20, 23]. In this whole architecture, negative feedback is required for generating oscillatory behavior, and positive feedback adds hysteresis and sharp transitions, contributing to the robustness and switch-like nature of the dynamics. Since the regulation involves rapid transitions and delays caused by multistep biochemical processes, the authors used Hill-type response functions in their models to account for these events. These functions, characterized by their sigmoidal

shape and rational structure, effectively approximate the ultrasensitive dynamics of regulatory interactions within the feedback circuits.

In this study, we revisit the CDK1–APC regulatory architecture proposed by Ferrell *et al.* [20], decomposing it into individual feedback components to understand how distinct motifs, ranging from simple positive or negative feedback to coupled feedback loops **(Fig. 7.1 (A))** and delayed regulation **(Fig. 7.1 (B))** contribute to oscillations.

Although Ferrell's mechanistic models offered great biological insights, they heavily rely on domain-specific knowledge and hand-crafted differential equations. Recent developments in computational biology have concentrated on **automated discovery of dynamical models directly from experimental time-series data**. In this work, we aim to rediscover the underlying dynamics using time-series data simulated from the original models. For this, we employ **Implicit SINDy** (Sparse Identification of Nonlinear Dynamics), a data-driven approach capable of recovering rational nonlinearities such as Hill functions. This approach allows us to derive interpretable dynamic models directly from data, without relying on prior knowledge of network structure or kinetics.

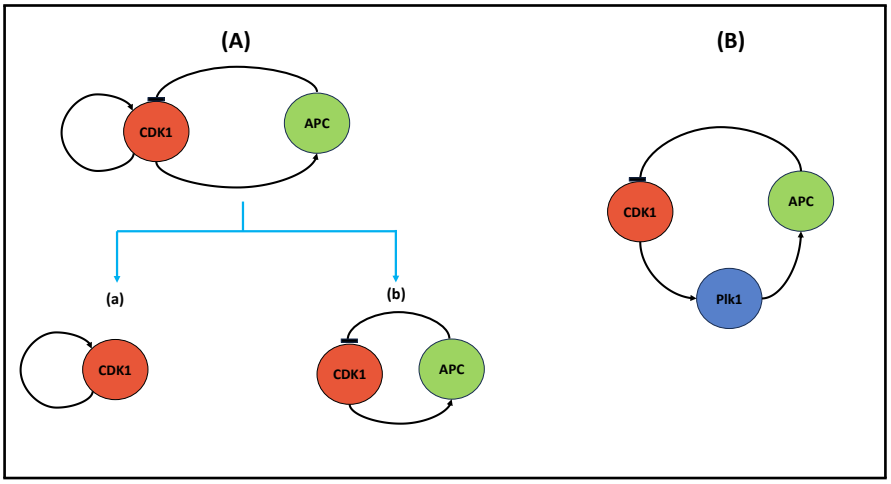

Figure 7.1. *Schematic of regulatory networks in CDK1–APC dynamics. (A) Feedback circuit combining positive and negative feedback in a two-component model of CDK1 and APC regulation: CDK1 activates APC, which in turn promotes CDK1 degradation (negative feedback). Simultaneously, CDK1 enhances its own activation by stimulating Cdc25 and inhibiting Wee1 (positive feedback). (a) Isolated positive feedback loop from (A), involving only CDK1 self-activation. (b) Isolated negative feedback loop from (A), where CDK1 activates APC and APC suppresses CDK1 activity. (B) Extended regulatory architecture where Plk1 acts as an intermediate between CDK1 and APC. This delayed negative feedback loop supports sustained oscillations even without explicit positive feedback, as captured in the three-variable model.*

Below, we apply our methodology to the models for each of the cases discussed in 7.1. Main features for each of the models are given in **Table 7.1**.

Table 7.1. *Main features of the models considered in this study.*

ID	Example 1	Example 2	Example 3	Example 4
State variables	1	2	2	3
Total parameters	5	8	11	11
Max degree $N(x)$	9	(9, 9)	(17, 9)	(9, 9, 9)
Max degree $D(x)$	8	(8, 8)	(16, 8)	(8, 8, 8)
Dynamics	Bistable	Damped oscillations	Limit-cycle oscillations	Limit-cycle oscillations

Example 1: A one-variable model of CDK1 regulation with a positive feedback:

We start with a one-variable model describing the regulation of cyclin-dependent kinase 1 (CDK1), a crucial regulator of mitotic entry in the cell cycle. The model comprises three processes: basal activation, linear decay, and a positive feedback via the self-regulation of CDK1. The feedback is characterized by a Hill function that represents ultrasensitive and cooperative activation mediated by regulatory pathways, such as Cdc25 phosphatase and Wee1 kinase. This simplified system isolates the impact of positive feedback and illustrates how bistability can arise without any additional regulatory component. The reaction network and its governing dynamics are given as follows:

(I) Reaction network:

R1: $\Theta \xrightarrow{\alpha_1} \mathrm{CDK1}^*$, Basal production.

R2: $\mathrm{CDK1}^* \xrightarrow{\beta_1} \varnothing$, Linear degradation.

R3: $\mathrm{CDK1}_{ina} + \mathrm{H}^+(\mathrm{CDK1}^*, K_3, n_3) \xrightarrow{\alpha_3} \mathrm{CDK1}^*$, Self-regulation.

(II) System dynamics:

$$\frac{d[\mathrm{CDK1}^*]}{dt} = \alpha_1 - \beta_1\,\mathrm{CDK1}^* + \alpha_3\,(1 - \mathrm{CDK1}^*)\,\frac{\mathrm{CDK1}^{*n_3}}{K_3^{n_3} + \mathrm{CDK1}^{*n_3}} \tag{7.1.1}$$

Here, $\mathrm{CDK1}^*$ and $\mathrm{CDK1}_{ina}$ represent the active and inactive form of CDK1, respectively, $H^+(\mathrm{CDK1}^*, K_3, n_3) := \frac{\mathrm{CDK1}^{*n_3}}{K_3^{n_3} + \mathrm{CDK1}^{*n_3}}$ is the Hill's activation function that has activation constant K_3 and Hill's coefficient n_3. α_1, β_1, and α_3 are the kinatic parameters. The last term in **Eq.**(7.1.1) represents CDK1 triggering its own production via an autoregulatory feedback. Also, $\mathrm{CDK1}^* + \mathrm{CDK1}_{ina} = \mathrm{CDK1}_T$, where $\mathrm{CDK1}_T$ is the total amount of CDK1. We take $\mathrm{CDK1}_T = 1$. This system shows bistability for the specific choice of parameters as shown in **Fig 7.2**.

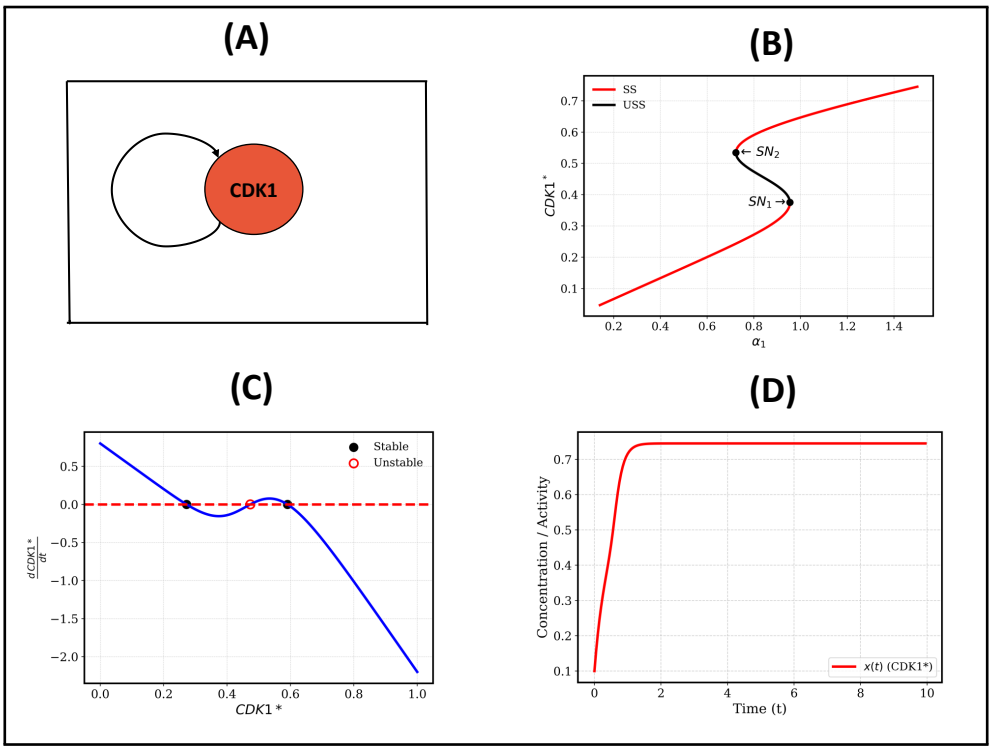

Figure 7.2. Circuit diagram, codimension-one bifurcation, rate-balance plot, and time series of CDK1 regulation. *(A) A circuit diagram of the model. (B) A plot of steady state $CDK1$ as a function of bifurcation parameter α. Red and black curves denote stable and unstable steady states, respectively. The kinetic parameters to simulate the ODE are $\beta_1 = 3, \alpha_3 = 3, K_3 = 0.5$, and $n_3 = 8$. At α_1 close to 0.72 and 0.95, the two steady states, one stable node and another unstable saddle, collide to give rise to saddle-node bifurcation, and the parameter at which this happens is called saddle-node (SN) bifurcation point. SN_1 and SN_2 denote two saddle-node bifurcations. (C) Rate-balance plot for the parameter $\alpha_1 = 0.8$. The rest of the parameters are the same. The black and red circles represent the steady states. (D) Time series of the system, starting with CDK1*[0] = 0 and approaching towards steady state.*

For simplicity of writing, we let $x := \mathrm{CDK1}^*$. Then, the **Eq.**(7.1.1) becomes:

$$\boxed{\dot{x} = \alpha_1 - \beta_1\, x + \alpha_3\,(1 - x)\,\frac{x^{n_3}}{K_3^{n_3} + x^{n_3}}} \tag{7.1.2}$$

In order to rediscover **Eq** (7.1.2) using implicit SINDy (SINDy-PI), we generate the synthetic data using the parameters and settings given in **Table 7.2** and **7.3** respectively.

Table 7.2. *Parameter values for CDK1 regulation Model .*

α_1	β_1	α_3	K_3	n_3
1.5	3	3	0.5	8

Table 7.3. *Settings used for simulation of one variable CDK1 regulation Model.*

Setting	N_{IC}	t_0	t_f	Δt
Value	1	0	500	0.05

Using the values of parameters from **Table 7.2**, **Eq 7.1.2** becomes:

$$\boxed{\dot{x} = 1.5 - 3\,x + 3\,(1 - x)\,\frac{x^8}{0.5^8 + x^8}} \tag{7.1.3}$$

Since we are utilizing synthetic time series data with a known true form, to make the further analysis more clear, we first express the model in its pure rational form :

$$\boxed{\dot{x} = \frac{-6.0\,x^9 + 4.5\,x^8 - 0.01171875\,x + 0.005859375}{1.0\,x^8 + 0.00390625}} \tag{7.1.4}$$

We, then proceed to construct the library of possible candidate terms. Note that the degree of the numerator and the denominator in **Eq.**(7.1.4) is 9 and 8, respectively. Also, when written in the implicit form, the equation is spanned by the set:

$$\underbrace{[1,\ x,\ x^8,\ x^9]}_{\text{state monomials}} \cup\ \dot{x}\ [\,x^8, 1]$$

Therefore, to ensure clarity and minimize computational cost, we choose the library for x:

$$\Theta(x, \dot{x}) = \left[\underbrace{1,\ x,\ x^8,\ x^9}_{\text{state monomials}},\ \dot{x},\ x\dot{x},\ x^8\dot{x},\ x^9\dot{x}\right]$$

The candidate library consists of three types of terms: pure monomials in the state variables, the time derivative of the state, and products of this derivative with those monomials. In order to make sure that the library is both minimal and enough expressive for recovering the true ODEs, we first extract the derivative-free monomials from the implicit form of the original equations. We next call `SindyPILibrary()`, which append the state derivative and all cross-terms generated by multiplying the derivative with each chosen monomial. The result is a compact yet complete basis of candidate functions that span the original dynamics. We employ the same strategy to construct the library in subsequent examples.

The next stage in the SINDy-PI framework involves solving the sparse regression problems for each state variable independently. In this case, only one state variable is present. In order to perform sparse regression, we first conduct a grid search by choosing a set of 20 threshold parameters logarithmically spaced between 10^{-6} and 10^{-1} to find the optimal value of threshold, λ (**Figure 7.3**). We observe that each metric returns a nearly identical value for the optimal threshold shown in **Table 7.4**.

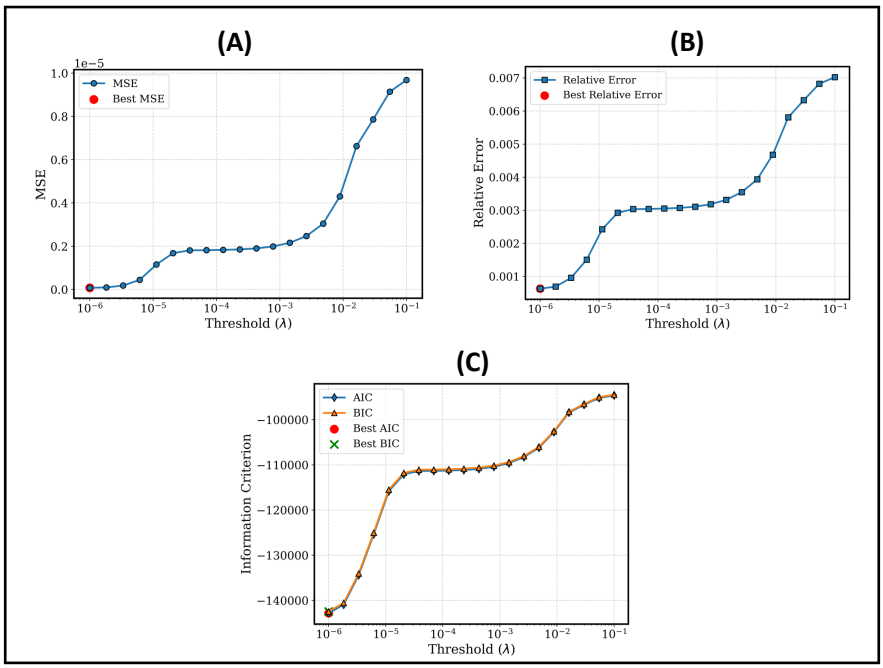

Figure 7.3. *Global diagnostic plots for determining the sparsity threshold for x in CDK1 regulation model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated over a logarithmically spaced grid from 10^{-6} to 10^{-1}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{10001\times 8}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and complexity-penalizing information criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 7.4****.*

Table 7.4. *Optimal Value of Threshold (λ) for Each Metric)*

Metric	**Best Threshold**
MSE	1×10^{-6}
Relative Error	1×10^{-6}
AIC	1×10^{-6}
BIC	1×10^{-6}

After determining the optimal threshold parameter λ, we reapply the SINDy-PI technique to generate a set of eight implicit equations. The performance of each implicit equation is assessed separately, as outlined in **Section 6.2.1**, using the same training data. **Table 7.5** shows the value of performance metrices together with the LHS term and the model index. This table ranks every equation based on relative error score along with other scores. The metrics may be used independently or collectively to evaluate the accuracy, and complexity of the model. An effective model typically has a high R^2, low relative error, and minimal AIC and BIC values.

Table 7.5. *Implicit equations for x sorted by lowest Relative Error corresponding to $\lambda = 1.0 \times 10^{-6}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
2	x^8	1.000 00	$5.393\,80\times10^{-6}$	−2.897 52	−2.897 09
3	x^9	1.000 00	$1.239\,50\times10^{-5}$	−2.790 00	−2.789 57
0	1	0.000 000	$1.737\,70\times10^{-4}$	−1.731 58	−1.731 08
1	x	9.999 09	$2.023\,50\times10^{-4}$	−1.760 14	−1.759 64
4	$\dot{x}$	9.999 39	$7.816\,00\times10^{-3}$	−1.667 50	−1.666 92
6	$x^8\dot{x}$	9.991 60	$2.896\,50\times10^{-2}$	−2.267 50	−2.267 07
7	$x^9\dot{x}$	9.990 49	$3.081\,80\times10^{-2}$	−2.334 25	−2.333 81
5	$x\,\dot{x}$	9.964 67	$5.937\,80\times10^{-2}$	−1.448 38	−1.447 94

Multiple models show a good performance score on the training data. Since a biologically interpretable model must not have singularities or negative terms in the denominator, we reformulate all models to explicit equations for further examination and obtain the following candidate models for x:

Candidate models for x:

$$\text{Model 0:}\quad \dot{x} = \frac{-5.726\,x^9 + 4.291\,x^8 - 0.01095\,x + 0.005712}{-0.0822\,x^9 + 1.0\,x^8 + 0.0007456\,x + 0.003754},$$

$$\text{Model 1:}\quad \dot{x} = \frac{5.415\,x^9 - 4.068\,x^8 + 0.01248\,x - 0.006045}{0.1552\,x^9 - 1.0\,x^8 + 0.001972\,x - 0.00431},$$

$$\text{Model 2:}\quad \dot{x} = \frac{-6.121\,x^9 + 4.595\,x^8 - 0.01366\,x + 0.006828}{1.0\,x^8 - 0.0007203\,x + 0.004642},$$

$$\text{Model 3:}\quad \dot{x} = \frac{-6.091\,x^9 + 4.541\,x^8 - 0.006395\,x + 0.004393}{1.0\,x^8 + 0.007161\,x + 0.002047},$$

$$\text{Model 4:}\quad \dot{x} = \frac{-5.633\,x^9 + 4.223\,x^8 - 0.01108\,x + 0.005706}{-0.1035\,x^9 + 1.0\,x^8 + 0.0002618\,x + 0.003816},$$

$$\text{Model 5:}\quad \dot{x} = \frac{-0.1884\,x^9 + 0.02904\,x - 0.008325}{1.449\,x^9 - 1.0\,x^8 + 0.03931\,x - 0.01049},$$

$$\text{Model 6:}\quad \dot{x} = \frac{-0.04865\,x^8 + 0.00734\,x - 0.0008533}{1.383\,x^9 - 1.0\,x^8 + 0.01476\,x - 0.002208},$$

$$\text{Model 7:}\quad \dot{x} = \frac{-0.04946\,x^8 + 0.007595\,x - 0.0009663}{1.389\,x^9 - 1.0\,x^8 + 0.01484\,x - 0.002279}.$$

Among all the possible models described above, only **Model 3** is both interpretable as well exhibits a good performance score. Consequently, we choose **Model 3** as the final model.

$$\dot{x} = \frac{-6.091\,x^9 + 4.541\,x^8 - 0.006395\,x + 0.004393}{1.0\,x^8 + 0.007161\,x + 0.002047}, \qquad \text{Extra Terms: 1.} \tag{7.1.5}$$

In the next step, we compare the original coefficients and the estimated coefficients for each term in the numerator and the denominator by listing the minimum and maximum values of the coefficients across all final candidate models. In this case, we obtained only one candidate model. Consequently, the minimum and maximum values are identical (**Table 7.6** and **7.7**).

Table 7.6. *Comparison of coefficients for the numerator terms for $\dot{x}$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	x^9	-6.0	-6.091	-6.091
2	x^8	4.5	4.541	4.541
3	x	-0.01171875	-0.006395	-0.006395
4	1	0.005859375	0.004393	0.004393

Table 7.7. *Comparison of coefficients for the denominator terms for $\dot{x}$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	x^8	1.0	1.0	1.0
2	x	0.0	0.007161	0.007161
3	1	0.00390625	0.002047	0.002047

The identified model accurately preserves all original terms together with their appropriate signs. We further validate its accuracy by calculating the steady state and examining the local interaction network. Despite incorporating one extra term, the model exhibit the same transient steady state behavior as in the original model shown in **Figure 7.4**.

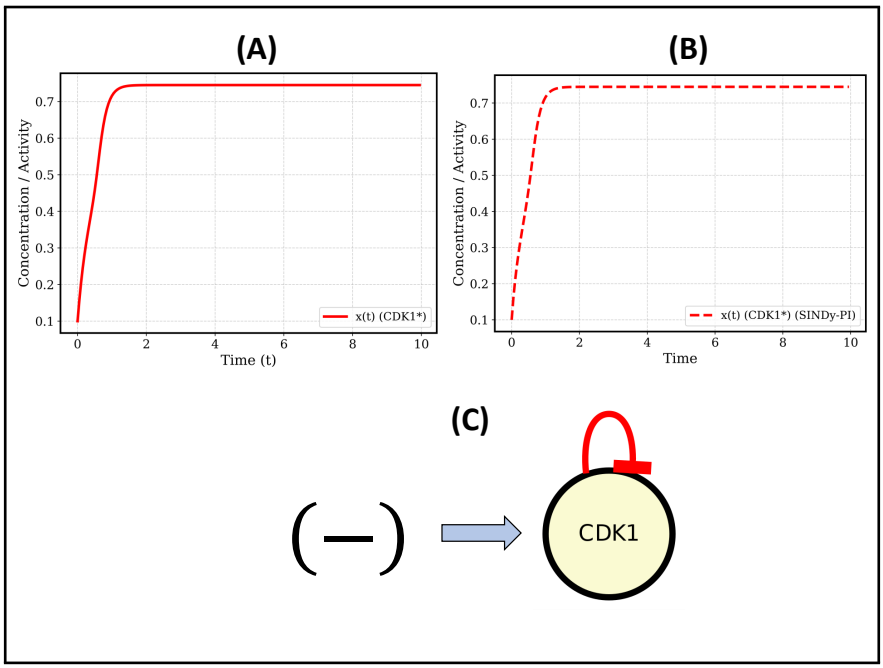

Figure 7.4. *Results from the Implicit SINDy (SINDy-PI) method applied to the CDK1 regulation model. (A) and (B) show the original and SINDy simulated time series of the CDK1 regulation model for the variable x. (C) represents the sign of the Jacobian evaluated at the steady state $x^* = 0.745$, along with the resulting interaction network for the identified model. The red arrows indicate negative influence (inhibition).*

Example 2: A two-variable model of CDK1–APC regulation with negative feedback

We next examine a two-variable model that captures a negative feedback loop between CDK1 and the Anaphase-Promoting Complex (APC). In this formulation, CDK1 activates APC, which promotes CDK1 degradation or inactivation, resulting in a negative feedback loop. The model has two coupled differential equations: one representing the active form of CDK1 and the other representing the active form of APC. Both regulatory interactions are modeled using Hill functions to incorporate ultrasensitivity. This model does not incorporate any positive feedback, hence lacking the structural prerequisites for bistability. The dynamics in this case generally display either a single stable steady state or damped oscillations, depending upon parameter values [23]. The reaction networks and interaction dynamics for this system are:

(I) Reaction network:

R1:	$\varnothing \xrightarrow{\alpha_1} \mathrm{CDK1}^*,$	Basal Production.
R2:	$\mathrm{CDK1}^* + \mathrm{H}^+(\mathrm{APC}^*, K_1, n_1) \xrightarrow{\beta_1} \mathrm{CDK1}_{ina},$	Regulation of CDK1 via APC.
R3:	$\mathrm{APC}_{ina} + \mathrm{H}^+(\mathrm{CDK1}^*, K_2, n_2) \xrightarrow{\alpha_2} \mathrm{APC}^*,$	Regulation of APC via CDK1.
R4:	$\mathrm{APC}^* \xrightarrow{\beta_3} \varnothing,$	Linear degradation.

(II) System dynamics:

$$\frac{d[\mathrm{CDK1}^*]}{dt} = \alpha_1 - \beta_1\,\mathrm{CDK1}^*\,\frac{\mathrm{APC}^{*n_1}}{K_1^{n_1} + \mathrm{APC}^{*n_1}}, \tag{7.1.6}$$

$$\frac{d[\mathrm{APC}^*]}{dt} = \alpha_2\,(1 - \mathrm{APC}^*)\,\frac{\mathrm{CDK1}^{*n_2}}{K_2^{n_2} + \mathrm{CDK1}^{*n_2}} - \beta_2\,\mathrm{APC}^* \tag{7.1.7}$$

Here, $\mathrm{CDK1}^*$, APC^* and $\mathrm{CDK1}_{ina}$, APC_{ina} represent the active and inactive form of CDK1 and APC, respectively. Also, $\mathrm{CDK1}^* + \mathrm{CDK1}_{ina} = \mathrm{CDK1}_T$, and $\mathrm{APC}^* + \mathrm{APC}_{ina} = \mathrm{APC}_T$, where $\mathrm{CDK1}_T$ and APC_T represent the total amount of CDK1 and APC respectively. We take $\mathrm{CDK1}_T = \mathrm{APC}_T = 1$. The parameters α_1, α_2, β_1, and β_2 are the kinetic parameters.

Note that, the second term in **Eq.**(7.1.6) delineates the inactivation of CDK1* by active APC, whereas the first term in **Eq.**(7.1.17) indicates the activation of APC by CDK1*. The terms $H^+(\mathrm{CDK1}^*, K_2, n_2)$ and $H^+(\mathrm{APC}^*, K_1, n_1)$ collectively form a negative feedback loop. The system exhibits damped oscillations for the choice of specific parameters, as illustrated in **Fig 7.5**. The parameter values and simulation settings to generate the training data are given below:

Table 7.8. *Parameter values for the two-variable CDK1–APC model.*

Parameter	α_1	α_2	β_1	β_2	K_1	K_2	K_3	n_1	n_2	n_3
Value	1.5	3	3	1	0.5	0.5	0.5	8	8	8

Table 7.9. *Settings used for simulation of CDK1–APC model.*

Setting	N_{IC}	t_0	t_f	Δt
Value	1	0	500	0.05

Again for simplicity, we let $x_1 := \mathrm{CDK1}^*$, $x_2 := \mathrm{APC}^*$. The equations (7.1.6) and (7.1.7) then transformed into:

$$\frac{dx_1}{dt} = \alpha_1 - \beta_1\,x_1\,\frac{x_2^{n_1}}{K_1^{n_1} + x_2^{n_1}}, \tag{7.1.8}$$

$$\frac{dx_2}{dt} = \alpha_2\,(1 - x_2)\,\frac{x_1^{n_2}}{K_2^{n_2} + x_1^{n_2}} - \beta_2\,x_2 \tag{7.1.9}$$

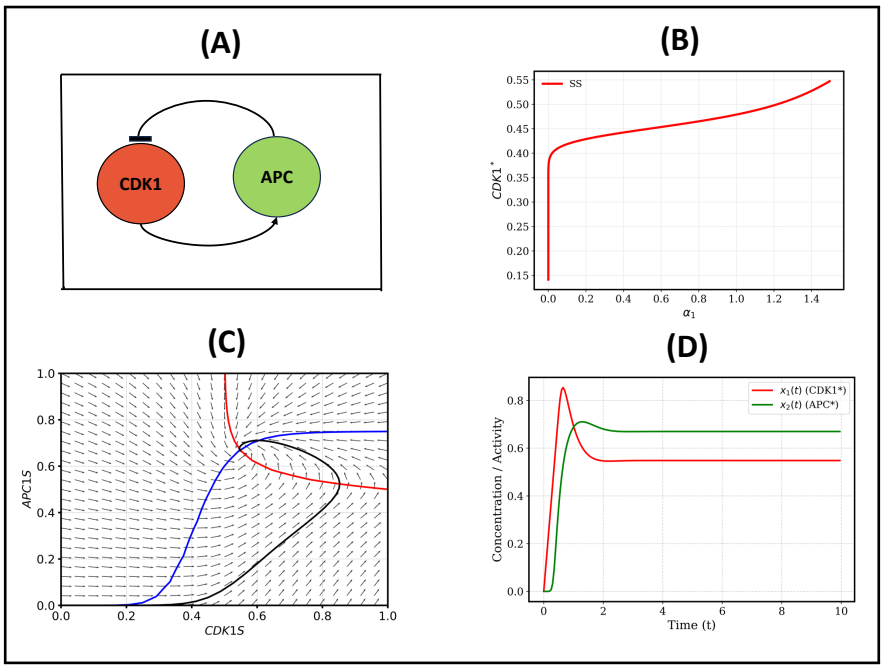

Figure 7.5. *Circuit diagram, codimension-one bifurcation diagram, phase diagram, and time series of CDK1-APC regulation. (A) A circuit diagram of the model. (B) A plot of steady state CDK1* as a function of bifurcation parameter α_1. It reveals a graded response, where the steady-state value of the system output increases smoothly and continuously, corresponding to the parameter α_1. (C) Phase plane diagram of the system. The black arrows represent direction fields. The red and green curves represent the two nullclines of the system. These two nullclines intersect at ($\mathrm{CDK1}^* \approx 0.54, \mathrm{APC} \approx 0.67$), which is a stable steady state. A black trajectory is shown, which starts from (0,0) and approaches toward the steady state.(D) Time series of the system, starting with $\mathrm{CDK1}^*[0] = \mathrm{APC}[0] = 0$, showing damped oscillations.*

Substituting the parameter values yields the following equations:

$$\boxed{\begin{aligned} \dot{x}_1 &= 1.5 - 3\,x_1\,\frac{x_2^8}{x_2^8 + 0.5^8} \\ \dot{x}_2 &= 3\,(1 - x_2)\,\frac{x_1^8}{x_1^8 + 0.5^8} - x_2 \end{aligned}} \tag{7.1.10}$$

$$\tag{7.1.11}$$

This can be further simplified into a purely rational form:

$$\dot{x}_1 = \frac{-3.0\,x_1\,x_2^8 \;+\; 1.5\,x_2^8 \;+\; 0.005859375}{1.0\,x_2^8 \;+\; 0.00390625} \tag{7.1.12}$$

$$\dot{x}_2 = \frac{-4.0\,x_1^8 x_2 + 3.0\,x_1^8 - 0.00390625\,x_2}{1.0\,x_1^8 + 0.00390625} \tag{7.1.13}$$

We build a separate library for both the state variables x_1 and x_2 and use SINDy-PI to recover each equation independently. The selected candidate libraries for x_1 and x_2 are:

$$\Theta_1(x, \dot{x}_1) = \left[\,1,\; x_2^8,\; x_1 x_2^8,\; \dot{x}_1,\; \dot{x}_1\,x_2^8,\; \dot{x}_1\,x_1\,x_2^8\,\right],$$

$$\Theta_2\left(x, \dot{x}_2\right) = \left[x_2, x_1^8, x_1^8 x_2, \dot{x}_2, x_2\dot{x}_2, x_1^8\dot{x}_2, x_1^8 x_2\dot{x}_2\right]$$

As in Example 1, we perform a grid search independently to identify the optimal value of λ for each equation. The optimal thresholds and equation-wise model performance for each x_1 and x_2 are summarized in the subsequent tables.

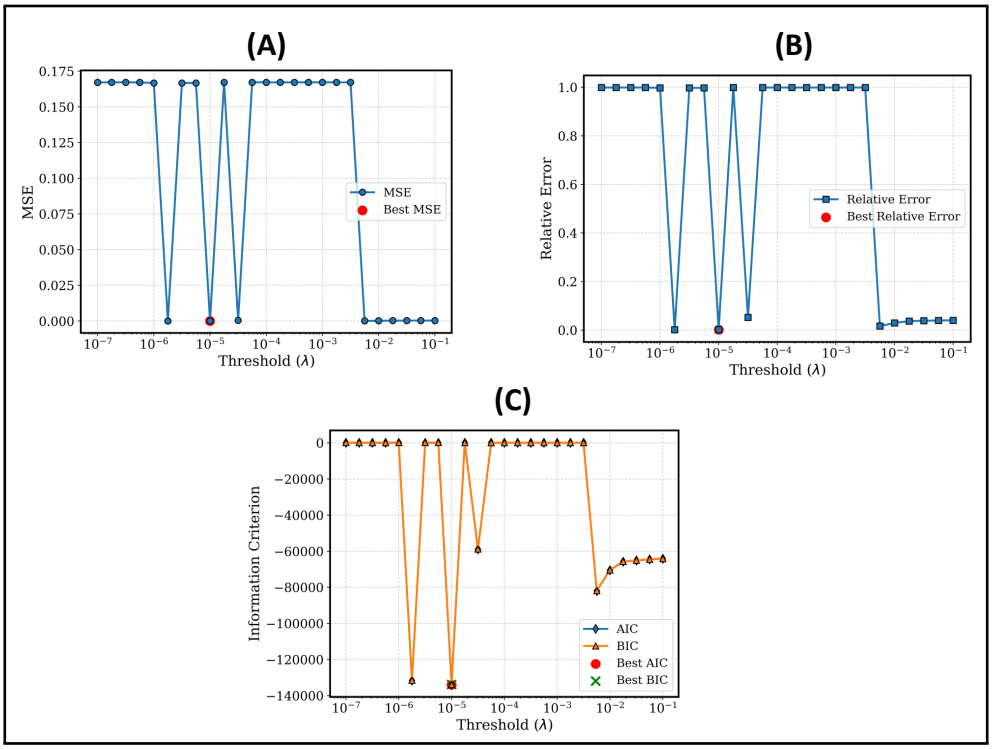

Figure 7.6. *Global diagnostic plots for determining the sparsity threshold for x_1 in CDK1-APC regulation model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated on 25 points logarithmically spaced between 10^{-7} to 10^{-1}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{10001\times 6}$ is approximated by $\hat{Y} = \Theta\,\Xi$, and model performance is assessed by residual norms and complexity-penalizing information criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 7.10****.*

Table 7.10. *Optimal Value of Threshold (λ) for $\dot{x}_1$ corresponding to each metric.*

Metric	Best Threshold
MSE	1×10^{-5}
Relative Error	1×10^{-5}
AIC	1×10^{-5}
BIC	1×10^{-5}

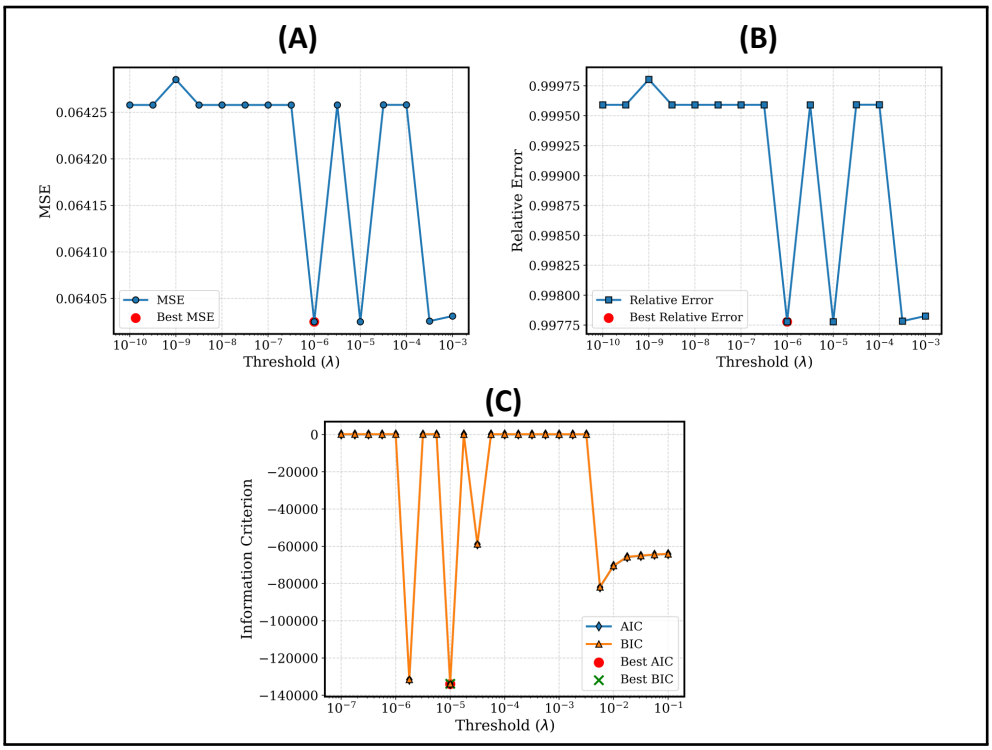

Figure 7.7. *Global diagnostic plots for determining the sparsity threshold for x_2 in CDK1-APC regulation model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated over 15 points logarithmically spaced between 10^{-10} to 10^{-3}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{10001\times 7}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and complexity-penalizing criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 7.11.***

Table 7.11. *Optimal Value of Threshold (λ) for $\dot{x}_2$ corresponding to each metric.*

Metric	Best Threshold
MSE	1×10^{-6}
Relative Error	1×10^{-6}
AIC	3.162×10^{-4}
BIC	1×10^{-9}

Table 7.12. *Implicit equations for x_1 sorted by lowest Relative Error corresponding to $\lambda = 1.0 \times 10^{-5}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
2	$x_1 x_2^8$	9.999 97	$7.681\,60\times10^{-5}$	−2.656 29	−2.656 01
1	x_2^8	9.999 96	$8.222\,80\times10^{-5}$	−2.522 54	−2.522 25
0	1	0.000 000	$6.847\,60\times10^{-4}$	−1.457 31	−1.456 88
3	$\dot{x}_1$	9.996 22	$1.943\,00\times10^{-2}$	−1.377 65	−1.377 29
4	$\dot{x}_1 x_2^8$	9.973 06	$5.185\,70\times10^{-2}$	−2.062 87	−2.062 58
5	$x_1 \dot{x}_1 x_2^8$	9.971 21	$5.361\,20\times10^{-2}$	−2.133 63	−2.133 35

Table 7.13. *Implicit equations for x_1 sorted by lowest Relative Error corresponding to $\lambda = 1 \times 10^{-6}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
1	x_1^8	1.000 00	$2.609\,40\times10^{-4}$	−2.569 09	−2.568 73
2	$x_1^8 x_2$	1.000 00	$3.120\,90\times10^{-4}$	−2.626 13	−2.625 77
5	$x_1^8 \dot{x}_2$	9.999 98	$1.528\,80\times10^{-3}$	−2.346 45	−2.346 02
6	$x_1^8 x_2 \dot{x}_2$	9.988 05	$3.455\,80\times10^{-2}$	−1.864 79	−1.864 42
3	$\dot{x}_2$	9.978 41	$4.643\,50\times10^{-2}$	−1.255 64	−1.255 21
4	$x_2 \dot{x}_2$	9.868 41	$1.146\,50\times10^{-1}$	−1.288 11	−1.287 68
0	x_2	−1.100 21	1.000 00	−8.026 66	−8.026 66

We select optimal $\lambda = 10^{-5}$ for x_1 and $\lambda = 10^{-6}$ for x_2, respectively and then apply implicit SINDy (SINDy-PI). The final candidate models and the corresponding comparison table for each variable is shown below:

Candidate models for x_1:

$$\text{Model 1:} \quad \dot{x}_1 = \frac{-3.002\, x_1\, x_2^8 \;+\; 1.498\, x_2^8 \;+\; 0.005995}{1.0\, x_2^8 \;+\; 0.004035}, \qquad \text{Extra Terms: 0}$$

$$\text{Model 2:} \quad \dot{x}_1 = \frac{-3.006\, x_1\, x_2^8 \;+\; 1.497\, x_2^8 \;+\; 0.006134}{1.0\, x_2^8 \;+\; 0.004131}, \qquad \text{Extra Terms: 0.}$$

Candidate models for x_2:

$$\text{Model 1:}\quad \dot{x}_2 = \frac{-4.017\,x_1^8\,x_2 \;+\; 3.019\,x_1^8 \;-\; 0.004001\,x_2}{1.0\,x_1^8 \;+\; 0.002226\,x_2 \;+\; 0.004209}, \qquad \text{Extra Terms: 1}$$

$$\text{Model 2:}\quad \dot{x}_2 = \frac{-4.018\,x_1^8\,x_2 \;+\; 3.020\,x_1^8 \;-\; 0.003999\,x_2}{1.0\,x_1^8 \;+\; 0.002173\,x_2 \;+\; 0.004221}, \qquad \text{Extra Terms: 1}$$

$$\text{Model 3:}\quad \dot{x}_2 = \frac{-4.018\,x_1^8\,x_2 \;+\; 3.020\,x_1^8 \;-\; 0.003998\,x_2}{0.0005981\,x_1^8\,x_2 \;+\; 1.0\,x_1^8 \;+\; 0.001927\,x_2 \;+\; 0.004284}, \qquad \text{Extra Terms: 2}$$

$$\text{Model 4:}\quad \dot{x}_2 = \frac{-4.247\,x_1^8\,x_2 \;+\; 3.227\,x_1^8 \;-\; 0.004660\,x_2}{0.1551\,x_1^8\,x_2 \;+\; 1.0\,x_1^8 \;+\; 0.01187\,x_2 \;+\; 0.003624}, \qquad \text{Extra Terms: 2.}$$

Table 7.14. *Comparison of coefficients for the numerator terms for $\dot{x}_1$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1x_2^8$	-3.0	-3.006	-3.002
2	x_2^8	1.5	1.497	1.498
3	1	0.005859375	0.005995	0.006134

Table 7.15. *Comparison of coefficients for the denominator terms for $\dot{x}_1$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	x_2^8	1.0	1.0	1.0
2	1	0.00390625	0.004035	0.004131

Table 7.16. *Comparison of coefficients for the numerator terms for $\dot{x}_2$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1^8x_2$	-4.0	-4.247	-4.017
2	x_1^8	3.0	3.019	3.227
3	x_2	-0.00390625	-0.004660	-0.003998

Table 7.17. *Comparison of coefficients for the denominator terms for $\dot{x}_2$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1^8x_2$	0	0	0.1551
2	x_1^8	1.0	1.0	1.0
3	x_2	0	0.001927	0.01187
4	1	0.00390625	0.003624	0.004284

We have a total 8 pair of candidate models for the original system. We determined the steady states and the Jacobian for all pair and then evaluated Jacobian there. Any combination of the models for x_1 and x_2 gives the steady state, $\approx (0.548, 0.670)$ aligning with the original

system. Also, for each case, the Jacobian exhibits the common pattern of signs matching with the original system shown in **Figure 7.8**.

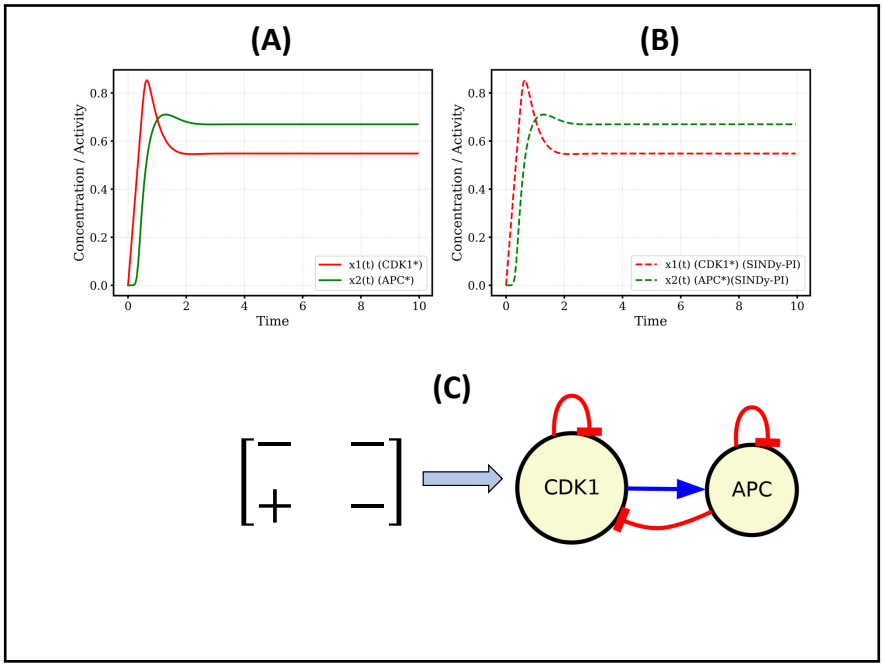

Figure 7.8. *Results from the Implicit SINDy (SINDy-PI) method applied to the CDK1-APC regulation model. (A) and (B) show the original and SINDy simulated time series of the CDK1-APC regulation model for the variable x_1 and x_2 respectively. (C) represents the sign of the Jacobian evaluated at the steady state $(x_1^*, x_2^*) = (0.548, 0.670)$, along with the resulting interaction network for the identified model. Blue arrows indicate positive regulation (activation), and red arrows indicate negative regulation (inhibition).*

Example 3: CDK1–APC regulation model with coupled feedback:

In the previous two examples, we analyzed the positive and negative feedback mechanisms regulating CDK1 independently. This example examines a two-variable model that integrates both positive and negative feedback mechanisms regulating CDK1. Similar to the previous paradigm, CDK1 activates APC, which then inhibits CDK1, thus forming a negative feedback loop. However, this model includes a nonlinear positive feedback term represented by a Hill-type function. It characterize CDK1 autoactivation through regulatory mechanisms involving Cdc25 phosphatase and Wee1 kinase and enables the system to exhibit bi-stability in CDK1 activity. The interplay between the bistable switch and the delayed negative feedback through the APC gives rise to sustained oscillations [20]. Consequently, the entire system captures both the decision-making function of CDK1 (via positive feedback) and the resetting role of APC (through negative feedback) in the control of the cell cycle. For this system, the reaction network and the governing dynamics are as follows:

(I) Reaction network:

R1: $\varnothing \xrightarrow{\alpha_1} \mathrm{CDK1}^*$, Basal Production.

R2: $\mathrm{CDK1}^* + \mathrm{H}^+(\mathrm{APC}^*, K_1, n_1) \xrightarrow{\beta_1} \mathrm{CDK}_{ina}$, Regulation of CDK1 via APC.

R3: $\mathrm{CDK1}_{ina} + \mathrm{H}^+(\mathrm{CDK1}^*, K_3, n_3) \xrightarrow{\alpha_3} \mathrm{CDK1}^*$, Self-regulation of CDK1

R4: $\mathrm{APC}_{ina} + \mathrm{H}^+(\mathrm{CDK1}^*, K_2, n_2) \xrightarrow{\alpha_2} \mathrm{APC}^*$, Regulation of APC via CDK1.

R5: $\mathrm{APC}^* \xrightarrow{\beta_3} \varnothing$, Linear degradation of APC.

(II) System dynamics:

$$\frac{d[\mathrm{CDK1}^*]}{dt} = \alpha_1 - \beta_1\,\mathrm{CDK1}^*\,\frac{\mathrm{APC}^{*n_1}}{K_1^{n_1} + \mathrm{APC}^{*n_1}} + \alpha_3\,(1 - \mathrm{CDK1}^*)\,\frac{\mathrm{CDK1}^{*n_3}}{K_3^{n_3} + \mathrm{CDK1}^{*n_3}}, \tag{7.1.14}$$

$$\frac{d[\mathrm{APC}^*]}{dt} = \alpha_2\,(1 - \mathrm{APC}^*)\,\frac{\mathrm{CDK1}^{*n_2}}{K_2^{n_2} + \mathrm{CDK1}^{*n_2}} - \beta_2\,\mathrm{APC}^* \tag{7.1.15}$$

The second term in **Eq.**(7.1.14) reflects CDK1* inactivation by active APC, whereas the third term captures positive autoregulation of CDK1*. The first term in **Eq.**(7.1.15) represents the activation of APC by CDK1*. Both terms $H^+(\mathrm{CDK1}^*, K_2, n_2)$ and $H^+(\mathrm{APC}^*, K_1, n_1)$ collectively constitute a **delayed negative feedback loop**. The system has sustained oscillations for specific choice of parameters shown in **Fig 7.9**.

Table 7.18. *Parameter values for the two-variable CDK1–APC model with a positive feedback.*

Parameter	α_1	α_2	α_3	β_1	β_2	K_1	K_2	K_3	n_1	n_2	n_3
Value	1.5	3	3	3	1	0.5	0.5	0.5	8	8	8

Table 7.19. *Settings used for simulation of CDK1–APC model with a positive feedback.*

Setting	N_{IC}	t_0	t_f	Δt
Value	1	0	50	0.05

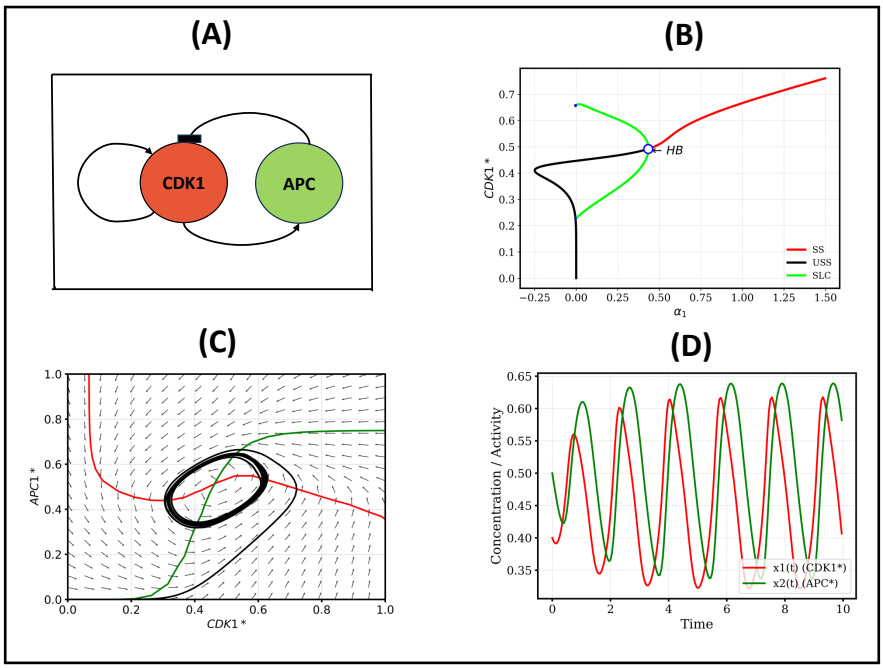

Figure 7.9. *Circuit diagram, one-parameter bifurcation diagram, phase diagram, and time series of CDK1-APC regulation with combined feedback. (A) A circuit diagram of the model.(B) A plot of steady state $CDK1$ as a function of bifurcation parameter α_1. Red and black curves denote stable and unstable steady states respectively. The kinetic parameters to simulate the ODE are $\alpha_2 = 3$, $\beta_1 = 3$, $\beta_2 = 3$, $\alpha_3 = 3$, $K_1 = K_2 = K_3 = 0.5$, and $n_1 = n_2 = n_3 = 8$. The bifurcation here is saddle-node of infinite period bifurcation (SNIPER) since a saddle and a stable node collapse into a single stationary point on a closed orbit . (C) Phase diagram of the system. The black arrows represent direction fields. The red and green curves represent the two nullclines of the system. These two nullclines intersects at $(\mathrm{CDK1}^* \approx 0.46, \mathrm{APC} \approx 0.51)$, which is a stable steady state. A black trajectory is shown which starts from (0,0) and converges to a limit cycle.(D) Time series of the system, starting with $(\mathrm{CDK1}^* \approx 0.46, \mathrm{APC} \approx 0.51)$, showing sustained oscillations.*

Following the same notation for the variables and for the parameters given in **Table 7.18**, the system becomes:

$$\dot{x_1} = 1.5 - 3\,x_1\,\frac{x_2^8}{0.5^8 + x_2^8} + 3\,(1 - x_1)\,\frac{x_1^8}{0.5^8 + x_1^8} \tag{7.1.16}$$

$$\dot{x_2} = 3\,(1 - x_2)\,\frac{x_1^8}{0.5^8 + x_1^8} - x_2 \tag{7.1.17}$$

which on further simplification gives:

$$\dot{x}_1 = \frac{-6.0\,x_1^9x_2^8 \;-\; 0.0117\,x_1^9 \;+\; 4.5\,x_1^8x_2^8 \;+\; 0.0176\,x_1^8 \;-\; 0.0117\,x_1x_2^8 \;+\; 0.00586\,x_2^8 \;+\; 2.29\times10^{-5}}{1.0\,x_1^8x_2^8 \;+\; 0.00391\,x_1^8 \;+\; 0.00391\,x_2^8 \;+\; 1.53\times10^{-5}},$$

$$\dot{x}_2 = \frac{-4.0\,x_1^8x_2 \;+\; 3.0\,x_1^8 \;-\; 0.00391\,x_2}{1.0\,x_1^8 \;+\; 0.00391}. \tag{7.1.18}$$

Analogous to the previous example, we implement SINDy-PI on the training data by constructing distinct libraries for the state variables x_1 and x_2. The selected libraries in this case are:

$$\Theta_1\big(x,\dot{x}_1\big) = \begin{Bmatrix} 1,\; x_1^8,\; x_1^9,\; x_2^8,\; x_1x_2^8,\; x_1^8x_2^8,\; x_1^9x_2^8,\; \dot{x}_1, \\ x_1^8\dot{x}_1,\; x_1^9\dot{x}_1,\; \dot{x}_1\,x_2^8,\; x_1\,\dot{x}_1\,x_2^8,\; x_1^8\,\dot{x}_1\,x_2^8,\; x_1^9\,\dot{x}_1\,x_2^8 \end{Bmatrix},$$

$$\Theta_2\big(x,\dot{x}_2\big) = \{x_2, x_1^8, x_1^8x_2, \dot{x}_2, x_2\dot{x}_2, x_1^8\dot{x}_2, x_1^8x_2\dot{x}_2\}.$$

The results of the grid search for optimal thresholds and equation-wise model performance for each x_1 and x_2 are summarized in the following tables.

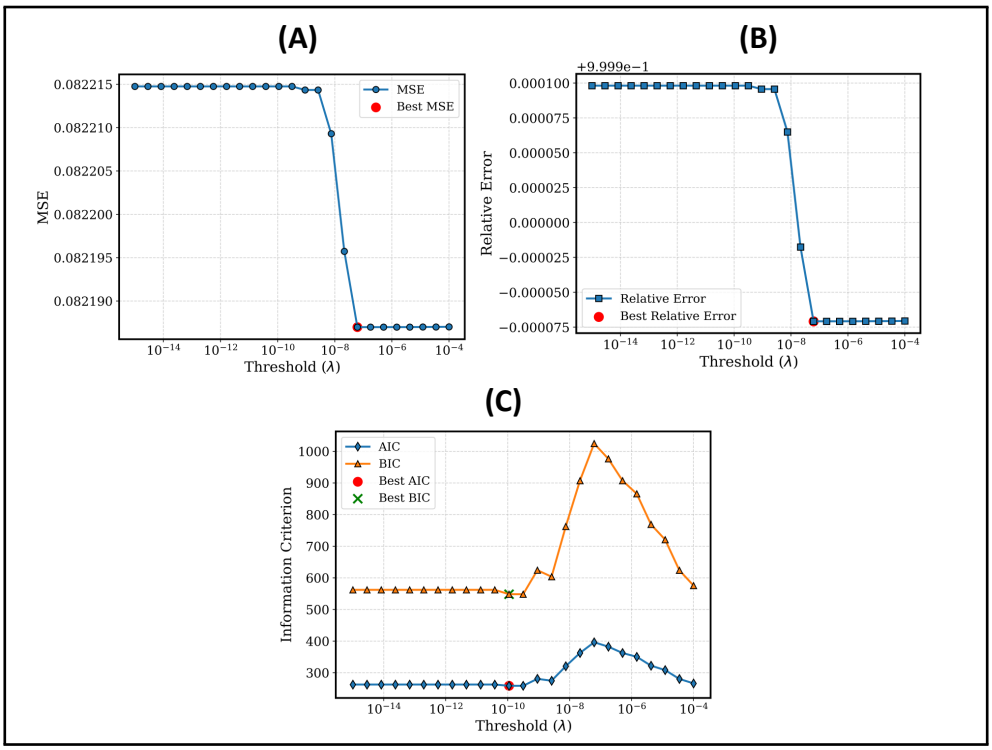

Figure 7.10. *Global diagnostic plots for determining the sparsity threshold for x_1 in CDK1-APC regulation model with a positive feedback. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated at 25 points logarithmically spaced between 10^{-15} to 10^{-4}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{1001\times 14}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and information criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are shown in* ***Table 7.20****.*

Table 7.20. *Optimal Value of Threshold (λ) for $\dot{x_1}$ corresponding to each metric.*

Metric	Best Threshold
MSE	6.19×10^{-8} (No any valid models)
Relative Error	6.19×10^{-8} (No any valid models)
AIC	1.101×10^{-10}
BIC	1.101×10^{-10}

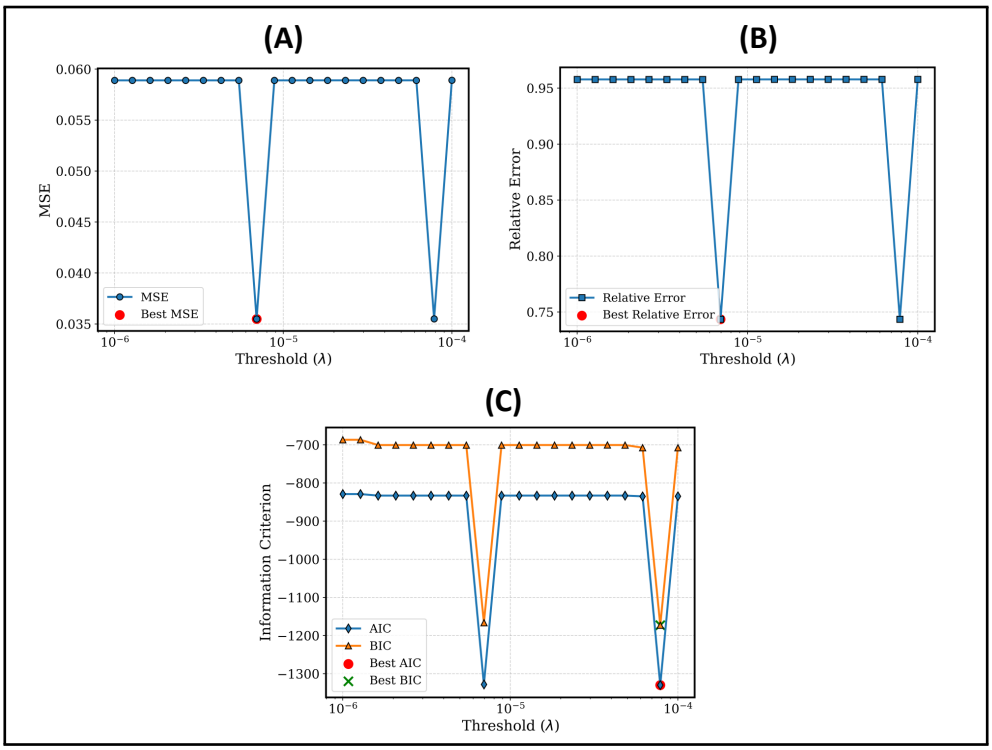

Figure 7.11. *Global diagnostic plots for determining the sparsity threshold for $x_2 1$ in CDK1-APC regulation model with a positive feedback. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated at 20 points logarithmically spaced between 10^{-6} to 10^{-4}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{1001\times 7}$ is approximated by $\hat{Y} = \Theta\boldsymbol{\Xi}$, and model performance is assessed by residual norms and complexity-penalizing criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 7.21****.*

Table 7.21. *Optimal Value of Threshold (λ) for $\dot{x_2}$ corresponding to each metric.*

Metric	Best Threshold
MSE	6.952×10^{-6}
Relative Error	6.952×10^{-6}
AIC	7.848×10^{-5}
BIC	7.848×10^{-5}

Table 7.22. *Implicit equations for x_1 sorted by lowest Relative Error corresponding to $\lambda = 1.101 \times 10^{-10}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
6	$x_1^9 x_2^8$	9.999 98	$1.138\,00\times10^{-3}$	−3.274 17	−3.268 28
5	$x_1^8 x_2^8$	9.999 98	$1.146\,00\times10^{-3}$	−3.166 13	−3.160 24
12	$x_1^8 \dot{x}_1 x_2^8$	9.999 81	$3.894\,30\times10^{-3}$	−3.153 66	−3.147 77
13	$x_1^9 \dot{x}_1 x_2^8$	9.999 68	$5.171\,90\times10^{-3}$	−3.203 13	−3.198 22
9	$x_1^9 \dot{x}_1$	9.987 12	$3.589\,30\times10^{-2}$	−1.900 11	−1.893 73
11	$x_1 \dot{x}_1 x_2^8$	−4.818 20	1.000 00	−1.208 28	−1.208 28
8	$x_1^8 \dot{x}_1$	−9.800 00	1.000 00	−1.139 70	−1.139 70

Table 7.23. *Implicit equations for x_1 sorted by lowest Relative Error corresponding to $\lambda = 6.952 \times 10^{-6}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
1	x_1^8	9.999 95	$1.821\,80\times10^{-3}$	−2.196 21	−2.193 75
2	$x_1^8 x_2$	9.999 90	$2.620\,30\times10^{-3}$	−2.246 57	−2.244 12
5	$x_1^8 \dot{x}_2$	9.999 17	$8.435\,20\times10^{-3}$	−1.966 15	−1.963 20
3	$\dot{x}_2$	9.990 63	$3.060\,50\times10^{-2}$	−8.775 82	−8.741 46
6	$x_1^8 x_2 \dot{x}_2$	9.933 73	$7.490\,70\times10^{-2}$	−1.671 01	−1.668 56
4	$x_2 \dot{x}_2$	9.834 02	$1.288\,10\times10^{-1}$	−7.374 92	−7.345 47
0	x_2	−1.119 09	1.000 00	−1.397 17	−1.397 17

With the best-fit thresholds determined from the grid search ($\lambda = 1.101 \times 10^{-10}$ for x_1 and $\lambda = 6.952 \times 10^{-6}$ for x_2), we apply SINDy-PI to reconstruct the governing dynamics. The resulting models, along with comparison tables, are provided below:

Candidate models for x_1:

$$\dot{x}_1 = \frac{-18.45\,x_1^9 x_2^8 \;-\; 0.03101\,x_1^9 \;+\; 10.05\,x_1^8 x_2^8 \;+\; 0.03487\,x_1^8 \;-\; 0.04254\,x_1 x_2^8 \;+\; 0.005333\,x_2^8 \;+\; 8.429\times10^{-6}}{2.459\,x_1^9 x_2^8 \;+\; 0.001347\,x_1^9 \;+\; 1.0\,x_1^8 x_2^8 \;+\; 0.01197\,x_1^8 \;+\; 0.01837\,x_1 x_2^8 \;+\; 0.005259\,x_2^8 \;+\; 4.153\times10^{-5}}, \quad \text{Extra Terms: 3}$$

$$\dot{x}_1 = \frac{-7.338\,x_1^9 x_2^8 \;-\; 0.01336\,x_1^9 \;+\; 3.96\,x_1^8 x_2^8 \;+\; 0.01476\,x_1^8 \;-\; 0.01667\,x_1 x_2^8 \;+\; 0.001956\,x_2^8 \;+\; 3.366\times10^{-6}}{0.1632\,x_1^9 x_2^8 \;+\; 0.0005767\,x_1^9 \;+\; 1.0\,x_1^8 x_2^8 \;+\; 0.004951\,x_1^8 \;+\; 0.003671\,x_1 x_2^8 \;+\; 0.003462\,x_2^8 \;+\; 1.716\times10^{-5}} \quad \text{Extra Terms: 3.} \tag{7.1.19}$$

Candidate models for x_2:

$$\dot{x}_2 = \frac{-3.919\,x_1^8 x_2 \;+\; 2.963\,x_1^8 \;-\; 0.004023\,x_2}{1.0\,x_1^8 \;+\; 0.0004261\,x_2 \;+\; 0.003882}, \quad \text{Extra Terms: 1.}$$

Table 7.24. *Comparison of coefficients for the numerator terms for* $\dot{x}_1$.

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1^9\, x_2^8$	-6.0	-18.45	-7.338
2	x_1^9	-0.0117	-0.03101	-0.01336
3	$x_1^8\, x_2^8$	4.5	3.96	10.05
4	x_1^8	0.0176	0.01476	0.03487
5	$x_1\, x_2^8$	-0.0117	-0.04254	-0.01667
6	x_2^8	0.00586	0.001956	0.005333
7	1	2.29×10^{-5}	3.366×10^{-6}	8.429×10^{-6}

Table 7.25. *Comparison of coefficients for the denominator terms for* $\dot{x}_1$.

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1^9\, x_2^8$	0.0	0.1632	2.459
2	x_1^9	0.0	0.0005767	0.001347
3	$x_1^8\, x_2^8$	1.0	1.0	1.0
4	x_1^8	0.00391	0.004951	0.01197
5	$x_1\, x_2^8$	0.0	0.003671	0.01837
6	x_2^8	0.00391	0.003462	0.005259
7	1	1.53×10^{-5}	1.716×10^{-5}	4.153×10^{-5}

Table 7.26. *Comparison of coefficients for the numerator terms for* $\dot{x}_2$.

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1^8\, x_2$	-4.0	-3.919	-3.919
2	x_1^8	3.0	2.963	2.963
3	x_2	-0.00391	-0.004023	-0.004023

Table 7.27. *Comparison of coefficients for the denominator terms for* $\dot{x}_2$.

S.No	Candidate Term	Original Coeff.	Min	Max
1	x_1^8	1.0	1.0	1.0
2	x_2	0.0	0.0004261	0.0004261
3	1	0.00390625	0.003882	0.003882

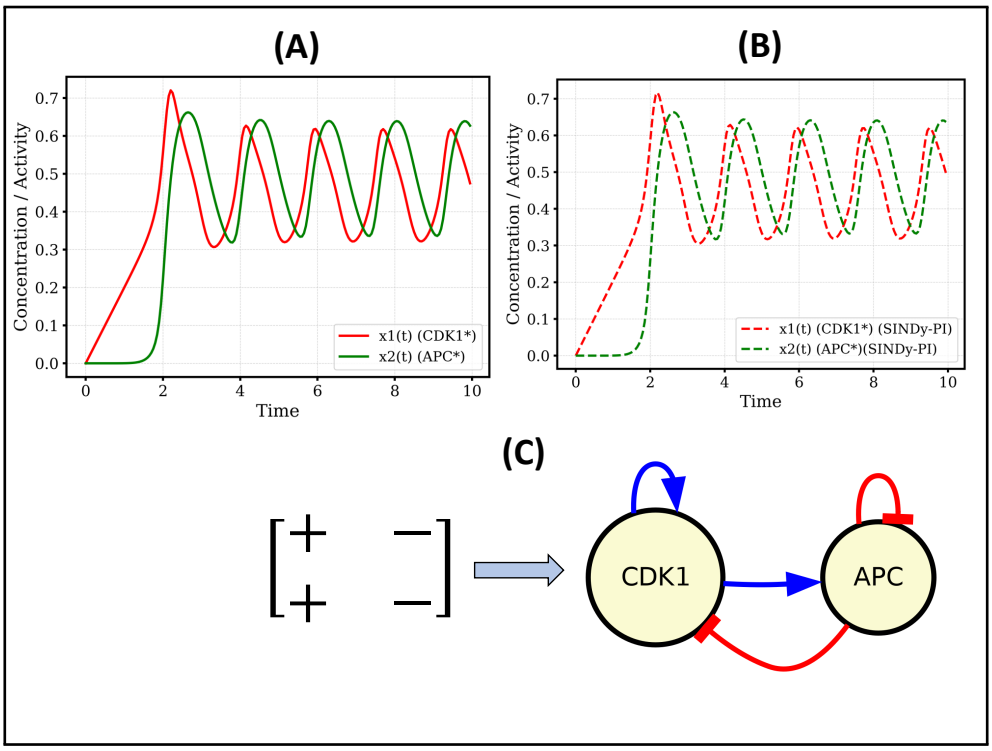

Figure 7.12. *Results from the Implicit SINDy (SINDy-PI) method applied to the CDK1-APC regulation model with a positive feedback. (A) and (B) show the original and SINDy simulated time series of the CDK1-APC regulation model for the variable x_1 and x_2 respectively. (C) represents the sign of the Jacobian evaluated at the steady state $(x_1^*, x_2^*) = (0.464, 0.511)$, along with the resulting interaction network forthe identifiedd model. Blue arrows indicate positive regulation (activation), and red arrows indicate negative regulation (inhibition).*

The candidate models for individual equations result in two pairs of candidate models for the entire system containing 4 extra terms for each. Similar to the previous example, we determined the steady states and the Jacobian for both pairs to determine the interaction network. Both combinations gave the steady state, $\approx (0.464, 0.511)$. Also, for each case, the Jacobian exhibits the same structure of signs shown in **Figure 7.12**.

Example 4: A three-ODE model of CDK1–Plk1-APC regulation:

While the addition of positive feedback in the previous example enabled sustained oscillations through a bistable switching mechanism, Ferrell et al. [20] also demonstrated that such oscillatory behavior can emerge solely from a negative feedback loop, provided the loop is sufficiently long and nonlinear. To explore this design principle, we now consider a three-variable model in which an intermediate species, Polo-like kinase 1 (Plk1), serves as a delay between CDK1 and APC. The reaction network and the governing equations for this system are

(I) Reaction network:

R1: $\varnothing \xrightarrow{\alpha_1} \mathrm{CDK1}^*$, Basal production of CDK1.

R2: $\mathrm{CDK1}^* + \mathrm{H}^+(\mathrm{APC}^*, K_1, n_1) \xrightarrow{\beta_1} \mathrm{CDK1}_{ina}$, Inactivation of CDK1 via APC.

R3: $\mathrm{Plk1}_{ina} + \mathrm{H}^+(\mathrm{CDK1}^*, K_2, n_2) \xrightarrow{\alpha_2} \mathrm{Plk1}^*$, Activation of Plk1 by CDK1.

R4: $\mathrm{Plk1}^* \xrightarrow{\beta_2} \varnothing$, Linear degradation of Plk1.

R5: $\mathrm{APC}_{ina} + \mathrm{H}^+(\mathrm{Plk1}^*, K_3, n_3) \xrightarrow{\alpha_3} \mathrm{APC}^*$, Activation of APC by Plk1.

R6: $\mathrm{APC}^* \xrightarrow{\beta_3} \varnothing$, Linear degradation of APC.

(II) System dynamics:

$$\frac{d[\mathrm{CDK1}^*]}{dt} = \alpha_1 \; - \; \beta_1 \, \mathrm{CDK1}^* \, \frac{\mathrm{APC}^{*\,n_1}}{K_1^{n_1} + \mathrm{APC}^{*\,n_1}}, \tag{7.1.20}$$

$$\frac{d[\mathrm{Plk1}^*]}{dt} = \alpha_2 \, (1 - \mathrm{Plk1}^*) \, \frac{\mathrm{CDK1}^{*\,n_2}}{K_2^{n_2} + \mathrm{CDK1}^{*\,n_2}} \; - \; \beta_2 \, \mathrm{Plk1}^*, \tag{7.1.21}$$

$$\frac{d[\mathrm{APC}^*]}{dt} = \alpha_3 \, (1 - \mathrm{APC}^*) \, \frac{\mathrm{Plk1}^{*\,n_3}}{K_3^{n_3} + \mathrm{Plk1}^{*\,n_3}} \; - \; \beta_3 \, \mathrm{APC}^*, \tag{7.1.22}$$

The variables $\mathrm{CDK1}^*$, $\mathrm{Plk1}^*$, and APC^* represent the active forms of CDK1, Plk1, and APC, respectively. Their corresponding inactive forms are represented as $\mathrm{CDK1}_{\mathrm{ina}}$, $\mathrm{Plk1}_{\mathrm{ina}}$, and $\mathrm{APC}_{\mathrm{ina}}$. Also, they satisfy the conservation relations such that $\mathrm{CDK1}^* + \mathrm{CDK1}_{\mathrm{ina}} = \mathrm{CDK1}_{\mathrm{T}}$, $\mathrm{Plk1}^* + \mathrm{Plk1}_{\mathrm{ina}} = \mathrm{Plk1}_{\mathrm{T}}$, and $\mathrm{APC}^* + \mathrm{APC}_{\mathrm{ina}} = \mathrm{APC}_{\mathrm{T}}$. We take $\mathrm{CDK1}_{\mathrm{T}} = \mathrm{Plk1}_{\mathrm{T}} = \mathrm{APC}_{\mathrm{T}} = 1$.

The delay introduced by the intermediate Plk1 allows the system to exhibit sustained oscillations even without explicit positive feedback (**Fig. 7.13**).

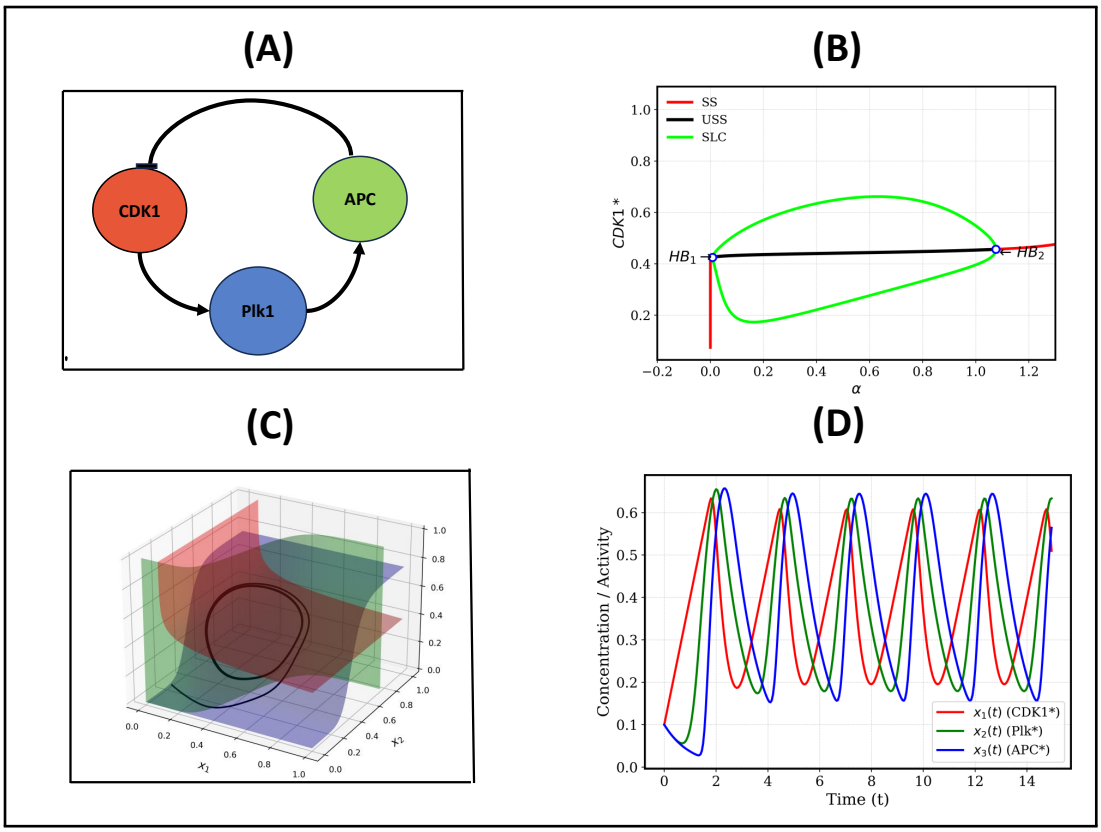

Figure 7.13. *Circuit diagram, one-parameter bifurcation diagram, phase diagram, and time series for a three-ODE model of CDK1, Plk1, and APC regulation. (A) A circuit diagram of the model.(B) A plot of steady state $CDK1$ as a function of bifurcation parameter α_1. Red and black curve denote stable and unstable steady states respectively. The kinetic parameters to simulate the ODE are $\alpha_2 = \alpha_3 = 3$, $\beta_1 = 3$, $\beta_2 = 1$, $K_1 = K_2 = K_3 = 0.5$, and $n_1 = n_2 = n_3 = 8$. HB1 and HB2 are Hopf bifurcations. The two Hopf bifurcations occur near $\alpha = 0.008$ and $\alpha = 1.076$. Near $\alpha = 0.008$, a stable steady state (red curve) loses its stability (black curve) and gives rise to a stable limit cycle (green curve) while near $\alpha = 1.076$ the stable limit cycle disappears, and the steady state becomes stable again. This is the case of supercritical Hopf, as the periodic orbit surrounds unstable steady states. (C) Phase diagram of the system. The black arrows represent direction fields. The light red, blue and green surfaces represent the null surfaces of the system. A black trajectory is shown which starts from $(0.1, 0.1, 0.1)$ and converges to a limit cycle. (D) Time series of the system, starting with $\text{CDK1}^* = 0.1$, $\text{Plk}^* = 0.1$, $\text{APC}^* = 0.1$, showing sustained oscillations.*

Table 7.28. *Parameter values for the three-ODE model of CDK1, Plk1, and APC Regulation.*

Parameter	α_1	α_2	α_3	β_1	β_2	β_3	K_1	K_2	K_3	n_1	n_2	n_3
Value	0.3	3	3	3	1	1	0.5	0.5	0.5	8	8	8

Table 7.29. *Settings used for simulation of CDK1-Plk1–APC model.*

Setting	N_{IC}	t_0	t_f	Δt
Value	1	0	50	0.05

we let $x_1 := \text{CDK1}^*, \quad x_2 := \text{Plk1}^*, \quad x_3 := \text{APC}^*$. The equations then become

$$\boxed{\begin{aligned}
\dot{x}_1 &= \alpha_1 - \beta_1\, x_1\, \frac{x_3^{n_1}}{K_1^{n_1} + x_3^{n_1}}, \\
\dot{x}_2 &= \alpha_2\, \big(1 - x_2\big)\, \frac{x_1^{n_2}}{K_2^{n_2} + x_1^{n_2}} - \beta_2\, x_2, \\
\dot{x}_3 &= \alpha_3\, \big(1 - x_3\big)\, \frac{x_2^{n_3}}{K_3^{n_3} + x_2^{n_3}} - \beta_3\, x_3.
\end{aligned}}$$

Substituting the parameters from **Table 7.28** and simplifying to pure rational form, we get:

$$\dot{x}_1 = \frac{-3.0\, x_1 x_3^8 + 0.3\, x_3^8 + 0.001171875}{1.0\, x_3^8 + 0.00390625} \tag{7.1.19}$$

$$\dot{x}_2 = \frac{-4.0\, x_1^8 x_2 + 3.0\, x_1^8 - 0.00390625\, x_2}{1.0\, x_1^8 + 0.00390625} \tag{7.1.20}$$

$$\dot{x}_3 = \frac{-4.0\, x_2^8 x_3 + 3.0\, x_2^8 - 0.00390625\, x_3}{1.0\, x_2^8 + 0.00390625} \tag{7.1.21}$$

To recover the dynamics of each state variable independently, we choose the following libraries and apply the SINDy-PI algorithm to each:

$$\Theta_1\big(x, \dot{x}_1\big) = \big[\ 1,\ x_3^8,\ x_1 x_3^8,\ \dot{x}_1,\ \dot{x}_1 x_3^8,\ x_1 \dot{x}_1 x_3^8\ \big]$$

$$\Theta_2\big(x, \dot{x}_2\big) = \big[\ x_1^8,\ x_2,\ x_1^8 x_2,\ \dot{x}_2,\ x_1^8 \dot{x}_2,\ x_2 \dot{x}_2,\ x_1^8 x_2 \dot{x}_2\ \big]$$

$$\Theta_3\big(x, \dot{x}_3\big) = \big[\ x_2^8,\ x_2^8 x_3,\ x_3,\ \dot{x}_3,\ x_2^8 \dot{x}_3,\ x_2^8 x_3 \dot{x}_3,\ x_3 \dot{x}_3\ \big]$$

we conducted a threshold grid search for each equation independently. The optimal thresholds and associated metrics for x_1, x_2 and x_3 are summarized in the following tables:

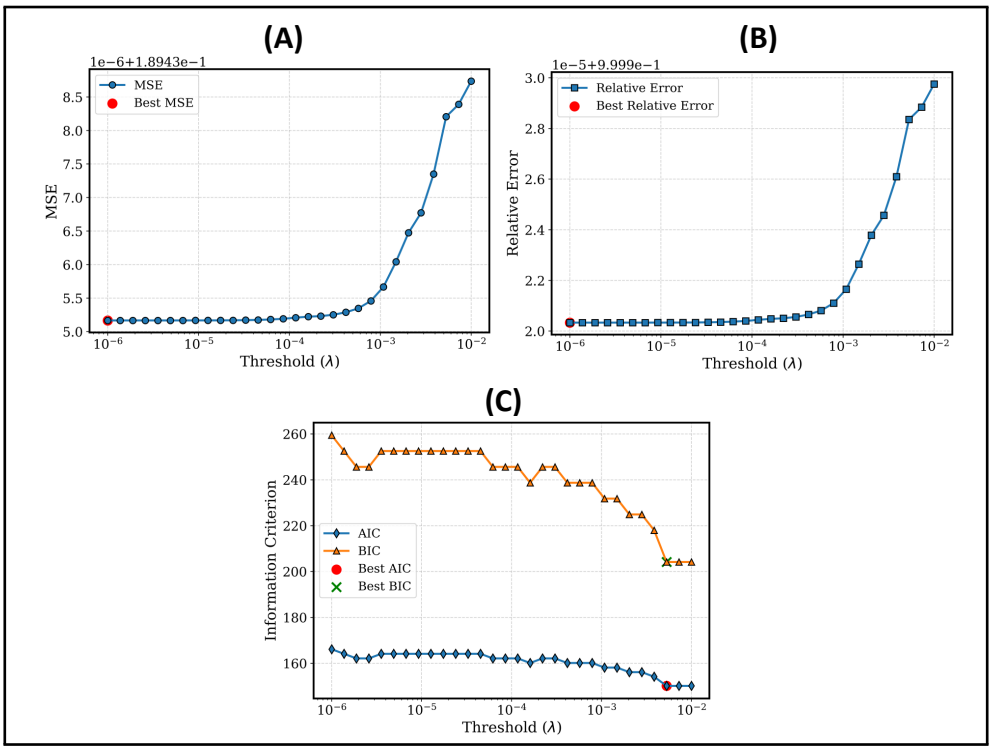

Figure 7.14. *Global diagnostic plots for determining the sparsity threshold for x_1 in CDK1-Plk-APC regulation model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated at 30 points logarithmically spaced between 10^{-6} to 10^{-2}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{1001\times 6}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and information criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 7.30****.*

Table 7.30. *Optimal Value of Threshold (λ) for Each Metric ($\dot{x}_1$)*

Metric	Best Threshold
MSE	1.0×10^{-6}
Relative Error	1.0×10^{-6}
AIC	0.005298
BIC	0.005298

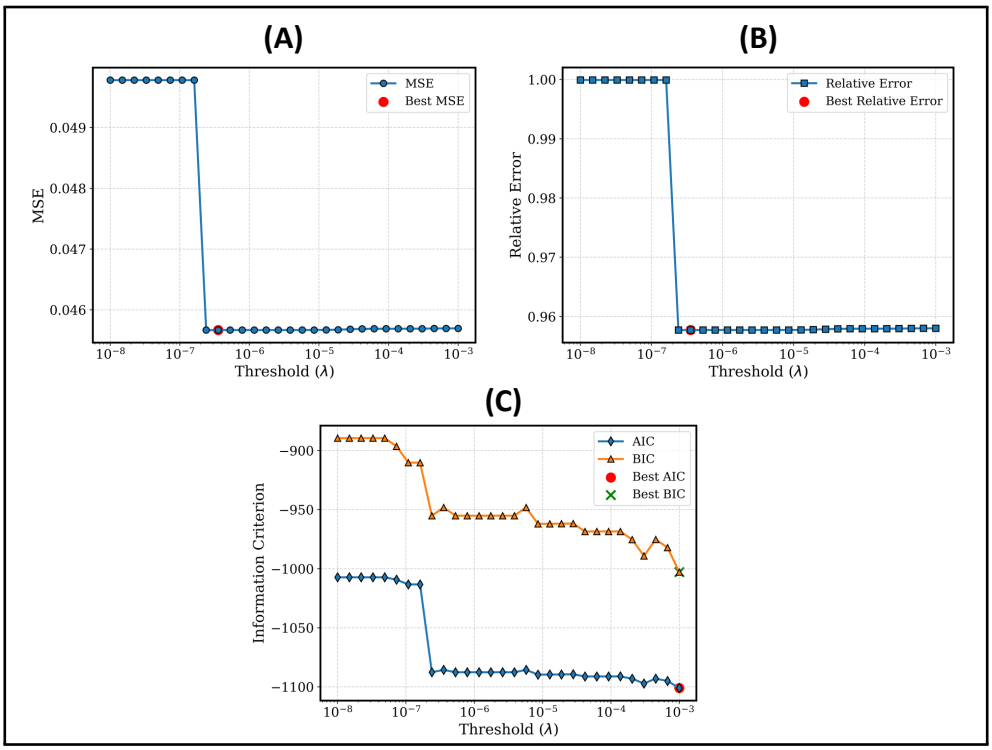

Figure 7.15. *Global diagnostic plots for determining the sparsity threshold for x_2 in CDK1-Plk-APC regulation model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated at 30 points logarithmically spaced between 10^{-8} to 10^{-3}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{1001\times 7}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and information criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 7.31****.*

Table 7.31. *Optimal Value of Threshold (λ) for Each Metric ($\dot{x}_2$)*

Metric	Best Threshold
MSE	3.562×10^{-7}
Relative Error	3.562×10^{-7}
AIC	0.001
BIC	0.001

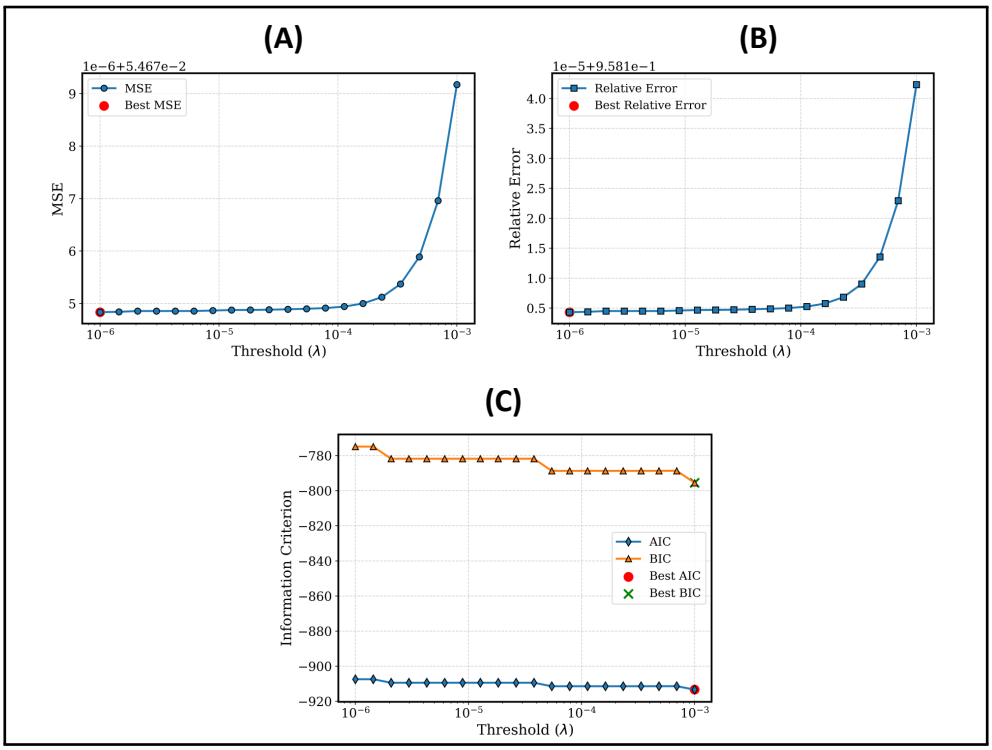

Figure 7.16. *Global diagnostic plots for determining the sparsity threshold for x_3 in CDK1-Plk-APC regulation model. (A) Mean Squared Error (MSE), (B) Relative Error, and (C) Information Criteria (AIC and BIC) are plotted as functions of the sparsity threshold λ, evaluated at 20 points logarithmically spaced between 10^{-6} to 10^{-3}. The implicit feature matrix $Y = \Theta(\mathbf{X}, \dot{\mathbf{X}}_1) \in \mathbb{R}^{1001\times 7}$ is approximated by $\hat{Y} = \Theta\Xi$, and model performance is assessed by residual norms and complexity-penalizing criteria. Optimal thresholds minimizing each metric are indicated with colored markers. The corresponding values are presented in* ***Table 7.32****.*

Table 7.32. *Optimal Value of Threshold (λ) for Each Metric ($\dot{x}_3$)*

Metric	Best Threshold
MSE	1.0×10^{-6}
Relative Error	1.0×10^{-6}
AIC	0.001
BIC	0.001

Table 7.33. *Implicit equations for x_1 sorted by lowest Relative Error corresponding to $\lambda = 1 \times 10^{-6}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
2	$x_1 x_3^8$	9.999 93	$2.284\,40\times10^{-3}$	−2.340 06	−2.337 61
4	$\dot{x}_1 x_3^8$	9.999 76	$4.331\,20\times10^{-3}$	−2.111 72	−2.109 75
1	x_3^8	9.999 31	$7.039\,60\times10^{-3}$	−1.889 70	−1.887 25
5	$x_1\,\dot{x}_1\,x_3^8$	9.958 30	$5.752\,90\times10^{-2}$	−1.793 99	−1.791 54
3	$\dot{x}_1$	−2.970 00	1.000 00	−1.992 61	−1.992 61
0	1	0.000 000	1.000 00	0.0000	0.0000

Table 7.34. *Implicit equations for x_2 sorted by lowest Relative Error corresponding to $\lambda = 3.562 \times 10^{-7}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
0	x_1^8	9.999 99	$6.146\,10\times10^{-4}$	−2.501 10	−2.498 64
2	$x_1^8 x_2$	9.999 99	$8.232\,50\times10^{-4}$	−2.558 09	−2.555 63
4	$x_1^8\,\dot{x}_2$	9.999 78	$4.157\,30\times10^{-3}$	−2.263 39	−2.260 45
6	$x_1^8 x_2\,\dot{x}_2$	9.981 58	$3.878\,10\times10^{-2}$	−1.943 98	−1.941 52
5	$x_2\,\dot{x}_2$	9.731 28	$1.639\,30\times10^{-1}$	−7.129 84	−7.095 48
3	$\dot{x}_2$	−2.200 00	1.000 00	−1.848 84	−1.848 84
1	x_2	−5.382 55	1.000 00	−1.827 20	−1.827 20

Table 7.35. *Implicit equations for x_3 sorted by lowest Relative Error corresponding to $\lambda = 1 \times 10^{-6}$.*

Eq	Feature	R^2	Relative Error	AIC	BIC
0	x_2^8	9.999 99	$8.599\,60\times10^{-4}$	−2.358 23	−2.355 78
1	$x_2^8 x_3$	9.999 98	$1.189\,70\times10^{-3}$	−2.414 84	−2.412 39
4	$x_2^8\,\dot{x}_3$	9.999 87	$3.257\,80\times10^{-3}$	−2.176 53	−2.173 59
5	$x_2^8 x_3\,\dot{x}_3$	9.973 84	$4.666\,80\times10^{-2}$	−1.784 82	−1.782 37
6	$x_3\,\dot{x}_3$	9.734 82	$1.628\,40\times10^{-1}$	−6.977 68	−6.948 23
2	x_3	−4.735 02	1.000 00	−1.736 84	−1.736 84
3	$\dot{x}_3$	−2.000 00	1.000 00	−1.584 31	−1.584 31

The choice of optimal thresholds $\lambda = 1.0 \times 10^{-6}$ for x_1, $\lambda = 3.562 \times 10^{-7}$ for x_2, and $\lambda = 1.0 \times 10^{-6}$, results in the following candidate models:

Candidate models for x_1:

$$\text{Model 1:} \quad \dot{x}_1 = \frac{-3.138\, x_1 x_3^8 \;+\; 0.3187\, x_3^8 \;+\; 0.001227}{0.1055\, x_1 x_3^8 \;+\; 1.0\, x_3^8 \;+\; 0.004132},$$

$$\text{Model 2:} \quad \dot{x}_1 = \frac{-3.026\, x_1 x_3^8 \;+\; 0.299\, x_3^8 \;+\; 0.001211}{0.02599\, x_1 x_3^8 \;+\; 1.0\, x_3^8 \;+\; 0.004077},$$

$$\text{Model 4:} \quad \dot{x}_1 = \frac{-2.987\, x_1 x_3^8 \;+\; 0.292\, x_3^8 \;+\; 0.001203}{1.0\, x_3^8 \;+\; 0.004049}.$$

Candidate models for x_2:

$$\text{Model 0:} \quad \dot{x}_2 = \frac{-4.031\, x_1^8 x_2 \;+\; 3.019\, x_1^8 \;-\; 0.00395\, x_2}{1.0\, x_1^8 \;+\; 0.0001882\, x_2 \;+\; 0.003896},$$

$$\text{Model 2:} \quad \dot{x}_2 = \frac{-4.035\, x_1^8 x_2 \;+\; 3.021\, x_1^8 \;-\; 0.003951\, x_2}{1.0\, x_1^8 \;+\; 0.0001796\, x_2 \;+\; 0.003903},$$

$$\text{Model 5:} \quad \dot{x}_2 = \frac{-4.088\, x_1^8 x_2 \;+\; 3.068\, x_1^8 \;-\; 0.004095\, x_2}{0.01309\, x_1^8 x_2 \;+\; 1.0\, x_1^8 \;+\; 0.000948\, x_2 \;+\; 0.003724}.$$

Candidate models for x_3:

$$\text{Model 0:} \quad \dot{x}_3 = \frac{-3.979\, x_2^8 x_3 \;+\; 2.993\, x_2^8 \;-\; 0.003982\, x_3}{1.0\, x_2^8 \;+\; 0.0001612\, x_3 \;+\; 0.003944},$$

$$\text{Model 1:} \quad \dot{x}_3 = \frac{-3.982\, x_2^8 x_3 \;+\; 2.994\, x_2^8 \;-\; 0.003976\, x_3}{1.0\, x_2^8 \;+\; 0.0001173\, x_3 \;+\; 0.003957}.$$

Table 7.36. *Comparison of coefficients for the numerator terms for $\dot{x}_1$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1 x_3^8$	-3.0	-3.138	-2.987
2	x_3^8	0.3	0.292	0.3187
3	1	0.001171875	0.001203	0.001227

Table 7.37. *Comparison of coefficients for the denominator terms for $\dot{x}_1$.*

S.No	Candidate Term	Original Coeff.	Min	Max
1	$x_1 x_3^8$	0	0	0.1055
2	x_3^8	1.0	1.0	1.0
3	1	0.00390625	0.004049	0.004132

Table 7.38. *Comparison of coefficients for the numerator terms for $\dot{x}_2$.*

S.No	Candidate Term	Original	Min	Max
1	$x_1^8 x_2$	-4.0	-4.088	-4.031
2	x_1^8	3.0	3.019	3.068
3	x_2	-0.00390625	-0.004095	-0.00395

Table 7.39. *Comparison of coefficients for the denominator terms for $\dot{x}_2$.*

S.No	Candidate Term	Original	Min	Max
1	$x_1^8 x_2$	0	0	0.01309
2	x_1^8	1.0	1.0	1.0
3	x_2	0	0.0001796	0.000948
4	1	0.00390625	0.003724	0.003903

Table 7.40. *Comparison of coefficients for the numerator terms for $\dot{x}_3$.*

S.No	Candidate Term	Original	Min	Max
1	$x_2^8 x_3$	-4.0	-3.982	-3.979
2	x_2^8	3.0	2.993	2.994
3	x_3	-0.00390625	-0.003982	-0.003976

Table 7.41. *Comparison of coefficients for the denominator terms for $\dot{x}_3$.*

S.No	Candidate Term	Original	Min	Max
1	x_2^8	1.0	1.0	1.0
2	x_3	0	0.0001173	0.0001612
3	1	0.00390625	0.003944	0.003957

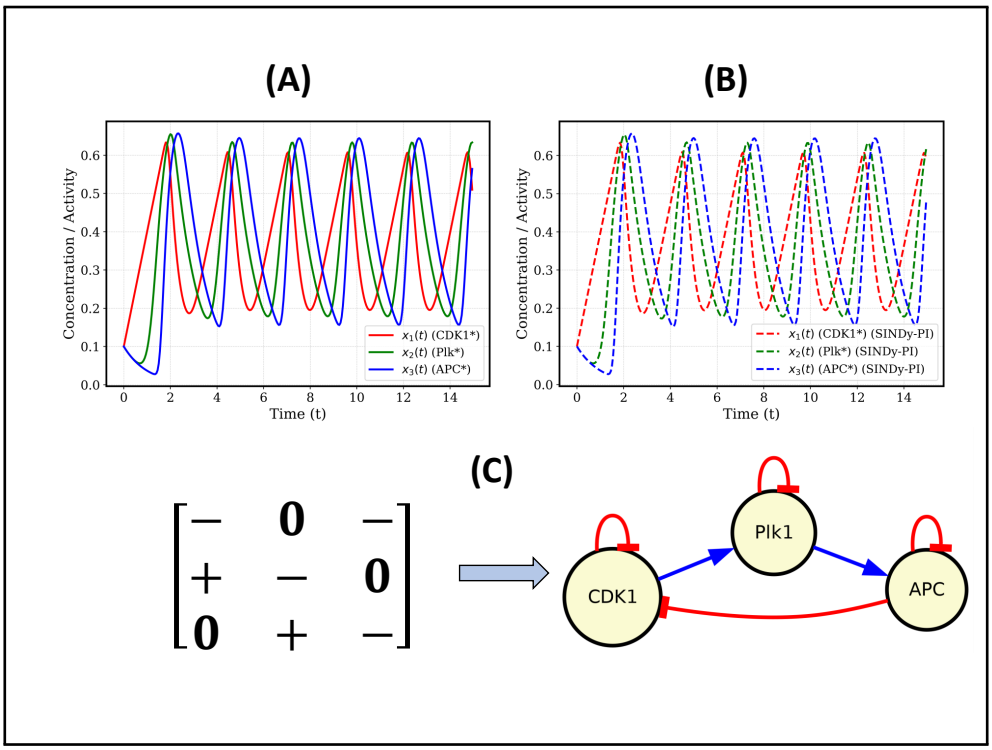

Figure 7.17. *Results from the Implicit SINDy (SINDy-PI) method applied to three ODE CDK1-Plk1-APC regulation model. (A) and (B) show the original and SINDy simulated time series of the CDK1-Plk1-APC regulation model for the variables x_1, x_2, and x_3 respectively. (C) represents the sign of the Jacobian evaluated at the steady state (x_1^*, x_2^*, x_3^*)= $(0.438, 0.436, 0.429)$, along with the resulting interaction network for identified models. Blue arrows indicate positive regulation (activation), and red arrows indicate negative regulation (inhibition). All the Jacobian corresponding to any combination of models exhibits the same steady state and sign structure, resulting into same interaction network.*

7.2 Sensitivity Analysis

In order to check that our data-driven reconstructions capture not only the algebraic form but also the underlying control dynamics of CDK1 regulatory networks, we performed a unified time-course sensitivity analysis across the four examples discussed above using COPASI (COmplex PAthway SImulator) [26].

For each model, we performed and compared the sensitivity for three versions:

- The original formulation involving mass action and Hill-type terms explicitly.
- Its algebraically equivalent pure rational form.
- The explicit rational SINDy-PI reconstruction.

For SINDy-PI reconstructed models, we selected the model involving a greater number of terms and then assigned the parameter values as the mean of the maximum and minimum across all valid and identified models for each case.

Below, we have shown the form of the models and scaled sensitivity plots for each of the four models for identifying the dominant parameters that exert the strongest positive or negative control over system behavior and compared which parameters exert the strongest positive or negative control on steady-state levels across original, rational, and data-driven models.

7.2.1 Sensitivity Analysis for one-variable model of CDK1 regulation

Table 7.42. *Comparison of CDK1 dynamics in one-variable model of CDK1 regulation: RHS for the original form (top), and the corresponding rational and SINDy-identified models (bottom). The parameter notation in both rational and SINDy-identified models has been taken to be the same for convenience, but that does not mean that they are exactly equal. The notation e_1 stands for extra terms. The scaled sensitivity analysis in each case is shown in* ***Figure 7.18***

.

Original Form	
$\alpha_1 - \beta_1\,\mathrm{CDK1} + \alpha_3(1-\mathrm{CDK1})\cdot\dfrac{\mathrm{CDK1}^{n_3}}{\mathrm{CDK1}^{n_3}+K_3^{n_3}}$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-a_1x^9+b_1x^8-c_1x+d_1}{x^8+f_1}$	$\dfrac{-a_1s^9+b_1s^8-c_1s+d_1}{s^8+e_1s+f_1}$

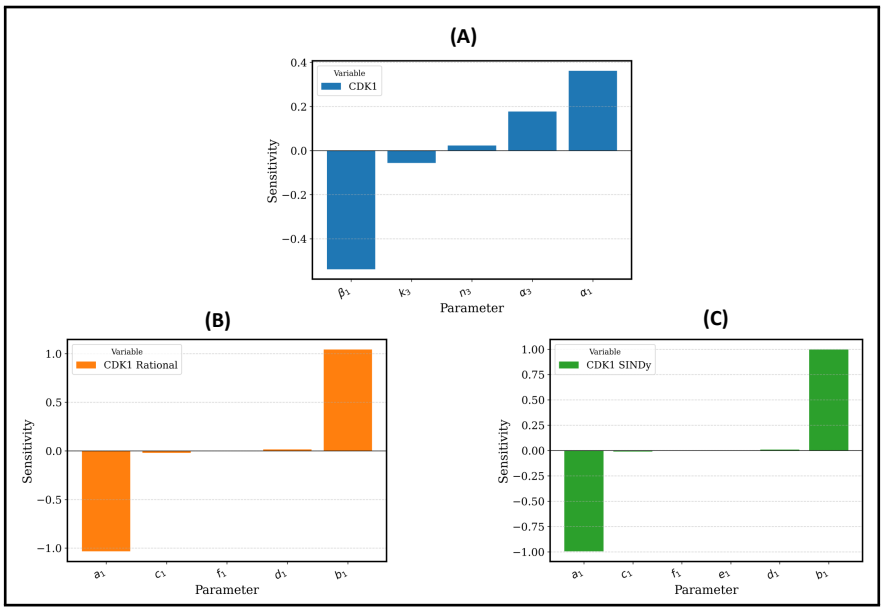

Figure 7.18. *Scaled sensitivities of CDK1 activity to each model parameter in Example 1, for three model formulations. (A) (Blue) Sensitivities computed for the original model. (B) (Orange) Sensitivities computed for the pure rational reformulation. (C) (Green) Sensitivities computed for the SINDy-identified models. The y-axis represents the scaled local sensitivity of the steady-state of CDK1 concentration to each parameter. Positive bars capture the control on activation effect, while negative bars indicate inhibitory control. The rational form of the model shows a different sensitivity pattern from the original due to their structure, but the same rational form and its SINDy reconstruction show a similar kind of control behaviour.*

7.2.2 Sensitivity Analysis for two-variable model of CDK1–APC regulation with negative feedback

Table 7.43. *Comparison of CDK1 dynamics in two-variable model of CDK1–APC regulation with negative feedback: RHS for the original form (top), and the corresponding rational and SINDy-identified approximations (bottom). The parameter notation in both rational and SINDy-identified models has been taken to be the same for convenience, but that does not imply that they are exactly equal. No extra terms are present in this case. The scaled sensitivity analysis in each case is shown in* ***Figure 7.19***

Original Form	
$\alpha_1 - \beta_1 \,\mathrm{CDK1} \cdot \dfrac{\mathrm{APC}^{n_1}}{K_1^{n_1} + \mathrm{APC}^{n_1}}$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-a_1\, x_1\, x_2^8 + b_1\, x_2^8 + c_1}{x_2^8 + d_1}$	$\dfrac{-a_1\, s_1\, s_2^8 + b_1\, s_2^8 + c_1}{s_2^8 + d_1}$

Table 7.44. *Comparison of APC dynamics in two-variable model of CDK1–APC regulation with negative feedback: RHS for the original form (top), and the corresponding rational and SINDy-identified models (bottom). The parameter notation in both rational and SINDy-identified models has been taken to be the same for convenience, but that does not imply that they are exactly equal. Two extra terms, namely e_{11} and e_12, are present in this case. The scaled sensitivity analysis in each case is shown in* ***Figure 7.19***

Original Form	
$\alpha_2\,(1 - \mathrm{APC}) \cdot \dfrac{\mathrm{CDK1}^{n_2}}{K_2^{n_2} + \mathrm{CDK1}^{n_2}} \;-\; \beta_2\,\mathrm{APC}$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-a_2\, x_1^8\, x_2 + b_2\, x_1^8 - c_2\, x_2}{x_1^8 + d_2}$	$\dfrac{-a_2\, s_1^8\, s_2 + b_2\, s_1^8 - c_2\, s_2}{e_{11}\, s_1^8\, s_2 + s_1^8 + e_{12}\, s_2 + d_2}$

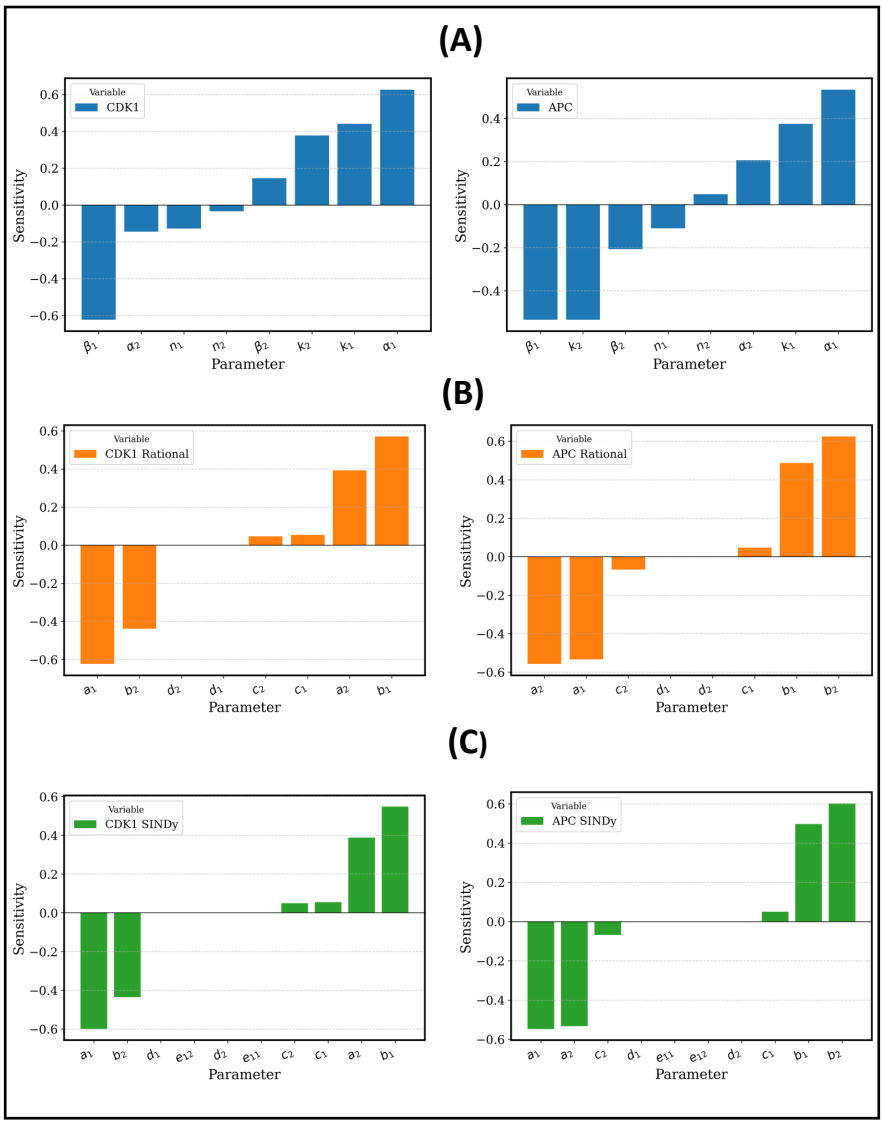

Figure 7.19. *Scaled sensitivities of CDK1-APC activity to each model parameter in Example 2, for three model formulations. Left and right panels in each (A) (Blue), (B) (Orange), and (C) (Green) show the scaled local sensitivity of the steady-state concentrations of CDK1 and APC, respectively, with respect to the kinetic parameters present in the models. Note that despite the increased number of terms in the SINDy reconstructed model, the core control behavior is preserved.*

7.2.3 Sensitivity Analysis for two-variable model of CDK1–APC regulation with positive feedback

Table 7.45. *Comparison of CDK1 dynamics in two-variable model of CDK1–APC regulation with positive feedback: RHS for the original form (top), and the corresponding rational and SINDy-identified models (bottom). The parameter notation in both rational and SINDy-identified models has been taken to be the same for convenience. Three extra terms are present in this case. The scaled sensitivity analysis in each case is shown in* ***Figure 7.20***

Original Form	
$\alpha_1 - \beta_1 \cdot \mathrm{CDK1} \cdot \dfrac{\mathrm{APC}^{n_1}}{K_1^{n_1} + \mathrm{APC}^{n_1}} + \alpha_3 \cdot (1 - \mathrm{CDK1}) \cdot \dfrac{\mathrm{CDK1}^{n_3}}{K_3^{n_3} + \mathrm{CDK1}^{n_3}}$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-a_1 x_1^9 x_2^8 - b_1 x_1^9 + c_1 x_1^8 x_2^8 + d_1 x_1^8 - f_1 x_1 x_2^8 + g_1 x_2^8 + h_1}{x_1^8 x_2^8 + p_1 x_1^8 + q_1 x_2^8 + r_1}$	$\dfrac{-a_1 s_1^9 s_2^8 - b_1 s_1^9 + c_1 s_1^8 s_2^8 + d_1 s_1^8 - f_1 s_1 s_2^8 + g_1 s_2^8 + h_1}{e_{11} s_1^9 s_2^8 + e_{12} s_1^9 + s_1^8 s_2^8 + p_1 s_1^8 + e_{13} s_1 s_2^8 + q_1 s_2^8 + r_1}$

Table 7.46. *Comparison of CDK1 dynamics in two-variable model of CDK1–APC regulation with positive feedback: RHS for the original form (top), and the corresponding rational and SINDy-identified models (bottom). The parameter notation in both rational and SINDy-identified models has been taken to be the same for convenience. Only one extra term is present in this case. The scaled sensitivity analysis in each case is shown in* ***Figure 7.20***

Original Form	
$\alpha_2 (1 - \mathrm{APC}) \dfrac{\mathrm{CDK1}^{n_2}}{K_2^{n_2} + \mathrm{CDK1}^{n_2}} - \beta_2 \mathrm{APC}$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-a_2 x_1^8 x_2 + b_2 x_1^8 - c_2 x_2}{x_1^8 + d_2}$	$\dfrac{-a_2 s_1^8 s_2 + b_2 s_1^8 - c_2 s_2}{s_1^8 + e_2 s_2 + d_2}$

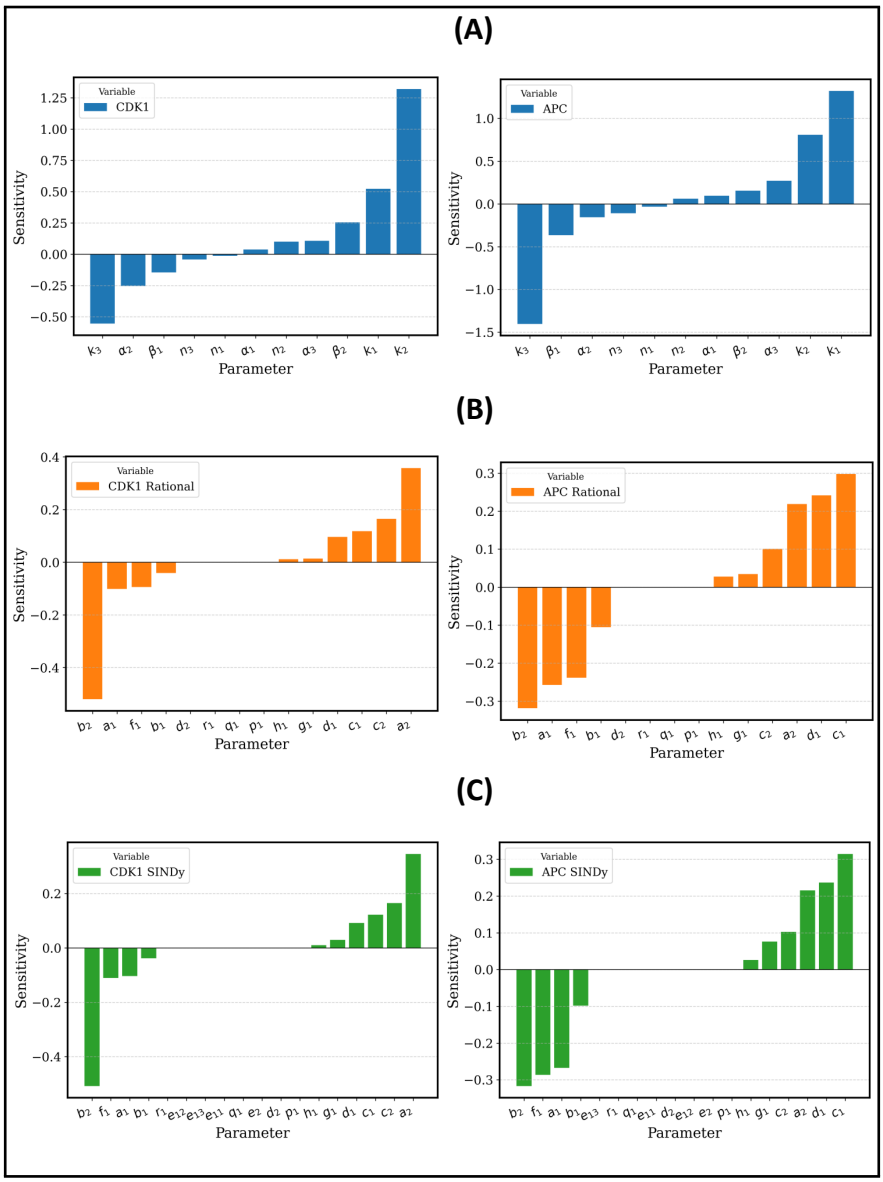

Figure 7.20. *Scaled sensitivities of CDK1-APC activity to each model parameter in Example 3, for three model formulations. Left and right panels in each (A) (Blue), (B) (Orange), and (C) (Green) show the scaled local sensitivity of the steady-state concentrations of CDK1 and APC, respectively, with respect to the kinetic parameters present in the models. While the SINDy-identified models include additional terms, their sensitivity profiles closely reflect the key regulatory influences captured in the original(rational models).*

7.2.4 Sensitivity Analysis for three-variable model of CDK1–Plk1-APC regulation

Table 7.47. *Comparison of CDK1 dynamics in three-variable model of CDK1-Plk1–APC regulation with positive feedback: RHS for the original form (top), and the corresponding rational and SINDy-identified models (bottom). The parameter notation in both rational and SINDy-identified models has been taken the same for convenience. The number of extra terms present in this case is one. The scaled sensitivity analysis in each case is shown in* ***Figure 7.21***

.

Original Form	
$\alpha_1 - \beta_1 \,\mathrm{CDK1}\left(\dfrac{\mathrm{APC}^{n_1}}{K_1^{n_1} + \mathrm{APC}^{n_1}}\right)$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-a_1\, x_1\, x_3^8 + b_1\, x_3^8 + c_1}{x_3^8 + d_1}$	$\dfrac{-a_1\, s_1\, s_3^8 + b_1\, s_3^8 + c_1}{e_1\, s_1\, s_3^8 + s_3^8 + d_1}$

Table 7.48. *Comparison of Plk1 dynamics in three-variable model of CDK1-Plk1–APC regulation with positive feedback: RHS for the original form (top), and the corresponding rational and SINDy-identified models (bottom). The parameter notation in both rational and SINDy-identified models has been taken to be the same for convenience. Two extra terms are present in this case. The scaled sensitivity analysis in each case is shown in* ***Figure 7.21***

.

Original Form	
$\alpha_2\,(1 - \mathrm{Plk1})\,\dfrac{\mathrm{CDK1}^{n_2}}{K_2^{n_2} + \mathrm{CDK1}^{n_2}} - \beta_2\,\mathrm{Plk1}$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-a_2\, x_1^8\, x_2 + b_2\, x_1^8 - c_2\, x_2}{x_1^8 + d_2}$	$\dfrac{-a_2\, s_1^8\, s_2 + b_2\, s_1^8 - c_2\, s_2}{e_{21}\, s_1^8\, s_2 + s_1^8 + e_{22}\, s_2 + d_2}$

Table 7.49. *Comparison of APC dynamics in three-variable model of CDK1-Plk1–APC regulation with positive feedback: RHS for-the original form (top), and the corresponding rational and SINDy-identified models (bottom). The parameter notation in both rational and SINDy-identified models has been taken to be the same for convenience. Only one extra term is present in this case. The scaled sensitivity analysis in each case is shown in* ***Figure 7.21***

.

Original Form	
$\alpha_3\,(1-\mathrm{APC})\,\dfrac{\mathrm{Plk1}^{n_3}}{K_3^{n_3}+\mathrm{Plk1}^{n_3}} \;-\; \beta_3\,\mathrm{APC}$	
Rational Reformulation	**SINDy-Identified Model**
$\dfrac{-\,a_3\,x_2^8\,x_3+\,b_3\,x_2^8-\,c_3\,x_3}{x_2^8+d_3}$	$\dfrac{-\,a_3\,s_2^8\,s_3+\,b_3\,s_2^8-\,c_3\,s_3}{s_2^8+e_3\,s_3+d_3}$

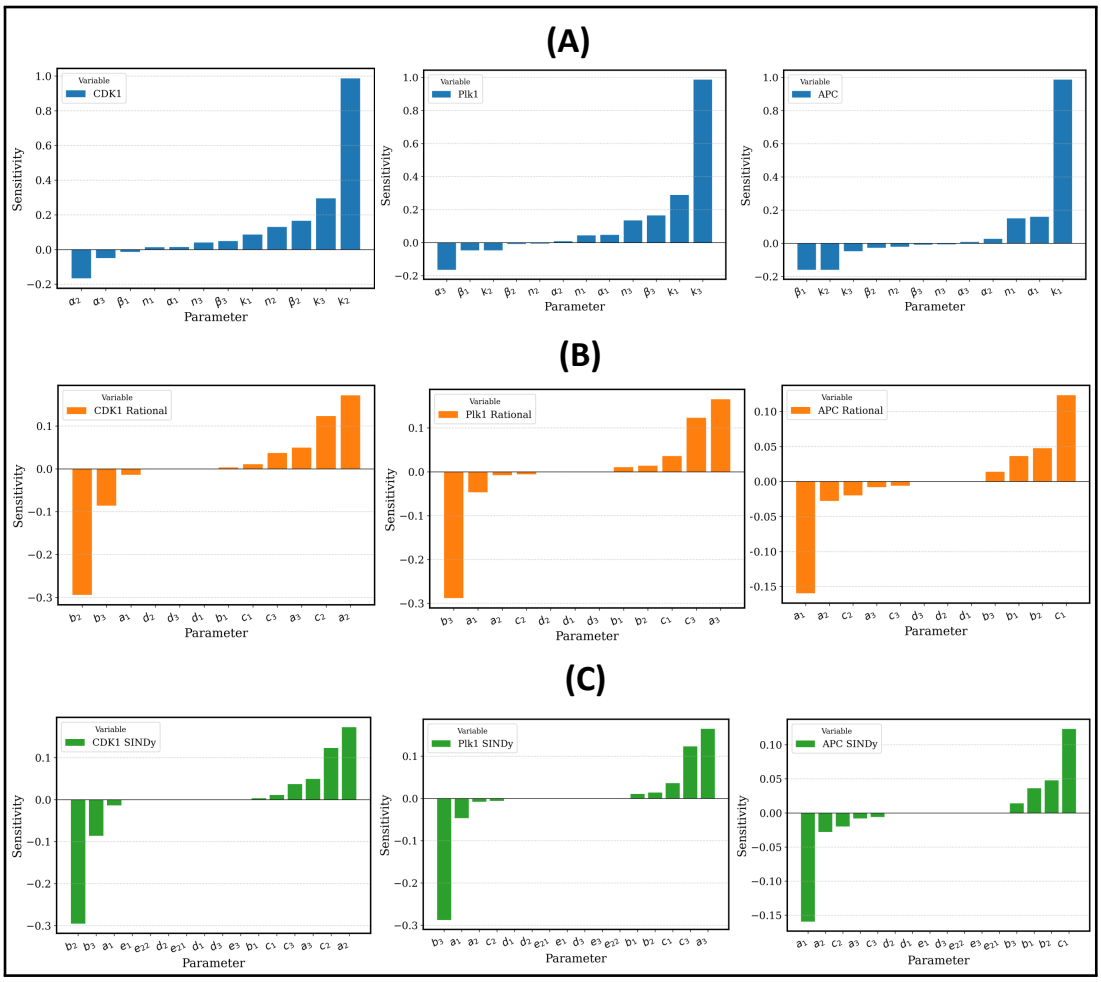

Figure 7.21. *Scaled sensitivities of CDK1-APC activity to each model parameter in Example 4, for three model formulations. Left, middle and right panels in each (A) (Blue), (B) (Orange), and (C) (Green) show the scaled local sensitivity of the steady-state concentrations of CDK1, Plk1 and APC, respectively, with respect to the kinetic parameters present in the models. Again, the dominant regulatory influences are the same across both models, indicating that control behavior is largely preserved in the identified models.*

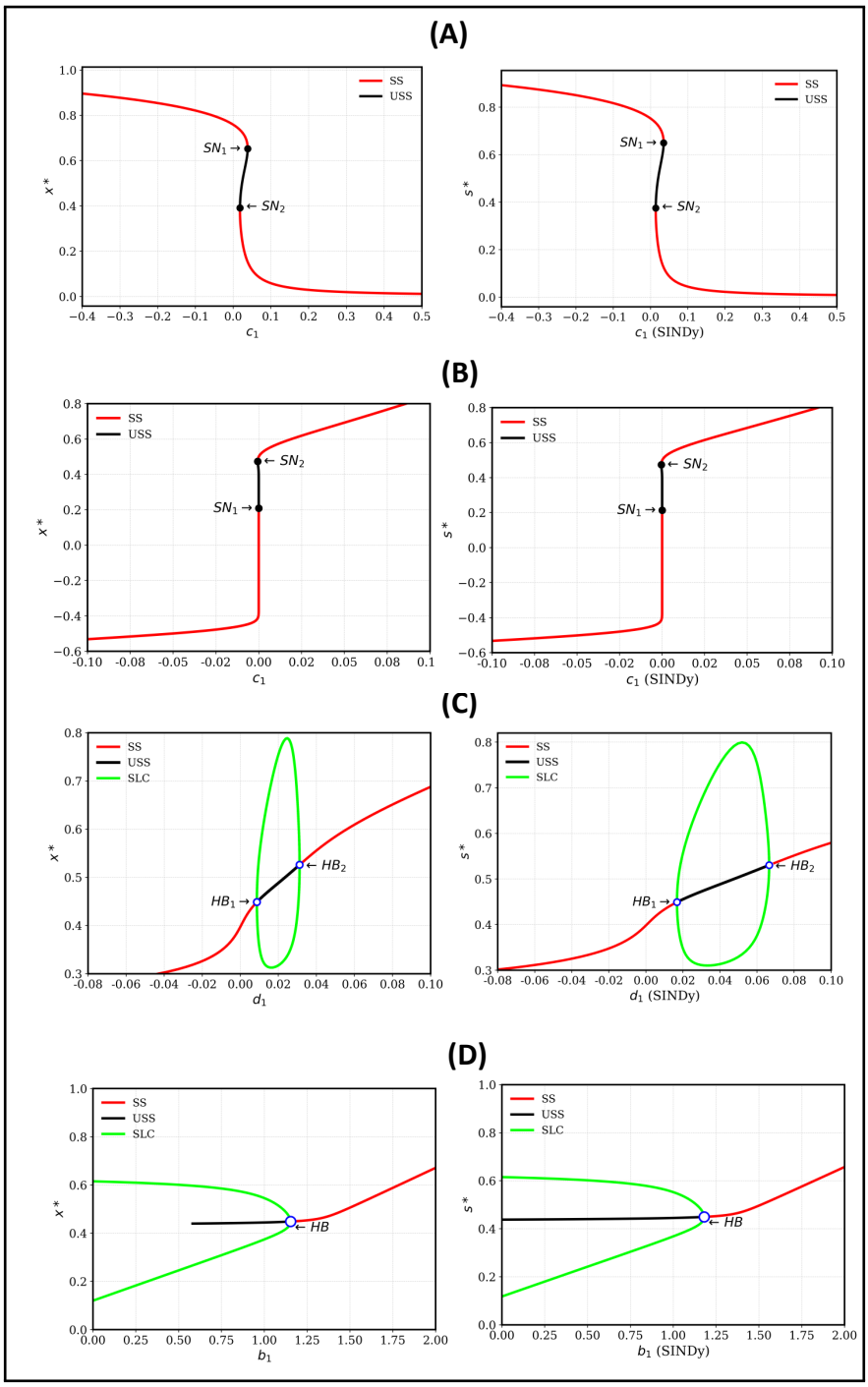

Figure 7.22. *Comparison of codimension-one bifurcation diagrams from original models and SINDy-inferred models. The left and right panels in each of (A)-(D) show the bifurcation diagram for the original rational model and its SINDy-recovered model for all four examples, respectively. Note that, in all cases, a similar kind of bifurcation is observed for all SINDy-identified models.*

Discussion

In this chapter, we used the SINDy-PI framework to rediscover the underlying regulatory interaction of CDK1-APC dynamics. We started from known models [20], simulated them for training data, reformulated equations to pure rational form, and built the library for training Implicit SINDy (SINDy-PI). The governing equations obtained in each case were written in explicit rational form only. We compared the minimum and maximum value of the coefficient across all models in their pure rational form to know the range of the learned parameters. We also simulated the time series obtained and determined the steady states from the SINDy-learned equation. In order to know the effect of learned parameters on steady-state concentration of each variable, we also performed local sensitivity analysis. In all cases, despite the presence of extra terms introduced by SINDy models, the dominant regulatory influences largely remained consistent. Also, we compared the bifurcation diagrams obtained for ground truth models (written in rational form) and for the SINDy-identified models. We selected the parameter having less scaled sensitivity for performing bifurcation. All the results obtained so far aligned with the known ground truth in each case. All the codes, sensitivity analysis data, and notebooks associated with this chapter are available at the repository `https://github.com/Alka-CBhub/Chapter-7`.

Limitation

The candidate library in all examples was constructed using terms present in the original equations to make the analysis simple and validate our approach and the tool. The learned models obtained in this way not only matched the behavior of the original equations but were also compact, containing few terms. This shows that **SINDy-PI may effectively reconstruct interpretable rational models without requiring a large basis**.

However, literatures [8, 43, 44] frequently suggests that a single trajectory may not be sufficient to correctly identify the model. Also, the coefficients in the rescaled model were dropped below a tolerance as proposed in the workflow, which may be context dependent and may affect the number of terms in the model. We recovered models back from a single trajectory in our analysis, which was simulated directly from the models. This happened probably due to the candidate library, which was constructed using the known models. In practical situations, where the actual expressions for nonlinearities are uncertain and data may be noisy or partial, the concept of multiple trajectories may become necessary to ensure model identifiability and interpretability.

Also, in order to select interpretable models from a set of candidate models, we assessed the performance of implicit equations only. Many literatures [8, 43] uses the RK-4 method to simulate the explicit form (independent from other variables) to assess their performance. Also, the learned equations were kept in explicit rational form, though they simulate the original behavior, but they are not completely interpretable in the sense that they don't reflect the well-known regulatory terms of the biological dictionary. The reason was the presence of some extra

terms in the denominator, which were not factorizable.

Overall, this study demonstrates that, despite many limitations, SINDy-PI can recover the models and preserve the essential features of the original system. These findings may provide the groundwork to extend the approach for more complex, noisy data and a generalized library.

Chapter 8

CAR T-Cell Therapy: The Role of Mathematical Modelling

8.1 Introduction to CAR T-Cell Therapy

Cancer is one of the most challenging diseases to understand and treat because of its genetic and molecular complexities, as well immune system's limited ability to distinguish malignant cells from healthy tissue. In recent years, adaptive immunotherapy for cancer has emerged as a powerful strategy to overcome this limitation by enabling immune cells to more effectively recognize and eliminate tumor cells. Among the most promising immunotherapies is **C**himeric **A**ntigen **R**eceptor (CAR) T-cell therapy, which involves engineering a patient's own T cells to detect and attack specific tumor-associated antigens. Once reinfused into the body, these modified T cells can actively recognize and bind to cancer cells, leading to their targeted destruction. Because of their ability to persist and expand in response to antigen exposure, CAR T cells are sometimes referred to as a "living drug" [66].

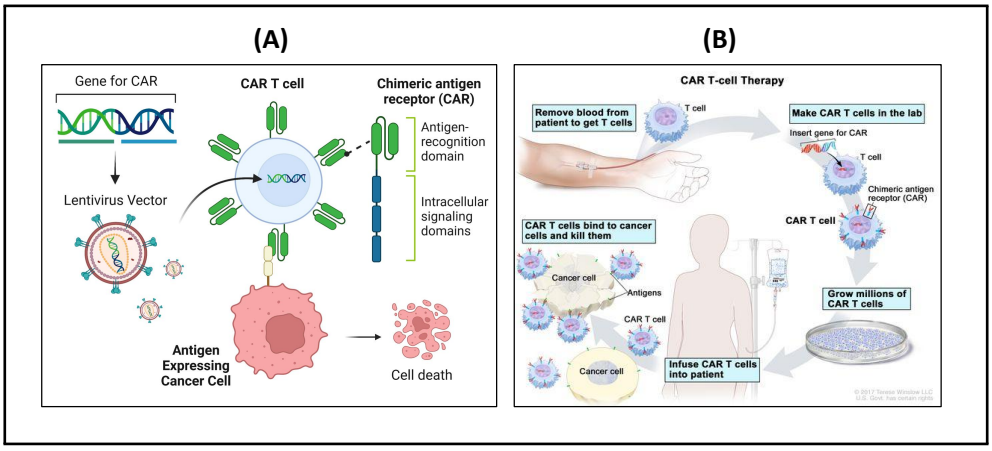

Figure 8.1. *Overview of CAR T cell design and its therapeutic workflow. This figure is adapted from the reference [66]. (A) The structure and mechanism of a chimeric antigen receptor (CAR) T-cell. (B) Clinical workflow of CAR T cell therapy.*

8.1.1 Biological Principles behind CAR T-Cell Function

CAR T-cell therapy is based on the fundamental biology of the immune system, particularly in the specialized role of native T cells. T cells are immune cells that help to detect and eliminate infected or abnormal cells. In their normal form, T lymphocytes identify antigens on cell surfaces via the major histocompatibility complex (MHC). However, many tumor cells avoid detection by downregulating antigen presentation. CAR T-cell therapy addresses this issue by modifying T cells with synthetic receptors that directly attach to tumor-associated antigens, avoiding the necessity for MHC presentation [66, 67].

Structure and Mechanism of CARs

Each chimeric antigen receptor (CAR) is composed of four essential domains for performing specific functions (**Figure 8.2**):

1. **Extracellular antigen-binding domain:** Typically produced from the single-chain variable fragment (scFv) of a monoclonal antibody. This domain provides specificity to the CAR for a particular antigen associated with tumors (e.g., CD19 or BCMA).
2. **A hinge or spacer region:** This provides flexibility and suitable orientation for the antigen-binding domain, enabling successful binding to its target.

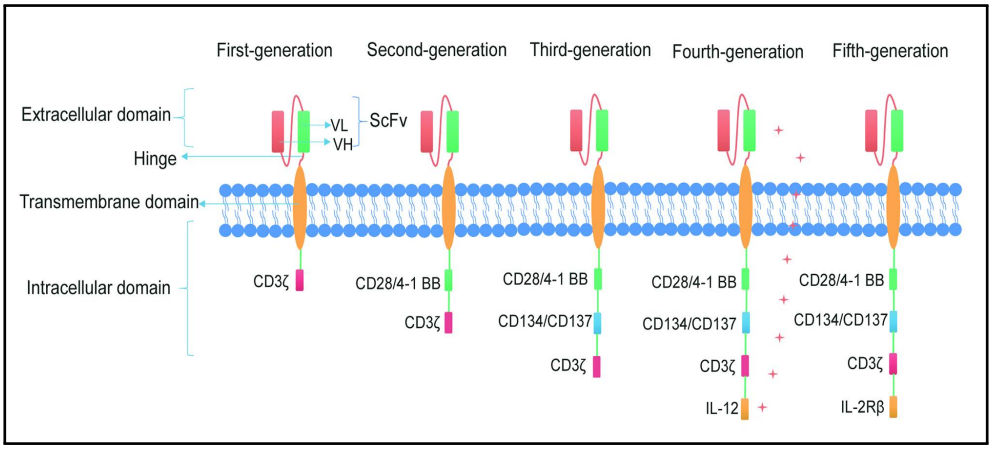

Figure 8.2. *Structural evolution of CAR T cells across five generations. This shows the evolution of CAR T-cell designs from the first to the fifth generation. Each receptor has an external antigen-binding domain (scFv), a hinge and transmembrane region, and an intracellular signaling domain. First-generation CARs comprise just the CD3ζ domain. Second-generation CARs incorporate a single co-stimulatory domain, such as CD28 or 4-1BB. Third-generation CARs integrate two co-stimulatory domains to enhance signaling. Fourth- and fifth-generation CARs incorporate supplementary components, such as cytokine secretion modules (e.g., IL-12 or IL-2Rβ) or transcriptional regulators, to enhance immunological functionality and longevity [68].*

3. **A transmembrane domain:** It links the exterior binding domain to the internal signaling machinery and anchors the receptor within the T cell membrane.

4. **An intracellular signaling domain:** This component initiates T cell activation in response to antigen binding. The structure and composition of this domain defines the CAR's "generation". First-generation CARs just include the CD3ζ domain, which causes basic activation. Second- and third-generation CARs contain one or more co-stimulatory domains, such as CD28 or 4-1BB, that significantly boost T cell growth, persistence, and cytokine production [67, 69].

Signaling and Effector Mechanisms of CAR T Cells

After recognizing a target antigen, CAR T-cells start a complex cascade of activation, proliferation, and cytotoxic activities. The process starts when the scFv domain of the CAR attaches to its target antigen, which results in receptor aggregation at the junction between the T cell and the tumor cell. This establishes an immunological synapse, which is defined as a specific signaling hub whose stability relies on both the antigen density and the affinity of the scFv.

The formation of this synapse activates the phosphorylation of immunoreceptor tyrosine-based activation motifs (ITAMs) in the CD3ζ domain, which triggers the downstream signaling pathways including PI3K/AKT, NF-κB, and MAPK. These pathways are significantly affected by the structure of the intracellular signaling domain [69].

In second and later generations of CAR, co-stimulatory domains play an important role in modulating T cell metabolism and function. For example, CD28-mediated co-stimulation facilitates rapid proliferation and effector differentiation by enhancing glycolytic metabolism. This enables immediate cytotoxic responses. In contrast, 4-1BB-mediated co-stimulation favors mitochondrial biogenesis and promotes the differentiation of memory T cells, hence facilitating long-term persistence and tumor surveillance [68].

CAR T-cells destroy tumor cells by two principal cytotoxic mechanisms: (1) The secretion of cytotoxic granules, where perforin creates membrane holes and granzymes initiate apoptosis; and (2) Fas–Fas-FasL-mediated signalling, which activates programmed cell death in target cells. These pathways are particularly crucial for overcoming tumor resistance and heterogeneity.

Besides direct cytotoxicity, CAR T cells release pro-inflammatory cytokines including IL-2, IFN-γ, and TNF-α. These cytokines augment the proliferation and viability of CAR T cells and attract supplementary immune effectors to the tumor location. In certain instances, a portion of CAR T cells evolves into memory phenotypes, which possess the ability for prolonged monitoring and reactivation upon re-encounter with antigens. This feature has resulted in their characterization as a "living drug."
Most importantly, the efficacy of CAR T-cells is unaffected by the patient's HLA status, making the treatment widely applicable. Nevertheless, resistance mechanisms, such as antigen

downregulation or the release of immunosuppressive molecules, can constrain the treatment effectiveness. These problems highlight the necessity of integrating immune evasion and feedback control into experimental systems and computational models [69, 70].

8.1.2 Clinical Successes and Limitations

CAR T-cell therapy has rapidly evolved from an experimental idea to a clinically approved treatment for many hematological malignancies. The therapy obtained its initial approval from the U.S. Food and Drug Administration (FDA) in 2017 for the treatment of relapsed or refractory B-cell acute lymphoblastic leukemia (B-ALL). Subsequently, it has been extended to span more hematological cancers, such as diffuse large B-cell lymphoma (DLBCL), mantle cell lymphoma (MCL), follicular lymphoma, and multiple myeloma. In several cases, especially when traditional therapies have been ineffective, CAR T cells have shown remarkable therapeutic outcomes [66, 69].

A key benefit of this treatment is its ability to achieve long-term remission, frequently with a single infusion, especially seen in pediatric B-ALL and adult DLBCL. These long-term responses are frequently linked to the presence of memory-like CAR T cell populations, which give ongoing protection against residual illness. Furthermore, using a patient's own T cells makes the therapy more individualized, boosting compatibility and lowering the hazards associated with immunological mismatch.

Despite these advances, significant clinical difficulties remain. Relapse is still prevalent in many individuals, typically due to antigen loss or T cell depletion. Toxicities such as cytokine release syndrome (CRS) and neurotoxicity remain major clinical concerns that require close monitoring and management. Furthermore, while the effectiveness in hematologic malignancies has been widely reported, extending this to solid tumors has been difficult due to various factors such as immunosuppressive microenvironments and restricted T cell penetration. **Table 8.1** summarizes these limitations and highlights ongoing research and innovation areas [1, 68].

8.1.3 Therapeutic and Manufacturing Strategies to Overcome Challenges

As discussed in the previous section, while CAR T-cell therapy has achieved success in hematologic malignancies, its application, especially to solid tumors, remains limited. In order to enhance clinical outcomes and ensure treatment consistency, researchers have developed a range of biological and engineering methods. These methods are summarized in **Table 8.2** and **Table 8.3**, highlighting both therapeutic combinations and manufacturing innovations.

Table 8.1. *Key Clinical Limitations and Challenges in CAR T Cell Therapy*

Category	Specific Challenge	Description
Relapse and Resistance	Antigen loss	Tumor cells may downregulate or loose expression of the target antigen (e.g., CD19), leading to immune escape.
	T cell exhaustion	Chronic stimulation or immunosuppressive signaling may impair CAR T cell function over time.
	Limited persistence	CAR T cells may decline too quickly post-infusion, reducing long-term antitumor activity.
Toxicity and Side Effects	Cytokine Release Syndrome (CRS)	An acute inflammatory response triggered by cytokine surge; ranges from mild to life-threatening.
	ICANS	Neurologic toxicity, such as confusion or seizures, sometimes reversible but poorly understood.
	B-cell aplasia	CD19-targeting may deplete healthy B cells, compromising immunity and increasing infection risk.
Challenges in Solid Tumors	Tumor microenvironment (TME)	Immunosuppressive cells, metabolic stress, and inhibitory molecules restrict CAR T cell function.
	Poor trafficking	Engineered T cells often struggle to reach or penetrate solid tumor sites.
	Antigen heterogeneity	Target antigens may also be present on healthy tissues, increasing off-target toxicity risk.

Table 8.2. *Therapeutic Combinations to Improve CAR T Cell Efficacy [1]*

Strategy	Mechanism	Purpose
Lymphodepletion Chemotherapy	Temporarily weakens the patient's immune system.	Boosts CAR T survival and expansion.
Radiotherapy (RT)	Causes local inflammation and weakens tumor defenses.	Helps CAR T cells enter the tumors more easily.
Oncolytic Viruses (OVs)	Uses viruses that infect and damage tumor cells, and boost immune signals.	Improves tumor microenvironment and antigen exposure.

Table 8.3. *Manufacturing Innovations to Improve CAR T Cell Consistency and Safety [1]*

Focus Area	**Innovation**	**Purpose**
Gene Editing	CRISPR or non-viral methods to insert genes.	Allows safer and more accurate modification of T cells.
Cell Culture Methods	Improved growth conditions with special signals (e.g., IL-7, IL-15) and shorter expansion time.	Helps in growing CAR T cells that last longer and stay effective.
T Cell Subtypes	Tuning CD4/CD8 ratios and enriching for early memory T cells.	Improves long-term efficacy and persistence.
Safety Features	Built-in “off-switches" (like Caspase-9) and checks for where genes are inserted.	Minimizes the risk off-target effects or cell malfunction.

8.2 Computational and Mathematical Modeling of CAR T-Cell Therapy

Recent advancements, such as gene editing, cytokine regulation, and multi-antigen targeting, have significantly enhanced the therapeutic effectiveness and reliability of CAR T-cell therapy. However, these improvements introduce additional layers of biological and technical complexity because **treatment efficacy is influenced not just by “design choices" but also by the dynamic interactions with tumor cells, cytokine environments, and individual patient immunological profiles. As these systems get more complicated, intuitive reasoning alone is no longer sufficient for predicting treatment results or guiding intervention strategies. From the identification of antigens and the development of immunological synapses to clonal proliferation, cytokine signaling, and the final elimination of tumors or immune fatigue, each phase is controlled by complex, nonlinear relationships. Also, each process spans several spatial and temporal dimensions and is influenced by variables like tumor heterogeneity, antigen evasion, and the immunosuppressive environment. These limitations highlight the need for predictive frameworks that can capture various dynamic behaviors and inform about the therapeutic design**.

In response to the above limitation, researchers are increasingly turning to mathematical models and computer simulations. They offer a formal and quantitative framework for simulating CAR T cell function, estimating all essential parameters, and testing various therapeutic scenarios in silico before any clinical application [1].

Mechanistic models, particularly those utilizing systems of ordinary differential equations (ODEs), have been extensively used in systems biology to explain these dynamics. These models formalize biological hypotheses and describe population-level interactions among CAR T cells, tumor cells, and other components of the immune system. These models are important not just for their prediction capability but also for their interpretability. They enable researchers to examine the impacts of various parameters, such as T cell dosage, antigen density,

or cytotoxicity rate, and conduct sensitivity analysis to determine the elements most critical to therapeutic effectiveness. For example, predator–prey models, which describe CAR T-cells as predators and tumor cells as prey, emphasize that T-cell fatigue or tumor recurrence may result in relapse [71].

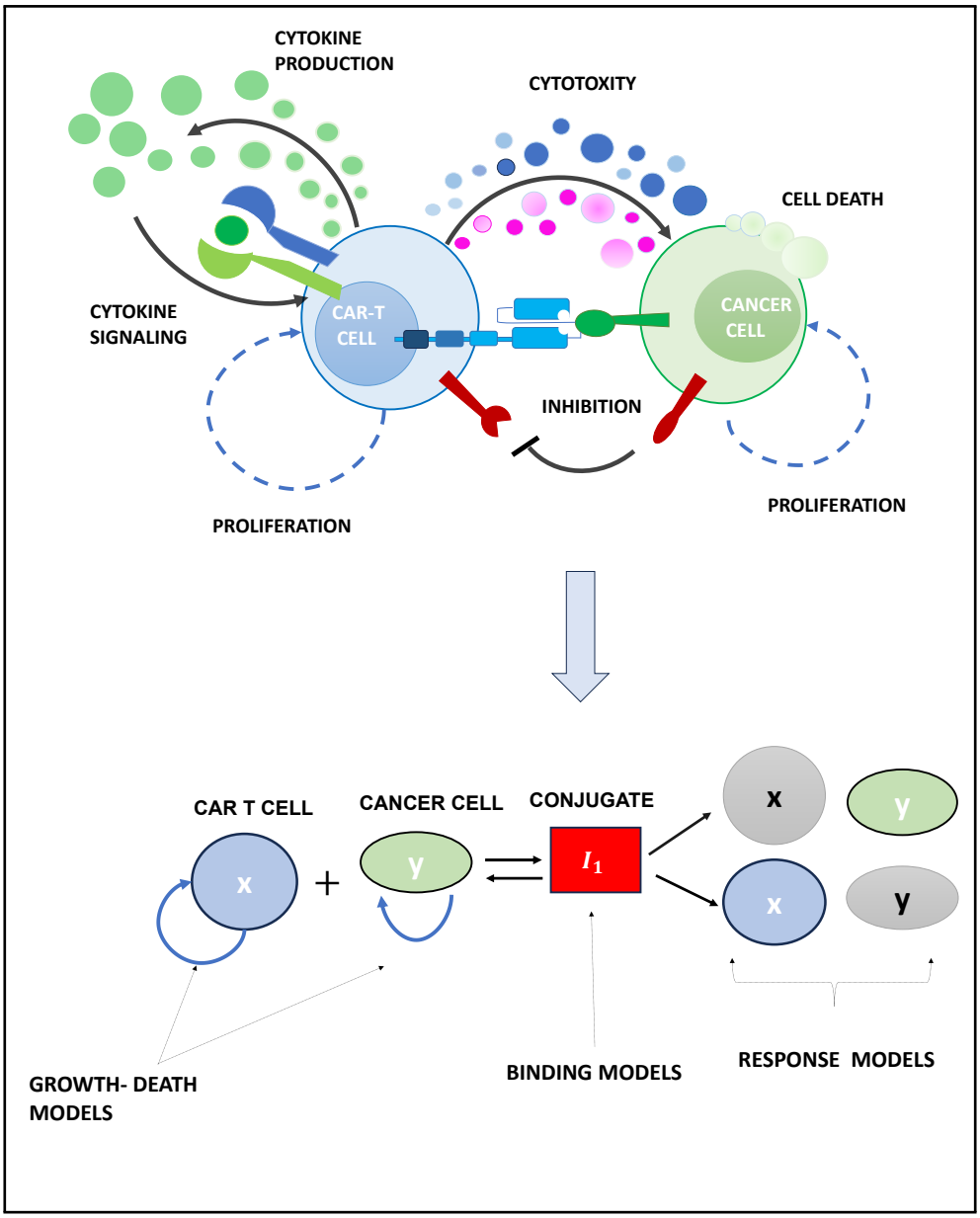

Figure 8.3. *Schematic overview of CAR T cell function and its regulatory mechanisms [1].*

Hardiansyah and Ng [72] used ODE-based models to demonstrate the impact of proliferation–exhaustion equilibrium on sustained tumor management. Similarly, Sahoo et al. [73] also employed a mechanistic model to investigate the influence of therapy factors on glioma tumor response and recurrence.

However, mechanistic modeling encounters two significant constraints. First, developing models that are both exhaustive and computationally feasible remains a significant challenge, particularly in complex systems such as CAR T cell treatment. Capturing essential biological interactions, like antigen binding, intracellular signaling, and cytokine feedback, frequently requires assumptions that may oversimplify or neglect critical dynamics. Furthermore, these models typically incorporate several parameters, many of which are challenging to assess directly or are only partially visible in clinical or experimental contexts. This may result in problems of parameter identifiability and model ambiguity.

This led researchers to adopt data-driven methodologies that provide system dynamics directly from time-series measurements. One such approach is the Sparse Identification of Nonlinear Dynamics (SINDy), which infers compact differential equations by selecting only the most relevant terms from a large candidate library. This makes it particularly suitable for biological systems in which the governing equations are only partially known or hidden by noise.

Brummer et al. [45] used SINDy to analyze CAR T cell killing test data, demonstrating its usefulness. Their strategy produced concise models that matched experimental dynamics while retaining biological interpretability, emphasizing essential aspects like saturation effects and effector depletion. As CAR T treatments get increasingly complicated, data-driven modeling methods such as SINDy provide a powerful means of identifying key processes, refining treatment strategies, and eventually tailoring therapies to specific patients.

8.3 Problem Statement

In this study, we aim to discover interpretable dynamical models that govern the interactions between effector cells, tumor cells, and other important immunological factors using Sparse Identification of Nonlinear Dynamics (SINDy). Specifically, **given time-series data (simulated) and a biologically informed library of candidate terms (from literature), our goal is to find ordinary differential equations that can represent the key structure of the effector-tumor cell interaction network**.

We try to see how SINDy, equipped with a feature library that reflects known immunological processes, can infer the type (e.g., activation, suppression, saturation) of interactions between system variables. The resulting models may be assessed for biological plausibility, prediction accuracy, and interpretability, with the **long-term goal of guiding therapy design and understanding critical therapeutic success thresholds (for example, tumor clearance, immune weariness, and cytokine overshoot)**.

This approach may bridge data-driven discovery with mechanistic modeling, providing a scalable framework for evaluating CAR T cell treatment dynamics in both hematological and

solid tumor settings.

Approach

We used three well-known ODE models from the literature [74] to create a structured modeling pipeline to see how SINDy can figure out understandable interaction dynamics in adoptive immunotherapy.

Table 8.4. *Summary of tumor–immune interaction models used for simulation and SINDy training*

Model	Variables	Focus
Model 1	E, T	Model tumor growth and elimination by effector cells and capture bistability between tumor-free and tumor-present states.
Model 2	E, T, I	Study tumor regulation via interleukin-mediated immune activation and explore therapy using effector and cytokine infusion.
Model 3	E, T, N	Model tumor-driven activation of naive T cells and study immune response amplification and tumor control.

To standardize the modeling framework and facilitate the use of traditional SINDy, we initially nondimensionalized each model to minimize parameter redundancy and accelerate numerical simulation. We produced time-series datasets from each nondimensional system using numerical integration and yielding three trajectory datasets including either two or three state variables.

To facilitate generalization and cross-model discovery, we created a unified dataset by aligning and concatenating the state variables from all three models. This produced a consolidated dataframe with columns (E, T, I, N). Essentially, effector cells (E) and tumor cells (T) are common in all models and are the primary variables of interest.

Next, we constructed a biologically informed library of candidate features to be used by SINDy. This library was created to incorporate all forms of regulatory, proliferative, and suppressive interactions seen in the three source models.

Once the library was constructed, the identities of the original models were intentionally disregarded. The system was regarded as if just the time-series data and the functional hypotheses (encoded in the library) were accessible, thereby replicating a true data-driven discovery scenario.

To assess the efficacy of this approach, we separately trained SINDy models by supplying the input for each appropriate subset of variables, namely (E, T), (E, T, I), (E, T, N), and the entire set of variables (E, T, I, N). This configuration reflects a minimal practical scenario in which users have access to specific measured variables (such as effector and tumor cell counts) and aim to reconstruct the underlying governing equations directly from experimental data.

Post-training, the identified SINDy models were simulated forward in time to examine their emerging dynamical behaviors. We evaluated the steady states of each system to deduce essential qualitative regimes, such as tumor removal or coexistence, and investigated how the underlying interactions influenced these results. This enabled us to evaluate the biological plausibility and interpretative significance of the inferred models, revealing possible therapeutic scenarios and system-level behaviors associated with CAR T cell dynamics.

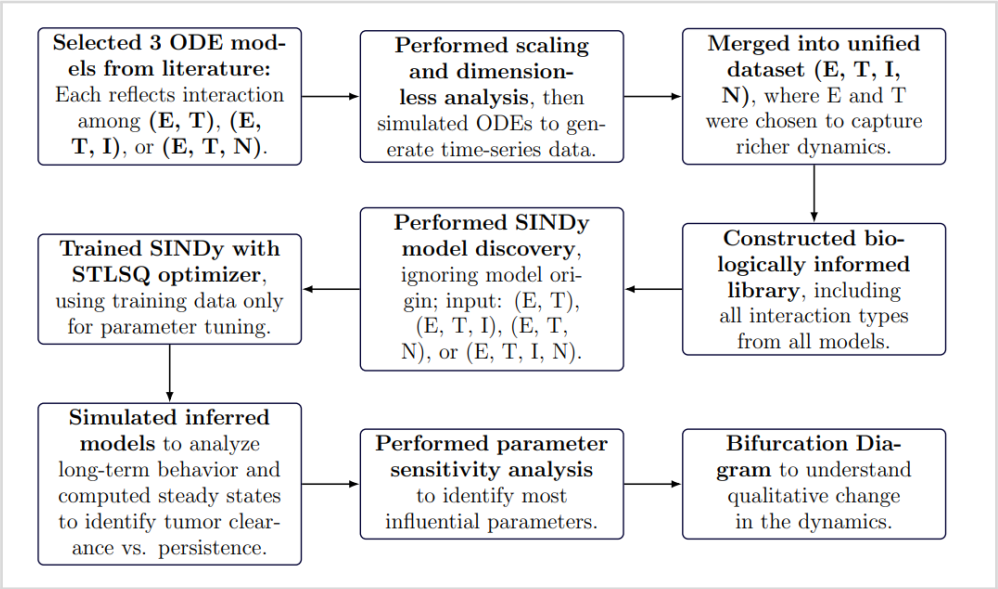

Figure 8.4. *Schematic of the workflow for discovering CAR T cell–tumor interaction dynamics using SINDy.*

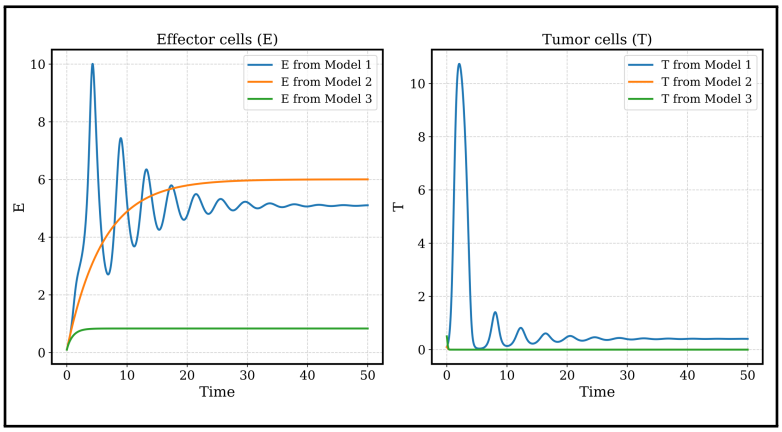

Figure 8.5. *Effector and tumor dynamics from the three source models. This illustrates the simulated time series for effector (E) and tumor (T) cell populations derived from the three previously stated literature-based ODE models. The left panel illustrates the time-series trajectories of effector cells (E), whereas the right panel depicts the dynamics of tumor cells (T). We chose E and T from model 1 to keep in the unified data frame because of their rich and diverse dynamical behaviors, including tumor clearance, persistence, and oscillatory immune responses.*

Results

8.3.1 Case I: Selecting E and T as Input Measurements:

Equations learned using SINDy:

$$\frac{dx}{dt} = \alpha - \beta\, x - \gamma_1\, xy + \gamma_2\, \frac{xy}{1+y}\,,$$
$$\frac{dy}{dt} = -\delta\, xy + r\, y - k\, y^2\,,$$

where $x :=$ E (effectors), $y :=$ T (tumors), and the parameters take the values,

$$\alpha = 1.000, \quad \beta = 1.000, \quad \gamma_1 = 0.168, \quad \gamma_2 = 3.022, \quad \delta = 0.843, \quad r = 4.369, \quad k = 0.176.$$

This predicted framework captures essential interactions between effectors and tumor cells. Effector cells are introduced at a constant rate and decrease through natural decay and exhaustion when they interact with the tumor. The presence of tumors also induces effector proliferation through a saturating response, indicating limited activation at higher tumor burdens. Tumors proliferate logistically but are subjected to elimination by CAR T-mediated cytotoxicity.

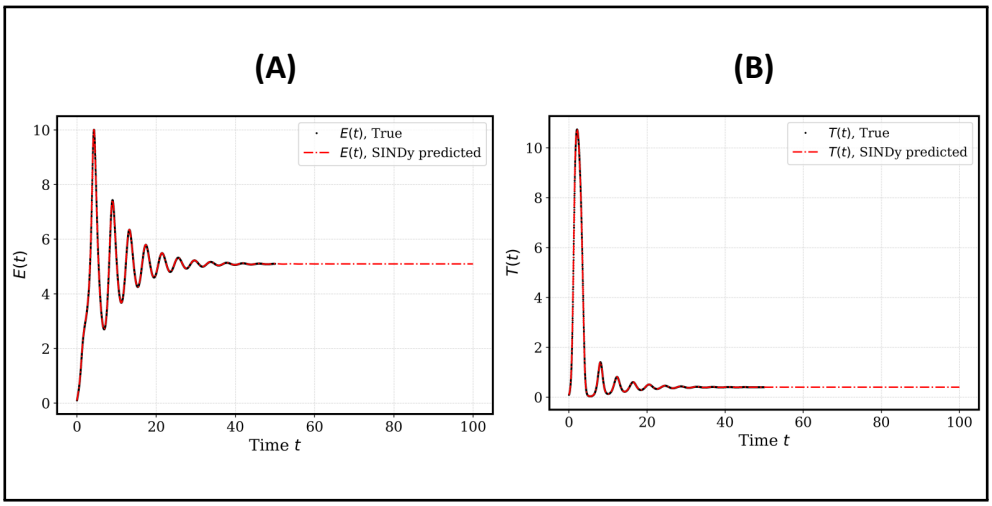

Figure 8.6. *Time-series evolution of the discovered SINDy model trained using (E ,T) as inputs. Panels (A) and (B) show the time evolution of the variables E (effector) and T (tumor) respectively. Solid black curves denote ground-truth data, whereas dashed red curves represent the trajectories predicted by SINDy.*

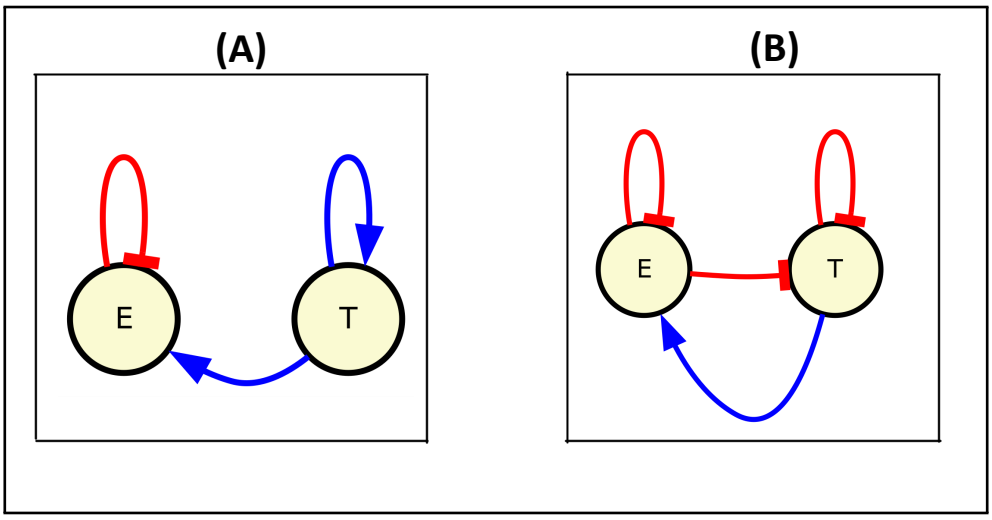

Figure 8.7. *Interaction networks drawn near the steady states discovered from the SINDy-inferred model for E-T interaction dynamics. (A) and (B) show the interaction network near the steady-state* ***tumor-free state*** $(1.000,\ 0)$*, and* ***(co-existance)*** $(5.10,\ 0.406)$*, respectively. Edges represent regulatory influences drawn using the signs of the Jacobian matrix evaluated at each equilibrium. Blue arrows indicate positive effects (activation), whereas red arrows indicate negative effects (inhibition). In panel (A), the negative loop on E may reflect natural decay or exhaustion of effector cells in the absence of any stimulation. The positive loop on T may arise from the underlying model structure, where tumor cells grow in the absence of immune pressure. This indicates that a small tumor regrowth can destabilize the system. Panel (B) corresponds to the coexistence regime, where effector cells persist due to tumor stimulation but are unable to fully eliminate the tumor, probably due to exhaustion or saturation.*

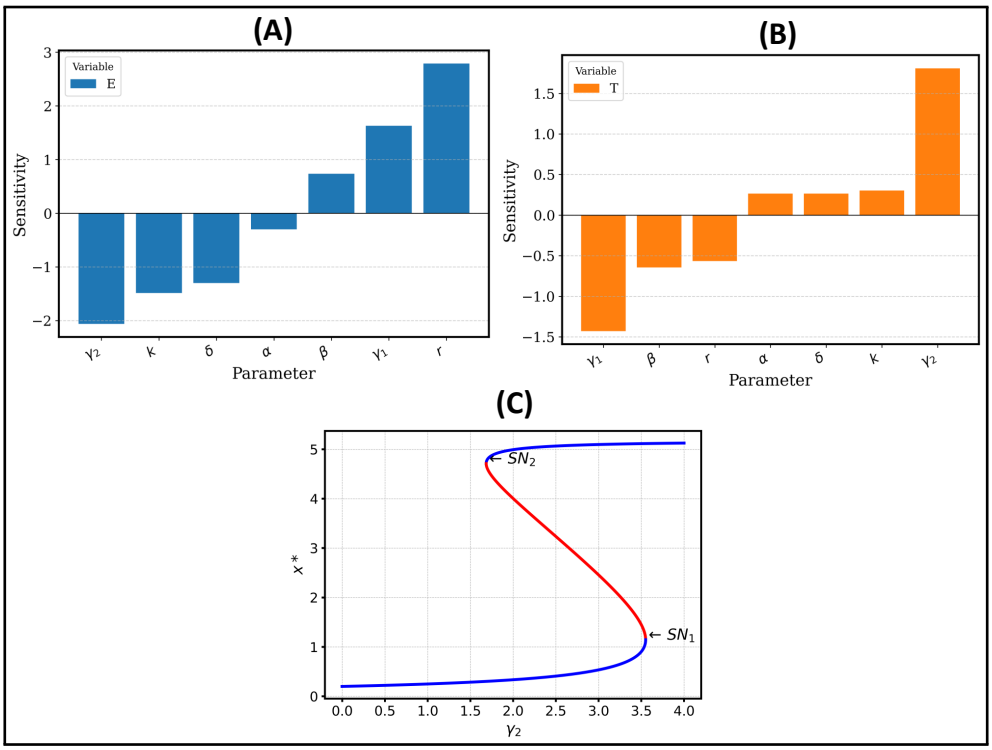

Figure 8.8. *Local parameter sensitivity analysis and one-parameter bifurcation diagram for the SINDy-inferred model for E-T dynamics. Panels (A) and (B) illustrate the scaled sensitivity of the steady-state values of effector cells (E) and tumor cells (T), respectively, in relation to each model parameter. A positive bar indicates that increasing the parameter increases the steady-state value of the variable, while a negative bar indicates a suppressive effect. In Panel (A), the effector population is most sensitive to γ_2 (tumor-driven stimulation) and r (tumor growth rate), implying that both stimulation and tumor burden significantly affect long-term effector persistence. In Panel (B), tumor steady-state levels are positively influenced by γ_2, while adversely impacted by γ_1 (exhaustion), β (effector decay), and r. These contrasting effects underscore a dynamic equilibrium between tumor-induced activation and effector-mediated inhibition that regulates overall tumor burden. Panel (C) shows the codimension one bifurcation diagram for the parameter γ_2. As γ_2 increases (more effector stimulation) , the system undergoes two saddle-node bifurcations near $\gamma_2 \approx 1.68$ and 3.55, respectively. Low γ_2 corresponds to effector failure, and high γ_2 makes the system stable in a no-tumor regime. Therefore, depending on initial conditions, it may fall into the tumor-controlled or tumor-dominant state.*

8.3.2 Case II: Selecting E, T, I as Input Measurements:

Equations learned using SINDy:

$$\begin{aligned}
\frac{dx}{dt} &= \alpha_1 - \alpha_2 x - \alpha_3 xy + \alpha_4 \left(\frac{xy}{1+y}\right), \\
\frac{dy}{dt} &= -\beta_1 xy + \beta_2 y - \beta_3 y^2, \\
\frac{dz}{dt} &= \gamma_1 x - \gamma_2 \left(\frac{xz}{1+z}\right) + \gamma_3 \left(\frac{zy}{1+y}\right) - \gamma_4 z.
\end{aligned}$$

where $x :=$ E, $y :=$ T, $z :=$ I, and the values of parameters are given in **Table 8.5**.

These equations suggest that effector cells (E) are produced at a constant rate and are stimulated by tumor presence by a saturation term ($\frac{xy}{1+y}$). Their inhibition occurs due to the natural decay, direct tumor contact (xy), and by interleukin. Tumor cells (T) exhibit logistic growth, and are reduced by interactions with effector cells. A saturating term involving interleukins ($\frac{zy}{1+y}$) supports its growth. Interleukin levels (I) are upregulated by effector cells and tumor–interleukin interaction, whereas they are reduced through interactions with E and by natural decay.

Table 8.5. *Parameter values from SINDy-inferred model (E, T, I)*

Effector (E)		**Tumor (T)**		**Interleukin (I)**	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
α_1	1.000	β_1	0.843	γ_1	0.039
α_2	1.000	β_2	4.369	γ_2	198.217
α_3	0.168	β_3	0.176	γ_3	55.130
α_4	3.022			γ_4	44.071

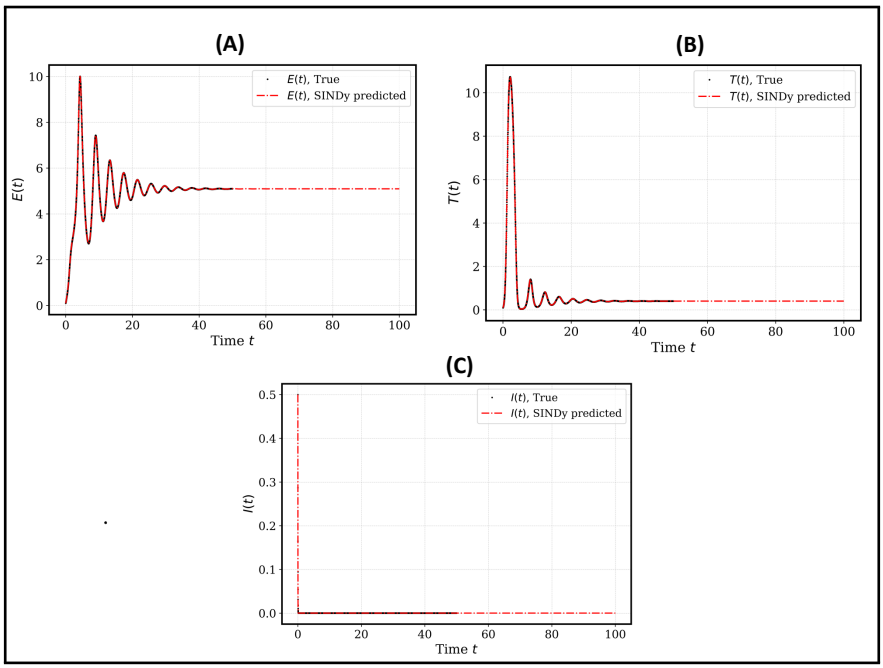

Figure 8.9. *Time-series evolution of the discovered SINDy model using (E ,T, I) as inputs. Panels (A)-(C) illustrate the time evolution of effector cells E(t), panel, tumor burden T(t), and interleukin I(t). In each panel, the solid black curves represent the ground-truth simulation from the original ODE system, while the dashed red curves represent predictions made by the SINDy-discovered model. The close match between predicted and true trajectories demonstrates the accuracy of the identified governing equations in capturing system dynamics.*

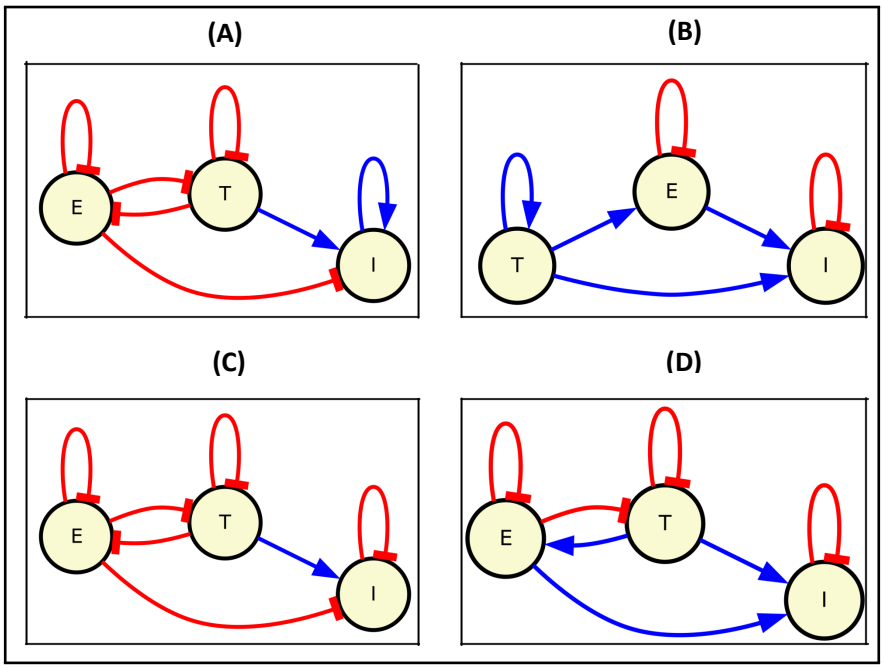

Figure 8.10. *Interaction networks drawn near the steady states discovered from the SINDy-inferred model for E-T-I dynamics. Each panel shows the local interaction network near a steady state. Panel (A) illustrates the interaction network at the steady state* $(0.543,\ 22.2,\ 11.4)$ ***(high tumor burden)****, panel (B) represents the network near* $(1.0,\ 0.0,\ 0.00016)$*,* ***(tumor free state)*** *panel (C) shows the interaction network near* $(2.42, 13.2,\ 0.00020)$*, and panel (D) represents the network near* $(5.10,\ 0.406,\ 0.00019)$ ***(low tumor burden)****. The negative self-loop on E likely indicates natural death or depletion of effector cells, probably due to the lack of continuous antigen stimulation. The positive loop on T may arise from intrinsic tumor growth in the absence of immunological pressure.*

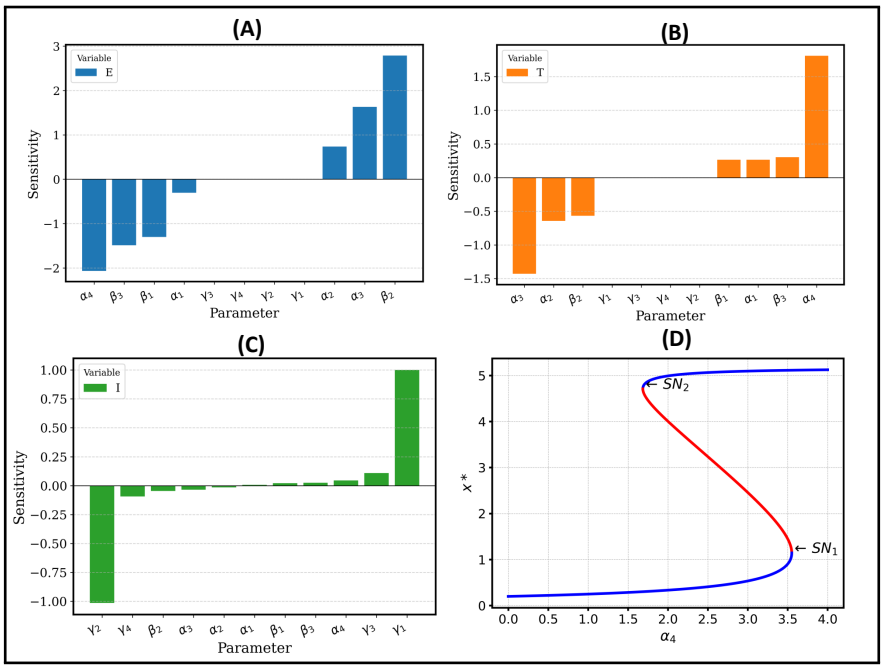

Figure 8.11. *Local scaled parameter sensitivity analysis and codimension-one bifurcation diagram for the SINDy-inferred model for E-T-I dynamics. Panels (A), (B), and (C) illustrate the local scaled parameter sensitivities of the steady-state values of effector cells (E), tumor cells (T), and interleukin (I), respectively. Panel (A) indicates that E is strongly activated by β_2, α_3 and is negatively influenced by α_4, β_3. Panel (B) demonstrates that T is very sensitive to α_3 and α_4. Panel (C) demonstrates that I is significantly enhanced by its own growth rate γ_1 and lowered by γ_2 (intrinsic decay), and all other factors have no effect on its steady state. Panel (D) shows the codimension one bifurcation diagram for the parameterα_4. The system also exhibits bistability with two saddle node bifurcations, marking the transition between the tumor-free and tumor-persistence states. As α_4 increases, the system shifts from a low-effector, tumor-dominated state to a high-effector, tumor-controlled regime. This illustrates the critical role of tumor-induced effector stimulation in therapy success.*

8.3.3 Case III: Selecting E, T, N as Input Measurements:

Equations learned using SINDy:

$$\begin{aligned}
\frac{dx}{dt} &= \alpha_1 - \alpha_2 x - \alpha_3 xy + \alpha_4 \left(\frac{zy}{1+y}\right) + \alpha_5 \left(\frac{xy}{1+y}\right), \\
\frac{dy}{dt} &= -\beta_1 xy + \beta_2 y - \beta_3 y^2, \\
\frac{dz}{dt} &= \gamma_1 - \gamma_2 z.
\end{aligned}$$

where $x :=$ E, $y :=$ T, $z :=$ N, and parameters take the values from **Table 8.6**.

This inferred model demonstrates that effector cells (E) are produced by constant input and stimulation from tumor cells and naive T cells, but are reduced by natural decay, direct tumor contact, and naive T cell activity. Tumor cells (T) expand logistically and are destroyed by

effector-mediated death. Further, naive T lymphocytes (N) are produced at a constant rate and depleted due to natural death

Table 8.6. *Parameter values from SINDy-inferred model (E, T, N)*

Effector (E)		**Tumor (T)**		**Naive T Cell (N)**	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
α_1	1.000	β_1	0.843	γ_1	1.000
α_2	1.000	β_2	4.369	γ_2	0.667
α_3	0.168	β_3	0.176		
α_4	0.001				
α_5	3.021				

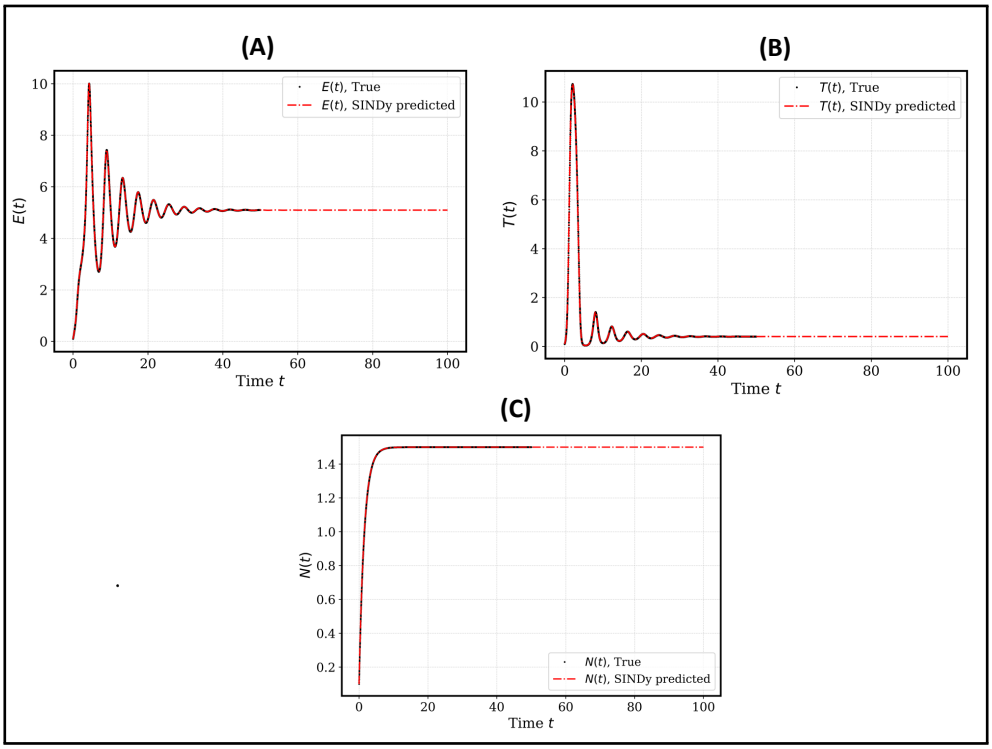

Figure 8.12. *Time-series evolution of the discovered SINDy model using (E ,T, N) as inputs. Panels (A-C) depict the time evolution of effector cells $E(t)$, tumor cells $T(t)$, and naive T cells $N(t)$, respectively. The solid black curves show the ground-truth simulation data, while the dashed red curves show the trajectories predicted by the inferred model. The predicted dynamics closely correspond to the actual system behavior, precisely capturing both transient oscillations and long-term steady-state convergence. The model accurately depicts early effector proliferation, tumor clearance, and the rapid increase and saturation of naïve T cells. Panel (D) shows the codimension one bifurcation diagram for the parameterα_5.*

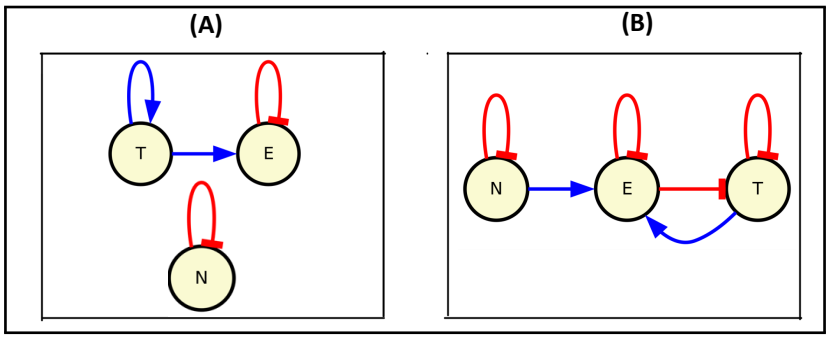

Figure 8.13. *Interaction networks drawn near the steady states discovered from the SINDy-inferred model for $E - T - N$ dynamics. Panel (A) illustrates the interaction network obtained from the Jacobian matrix evaluated near the steady state $(1.0,\ 0.0,\ 1.5)$* ***(tumor free state*** *of the SINDy-inferred model. Effector cells show self-inhibition and obtain positive activation from tumor cells, possibly indicating immunological stimulation in response to tumor presence. Panel (B) demonstrates the interaction network obtained near the steady state $(5.10,\ 0.406,\ 1.50)$* ***(coexistence)****. Effector cells show self-inhibition and obtain positive activation from tumor cells, as well as from naive T cell.*

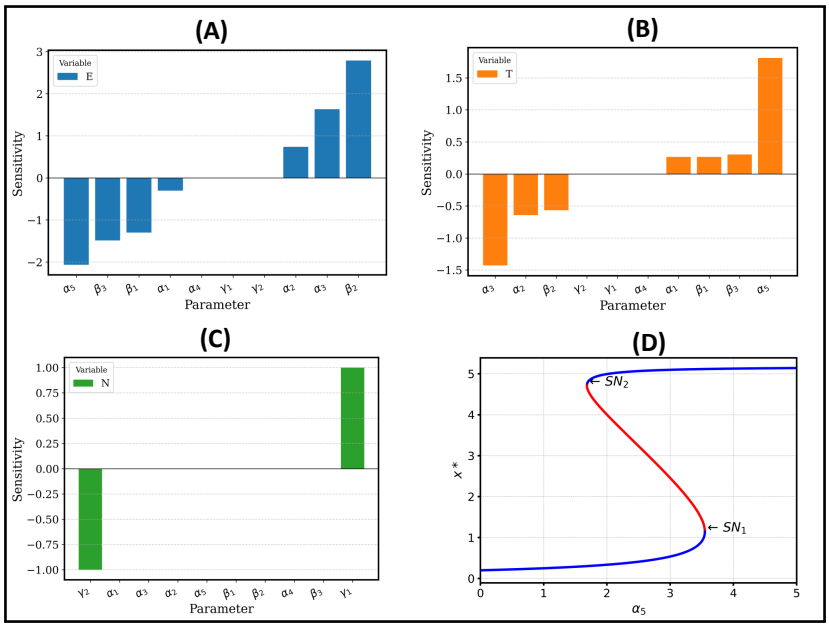

Figure 8.14. *Local scaled parameter sensitivity analysis for the SINDy-inferred model using (E, T, N) dynamics. Panels (A), (B), and (C) illustrate the local parameter sensitivities of the steady-state values of effector cells (E), tumor cells (T), and naive T cells (N), respectively. Panel (A) illustrates that E is significantly reduced by β_2 (immune-mediated tumor elimination). In Panel (B), T exhibits the highest positive sensitivity to α_5 (tumor-induced effector activation) and the highest negative sensitivity to α_3 (effector decay). Panel (C) indicates that N is dominantly governed by its production rate γ_1 (positive) and decay rate γ_2 (negative), with negligible impact from other parameters. The system shows bistability for the parameter α_5. If the tumor-stimulated effector activation α_5 is strong enough, the system jumps to immune-dominant state, otherwise, it remains in tumor dominant state.*

8.3.4 Case IV: Selecting E, T, I, N as Input Measurements:

Equations learned using SINDy:

$$\begin{aligned}
\frac{dx}{dt} &= \alpha_1 - \alpha_2 x - \alpha_3 xy + \alpha_4 \left(\frac{xy}{1+y}\right), \\
\frac{dy}{dt} &= -\beta_1 xy + \beta_2 y - \beta_3 y^2, \\
\frac{dz}{dt} &= \gamma_1 - \gamma_2 z - \gamma_3 \left(\frac{zy}{1+y}\right) - \gamma_4 u, \\
\frac{du}{dt} &= \delta_1 - \delta_2 u.
\end{aligned}$$

where $x :=$ E, $y :=$ T, $z :=$I and $u :=$N. The values for the parameters is shown in **Table 8.7**.

This inferred model demonstrates that the persistence of effector cells is simultaneously influenced by tumor-induced activation. Tumor burden is regulated by immunological pressure while being regulated by interleukin-mediated feedback. The dynamics of interleukin are regulated by nonlinear feedback loops.

Table 8.7. *Parameter values from SINDy-inferred model (E, T, I, N)*

Effector Cells (E)		**Tumor Cells (T)**		**Interleukin (I)**		**Naive T cells (N)**	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
α_1	1.000	β_1	0.843	γ_1	0.076	δ_1	1.000
α_2	1.000	β_2	4.369	γ_2	15.418	δ_2	0.667
α_3	0.168	β_3	0.176	γ_3	417.170		
α_4	3.022			γ_4	0.033		

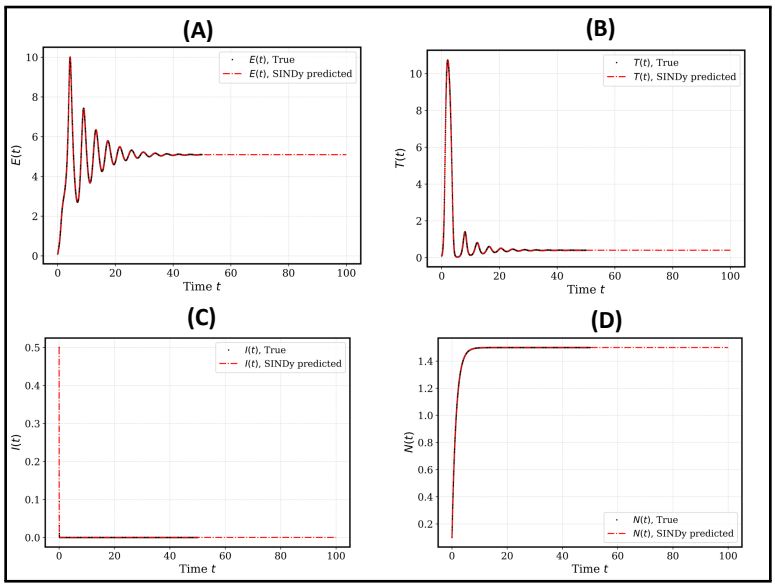

Figure 8.15. *Time-series evolution of the discovered SINDy model using* (E, T, I, N) *as inputs. Panel (A)-(D) show the time evolution of effector cells* $E(t)$*, tumor cells* $T(t)$*, interleukin* $I(t)$*, and naïve T cells* $N(t)$*, respectively. The solid black curves represent the ground-truth simulation data, while the dashed red curves indicate the trajectories predicted by the inferred model. The predicted dynamics closely match the true system behavior, accurately capturing both transient oscillations and long-term steady-state convergence. Also, the model captures the effector proliferation and tumor clearance, as well as the decay, rapid rise and saturation of inerleukin and naive T cells in the same way as in the previous inferred models.*

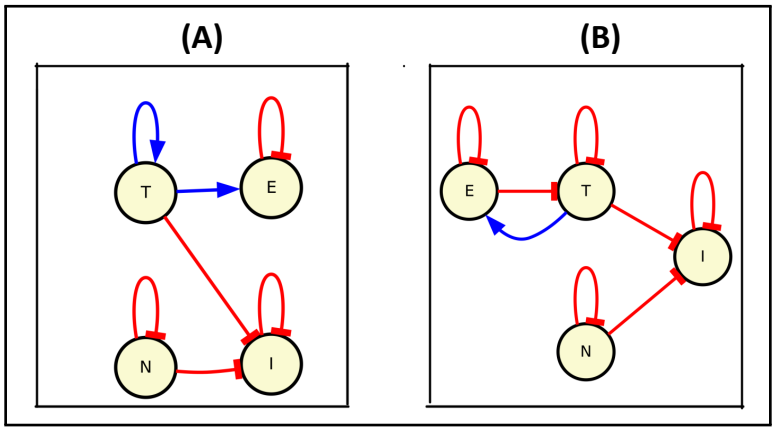

Figure 8.16. *Interaction networks drawn near the steady states discovered from the SINDy-inferred model for E-T-I-N dynamics. Interaction networks derived from the Jacobian matrices, evaluated near two distinct steady states* ***(tumor free regime)*** $(1.00,\ 0.00,\ 0.00172,\ 1.50)$ *and* $(5.10,\ 0.406,\ 0.00019,\ 1.50)$ ***the coexistence regime,*** *respectively. In both states, the effector population* (E) *inhibit itself. Naive T cells* (N) *and interleukin (I) show the same interaction effect in both cases, while tumor* (T) *activates E in both regime.*

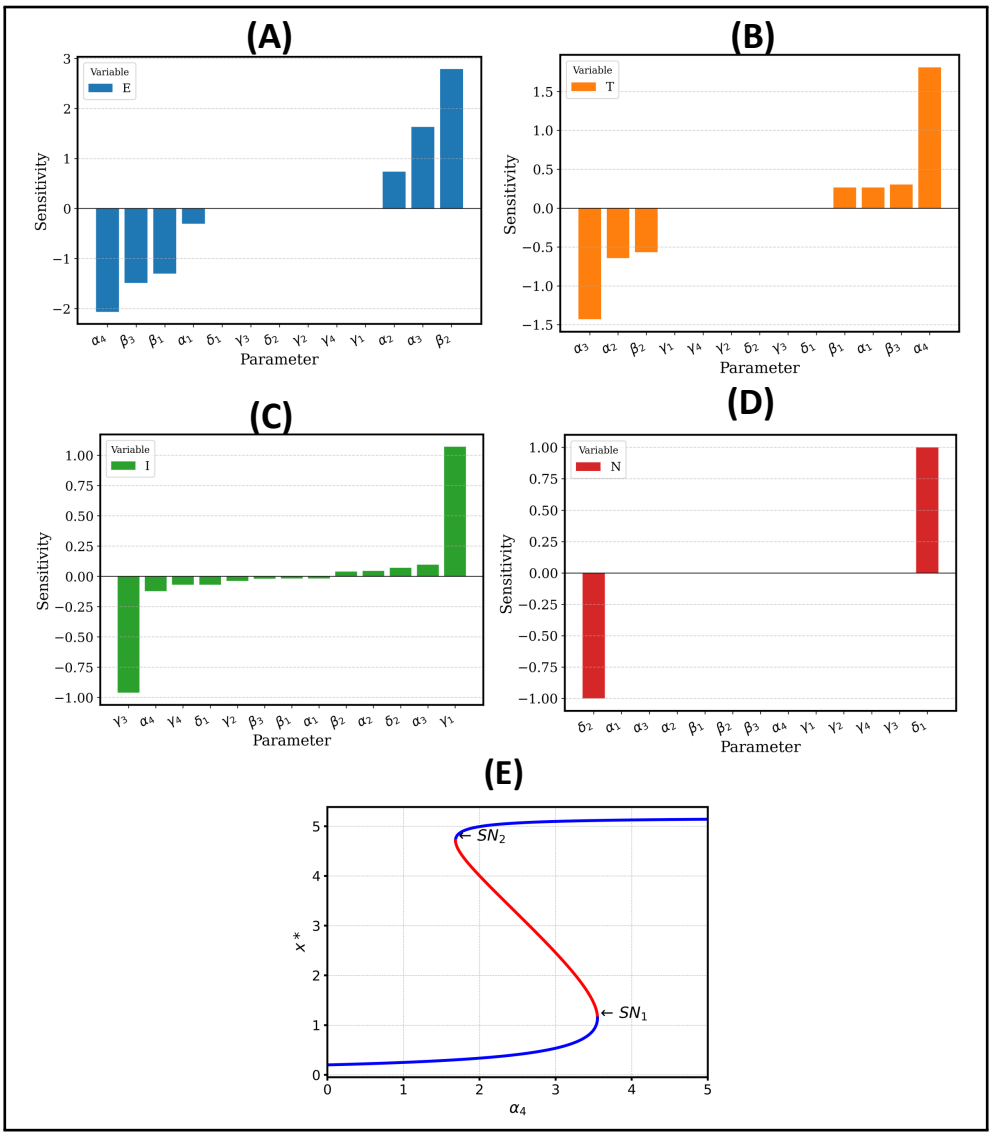

Figure 8.17. *Local parameter sensitivity analysis and bifurcation diagram for the SINDy-inferred model using (E, T, I, N) dynamics. Panels (A)–(D) illustrate the local parameter sensitivity of the steady-state values of effector cells (E), tumor cells (T), interleukin (I), and naive T cells (N), respectively. Panel (A) indicates that E is primarily affected by α_4 and decay rate β_2 . Panel (B) indicates that T is mainly influenced by effector-mediated factors α_4 (positive) and α_3 (negative) Panel (C) illustrates that I is suppressed by tumor-mediated inhibition through γ_3 and favorably influenced by its own proliferation factor γ_1. Panel (D) illustrates that N is closely regulated by its own production δ_1 (positive) and decay δ_4 (negative), with minimal impact from the remaining parameters of the system. Panel (E) shows the codimension one bifurcation diagram for the parameter α_4 (tumor-stimulated effector proliferation). This system also shows bistability for certain choice of α_4. The treatment success depends on the initial immune level.*

Discussion

In this work, we developed a comprehensive modeling pipeline to assess whether SINDy can robustly recover interpretable immune–tumor dynamics directly from time series data, without depending on pre-existing mechanistic assumptions. We approached this by creating a unified dataset from three well-established ODE models, each emphasizing a different biological mechanism (effector-driven killing, cytokine activation, and naïve T cell recruitment) and a biologically informed library. Then, we trained SINDy models on this dataset using various observational settings to simulate minimal experimental scenario. Across all situations, we observed that the identified models not only reproduced important dynamical characteristics but also captured crucial regulatory influences. All the models exhibited bistability for certain choices of parameters, which can lead to treatment outcomes depending on the initial immune state. This illustrates that SINDy, when directed by a proper library, can generate predictive models of complicated immunotherapy dynamics. All source codes, simulation scripts, and SINDy workflows used in this study are available at https://github.com/Alka-CBhub/Chapter-8.

Limitations

Although SINDy aims to produce sparse models, several inferred models in our study maintained a considerable number of terms. This is partially due to our selected range of hyperparameters, within which the R^2 score were calculated on the training data. The R^2 was found to be almost constant across all choices of hyperparameters, and we selected the parameters for which we don't get too sparse models. Opting for more stringent regularization may have produced simpler dynamics and perhaps modified the quantity or characteristics of the identified stable states.

Additionally, all training and assessment were performed on simulated, noise-free data. In practical experimental contexts, measurement noise, limited observability, and irregular sampling may significantly affect performance.

Moreover, we created a biologically informed library, which still assumes a predefined set of potential interaction motifs, which may bias inference toward predicted forms and restrict the finding of any new mechanisms that are not represented in the candidate library.

Other Approaches

Approach I

To make our task compatible with classical SINDy, we performed non-dimensionalization and variable rescaling on the original ODE models before simulation. Since these operations are linear in nature, they don't affect the qualitative properties of the system, so in this sense, this process can be useful. Also, this approach removes parameters like saturation constants by normalization, thus simplifying the equations. While this helps to simplify the modeling work and reduces the number of parameters, it also leads to the loss of biologically significant

parameters. In many situations, keeping those parameters may be is critical, making traditional SINDy less suited.

In these situations, implicit modeling techniques such as Implicit SINDy or SINDy-PI become essential. These approaches allow for rational nonlinearities and can capture more biologically realistic dynamics. However, they offer significant challenges. Constructing a biologically relevant implicit library is not easy; it requires rigorous derivation of algebraic connections for each interaction and manual encoding into the feature set. Furthermore, SINDy-PI is based on convex optimization solvers (e.g., CVXPY), which might be unstable or infeasible for big, multidimensional, or poorly scaled datasets.

This is a practical restriction for using SINDy-PI to realistic CAR T cell datasets, which often have larger time spans, severe nonlinearities, and substantial dynamic fluctuation spanning many orders of magnitude.

Approach II

Even if SINDy-PI can be employed successfully, it requires choosing a suitable feature library customized for each case of input data. This indicates that complete automation is impractical in such cases.

To advance towards a more automated discovery framework, one may use the adaptability of vectorized SINDy-PI libraries by providing a minimal set of general biological interactions (e.g., activation, inhibition, saturation). Such libraries may accommodate diverse quantities of input variables without necessitating change. Nonetheless, vectorization also adds extra terms, particularly coupled derivatives among unrelated variables. Consequently, preprocessing on the implicit feature matrix, or symbolic post-processing, is crucial for eliminating extra derivative terms and preserving just the accurate implicit equation for each variable. This stage guarantees that the final model is interpretable, physiologically reasonable, and free from the artifacts of excessive library development.

Chapter 9

Discussion

This thesis provides a unified framework for discovering and analyzing the governing equations of biological systems using a hybrid approach that blends classical dynamical systems theory with modern sparse regression techniques. Building on the Sparse Identification of Nonlinear Dynamics and its variant SINDy-PI, we systematically demonstrated how sparse modeling techniques can recover governing equations and regulatory interactions from time-series data, even in the presence of nonlinearities, feedback, and hidden variables.

The framework developed in this thesis addresses the challenge of modeling complex biological systems by learning governing equations directly from time-series data. Using Takens' embedding theorem, we identify the minimum number of dimensions required for further model identification and reconstruct phase-space trajectories from limited measurements (in our case, a scalar time series). The Sparse Identification of Nonlinear Dynamics (SINDy) then infers minimal and interpretable ODEs, while its extension, SINDy-PI, handles rational nonlinearities commonly found in biological kinetics, such as Michaelis-Menten and Hill functions. This combination yields symbolic models that are both predictive and interpretable, capturing dynamics like oscillations and bistability.

The methodology was validated on several canonical biological models, including the FitzHugh-Nagumo and Goodwin oscillators, the Oregonator and glycolytic models, and mass-action and microbial growth systems. In each case, the inferred models successfully recovered the system's core dynamics and regulatory structure, confirmed through Jacobian analysis and bifurcation diagrams.

We applied the framework to two biologically relevant systems: the eukaryotic cell cycle and tumor-immune interactions in cancer immunotherapy. From synthetic data emulating experimental observations, we successfully rediscover governing equations that predict system behavior and reveal key regulatory motifs. Sensitivity analysis and bifurcation techniques further confirm that the inferred models exhibit known transitions and critical points consistent with biological expectations. All materials, codes and data related to this thesis are available at https://github.com/Alka-CBhub/MTechCB-Thesis.

Deliverables

- The framework enables discovering governing equations from time-series data, including situations when fewer variables are known and the remaining can be reconstructed via phase-space reconstruction.
- It supports discovery of both polynomial and rational nonlinearities, capturing behaviors like oscillations, bistability, and saturation.
- The pipeline includes integrated diagnostics such as sensitivity analysis, Jacobian-based interaction network, and bifurcation analysis for dynamic interpretation and validation.

Limitations and Future Directions

- **Synthetic validation only:** All analysis in this thesis was conducted on synthetic time-series data. Application to experimental or clinical datasets remains an important next step to assess robustness under real-world measurement noise and biological variability.
- **Sensitivity to noise:** While the framework performed well on clean synthetic data, its performance under noisy conditions remains to be evaluated. Incorporating denoising methods or regularized differentiation could improve reliability.
- **Structural identifiability:** Identifiability analysis was not systematically performed on the inferred models. Verifying structural identifiability using tools such as StrikePy is essential to ensure uniqueness and interpretability of model parameters.
- **Model simplification and factorization:** Although SINDy-PI successfully captured the system dynamics, some inferred rational expressions were not easily factorizable into known biological forms. Enforcing additional constraints or symbolic pattern matching could improve biological relevance.
- **Hyperparameter tuning:** Sparse regression in SINDy-PI depends heavily on threshold selection. We used uniform thresholding across all implicit equations. A more robust and automated strategy is needed to perform hyperparameter tuning.

Conclusion

In conclusion, this thesis advanced the area of systems biology by proposing a systematic and interpretable framework for inferring nonlinear dynamical models directly from time-series data. The suggested technique successfully combines mechanistic and data-driven strategies, providing a scalable alternative to conventional modeling strategies. By enabling data-driven discovery of biological systems with limited knowledge, this work opens new opportunities for hypothesis generation, mechanistic understanding, and treatment design in contexts where traditional modeling is insufficient.

Bibliography

[1] A. S. Colina, V. Shah, R. K. Shah, T. Kozlik, R. K. Dash, S. Terhune, and A. E. Zamora. Current advances in experimental and computational approaches to enhance CAR T cell manufacturing protocols and improve clinical efficacy. Frontiers in Molecular Medicine, 4:1310002, 2024.

[2] Lee A. Segel and Leah Edelstein-Keshet. Chapter 1: Introduction, chapter 1, pages 1–12. Society for Industrial and Applied Mathematics, 2013.

[3] Mikahl Banwarth-Kuhn and Suzanne Sindi. How and why to build a mathematical model: A case study using prion aggregation. Journal of Biological Chemistry, 295(15):5022–5035, 2020.

[4] Sean T. Vittadello and Michael P. H. Stumpf. Open problems in mathematical biology. Mathematical Biosciences, 354:108926, 2022.

[5] Bartosz Prokop and Lendert Gelens. From biological data to oscillator models using sindy. iScience, 27(4):109316, 2024.

[6] Haitao Yin, Rachel L. Kanasty, Ahmed A. Eltoukhy, Anthony J. Vegas, Jason R. Dorkin, and Daniel G. Anderson. Non-viral vectors for gene-based therapy. Nature Reviews Genetics, 15(8):541–555, August 2014. Epub 2014 Jul 15.

[7] Richard M. Plenge, Edward M. Scolnick, and David Altshuler. Validating therapeutic targets through human genetics. Nature Reviews Drug Discovery, 12(8):581–594, August 2013. Epub 2013 Jul 19.

[8] Niall Mangan, Steven Brunton, Joshua Proctor, and J. Kutz. Inferring biological networks by sparse identification of nonlinear dynamics. IEEE Transactions on Molecular, Biological and Multi-Scale Communications, 2, 05 2016.

[9] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proceedings of the National Academy of Sciences, 113(15):3932–3937, 2016.

[10] J. D. Murray. Mathematical Biology I: An Introduction, volume 1 of Interdisciplinary Applied Mathematics. Springer, 3 edition, 2002.

[11] René Thomas and Richard D'Ari. Biological Feedback. CRC Press, 1st edition, 1990.

[12] Uri Alon. An Introduction to Systems Biology: Design Principles of Biological Circuits. Chapman & Hall/CRC, 2006.

[13] Andrejs E. Treibergs and Philip Hartman. The hartman–grobman theorem, 2008. Accessed via Semantic Scholar.

[14] Robert M. May. Stability and Complexity in Model Ecosystems, volume 1. Princeton University Press, 1974.

[15] John J. Tyson. Classification of instabilities in chemical reaction systems. The Journal of Chemical Physics, 62(3):1010–1015, 1975.

[16] Marcelo A. Savi. Nonlinear Dynamics and Chaos, pages 93–117. Springer International Publishing, Cham, 2016.

[17] John J. Tyson and Bela Novak. A dynamical paradigm for molecular cell biology. Trends in Cell Biology, 30(7):504–515, July 2020.

[18] John J. Tyson. Cell cycle vignettes: Analysis of cell cycle dynamics by bifurcation theory. In Werner Dubitzky, Olaf Wolkenhauer, Kwang-Hyun Cho, and Hiroaki Yokota, editors, Encyclopedia of Systems Biology. Springer Verlag, Heidelberg, Germany, 2011.

[19] John J. Tyson. Cell cycle model analysis: Bifurcation theory. In Werner Dubitzky, Olaf Wolkenhauer, Kwang-Hyun Cho, and Hiroki Yokota, editors, Encyclopedia of Systems Biology, pages 274–278. Springer, New York, NY, USA, 2013.

[20] James E. Jr Ferrell, Tiffany Y.-C. Tsai, and Quanhua Yang. Modeling the cell cycle: Why do certain circuits oscillate? Cell, 144(6):874–885, March 2011.

[21] Bard Ermentrout. Simulating, Analyzing, and Animating Dynamical Systems: A Guide to XPPAUT for Researchers and Students. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1 edition, 2002.

[22] A. Dhooge, W. Govaerts, and Yu. A. Kuznetsov. Matcont: A matlab package for numerical bifurcation analysis of odes. ACM Transactions on Mathematical Software, 29(2):141–164, 2003.

[23] Denis Thieffry. Dynamical roles of biological regulatory circuits. Briefings in Bioinformatics, 8(4):220–225, 2007.

[24] Stefano Nichele. Review of introduction to the modeling and analysis of complex systems, h.sayama (ed.), open suny textbooks (2015). Artificial Life, 22(3):424–427, 2016. ISBN 978-1-942341-06-2 (color), 978-1-942341-08-6 (print), 978-1-942341-09-3 (ebook).

[25] Lucia Breierová and Mark Choudhari. An introduction to sensitivity analysis. Technical report, Massachusetts Institute of Technology, MIT OpenCourseWare, 2001. Accessed 22 July 2025.

[26] Stefan Hoops, Sven Sahle, Ralph Gauges, Christine Lee, Jürgen Pahle, Natalia Simus, Mudita Singhal, Liang Xu, Pedro Mendes, and Ursula Kummer. COPASI—a complex pathway simulator. Bioinformatics, 22(24):3067–3074, December 2006.

[27] Richard FitzHugh. Impulses and Physiological States in Theoretical Models of Nerve Membrane. Biophysical Journal, 1(6):445–466, 1961.

[28] Rubin R. Aliev and Alexander V. Panfilov. A simple two-variable model of cardiac excitation. Chaos, Solitons & Fractals, 7(3):293–301, 1996.

[29] Brian C. Goodwin. Oscillatory behavior in enzymatic control processes. Advances in Enzyme Regulation, 3:425–437, 1965.

[30] Didier Gonze and Peter Ruoff. The goodwin oscillator and its legacy. Acta Biotheoretica, 69:857–874, 2021.

[31] Michael Hopkins, John J. Tyson, and Béla Novák. Cell-cycle transitions: A common role for stoichiometric inhibitors. Molecular Biology of the Cell, 28(23):3437–3446, 2017.

[32] Irene Otero-Muras, Gábor Szederkényi, Antonio A. Alonso, and Katalin M. Hangos. Inducing sustained oscillations in mass action kinetic networks of a certain class. IFAC Proceedings Volumes, 45(15):475–480, 2012. 8th IFAC Symposium on Advanced Control of Chemical Processes.

[33] James E. Ferrell and Sang Hoon Ha. Ultrasensitivity part iii: cascades, bistable switches, and oscillators. Trends in Biochemical Sciences, 39(12):612–618, 2014.

[34] Floris Takens. Detecting strange attractors in turbulence. In David Rand and Lai-Sang Young, editors, Dynamical Systems and Turbulence, Warwick 1980, volume 898 of Lecture Notes in Mathematics, pages 366–381, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg.

[35] Michael E. Wall, Andreas Rechtsteiner, and Luis M. Rocha. Singular value decomposition and principal component analysis. In Daniel P. Berrar, Werner Dubitzky, and Martin Granzow, editors, A Practical Approach to Microarray Data Analysis, pages 91–109. Springer US, Boston, MA, 2003.

[36] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw. Geometry from a time series. Phys. Rev. Lett., 45(9):712–716, Sep 1980.

[37] Holger Kantz and Thomas Schreiber. Nonlinear Time Series Analysis. Cambridge University Press, 2 edition, 2003.

[38] Matthew B. Kennel, Reggie Brown, and Henry D. I. Abarbanel. Determining embedding dimension for phase-space reconstruction using a geometrical construction. Physical Review A, 45(6):3403–3411, 1992.

[39] Eugene Tan, Shannon Algar, Débora Corrêa, Michael Small, Thomas Stemler, and David Walker. Selecting embedding delays: An overview of embedding techniques and a new method using persistent homology. Chaos: An Interdisciplinary Journal of Nonlinear Science, 33(3):032101, Mar 2023.

[40] Steven L. Brunton and J. Nathan Kutz. Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control. Cambridge University Press, 2 edition, 2022.

[41] Roger Hoerl. Ridge regression: A historical context. Technometrics, 62:420–425, November 2020.

[42] Robert Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B: Statistical Methodology, 58(1):267–288, 1996.

[43] Kadierdan Kaheman, J. Nathan Kutz, and Steven L. Brunton. SINDy-PI: a robust algorithm for parallel implicit sparse identification of nonlinear dynamics. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 476(2242):20200279, 2020.

[44] Gerard Massonis, Alejandro F. Villaverde, and Julio R. Banga. Distilling identifiable and interpretable dynamic models from biological data. PLOS Computational Biology, 19(10):e1011014, 2023.

[45] Alexander B. Brummer, Agata Xella, Ryan Woodall, Vikram Adhikarla, Heyrim Cho, Margarita Gutova, Christine E. Brown, and Russell C. Rockne. Data-driven model discovery and interpretation for car t-cell killing using sparse identification and latent variables. Frontiers in Immunology, 14:1115536, 2023.

[46] Felix E. Nolet, Alexandra Vandervelde, Arno Vanderbeke, Liliana Piñeros, Jeremy B. Chang, and Lendert Gelens. Nuclei determine the spatial origin of mitotic waves. eLife, 9:e52868, May 2020.

[47] C. Gray. An analysis of the belousov-zhabotinskii reaction. Rose–Hulman Undergraduate Mathematics Journal, 3:1, 2002.

[48] Richard J. Field, Endre Koros, and Richard M. Noyes. Oscillations in chemical systems. ii. thorough analysis of temporal oscillation in the bromate-cerium-malonic acid system. Journal of the American Chemical Society, 94(25):8649–8664, 1972.

[49] E. E. Sel'kov. Self-oscillations in glycolysis: 1. a simple kinetic model. European Journal of Biochemistry, 4(1):79–86, 1968.

[50] Qamar Din and Kamran Haider. Discretization, bifurcation analysis and chaos control for Schnakenberg model. Journal of Mathematical Chemistry, 58(8):1615–1649, sep 2020.

[51] Eugene Tan, Shannon Algar, Débora Corrêa, Michael Small, Thomas Stemler, and David Walker. Selecting embedding delays: An overview of embedding techniques and a new method using persistent homology. Chaos: An Interdisciplinary Journal of Nonlinear Science, 33(3), 2023.

[52] Elizabeth Bradley and Holger Kantz. Nonlinear time-series analysis revisited. Chaos: An Interdisciplinary Journal of Nonlinear Science, 25(9), April 2015.

[53] Tim Sauer, James A. Yorke, and Martin Casdagli. Embedology. Journal of Statistical Physics, 65(3-4):579–616, 1991.

[54] Firas A. Khasawneh, Elizabeth Munch, Danielle Barnes, Max M. Chumley, İsmail Güzel, Audun D. Myers, Sunia Tanweer, Sarah Tymochko, and Melih Yesilli. Teaspoon: A python package for topological signal processing. Journal of Open Source Software, 10(107):7243, March 2025.

[55] Guillaume Tauzin, Umberto Lupo, Lewis Tunstall, Julian Burella Pérez, Matteo Caorsi, Anibal Medina-Mardones, Alberto Dassatti, and Kathryn Hess. giotto-tda: A topological data analysis toolkit for machine learning and data exploration, 2020.

[56] Rainer Hegger, Holger Kantz, and Thomas Schreiber. Practical implementation of nonlinear time series methods: The tisean package. Chaos: An Interdisciplinary Journal of Nonlinear Science, 9(2):413–435, June 1999.

[57] Linan Zhang and Hayden Schaeffer. On the convergence of the sindy algorithm. Multiscale Modeling & Simulation, 17(3):948–972, 2019.

[58] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, and Jonathan Taylor. An Introduction to Statistical Learning with Applications in Python, volume 398. eTextbooks for Students, 1st edition, 2023.

[59] Peng Zheng, Travis Askham, Steven L. Brunton, J. Nathan Kutz, and Aleksandr Y. Aravkin. A unified framework for sparse relaxed regularized regression: Sr3. IEEE Access, 7:1404–1423, 2019.

[60] Brian de Silva, Kathleen Champion, Markus Quade, Jean-Christophe Loiseau, J. Nathan Kutz, and Steven L. Brunton. PySINDy: A python package for the sparse identification of nonlinear dynamical systems from data. Journal of Open Source Software, 5(49):2104, 2020.

[61] Alan A. Kaptanoglu, Brian M. de Silva, Urban Fasel, Kadierdan Kaheman, Andy J. Goldschmidt, Jared Callaham, Charles B. Delahunt, Zachary G. Nicolaou, Kathleen

Champion, Jean-Christophe Loiseau, J. Nathan Kutz, and Steven L. Brunton. PySINDy: A comprehensive python package for robust sparse system identification. Journal of Open Source Software, 7(69):3994, 2022.

[62] Qing Qu, Ju Sun, and John Wright. Finding a sparse vector in a subspace: Linear sparsity using alternating directions. IEEE Transactions on Information Theory, 62(10):5855–5880, 2016.

[63] Leland H. Hartwell and Ted A. Weinert. Checkpoints: controls that ensure the order of cell cycle events. Science, 246(4930):629–634, 1989.

[64] Andrew W. Murray and Marc W. Kirschner. Cyclin synthesis drives the early embryonic cell cycle. Nature, 339(6222):275–280, 1989.

[65] Andrew W. Murray and Marc W. Kirschner. Dominoes and clocks: the union of two views of the cell cycle. Science, 246(4930):614–621, 1989.

[66] National Cancer Institute. Car t cells: Engineering patients' immune cells to treat their cancers. https://www.cancer.gov/about-cancer/treatment/research/car-t-cells, February 2025. Updated February 26, 2025; accessed June 23, 2025.

[67] Carl H. June and Michel Sadelain. Chimeric antigen receptor therapy. New England Journal of Medicine, 379(1):64–73, July 2018.

[68] T. Chen, M. Wang, Y. Chen, and et al. Current challenges and therapeutic advances of CAR-T cell therapy for solid tumors. Cancer Cell International, 24:133, 2024.

[69] Felix Korell, Trisha R. Berger, and Marcela V. Maus. Understanding car t cell–tumor interactions: Paving the way for successful clinical outcomes. Med, 3(8):538–564, August 2022.

[70] Kristen Newick, Sean O'Brien, Edmund Moon, and Steven M. Albelda. Car t cell therapy for solid tumors. Annual Review of Medicine, 68:139–152, January 2017. Epub 2016 Nov 17.

[71] L. R. C. Barros, B. J. Rodrigues, and R. C. Almeida. CAR-T cell Goes on a Mathematical Model. Journal of Cell Immunology, 2(1):31–37, 2020.

[72] D. Hardiansyah and C. M. Ng. Quantitative systems pharmacology model of chimeric antigen receptor t-cell therapy. Clinical and Translational Science, 12(4):343–349, July 2019. Epub 2019 Apr 20.

[73] Pranav Sahoo, Xiaohui Yang, Dan Abler, Daniel Maestrini, Vikram Adhikarla, Drew Frankhouser, Heyrim Cho, Victor Machuca, Dandan Wang, Matthew Barish, Margarita Gutova, Sergio Branciamore, Christine E. Brown, and Russell C. Rockne. Mathematical deconvolution of car t-cell proliferation and exhaustion from real-time killing assay data.

Journal of the Royal Society Interface, 17(162):20190734, January 2020. Epub 2020 Jan 15.

[74] Amanda Talkington, Clarisse Dantoin, and Rick Durrett. Ordinary differential equation models for adoptive immunotherapy. Bulletin of Mathematical Biology, 80:1059–1083, 2018.