5-31-2024

Marking estimation in petri nets using dynamic mode decomposition

Aditya Kale
New Jersey Institute of Technology

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

MARKING ESTIMATION IN PETRI NETS USING DYNAMIC MODE DECOMPOSITION

by
Aditya Kale

Petri Nets (PNs) are a well-established framework for modeling and analyzing complex systems with interacting, concurrent processes. This thesis extends traditional Petri Net methodologies by integrating Dynamic Mode Decomposition (DMD), a technique originally developed for fluid dynamics, to analyze Continuous Petri Nets (CPNs). By applying DMD to CPN marking evolution, this research constructs a reduced-order model that captures its dynamics through dynamic modes and eigenvalues, enabling prediction of future markings without detailed knowledge of transition firings or underlying deterministic models. The principal contribution of this research is extending the kit of tools available for analysis of CPN dynamics, providing insights into CPN stability and boundedness through the eigenvalues of the DMD operator. Through a series of case studies on established CPN models from PN literature, this work showcases the effectiveness of the DMD operator in forecasting and reconstructing marking evolutions. This work advances the theoretical framework of Petri Nets and opens avenues for future research in integrating data-driven analysis techniques with traditional modeling tools. This thesis is structured to first lay down the theoretical underpinnings of Petri Nets and Dynamic Mode Decomposition, followed by a discussion on the methodology and application of DMD to CPNs, culminating in a presentation of results and conclusions that highlight the potential of this integrated approach.

MARKING ESTIMATION IN PETRI NETS USING DYNAMIC MODE DECOMPOSITION

by
Aditya Kale

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering

Helen and John C. Hartmann
Department of Electrical and Computer Engineering

May 2024

APPROVAL PAGE

MARKING ESTIMATION IN PETRI NETS USING DYNAMIC MODE DECOMPOSITION

Aditya Kale

Mengchu Zhou, Dissertation Advisor — Date
Distinguished Professor, Electrical and Computer Engineering, NJIT

Dr. Marcos Netto, Committee Member — Date
Assistant Professor, Electrical and Computer Engineering, NJIT

Dr. Philip Pong, Committee Member — Date
Associate Professor, Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Aditya Kale

Degree: Master of Science

Date: May 2024

Date of Birth:

Place of Birth:

Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2024
- Bachelor of Engineering in Instrumentation Engineering,
  Vivekanand Education Society's Institute of Technology, Mumbai, India, 2020

Major: Electrical Engineering

Presentations and Publications:

R. Anchan, A. Pillay, A. Kale, A. Bhadricha and S. P. Ram, "Optimal Bipolar Lead Placement in Electrooculography (EOG): A Comparative Study with an Emphasis on Prolonged Blinks," *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Saharanpur, India, 2020,* pp. 1-7

Ashwin Pillay, Aditya Kale, Raj Anchan, Aniket Bhadricha, Sangeetha Prasanna Ram, "Real-Time Detection of Drowsiness Among Vehicle Drivers: A Machine Learning Algorithm for Embedded Systems," *arXiv:2111.03177. Available at: https://arxiv.org/abs/2111.03177*

ACKNOWLEDGMENT

I would like to express my deepest gratitude to my advisor, Dr. Mengchu Zhou, for his invaluable guidance, patience, and support throughout the course of this research. I am also immensely thankful to the members of my defense committee, Dr. Marcos Netto and Dr. Philip Pong, for their knowledge and expertise.

I am grateful for the financial support I received through my Research Assistantship at New Jersey Institute of Technology under Dr. Marcos Netto. This opportunity has been crucial in allowing me to focus fully on my research.

Finally, I extend my heartfelt appreciation to my family: my mother Archana and my father Prasad, for their endless love and support. I would not be here without them. I would also like to acknowledge my girlfriend, Elenice, who has provided me with moral support and companionship during this challenging academic journey.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF SYMBOLS

Latin

$\mathbb{N}$	Set of all natural numbers
$\mathbb{R}$	Set of all real numbers
$\mathbb{R}_+$	Set of all positive real numbers
$\mathbb{C}$	Set of all complex numbers

Greek

$\eta(t)$	Intrinsic speed of a transition t
λ_i	i^{th} Eigenvalue of the DMD operator
Λ	Diagonal eigenvalue matrix of the DMD operator
ϕ_i	i^{th} Dynamic mode of the DMD operator
Φ	Matrix of the DMD dynamic modes

LIST OF ABBREVIATIONS

PN	Petri Net
CPN	Continuous Petri Net
DMD	Dynamic Mode Decomposition
SVD	Singular Value Decomposition
sDMD	Streaming Dynamic Mode Decomposition

CHAPTER 1

INTRODUCTION

Petri Nets, conceptualized by Carl Adam Petri in his 1962 dissertation [1], serve as a robust framework for modeling discrete event systems across a wide array of disciplines. PNs excel in modeling complex systems that involve concurrency, synchronization and distributed operation. Since their inception, PNs have been widely used to model a variety of systems, from manufacturing processes to computer networks and traffic control systems.

1.1 Petri Nets

At its core, a Petri Net is a directed graph which consists of two types of nodes: *places* and *transitions*. *Arcs*, with associated weights, connect places to transitions and transitions to places. The state of a Petri Net is defined by its marking, which indicates the distribution of tokens across its places. These tokens are moved according to the net's transition enabling and firing rules, simulating the dynamic behavior of the system.

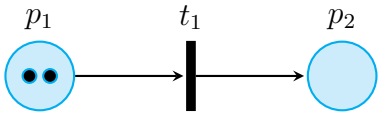


Figure 1.1 A Petri Net model with two places and a transition joining them. The marking of this PN is a vector $m = [2,\ 0]^T$.

A transition is considered enabled when each of its upstream places contains *at least* as many tokens as the weight of the connecting arc. For instance, in Figure (1.1), the transition t_1 is enabled because its input place p_1 holds 2 tokens, exceeding the required minimum of 1 token specified by the arc's weight. Upon firing, t_1 will

transfer one token from p_1 to p_2, altering the net's marking to represent the system's new state.

1.2 Continuous Petri Nets

Continuous Petri Nets (CPNs) are a relaxation on classical PNs (discrete PNs). CPNs extend the Petri Net concept to allow continuous token flow through its transitions. Its tokens are positive real-numbered quantities that change smoothly over time. CPNs are particularly useful in applications where discrete models become unwieldy due to large state spaces or where transitions occur with a continuous frequency.

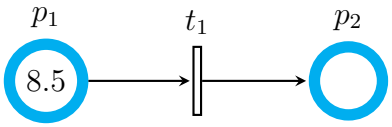


Figure 1.2 A CPN model similar to the discrete one. The marking of this PN is a vector $m = [8.5,\ 0]^T$.

Figure 1.2 shows a basic CPN where two places are connected by a continuous transition. Unlike their discrete counterparts, continuous transitions manipulate token levels at a defined rate, effectively consuming and generating tokens proportionally to their defined firing rates and arc weights. This allows for a more granular control and analysis of system dynamics.

1.3 Dynamic Mode Decomposition

Dynamic Mode Decomposition (DMD) is a numerical algorithm that identifies patterns in complex systems by decomposing the state evolution into a set of modes, each associated with a specific frequency and growth/decay rate [2]. Originally developed in the context of fluid flows, DMD can extract dynamic features from any time series data without requiring a prior model.

1.4 Application of DMD to CPN Marking Evolution

This thesis explores the application of DMD to the marking evolution of CPNs. By treating the evolution of CPN markings over time as a time series, DMD can be used to derive a reduced-order model that captures the dominant dynamics of the system. This approach provides interesting insights into the CPN model and aids in forecasting future markings based on the past ones.

1.5 Research Objectives

This thesis aims to:

- Demonstrate the utility of DMD in simplifying the analysis of CPN dynamics.
- Provide a methodology for applying DMD to Petri Nets, enhancing the traditional analysis techniques.
- Explore some of the avenues that DMD opens in CPN analysis such as marking prediction and stability analysis.

1.6 Thesis Structure

The remainder of this thesis is structured into different chapter focusing on specific topics of research:

Chapter 2: Petri Nets explores the modeling tools of Petri Nets and Continuous Petri Nets. It delves into the formal definition, the rules governing their dynamics, and the fundamental equations that evolve the states of these models.

Chapter 3: Dynamic Mode Decomposition gives a foundational basis for DMD. It presents the concepts and equations by which DMD operates and presents the Streaming DMD variant that is used during implementation.

Chapter 4: Petri Nets and DMD combines the separate ideas of CPNs and DMD to construct a framework to implement and analyze a CPN marking evolution using DMD.

Chapter 5: Implementation and Results utilizes the framework built in Chapter 4 to test on the marking evolutions of CPN models from literature. It presents the algorithm for using this framework and shows its results on the said CPN models.

Chapter 6: Conclusion concludes the thesis with a statement of objectives, summary of findings and limitations of this approach. Future directions for research in this area are also discussed.

CHAPTER 2

PETRI NETS

Carl Adam Petri introduced Petri Nets in his doctoral dissertation in 1962 as a graphical and mathematical tool for concurrent and synchronous systems [1].

2.1 Discrete Petri Nets

2.1.1 Definition

A discrete PN is formally defined as a 5-tuple $N = \langle P,\ T,\ I,\ O,\ m_0 \rangle$ such that:

$P \in \{p_1,\ p_2,\ \ldots,\ p_n\}$ is a finite, non-empty, set of places;

$T \in \{t_1,\ t_2,\ \ldots,\ t_l\}$ is a finite, non-empty, set of transitions;

$P \cap T = \emptyset$;

$I \in \mathbb{N}^{|P| \times |T|}$ is the input incidence matrix;

$O \in \mathbb{N}^{|P| \times |T|}$ is the output incidence matrix;

$m_0 : P \to \mathbb{N}$ is the initial marking.

The set of all input (output) places to a transition $t \in T$ is denoted as $\bullet t$ ($t\bullet$) and the set of all input (output) transitions to a place $p \in P$ is denoted as $\bullet p$ ($p\bullet$).

2.1.2 Dynamics

The dynamics of a discrete PN are governed by the *enabling* and *firing* of transitions:

A transition $t_j \in T$ is **enabled** at a marking m if and only if for each $p_i \in \bullet t,\ m(p_i) \geq I_{ij}$.

When an enabled transition t_j **fires**, it consumes tokens from its input places and produces tokens at its output places according to the input and output incidence matrices. Specifically, the firing of t_j results in a new marking m' where, for each $p_i \in \bullet t$,

$$m'(p_i) = m(p_i) - I_{ij} + O_{ij} \tag{2.1}$$

This firing relation is denoted as $m \xrightarrow{t_j} m'$.

2.1.3 Fundamental Equation

The changes in markings of places due to transition firings can be mathematically captured using the fundamental state equation, which computes the reachable markings from an initial marking:

$$m' = m_0 + C \cdot \sigma \tag{2.2}$$

where:

$C = O - I$ is the incidence matrix:

$\sigma \in \mathbb{N}^{|T|}$ is the firing count vector indicating the number of times each transition has fired.

2.2 Continuous Petri Nets

Continuous Petri Nets (CPNs) extend the discrete Petri Net framework to accommodate continuous changes in the marking of places, thereby making them highly suitable for systems with a large number of states. CPN models provide a good approximation of such systems as telecommunication systems, traffic systems etc. that do have a large number of states [3].

2.2.1 Definition

CPNs adapt the structure of discrete Petri Nets to handle real-numbered token counts, modifying the definitions of the incidence mappings and the initial marking:

$I \in \mathbb{R}_+^{|P|\times|T|}$;

$O \in \mathbb{R}_+^{|P|\times|T|}$;

$m_0 : P \to \mathbb{R}_+$.

These changes allow CPNs to model systems where changes to the state happen continuously and quantities are not necessarily integer-valued.

2.2.2 Dynamics

In CPNs, transitions consume and produce resources at continuous rates, which requires a modification of the enabling condition from the discrete case:

A transition $t_j \in T$ is enabled at a marking m if for each $p_i \in {}^\bullet t_j,\ m(p_i) > 0$.

The concept of enabling is extended to include a degree of enabling:

$$q(t, m) = \min_{p_i \in {}^\bullet t} \left(\frac{m(p_i)}{I_{ij}} \right), \tag{2.3}$$

where $q(t, m)$ represents the maximum possible extent to which transition t can fire given the current marking.

If $q(t, m) > 0$, the transition is considered q-enabled and can proceed to fire.

This continuous firing mechanism allows CPNs to dynamically adjust their behavior based on the fluctuating availability of resources.

2.2.3 Fundamental Equation

An enabled transition can fire from a marking m at a rate $0 \leq \alpha \leq q(t, m)$ leading to a new marking m' governed by the equation:

$$m' = m + \alpha C \tag{2.4}$$

A reachable marking m' from m_0 can thus be estimated directly from the fundamental state equation (2.2) [4][5].

In the continuous case, the firing count vector, $\sigma \in \mathbb{R}_+^{|T|}$, is updated over time by summing the rates at which each transition fires, reflecting the continuous throughput of the system.

$$\sigma_j = \sum_{h=1}^{k} \alpha_{j,h} \tag{2.5}$$

where σ_j is the firing count of a transition t_j, $\alpha_{j,h}$ is the firing rate of the transition t_j at the h^{th} firing event, and k is the number of firing events considered during the simulation.

2.2.4 Infinite Server Semantics

In the modeling of CPNs, the concept of *server semantics* significantly influences the behavior of transitions. The two predominant types are *finite server semantics* and *infinite server semantics* [3]. Under finite server semantics, the firing rate α_j of a transition t_j is determined solely by its enabling degree $q(t_j, m)$. Contrastingly, in infinite server semantics, each transition is associated with an intrinsic speed $\eta(t_j)$, enhancing its throughput capacity. Consequently, the firing rate under infinite server semantics is the product of the transition speed and the enabling degree [6]:

$$\alpha_j = \eta(t_j) \cdot q(t_j, m) \tag{2.6}$$

This mechanism allows for more nuanced modeling in scenarios where transition activity is influenced by the its inherent properties of the physical process it represents well as the system's state.

2.2.5 Boundedness

Boundedness of a traditional Petri Net is a property that defines a marking m' such that every marking m'' reachable from m_0 does not exceed m' in any of its places [7]. If a PN is not bounded, the number of tokens in its places continue to grow infinitely.

Determining Boundedness For continuous Petri Nets, boundedness is often analyzed by examining the reachability space of the net [5]. Given a CPN, N, and an initial marking, m_0, the reachability space of N, RS(N, m_0), is a convex set [8] of all markings reachable from m_0 through a finite set of transition firings at rates encapsulated in σ. N is considered bounded if and only if, for all $b_i \in \mathbb{R}_+$and $p_i \in P$, there exists $m \in \mathrm{RS}(N,\ m_0),$ such that $m(p_i) \leq b_i$.

This formalism implies that for a CPN to be considered bounded, the markings within RS must be constrained by maximum values for each place in the net. If any place within the net can exceed its bound through some finite sequence of firings, then the net is considered unbounded.

2.3 Survey on PN Analysis Literature

In the exploration of PN analysis techniques, [9] introduces critical methodologies for performance analysis in manufacturing systems by leveraging the relationship between transition firings and the fluctuation of buffer contents in PNs. This work provides essential insights into production processes, offering both theoretical and practical solutions to deduce transition firing frequencies.

On the topic of system observability, [10] delves into the observability challenges in CPNs governed by infinite server semantics. The study establishes a novel approach to transforming the inherently nonlinear observability problem into a manageable linear format under specific conditions. This transformation facilitates the development of algorithms capable of estimating unobservable markings from observable data.

Further extending the analysis of CPNs, [11] explores challenges and strategies in managing CPNs using Model Predictive Control (MPC). This study primarily focuses on the control problem of transitioning from an initial state to a desired steady state by minimizing a cost function. The authors present a sampled, discrete-time model of a CPN, maintaining essential characteristics of the continuous model, for the application of MPC. Through a discussion on feasibility and asymptotic stability, it is proved that, for CPNs controlled via MPC, the former is guaranteed while the latter is not.

Lastly, [5] contributes to the formal analysis of CPNs by proposing an automated framework that ensures the safety and compliance of CPN operations with predefined Linear Temporal Logic conditions. This framework focuses on verifying the unreachability of undesirable states from specific initial conditions and validating the adherence of system trajectories to safety and operational standards under infinite server semantics.

CHAPTER 3

DYNAMIC MODE DECOMPOSITION

Dynamic Mode Decomposition (DMD) is a pivotal numerical method in the field of fluid dynamics, and its application has rapidly expanded across various domains of science and engineering. Originally conceptualized as a method to dissect complex flow datasets, DMD facilitates the decomposition of high-dimensional data into a series of dynamically significant modes without the necessity for a priori knowledge of the underlying dynamics. This capability makes DMD an invaluable tool for both theoretical research and practical applications, where understanding the fundamental temporal and spatial behavior of complex systems is crucial. By leveraging singular value decomposition (SVD), DMD isolates patterns and frequencies that describe the evolution of dynamic systems, providing insights that are both profound and actionable. This chapter delves into the mathematical foundations of DMD and explores its algorithmic implementation.

3.1 Mathematical Foundation

3.1.1 Data Matrix and Linear Operator

Given a sequential set of state vectors $\{x_1,\ x_2,\ \ldots,\ x_l\}$, where each $x_k \in \mathbb{R}^n$ represents the system state at the k^{th} time step, DMD operates by arranging this dataset into two matrices:

- Data matrix $X = [x_1,\ x_2,\ \ldots,\ x_{l-1}]$ which includes all except the last vector, and
- Shifted data matrix $X' = [x_2,\ x_3,\ \ldots,\ x_l]$ which includes all except the first vector.

The objective of DMD, then, is to find a linear operator $A \in \mathbb{R}^{n \times n}$ that best describes the evolution of the system from each state in X to the subsequent state in X'.

$$X' = AX. \tag{3.1}$$

This equation models the dynamics of the system by approximating the transformation A, which, when applied to X, results in X'. In essence, A acts as a propagator of the system's state from one time step to the next.

The DMD operator A computed here minimizes the cost function [12]

$$J = \sum_{k=1}^{l} ||x'_k - Ax_k||^2 = ||X' - AX||_F^2 \tag{3.2}$$

where, $|| \cdot ||$ is the vector ℓ^2-norm and $|| \cdot ||_F$ is the matrix Frobenius norm.

3.1.2 Singular Value Decomposition

In systems characterized by a large number of states n in their state vectors, directly computing the dynamic modes and the operator A can become computationally intensive. This problem is addressed by SVD which provides an optimal low-rank approximation to X. The SVD-based DMD algorithm is more numerically stable and generally accepted as the defining standard DMD algorithm [13].

Upon constructing the matrix X, the next step in DMD involves computing its reduced SVD:

$$X = U\Sigma V^* \tag{3.3}$$

where,

$U \in \mathbb{R}^{n \times r}$ is a matrix whose columns are the left singular vectors of X, representing the orthonormal basis for the range of X,

$\Sigma \in \mathbb{R}^{r \times r}$ is a diagonal matrix containing the singular values of X, which quantify the information content along each corresponding singular vector,

$V \in \mathbb{R}^{m \times r}$ contains the right singular vectors of X, representing the orthonormal basis for its co-range; V^* denotes the complex-conjugate transpose of V and,

r is the rank of X ($r < n$), indicating the non-zero singular values.

SVD simplifies the computation of A by projecting it onto a lower-dimensional subspace defined by the significant singular vectors:

$$\tilde{A} = U^* X' V \Sigma^{-1} \tag{3.4}$$

where,

$U^* X' V$ represents the core interactions between the modes captured in X and their advancements in X' and,

Σ^{-1} scales these interactions inversely by the singular values, normalizing the influence of each mode based on its information content.

3.1.3 Eigendecomposition and Reconstruction

The eigendecomposition of $\tilde{A}$ reveals the dominant dynamics of the system:

$$\tilde{A} W = \Lambda W. \tag{3.5}$$

Here, W contains the eigenvectors and Λ is a diagonal matrix consisting of the corresponding eigenvalues of $\tilde{A}$. These metrics allow us to recover the full state system dynamics in a computationally efficient manner [14].

Each eigenvalue λ_i and its corresponding eigenvector w_i define a dynamic mode, which characterizes the behavior of the system over time. These modes can be computed for the full system by projecting the eigenvectors back into its higher-dimensional space:

$$\hat{\phi}_i = U w_i \tag{3.6}$$

With the dynamic modes $\Phi = [\phi_1,\ \phi_2,\ \ldots,\ \phi_r]$, we can reconstruct an approximation of the system's behavior at any time step k:

$$x_k = \Phi e^{\Omega k} b \tag{3.7}$$

where,

$b = [b_1,\ b_2,\ \ldots,\ b_r]^T$ is the initial amplitude vector of each dynamic mode, which weigh the contribution of each mode at the initial time,

$\Omega = \text{diag}(\frac{1}{\Delta k}[\ln(\lambda_1), \ln(\lambda_2), \ldots, \ln(\lambda_r)])$ scales the logarithm of each eigenvalue by the time step Δk, converting the growth rates to a format that applies over the discrete time steps used in data collection.

3.2 Survey on DMD Areas of Application

DMD has been extensively applied across various fields since its initial development, which primarily addressed challenges in fluid dynamics. [2] demonstrated DMD's efficacy in capturing dominant flow structures in fluid dynamics, beginning with the simulation of flow over a square cavity where the said flow exhibits self-sustained oscillations.

The versatility of DMD extends to experimental data analysis, as seen in further usages in [2] where DMD is applied to fluid dynamics experiments involving complex setups such as a U-shaped thin steel frame supporting a flexible latex membrane, and the dynamics of a jet flowing between two cylinders.

The adaptation of DMD to real-time systems led to the development of Online DMD [12]. This variant updates the DMD operator on the fly, allowing for continuous refinement of the model as new data becomes available. It is employed to effectively analyze pressure fluctuations in a wind tunnel experiment, providing insights into the flow dynamics over a flat plate, illustrating the method's applicability to aerodynamic studies.

DMD has also been adapted for control systems through the introduction of dynamic mode decomposition with control (DMDc) [15]. This extension incorporates external control inputs into the DMD framework, enhancing its predictive capabilities for controlled systems. Demonstrations on various systems, from unstable linear models with controllers to large-scale stable systems, underscore DMDc's potential in control theory and engineering applications.

The development of multi-resolution DMD (mrDMD) [16] represents an evolution of the technique that enables the separation of modes into different spatio-temporal scales. This method has proven effective in decomposing video data into spatial and temporal features, facilitating the analysis of complex, multi-scale phenomena.

3.3 Streaming DMD

Streaming dynamic mode decomposition (sDMD) is an advanced variant of the standard DMD technique, designed to efficiently process and analyze data streams in real-time [17]. This approach is especially valuable in scenarios where data is continually generated or updated, such as during ongoing experimental measurements or in continuous simulation environments. It adapts the standard DMD methodology to accommodate continuously updating datasets without the need to recompute the entire DMD analysis from scratch.

In the sDMD framework, the DMD operator and the associated dynamic modes and eigenvalues are updated incrementally at each time step as new data becomes available. This process allows for the real-time prediction of future states of the system based on the most current system information. The predictive model at any time step k is given as:

$$x_{k+1} = \Phi_k \cdot \Lambda_k \cdot b_k \tag{3.8}$$

where, Φ_k is the matrix of dynamic modes at step k and Λ_k is the diagonal matrix of eigenvalues at k. The vector b_k is given as

$$b_k = \Phi_k^\dagger \cdot x_k \tag{3.9}$$

where $\Phi_k^\dagger$ is the Moore-Penrose pseudoinverse of Φ_k. b_k encapsulates the current state vector's influence on future dynamics.

3.3.1 Application Example: Predicting Lorenz System Dynamics

This subsection demonstrates the application of streaming DMD, as outlined in Algorithm 2, to predict the evolution of the Lorenz system– a set of three interlinked ordinary differential equations known for their chaotic behavior [18]. The Lorenz system serves as an ideal testbed for this methodology due to its complex, non-linear dynamics that are sensitive to initial conditions.

The model of the Lorenz system is defined by the following differential equations:

$$\begin{aligned}
\frac{\mathrm{d}x}{\mathrm{d}t} &= \sigma(y - x) \\
\frac{\mathrm{d}y}{\mathrm{d}t} &= x(\rho - z) - y \\
\frac{\mathrm{d}z}{\mathrm{d}t} &= xy - \beta z
\end{aligned}$$

The choice of the Lorenz system also illustrates a theoretical framework where any system of ODEs can be transformed into equivalent Continuous Petri Nets, maintaining the same dynamical properties [19]. Although this transformation is not performed here, the implication is significant: it suggests the potential for applying the same DMD analysis to any CPN representation of ODEs as one would to the ODE system itself.

For this demonstration, the initial state of the Lorenz system is set to $x_0 = [1, 1, 1]$, with parameters $\rho = 28$, $\sigma = 10$, and $\beta = \frac{10}{3}$.

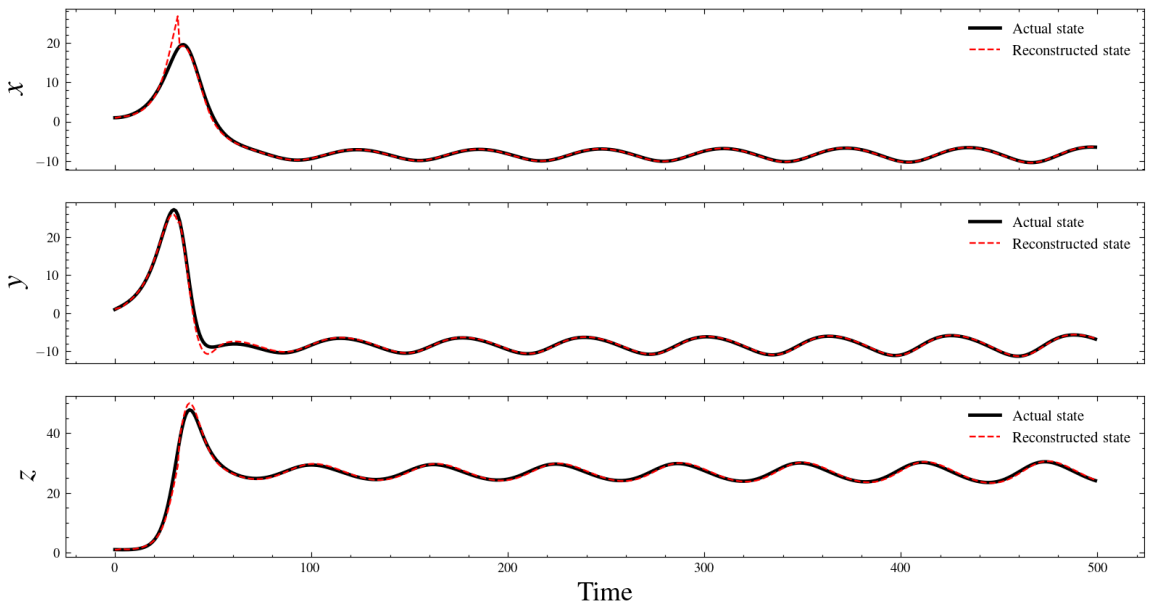


Figure 3.1 Comparison of actual and sDMD-reconstructed states of the Lorenz System over time.

Figure 3.1 shows the Lorenz system simulated for 500 time steps with the time difference between two consecutive steps being 0.6. The state reconstruction by sDMD for all states closely follows the actual states over time.

This example not only validates the capabilities of streaming DMD in capturing complex system dynamics but also sets a precedent for its application in converting and analyzing ODE-based systems as CPNs.

CHAPTER 4

PETRI NETS AND DMD

This chapter explores an innovative approach to understanding the dynamic behavior of Petri Nets, specifically focusing on continuous PNs, through the framework of Dynamic Mode Decomposition, a method originally developed for fluid dynamics and recently applied to a broader range of systems exhibiting complex, time-varying behaviors.

4.1 Markings

The state of a Petri Net is captured by its marking, representing the distribution of tokens across its places. The evolution of these markings through transitions leads to the formation of a directed graph called a reachability graph [20], a key concept for understanding the potential behaviors of the system.

For continuous Petri Nets, reachability involves macro-markings [3], which group states with identical sets of marked places. This concept simplifies the complexity inherent in continuous systems, where the exact number of tokens may vary smoothly.

4.1.1 Markings as Time Series

One novel aspect of this approach is conceptualizing the sequence of markings as time series data:

$$M = [m_0,\ m_1,\ \ldots,\ m_l]$$

where, each $m_i \in \mathbb{R}_+^{\kappa}$ denotes the state of markings at discrete time intervals, encompassing l total steps. This representation allows us to apply time series analysis techniques, such as DMD, to extract dynamic features and predict future states of the Petri Net.

Discrete Time Series For the effective analysis of markings that change continuously over time, they need to be discretized into a sequence of time-stamped samples:

- **Sampled Computation:** Markings are computed at fixed intervals to form a discrete time series, where m_k specifically denotes the marking at the k^{th} time step. This contrasts with traditional Petri Net notation, where markings are typically identified in relation to their sequence from the initial marking. Each marking is derived from the previous marking, allowing the firing count vector to be computed as the firing rate at each step:

$$\sigma_j = \alpha_{j,k} \tag{4.1}$$

- **Initial Marking Representation:** The initial marking, m_0, corresponds to the state of the CPN at the start of the simulation period.

- **Time Interval Δk:** The selection of the time interval Δk between successive markings is crucial. The resolution of the time series—and consequently, the detail and sensitivity of the dynamic analysis—depends heavily on the choice of Δk. Shorter intervals capture more granular changes, while longer intervals tend toward making the model into a discrete PN.

This approach of sampling continuous markings into discrete intervals effectively adapts the continuous characteristics of CPN markings for analysis by DMD.

Marking Arrangement In Dynamic Mode Decomposition, the matrix $M \in \mathbb{R}_{+}^{n \times l}$ representing the markings over time steps is arranged into two matrices: the data matrix X and the shifted data matrix X'. When inputted to DMD, these matrices facilitate the computation of a transformation operator $D \in \mathbb{R}^{r \times r}$, capturing the essential dynamics of the marking evolution. The eigendecomposition of D gives us the matrix $\Phi \in \mathbb{C}^{n \times r}$ whose columns represent the dynamic modes, and the diagonal matrix of eigenvalues $\Lambda \in \mathbb{C}^{r \times r}$, indicating the growth or decay rates of these modes.

4.2 CPN Dynamics

The dynamic modes derived from DMD provide a succinct representation of the dominant dynamics influencing the marking evolution in a CPN. These modes, when analyzed, reveal the system's behavior over time, including stability and boundedness.

4.2.1 Boundedness

Through the DMD analysis of marking evolution, the eigenvalues obtained can serve a purpose in determining if the CPN is bounded or not. Let $\lambda_i = \Lambda_{i,i}$ be the eigenvalue associated with the i^{th} dynamic mode $\phi_i = \Phi_{\cdot,i}$:

> $\mathrm{Re}(\lambda_\mathbf{i}) > \mathbf{0}$, suggests that the corresponding dynamic mode grows exponentially, indicating that the system may be potentially **unbounded**. If any place associated with this mode increases without bounds, it implies that the CPN allows an infinite accumulation of tokens.
>
> $\mathrm{Re}(\lambda_\mathbf{i}) \leq \mathbf{0}$, indicates a mode that decays, suggesting that the markings stabilize or diminish over time, implying that the CPN is **bounded**.

4.2.2 Forecasting

Forecasting in Continuous Petri Nets (CPNs) typically requires detailed knowledge of the sequence and magnitude of transition firings to predict future markings based on the fundamental state equation, i.e., (2.2). This traditional approach, although effective, can become difficult to compute in systems with a very large number of places.

Dynamic Mode Decomposition offers a powerful alternative by enabling direct prediction of future markings from the initial one. With the DMD operator D, we can compute future states m_k directly from the initial marking m_0 via (3.7), effectively bypassing the need to track each transition firing explicitly. This method leverages

the dynamics encoded within the DMD modes and eigenvalues to encapsulate the entire system's behavior over time.

Streaming DMD for Real-Time Updates Streaming DMD extends this functionality by continuously updating the DMD operator with new markings during the system's operation. This adaptive approach allows the model to incorporate the latest markings and improve the accuracy of future marking predictions. By updating the operator based on the most recent marking m_k, Streaming DMD can predict the next marking m_{k+1} in real-time using (3.8), thus maintaining high predictive fidelity.

CHAPTER 5

IMPLEMENTATION AND RESULTS

This chapter presents the implementation details of the Streaming Dynamic Mode Decomposition applied to Continuous Petri Nets and discusses the results obtained from various test cases. The algorithm's efficacy in predicting future markings and its dynamic adaptability in real-time data processing are evaluated through hypothetical and practical scenarios.

5.1 Implementation

This section describes the technical realization of the proposed approach, which is primarily conducted using Python. The implementation leverages widely used open-source libraries: NumPy for numerical data handling and Matplotlib for visualization purposes. The process is bifurcated into two major components:

- **Construction and Simulation of the CPN:** This part involves programming a Continuous Petri Net (CPN) using incidence matrices to build the marking evolution matrix M.
- **Application of Streaming Dynamic Mode Decomposition:** This involves applying the Streaming DMD algorithm to matrix M for real-time predictions and computation of the DMD operator, which elucidates the CPN model's dynamics.

5.1.1 CPN Program

The CPN program generates the time-series data required for subsequent analysis with DMD. Algorithm 1 outlines the procedure to compute the marking evolution matrix M, encapsulating the dynamics of a CPN:

Algorithm 1 Algorithm to compute the matrix M

Require: Set of places P, Set of transitions T, Pre-incidence matrix I, Post-incidence matrix O, initial marking at the first time step m_0, transition speed vector η.

Ensure: Marking evolution matrix M, where $m_0 \in \mathbb{R}_+^P$, $\eta \in \mathbb{R}_+^T$.

procedure SIMULATECPN(l, Δk)
    Initialize $M \leftarrow [m_0^T]$
    Compute $C \leftarrow O - I$
    for $k = 2$ to $l - 1$ **do**
        $\sigma \leftarrow$ GETENABLINGDEGREE($M[:, k]$, η)
        $m \leftarrow \Delta k(C \cdot \sigma)$
        $m \leftarrow$ MAXIMUM(0, m)
        APPENDCOLUMNTOMATRIX(M, m^T)
    end for
    return M
end procedure
function GETENABLINGDEGREE(m, η)
    $\sigma \leftarrow [\infty]_{|T| \times 1}$
    for $t = 1$ to $|T|$ **do**
        ratios $\leftarrow$ SUBSTITUTEWHERE($I[:, t] > 0$, $m / I[:, t]$)
        $\sigma[t] \leftarrow$ MIN(ratios) $\cdot \eta[t]$
    end for
    return σ
end function

Algorithm 1 details the simulation steps required to compute matrix M, from initializing the first marking to calculating the enabling degrees and updating the matrix for each time step. The enabling degrees are determined by the function *GetEnablingDegree*, which computes the transition firing rates based on the current

marking and predefined transition speeds. This implies that the program inherently models infinite server semantics. To model finite server semantics for any transition, t_j, simply set $\eta(t_j) = 1$. Δk determines the simulation step size, which affects the granularity of the simulation.

5.1.2 sDMD Program

The implementation of the sDMD algorithm for Petri Net markings is designed to dynamically predict future states of the system based on its initial conditions and subsequent evolution. It makes use of the streaming algorithm developed in [21].

Operational Mechanics of sDMD At each time step k, sDMD performs the following operations for a CPN:

1. **Data Assimilation:** Integrate the new marking data into the existing DMD model.

2. **Model Update:** Recalculate the dynamic modes and eigenvalues to reflect the new state of the system.

3. **Prediction:** Use the updated model to forecast the next marking m_{k+1}.

   Algorithm 2 outlines the step-by-step procedure.

Algorithm 2 Streaming DMD for Continuous Petri Net Markings

Require: Marking evolution matrix $M \in \mathbb{R}_{+}^{n \times l}$.

Ensure: Prediction of future markings X_p updated dynamically.

procedure StreamingRun(l)
  Initialize $X \leftarrow M[:, 1 : l-1]$ and $X' \leftarrow M[:, 2 : l]$
  Initialize $X_p \leftarrow [M[:, 1]]$
  Initialize $\varrho \leftarrow [\,]_{n \times 0}$ ▷ Matrix of residuals over time
  Compute $D \leftarrow$ UpdateDMDModel($X[:, 1]$, $X'[:, 1]$)
  for $k = 2$ to l **do**
    $\Phi,\ \Lambda \leftarrow$ ComputeModes(D)
    $b \leftarrow$ ComputeAmplitudeVector($X[:, k-1],\ \Phi$)
    $m_p \leftarrow$ PredictMarking($\Phi,\ \Lambda,\ b$)
    $r \leftarrow X'[:, k] - m_p^T$
    AppendColumnToMatrix($X_p,\ m_p^T$)
    AppendColumnToMatrix($\varrho,\ r$)
    $D \leftarrow$ UpdateDMDModel($X[:, k],\ X'[:, k]$)
  end for
  return X_p, $E[\varrho]$
end procedure
function ComputeAmplitudeVector($m,\ \Phi$)
  $b \leftarrow \Phi^{\dagger} \cdot m$
  return b
end function

Functions *ComputeModes* and *UpdateDMDModel*, implemented in [21], are integral to the sDMD process, facilitating the dynamic adaptation of the model to new data. Algorithm 2 not only predicts future markings but also updates the DMD operator in real-time to refine predictions continuously. During the streaming run,

the residual vector, r, is computed at each time step to gauge how well the sDMD algorithm is estimating the next marking. At the end of the streaming run, the mean of the residuals over time, $E[\varrho]$, provides a metric to evaluate the accuracy of the predictions. The final DMD operator, computed at the last iteration, captures the comprehensive dynamics of CPN marking evolution, essential for reconstructing the entire sequence to evaluate the model's accuracy.

Algorithm 3 Reconstruction of CPN Marking Evolution

Require: Initial CPN marking at the first time step m_0, DMD operator D.

procedure ReconstructMarkingEvolution(l)
    Initialize $M_r \leftarrow [m_0^T]$
    Compute $\Phi,\ \Lambda \leftarrow$ ComputeModes(D)
    Compute $b,\ \Omega \leftarrow$ Projection($m_0,\ \Lambda$)
    for $k = 2$ to l **do**
        $b_k \leftarrow be^{\Omega k}$
        $m_r \leftarrow \Phi \cdot b_k$
        AppendColumnToMatrix($M_r,\ m_r$)
    end for
    return M_r
end procedure
function Projection(m_0, Φ, Λ)
    $b \leftarrow \Phi^{\dagger} \cdot m_0$
    $\Omega \leftarrow \ln(\Lambda)$
    return b, Ω
end function

5.2 Testing Results

The efficacy of the proposed approach is evaluated through simulations conducted on four distinct Continuous Petri Net (CPN) models, each exhibiting unique dynamic behaviors. These models are implemented and simulated using the specialized Python framework developed for this thesis. Streaming DMD is applied to the temporal matrices representing the marking evolutions across these models.

The results are systematically presented in three separate subplots for each model:

- **Marking Evolution:** This plot displays the actual progression of markings within the CPN across all places.
- **Prediction and Reconstruction:** This includes marking evolution plots of all places
    1. predicted during the Streaming run (depicted as a solid blue line) and
    2. reconstructed after the end of the Streaming run from the final DMD operator (shown as a dashed red line).
- **Eigenvalue Analysis:** The third subplot visualizes the eigenvalues derived from the DMD operator obtained in the final step of the Streaming algorithm. These eigenvalues are plotted on the complex plane to illustrate both their real and imaginary components.

The temporal resolution, Δk, and the total number of simulation steps, l, are carefully selected through an iterative process of experimental adjustments. This ensures that the chosen parameters effectively capture the nuanced changes in markings and encompass all critical dynamics within the simulation period.

5.2.1 Traffic Flow in Road Sections

This case study utilizes a Continuous Petri Net (CPN) model of a three-section traffic system, as described in [22]. This model simulates the flow of vehicular traffic across a stretch of road, delineating each road section into distinct CPN components. The model comprises three interconnected sections, each represented by a trio of places within the network. For example, Section 1 consists of places p_1, p_2, and p_3, alongside transitions t_1 and t_2 as illustrated in Figure 5.1. t_1 represents the inflow of vehicles into the section, and t_2 the outflow from the section. The marking of p_1 quantifies the number of vehicles currently within the section. p_2 functions as a capacity limiter, restricting the maximum number of vehicles that can be accommodated in the section at any given time. p_3 regulates the traffic flow rate.

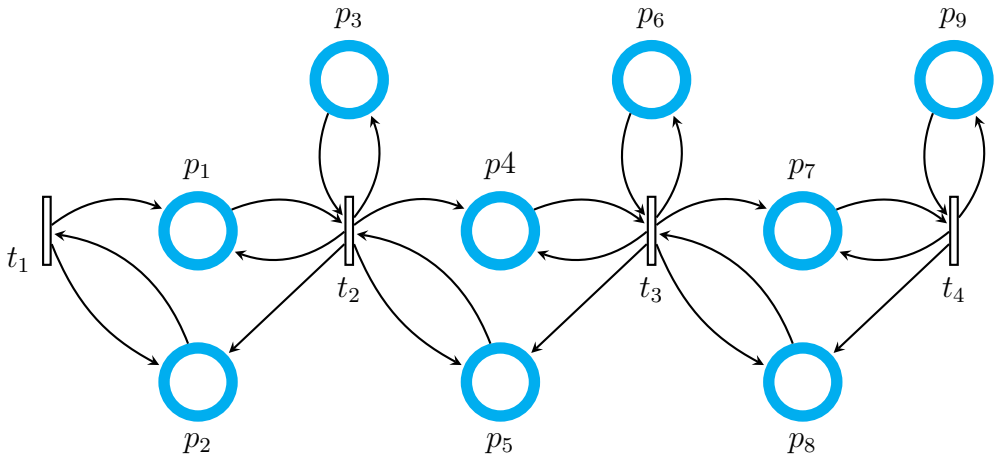


Figure 5.1 Three sections.

The transitions of this CPN model operate under infinite server semantics. In this example, $\eta(t_1) = \eta(t_2) = \eta(t_3) = \eta(t_4) = 5$. The arc weights are as given by the

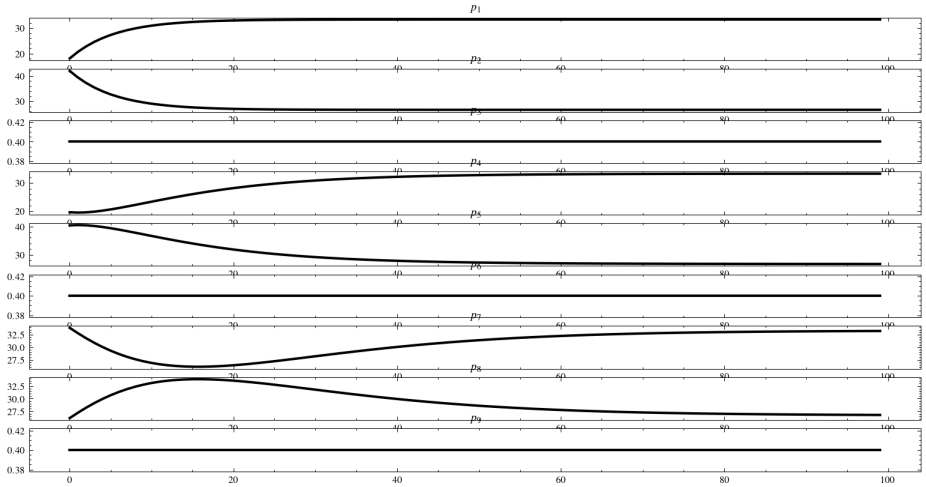
input- and output-incidence matrices:

$$I = \begin{bmatrix} 0 & q1 & 0 & 0 \\ r1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & q2 & 0 \\ 0 & r2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & q3 \\ 0 & 0 & r3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; O = \begin{bmatrix} 1 & q1-1 & 0 & 0 \\ r1-1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & q2-1 & 0 \\ 0 & r2-1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & q3-1 \\ 0 & 0 & r3-1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

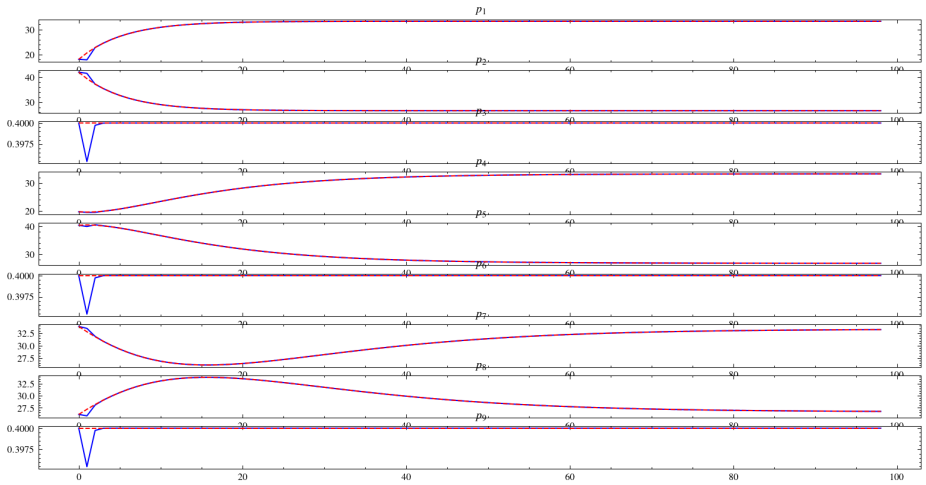
where, $q1 = q2 = q3 = 100$ and $r1 = r2 = r3 = 80$. Table 5.1 lists down all the nodes of this CPN and what they represent in the road sections being modeled.

Table 5.1 Elements of the Road Sections CPN

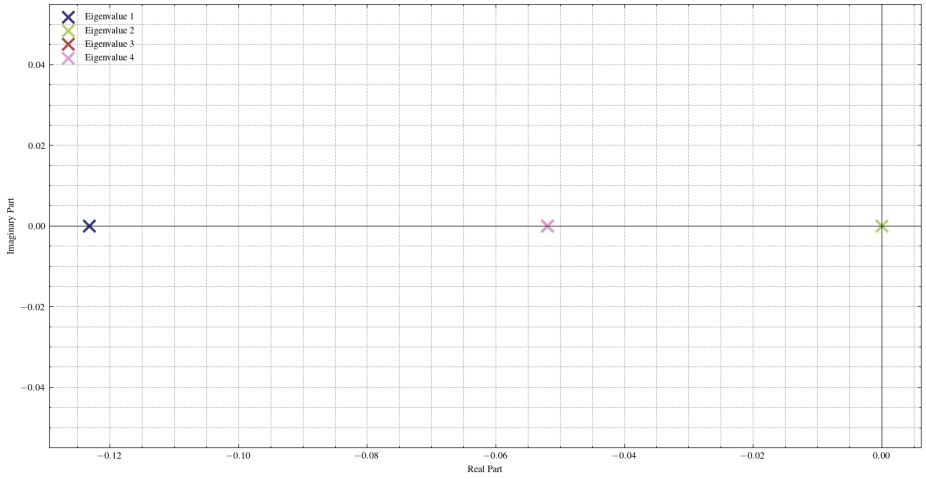
CPN element	Meaning
p_1	Vehicles in road section 1
p_2	Capacity limiter for section 1
p_3	Traffic rate regulator for outflow from section 1 and into section 2
p_4	Vehicles in road section 2
p_5	Capacity limiter for section 2
p_6	Traffic rate regulator for outflow from section 2 and into section 3
p_7	Vehicles in road section 3
p_8	Capacity limiter for section 3
p_9	Traffic rate regulator for outflow from section 3
t_1	Inflow into section 1
t_2	Outflow from section 1 into section 2
t_3	Outflow from section 2 into section 3
t_4	Outflow from section 3



(a) Marking Evolution of the traffic sections CPN model.



(b) Streaming prediction and DMD reconstruction of the marking evolution.



(c) Eigenvalues of the DMD operator.

Figure 5.2 DMD analysis of the road sections CPN model. $m_0 = [15,\ 45,\ 0.4,\ 20,\ 40,\ 0.4,\ 35,\ 25,\ 0.4]^T$, $l = 100$, $\Delta k = 1.5$.

The actual marking evolution of the CPN model displays straightforward and stable dynamics (Figure 5.8a). This simplicity in behavior facilitates the sDMD algorithm's capacity to quickly synchronize with the system's dynamics. After an initial adaptation phase, it delivers consistently accurate predictions throughout the simulation period. The reconstructed marking evolution precisely mirrors the actual trajectory of the system.

The analysis of the eigenvalues highlights the system's bounded and stable nature. From Figure 5.8c, the eigenvalues are $\lambda_1 = -1.23e-1$, $\lambda_2 = -2.88e-15$, and $\lambda_3 = \lambda_4 = -5.197e-2$. All the eigenvalues in this instance are real and less than 0, indicating that each dynamic mode of the system is stable, without any oscillatory dynamics and the CPN is bounded. In this example, the marking evolution exhibits 5 eigenvalues, suggesting that the DMD operator D is a 5×5 matrix. With the rank of M being 4, the sDMD algorithm effectively captures all dominant dynamics with just one additional dimension than the inherent rank of M.

For this example, the mean residual vector is $E[\varrho] = [0.1575,\ -0.1501,\ 4.9e-5,\ 0.144,\ -0.1367,\ 4.9e-5,\ -0.0031,\ 0.01,\ 4.9e-5]^T$. This shows that the streaming predictions follow the actual evolution very closely.

This example serves to demonstrate the efficacy of sDMD in handling CPN models characterized by non-complex dynamical patterns. This is significant because despite the original purpose of it being to model the extremely complex dynamics of fluid flow, DMD still works well with systems that exhibit very little in the way to dynamical patterns over time.

5.2.2 CPN Examples from Literature

The following two example are derived from [23].

Example 1 This CPN model, illustrated in Figure 5.3, employs infinite server semantics to govern transition firings. It is designed in a way to never reach a steady

state in places p_1 through p_4. p_5 and p_6 act as firing speed regulators. In this setup, $\eta(t_1) = \eta(t_5) = 1$, $\eta(t_2) = \eta(t_3) = 10$ and $\eta(t_4) = 20$.

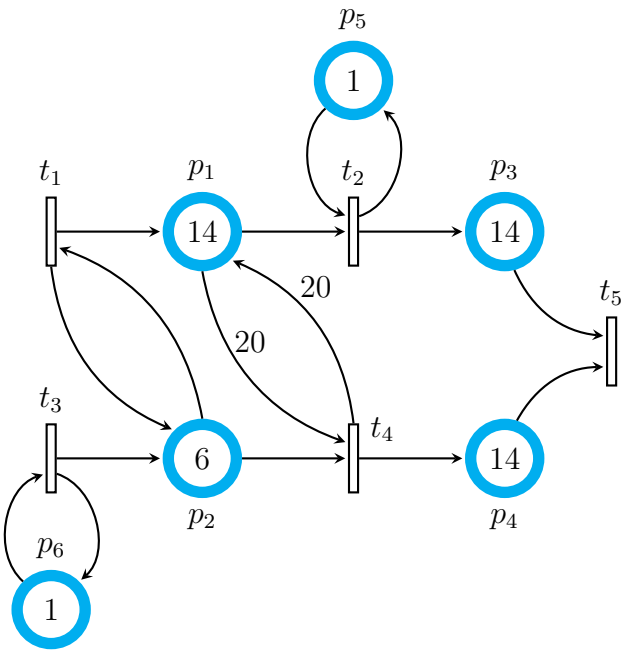
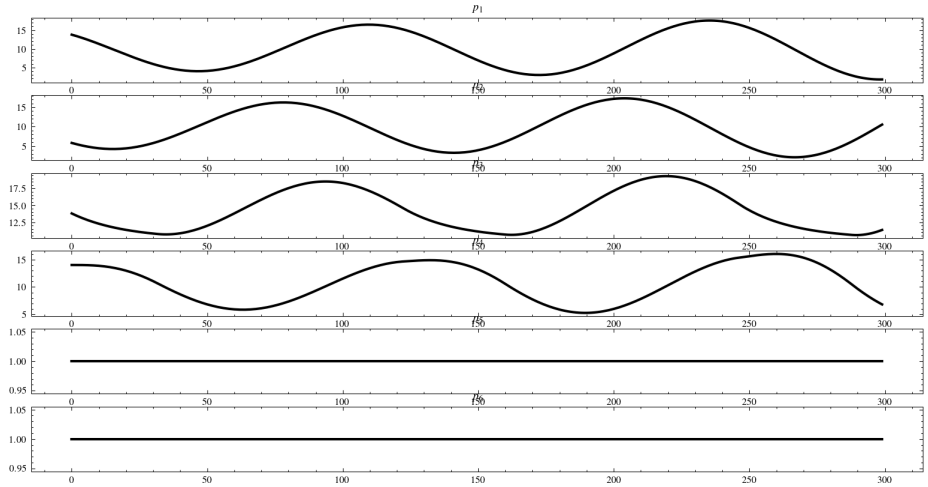
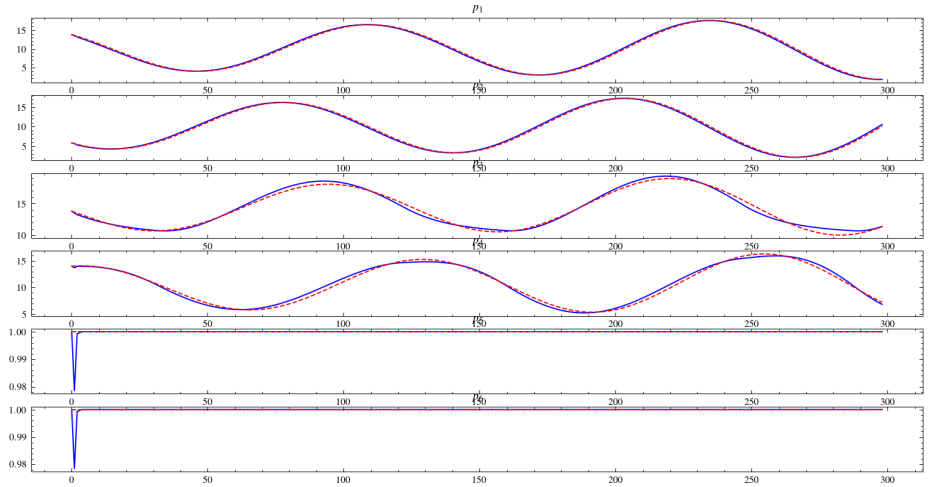


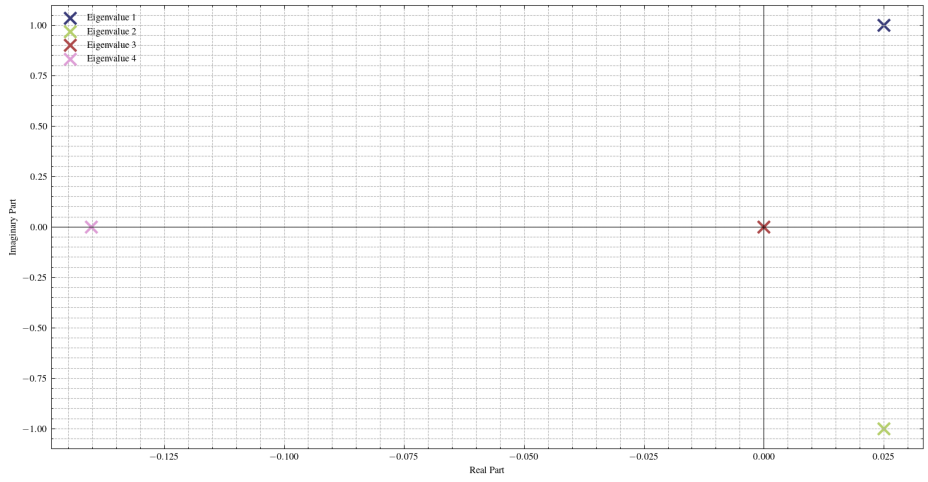
Figure 5.3 CPN example.



(a) Marking Evolution of the CPN model.



(b) Streaming prediction and DMD reconstruction of the marking evolution.



(c) Eigenvalues of the DMD operator.

Figure 5.4 DMD analysis of the CPN model. $m_0 = [14,\ 6,\ 14,\ 14,\ 1,\ 1]^T$, $l = 300$, $\Delta k = 0.05$.

This CPN model is characterized by oscillatory dynamics within places p_1 through p_4, which are the focal places of interest. The mean residual vector from the streaming predictions, $E[\varrho] = [-0.039,\ 0.016,\ -0.011,\ -0.027,\ 3.7e-5,\ 3.7e-5]^T$, indicates a close match between the streaming DMD predictions and the actual system behavior, as evident from the comparison between Figure 5.4a and Figure 5.4b. Both the reconstructed and actual marking evolutions demonstrate more complex dynamics compared to the previous example, making them suitable candidates for analysis using cross-correlation techniques. The Spearman correlation coefficients for places p_1 through p_4 are exceptionally high, at 1, 1, 0.985, and 0.995 respectively. These values underscore the precision with which the DMD operator D captures the underlying dynamics of the CPN.

Further analysis of the eigenvalues of D provides insights into the stability and bounded nature of the CPN. From Figure 5.4c, $\lambda_1 = 2.5e-2+j0.99$, $\lambda_2 = 2.5e-2-j0.99$, and $\lambda_3 = 7.9e-14$, and $\lambda_4 = -1.4e-1$. λ_1 and λ_2 are complex conjugate pairs with a near 0 positive real part: this suggests the presence of oscillatory modes with slowly growing, sustained oscillations. This behavior arises due to the $p_1 - t_4$ loop and the $t_1 - p_2$ loop in the CPN (Figure 5.3). The real parts of λ_1, λ_2, and λ_3 are greater than 0, suggesting that the system is unbounded. This examination highlights the effectiveness of streaming DMD in capturing and predicting the oscillatory and unbounded behavior inherent in this CPN example.

Example 2 Similar to the previous scenario, this CPN, depicted in Figure 5.5, operates under infinite server semantics. However, it differs notably in its dynamics; it does not display the sinusoidal marking evolutions observed in the earlier example and exhibits a markedly more stable behavior. In this instance, $\eta(t_1) = \eta(t_3) = \eta(t_4) = 1$, $\eta(t_2) = 2$, and $\eta(t_5) = 0.5$.

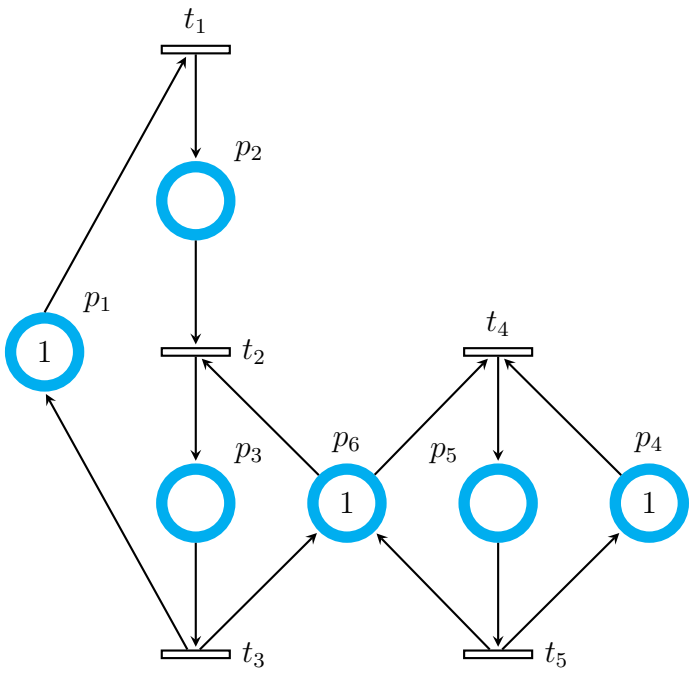
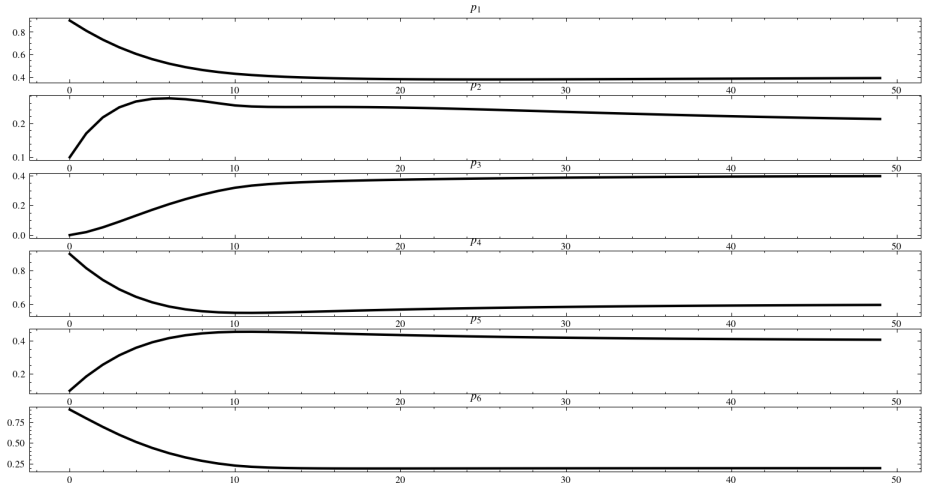
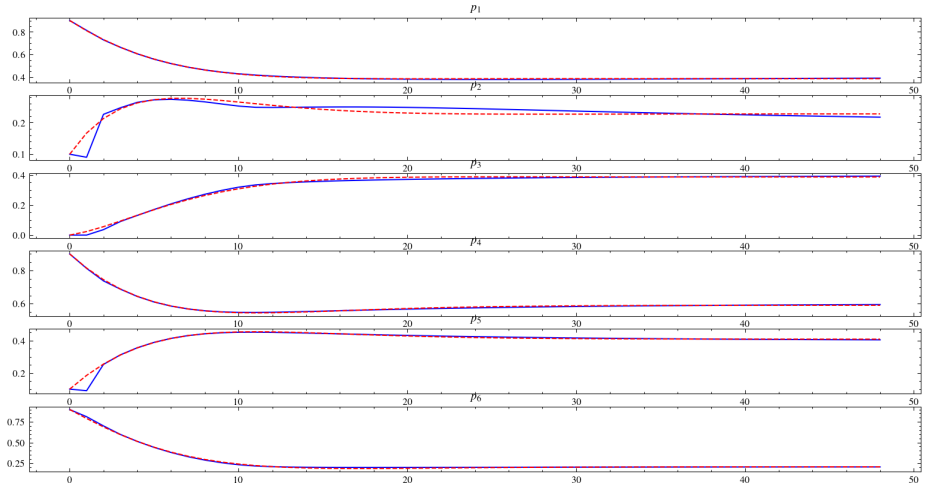


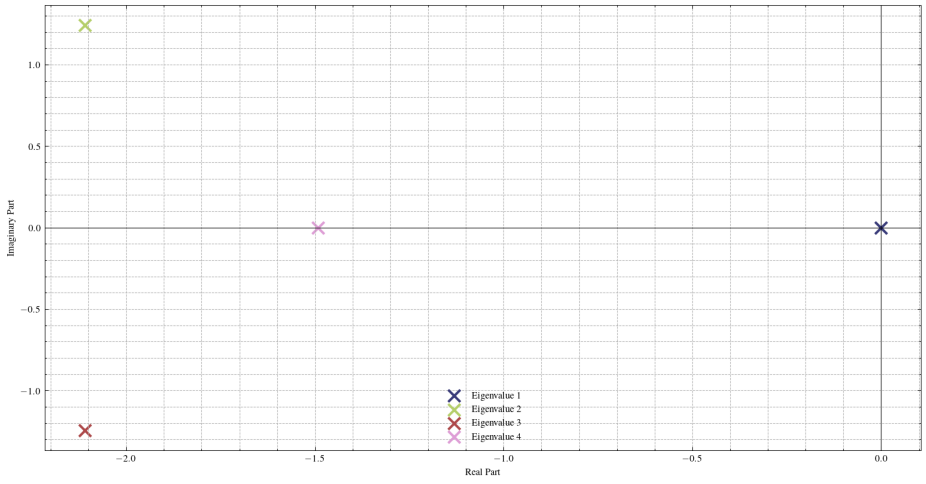
Figure 5.5 CPN example 2.



(a) Marking Evolution of the CPN model.



(b) Streaming prediction and DMD reconstruction of the marking evolution.



(c) Eigenvalues of the DMD operator.

Figure 5.6 DMD analysis of the CPN model. $m_0 = [1,\ 0,\ 0,\ 1,\ 0,\ 1]^T$, $l = 50$, $\Delta k = 0.1$.

The sDMD algorithm, as demonstrated in Figure 5.6b, initially takes a few time steps to adapt but eventually aligns closely with the actual marking evolution of all places within the network. The mean residual vector, $E[\varrho] = [-0.008,\ 0.0003,\ 0.01,\ -0.004,\ 0.006,\ -0.015]^T$, underscores the precision of sDMD in tracking this system's dynamics.

The Spearman correlation coefficients for the correlation between actual and reconstructed markings across places p_1 through p_6 are, respectively, 0.844, 0.735, 0.679, 0.998, 0.998, and 0.959. These coefficients indicate that the operator is attuned to evolution dynamics by the end of the simulation duration.

Regarding the eigenvalues (Figure 5.6c), $\lambda_1 = -3.1e{-}14$, $\lambda_2 = -2.109{+}j1.2422$, $\lambda_3 = -2.109{-}j1.2422$, and $\lambda_4 = -1.5$. The complex conjugate pairs, λ_2 and λ_3, with their negative real components, indicate decaying oscillations. This characteristic suggests that while these modes exhibit oscillatory behavior, they diminish over time, leading to a stabilization of markings over time. The absence of eigenvalues with positive real parts further confirms that unbounded growth in any of the places is not anticipated. The dimensionality of D is 4×4, which matches the rank of M.

5.2.3 SIRS Epidemiological Model

Epidemiological models provide vital frameworks for understanding and predicting the dynamics of disease spread within populations. These models not only explain the mechanisms that influence disease transmission but also assist in evaluating the impact of health policies, such as vaccination strategies [24].

Among these, the SIRS (Susceptible-Infectious-Recovered-Susceptible) model divides the population into three epidemiological classes: Susceptible (S), Infectious (I), and Recovered (R) [25]. Individuals in the Susceptible category can contract the disease, moving to the Infectious category at a rate β ($\eta(t_3)$), which is the transmission rate. Those in the Infectious category recover and move to the Recovered category

at a rate γ ($\eta(t_2)$), the recovery rate. Recovered individuals lose immunity over time, returning to the Susceptible category at a rate f ($\eta(t_1)$), which reflects the waning immunity or loss of protection [26].

This CPN model, of the SIRS dynamics, illustrated in Figure 5.7, utilizes the flexible PNet framework developed by Chay et al. [26][27]. The components of the model and what they represent is outlined in Table 5.2.

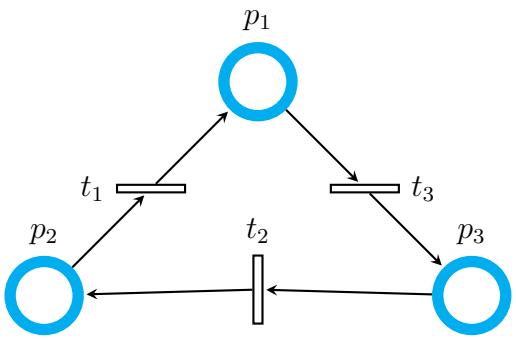
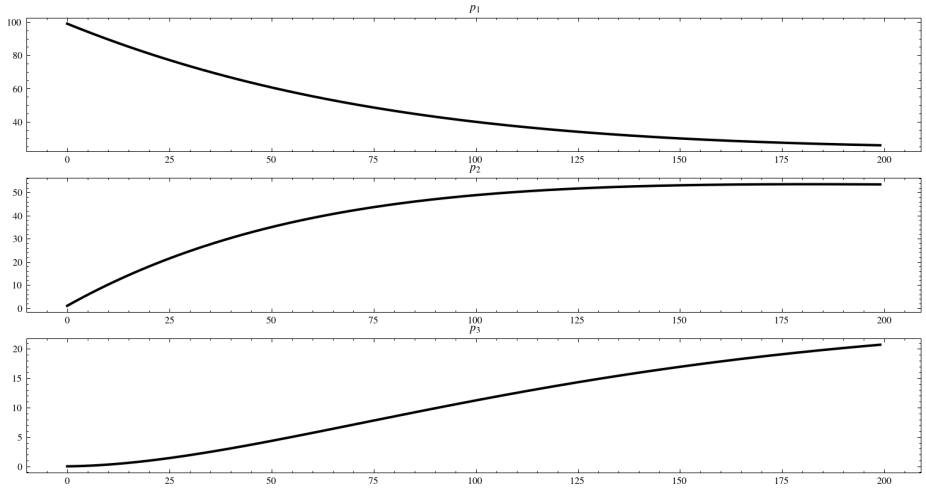


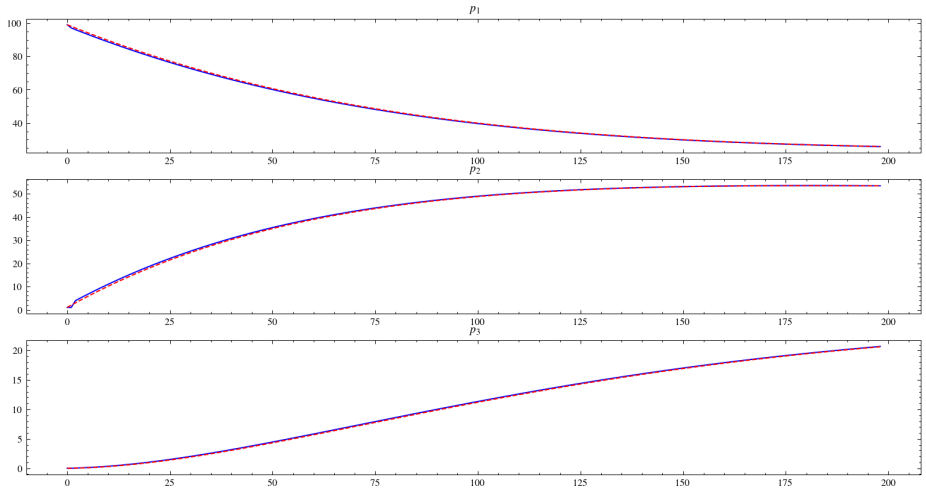
Figure 5.7 CPN representing the SIRS model.

Table 5.2 Elements of the SIRS CPN Model

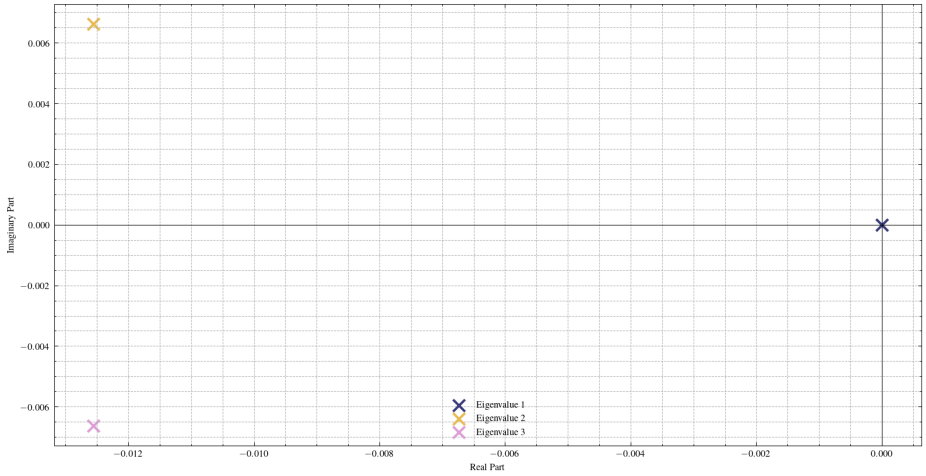
CPN element	**Meaning**
p_1	susceptible class
p_2	recovered class
p_3	infected class
t_1	recovered population becoming resusceptible (rate f)
t_2	infected population recovering (rate γ)
t_3	susceptible population getting infected (rate β)



(a) Marking Evolution of the SIRS CPN model.



(b) Streaming prediction and DMD reconstruction of the marking evolution.



(c) Eigenvalues of the DMD operator.

Figure 5.8 DMD analysis of the SIRS CPN model. $m_0 = [100,\ 0,\ 0]^T$, $l = 200$, $\Delta k = 1$. $\gamma = 0.005$, $\beta = 0.01$, $f = 0.01$.

The marking evolution within the SIRS CPN model is accurately captured by the sDMD algorithm, as evidenced by both the predictions and subsequent reconstruction (Figures 5.8a, 5.8b). The mean residuals for all places are impressively low: $E[\varrho] = [-9.85e-10,\ 0.009,\ 0.0002]$; indicating minimal deviation from actual dynamics. Additionally, the Spearman correlation coefficients between the reconstructed and actual marking evolutions are consistently at 1 for all places, underscoring the high fidelity of the reconstructions.

In this example, the analysis of D, which is a 3×3 matrix, reveals full utilization of the available dimensions, as evidenced by the three eigenvalues suggesting that M is of full rank. From Figure 5.8c, all eigenvalues have real parts less than 0: $\lambda_1 = -1.5e-14$, $\lambda_2 = -1.25e-2+j0.006$, $\lambda_3 = -1.25e-2-j0.006$; confirming the bounded nature of the CPN's dynamics. λ_2 and λ_3 are complex conjugate pairs with negative real parts, reflecting the presence of oscillatory modes with decaying oscillations within the system.

5.3 Limitations and Future Scope

While the current approach focuses primarily on CPNs due to their continuous nature and potential for complex dynamics, there is significant potential to extend this methodology to other types of Petri Nets, such as Stochastic and Hybrid Petri Nets. These systems, characterized by inherent uncertainty and discrete events, present an exciting frontier for DMD. Adapting DMD to accommodate the discrete dynamics of these systems could enhance our understanding of both, DMD and methods to analyze discrete dynamics.

The DMD operator excels in capturing the dynamics of a trajectory from a specific initial marking, its application to a broader set of initial markings remains a challenge. To address this, future research could explore multi-trajectory DMD, which involves applying the DMD algorithm to an ensemble of marking

trajectories to capture a more extensive portion of the system's reachability space [28]. Additionally, techniques like time delay embedding could be investigated to enhance the dimensional space of the dataset, potentially leading to a more generalized model that better encapsulates the diverse dynamics of CPNs [29].

CHAPTER 6

CONCLUSION

This thesis has explored the integration of Dynamic Mode Decomposition (DMD) with Continuous Petri Nets (CPNs) to develop a novel analytical approach for modeling the dynamics of CPNs. The primary contribution of this research has been to demonstrate that DMD can effectively capture the dynamic behavior of CPNs, enabling the prediction of future markings and offering insights into system stability and boundedness without detailed knowledge of transition firings or the underlying Petri Net model.

The findings from this thesis confirm that DMD is a robust tool for capturing the dominant dynamics of CPNs, as evidenced by the precision with which the DMD operator predicted future markings across various CPN models. The application of the Streaming DMD algorithm has showcased the potential of real-time data-driven analysis in capturing and predicting the evolution of system markings effectively. Furthermore, the analysis of eigenvalues derived from the DMD operator can provide valuable insights into the system's dynamics, revealing aspects of stability and boundedness.

The practical implementation of the Dynamic Mode Decomposition with Continuous Petri Nets has been carried out using Python, leveraging libraries such as NumPy and Matplotlib for numerical operations and data visualization, respectively. This implementation facilitates the simulation of CPN models and the application of the sDMD algorithm. Future implementations could explore optimizations and enhancements, such as parallel computing techniques or integration with other software tools, to handle larger PNs and more complex network configurations.

Despite the successes of this research, there are several areas where further work could enhance and extend the current work:

- Extending the application of DMD to other types of Petri Nets, such as Stochastic and Hybrid Petri Nets, to handle discrete events and inherent uncertainties.

- Investigating the integration of multi-trajectory DMD and time delay embedding techniques to improve the generalizability of the DMD operator across different initial markings and system configurations.

In conclusion, this thesis presents a novel application of Dynamic Mode Decomposition to Continuous Petri Nets. It extends the theoretical framework of CPN analysis and offers a new direction for the development of sophisticated, data-driven tools in the analysis of Petri Nets.

REFERENCES

[1] C. A. Petri, "Kommunikation mit Automaten," Ph.D. dissertation, Technische Universität Darmstadt, 1962.

[2] P. J. SCHMID, "Dynamic mode decomposition of numerical and experimental data," *Journal of Fluid Mechanics*, vol. 656, p. 5–28, 2010.

[3] H. ALLA and R. DAVID, "Continuous and hybrid petri nets," *Journal of Circuits, Systems and Computers*, vol. 08, no. 01, pp. 159–188, 1998. [Online]. Available: https://doi.org/10.1142/S0218126698000079

[4] C. Mahulea, A. Giua, L. Recalde, C. Seatzu, and M. Silva, "On sampling continuous timed petri nets: Reachability "equivalence" under infinite servers semantics," *IFAC Proceedings Volumes*, vol. 39, no. 5, pp. 37–43, 2006, 2nd IFAC Conference on Analysis and Design of Hybrid Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474667015328548

[5] M. Kloetzer, C. Mahulea, C. Belta, L. Recalde, and M. Silva, "Formal analysis of timed continuous petri nets," in *2008 47th IEEE Conference on Decision and Control*, 2008, pp. 245–250.

[6] M. Silva and L. Recalde, "On fluidification of petri nets: from discrete to hybrid and continuous models," *Annual Reviews in Control*, vol. 28, no. 2, pp. 253–266, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1367578804000288

[7] M. Blondin, A. Finkel, C. Haase, and S. Haddad, "The logical view on continuous petri nets," *ACM Trans. Comput. Logic*, vol. 18, no. 3, aug 2017. [Online]. Available: https://doi.org/10.1145/3105908

[8] E. Fraca and S. Haddad, "Complexity analysis of continuous petri nets," in *Application and Theory of Petri Nets and Concurrency*, J.-M. Colom and J. Desel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 170–189.

[9] T. Gu and R. Dong, "A novel continuous model to approximate time petri nets: Modelling and analysis," online.

[10] C. Mahulea, L. Recalde, and M. Silva, "Observability of continuous petri nets with infinite server semantics," *Nonlinear Analysis: Hybrid Systems*, vol. 4, no. 2, pp. 219–232, 2010, iFAC World Congress 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1751570X0900082X

[11] C. Mahulea, A. Giua, L. Recalde, C. Seatzu, and M. Silva, "Optimal model predictive control of timed continuous petri nets," *IEEE Transactions on Automatic Control*, vol. 53, no. 7, pp. 1731–1735, 2008.

[12] H. Zhang, C. W. Rowley, E. A. Deem, and L. N. Cattafesta, "Online dynamic mode decomposition for time-varying systems," *SIAM Journal on Applied Dynamical Systems*, vol. 18, no. 3, pp. 1586–1609, 2019. [Online]. Available: https://doi.org/10.1137/18M1192329

[13] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. Nathan Kutz, "On dynamic mode decomposition: Theory and applications," *Journal of Computational Dynamics*, vol. 1, no. 2, p. 391–421, 2014. [Online]. Available: http://dx.doi.org/10.3934/jcd.2014.1.391

[14] G. Snyder and Z. Song, "Koopman operator theory for nonlinear dynamic modeling using dynamic mode decomposition," 2021.

[15] J. L. Proctor, S. L. Brunton, and J. N. Kutz, "Dynamic mode decomposition with control," *SIAM Journal on Applied Dynamical Systems*, vol. 15, no. 1, pp. 142–161, 2016. [Online]. Available: https://doi.org/10.1137/15M1013857

[16] J. N. Kutz, X. Fu, and S. L. Brunton, "Multiresolution dynamic mode decomposition," *SIAM Journal on Applied Dynamical Systems*, vol. 15, no. 2, pp. 713–735, 2016. [Online]. Available: https://doi.org/10.1137/15M1023543

[17] M. S. Hemati, M. O. Williams, and C. W. Rowley, "Dynamic mode decomposition for large and streaming datasets," *Physics of Fluids*, vol. 26, no. 11, Nov. 2014. [Online]. Available: http://dx.doi.org/10.1063/1.4901016

[18] E. N. Lorenz, *Deterministic Nonperiodic Flow*. New York, NY: Springer New York, 2004, pp. 25–36. [Online]. Available: https://doi.org/10.1007/978-0-387-21830-4_2

[19] S. Soliman and M. Heiner, "A unique transformation from ordinary differential equations to reaction networks," *PLOS ONE*, vol. 5, no. 12, pp. 1–6, 12 2010. [Online]. Available: https://doi.org/10.1371/journal.pone.0014284

[20] X. Ye, J. Zhou, and X. Song, "On reachability graphs of petri nets," *Computers & Electrical Engineering*, vol. 29, no. 2, pp. 263–272, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790601000349

[21] C. W. Rowley, "dmdtools: Streaming dynamic mode decomposition tools for python," https://github.com/cwrowley/dmdtools/blob/master/python/dmdtools/streaming.py, 2021, accessed: 2024-04-27.

[22] J. J. Julvez and R. K. Boel, "A continuous petri net approach for model predictive control of traffic systems," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 40, no. 4, pp. 686–697, 2010.

[23] L. Recalde, S. Haddad, and M. Silva, "Continuous petri nets: Expressive power and decidability issues," in *Automated Technology for Verification and Analysis*, K. S. Namjoshi, T. Yoneda, T. Higashino, and Y. Okamura, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 362–377.

[24] F. Brauer, *Compartmental Models in Epidemiology.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 19–79. [Online]. Available: https://doi.org/10.1007/978-3-540-78911-6_2

[25] Y. Chen, J. Yang, and F. Zhang, "The global stability of an sirs model with infection age," *Mathematical Biosciences and Engineering*, vol. 11, no. 3, pp. 449–469, 2014.

[26] Z. E. Chay, B. F. Goh, and M. H. Ling, "Pnet: A python library for petri net modeling and simulation," 2023.

[27] M. Ling, "Pnet module in copads," 2016, accessed: 2023-04-28. [Online]. Available: https://github.com/mauriceling/copads/blob/master/copads/pnet.py

[28] R. Anzaki, S. Yamada, T. Tsutsui, and T. Matsuzawa, "Multi-trajectory dynamic mode decomposition," 2024, accessed: 2024-04-26. [Online]. Available: https://doi.org/10.51094/jxiv.602

[29] D. Dylewsky, E. Kaiser, S. L. Brunton, and J. N. Kutz, "Principal component trajectories for modeling spectrally continuous dynamics as forced linear systems," *Phys. Rev. E*, vol. 105, p. 015312, Jan 2022. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.105.015312