# FREQUENCY RESPONSE OF SELF-SIMILAR DYNAMIC NETWORKS WITH APPLICATIONS TO HEALTH MONITORING AND CONTROL

A Dissertation

Submitted to the Graduate School
of the University of Notre Dame
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

by

Xiangyu Ni

J. William Goodwine , Director

Graduate Program in Aerospace and Mechanical Engineering
Notre Dame, Indiana
February 2021

# FREQUENCY RESPONSE OF SELF-SIMILAR DYNAMIC NETWORKS WITH APPLICATIONS TO HEALTH MONITORING AND CONTROL

Abstract

by

Xiangyu Ni

While humankind's intellectual limit is advancing forward, the implementations of large-scale systems are expanding as well. No matter whether possessing interconnected appearances, those large-scale systems are often mathematically modelled as dynamic networks which involve abundant nodes and intricate rules regarding the interaction among those nodes. Hence, it can be anticipated that plentiful research concerning dynamic networks exist in literature, including studies about graph theories, multi-agent systems and materials' complicated behaviors. However, there exist few applications of frequency-domain methods to a general class of dynamic networks, mostly due to the complexity of computing their frequency response. That gap is filled by this dissertation, which shows that problem is tractable once self-similarities are leveraged and that the resultant outcomes can be further employed in dynamic networks' simulation, monitoring and control through available tools in the frequency domain.

The proposed method in this work could efficiently simulate some quantities spreading over complex systems that are typically described by partial differential equations. In addition to being efficient, that method offers another degree of freedom by handling the situation where the physical properties of those complex systems are nonuniformly distributed, in which case partial differential equations are almost

impossible to be solved analytically. For health monitoring, this dissertation suggests a new feature for candidate damage cases through their frequency response which can identify the existence, location and extent of the damage state. For controlling dynamic networks, this study takes advantage of the fact that the frequency response of a dynamic network is likely to form a set of neighboring plants when it is undergoing some variations in order to design a unified controller for those different situations by using robust control methods.

Another merit of this work is that it provides specific examples, *i.e.*, infinite dynamic networks, where fractional and irrational transfer functions naturally come to light. That offers the possibilities for understanding the physical meaning of fractional-order derivatives and implicit operators in the future.

The main motivation of this dissertation is to advocate studying dynamic networks through their frequency response. The author hopes that this work would build a bridge between monitoring and controlling complex systems and numerous frequency-domain tools.

*In memory of my grandmother*

QIONGZHI WANG

*(December 1928 - August 2020)*

CONTENTS

FIGURES

TABLES

# ACKNOWLEDGMENTS

I would like to offer my gratitude to my advisor, Dr. J. William Goodwine, and my committee members, Dr. Panos Antsaklis, Dr. James Schmiedeler, and Dr. Patrick Wensing. I also thank my colleagues in the Robotics and Controls Group, all of my course instructors, and people whom I am fortunate to know at the University of Notre Dame. The funding support from the National Science Foundation is gratefully acknowledged as well. Last but not least, I sincerely thank my parents and grandparents, and my girlfriend, Jianshan, and all my friends who navigate me through this special journey of my life.

# CHAPTER 1

# INTRODUCTION AND CONTEXTUAL BACKGROUND

## 1.1 Introduction

Networked dynamical systems are ubiquitous. Some of them are shaped by nature, *e.g.*, a river network within a drainage basin [93], and the circulatory system in our body [67]. Others are purposefully designed in that form. For example, electrical grids and cellular networks are set for our convenience, and fins with fractal geometries are utilized to enhance heat exchangers' performance [65]. In addition, numerous systems can be modeled by networks in the similar spirit of finite element methods. For instance, transmission lines can be viewed as networks consisting of various electrical elements [169]. Buildings are often approximated by a shear-frame structure with a lumped-mass planar network in the vibration control research area [75].

Because of countless applications brought by networked dynamical systems, it is imperative to understand their behaviors. As indicated by the title of this dissertation, one of its contributions is gaining that aforementioned knowledge from the perspective of networks' frequency response, which is typically challenging, due to the following two reasons. First, networks often consist of a great number of nodes which lead to a state vector with a considerable size. Second, internal interactions among nodes can be intricate. Therefore, this dissertation focuses on self-similar networks where the interaction rule between any two adjacent generations is invariant throughout the entire network. By leveraging that self-similarity, the resultant response between any two nodes inside a network can be acquired directly without simulating other nodes.

One merit of this work is that its computations of networks' frequency response and transfer functions are not limited by the research domain. Based on previous descriptions, we can easily observe that the employments of networked dynamical systems cover numerous areas. As a result, experts in each field often have their own customs to handle the corresponding applications. For instance, structural engineering researchers may appreciate the equation of motion,

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{C}\dot{\boldsymbol{x}} + \boldsymbol{K}\boldsymbol{x} = \boldsymbol{F},$$

where $\boldsymbol{M}$ is the mass matrix, $\boldsymbol{C}$ is the damping matrix, $\boldsymbol{K}$ is the stiffness matrix, and $\boldsymbol{F}$ represents the input forces. On the other hand, electrical engineers may be more familiar with Kirchhoff's circuit laws. In contrast, this work is designed for a general class of networks and connects them to their frequency response regardless of application domains. By doing so, the author hopes to offer experts with different backgrounds a tool to efficiently obtain their networks' frequency response which can be utilized in their further analysis.

In this dissertation, the knowledge of networked dynamical systems' frequency response and transfer functions is employed for monitoring their health and controlling their behavior. The motivation of networks' health monitoring is based on a well known fact that damages happening locally at some nodes inside a large-scale network may initiate an avalanche effect and collapse the efficiency and safety of the entire system. Those cascading failures can occur in different types of networks, such as infrastructure, social and economy systems. One notable example in US and Canada may be the Northeast blackout of 2003 [20, 33]. As a result, this dissertation proposes a model-based component-level health monitoring strategy for networked systems with the goal of returning a list of possibly damaged components and their corresponding damage amounts given the frequency response measurements of the

entire network.

Another incentive of this work is to control networked dynamical systems' behavior. Large amounts of research have been conducted concerning this type of problem especially for multi-agent systems. One challenge of controlling large-scale networks is that they are more likely to operate in different conditions compared to simple systems. Thus, a robust control strategy is required. For example, the number of cooperating unmanned aerial vehicles may change depending on implementation environments, or components' parameters inside a network may vary. This dissertation addresses that challenge by regarding a network's frequency response under various cases as a set of neighboring plants with uncertainties. Then, using robust control methods, this dissertation designs a unified controller for all systems in that set guaranteeing their stability and performance.

One novelty of networked dynamical systems' frequency response and transfer functions is that non-integer-order dynamics emerges naturally. As illustrated in Chapter 2, to exactly describe an infinite network's frequency response, non-integer-order expressions have to be involved in its transfer function. In fact, when a finite network's behavior converges to that of the corresponding infinite variant, its integer-order transfer function also converges to a non-integer-order expression. That expression may consist of two types of functions in the Laplace variable $s$. One type entails fractional orders of $s$ which corresponds to fractional-order integro-differentials in the time domain. The other type includes irrational expressions of $s$ which do not have clear equivalent time-domain operations yet.

The rest of this dissertation is organized as follows. The remainder of this chapter establishes the context for the results of this dissertation where literature context is reviewed. Chapter 2 demonstrates one main contribution of this work where algorithms of computing both frequency response and transfer functions for self-similar networked dynamical systems under various operating conditions are elucidated. To

showcase those algorithms' capability of quickly predicting real networks' behavior, in Chapter 3, they are applied to simulating high-speed electrical railway track circuits which determine the availability of a rail section. Chapter 4 proposes component-level damage detection methods for networks making use of those modeling algorithms presented in Chapter 2. Chapter 5 tackles the robust control problem where a unified controller is designed for a network under different operating conditions by considering them as a set of uncertain plants. Chapter 6 offers a rational approximation method of some irrational expressions from the perspective of networked dynamical systems. Finally, Chapter 7 concludes this dissertation and suggests future work.

## 1.2 Contextual Background

This section lays the contextual groundwork for the results in this dissertation. Section 1.2.1 lists the applications of dynamical networked systems with emphasis on simulating their response, monitoring their components' health and controlling their behavior. Then, Sections 1.2.2 to 1.2.4 address each of those three aspects, where an outline of how each topic has evolved in the literature is discussed in detail. Additionally, the research gap filled by this work for each of those three topics is also analyzed. Finally, Section 1.2.5 reviews the history and the preliminaries of fractional calculus, which, in Chapter 2, is proved to be an inevitable component for infinite self-similar networks' transfer functions.

### 1.2.1 Networked Dynamical Systems and Their Applications

It is unsurprising to see that research works regarding networked dynamical systems come from various perspectives given a great number of real-life applications. One aspect initially emerged from thermodynamics with the currently leading principle called the Constructal Law, proposed by Dr. Bejan in 1996, which gives a reason for the natural occurrence of flows patterns and predicts the progression of their con-

figurations [12, 13], such as river networks [134] and circulatory systems [115]. The main idea is that those configurations are continuously "optimized" by nature in accordance with the argument that "For a finite-size flow system to persist in time, it must evolve such that it provides greater and greater access to the currents that flow through it." [14, 15] This collection of works relates a flow's behavior to the shape of its networked container.

Another field of study concerning the networks' topology is often linked with graph theory, whose starting point is usually credited to Euler's solution to the Königsberg Bridge Problem [17]. According to a detailed review [107], the primary goal of this field is to understand the structure and function of complex networks, where the empirical studies of the former, the structure of real-life networks, are the outset of this research area. Those networks include social networks [16], information networks like the World Wide Web [66], technological networks like cellular networks [161], and biological networks [150]. The focus of those empirical studies at first was to quantify a number of statistical properties of networks being important for their function, which gradually unveiled that many statistical properties are commonly shared by networks from different branches of science. The most famous one probably is the small-world effect, demonstrated by Stanley Milgram's letter-passing experiment in the 1960s [99], which states that most pairs of vertices are connected by a short path in networks. Those observations of networks' non-randomness inspired thoughts about seeking some possible mechanisms directing their formation which resulted in abundant network models. The ongoing focus of this research domain mostly shifts to explanations of processes occurring on networks *via* those mathematical models, for example, search and navigation processes, and network transmission and epidemiology [121].

As for the researchers in robotics, the most heavily investigated topic about networked dynamical systems is multi-agent systems. The objective for this discipline is

to design a strategy for two or more agents to achieve a global task in a cooperative manner, though the knowledge about networks' topology and graph theory is still crucial [21]. One example global task is consensus problems where the states of all robots are asked to converge to the same value [135]. Multi-agent systems have at least two outstanding advantages over single-agent systems. First, multi-agent systems are more robust in the sense that if a few agents fail, the other agents could quickly adapt the situation to continue finishing the global task. Second, multi-agent systems can cover a much larger physical area simultaneously, or equivalently, execute an involved task in a temporally parallel manner. The above two benefits result in a wide spectrum of applications, such as search and rescue [29], distributed map merging [5], collective transport [137], clock synchronization [142], sensor fusion [148], localization [146], distributed support vector machine [50], and distributed air-conditioning optimization [60]. The continuing emphases include incorporation between network topologies and agent dynamics, rigid formations in three-dimensional space, and fully autonomous and distributed multi-agent systems.

Approximating intricate dynamical systems by using networks also attracts significant attentions, which, however, can be easily neglected by researchers concentrating on networks since those systems usually do not possess networked appearances. For instance, a long transmission line can be estimated by a electrical network where electrical properties like resistance are lumped at each subsection [102], which is often used to model railway track circuits with the purpose of detecting whether a certain portion of a track is occupied [169, 141, 22]. A closely related equivalent is the representation of a multi-story building as a shear-frame structure where a pair of parallel spring and damper connecting two neighboring masses resembles the shearing motion between two adjacent floors, which is often employed in structural vibration control [75, 53, 57, 64]. Another class of applications falling into this classification is about reproducing materials' complicated behavior through large-scale networks. For a wide

range of composite materials, one prominent phenomenon is that the magnitude of their overall admittance is always in a power-law scaling relation with the frequency under the alternating current field within an intermediate frequency region, which is called the universal dielectric response, proposed by Dr. Jonscher [71, 72]. To date, there is yet no agreement on the origins of such a response. However, it is possible to reproduce that phenomenon by using random resistor-capacitor networks and show that the ratio of quantities between resistors and capacitors determines the slope of that power-law relation [95, 3, 167]. Interestingly, the same fact is also observed from random biphasic mechanical truss networks which are constituted by springs with two different constants [105]. Finally, using networks to model the rheology of viscoelastic behavior is another example [62, 154].

For the last category of applications, the nodes inside networks are often linked to each other through idealized basic elements such as springs and capacitors, which is the main type of network considered by this dissertation, although this work relaxes that limitation to all linear time-invariant operators, like proportional-integral-derivative controller blocks. It is also worth pointing out that the main contents provided by this work at the current state are far from multi-agent systems despite the fact that the entailed networks can be viewed as being composed of many robots. Actually, the relation between the study regarding multi-agent systems and this work can be analogized by the connection between controller synthesis and controller analysis. According to [21], the main goal of multi-agent systems' research is to compose a strategy so that each robot can make decisions based on its local knowledge of the surroundings to perform a global task cooperatively. Therefore, essentially, multi-agent systems' research strives to solve a controller synthesis problem. In contrast, the networks in this dissertation come together with a defined rules of interaction among agents. As a result, it is more like the case where a controller has already been designed and we need to analyze the system's closed-loop performance, and thus

a controller analysis problem. However, it does not necessarily imply that this work can never contribute to the multi-agent systems' research in the future. The details regarding that potential extension is discussed in Chapter 7.

One conclusion can be drawn from the above review of applications is that, like any other dynamical systems, networked ones have research work from both passive and active directions. On one hand, researchers attempt to understand a network's spontaneous evolution in time through modeling and characterization. On the other hand, researchers design and modify a network to control its behavior or enhance its performance. The Constructal Law is a great example, which later becomes a scientific principle in engineering design, with the idea of mimicking how nature "makes optimal decisions" in order to come up with strategies for more effective connections of heat and fluid flow, people, goods and information [23]. Studies concerning networks' topology and graph theory concentrate more on the passive direction, while multi-agent systems clearly follow the active direction. Similarly, understanding intricate dynamical systems' behavior through network approximations comes from the passive direction, while controlling buildings' vibrations *via* simplified network models is on the active direction. Undoubtedly, both directions are indispensable. The observations of a dynamical system's behavior often result in a theoretical mathematical model which can simulate its responses to different impacts under various environments. Those predictions enable developers to validate their design and assist them to obtain a better performance from the system without frequently conducting hardware experiments [70].

From a passive perspective, this dissertation computes the frequency response and transfer functions between any two nodes within a network. Besides, it also detects potential damages inside networks' components. From an active perspective, this dissertation designs a unified controller to enhance networks' stability and performance under various operating conditions. The unique feature about this work is its

frequency-domain approach where the knowledge concerning a networked dynamical system's frequency response is the core and is employed in both health monitoring and control problems. In the next three sections, detailed literature review for the above three aspects, modeling, health monitoring, and control, are provided respectively.

### 1.2.2 Modeling Networked Dynamical Systems

This section reviews networks' modeling arising from the following two research fields. The first field is research about graphs which focuses more on *networks* [107]. The network models used in this research discipline are mostly abstract and mathematical, which cannot be comfortably drawn on paper due to their extensive scales. Those mathematical models are used to derive statistical properties of the corresponding networks, from which researchers understand how their structure is connected with their functioning as well as understand the processes occurring on them. The second research field emphasizes more on *dynamics* when dynamical systems are connected. In this domain, the nodes are variables such as force and acceleration, and the edges represent modules like a double integrator scaled by a mass relating accelerations to forces [164], which is similar to the block diagram of control systems. This dissertation belongs to the second set of research although what the nodes and edges represent is different. In this dissertation, nodes are similar to those in multi-agent systems which have differential equations describing their evolutions depending on their local knowledge of the surroundings whose availabilities are denoted by edges.

The research on graphs started with empirical studies to quantify a number of statistical properties of networks [107]. The following are several examples of those properties. The mean shortest distance, which for an undirected network with $n$

vertices is defined as

$$l = \frac{1}{\frac{1}{2}n(n+1)} \sum_{i \geq j} d_{ij},$$

where $d_{ij}$ is the shortest distance between the vertex $i$ and vertex $j$. That mean shortest distance is usually much smaller than the number of vertices for real-life networks, which becomes the so-called small-world effect proving how fast information or a virus could spread throughout a population [30] (as demonstrated by the year 2020). A network's transitivity is the probability of three vertices being all directly connected in a network, or the density of triangles in a network's graph, which is quantified by a clustering coefficient. An example is the probability that a friend of your friend is also your friend. Transitivity usually converges to a nonzero constant as the number of vertices increases for various types of real networks. A vertex's degree inside a network equals the number of edges connected to it. The probability $p_k$ is then defined as the fraction of vertices that have degree $k$, whose distribution is often highly right-skewed in most networks, meaning that the distribution has a longer tail in the high-$k$ region compared to the low-$k$ one. Moreover, many of them follow power laws in the high-$k$ region: $p_k \approx k^{-\alpha}$ [10], which provides evidence that large networks are often organized into a scale-free condition by themselves. Others follow an exponential decay: $p_k \approx e^{-k/\kappa}$ [143]. One related property is the network resilience which measures the toughness of a network against the removal of its vertices. That can be quantified by how the mean shortest distance grows with an increasing number of vertices being removed, which is meaningful for studies of vaccination [120] and research against sabotage of the Internet [2]. Many real networks, especially social networks, also exhibit community structures, that is many groups of vertices have a higher density of edges within them, while have fewer connections among groups, which leads to a research topic called community discovery [136].

From the above list of networks' representative properties, we can observe that those properties are usually shared by many classes of networks. That non-randomness inspires researchers to propose mathematical models for complex networks, which have four main types [107]. The first type is called random graphs, denoted by $G_{n,p}$, where the network has $n$ vertices and each pair of them is connected with the probability $p$ [45]. Random graphs are a great starting point of comprehending complex networks' function. However, they don't resemble most of the above-mentioned common properties of real networks except for the small-world effect, some of which can be overcome by their generalizations [108]. The second type is exponential random graphs [151] and Markov graphs [51] which is a candidate model to understand networks' transitivity. The problem of this type of model is that under some special conditions, the networks generated have subsets of vertices where every possible edge exists, which is not frequently observed in real networks. The third type is small-world model, which is another candidate to understand the transitivity [163]. This class of networks is more tractable in the sense that those statistical properties can be solved analytically and are often consistent with the observations. As a result, small-world models become the substrate of understanding the processes taking place on networks. The last type is called models of network growth, which is used to explain how those common statistical properties emerge in the first place. The models of this type gradually increase in the number of vertices and edges to imitate the growth of real networks [9]. Note that the last type of models is no longer static as opposed to the first three types. Therefore, some literature names them dynamic networks to distinguish them from the others [81].

The second class of research focuses on modeling the dynamics of networked systems, which is often data-driven, since applying principles of physics to each element within a large-scale network is tedious and error-prone. Therefore, identification and learning are often used interchangeably with the word modeling regarding this topic.

The relation between this domain and the previous one is that the networks in this research area can be viewed as extensions from those in the previous one with dynamical behaviors included in the edges. That relation provides some graph models as candidates for the model selection in this research area [8]. However, there are other candidate models from distinct origins, which justifies the previous claim that this research domain is not a subset of the previous one. The other candidate models include probabilistic models [78], state-space network models [59], and transfer function network models [28]. The basic idea is to select the model from a set which offers the best reproduction of the observations, although different models typically require different identification methods. For example, state-space network models are usually obtained by subspace identification based on state-space realizations [59], while transfer function network models are normally acquired by prediction error methods which is closely related to Maximum Likelihood estimators [156] or by Bayesian estimators when theoretical information is available [28]. The modeling goals are identifying the connecting structure of networks [140], identifying one dynamical system [46], and identifying all dynamical systems with the knowledge of networks' topology [54].

Similar to the second research area, this dissertation models networked dynamical systems by the frequency response and transfer functions between any two nodes. However, those responses are obtained by the computation which gradually takes into account every differential equation describing each node's dynamics given by the principles of physics along with the knowledge about networks' topology. This procedure can be intricate for general networks, so this dissertation restricts its focus to self-similar networks which largely reduces the complexity [92]. The knowledge from modeling is subsequently employed in monitoring networks' health and controlling their performance.

### 1.2.3 Monitoring Networked Dynamical Systems' Health

This section reviews two topics that are closely related to this dissertation, namely anomaly detection in research of graphs [132] and structural health monitoring [47]. Both of them aim at discovering undesired behaviors of large-scale systems where the former focuses more on systems that can be represented by networks, while the latter concentrates on engineering structures. However, due to the reason stated in Section 1.2.1, networked dynamical systems can be employed to approximate continuum systems. As a result, both fields of studies are of great interest to this dissertation.

Anomaly detection for dynamic graphs looks for objects, interconnections and time instances that are special compared to the others inside complex systems. The examples of applications include detection of wildfires [27] and hurricanes [26], protecting network systems against intrusion [166], and rooting out abnormal users and events [128, 159]. According to [132], there are four types of anomalies being handled in the literature. The first two types are anomalous vertices and anomalous edges where a subset of vertices and edges with unusual evolution is returned respectively [73]. The third type includes anomalous subgraphs which mostly deal with irregular behaviors of communities such as splitting, merging, disappearance, and reappearance. One typical application is to monitor changes and threats in social networks [61]. The last type includes anomalous events and changes, where anomalous events indicate the time instants when the network's behavior is significantly different, and anomalous changes denote the time instants when the network's behavior drifts to a distinct status. Event detection attracts much attention in data mining which, for instance, can be applied to identifying the moments when molecular dynamics simulations perform differently [131]. Change detection is widely implemented in human interactions like, for example, search of scholars' change in research interest by tracking their history of co-authorship [80]. Various anomaly detection methods exist in the literature, most of which can handle multiple types of anomalies [132]. The

fundamental idea is devising a scoring function mapping from a set of objects (*e.g.*, vertices) to a real number, and anomaly occurs where the variation in that scoring function becomes greater than some threshold [170].

Structural health monitoring is usually implemented on physical systems, like aerospace, civil and mechanical engineering infrastructure. According to [47], many researchers view the structural health monitoring problem as one application of statistical pattern recognition which is a classification procedure using statistical decision theory to draw class boundaries. The steps of statistical pattern recognition usually include preprocessing, feature extraction, learning and classification [69]. Correspondingly, [47] lists the following four-step paradigm for structural health monitoring problems.

1. Operational evaluation,
2. Data acquisition, normalization and cleansing,
3. Feature selection and information condensation, and
4. Statistical model development for feature discrimination.

Operational evaluation answers the questions like what will be monitored and how the measurements will be acquired [147]. The second step preprocesses the data where operations like separating changes in measurements caused by damage from those caused by environment are performed [152]. The third step picks out the data features that are sensitive to the expected damage. Common choices include vibration amplitude, frequency and modes [153]. Other methods include using simulations or experimentally validated finite element models to understand the expected damage's impact [52], or performing realistic loading tests on degraded structures [88]. The last step acts on the extracted features from the previous step to quantify the damage state. According to [138], the damage state of a system comprises the following five elements in an increasingly demanding order.

1. Existence. Is there a damage in the system?
2. Location. Where is the damage in the system?
3. Type. What kind of damage is present?
4. Extent. How severe is the damage?
5. Prognosis. How much useful life remains?

Learning algorithms are frequently employed to answer the above questions [165, 48]. Supervised learning, when data from both the undamaged and damaged structure are available, can often determine the answers to all five questions regarding the damage state [114]. On the other hand, unsupervised learning which does not contain examples from the damaged structure concentrates more on the existence and location problems [44], though attempts on solving the remaining three questions attracts increasing attention [43]. Recently, semi-supervised learning algorithms also have appeared in the literature which use both labeled and unlabeled data for classification where unlabeled data is collected while the structure is in service and labeled data is obtained during annual visual inspections [24].

In the context of anomaly detection for dynamic graphs, this dissertation searches for anomalous vertices and edges, at which the damaged components are located. In the settings of structural health monitoring, this work leverages the mismatch between the computed frequency response and the measured one as the feature, and quantifies the damage state in the perspectives of existence, location and extent.

### 1.2.4 Controlling Networked Dynamical Systems

Multi-agent systems are probably the most studied topics in the area of networked control. Research concerning multi-agent systems seeks a control strategy for a team of robots to collaboratively attain a global task. In addition to the consensus problem mentioned in Section 1.2.1, global tasks also include another similar mission called leader-follower coordination where all agents are required to converge to a state set

by the leader, although the leader's state may be only available to a portion of the followers [68]. Flocking problems ask all members to maintain equal distances from their neighbors [117], whereas formation control is a harder task where a group of agents are asked to establish a predefined configuration [49]. Coverage control requests a team of moving sensors to reach the positions which give rise to the optimal performance regarding a collective measurement of stochastic events occurring inside a convex polytope [31]. Incidentally, the idea of multi-agent systems can actually be applied to some numerical evaluations, such as distributed parameter estimation [74], distributed regression [58], distributed Kalman filter [116], and distributed optimization [106, 130].

The control problems in this dissertation take advantage of one vital observation from the modeling part that the frequency response of networked dynamical systems under various operating conditions make up a set of uncertain and neighboring plants. Therefore, this dissertation leverages robust control concepts to synthesize a unified controller for networked dynamical systems in different circumstances to guarantee their stability and performance. In the following section, literature regarding robust control is reviewed.

#### 1.2.4.1 Robust Control

Two main tasks of robust control are robustness analysis and controller synthesis, where the former analyzes if a (controlled) system is stable or quantifies its stability margin under a certain amount of uncertainties, and the latter seeks a suitable controller to achieve stability or even optimal performance under uncertainties [124]. The uncertainties $\Delta(s)$ are categorized into two classes: the unstructured ones and the structured ones. Unstructured uncertainties allows $\Delta(s)$ to be a full matrix but with an additional constraint, for instance, the bounded real constraint: $\|\Delta(s)\|_\infty \leq \eta$ with $\eta \in \mathbb{R}^+$[171]. Other constraints include norm bounded condition [122], positive

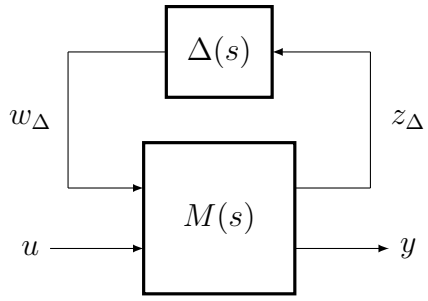


Figure 1.1. $M$-$\Delta$ structure for robust stability analysis.

real condition [7], and negative imaginary condition [83]. On the other hand, structured uncertainties normally have particular descriptions. One common description is that $\Delta(s)$ is a block diagonal matrix, that is

$$\Delta(s) = \begin{bmatrix} q_1 I_{m_1} & \cdots & 0 & & & \\ \vdots & \ddots & \vdots & & & \\ 0 & \cdots & q_l I_{m_l} & & & \\ & & & \Delta_1(s) & \cdots & 0 \\ & & & \vdots & \ddots & \vdots \\ & & & 0 & \cdots & \Delta_b(s) \end{bmatrix}, \tag{1.1}$$

where $q_i$ are real or complex uncertain parameters repeated with the multiplicity $m_i$ and $\Delta_i(s)$ are full-block stable and proper unknown transfer function matrices. Other descriptions include time-domain and frequency-domain integral quadratic constraints [97], though they are not directly posed on $\Delta(s)$.

To analyze the robust stability, the block diagram of a system is often converted to the $M$-$\Delta$ structure as shown in Figure 1.1, where $M(s)$ is the known plant [171]. Note that the $M$-$\Delta$ structure can also be presented in the state-space formulation. The most famous proposition about the robust stability regarding unstructured uncertainties is the small gain theorem, which states that if both $M(s)$ and $\Delta(s)$ are

proper, rational, and analytic in the closed right half plane, then the interconnection from $w_\Delta$ to $z_\Delta$ is well posed and internally stable for all $\|\Delta(s)\|_\infty \leq \eta$ if and only if $\|M(s)\|_\infty < 1/\eta$ ($\eta > 0$). Other stability theorems concerning unstructured uncertainties include those imposed on state-space representations [38, 129], quadratic stability with norm bounded uncertainty [77], passivity theorem for the $M$-$\Delta$ structure with negative feedback, and the negative imaginary stability theorem [83]. Analyzing robust stability with respect to structured uncertainties is often generalized from the above results to adapt specific descriptions about the uncertainty $\Delta(s)$. Perhaps one of the most well-known generalizations is the structured singular value, which relaxes the domain of searching for the largest singular value from all possible uncertainties to a subset of that. Rigorously, if all structured uncertainties $\Delta(s)$ form a set $\mathcal{D}$, then the structured singular value of $M(j\omega)$ is defined as

$$\mu_{\mathcal{D}}(M(j\omega)) = \frac{1}{\min\{\overline{\sigma}(\Delta) : \det(I - M(j\omega)\Delta) = 0, \Delta \in \mathcal{D}\}}. \tag{1.2}$$

With the above generalization, the aforementioned small gain theorem can be extended to its counterpart, small $\mu$ theory, where the interconnection is well posed and internally stable for all $\Delta(s) \in \mathcal{D}$ with $\|\Delta(s)\| \leq \eta$ if and only if $\sup_{\omega\in\mathbb{R}^+} \mu_{\mathcal{D}}(M(j\omega)) < 1/\eta$. In other words, the structured singular value of $M$ is a measurement of the stability margin under structured uncertainties. Incidentally, the concepts of the structured singular value [40] and its inverse, the multivariate stability margin [139], were published in the same issue of the same volume of the same journal by two different people. The computation of the structured singular value in Equation (1.2) is generally nonconvex, and thus is NP-hard. Therefore, in literature, the structured singular value's bounds are frequently evaluated. Specifically, when $q_i \in \mathbb{C}$, $\Delta_i \in \mathbb{C}^{r_i,r_i}$ in Equation (1.1), and $M \in \mathbb{C}^{r,r}$ where $r = \sum_{i=1}^{l} m_i + \sum_{i=1}^{b} r_i$, the set $\mathcal{W}$

can be formed as

$$\mathcal{W} = \{\text{bdiag}(D_1, \ldots, D_l, d_1 I_{r_1}, \ldots, d_{b-1} I_{r_{b-1}}, I_{r_b})\},$$

whose elements have the block diagonal structure similar to that in Equation (1.1) where $D_i \succ 0$, $d_i > 0$. Then, the following inequality holds [171].

$$\mu_{\mathcal{D}}(M) \leq \inf_{D \in \mathcal{W}} \overline{\sigma}(DMD^{-1}). \tag{1.3}$$

Other robust stability criteria with respect to different structured uncertainties encompass those posed on parametric uncertainties based on Hurwitz stability [100, 11], quadratic stability with structured uncertainty [19], and stability with both time-domain [126] and frequency-domain integral quadratic constraints [97]. It is worth noting that the last two criteria can be formulated as linear matrix inequalities (LMI) which are utilized frequently in controller design problems due to convexity.

Another goal of robust control is to design the compensator that guarantees stability and performance given a set of possible uncertainties. The main idea is to search for a controller so that the system satisfies the aforementioned robust stability criteria, and sometimes offers optimal performance as well. The simplest method maybe is loop-shaping which is usually applied to single-input single-output systems with unstructured uncertainties, in which case $\|M(j\omega)\|_\infty$ is the peak gain that is available directly in the Bode magnitude plot. Due to the small gain theorem, the main goal is then to shape $|M(j\omega)|$ such that its gain is less than $1/\eta$ across all frequencies, while additional performance requirements can also be synthesized, such as tracking, disturbance rejection and noise attenuation [96]. In more complicated cases, $H_\infty$ control is often employed to make the system satisfy the small gain theorem. One main approach of solving $H_\infty$ control problems is to derive the equivalent algebraic Riccati equation which then can lead to both state feedback [125] and output feedback $H_\infty$

controllers [39]. However, the disadvantage is that some specific assumptions are required for the uncontrolled plant $G(s)$. To overcome that drawback, an alternative way to handle $H_\infty$ problems is solving the corresponding linear matrix inequalities [19, 41]. However, that is more computationally demanding compared to dealing with algebraic Riccati equation. Other methods to tackle $H_\infty$ control problems include coprime factorization [56] and ploynomial approaches [82].

Performance requirements can also be incorporated with the requisite for robust stability. One example provided by [123] requires a linear system to be quadratically stable while an LQR-like (Linear-quadratic regulator) cost function has a guaranteed upper bound under unstructured, norm bounded uncertainties. The solution to that problem is a state feedback controller given by an algebraic Riccati equation. A similar question is also posted with respect to structured uncertainties with stochastic integral quadratic constraints where an LQG-like (Linear-quadratic-Gaussian) cost function is used [126].

Another famous method dealing with structured uncertainties called $\mu$ synthesis is often employed especially when the specifications for robustness cannot be expressed on (complementary) sensitivity functions alone. The concept of $\mu$ synthesis is far from intricate. Recall the upper bound of the structured singular value presented in Equation (1.3). If the nominal plant is denoted by $P$, and the controller is denoted by $K$, the $M$ in the $M$-$\Delta$ structure (Figure 1.1) can be represented by $M(P, K)$. Then, the idea of $\mu$ synthesis, taking the small $\mu$ theory into account, is to search for the controller $K$ to achieve the minimum of that upper bound, *i.e.*,

$$\inf_K \inf_D \overline{\sigma}(D(\omega)M(P,K)(j\omega)D^{-1}(\omega))$$

overall all frequencies $\omega$. A two-stage optimization process is often employed to find that controller, where $K$ and $D$ are found in turn repeatedly, and thus it is called $D$-$K$

iteration [149]. Other methods coping with structured uncertainties include searching for a state feedback controller guaranteeing a linear system's quadratic stability under parametric uncertainties by solving linear matrix inequalities [19]. It is worth noting that, in recent years, there is also an emergence of employing a stochastic method, called statistical learning theory, to design controllers for uncertain systems [79, 157].

### 1.2.5 Fractional Calculus

In the modeling part of this dissertation (Chapter 2), we would observe that non-integer-order dynamics appear in infinite self-similar networks' transfer functions. In fact, finite networks' transfer functions are always integer-order. However, the highest order of that would grow as the size of networks increases. Eventually, when a finite network converges to its infinite counterpart, its transfer function also converges to another transfer function with fractional order of $s$ or even irrational expressions of $s$, where $s$ is the Laplace variable. Therefore, this section reviews the literature concerning fractional calculus to complete the entire contextual background of this dissertation.

Fractional calculus is a study regarding real number powers of the integro-differential operator denoted by $D$. For instance, $D^2 f$ means the second-order derivative of the function $f$, $D^{-2} f$ means to integrate $f$ twice, and $D^0 f$ means $f$ itself. That notation is clearly built upon the intuition that the derivative of the integral of a function gives back that function itself, which is formally called the second fundamental theorem of calculus [4]. Note that the other order, the integral of the derivative of a function gives back that function itself, is false in general because of the constant of integration.

Due to that unification of integral and derivative, it is natural to ask what are those in between $D^1 f$ and $D^2 f$, which leads to the studies of fractional calculus. In fact, research in fractional calculus dates back to the birth of calculus [90]. It should

also be noted that the powers of the integro-differential operator $D$ could be further extended to complex numbers in literature [89, 6], though they are not the focus of this research work.

The computations of fractional-order derivatives are almost always extended from those of integer-order ones. Here, three categories of examples are reviewed. First, generalizations of derivatives of monomials, like $t^2$, to fractional orders are presented. Second, generalizations based on the second fundamental theorem of calculus, namely Riemann-Liouville definition and Caputo definition, are reviewed. Third, generalizations based on the limit definition of derivatives, such as Grünwald-Letnikov definition, are evaluated. More detailed information can be found on any fractional calculus textbook, *e.g.*, [155].

For a monomial, $t^\lambda$ ($t \in \mathbb{R}^+$ and $\lambda \in \mathbb{Z}^+$), some of its integer-order integrals and derivatives are

$$
\begin{aligned}
{}_0D_t^{-2}\tau^\lambda &= \frac{t^{\lambda+2}}{(\lambda+2)(\lambda+1)} = \frac{\lambda!}{(\lambda+2)!}t^{\lambda+2},\\
{}_0D_t^{-1}\tau^\lambda &= \frac{t^{\lambda+1}}{\lambda+1} = \frac{\lambda!}{(\lambda+1)!}t^{\lambda+1},\\
D^0 t^\lambda &= t^\lambda = \frac{\lambda!}{\lambda!}t^\lambda,\\
D^1 t^\lambda &= \lambda t^{\lambda-1} = \frac{\lambda!}{(\lambda-1)!}t^{\lambda-1},\\
D^2 t^\lambda &= \lambda(\lambda-1)t^{\lambda-2} = \frac{\lambda!}{(\lambda-2)!}t^{\lambda-2}.
\end{aligned}
$$

As a result, in the case of integer-order, a concise conclusion can be drawn that

$$
{}_0D_t^n t^\lambda = \frac{\lambda!}{(\lambda-n)!}t^{\lambda-n} \qquad n \in \mathbb{Z}. \tag{1.4}
$$

Note that the prescript 0 and the postscript $t$ of the operator $D$ indicate the lower and upper limit of the integration respectively, which can be ignored for integer-order

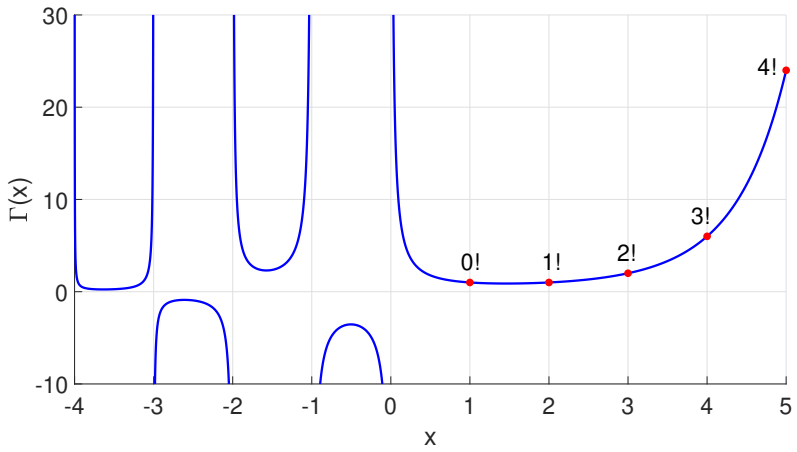


Figure 1.2. The Gamma function.

derivatives. It is straightforward to extend Equation (1.4) to fractional orders where $n \in \mathbb{R}$. One issue is that the factorial is only defined for nonnegative integers, so a general way to extend that is by using the Gamma function $\Gamma(x)$ defined as

$$\Gamma(x) = \begin{cases} \displaystyle\int_0^{+\infty} e^{-y} y^{x-1} dy, & x \in \mathbb{R}^+, \\ \displaystyle\frac{\Gamma(x - \lfloor x \rfloor)}{\prod_{k=0}^{-\lfloor x \rfloor - 1}(x+k)}, & x \in \mathbb{R}^- \backslash \mathbb{Z}^-, \end{cases}$$

where $\lfloor \cdot \rfloor$ indicates the floor function. It can be shown that $\Gamma(x+1) = x!$ when $x$ is a nonnegative integer as illustrated by Figure 1.2. Therefore, one way of generalizing the derivatives of $t^\lambda$ to fractional orders is

$${}_0D_t^\alpha t^\lambda = \frac{\Gamma(\lambda+1)}{\Gamma(\lambda-\alpha+1)} t^{\lambda-\alpha} \quad \alpha \in \mathbb{R},\ \lambda \in \mathbb{R}\backslash\mathbb{Z}^-,\ \lambda - \alpha \in \mathbb{R}\backslash\mathbb{Z}^-,\ t \in \mathbb{R}^+. \tag{1.5}$$

The above result is exemplified by $f(t) = t^2$ in Figure 1.3, from which the fact that fractional-order derivatives interpolate integer-order derivatives can be observed. Note that some other special functions' fractional-order derivatives can also be generalized by this way, such as $e^{\lambda t}$, $\sin(\lambda t)$, and $\cos(\lambda t)$ [155].

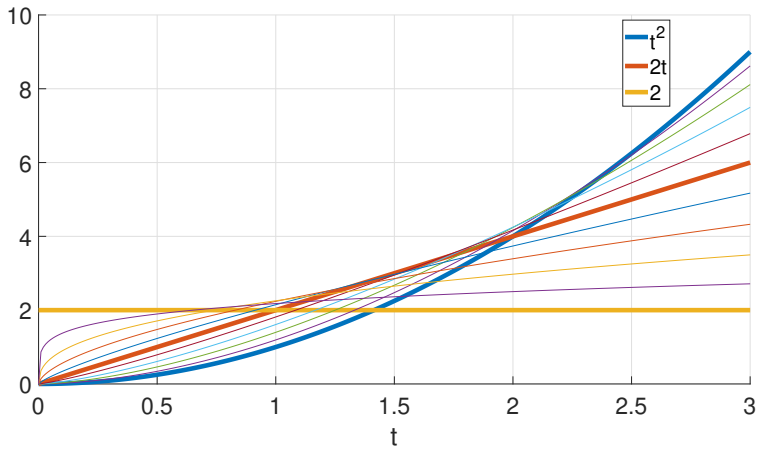


Figure 1.3. The result of Equation (1.5) being applied to $f(t) = t^2$. The thick curves are integer-order derivatives. The thin curves are fractional-order derivatives where $\alpha = 1.8, 1.6, 1.4, 1.2, 0.8, 0.6, 0.4, 0.2$ from the lowest to the highest at $t = 3$.

The second category of generalizations is based on the aforementioned second fundamental theorem of calculus. Since how to compute fractional-order derivatives is unknown, they are kept as integer-order but with the cost that integrations have to be fractional, *i.e.*

$$D^{\alpha} = D^{\lceil \alpha \rceil} D^{\alpha - \lceil \alpha \rceil} = D^{\alpha - \lceil \alpha \rceil} D^{\lceil \alpha \rceil}, \qquad \alpha \in \mathbb{R}^{+}, \tag{1.6}$$

where $D^{\lceil \alpha \rceil}$ is an integer-order derivative and $D^{\alpha - \lceil \alpha \rceil}$ is a fractional-order integral which can be derived from Cauchy's formula. The Cauchy's formula states that when integrating a function $n$ times ($n \in \mathbb{N}$), the result is equivalent to a single

integration, where

$$
\begin{aligned}
{}_cD_t^{-n} f(t) &= \underbrace{\int_c^t \cdots \int_c^t f(\tau) d\tau \cdots d\tau}_{n \text{ integrations}} = \int_c^t \frac{(t-\tau)^{n-1}}{(n-1)!} f(\tau) d\tau, \quad t > c, \\
{}_tD_c^{-n} f(t) &= \underbrace{\int_t^c \cdots \int_t^c f(\tau) d\tau \cdots d\tau}_{n \text{ integrations}} = \int_t^c \frac{(\tau-t)^{n-1}}{(n-1)!} f(\tau) d\tau, \quad t < c.
\end{aligned}
$$

Then, using the Gamma function, the above equations can also be generalized for $\alpha \in \mathbb{R}^-$, where

$$
\begin{aligned}
{}_cD_t^{\alpha} f(t) &= \int_c^t \frac{(t-\tau)^{-\alpha-1}}{\Gamma(-\alpha)} f(\tau) d\tau, \quad t > c, \\
{}_tD_c^{\alpha} f(t) &= \int_t^c \frac{(\tau-t)^{-\alpha-1}}{\Gamma(-\alpha)} f(\tau) d\tau, \quad t < c.
\end{aligned}
$$

Incorporating the above equations with the idea present in Equation (1.6), two fractional-order derivatives can be derived. The first is the *Riemann-Liouville* definition.

$$
{}^{RL}_{c}D_t^{\alpha} f(t) = \begin{cases} \displaystyle\int_c^t \frac{(t-\tau)^{-\alpha-1}}{\Gamma(-\alpha)} f(\tau) d\tau, & \alpha \in \mathbb{R}^-, \\ f(t), & \alpha = 0, \\ \displaystyle\frac{d^{\lceil \alpha \rceil}}{dt^{\lceil \alpha \rceil}} \int_c^t \frac{(t-\tau)^{\lceil \alpha \rceil - \alpha - 1}}{\Gamma(\lceil \alpha \rceil - \alpha)} f(\tau) d\tau, & \alpha \in \mathbb{R}^+. \end{cases}
$$

$$
{}^{RL}_{t}D_c^{\alpha} f(t) = \begin{cases} \displaystyle\int_t^c \frac{(\tau-t)^{-\alpha-1}}{\Gamma(-\alpha)} f(\tau) d\tau, & \alpha \in \mathbb{R}^-, \\ f(t), & \alpha = 0, \\ \displaystyle(-1)^{\lceil \alpha \rceil} \frac{d^{\lceil \alpha \rceil}}{dt^{\lceil \alpha \rceil}} \int_t^c \frac{(\tau-t)^{\lceil \alpha \rceil - \alpha - 1}}{\Gamma(\lceil \alpha \rceil - \alpha)} f(\tau) d\tau, & \alpha \in \mathbb{R}^+. \end{cases}
$$

The second is the *Caputo* definition, which swaps the sequence of derivative and

integral.

$$
{}_{c}^{C}D_{t}^{\alpha}f(t) = \begin{cases} \displaystyle\int_{c}^{t} \frac{(t-\tau)^{-\alpha-1}}{\Gamma(-\alpha)} f(\tau)d\tau, & \alpha \in \mathbb{R}^{-}, \\ f(t), & \alpha = 0, \\ \displaystyle\int_{c}^{t} \frac{(t-\tau)^{\lceil\alpha\rceil-\alpha-1}}{\Gamma(\lceil\alpha\rceil-\alpha)} \frac{d^{\lceil\alpha\rceil}}{d\tau^{\lceil\alpha\rceil}} f(\tau)d\tau, & \alpha \in \mathbb{R}^{+}. \end{cases}
$$

$$
{}_{t}^{C}D_{c}^{\alpha}f(t) = \begin{cases} \displaystyle\int_{t}^{c} \frac{(\tau-t)^{-\alpha-1}}{\Gamma(-\alpha)} f(\tau)d\tau, & \alpha \in \mathbb{R}^{-}, \\ f(t), & \alpha = 0, \\ \displaystyle(-1)^{\lceil\alpha\rceil} \int_{t}^{c} \frac{(\tau-t)^{\lceil\alpha\rceil-\alpha-1}}{\Gamma(\lceil\alpha\rceil-\alpha)} \frac{d^{\lceil\alpha\rceil}}{d\tau^{\lceil\alpha\rceil}} f(\tau)d\tau, & \alpha \in \mathbb{R}^{+}. \end{cases}
$$

One key feature of fractional-order derivatives as opposed to integer-order ones is that they are non-local, which can be observed from the above definitions since integrations are blended with derivatives.

The last category reviewed here is a generalization based on the limit definition of derivatives, where

$$
D^{1}f(t) = \frac{d}{dt}f(t) = \lim_{h\to 0} \frac{f(x+h)-f(x)}{h}.
$$

Then, the following steps can be induced, which, rigorously speaking, need proof of the synchronized convergences of different limitations by using the mean value theorem:

$$
\begin{aligned}
D^{2}f(t) &= \lim_{h\to 0} \frac{f(x+2h)-2f(x+h)+f(x)}{h^{2}}, \\
D^{3}f(t) &= \lim_{h\to 0} \frac{f(x+3h)-3f(x+2h)+3f(x+h)-f(x)}{h^{3}}.
\end{aligned}
$$

In conclusion, for $n \in \mathbb{N}$,

$$
D^{n}f(t) = \lim_{h\to 0} \frac{\sum_{k=0}^{n}(-1)^{k}\binom{n}{k}f(x+(n-k)h)}{h^{n}},
$$

where $\binom{n}{k}$ is the binomial coefficient, which is usually defined by

$$\binom{n}{k} = \frac{n!}{k!(n-k)!},$$

when both $n \in \mathbb{Z}^+$ and $0 \leq k \leq n \in \mathbb{N}$. Note that when $k > n \in \mathbb{N}$, $\binom{n}{k} = 0$ by definition. The binomial coefficient can also be generalized by the Gamma function, where

$$\binom{\alpha}{k} = \frac{\Gamma(\alpha+1)}{\Gamma(k+1)\Gamma(\alpha-k+1)} \quad \alpha \in \mathbb{R}\backslash\mathbb{Z}^-,\ k \in \mathbb{R}\backslash\mathbb{Z}^-,\ \alpha-k \in \mathbb{R}\backslash\mathbb{Z}^-.$$

As a result, the *Grünwald-Letnikov* fractional derivative is given by

$$D^\alpha f(t) = \lim_{h\to 0} \frac{\sum_{k=0}^{\infty}(-1)^k\binom{\alpha}{k}f(x+(\alpha-k)h)}{h^\alpha} \quad \alpha \in \mathbb{R}^+.$$

Note that the Grünwald-Letnikov definition is also non-local. Moreover, the upper limit of the summation is $\infty$ because the generalized binomial coefficients are non-zero for $k > \alpha$ when $\alpha \notin \mathbb{N}$ as opposed to the classical ones. In implementations, the Grünwald-Letnikov definition is often employed as a numerical approximation of fractional-order derivatives, *e.g.*, see [104, 25]. It is also worth noting that the aforementioned three definitions would not give the same result in general. They can be understood as different ways to draw a continuous curve passing through lots of discrete datapoints. So far, there are no conclusions regarding which one of them is the most useful one in any cases.

Recall that for integer-order derivatives and integrals, it can be proved through

Laplace transforms that, when $n \in \mathbb{N}$,

$$\mathcal{L}\left[\frac{d^n}{dt^n}f(t)\right] = s^n F(s) - \sum_{k=0}^{n-1} s^{n-k-1}\frac{d^k}{dt^k}f(0),$$

$$\mathcal{L}\left[\underbrace{\int_0^t \cdots \int_0^t f(\tau)d\tau \cdots d\tau}_{n \text{ generations}}\right] = s^{-n}F(s),$$

where $\mathcal{L}[f(t)] = F(s)$. In the case of fractional orders, it can be shown that the aforementioned Riemann-Liouville and Caputo definitions have the similar results, where

$$\mathcal{L}\left[{}^{RL}_{0}D^{\alpha}_{t}\right] = \begin{cases} s^{\alpha}F(s), & \alpha \in \mathbb{R}^{-}, \\ F(s), & \alpha = 0, \\ s^{\alpha}F(s) - \sum_{k=0}^{\lceil\alpha\rceil-1} s^{k}\,{}_{0}D_{t}^{\alpha-k-1}f(0), & \alpha \in \mathbb{R}^{+}. \end{cases}$$

$$\mathcal{L}\left[{}^{C}_{0}D^{\alpha}_{t}\right] = \begin{cases} s^{\alpha}F(s), & \alpha \in \mathbb{R}^{-}, \\ F(s), & \alpha = 0, \\ s^{\alpha}F(s) - \sum_{k=0}^{\lceil\alpha\rceil-1} s^{\alpha-k-1}D^{k}f(0), & \alpha \in \mathbb{R}^{+}. \end{cases}$$

Note that when $\alpha \in \mathbb{R}^{+}$, the Laplace transform of the Riemann-Liouville definition requires fractional-order initial conditions, while the Caputo definition requires integer-order initial conditions.

This dissertation demonstrates that fractional orders of $s$ naturally appear in some infinite self-similar networks' transfer functions, which means that their dynamics are inherently fractional. In addition, a more novel observation is that some infinite self-similar networks' transfer functions even consist of irrational expressions of $s$, such as $\sqrt{s+1}$. The time-domain operators corresponding to those irrational frequency-

domain expressions are still unknown. In the previous work from the author's research group, they were called implicit operators [86, 87].

In literature, the applications of fractional-order derivatives can be classified into two groups. On the one hand, they are used to describe systems' dynamics. On the other hand, they are employed to design controllers. When used to model systems, three characteristics of fractional-order derivatives are often leveraged. First of all, as mentioned above, fractional-order derivatives are non-local, meaning that when a fractional-order derivative is computed at time $t$, the history information before that instance is also required. The corresponding applications include modeling epidemics [1]. A physically based approach to non-local elasticity theory is introduced in [36]. The fact that fractional-order dynamics exist in non-local heat transfer and mechanics is shown in [18] and [32]. The second characteristic of a linear fractional-order system is that its time-domain response can follow a power law decay rate, which also leads to many applications. Chapter 1 in [91] shows a modeling example of the firing rate for premotor neurons in the visual system while an eyeball is scanning words. A fractal network model to describe a power law behavior in soft tissue in proposed in [76]. The third characteristic of fractional-order derivative is its link to the nature of infinite dimensionality, which is also the reason why fractional calculus emerges in this work. The corresponding applications embrace semi-infinite lossy transmission lines [160], heat diffusion through a semi-infinite solid [145], and neutron transport in a nuclear reactor [158].

Applying fractional calculus to controller design is often due to its extra design freedom given by the system order. The main goal is usually to outperform classical integer-order controllers. For example, a previous research work in the author's group studied fractional-order controllers for dynamic walking [85]. Other researchers use fractional-order controllers to replace high integer-order controllers for easier realizations [103]. Other extensions from classical integer-order control include fractional-

order PID control [168, 127], Kalman filter [144], state-space approaches [34, 118, 133], root-locus methods [98], and digital control [34, 119].

# CHAPTER 2

# FREQUENCY RESPONSE AND TRANSFER FUNCTIONS OF SELF-SIMILAR NETWORKED DYNAMICAL SYSTEMS

This chapter lists four algorithms for computing the dynamics of a general class of networked dynamical systems in the following four situations:

1. Frequency response for finite networks,
2. Transfer functions for finite networks,
3. Frequency response for infinite networks,
4. Transfer functions for infinite networks.

Here, a *transfer function* refers to the analytical expression of $G(s)$ which describes a system's behavior through the ratio of the output signal to its input signal in the frequency domain. Its corresponding *frequency response* is obtained by sampling $G(s)$ at a sequence of angular frequencies $\omega$, *i.e.*, $G(j\omega)$. Section 2.1 lists assumptions for that general class of networks to which the modeling methods in this chapter are applicable. In addition, it also delineates the preliminaries such as notation used in this chapter. Section 2.2 discusses recurrence formulas which are the core of all modeling algorithms. Section 2.3 talks about the two algorithms regarding finite networks. Section 2.4 briefly introduces how to evaluate infinite networks' dynamics in a special case, which is also a foundation of calculating all infinite networks' frequency response and transfer functions in this dissertation as showcased by Section 2.5. The contents in this chapter have been published in [109, 111, 112].

## 2.1 Assumptions and Preliminaries

For a network to be qualified for the approach proposed in this chapter, it must satisfy the following assumptions.

- (A-1) The network is one-dimensional.
- (A-2) The network is self-similar. That is, its topology is invariant throughout all generations.
- (A-3) All components within the network are connected either in series or in parallel.
- (A-4) All components are linear and time-invariant, such as dampers, capacitors, or transfer function blocks.
- (A-5) For infinite networks only: The network has a finite number of damaged components.
- (A-6) For infinite networks only: The network's undamaged transfer function can be obtained.

Note that the first four assumptions from (A-1) to (A-4) are for both finite and infinite networks, whereas the last two assumptions (A-5) and (A-6) are for infinite networks only. How to obtain an infinite network's undamaged transfer function mentioned in the assumption (A-6) is discussed in Section 2.4.

The definitions of a damaged network and an undamaged network are established as follows. For instance, there is a network consisting of 5 springs and 5 dampers, denoted as $k_1$ to $k_5$ and $b_1$ to $b_5$. Each type of component has its undamaged constant, indicated by the undamaged spring constant $k$ and the undamaged damper constant $b$. Then, when that network is undamaged, all components' constants are same as their corresponding undamaged ones, *i.e.*, $\forall i = 1, \ldots, 5$, $k_i = k$, and $b_i = b$. Otherwise, that network is damaged, in which a pair of two lists $(\boldsymbol{l}, \boldsymbol{e})$ is employed to represent a specific damage case. The letter $\boldsymbol{l}$ is the list of damaged components, and $\boldsymbol{e}$ is the corresponding damage amounts. For example,

$$(\boldsymbol{l}, \boldsymbol{e}) = ([k_1, b_2], [0.3, 0.4])$$

designates the damage case where $k_1 = 0.3k$ and $b_2 = 0.4b$, while all the other springs and dampers have unchanged stiffness and damping constants. Note that, in fact, $(\boldsymbol{l}, \boldsymbol{e})$ can also denote the undamaged case. In all modeling algorithms in this chapter, the undamaged case is indicated by empty lists $\boldsymbol{l}$ and $\boldsymbol{e}$. In the health monitoring part of this dissertation (Chapter 4), the undamaged case is sometimes represented by a special $\boldsymbol{e}$ where all elements in $\boldsymbol{e}$ are one.

The nomenclature of frequency response and transfer function used in this chapter is listed as follows.

- $G(s)$ is a general transfer function in Laplace variable $s$. $G(j\omega)$ is the corresponding frequency response sampled at a sequence of angular frequencies $\omega$.
- $G_r(s)$ is the recurrence formula for a self-similar network introduced in Section 2.2.
- $G_{si}(s)$ is the transfer function for the $i$-th subnetwork inside a self-similar network also introduced in Section 2.2.
- $G_1(s)$ is the transfer function when a network only has one generation first used in Section 2.3.1.
- $G_{g,(\boldsymbol{l},\boldsymbol{e})}(\cdot)$ is designated for a specific configuration of a network. The positive integer $g$ denotes the number of generations inside a network. When that network is infinitely large, $g = \infty$. The pair of two lists $(\boldsymbol{l}, \boldsymbol{e})$ indicates a particular damage case. When that network is undamaged, $(\boldsymbol{l}, \boldsymbol{e})$ is simply replaced by $\varnothing$.

Next, the three example networks which are used throughout this chapter are introduced to showcase that a vast range of networked dynamical systems could leverage the methods proposed in this dissertation. The first example is the *mechanical tree* network as shown in Figure 2.1, which has been used to model the relaxation of the aortic valve [37] and materials' viscoelastic behaviors [62] in literature. In any numerical computations in this dissertation, the undamaged constants are assumed to be $k = 2N/m$ and $b = 1Ns/m$. The dynamics of interest is the ratio of its length $X_{1,1}$ to the force exerted at $x_{1,1}$, $F$, in the frequency domain.

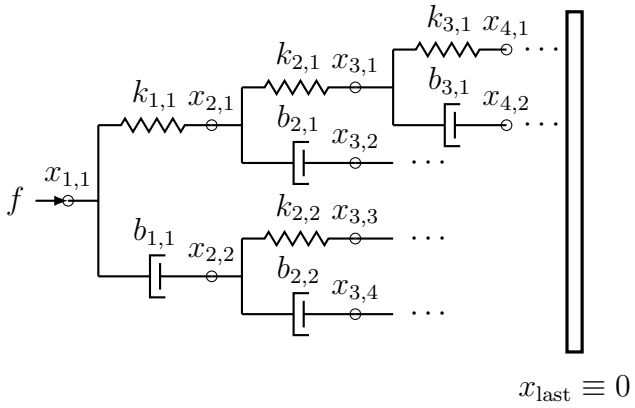


Figure 2.1. *Mechanical tree* network.

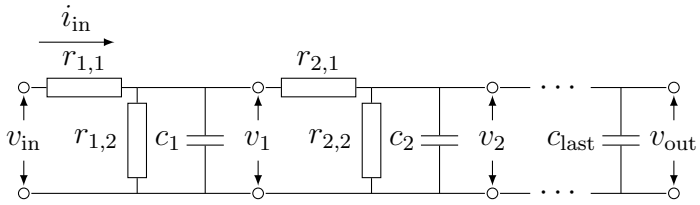


Figure 2.2. *Electrical ladder* network.

The second example is the *electrical ladder* network as shown in Figure 2.2, which is used to approximate transmission lines in literature [169, 102, 141, 22]. Its counterpart constructed by mechanical components is also utilized to approximate structures like buildings and bridges [75, 53, 57, 64]. The undamaged constants are $r_1 = 10\Omega$, $r_2 = 1k\Omega$, and $c = 100\mu F$. When undamaged, $\forall i \in \mathbb{Z}^+$, $r_{i,1} = r_1$, $r_{i,2} = r_2$, and $c_i = c$. The dynamics of interest is the input impedance $V_{\text{in}}/I_{\text{in}}$ in the frequency domain.

The third example is the *mechanical ladder* network as shown in Figure 2.3. The significance of this example is that it includes nonzero masses and proportional-integral-derivative (PID) controller blocks, which is designed purposefully to exhibit

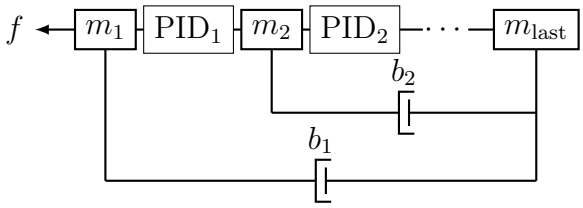


Figure 2.3. *Mechanical ladder* network.

that the methods proposed in this dissertation have potential to impact physical systems such as multi-agent structures. In fact, this example can be viewed as a line of vehicles moving together in the same direction where the separation between every two neighboring cars is maintained by a PID controller. In addition, every vehicle follows the speed of the leading vehicle ($m_{\text{last}}$) through a damper-like controller. All masses are assumed to be $1kg$. Moreover, the undamaged constants are $k_p = 10N/m$, $k_i = 0.5N/ms$, $k_d = 2Ns/m$, and $b = 1Ns/m$. The dynamics of interest is the ratio of the entire line's length $X$ to the disturbance at the $m_1$, $F$, in the frequency domain.

It is worth noting that the dynamics of interest are all at the ends in the aforementioned three examples. In fact, frequency response and transfer functions between any two nodes inside a dynamic network obeying the assumptions (A-1) to (A-6) can be obtained by the modeling algorithms illustrated in this chapter. Furthermore, the methods proposed in this chapter do not depend on the value of those assumed undamaged constants.

## 2.2 Recurrence Formula

In this dissertation, a *recurrence formula* for a self-similar network is defined as a frequency-domain equation relating its overall dynamics to its subnetworks' dynamics. As we shall see later, recurrence formulas form the core of all four modeling algorithms proposed in this chapter. The construction of recurrence formulas is

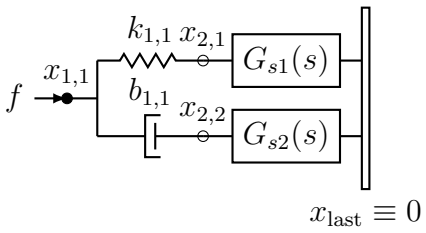


Figure 2.4. A simplified version of the mechanical tree network in Figure 2.1 used to derive its recurrence formula.

exemplified through the aforementioned three examples. The key idea is to derive a network's transfer function assuming that all of its subnetworks' transfer functions are known.

For the mechanical tree network in Figure 2.1, the transfer functions of two subnetworks are assumed to be available. That is,

$$G_{s1}(s) = \frac{X_{2,1}(s)}{F_1(s)},$$
$$G_{s2}(s) = \frac{X_{2,2}(s)}{F_2(s)},$$

where $F_1(s) + F_2(s) = F(s)$. The corresponding illustration is shown in Figure 2.4. When the first generation is taken into account, the forces $F_1(s)$ and $F_2(s)$ have their specific representations, which are

$$F_1(s) = k_{1,1}(X_{1,1}(s) - X_{2,1}(s)),$$
$$F_2(s) = b_{1,1}s(X_{1,1}(s) - X_{2,2}(s)).$$

The above two equations lead to that

$$\frac{X_{1,1}(s)}{F_1(s)} = \frac{1}{k_{1,1}} + G_{s1}(s),$$
$$\frac{X_{1,1}(s)}{F_2(s)} = \frac{1}{b_{1,1}s} + G_{s2}(s).$$

Then,

$$\frac{X_{1,1}(s)}{F(s)} = \frac{X_{1,1}(s)}{F_1(s) + F_2(s)} = \frac{1}{\dfrac{1}{\dfrac{1}{k_{1,1}} + G_{s1}(s)} + \dfrac{1}{\dfrac{1}{b_{1,1}s} + G_{s2}(s)}}.$$

Note that $X_{1,1}(s)/F(s)$ represents the dynamics of the entire mechanical tree network. As a result, the recurrence formula for the mechanical tree network is

$$\begin{aligned} G_r(s) &= \frac{1}{\dfrac{1}{\dfrac{1}{k_{1,1}} + G_{s1}(s)} + \dfrac{1}{\dfrac{1}{b_{1,1}s} + G_{s2}(s)}} \\ &= \frac{k_{1,1}b_{1,1}sG_{s1}(s)G_{s2}(s) + k_{1,1}G_{s1}(s) + b_{1,1}sG_{s2}(s) + 1}{k_{1,1}b_{1,1}s(G_{s1}(s) + G_{s2}(s)) + k_{1,1} + b_{1,1}s}. \end{aligned} \tag{2.1}$$

Note that Equation (2.1) actually directly follows the series and parallel connecting rules of idealized mechanical components since no masses are involved.

For the electrical ladder network in Figure 2.2, the input impedance of its sub-network is assumed to be known, which is indicated by

$$G_{s1}(s) = \frac{V_1(s)}{I_1(s)}.$$

The illustration of that is shown in Figure 2.5. The derivation of its recurrence formula is similar to that of the mechanical tree network's. Therefore, that is omitted

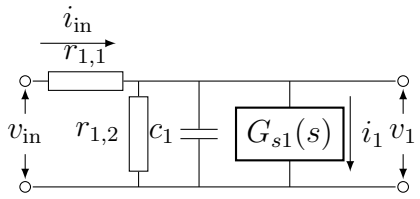


Figure 2.5. A simplified version of the electrical ladder network in Figure 2.2 used to derive its recurrence formula.

here. The result is directly led by the series and parallel connection rules of idealized electrical components (or, by the concept of equivalent impedance), *i.e.*,

$$\begin{aligned} G_r(s) = \frac{V_{\text{in}}(s)}{I_{\text{in}}(s)} &= r_{1,1} + \cfrac{1}{\cfrac{1}{r_{1,2}} + c_1 s + \cfrac{1}{G_{s1}(s)}} \\ &= \frac{(r_{1,1}r_{1,2}c_1 s + r_{1,1} + r_{1,2})\, G_{s1}(s) + r_{1,1}r_{1,2}}{(r_{1,2}c_1 s + 1)G_{s1}(s) + r_{1,2}}. \end{aligned} \tag{2.2}$$

For the mechanical ladder network in Figure 2.3, the dynamics of the subnetwork is again assumed to be known, where

$$G_{s1}(s) = \frac{X_s(s)}{F_s(s)},$$

as shown in Figure 2.6. Similar to the mechanical tree network, when the first generation is taken into account, the force $F_s(s)$ is determined by the controller block

$\text{PID}_1$. Hence,

$$G_{s1}(s) = \frac{X_s(s)}{\underbrace{\left(k_{p1} + \frac{k_{i1}}{s} + k_{d1}s\right)}_{K_1(s)} X_1(s)} = \frac{X_s(s)}{K_1(s)X_1(s)}.$$

Next, assume that $m_{\text{last}}$ is always moving at a constant speed. That can be achieved when the disturbance $f(t)$ has negligible impacts on $m_{\text{last}}$, *e.g.*, $f(t)$ is small, or the mechanical ladder network possesses many generations, or $m_{\text{last}}$ has an external controller to maintain its constant speed. From Newton's second law of motion, at $m_1$,

$$m_1 s^2(X_1(s) + X_s(s)) = F(s) - K_1(s)X_1(s) - b_1 s(X_1(s) + X_s(s)),$$

which leads to

$$(m_1 s^2 + b_1 s + K_1(s))X_1(s) + (m_1 s^2 + b_1 s)X_s(s) = F(s).$$

Because $X_s(s) = G_{s1}(s)K_1(s)X_1(s)$, we then have the following two equations

$$\begin{aligned}((m_1 s^2 + b_1 s)(G_{s1}(s)K_1(s) + 1) + K_1(s))X_1(s) &= F(s)\\ \frac{(m_1 s^2 + b_1 s)(G_{s1}(s)K_1(s) + 1) + K_1(s)}{G_{s1}(s)K_1(s)}X_s(s) &= F(s).\end{aligned}$$

Finally, the recurrence formula of the mechanical ladder network is given by

$$G_r(s) = \frac{X(s)}{F(s)} = \frac{X_1(s) + X_s(s)}{F(s)} = \frac{G_{s1}(s)K_1(s) + 1}{(m_1 s^2 + b_1 s)(G_{s1}(s)K_1(s) + 1) + K_1(s)}. \quad (2.3)$$

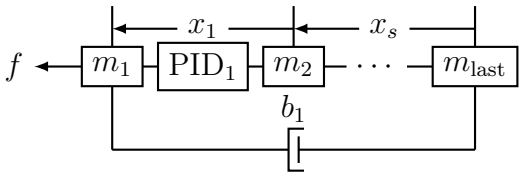


Figure 2.6. A simplified version of the mechanical ladder network in Figure 2.3 used to derive its recurrence formula.

## 2.3 Finite Networks

In this section, the two algorithms of computing frequency response and transfer functions for finite self-similar networked dynamical systems are presented. Then, in Section 2.3.3, the correctness of the results are verified.

The main idea is simply using a network's recurrence formula from the first generation repeatedly until the last one. However, the tricky part is correctly plugging the corresponding components' constants into their respective iterations. Therefore, the algorithms in this chapter leverage a recursive process to assure that aspect of correctness.

### 2.3.1 Frequency Response

The algorithm to compute the frequency response of a self-similar finite dynamic network is listed in Algorithm 1. The algorithm starts with the `partition()` function which splits the damage case of the entire network `(l,e)` into two groups. The first group is the damage case at the first generation `(l1,e1)`. The other group contains all damage cases for subnetworks, where the damage case `(lS[idx],eS[idx])` is with respect to the `idx`-th subnetwork. As a concrete example, for the mechanical tree network in Figure 2.1, suppose its damage case is

$$\texttt{(l,e)} = ([k_{1,1}, b_{2,1}, k_{3,2}, b_{3,3}], [0.1, 0.2, 0.3, 0.4]).$$

**Algorithm 1** Pseudocode of computing finite networks' frequency response. It computes the frequency response `G` at the angular frequency `w` for a finite network with `nG` number of generations given its damage case `(l,e)` and the undamaged constants `undCst`.

```
function G = freqFin(l,e,undCst,w,nG)
s = 1j*w;
[l1,e1,lS,eS] = partition(l,e);
g1Cst = getG1Cst(l1,e1,undCst);
if nG == 1 then
  G = G1(g1Cst,s);
else
  nG = nG-1;
  for idx from 1 to nS do
    GS[idx] = freqFin(lS[idx],eS[idx],undCst,w,nG);
  end for
  G = Gr(g1Cst,GS,s);
end if
```

Then, the `partition()` function returns the following results.

$$\texttt{(l1,e1)} = ([k_{1,1}], [0.1]),$$

$$\texttt{(lS[1],eS[1])} = ([b_{1,1}, k_{2,2}], [0.2, 0.3]),$$

$$\texttt{(lS[2],eS[2])} = ([b_{2,1}], [0.4]).$$

Note that the indices of the components in `lS` are with respect to their corresponding subnetworks. That is the reason why $b_{2,1}$ in `l` is converted to $b_{1,1}$ in `lS[1]`. Then, the damage case at the first generation `(l1,e1)` is used to compute the values of constants at the first generation `g1Cst` by the `getG1Cst()` function. For the above example, `g1Cst` includes that

$$k_{1,1} = 0.1k = 0.2N/m \quad \text{and} \quad b_{1,1} = b = 1Ns/m.$$

Then, the algorithm breaks into two parts determined by the criterion if the network has only one generation, *i.e.*, `nG` $= 1$. If so, the result is directly returned by

the `G1()` function given the constants at the first generation. It is worth noting that case is also the base case of the entire recursive procedure. The computations in the `G1()` function can be easily derived. For the mechanical tree network, it is

$$G_1(s) = \frac{1}{k_{1,1} + b_{1,1}s}. \tag{2.4}$$

For the electrical ladder network, it is

$$G_1(s) = r_{1,1} + \cfrac{1}{\cfrac{1}{r_{1,2}} + c_1 s} = \frac{r_{1,1}r_{1,2}c_1 s + r_{1,1} + r_{1,2}}{r_{1,2}c_1 s + 1}. \tag{2.5}$$

For the mechanical ladder network, it is

$$G_1(s) = \frac{1}{m_1 s^2 + b_1 s + K_1(s)}. \tag{2.6}$$

If the network has more than one generation, *i.e.*, `nG` $> 1$, the algorithm recursively calls itself for each of its subnetworks to compute their frequency responses. Those frequency responses are then used to compute the final result, that is the frequency response of the entire network, through those recurrence formulas implemented in the `Gr()` function.

Note that Algorithm 1 is less likely to cause mistakes as opposed to manually inserting the values of components' constants within a large network at every iteration of computation. As long as Algorithm 1 is coded correctly, that type of error would not occur. Another advantage of Algorithm 1 is its adaptability. It can deal with all finite dynamic networks satisfying the assumptions from (A-1) to (A-4) in Section 2.1.

The frequency response of the three examples under some specific situations acquired by Algorithm 1 are shown in Figures 2.7 to 2.9. Note that Algorithm 1 is also able to compute frequency response of an undamaged finite network, in which case

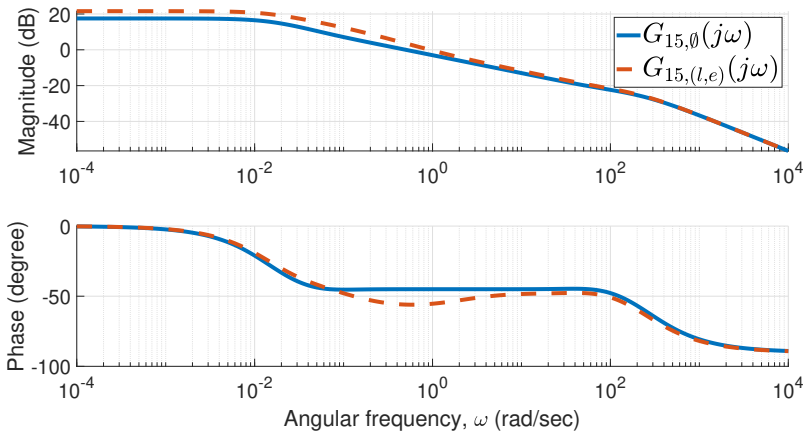


Figure 2.7. Frequency response for two 15-generation mechanical tree networks. The blue curve is for the undamaged case, $G_{15,\varnothing}(j\omega)$. The red dashed curve is for a damage case, $G_{15,(\boldsymbol{l},\boldsymbol{e})}(j\omega)$, where $(\boldsymbol{l},\boldsymbol{e}) = ([k_{2,1}, k_{2,2}, b_{3,1}], [0.1, 0.2, 0.3])$.

the input arguments `l` and `e` should be two empty lists.

### 2.3.2 Transfer Functions

This section proposes an algorithm for computing transfer functions of finite dynamic networks that satisfy the assumptions (A-1) to (A-4) in Section 2.1. The recursive structure of the algorithm in this section is the same as that of Algorithm 1. The difference is that the algorithm in this section does not depend on angular frequencies `w`, and it purely operates on the coefficients of transfer functions. That is guaranteed by the equivalence between polynomial multiplications and tensor convolutions. Therefore, in what follows, that equivalence is reviewed first.

#### 2.3.2.1 Equivalence between Polynomial Multiplication and Tensor Convolution

Here, only vector convolutions and matrix convolutions are reviewed, since they cover all the examples appearing in this dissertation. However, note that the equivalence is actually satisfied for all finite-dimensional tensors.

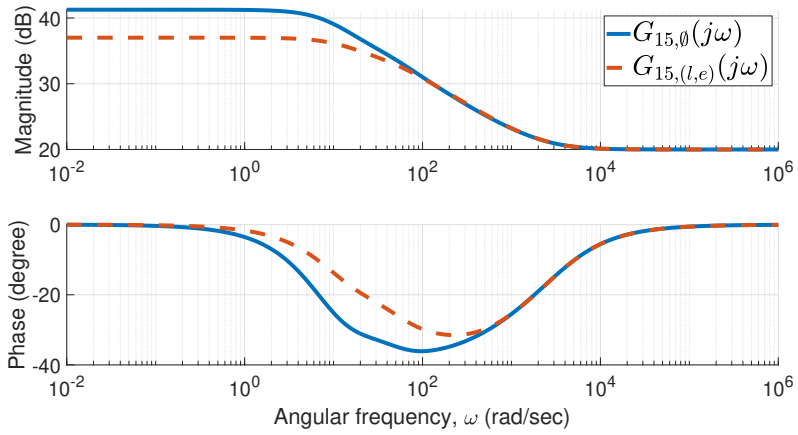


Figure 2.8. Input impedance for two 15-generation electrical ladder networks. The blue curve is for the undamaged case, $G_{15,\varnothing}(j\omega)$. The red dashed curve is for a damage case, $G_{15,(\boldsymbol{l},\boldsymbol{e})}(j\omega)$, where $(\boldsymbol{l},\boldsymbol{e}) = ([r_{2,2}],[0.1])$.

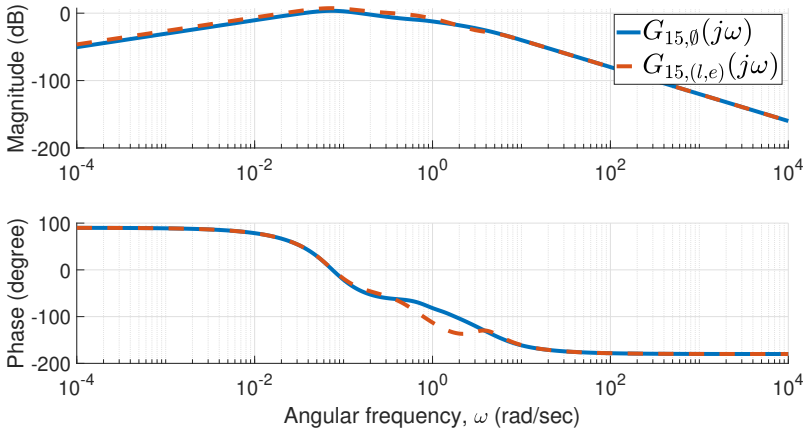


Figure 2.9. Frequency response for two 15-generation mechanical ladder networks. The blue curve is for the undamaged case, $G_{15,\varnothing}(j\omega)$. The red dashed curve is for a damage case, $G_{15,(\boldsymbol{l},\boldsymbol{e})}(j\omega)$, where $(\boldsymbol{l},\boldsymbol{e}) = ([k_{p2},k_{i2},k_{d2}],[0.1,0.1,0.1])$.

- In the case of vector convolution, $C = A * B$ is defined as

$$C(k) = \sum_p A(p)B(k-p+1),$$

where $p$ runs over all values that lead to legal subscripts of $A(p)$ and $B(k-p+1)$. For two univariate polynomials,

$$\begin{aligned} a(x) &= a_1 x^{n_A-1} + a_2 x^{n_A-2} + \cdots + a_{n_A}, \\ b(x) &= b_1 x^{n_B-1} + b_2 x^{n_B-2} + \cdots + b_{n_B}, \end{aligned}$$

their coefficient vectors are defined as

$$\begin{aligned} A &= \begin{bmatrix} a_1 & a_2 & \cdots & a_{n_A} \end{bmatrix}, \\ B &= \begin{bmatrix} b_1 & b_2 & \cdots & b_{n_B} \end{bmatrix}. \end{aligned}$$

It can be shown that the vector convolution of $A$ and $B$, $C = A * B$, is the coefficient vector of the corresponding univariate polynomial multiplication,

$$c(x) = a(x)b(x).$$

- In the case of matrix convolution, $C = A * B$, is defined as

$$C(j,k) = \sum_p \sum_q A(p,q)B(j-p+1, k-q+1),$$

where $p$ and $q$ run over all values that lead to legal subscripts of $A(p,q)$ and $B(j-p+1, k-q+1)$. For two bivariate polynomial,

$$\begin{aligned} a(x,y) &= a_{1,1}x^{n_A-1}y^0 + \cdots + a_{1,n_A}x^0y^{n_A-1} + a_{2,2}x^{n_A-2}y^0 + \cdots + a_{2,n_A}x^0y^{n_A-2} \\ &\quad + \cdots + a_{n_A,n_A}x^0y^0, \\ b(x,y) &= b_{1,1}x^{n_B-1}y^0 + \cdots + b_{1,n_B}x^0y^{n_B-1} + b_{2,2}x^{n_B-2}y^0 + \cdots + b_{2,n_B}x^0y^{n_B-2} \\ &\quad + \cdots + b_{n_B,n_B}x^0y^0, \end{aligned}$$

their coefficient matrices are defined as

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n_A-1} & a_{1,n_A} \\ 0 & a_{2,2} & \cdots & a_{2,n_A-1} & a_{2,n_A} \\ 0 & 0 & \cdots & a_{3,n_A-1} & a_{3,n_A} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & a_{n_A,n_A} \end{bmatrix},$$

$$B = \begin{bmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,n_B-1} & b_{1,n_B} \\ 0 & b_{2,2} & \cdots & b_{2,n_B-1} & b_{2,n_B} \\ 0 & 0 & \cdots & b_{3,n_B-1} & b_{3,n_B} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & b_{n_B,n_B} \end{bmatrix}.$$

It can be shown that the matrix convolution of $A$ and $B$, $C = A * B$, is the coefficient matrix of the corresponding bivariate polynomial multiplication,

$$c(x, y) = a(x, y)b(x, y).$$

Additionally, a related new operator for additions between two coefficient vectors and matrices is defined here. The operator $\oplus$ can be applied between two coefficient vectors or matrices with different dimensions. The result of that should be consistent with the addition between two univariate or bivariate polynomials. Two examples are provided as follows.

- In the case of between two coefficient vectors,

$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \oplus \begin{bmatrix} b_1 & b_2 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 + b_1 & a_3 + b_2 \end{bmatrix}. \tag{2.7}$$

- In the case of between two coefficient matrices,

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ 0 & a_{2,2} \end{bmatrix} \oplus \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ 0 & b_{2,2} & b_{2,3} \\ 0 & 0 & b_{3,3} \end{bmatrix} = \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ 0 & a_{1,1} + b_{2,2} & a_{1,2} + b_{2,3} \\ 0 & 0 & a_{2,2} + b_{3,3} \end{bmatrix}. \tag{2.8}$$

#### 2.3.2.2 Algorithm for Transfer Functions

The computation of finite networks' transfer functions is based on the observation as follows. Recall that the computation of a finite network's frequency response repeatedly uses its recurrence formula $G_r(s)$ starting with the transfer function when

it only has one generation, $G_1(s)$. Moreover, note that both $G_r(s)$ and $G_1(s)$ are rational expressions where the numerator and denominator of both are polynomials in the Laplace variable $s$. Therefore, the result of combining them together, that is a finite network's transfer function, is also a rational expression of $s$. Hence, the coefficient vectors of both its numerator and denominator can be obtained directly, which leads to the analytical expression of a finite network's transfer function.

For the mechanical tree example, its one-generation transfer function $G_1(s)$ in Equation (2.4) can be converted to two coefficient vectors for the numerator and denominator, $\boldsymbol{c_{N1}}$ and $\boldsymbol{c_{D1}}$, where

$$\boldsymbol{c_{N1}} = \begin{bmatrix} 1 \end{bmatrix},$$
$$\boldsymbol{c_{D1}} = \begin{bmatrix} b_{1,1} & k_{1,1} \end{bmatrix}.$$

Define the transfer functions of two subnetworks as

$$G_{s1}(s) = \frac{N_{s1}(s)}{D_{s1}(s)}, \quad \text{and} \quad G_{s2}(s) = \frac{N_{s2}(s)}{D_{s2}(s)}.$$

Substituting them into the recurrence formula in Equation (2.1) leads to

$$G_r(s) = \frac{k_{1,1}b_{1,1}sN_{s1}N_{s2} + k_{1,1}N_{s1}D_{s2} + b_{1,1}sN_{s2}D_{s1} + D_{s1}D_{s2}}{k_{1,1}b_{1,1}s(N_{s1}D_{s2} + N_{s2}D_{s1}) + (k_{1,1} + b_{1,1}s)D_{s1}D_{s2}}.$$

Therefore, if the coefficient vectors for the two subnetworks are defined as $\boldsymbol{c_{Ns1}}$, $\boldsymbol{c_{Ds1}}$, $\boldsymbol{c_{Ns2}}$, and $\boldsymbol{c_{Ds2}}$, its recurrence formula $G_r(s)$ in Equation (2.1) can be converted to

the following two equations which only operate on the coefficients.

$$\begin{aligned}\boldsymbol{c_{Nr}} = &\begin{bmatrix} k_{1,1}b_{1,1} & 0 \end{bmatrix} * \boldsymbol{c_{Ns1}} * \boldsymbol{c_{Ns2}} \oplus \begin{bmatrix} k_{1,1} \end{bmatrix} * \boldsymbol{c_{Ns1}} * \boldsymbol{c_{Ds2}} \\ &\oplus \begin{bmatrix} b_{1,1} & 0 \end{bmatrix} * \boldsymbol{c_{Ns2}} * \boldsymbol{c_{Ds1}} \oplus \boldsymbol{c_{Ds1}} * \boldsymbol{c_{Ds2}}, \end{aligned} \tag{2.9}$$

$$\boldsymbol{c_{Dr}} = \begin{bmatrix} k_{1,1}b_{1,1} & 0 \end{bmatrix} * (\boldsymbol{c_{Ns1}} * \boldsymbol{c_{Ds2}} \oplus \boldsymbol{c_{Ns2}} * \boldsymbol{c_{Ds1}}) \oplus \begin{bmatrix} b_{1,1} & k_{1,1} \end{bmatrix} * \boldsymbol{c_{Ds1}} * \boldsymbol{c_{Ds2}}, \tag{2.10}$$

where the operator $*$ means vector convolutions, and $\oplus$ is defined as the addition between two coefficient vectors according to Equation (2.7).

Similar procedures can be applied to the other two examples. For the electrical ladder network, its one-generation transfer function $G_1(s)$ in Equation (2.5) can be converted to

$$\begin{aligned} \boldsymbol{c_{N1}} &= \begin{bmatrix} r_{1,1}r_{1,2}c_1 & r_{1,1} + r_{1,2} \end{bmatrix}, \\ \boldsymbol{c_{D1}} &= \begin{bmatrix} r_{1,2}c_1 & 1 \end{bmatrix}, \end{aligned}$$

and its recurrence formula $G_r(s)$ in Equation (2.2) can be converted to

$$\boldsymbol{c_{Nr}} = \begin{bmatrix} r_{1,1}r_{1,2}c_1 & r_{1,1} + r_{1,2} \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus r_{1,1}r_{1,2}\boldsymbol{c_{Ds1}}, \tag{2.11}$$

$$\boldsymbol{c_{Dr}} = \begin{bmatrix} r_{1,2}c_1 & 1 \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus r_{1,2}\boldsymbol{c_{Ds1}}. \tag{2.12}$$

For the mechanical ladder network, its one-generation transfer function $G_1(s)$ in Equation (2.6) can be converted to

$$\begin{aligned} \boldsymbol{c_{N1}} &= \begin{bmatrix} 1 & 0 \end{bmatrix}, \\ \boldsymbol{c_{D1}} &= \begin{bmatrix} m_1 & b_1 + k_{d1} & k_{p1} & k_{i1} \end{bmatrix}. \end{aligned}$$

and its recurrence formula $G_r(s)$ in Equation (2.3) can be converted to

$$\boldsymbol{c_{Nr}} = \begin{bmatrix} k_{d1} & k_{p1} & k_{i1} \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus \begin{bmatrix} 1 & 0 \end{bmatrix} * \boldsymbol{c_{Ds1}},$$
$$\boldsymbol{c_{Dr}} = \begin{bmatrix} m_1 & b_1 & 0 \end{bmatrix} * \begin{bmatrix} k_{d1} & k_{p1} & k_{i1} \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus \begin{bmatrix} m_1 & b_1 + k_{d1} & k_{p1} & k_{i1} \end{bmatrix} * \boldsymbol{c_{Ds1}}.$$

Finally, the pseudocode of computing finite networks' transfer functions is listed in Algorithm 2. The structure is same as its counterpart for frequency response in

**Algorithm 2** Pseudocode of computing finite networks' transfer function. It computes the coefficient vectors `cN` and `cD` of an `nG`-generation network's transfer function given its damage case `(l,e)` and the undamaged constants `undCst`.

```
1: function [cN,cD] = tranFin(l,e,undCst,nG)
2: [l1,e1,lS,eS] = partition(l,e);
3: g1Cst = getG1Cst(l1,e1,undCst);
4: if nG == 1 then
5:   [cN,cD] = C1(g1Cst);
6: else
7:   nG = nG-1;
8:   for idx from 1 to nS do
9:     [cNS[idx],cDS[idx]] = tranFin(lS[idx],eS[idx],undCst,nG);
10:  end for
11:  [cN,cD] = Cr(g1Cst,cNS,cDS);
12:  [cN,cD] = simplify(cN,cD);
13: end if
```

Algorithm 1. The main difference is that Algorithm 2 is independent of angular frequencies `w`, and it returns the coefficient vectors `cN` and `cD` of a transfer function $G(s)$. The `G1()` function in Algorithm 1 is replaced by the `C1()` function in Algorithm 2 which returns $\boldsymbol{c_{N1}}$ and $\boldsymbol{c_{D1}}$. The `Gr()` function in Algorithm 1 is replaced by the `Cr()` function in Algorithm 2 whose implementation is given by the expressions for $\boldsymbol{c_{Nr}}$ and $\boldsymbol{c_{Dr}}$. In addition, there exists a new function called `simplify()` in Algo-

rithm 2, which reduces the resultant coefficient vectors from the `Cr()` function. For example, the order of two coefficient vectors can be lowered by the same amount, like

$$\boldsymbol{c_N} = \begin{bmatrix} 1 & 2 & 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{c_D} = \begin{bmatrix} 3 & 4 & 0 & 0 \end{bmatrix}$$

being simplified to

$$\boldsymbol{c_N} = \begin{bmatrix} 1 & 2 \end{bmatrix} \quad \text{and} \quad \boldsymbol{c_D} = \begin{bmatrix} 3 & 4 & 0 \end{bmatrix}.$$

Another possible simplification is that all elements in both $\boldsymbol{c_N}$ and $\boldsymbol{c_D}$ can be divided by the same number, such as

$$\boldsymbol{c_N} = \begin{bmatrix} 2 & 4 \end{bmatrix} \quad \text{and} \quad \boldsymbol{c_D} = \begin{bmatrix} 6 & 8 \end{bmatrix}$$

being simplified to

$$\boldsymbol{c_N} = \begin{bmatrix} 1 & 2 \end{bmatrix} \quad \text{and} \quad \boldsymbol{c_D} = \begin{bmatrix} 3 & 4 \end{bmatrix}.$$

The results for the three examples in some specific situations are presented as follows. For a two-generation mechanical tree network whose damage case is $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{2,1}, k_{2,2}], [0.1, 0.2])$, its transfer function is

$$G_{2,(\boldsymbol{l},\boldsymbol{e})}(s) = \frac{2s^2 + 4.8s + 0.88}{s^3 + 6.6s^2 + 2.48s + 0.16}. \tag{2.13}$$

For a four-generation electrical ladder network whose damage case is

$$(\boldsymbol{l}, \boldsymbol{e}) = ([r_{2,2}, r_{3,2}], [0.1, 0.1]),$$

the analytical expression of its input impedance is

$$G_{4,(\boldsymbol{l},\boldsymbol{e})}(s) = \frac{s^4 + 7220s^3 + 1.6 \times 10^7 s^2 + 1.2 \times 10^{10} s + 1.6 \times 10^{12}}{0.1s^4 + 622s^3 + 1.1 \times 10^6 s^2 + 5 \times 10^8 s + 2.4 \times 10^{10}}. \tag{2.14}$$

For a two-generation mechanical ladder network whose damage case is

$$(\boldsymbol{l}, \boldsymbol{e}) = ([k_{p2}, k_{i2}, k_{d2}], [0.1, 0.1, 0.1]),$$

its transfer function is

$$G_{2,(\boldsymbol{l},\boldsymbol{e})}(s) = \frac{s^4 + 3.2s^3 + 11s^2 + 0.55s}{s^6 + 6.2s^5 + 26.6s^4 + 26.05s^3 + 11.25s^2 + s + 0.025}. \tag{2.15}$$

Again, it is worth emphasizing that Algorithm 2 is able to compute transfer function of an undamaged finite network, in which case the input arguments `l` and `e` should be two empty lists.

### 2.3.3 Correctness Check

The correctness of the results obtained by the two algorithms in Sections 2.3.1 and 2.3.2 is checked here. That confirmation is twofold. First, a frequency response obtained from Algorithm 1 should be consistent with its transfer function obtained from Algorithm 2. The comparison is cast between a frequency response $G(j\omega)$ from Algorithm 1 and a sampling of its transfer function $G(s)$ from Algorithm 2 at a sequence of frequencies. That consistency is displayed by Figures 2.10 to 2.12.

The second confirmation is whether the frequency response and transfer functions obtained by Algorithms 1 and 2 are consistent with their time-domain response. On the one hand, the time-domain response of a network is obtained by the `lsim()` function in `MATLAB` given its transfer function from Algorithm 2 and an input signal $u(t)$. On the other hand, that time-domain response can also be obtained by the

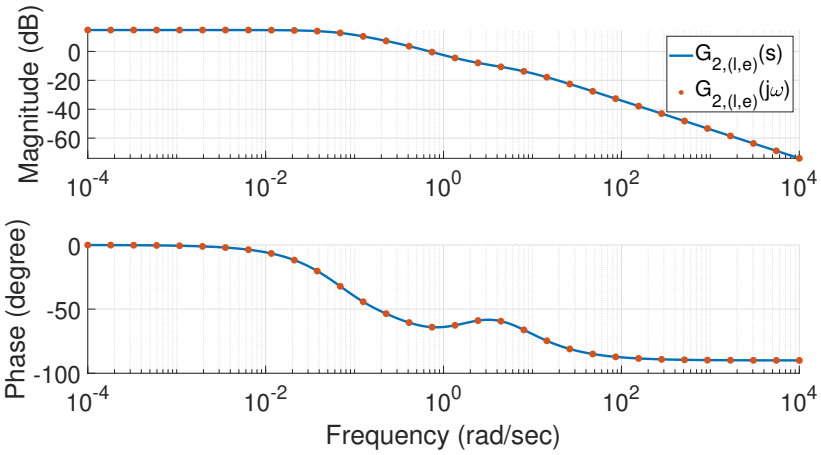


Figure 2.10. The consistency between a finite mechanical tree network's transfer function in Equation (2.13) and its corresponding frequency response from Algorithm 1.

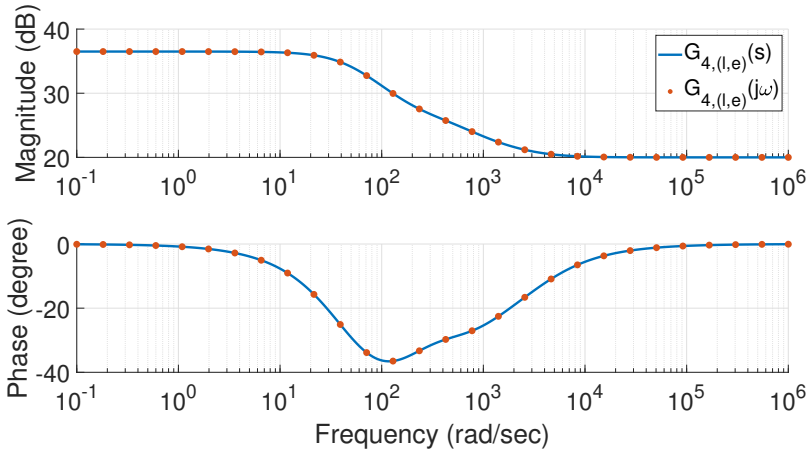


Figure 2.11. The consistency between a finite electrical ladder network's transfer function in Equation (2.14) and its corresponding frequency response from Algorithm 1.

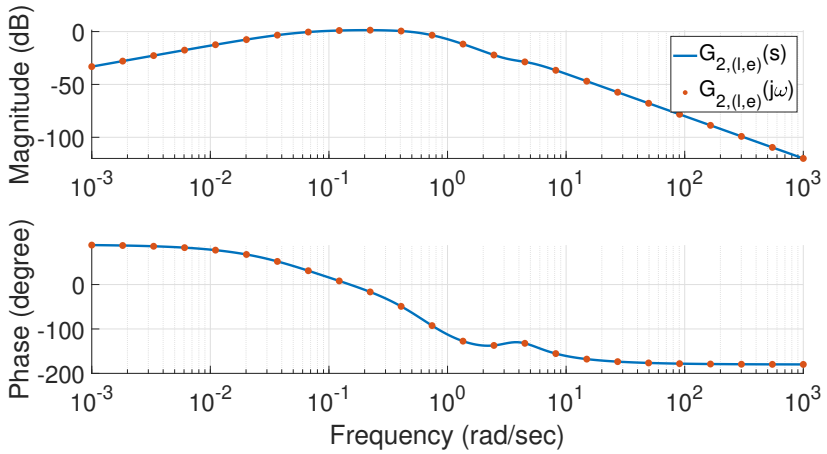


Figure 2.12. The consistency between a finite mechanical ladder network's transfer function in Equation (2.15) and its corresponding frequency response from Algorithm 1.

numerical integration of the differential equations describing that network's dynamics given the same input signal $u(t)$.

For the mechanical tree network in Figure 2.1 with two generations, the following equations of force balance can be formulated.

$$\begin{aligned} f &= k_{1,1}(x_{1,1} - x_{2,1}) + b_{1,1}(\dot{x}_{1,1} - \dot{x}_{2,2}), \\ k_{1,1}(x_{1,1} - x_{2,1}) &= k_{2,1}x_{2,1} + b_{2,1}\dot{x}_{2,1}, \\ b_{1,1}(\dot{x}_{1,1} - \dot{x}_{2,2}) &= k_{2,2}x_{2,2} + b_{2,2}\dot{x}_{2,2}, \end{aligned}$$

which result in a system of equations

$$\begin{bmatrix} b_{1,1} & 0 & -b_{1,1} \\ 0 & b_{2,1} & 0 \\ b_{1,1} & 0 & -b_{1,1} - b_{2,2} \end{bmatrix} \underbrace{\begin{bmatrix} \dot{x}_{1,1} \\ \dot{x}_{2,1} \\ \dot{x}_{2,2} \end{bmatrix}}_{\dot{x}} = \begin{bmatrix} f - k_{1,1}(x_{1,1} - x_{2,1}) \\ k_{1,1}(x_{1,1} - x_{2,1}) - k_{2,1}x_{2,1} \\ k_{2,2}x_{2,2} \end{bmatrix}.$$

Then, the time-domain response is obtained by using the `ode45()` function in `MATLAB`

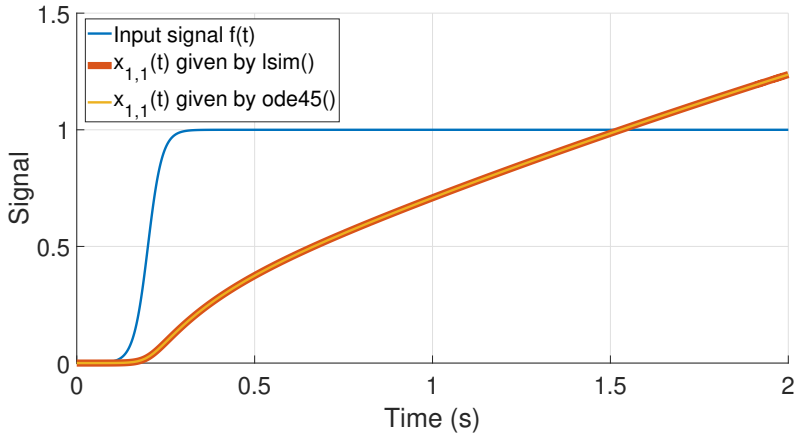


Figure 2.13. Two ways of obtaining a finite mechanical tree network's time-domain response $x_{1,1}(t)$ give the same result, which confirms the correctness of the transfer function in Equation (2.13).

to integrate $\dot{x}$ solved from the above system of equations. The input signal is a logistic function $f(t) = 1/(1 + e^{-50(t-0.2)})$. The consistency between the above result and the one given by the `lsim()` with the transfer function in Equation (2.13) for the same input $f(t)$ is shown in Figure 2.13, which confirms the correctness from the perspective of time-domain response.

For the electrical ladder network in Figure 2.2 with four generations, the following system of differential equations is derived from Kirchhoff's circuit laws.

$$\begin{aligned}
\dot{v}_1 &= \frac{1}{c_1}\left(i_{\text{in}} - \frac{v_1}{r_{1,2}} - \frac{v_1 - v_2}{r_{2,1}}\right), \\
\dot{v}_2 &= \frac{1}{c_2}\left(\frac{v_1 - v_2}{r_{2,1}} - \frac{v_2}{r_{2,2}} - \frac{v_2 - v_3}{r_{3,1}}\right), \\
\dot{v}_3 &= \frac{1}{c_3}\left(\frac{v_2 - v_3}{r_{3,1}} - \frac{v_3}{r_{3,2}} - \frac{v_3 - v_4}{r_{4,1}}\right), \\
\dot{v}_4 &= \frac{1}{c_4}\left(\frac{v_3 - v_4}{r_{4,1}} - \frac{v_4}{r_{4,2}}\right)
\end{aligned}$$

The numerical integration of the above system of differential equations is performed by the `ode45()` function given the input signal $i_{\text{in}}(t) = 1/(1 + e^{-50(t-0.2)})$. Once

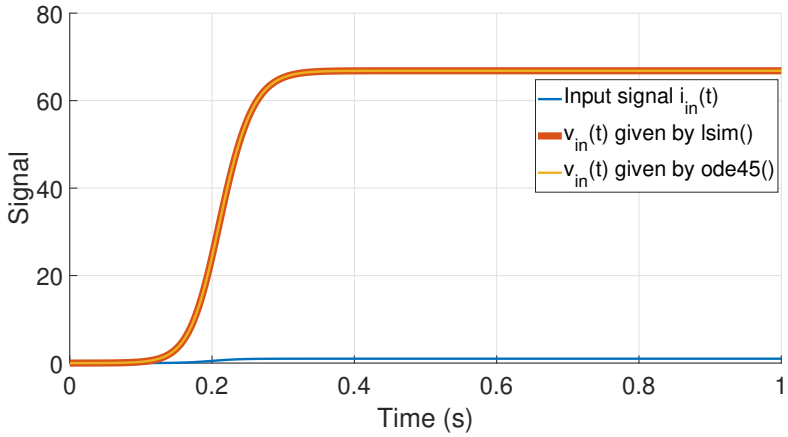


Figure 2.14. Two ways of obtaining a finite electrical ladder network's time-domain response $v_{\text{in}}(t)$ give the same result, which confirms the correctness of the transfer function in Equation (2.14).

$v_1(t)$ is obtained from the integration, $v_{\text{in}}(t)$ can also be computed from $v_{\text{in}}(t) = v_1(t) + r_{1,1}i_{\text{in}}(t)$. That is compared with the result given by the `lsim()` on the transfer function in Equation (2.14), which is plotted in Figure 2.14.

For the mechanical ladder network in Figure 2.3 with two generations, the following equations of motion can be acquired by Newton's second law.

$$
\begin{aligned}
m_1\ddot{x}_1 &= f - k_{p1}(x_1 - x_2) - k_{i1}\int_0^t (x_1 - x_2)d\tau - k_{d1}(\dot{x}_1 - \dot{x}_2) - b_1\dot{x}_1, \\
m_2\ddot{x}_2 &= k_{p1}(x_1 - x_2) + k_{i1}\int_0^t (x_1 - x_2)d\tau + k_{d1}(\dot{x}_1 - \dot{x}_2) - k_{p2}x_2 - k_{i2}\int_0^t x_2 d\tau \\
&\quad - k_{d2}\dot{x}_2 - b_2\dot{x}_2.
\end{aligned}
$$

Note that $x_1$ and $x_2$ are the absolute displacements of $m_1$ and $m_2$ which are different from those in Figure 2.6. The above two equations of motion can be organized into

a matrix form, where

$$\underbrace{\begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix}}_{\boldsymbol{M}} \underbrace{\begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{bmatrix}}_{\ddot{\boldsymbol{X}}} + \underbrace{\begin{bmatrix} k_{d1}+b_1 & -k_{d1} \\ -k_{d1} & k_{d1}+k_{d2}+b_2 \end{bmatrix}}_{\boldsymbol{B}} \underbrace{\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}}_{\dot{\boldsymbol{X}}} + \underbrace{\begin{bmatrix} k_{p1} & -k_{p1} \\ -k_{p1} & k_{p1}+k_{p2} \end{bmatrix}}_{\boldsymbol{K}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\boldsymbol{X}}$$
$$= \underbrace{\begin{bmatrix} f - k_{i1}\int_0^t (x_1 - x_2)d\tau \\ k_{i1}\int_0^t (x_1 - x_2)d\tau - k_{i2}\int_0^t x_2 d\tau \end{bmatrix}}_{\boldsymbol{U}}.$$

If define that $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{X}^\top & \dot{\boldsymbol{X}}^\top \end{bmatrix}^\top$, the corresponding system of first-order differential equations is given by

$$\dot{\boldsymbol{Q}} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{M} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ -\boldsymbol{K} & -\boldsymbol{B} \end{bmatrix} \boldsymbol{Q} + \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{M} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{U} \end{bmatrix},$$

which can then be numerically integrated given the input signal $f(t) = 1/(1 + e^{-50(t-0.2)})$, where $\boldsymbol{I}$ is the identity matrix. Because $\boldsymbol{U}$ contains the integrations of state variables, `ode45()` in not employed this time. Instead, a Riemann sum with constant time steps is performed. The resultant length of the entire two-generation mechanical ladder network, $x_1(t)$, is compared with the result given by the `lsim()` on the transfer function in Equation (2.15), which is plotted in Figure 2.15.

## 2.4 Undamaged Infinite Networks' Transfer Functions

For an infinite network that satisfies the assumptions (A-1) to (A-6) in Section 2.1, its transfer function when undamaged is a crucial part of finding other transfer functions in more general cases. In fact, that plays the same role as those one-generation transfer functions $G_1(s)$ for finite networks in Algorithms 1 and 2. Therefore, how to compute them is reviewed in this section. Note that the method to evaluate them

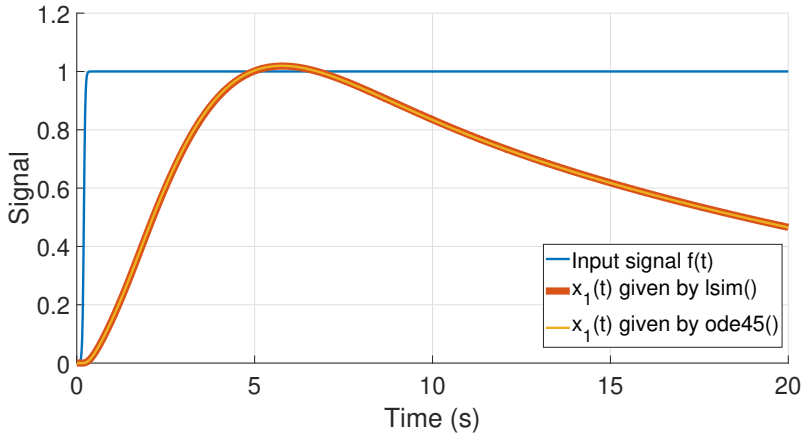


Figure 2.15. Two ways of obtaining a finite mechanical ladder network's time-domain response $x_1(t)$ give the same result, which confirms the correctness of the transfer function in Equation (2.15).

is from existing work in literature [55, 84, 94] rather than the author's own work. However, it is still repeated here for the purpose of completeness.

One observation is that when a self-similar infinite network is undamaged, its transfer function is the same as its subnetworks', *i.e.*, $G_{si}(s) = G_{\infty,\varnothing}(s)$ for all $i$. Specifically, for an infinite mechanical tree network, when undamaged, the following relation can be established from its recurrence formula in Equation (2.1).

$$G_{\infty,\varnothing}(s) = \frac{kbsG^2_{\infty,\varnothing}(s) + (bs+k)G_{\infty,\varnothing}(s) + 1}{2kbsG_{\infty,\varnothing}(s) + k + bs}.$$

Note that $k_{1,1} = k$ and $b_{1,1} = b$ in the case of no damage. The above equation leads to

$$kbsG^2_{\infty,\varnothing}(s) = 1,$$

which results in the undamaged transfer function for an infinite mechanical tree network being

$$G_{\infty,\varnothing}(s) = \frac{1}{\sqrt{kbs}}. \tag{2.16}$$

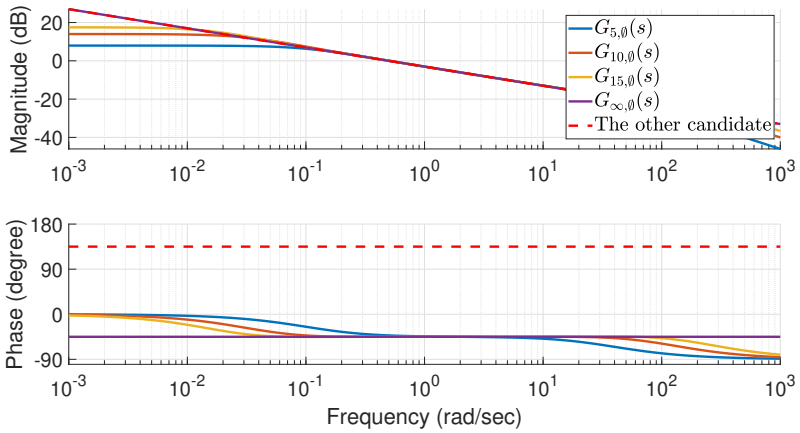


Figure 2.16. Finite mechanical tree networks' undamaged transfer functions $G_{g,\varnothing}(s)$ converge to the infinite mechanical tree network's undamaged transfer function $G_{\infty,\varnothing}(s)$ in Equation (2.16) as $g \to \infty$. The other candidate $-1/\sqrt{kbs}$ is not the limiting point of convergence, so it is not suitable.

The other candidate result $-1/\sqrt{kbs}$ is not suitable because it is actually a non-minimum-phase system which is physically impossible for a mechanical tree network in Figure 2.1. That can be observed from the fact that, when undamaged, finite mechanical trees' transfer functions converge to $1/\sqrt{kbs}$ instead of the other one as the number of generations grows. (See Figure 2.16.)

Similar procedures can be applied to the other two examples. For an infinite electrical ladder network, when undamaged, its recurrence formula leads to the following equation,

$$G_{\infty,\varnothing}(s) = \frac{(r_1 r_2 cs + r_1 + r_2)\, G_{\infty,\varnothing}(s) + r_1 r_2}{(r_2 cs + 1) G_{\infty,\varnothing}(s) + r_2},$$

where $r_{1,1} = r_1$, $r_{1,2} = r_2$, and $c_1 = c$ when undamaged. That further leads to a quadratic equation in $G_{\infty,\varnothing}(s)$,

$$(r_2 cs + 1) G_{\infty,\varnothing}^2(s) - (r_1 r_2 cs + r_1) G_{\infty,\varnothing}(s) - r_1 r_2 = 0.$$

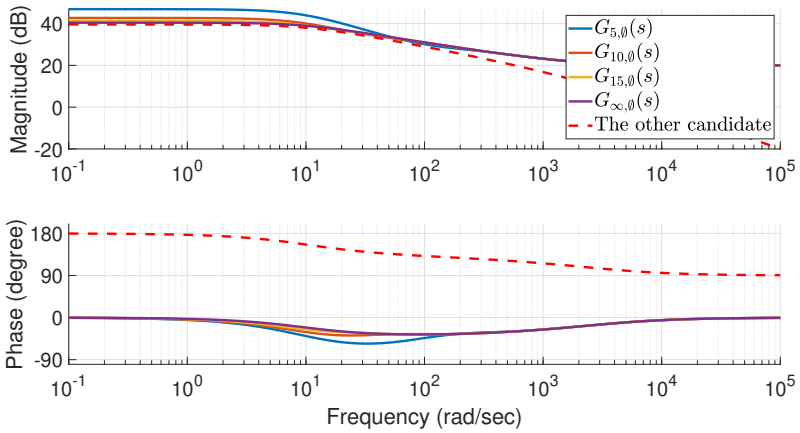


Figure 2.17. Finite electrical ladder networks' undamaged transfer functions $G_{g,\varnothing}(s)$ converge to the infinite electrical ladder network's undamaged transfer function $G_{\infty,\varnothing}(s)$ in Equation (2.17) as $g \to \infty$. The other candidate is not the limiting point of convergence, so it is not suitable.

Therefore, the undamaged transfer function of infinite electrical ladder network is

$$G_{\infty,\varnothing}(s) = \frac{s + \dfrac{1}{r_2 c} + \sqrt{s^2 + \dfrac{2r_1 + 4r_2}{r_1 r_2 c}s + \dfrac{r_1 + 4r_2}{r_1 r_2^2 c^2}}}{\dfrac{2}{r_1}s + \dfrac{2}{r_1 r_2 c}}. \tag{2.17}$$

The other candidate is also eliminated for the same reason, *i.e.*, it is not the limiting point of convergence. That can be observed in Figure 2.17.

For an infinite mechanical ladder network, when undamaged, its recurrence formula results in

$$G_{\infty,\varnothing}(s) = \frac{G_{\infty,\varnothing}(s)K(s) + 1}{(ms^2 + bs)(G_{\infty,\varnothing}(s)K(s) + 1) + K(s)},$$

where $K(s) = k_p + k_i/s + k_d s$. The equation leads to another quadratic equation in

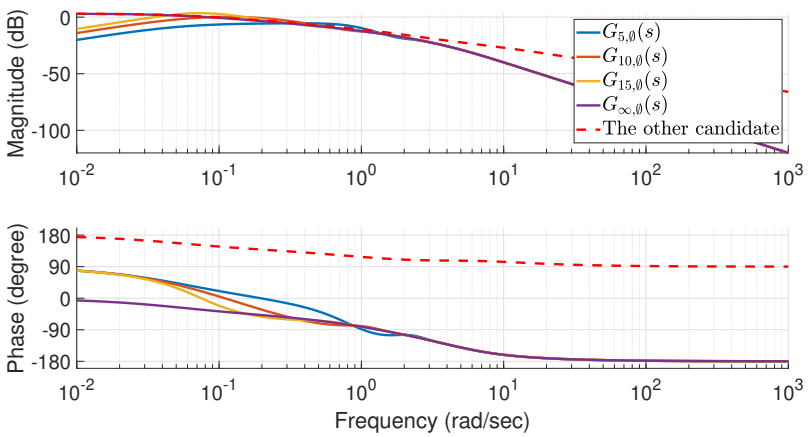


Figure 2.18. Finite mechanical ladder networks' undamaged transfer functions $G_{g,\varnothing}(s)$ converge to the infinite mechanical ladder network's undamaged transfer function $G_{\infty,\varnothing}(s)$ in Equation (2.18) as $g \to \infty$. The other candidate is not the limiting point of convergence, so it is not suitable.

$G_{\infty,\varnothing}(s)$,

$$[mk_d s^3 + (mk_p + bk_d)s^2 + (mk_i + bk_p)s + bk_i]G^2_{\infty,\varnothing}(s) + (ms^2 + bs)G_{\infty,\varnothing}(s) - 1 = 0.$$

Therefore, the undamaged transfer function of infinite mechanical ladder network is

$$G_{\infty,\varnothing}(s) = \frac{-ms^2 - bs + A(s)}{2[mk_d s^3 + (mk_p + bk_d)s^2 + (mk_i + bk_p)s + bk_i]}, \tag{2.18}$$

where

$$A(s) = [m^2 s^4 + (2mb + 4mk_d)s^3 + (b^2 + 4mk_p + 4bk_d)s^2 + 4(mk_i + bk_p)s + 4bk_i]^{\frac{1}{2}}.$$

Again, the other candidate is eliminated for the same reason, and the convergence of finite mechanical ladder networks' undamaged transfer functions to the infinite one is shown in Figure 2.18.

The following section shows how those undamaged transfer functions are employed to compute other transfer functions of infinite networks in more general cases.

## 2.5 General Infinite Networks

In this section, two algorithms to compute frequency response and transfer functions of infinite self-similar dynamic networks are presented, especially when they are damaged. The basic idea is similar to its counterpart for finite networks in Section 2.3. However, some indispensable changes are made to adapt for infinite networks.

### 2.5.1 Frequency Response

Recall that when evaluating frequency response for a finite network, the entire procedure starts with its one-generation transfer function $G_1(s)$ and employs its recurrence formula $G_r(s)$ in iterations until its total number of generations is achieved. That does not fit the case of infinite networks, as the number of iterations would become infinite. To overcome that difficulty, the assumption (A-5) in Section 2.1 is set which requires that an infinite network has a finite number of damaged components. For an infinite network fulfilling that assumption, it has to contain a deepest generation after which all subnetworks are undamaged. Then, that deepest generation is the starting point of the entire computation, thus undamaged transfer functions for infinite networks $G_{\infty,\varnothing}(s)$ play the same role as one-generation transfer functions for finite networks $G_1(s)$ in the entire modeling procedure. By doing that, the number of iterations is limited to a finite number for infinite networks.

As a simple example, assume that an infinite mechanical tree network in Figure 2.1 has a damage case $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{1,1}], [0.1])$. Then, both subnetworks after the second generation are undamaged. Since the network is infinitely large, both of them have the same transfer functions as in Equation (2.16). That would be the starting point of the computation. In fact, the frequency response of that infinite mechanical tree

network can be evaluated by only using the recurrence formula in Equation (2.1) once. That is,

$$G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega) = \frac{k_{1,1}b_{1,1}j\omega G^2_{\infty,\varnothing}(j\omega) + (k_{1,1} + b_{1,1}j\omega)G_{\infty,\varnothing}(j\omega) + 1}{2k_{1,1}b_{1,1}j\omega G_{\infty,\varnothing}(j\omega) + k_{1,1} + b_{1,1}j\omega},$$

where $k_{1,1} = 0.1k$, $b_{1,1} = b$, and $G_{\infty,\varnothing}(j\omega) = 1/\sqrt{kbj\omega}$.

In general, the pseudocode for computing frequency response of infinite networks that satisfy the assumptions (A-1) to (A-6) in Section 2.1 is listed in Algorithm 3, which is the same as Algorithm 1 except for the base case. For finite networks in

**Algorithm 3** Pseudocode of computing infinite networks' frequency response. It computes the frequency response `G` at the angular frequency `w` for an infinite network given its damage case `(l,e)` and the undamaged constants `undCst`.

```
1: function G = freqInf(l,e,undCst,w)
2: s = i*w;
3: if isEmpty(l) then
4:   G = GUnd(undCst,s);
5: else
6:   [l1,e1,lS,eS] = partition(l,e);
7:   for idx from 1 to nS do
8:     GS[idx] = freqInf(lS[idx],eS[idx],undCst,w);
9:   end for
10:  g1Cst = getG1Cst(l1,e1,undCst);
11:  G = Gr(g1Cst,GS,s);
12: end if
```

Algorithm 1, the base case is determined by the criterion if the network has only one generation. Here, for infinite networks in Algorithm 3, the base case is determined by the criterion if the network is undamaged, which is characterized by an empty list of damaged components $\boldsymbol{l}$. The computation of the base case is also replaced by the `GUnd()` function which implements undamaged transfer functions for infinite

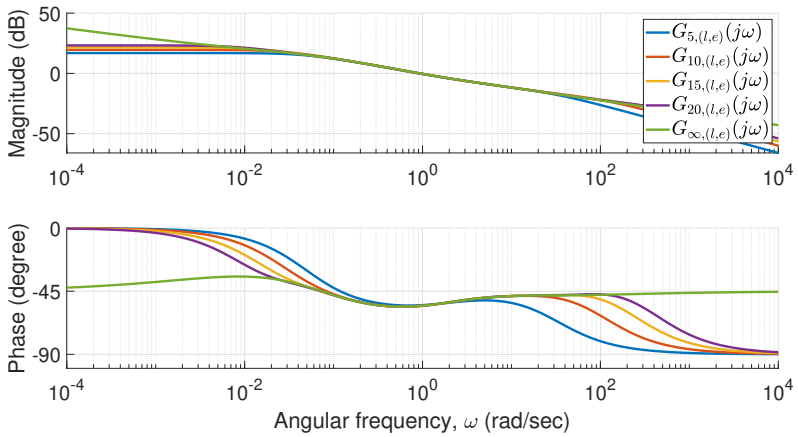


Figure 2.19. Frequency response for mechanical tree networks in Figure 2.1 when the damage case is $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{2,1}, k_{2,2}, b_{3,1}], [0.1, 0.2, 0.3])$. Finite networks' frequency response are computed by Algorithm 1, whereas the infinite network's is computed by Algorithm 3.

networks $G_{\infty,\varnothing}(s)$, such as those in Section 2.4. The resultant frequency response for the three examples networks under some specific damage cases are shown in Figures 2.19 to 2.21, where the convergence of finite networks' frequency response to infinite networks' under the same damage cases is also plotted to verify the correctness of the results.

### 2.5.2 Transfer Functions

Recall that to compute finite networks' transfer functions, the coefficient vectors of one-generation transfer functions $G_1(s)$ are extracted, and recurrence formulas $G_r(s)$ are also revised accordingly. Since the role of $G_1(s)$ is replaced by undamaged transfer functions of infinite networks $G_{\infty,\varnothing}(s)$ here, it seems straightforward to extract coefficient vectors of $G_{\infty,\varnothing}(s)$ as well. However, some adaptations have to be made because $G_{\infty,\varnothing}(s)$'s are no longer rational expressions as shown in Equations (2.16) to (2.18). Instead of extracting coefficients of $s$ directly, in the case of infinite networks, those of some expressions in $s$, denoted by $\phi_i(s)$, are pulled out.

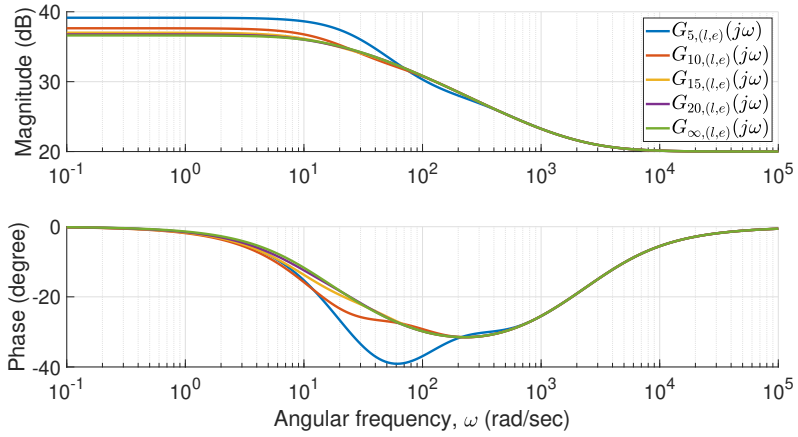


Figure 2.20. Frequency response for electrical ladder networks in Figure 2.2 when the damage case is $(\boldsymbol{l}, \boldsymbol{e}) = ([r_{2,2}], [0.1])$. Finite networks' frequency response are computed by Algorithm 1, whereas the infinite network's is computed by Algorithm 3.

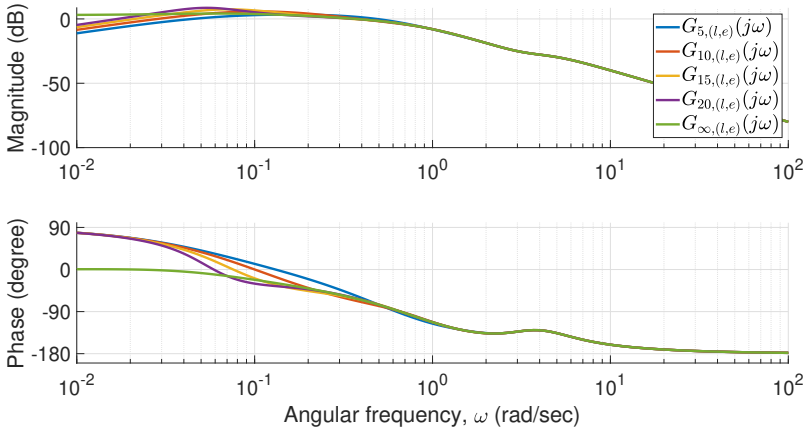


Figure 2.21. Frequency response for mechanical ladder networks in Figure 2.3 when the damage case is $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{p2}, k_{i2}, k_{d2}], [0.1, 0.1, 0.1])$. Finite networks' frequency response are computed by Algorithm 1, whereas the infinite network's is computed by Algorithm 3.

Specifically, for infinite mechanical tree network, the numerator and denominator of its undamaged transfer function

$$G_{\infty,\varnothing}(s) = \frac{1}{\sqrt{kbs}}$$

can be viewed as two polynomials with respect to the variable $\phi_1(s) = \sqrt{s}$. Then, the coefficient vectors of its undamaged transfer functions are

$$\boldsymbol{c_{N\infty,\varnothing}} = \begin{bmatrix} 1 \end{bmatrix},$$
$$\boldsymbol{c_{D\infty,\varnothing}} = \begin{bmatrix} \sqrt{kb} & 0 \end{bmatrix}.$$

Moreover, because its recurrence formula $G_r(s)$ is still a rational expression, the numerator and denominator of damaged transfer functions of infinite mechanical tree networks, obtained by repeatedly applying $G_r(s)$ starting with $G_{\infty,\varnothing}(s)$, are also polynomials with respect to $\phi_1(s) = \sqrt{s}$.

The reason for that is the operations in a rational expression are all basic mathematical operations. None of them could dissolve that $\phi_1(s) = \sqrt{s}$ into a univariate polynomial of $s$ unless all exponents are even. Hence, $\phi_1(s) = \sqrt{s}$ must exist in the numerator and denominator of damaged transfer functions of infinite mechanical tree networks. Luckily, in this case, the variable $s$ is the square of $\phi_1(s) = \sqrt{s}$, so the numerator and denominator can be viewed as univariate polynomials with respect to $\sqrt{s}$. General speaking, as demonstrated in the other two examples later, that relation is not obeyed. As a result, those transfer functions' numerator and denominator have to be regarded as multivariate polynomials with respect to both $s$ and $\phi_i(s)$.

Therefore, in this case, the mechanical tree network's recurrence formula $G_r(s)$

in Equation (2.1) can be revised as

$$\begin{aligned}\boldsymbol{c_{Nr}} = &\begin{bmatrix} k_{1,1}b_{1,1} & 0 & 0 \end{bmatrix} * \boldsymbol{c_{Ns1}} * \boldsymbol{c_{Ns2}} \oplus \begin{bmatrix} k_{1,1} \end{bmatrix} * \boldsymbol{c_{Ns1}} * \boldsymbol{c_{Ds2}} \\ &\oplus \begin{bmatrix} b_{1,1} & 0 & 0 \end{bmatrix} * \boldsymbol{c_{Ns2}} * \boldsymbol{c_{Ds1}} \oplus \boldsymbol{c_{Ds1}} * \boldsymbol{c_{Ds2}}, \qquad (2.19) \\ \boldsymbol{c_{Dr}} = &\begin{bmatrix} k_{1,1}b_{1,1} & 0 & 0 \end{bmatrix} * (\boldsymbol{c_{Ns1}} * \boldsymbol{c_{Ds2}} \oplus \boldsymbol{c_{Ns2}} * \boldsymbol{c_{Ds1}}) \\ &\oplus \begin{bmatrix} b_{1,1} & 0 & k_{1,1} \end{bmatrix} * \boldsymbol{c_{Ds1}} * \boldsymbol{c_{Ds2}}, \qquad (2.20)\end{aligned}$$

which only operate on coefficients and are independent of frequency $\omega$. In Equations (2.19) and (2.20), $\boldsymbol{c_{Ns1}}$, $\boldsymbol{c_{Ds1}}$, $\boldsymbol{c_{Ns2}}$, and $\boldsymbol{c_{Ds2}}$ are the coefficient vectors of both subnetworks with respect to the variable $\phi_1(s) = \sqrt{s}$. In addition, $*$ is the vector convolution operator, and $\oplus$ is a new vector addition operator defined in Section 2.3.2.1. Be careful that Equations (2.19) and (2.20) are different from their counterparts for finite mechanical tree networks in Equations (2.9) and (2.10) because the variable of polynomials for finite mechanical tree networks is $s$, while that for infinite mechanical tree networks is $\phi_1(s) = \sqrt{s}$.

For infinite electrical ladder network, its undamaged transfer function is

$$G_{\infty,\varnothing}(s) = \frac{s + \dfrac{1}{r_2 c} + \sqrt{s^2 + \dfrac{2r_1 + 4r_2}{r_1 r_2 c}s + \dfrac{r_1 + 4r_2}{r_1 r_2^2 c^2}}}{\dfrac{2}{r_1}s + \dfrac{2}{r_1 r_2 c}},$$

which is more complicated compared to the mechanical tree network because it contains an irrational expression of $s$. In this case, the numerator and denominator are regarded as two bivariate polynomials whose two variables are

$$\phi_1(s) = s \quad \text{and} \quad \phi_2(s) = \sqrt{s^2 + \frac{2r_1 + 4r_2}{r_1 r_2 c}s + \frac{r_1 + 4r_2}{r_1 r_2^2 c^2}}.$$

According to the definition of coefficient matrices for bivariate polynomials in Section 2.3.2.1, the coefficient matrices for infinite electrical ladder network's undamaged transfer function are

$$\boldsymbol{c_{N\infty,\varnothing}} = \begin{bmatrix} 1 & 1 \\ 0 & \frac{1}{r_2 c} \end{bmatrix},$$

$$\boldsymbol{c_{D\infty,\varnothing}} = \begin{bmatrix} \frac{2}{r_1} & 0 \\ 0 & \frac{2}{r_1 r_2 c} \end{bmatrix}.$$

Correspondingly, its recurrence formula $G_r(s)$ in Equation (2.2) should be revised to

$$\boldsymbol{c_{Nr}} = \begin{bmatrix} r_{1,1}r_{1,2}c_1 & 0 \\ 0 & r_{1,1}+r_{1,2} \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus r_{1,1}r_{1,2}\boldsymbol{c_{Ds1}}, \tag{2.21}$$

$$\boldsymbol{c_{Dr}} = \begin{bmatrix} r_{1,2}c_1 & 0 \\ 0 & 1 \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus r_{1,2}\boldsymbol{c_{Ds1}}, \tag{2.22}$$

where $*$ is the operator of matrix convolution and $\oplus$ is a new operator of matrix addition defined in Section 2.3.2.1. Again, note that Equations (2.21) and (2.22) are different from their counterparts Equations (2.11) and (2.12) for finite electrical ladder networks because the variables of polynomials are different.

The same situation happens for infinite mechanical ladder networks whose undamaged transfer function is

$$G_{\infty,\varnothing}(s) = \frac{-ms^2 - bs + A(s)}{2[mk_d s^3 + (mk_p + bk_d)s^2 + (mk_i + bk_p)s + bk_i]},$$

where $A(s)$ is defined in Equation (2.18). In this case, the numerator and denominator

of $G_{\infty,\varnothing}(s)$ can also be regarded as two bivariate polynomials whose two variables are

$$\phi_1(s) = s \quad \text{and} \quad \phi_2(s) = A(s).$$

Then, the coefficient matrices of $G_{\infty,\varnothing}(s)$ are

$$\boldsymbol{c_{N\infty,\varnothing}} = \begin{bmatrix} -m & 0 & 0 \\ 0 & -b & 1 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{c_{D\infty,\varnothing}} = \begin{bmatrix} 2mk_d & 0 & 0 & 0 \\ 0 & 2(mk_p + bk_d) & 0 & 0 \\ 0 & 0 & 2(mk_i + bk_p) & 0 \\ 0 & 0 & 0 & 2bk_i \end{bmatrix}.$$

The corresponding recurrence formula $G_r(s)$ in Equation (2.3) is revised to

$$\boldsymbol{c_{Nr}} = \begin{bmatrix} k_{d1} & 0 & 0 \\ 0 & k_{p1} & 0 \\ 0 & 0 & k_{i1} \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} * \boldsymbol{c_{Ds1}},$$

$$\boldsymbol{c_{Dr}} = \begin{bmatrix} m_1 & 0 & 0 \\ 0 & b_1 & 0 \\ 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} k_{d1} & 0 & 0 \\ 0 & k_{p1} & 0 \\ 0 & 0 & k_{i1} \end{bmatrix} * \boldsymbol{c_{Ns1}} \oplus \begin{bmatrix} m_1 & 0 & 0 & 0 \\ 0 & b_1 + k_{d1} & 0 & 0 \\ 0 & 0 & k_{p1} & 0 \\ 0 & 0 & 0 & k_{i1} \end{bmatrix} * \boldsymbol{c_{Ds1}}.$$

Once coefficient vectors and matrices are extracted from $G_{\infty,\varnothing}(s)$ and the recurrence formula $G_r(s)$ is revised to the form of $\boldsymbol{c_{Nr}}$ and $\boldsymbol{c_{Dr}}$ which directly operate on the coefficients, Algorithm 4 is ready for computing transfer functions of those infinite networks fulfilling the assumptions (A-1) to (A-6) in Section 2.1. The structure of Algorithm 4 is the same as that of Algorithm 3 which returns an infinite network's

**Algorithm 4** Pseudocode of computing infinite networks' transfer functions. It computes the coefficient tensors `cN` and `cD` of an infinite network's transfer function given its damage case `(l,e)` and the undamaged constants `undCst`.

```
1: function [cN,cD] = tranInf(l,e,undCst)
2: if isEmpty(l) then
3:   [cN,cD] = CUnd(undCst);
4: else
5:   [l1,e1,lS,eS] = partition(l,e);
6:   for idx from 1 to nS do
7:     [cNS[idx],cDS[idx]] = tranInf(lS[idx],eS[idx],undCst);
8:   end for
9:   g1Cst = getG1Cst(l1,e1,undCst);
10:  [cN,cD] = Cr(g1Cst,cNS,cDS);
11:  [cN,cD] = simplify(cN,cD);
12: end if
```

frequency response. The `GUnd()` function in Algorithm 3 is replaced by the `CUnd()` function in Algorithm 4, which outputs the coefficient tensors for an infinite network's undamaged transfer function $\boldsymbol{c}_{\boldsymbol{N}\infty,\varnothing}$ and $\boldsymbol{c}_{\boldsymbol{D}\infty,\varnothing}$. The `Gr()` function in Algorithm 3 is replaced by the `Cr()` function in Algorithm 4, which implements the computations of revised recurrence formulas $\boldsymbol{c}_{\boldsymbol{N}\boldsymbol{r}}$ and $\boldsymbol{c}_{\boldsymbol{D}\boldsymbol{r}}$. The `simplify()` function in Algorithm 4 is similar to that in Algorithm 2.

It is worth noting that the coefficient tensor is not unique for a fractional or irrational transfer function because transfer functions are always univariate with the only variable $s$. They are manufactured to multivariate polynomials with respect to variables $\phi_i(s)$ to enable that computations can take place only on the coefficients of those fractional or irrational transfer functions. Take the following irrational function $T(s)$ as an example, where

$$T(s) = s + 1 + \sqrt{s+1} = \left(\sqrt{s+1}\right)^2 + \sqrt{s+1}.$$

$T(s)$ can be regarded as a bivariate polynomial with the two variables $\phi_1(s) = s$ and

$\phi_2(s) = \sqrt{s+1}$. Then, both of the following two coefficient matrices represent $T(s)$.

$$\boldsymbol{c_1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{c_2} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

Finally, the transfer functions obtained by Algorithm 4 for the three example networks in some specific cases are showcased here. For an infinite mechanical tree network in Figure 2.1 with the damage case $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{2,1}, b_{2,1}], [0.1, 0.2])$, its transfer function is

$$G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s) = \frac{0.71s^2 + 2.20s^{\frac{3}{2}} + 9.62s + 12.20s^{\frac{1}{2}} + 1.41}{s^{\frac{5}{2}} + 3.11s^2 + 13.60s^{\frac{3}{2}} + 3.40s + 2.00s^{\frac{1}{2}}}. \tag{2.23}$$

For an infinite electrical ladder network in Figure 2.2 with the damage case $(\boldsymbol{l}, \boldsymbol{e}) = ([r_{2,2}, r_{3,2}], [0.1, 0.1])$, its transfer function is

$$G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s) = \frac{N_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s)}{D_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s)}, \tag{2.24}$$

where

$$\begin{aligned} N_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s) &= s^4 + s^3\phi_2(s) + 7.2 \times 10^3 s^3 + 5.2 \times 10^3 s^2\phi_2(s) + 1.5 \times 10^7 s^2 \\ &\quad + 6.7 \times 10^6 s\phi_2(s) + 8.1 \times 10^9 s + 1.5 \times 10^9 \phi_2(s) + 8.0 \times 10^{10}, \\ D_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s) &= 0.1s^4 + 0.1s^3\phi_2(s) + 622s^3 + 421s^2\phi_2(s) + 9.8 \times 10^5 s^2 \\ &\quad + 3.5 \times 10^5 s\phi_2(s) + 2.6 \times 10^8 s + 2.2 \times 10^7 \phi_2(s) + 2.5 \times 10^9, \\ \phi_2(s) &= \sqrt{s^2 + 4020s + 40100}. \end{aligned}$$

For an infinite mechanical ladder network in Figure 2.3 with the damage case

$$(\boldsymbol{l}, \boldsymbol{e}) = ([k_{p2}, k_{i2}, k_{d2}], [0.1, 0.1, 0.1]),$$

the coefficient matrices of its transfer function are

$$c_{N\infty,(\boldsymbol{l},\boldsymbol{e})} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 9.21 & 0.05 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 35.6 & 0.42 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 84.0 & 1.33 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 62.1 & 2.70 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5.66 & 0.26 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.14 & 0.01 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$c_{D\infty,(\boldsymbol{l},\boldsymbol{e})} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 12.2 & 0.05 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 69.2 & 0.58 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 219 & 2.91 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 311 & 7.76 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 217 & 5.91 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 74.8 & 0.54 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8.63 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.40 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix}, \tag{2.25}$$

with two variables

$$\phi_1(s) = s \quad \text{and} \quad \phi_2(s) = \sqrt{s^4 + 10s^3 + 49s^2 + 42s + 2}.$$

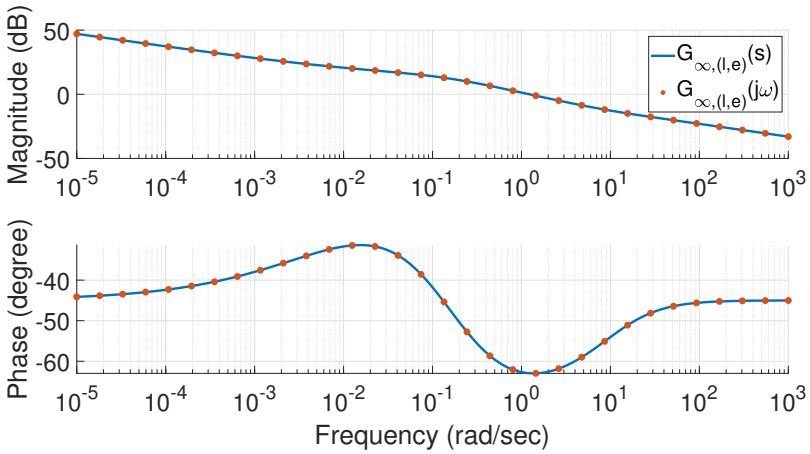


Figure 2.22. The consistency between an infinite mechanical tree network's transfer function in Equation (2.23) and its corresponding frequency response from Algorithm 3.

### 2.5.3 Correctness Check

The correctness of the frequency response and transfer functions returned by Algorithms 3 and 4 for the three example networks when they are infinitely large is demonstrated here. The verification is threefold. First, for the same damage case, a finite network's frequency response should converge to its corresponding infinite network's. That has already been proved previously in Figures 2.19 to 2.21. Second, for the same infinite network, its frequency response from Algorithm 3 should be consistent with its transfer function from Algorithm 4. That consistency is proved by Figures 2.22 to 2.24.

Third, the time-domain response of an infinite network's transfer function returned by Algorithm 4 should be consistent with the numerical integration of the corresponding differential equations describing that network's dynamics. Due to the lack of simulating time-domain response for irrational transfer functions, that consistency is verified only for an infinite mechanical tree network with the transfer function in Equation (2.23), which is fractional. The time-domain response of that transfer

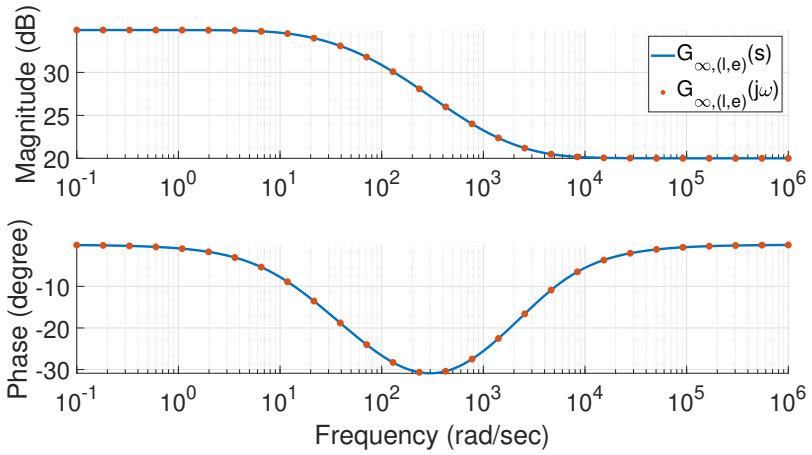


Figure 2.23. The consistency between an infinite electrical ladder network's transfer function in Equation (2.24) and its corresponding frequency response from Algorithm 3.

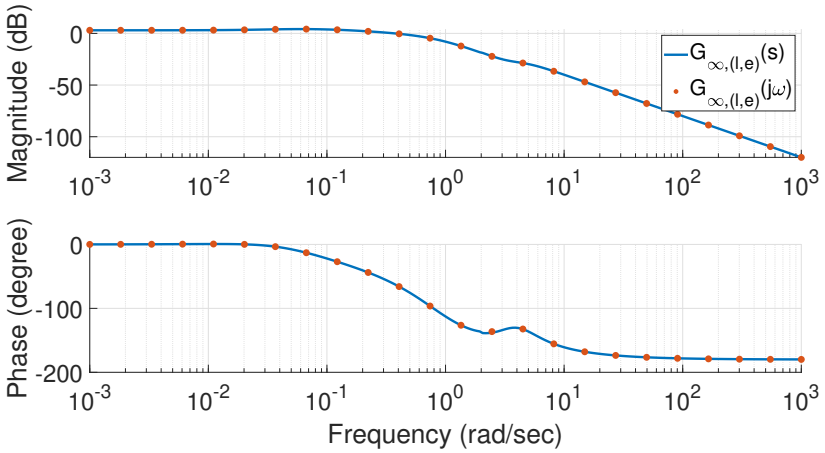


Figure 2.24. The consistency between an infinite mechanical ladder network's transfer function in Equation (2.25) and its corresponding frequency response from Algorithm 3.

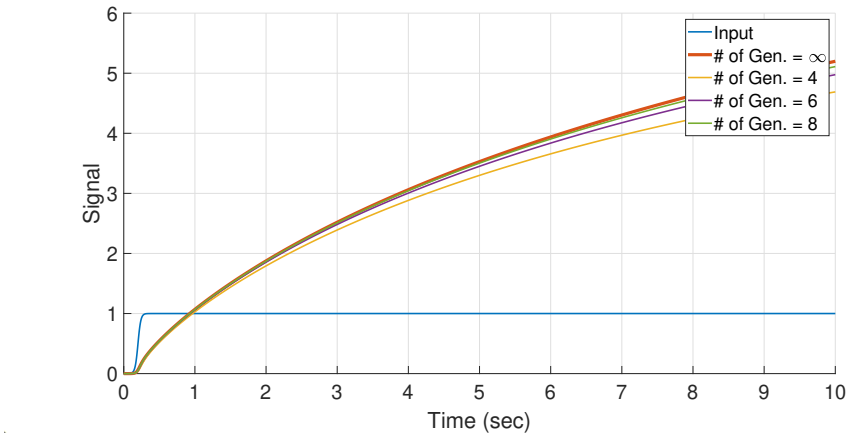


Figure 2.25. The blue curve is the input signal $f(t) = 1/(1 + e^{-50(t-0.2)})$. The thick red curve is the time-domain response of an infinite mechanical tree network's transfer function in Equation (2.23) given by the `lsim()` function in the TOFT toolbox [25]. The other three time-domain responses are obtained by using `ode45()` to integrate the differential equations that describe a four, six, and eight-generation mechanical tree network's dynamics.

function is obtained by using the `lsim()` function provided by the TOFT toolbox [25]. The input signal is $f(t) = 1/(1 + e^{-50(t-0.2)})$. On the other hand, the numerical integration of differential equations cannot be performed on an infinite mechanical tree network. Therefore, that numerical integration through `ode45()` is carried out on three finite mechanical tree networks, with four, six and eight generations respectively. The differential equations describing those three finite mechanical tree networks are similar to those in Section 2.3.3. The convergence of those three finite networks' time-domain responses to the infinite network's is shown in Figure 2.25, which indirectly verifies that consistency.

## 2.6 Concluding Remarks

### 2.6.1 Computation Time

Because all four algorithms are recursive, their running time depends on how soon the base case is reached. Therefore, for finite networks, the running time of Algorithms 1 and 2 relies on those networks' size because the base case is their one-generation transfer functions. In contrast, for infinite networks, the running time of Algorithms 3 and 4 depends on the deepest generation where damage resides because the base case is their undamaged transfer functions. As a result, if the frequency response of a large but finite network with shallow damages is needed, approximating that with its infinite variant is possibly a good idea to save the computation time.

For example, the frequency response of a 20-generation mechanical tree network with a shallow damage case at the first generation, $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{1,1}, b_{1,1}], [0.1, 0.2])$, is required. The computation time of using Algorithm 1 to exactly evaluate that frequency response takes 7.2 seconds. In contrast, using Algorithm 3 to approximate that only entails 0.005 seconds. Both computations are implemented in `MATLAB R2020b` on `Intel i7-10510u`. The comparison between the exact frequency response and the approximated one is shown in Figure 2.26, from which we can confirm that the approximation is accurate within the frequency range from 0.2 rad/sec to 100 rad/sec.

### 2.6.2 Fractional or Irrational Nature of Infinite Networks' Dynamics

For an infinite self-similar dynamic network, it is very likely that its transfer function is of non-integer order as illustrated by the three examples in this chapter. Some of those transfer functions are fractional, like an infinite mechanical tree network. Others are irrational, like an infinite electrical or mechanical ladder network. However, finite self-similar networks' dynamics are always of integer order as long as they

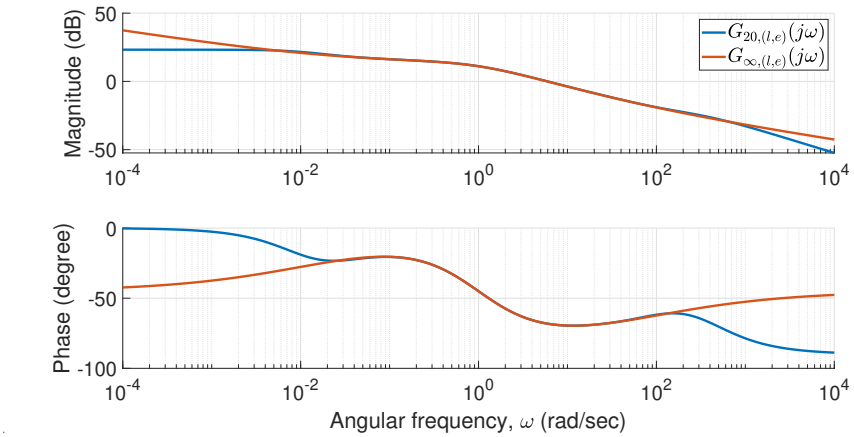


Figure 2.26. The blue curve is the exact frequency response of a 20-generation mechanical tree network with the damage case $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{1,1}, b_{1,1}], [0.1, 0.2])$ obtained by Algorithm 1. The red curve is the approximated one using its infinite variant through Algorithm 3.

satisfy the assumptions (A-1) to (A-4) in Section 2.1. That can be understood as a convergent behavior that while the size of a finite network grows, its transfer function's order is also increasing. Eventually, when its size grows to infinity, its high-order transfer function converges to a non-integer-order one. As a concrete example, consider a mechanical tree network with the damage case $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{1,1}, b_{1,1}], [0.1, 0.2])$. Its transfer function as the number of generations grows is listed below. The finite transfer functions are given by Algorithm 2, and the infinite one is given by Algo-

rithm 4.

$$
\begin{aligned}
G_{1,(\boldsymbol{l},\boldsymbol{e})} &= \frac{1}{0.2s+0.2},\\
G_{2,(\boldsymbol{l},\boldsymbol{e})} &= \frac{6s^2+23.2s+22}{s^3+5.4s^2+8.8s+4},\\
G_{3,(\boldsymbol{l},\boldsymbol{e})} &= \frac{7s^6+132.8s^5+886s^4+2550.4s^3+3368s^2+1916.8s+384}{s^7+21.8s^6+172s^5+615.2s^4+1094.4s^3+976s^2+409.6s+64},\\
&\vdots\\
G_{\infty,(\boldsymbol{l},\boldsymbol{e})} &= \frac{0.7071s+5.1s^{\frac{1}{2}}+0.7071}{s^{\frac{3}{2}}+0.2828s+s^{\frac{1}{2}}}.
\end{aligned}
$$

That convergent behavior can be leveraged to find rational approximations of fractional and some irrational functions as discussed in Chapter 6.

### 2.6.3 Effects of Varying a Network's Status on Its Dynamics

One crucial discovery after writing down a network's transfer function is that the effect of varying its status on its dynamics can be isolated. That isolation is described as a multiplicative disturbance here. Specifically, a self-similar network's transfer function is always a ratio between two functions of $s$. Therefore, if at two different statuses $a$ and $b$, a network's respective transfer functions are

$$G_a(s) = \frac{N_a(s)}{D_a(s)} \quad \text{and} \quad G_b(s) = \frac{N_b(s)}{D_b(s)},$$

then the effect of changing that network's status from $a$ to $b$ on its transfer function can be expressed in terms of a multiplicative disturbance $\Delta(s)$ where

$$\Delta(s) = \frac{G_b(s)}{G_a(s)} = \frac{N_b(s)D_a(s)}{N_a(s)D_b(s)}.$$

There exist at least two meaningful perspectives that can be explored in the context of large networks. The first one is quantifying the approximation error of

estimating a finite network's transfer function using the corresponding infinite one's. As an example, for an undamaged mechanical tree network, its transfer functions when it is infinitely large and when it has three generations are

$$G_{\infty,\varnothing}(s) = \frac{1}{1.4142\sqrt{s}},$$
$$G_{3,\varnothing}(s) = \frac{3s^6 + 62s^5 + 428s^4 + 1272s^3 + 1712s^2 + 992s + 192}{s^7 + 30s^6 + 300s^5 + 1288s^4 + 2576s^3 + 2400s^2 + 960s + 128}.$$

Then, the error of approximating the three-generation network using the infinite network is

$$\begin{aligned}\Delta(s) &= \frac{G_{3,\varnothing}(s)}{G_{\infty,\varnothing}(s)} \\ &= \frac{4.243s^{\frac{13}{2}} + 87.68s^{\frac{11}{2}} + 605.3s^{\frac{9}{2}} + 1799s^{\frac{7}{2}} + 2421s^{\frac{5}{2}} + 1403s^{\frac{3}{2}} + 271.5s^{\frac{1}{2}}}{s^7 + 30s^6 + 300s^5 + 1288s^4 + 2576s^3 + 2400s^2 + 960s + 128}.\end{aligned}$$

The second perspective is to study the effect brought by a network's damages on its dynamics. For an infinite mechanical tree network whose damage case is $(\boldsymbol{l}, \boldsymbol{e}) = ([k_{2,1}, b_{2,1}], [0.1, 0.2])$, its transfer function is

$$G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s) = \frac{0.7071s^2 + 2.2s^{\frac{3}{2}} + 9.6167s + 12.2s^{\frac{1}{2}} + 1.4142}{s^{\frac{5}{2}} + 3.1113s^2 + 13.6s^{\frac{3}{2}} + 3.3941s + 2s^{\frac{1}{2}}}.$$

As a result, the effect brought by that damage case is

$$\Delta(s) = \frac{G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s)}{G_{\infty,\varnothing}(s)} = \frac{s^2 + 3.1113s^{\frac{3}{2}} + 13.6s + 17.2534s^{\frac{1}{2}} + 2}{s^2 + 3.1113s^{\frac{3}{2}} + 13.6s + 3.3941s^{\frac{1}{2}} + 2}.$$

Therefore, while a network's status is varying within some extents, its frequency response is very likely to also change within some range. That variation can form a set of neighboring plants to which a unified controller can be obtained through robust control methods so that stability and performance are guaranteed for all plants in that set.

In the following Chapters 3 to 6, we can see how the above knowledge concerning frequency response and transfer functions of self-similar networks is applied to meaningful purposes.

# CHAPTER 3

# SIMULATING UNEVENLY DISTRIBUTED TRANSMISSION LINES WITH APPLICATION TO RAILWAY SAFETY MONITORING

In this chapter, an application example of the modeling methods developed in Chapter 2 to simulating dynamic networks is presented. Specifically, those methods are leveraged to efficiently approximate voltage and current along a transmission line whose electrical properties are unevenly distributed, that is, in the language of modeling methods in Chapter 2, that transmission line is damaged. Transmission line models are widely used in the studies concerning railway track circuits which detect if a certain sector of track is occupied [22, 141, 169]. Thanks to those modeling methods' capabilities of computing damaged frequency response, they can simulate how voltage and current would vary along a track circuit when degradations occur and while a train is passing through that track. The contents in this chapter appear in [113].

## 3.1 Transmission Line Theory

In this section, transmission line theory, credited to Oliver Heaviside, is briefly reviewed, which determines the voltage and current along a wire with respect to both spatial and temporal parameters. The transmission line model is shown in Figure 3.1 whose four electrical properties are listed in Table 3.1. The goal of the transmission line theory is to evaluate spatial and temporal distribution of voltage and current, *i.e.*, $v(x,t)$ and $i(x,t)$, given the boundary conditions at $x = 0$ and the constant values of the electrical properties listed in Table 3.1.

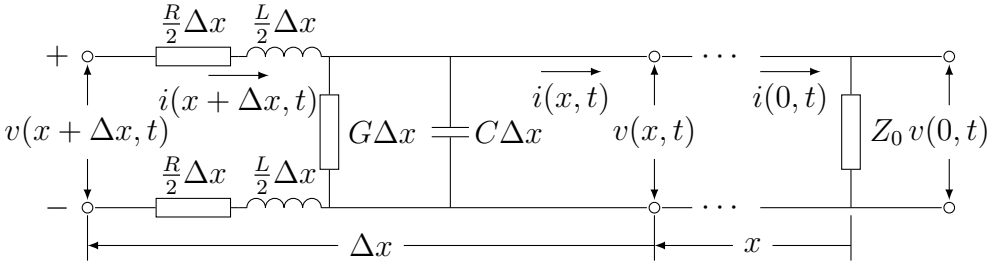


Figure 3.1. Model for transmission line theory. The left hand side is the input/transmitter end. The right hand side is the output/receiver end, where $x = 0$.

TABLE 3.1

ELECTRICAL PROPERTIES USED IN THE TRANSMISSION LINE MODEL. ALL CONSTANT VALUES OF THOSE PROPERTIES ARE POSITIVE REAL NUMBERS.

| | | |
|---|---|---|
| $R$ | Series resistance | $\Omega/m$ |
| $L$ | Series inductance | $H/m$ |
| $G$ | Shunt conductance | $S/m$ |
| $C$ | Shunt capacitance | $F/m$ |

For an evenly distributed transmission line, the following two equations within an infinitesimal distance $\Delta x$ are given by Kirchhoff's circuit laws.

$$v(x+\Delta x,t) - R\Delta x i(x+\Delta x,t) - L\Delta x \frac{\partial i(x+\Delta x,t)}{\partial t} - v(x,t) = 0,$$

and

$$i(x+\Delta x,t) - G\Delta x v(x,t) - C\Delta x \frac{\partial v(x,t)}{\partial t} - i(x,t) = 0.$$

Dividing by $\Delta x$ gives

$$\frac{v(x+\Delta x,t) - v(x,t)}{\Delta x} - Ri(x+\Delta x,t) - L\frac{\partial i(x+\Delta x,t)}{\partial t} = 0,$$

and

$$\frac{i(x+\Delta x,t) - i(x,t)}{\Delta x} - Gv(x,t) - C\frac{\partial v(x,t)}{\partial t} = 0.$$

Taking the limit $\Delta x \to 0$ results in the following system of two partial differential equations which are frequently called the telegrapher's equations in literature.

$$\frac{\partial v(x,t)}{\partial x} = Ri(x,t) + L\frac{\partial i(x,t)}{\partial t}, \tag{3.1}$$

$$\frac{\partial i(x,t)}{\partial x} = Gv(x,t) + C\frac{\partial v(x,t)}{\partial t}. \tag{3.2}$$

The solutions to telegrapher's equations can be found by using separation of variables and assuming the following forms, where

$$v(x,t) = \mathrm{Re}\{v(x)e^{j(\omega t+\phi)}\}, \tag{3.3}$$

$$i(x,t) = \mathrm{Re}\{i(x)e^{j(\omega t+\phi)}\}. \tag{3.4}$$

It is worth emphasizing that $v(x)$ and $i(x)$ are complex numbers instead of real numbers, so $v(x)e^{j(\omega t+\phi)}$ and $i(x)e^{j(\omega t+\phi)}$ should not be confused with phasor representa-

tions. Before going to the next step, the following theorem needs to be established.

**Theorem 1.** *For all* $\alpha(x) \in \mathbb{C}$,

$$\frac{\partial}{\partial x} Re\{\alpha(x)e^{j(\omega t+\phi)}\} = Re\left\{\frac{d}{dx}\alpha(x)e^{j(\omega t+\phi)}\right\},$$
$$\frac{\partial}{\partial t} Re\{\alpha(x)e^{j(\omega t+\phi)}\} = Re\{j\omega\alpha(x)e^{j(\omega t+\phi)}\}.$$

*Proof.* Assume that $\alpha(x) = A(x) + jB(x)$, where $A(x),\ B(x) \in \mathbb{R}$. Then, we have the following results:

$$\begin{aligned}
\mathrm{Re}\{\alpha(x)e^{j(\omega t+\phi)}\} &= A(x)\cos(\omega t+\phi) - B(x)\sin(\omega t+\phi),\\
\frac{\partial}{\partial x}\mathrm{Re}\{\alpha(x)e^{j(\omega t+\phi)}\} &= \frac{dA(x)}{dx}\cos(\omega t+\phi) - \frac{dB(x)}{dx}\sin(\omega t+\phi),\\
\frac{\partial}{\partial t}\mathrm{Re}\{\alpha(x)e^{j(\omega t+\phi)}\} &= -\omega A(x)\sin(\omega t+\phi) - \omega B(x)\cos(\omega t+\phi).
\end{aligned}$$

On the other side of the equations, we have

$$\begin{aligned}
\frac{d}{dx}\alpha(x)e^{j(\omega t+\phi)} &= \left(\frac{dA(x)}{dx} + j\frac{dB(x)}{dx}\right)(\cos(\omega t+\phi) + j\sin(\omega t+\phi)),\\
j\omega\alpha(x)e^{j(\omega t+\phi)} &= (-\omega B(x) + j\omega A(x))(\cos(\omega t+\phi) + j\sin(\omega t+\phi)).
\end{aligned}$$

Therefore, both equations are satisfied. □

Due to Theorem 1, the telegrapher's equations in Equations (3.1) and (3.2) under the assumed solutions' form in Equations (3.3) and (3.4) become

$$\mathrm{Re}\left\{\frac{d}{dx}v(x)e^{j(\omega t+\phi)}\right\} = \mathrm{Re}\{(R+j\omega L)i(x)e^{j(\omega t+\phi)}\},$$
$$\mathrm{Re}\left\{\frac{d}{dx}i(x)e^{j(\omega t+\phi)}\right\} = \mathrm{Re}\{(G+j\omega C)v(x)e^{j(\omega t+\phi)}\}.$$

The operator of taking real parts, $\mathrm{Re}\{\cdot\}$, can be discarded from the above equations, which leads to a sufficient condition of the original telegrapher's equations under the

assumed solutions' form in Equations (3.3) and (3.4). Therefore, any solutions to the ordinary differential equations without the operators $\mathrm{Re}\{\cdot\}$ would also satisfy the original telegrapher equations. Hence, now, the goal is to solve the following two ordinary differential equations:

$$\frac{d}{dx}v(x) = (R + j\omega L)i(x),$$
$$\frac{d}{dx}i(x) = (G + j\omega C)v(x),$$

which can be decoupled as

$$\frac{d^2}{dx^2}v(x) = \gamma^2 v(x) \quad \text{and} \quad \frac{d^2}{dx^2}i(x) = \gamma^2 i(x),$$

where $\gamma = \sqrt{(R + j\omega L)(G + j\omega C)}$. Using the boundary conditions at $x = 0$,

$$v(0,t) = \mathrm{Re}\{v(0)e^{j(\omega t+\phi)}\} \quad \text{and} \quad i(0,t) = \mathrm{Re}\{i(0)e^{j(\omega t+\phi)}\}$$

the final results are

$$v(x,t) = \mathrm{Re}\{v(x)e^{j(\omega t+\phi)}\}, \tag{3.5}$$
$$i(x,t) = \mathrm{Re}\{i(x)e^{j(\omega t+\phi)}\}, \tag{3.6}$$

where

$$
\begin{aligned}
v(x) &= \frac{v(0)}{1+\mu}(e^{\gamma x} + \mu e^{-\gamma x}), \\
i(x) &= \frac{i(0)}{1-\mu}(e^{\gamma x} - \mu e^{-\gamma x}), \\
\mu &= \frac{Z_0 - Z_c}{Z_0 + Z_c}, \\
Z_c &= \sqrt{\frac{R + j\omega L}{G + j\omega C}}.
\end{aligned}
$$

Note that for an unevenly distributed transmission line, those electrical properties in Table 3.1 may vary with the distance $x$ and time $t$, which makes such simple solutions difficult or impossible to obtain.

## 3.2 Circuit Network Model

This section uses a modeling algorithm from Chapter 2 to approximate voltage and current along a transmission line. Most importantly, that approximation is not restricted to the case where all electrical properties are constant. The method starts with a circuit network model in Figure 3.2 which divides a long transmission line in Figure 3.1 into $n$ subsections with equal length where electrical properties are lumped into each subsection. The goal is to compute the voltage and current at every node connecting two adjacent subsections, *i.e.*, $v_g$ and $i_g$ in Figure 3.2. Those would be discrete approximations of the continuous results, $v(x,t)$ and $i(x,t)$ in Equations (3.5) and (3.6), given by transmission line theory. The undamaged case is defined as $\forall i$, $r_{i,1} = r_{i,2} = r$, $l_{i,1} = l_{i,2} = l$, $rb_i = rb$, and $c_i = c$. Again, a damage case is denoted by a pair of two lists $(\boldsymbol{l}, \boldsymbol{e})$.

The algorithm computing frequency response of finite networks in Section 2.3.1 is revised for the specific application in this chapter, as listed in Algorithm 5. The first returned value $\mathtt{Z}$ is the impedance of the entire finite network, *i.e.*, $\mathtt{Z} = V_{\text{in}}/I_{\text{in}}$. The

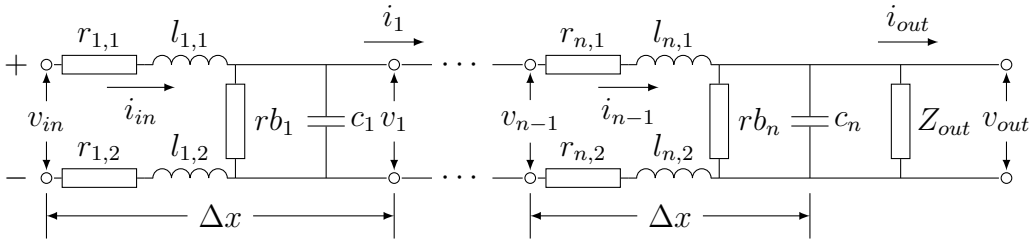


Figure 3.2. Circuit network model with $n$ subsections to approximate a transmission line in Figure 3.1.

**Algorithm 5** Computing the impedance `Z` and voltage gain `H` at the angular frequency `w` for a circuit network in Figure 3.2 with `nG` number of generations given its damage case `(l,e)`, the undamaged constants `undCst`, and the impedance at the output end `zOut`.

```
1: function[Z,H]=freqResp(l,e,undCst,zOut,w,nG)
2: s = j*w;
3: [l1,e1,lS,eS] = partition(l,e);
4: g1Cst = getG1Cst(l1,e1,undCst);
5: if nG == 0 then
6:   [Z,H] = G0(zOut,s);
7: else
8:   nG = nG-1;
9:   [ZS,HS]=freqResp(lS,eS,undCst,zOut,w,nG);
10:  Z = Zr(g1Cst,ZS,s);
11:  H = Hr(g1Cst,Z,HS,s);
12:  save(Z,H);
13: end if
```

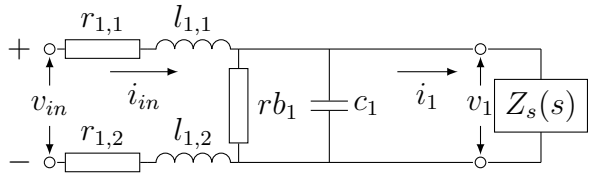


Figure 3.3. The illustration for deriving recurrence formulas $Z_r(s)$ and $H_r(s)$ used in the `Zr()` and `Hr()` functions. The impedance $Z_s(s) = V_1(s)/I_1(s)$ and the voltage gain $H_s(s) = V_{\text{out}}(s)/V_1(s)$ of the subnetwork after the first generation are assumed to be known ($H_s(s)$ is not shown in this figure).

other returned value `H` is the voltage gain of the entire network, *i.e.*, $\texttt{H} = V_{\text{out}}/V_{\text{in}}$. Note that the function `save(Z,H)` externally stores intermediate impedances $Z_g = V_g/I_g$ and voltage gains $H_g = V_{\text{out}}/V_g$ at every node $g$. Then, because the states at the output end in Figure 3.2 are assumed to be known, which are equivalent to the boundary conditions at $x = 0$ for the transmission line model in Figure 3.1, the intermediate voltages $V_g$ and currents $I_g$ at every node $g$, including $V_{\text{in}}$ and $I_{\text{in}}$, can be obtained after just one run of Algorithm 5.

The base case in Algorithm 5 is when the network in Figure 3.2 has no generations, in which case the returned impedance $\texttt{Z} = V_{\text{in}}/I_{\text{in}} = V_{\text{out}}/I_{\text{out}} = \texttt{zOut}$, and the returned voltage gain $\texttt{H} = V_{\text{out}}/V_{\text{in}} = V_{\text{out}}/V_{\text{out}} = 1$. The recurrence formulas used in the `Zr()` and `Hr()` functions are derived as follows. The derivation uses the illustration in Figure 3.3. The recurrence formula for impedance $Z_r(s)$ is obtained directly per the rules of computing equivalent impedance for electrical circuits, where

$$Z_r(s) = r_{1,1} + r_{1,2} + l_{1,1}s + l_{1,2}s + \frac{1}{\dfrac{1}{rb_1} + c_1 s + \dfrac{1}{Z_s(s)}}.$$

The recurrence formula for voltage gain $H_r(s)$ is acquired by the rule that in series circuits, the voltage across a part of that circuit is proportional to the impedance of

that part. Therefore,

$$\begin{aligned}\frac{V_1(s)}{V_{\text{in}}(s)} &= \cfrac{\cfrac{1}{\cfrac{1}{rb_1}+c_1 s+\cfrac{1}{Z_s(s)}}}{r_{1,1}+r_{1,2}+l_{1,1}s+l_{1,2}s+\cfrac{1}{\cfrac{1}{rb_1}+c_1 s+\cfrac{1}{Z_s(s)}}} \\ &= 1-\frac{r_{1,1}+r_{1,2}+l_{1,1}s+l_{1,2}s}{Z_r(s)}\end{aligned}$$

As a result, the recurrence formula for voltage gain is

$$H_r(s)=\frac{V_{\text{out}}(s)}{V_{\text{in}}(s)}=H_s(s)\left(1-\frac{r_{1,1}+r_{1,2}+l_{1,1}s+l_{1,2}s}{Z_r(s)}\right).$$

### 3.2.1 Results Verification

This section verifies the correctness of the results from Algorithm 5 by comparing the approximated voltage and current along an evenly distributed transmission line to the actual ones given by transmission line theory in Equations (3.5) and (3.6). The boundary conditions at the receiver end are assumed to be

$$\begin{aligned}v(0,t) &= \text{Re}\{V_{\text{out}}\}=\text{Re}\{110e^{j4600\pi t}\}V, \\ Z_0 &= Z_{\text{out}}=500\Omega, \\ i(0,t) &= \text{Re}\{I_{\text{out}}\}=\text{Re}\{0.22e^{j4600\pi t}\}A,\end{aligned} \tag{3.7}$$

which are taken from [162]. The length of the entire transmission line is set to be $1170m$, which is from [169]. The constants for electrical poperties in Table 3.1 are

from [63] at the frequency $2300Hz$, where

$$R = 2.5m\Omega/m,$$
$$L = 1.8\mu H/m,$$
$$G = 20\mu S/m,$$
$$C = 0.2nF/m.$$

By knowing the above quantities, the actual voltage $v(x,t)$ and current $i(x,t)$ given by transmission line theory can be computed according to Equations (3.5) and (3.6).

For the circuit network model in Figure 3.2 with $n$ subsections, the distance of each one is $\Delta x = 1170/n$ meters. Therefore, the undamaged constants are

$$\begin{aligned} r &= 2.5\Delta x/2 \ m\Omega, \\ l &= 1.8\Delta x/2 \ \mu H, \\ rb &= 1/(20 \times 10^{-6}\Delta x) \ \Omega, \\ c &= 0.2\Delta x \ nF, \end{aligned} \tag{3.8}$$

which are grouped into the `undCst` to call Algorithm 5. In addition, both `l` and `e` are empty lists indicating the intact case, `zOut` $= 500$, `w` $= 4600\pi$, and `nG` $= n$. Then, Algorithm 5 saves the impedance $Z_g = V_g/I_g$ and voltage gain $H_g = V_{\text{out}}/V_g$ at each node $g$ between any two adjacent subsections, which are shown in Figures 3.4 and 3.5 for the undamaged network with fifty generations. After that, given the same boundary condition $v(0,t)$ in Equation (3.7), *i.e.*, $V_{\text{out}}$ here, the resultant $V_g \in \mathbb{C}$ and $I_g \in \mathbb{C}$ at those nodes $g$ can be derived. The real parts of those complex numbers become the approximated voltage and current along the same transmission line given by the circuit network model.

The comparison between the transmission line theory's result and the circuit net-

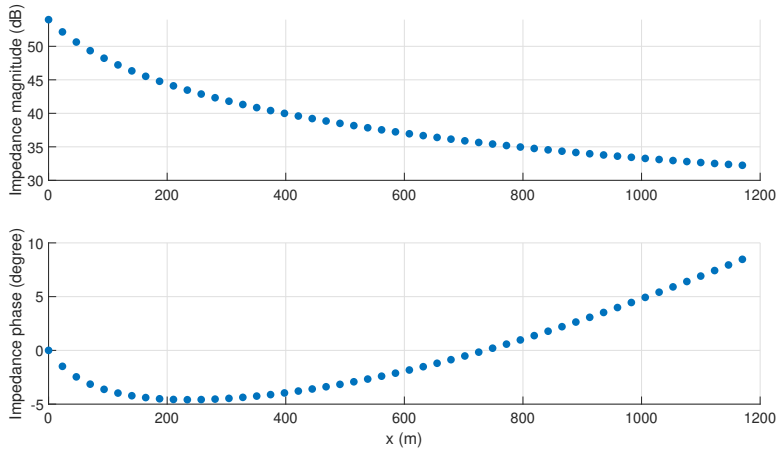


Figure 3.4. Impedance $Z_g = V_g/I_g$ (when $\omega = 4600\pi$ rad/sec) at each node $g$ connecting two adjacent subsections in the undamaged 50-generation circuit network model obtained by Algorithm 5.

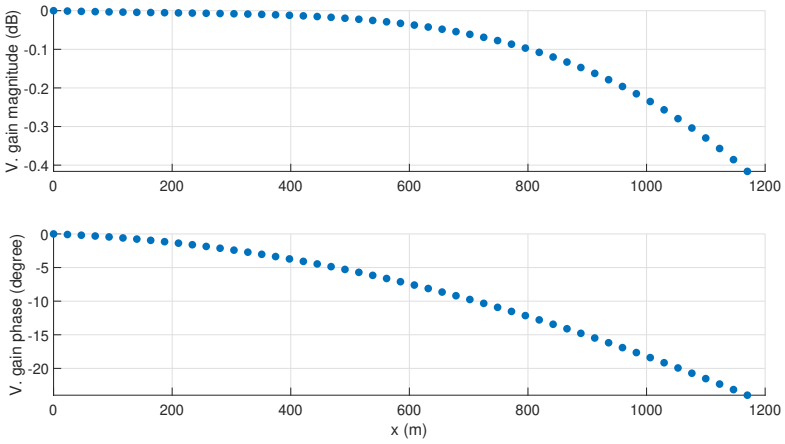


Figure 3.5. Voltage gain $H_g = V_{\text{out}}/V_g$ (when $\omega = 4600\pi$ rad/sec) at each node $g$ connecting two adjacent subsections in the undamaged 50-generation circuit network model obtained by Algorithm 5.

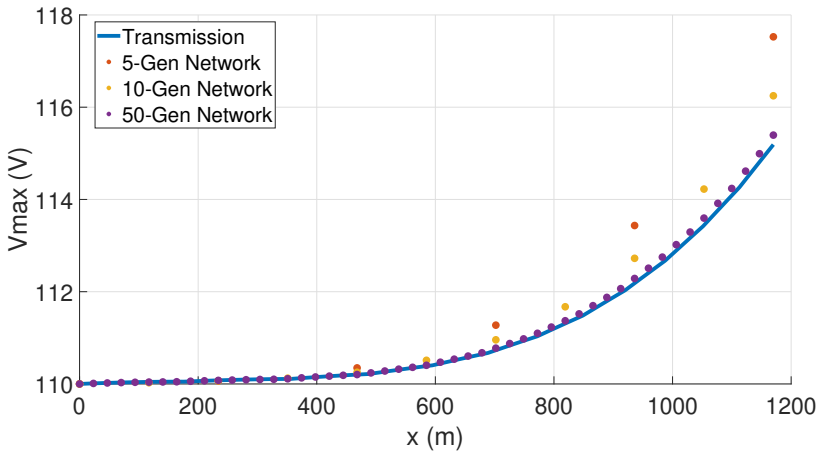


Figure 3.6. Maximum voltage $\max_t(|v(x,t)|)$ *versus* the distance $x$. The blue curve is given by transmission line theory. The dots are from the circuit network model given by Algorithm 5.

work's result are shown in Figures 3.6 and 3.7, where $\max_t(|v(x,t)|)$ and $\max_t(|i(x,t)|)$ are plotted *versus* the distance $x$. For the circuit network model in Figure 3.2, three networks with five, ten, and fifty generations are tested. From Figure 3.6, we see that the $V_{\max}$ given by the circuit network model converges to the one obtained by using transmission line theory as the number of generations increases. From Figure 3.7, we observe that the $I_{\max}$ given by both methods almost overlap each other. In addition, to prove that both magnitudes and phases are correct, Figure 3.8 compares two $v(1170,t)$ at the transmitter end evaluated by both models. In conclusion, we can confirm that the circuit network model as well as Algorithm 5 provide a reasonable approximation of a transmission line model when the electrical properties are evenly distributed.

## 3.3 Application to Railway Track Circuits

From previous discussions, it is understood that traditional transmission line theory could hardly handle the situation where the electrical properties are unevenly

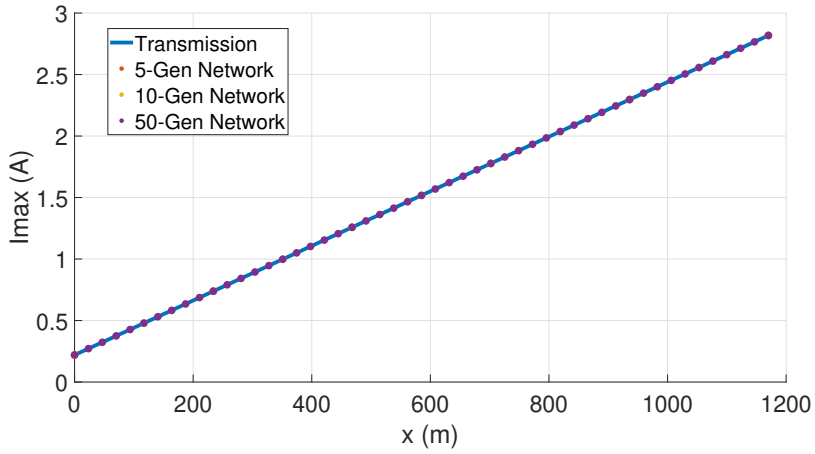


Figure 3.7. Maximum current $\max_t(|i(x,t)|)$ *versus* the distance $x$. The blue curve is given by transmission line theory. The dots are from the circuit network model given by Algorithm 5.

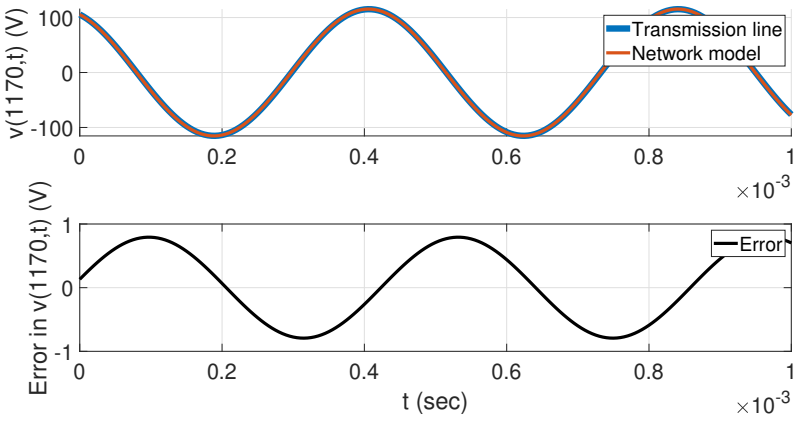


Figure 3.8. The upper figure plots $v(1170,t)$ evaluated by two different methods, where the blue curve is from transmission line theory, while the red curve is from the circuit network model with fifty generations. Note that the two curves overlap each other. The lower figure shows the error between those two curves in the upper figure.

distributed. This section showcases that the circuit network model along with Algorithm 5 offer an alternative to overcome that difficulty. Specifically, that method is applied to railway track circuits to illustrate that capability through real examples. It is worth emphasizing that railway track circuits are more likely to have unevenly distributed electrical properties as opposed to power wires or antennas. For instance, humidity in both soil and ballast bed can impact those properties as indicated by [63]. In addition, some sudden external influences, like lightning, can cause damage to a track circuit system which may lead to a catastrophic safety monitoring failures where two trains are present within the same track segment. In this section, two types of situations are considered. First, it illustrates how voltage and current would change along a degraded sector of railway track circuit. Second, it demonstrates how voltage would vary when a train is passing through a sector of that circuit.

All constants used in this section are the same as those in Section 3.2.1. Besides, the number of subsections is fixed at 117 here, so each subsection takes 10 meters. In addition, at the transmitter end, $V_{max}$ is fixed at the one in Figure 3.6, *i.e.*, $V_{max} = 115V$ at $x = 1170m$. That means the boundary conditions are given at the transmitter end instead of the receiver end in this section.

The first example of unevenly distribution exhibited here is ballast degradation, which means unusual current leakage between the rails through the ballast [35]. In this damage case, some shunt resistance $rb_g$ become lower and shunt capacitance $c_g$ become higher. Suppose a ballast degradation happens between $x = 100m$ and $x = 1000m$, that is between subsection 18 and subsection 107. Note that the order of subsections is opposite to the direction of $x$. Hence, the list of damaged components is

$$\boldsymbol{l} = \begin{bmatrix} rb_{18} & \cdots & rb_{107} & c_{18} & \cdots & c_{107} \end{bmatrix}.$$

The assumed list of damage amounts $\boldsymbol{e}$ is plotted in Figure 3.9. When the above damage case $(\boldsymbol{l}, \boldsymbol{e})$ is used in Algorithm 5, the resultant voltage and current distri-

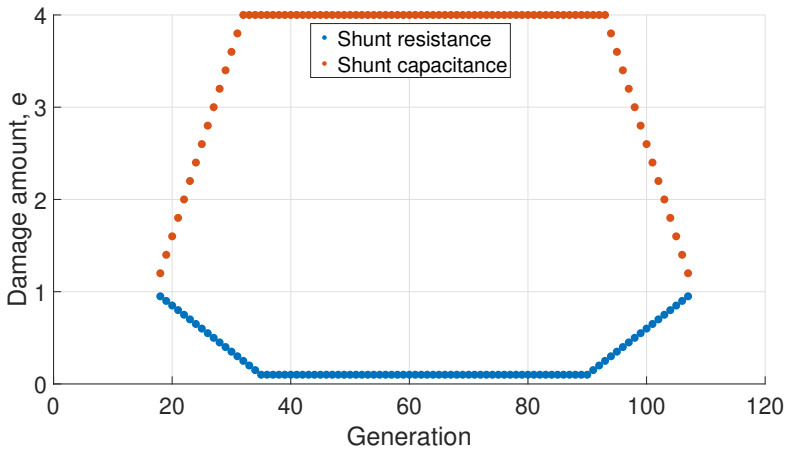


Figure 3.9. The list of damage amounts $\boldsymbol{e}$ is plotted *versus* generations where the blue dots are for the shunt resistance from $rb_{18}$ to $rb_{107}$, and the red dots are for the shunt capacitance from $c_{18}$ to $c_{107}$.

bution along the track for this type of ballast degradation are shown in Figures 3.10 and 3.11. Note that the other two damage cases mentioned in [35], insulation imperfections and rail conductance impairments, can also be simulated by the circuit network model.

The second example of application to railway track circuits is simulating the current at the receiver end while a train is passing through a sector of an intact track. Although the track itself is undamaged, a train's progression through it can be regarded as a sequence of damage cases.

Assume that a 190-meter train is moving from the receiver end to the transmitter end at a constant speed $100m/s$, *i.e.*, it passes one subsection every 0.1 seconds. Besides, that train has one wheel base every 10 meters, so there are 20 wheel bases in total. Each wheel base acts as an additional shunt resistor across two rails with resistance $r_w = 102\Omega$. In the layout of the circuit network model used in this section where each subsection takes $\Delta x = 10$ meters, according to Equation (3.8), the

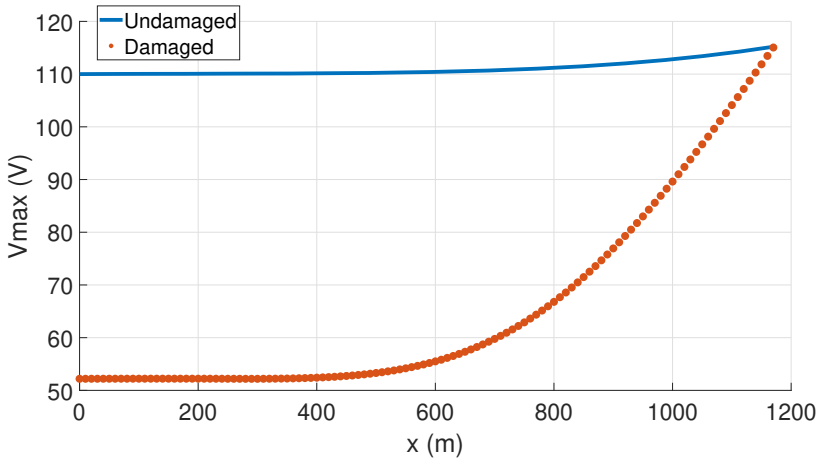


Figure 3.10. The distribution of $\max_t(|v(x,t)|)$ at each node between two adjacent subsections obtained by the circuit network model when the ballast degradation occurs. The undamaged curve is obtained by transmission line theory which is the same as that in Figure 3.6.

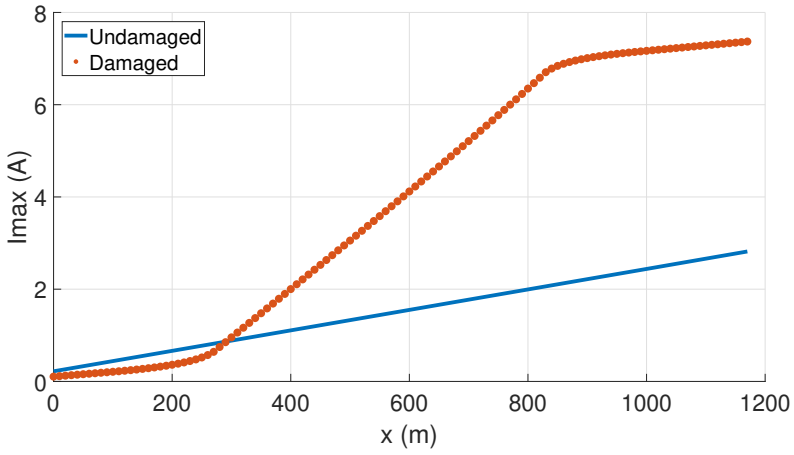


Figure 3.11. The distribution of $\max_t(|i(x,t)|)$ at each node between two adjacent subsections obtained by the circuit network model when the ballast degradation occurs. The undamaged curve is obtained by transmission line theory which is the same as that in Figure 3.7.

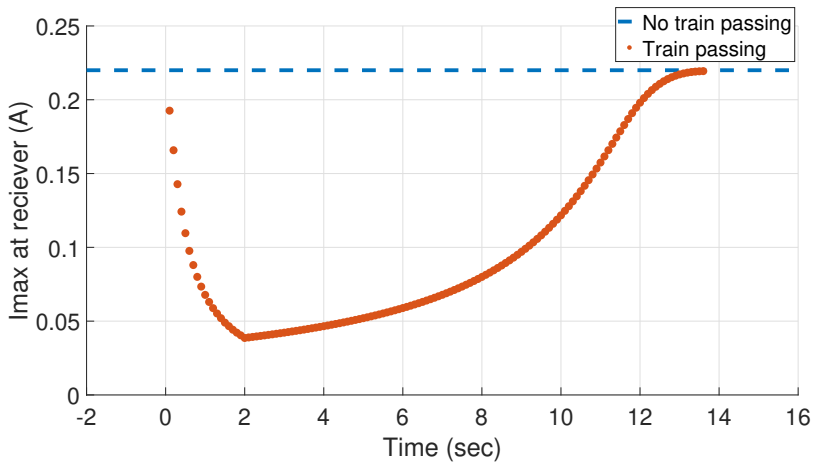


Figure 3.12. The variation in the current at the receiver end, $\max_t |i(0,t)|$, as the train passing through an intact track obtained by the circuit network model.

undamaged value of the shunt resistance is

$$rb = 1/(20 \times 10^{-6} \times 10) = 5000\Omega.$$

When the shunt resistance $rb$ is connected to the resistance $r_w$ of one wheel base in parallel, the equivalent resistance is $100\Omega$. In other words, when one wheel base is within one subsection, we can consider that as if that corresponding shunt resistance is damaged by a factor of 0.02. Therefore, this example of a train moving along an intact track can be regarded as a time series of damage cases, where the correspondence between the time instance and the damage case is listed in Table 3.2. At each time instance in Table 3.2, the corresponding damage case in the second column is used to call Algorithm 5. Then, the current at the receiver end, $\max_t |i(0,t)|$ as the train is passing through this sector of track can be evaluated, which is plotted in Figure 3.12. Note that Figure 3.12 shares similar characteristics with real measurements presented in [35].

It is worth pointing out that every damage case takes less than one second to com-

TABLE 3.2

THE EQUIVALENT DAMAGE CASE AT EACH TIME INSTANCE FOR THE TRAIN PASSING EXAMPLE.

| Time (sec) | Damage case $(\boldsymbol{l}, \boldsymbol{e})$ |
|---|---|
| 0.1 | $([rb_{117}], [0.02])$ |
| 0.2 | $([rb_{117}, rb_{116}], [0.02, 0.02])$ |
| $\vdots$ | $\vdots$ |
| 2 | $([rb_{117}, \cdots, rb_{98}], [0.02, \cdots, 0.02])$ |
| 2.1 | $([rb_{116}, \cdots, rb_{97}], [0.02, \cdots, 0.02])$ |
| $\vdots$ | $\vdots$ |
| 11.7 | $([rb_{20}, \cdots, rb_1], [0.02, \cdots, 0.02])$ |
| 11.8 | $([rb_{19}, \cdots, rb_1], [0.02, \cdots, 0.02])$ |
| $\vdots$ | $\vdots$ |
| 13.6 | $([rb_1], [0.02])$ |

pute with the setup of the 117-generation network in this section. The computation is run on `MATLAB R2020b` with a single CPU of Intel Core i7-10510U Processor.

## 3.4 Concluding Remarks

This chapter presented the application of the modeling methods in Chapter 2 to simulating a realistic system, a railway track circuit. Two scenarios are considered, one of which concerns a damage situation when a ballast degradation occurs, while the other quantifies the current's variation as a train is passing through. From a general point of view, they are examples of approximating voltage and current along a transmission line when its electrical properties are unevenly distributed. In the next two chapters, we are going to see other two applications of the modeling methods in Chapter 2, that is monitoring and controlling networked dynamical systems.

# CHAPTER 4

# HEALTH MONITORING OF NETWORKED DYNAMICAL SYSTEMS

Dynamic networks are often large-scale and consist of numerous components. Thus, they are far easier to encounter damages compared to simple systems. As a result, it is essential to track their health and locate deleterious components. This chapter illustrates health monitoring methods for networked dynamical systems through their frequency response given by the modeling algorithms from Chapter 2. The method leverages the mismatch between the measured frequency response and the computed one as the feature and identifies the existence, location and extent of damage.

This chapter starts with a basic damage detection procedure where the list of the most likely damaged components along with their identified damage amounts are returned. Built upon that, a modified process leverages agglomerative hierarchical clustering to provide a set of possible damage cases instead of only giving the most likely one. The idea is that the actual damage case has more probability to be covered by that narrowed shortlist so that it can be discovered more efficiently as opposed to a thorough inspection of the entire network. Finally, this chapter ends with exhibiting how the damage detection method could cooperate with hardware inspections in order to save the time of examining large networks' health. Although the networks used in this chapter are all infinitely large, the methods proposed here are applicable to all networks fulfilling the assumptions (A-1) to (A-6) in Section 2.1. The contents of this chapter appear in [110, 111]

## 4.1 Basic Damage Detection Procedure

This section gives a basic damage detection procedure which returns the damage case $(\boldsymbol{l}, \boldsymbol{e})$ whose frequency response $G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega_s)$ best matches the measurements $\overline{G}(j\omega_s)$. That straightforward idea leads to the following minimization problem.

$$\min_{(\boldsymbol{l},\boldsymbol{e})} J(\boldsymbol{l}, \boldsymbol{e}) = \sum_{\omega_s} \frac{\|G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega_s) - \overline{G}(j\omega_s)\|}{\|\overline{G}(j\omega_s)\|}, \tag{4.1}$$

subject to

$$\forall \epsilon \in \boldsymbol{e},\ \epsilon > 0,$$

where $G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega_s)$ is computed by a modeling algorithm from Chapter 2 at the sampling frequencies $\omega_s$, $\overline{G}(j\omega_s)$ is a damaged network's frequency response measurement, and the operator $\|\cdot\|$ indicates the Euclidean norm of complex numbers. However, because the list of damaged components $\boldsymbol{l}$ is a discrete variable and the list of damage amounts $\boldsymbol{e}$ is a continuous variable, the optimization problem, Equation (4.1), is a mixed-integer nonlinear programming problem. To avoid that, there exist two bypasses. First, the list $\boldsymbol{l}$ could include all components, which does not yield correct result in most of the intricate cases. Therefore, the methods in this chapter go through the other way where a set of all candidate $\boldsymbol{l}$, $\mathcal{L}$, is selected by users in advance. Then, the basic damage detection procedure forms the following three-step paradigm.

- Step 1: A set of candidate lists of damaged components $\boldsymbol{l}$ is chosen, denoted by $\mathcal{L}$.
- Step 2: For all $\boldsymbol{l} \in \mathcal{L}$, the following nonlinear optimization problem is solved.

  $$\min_{\boldsymbol{e}} J(\boldsymbol{e}) = \sum_{\omega_s} \frac{\|G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega_s) - \overline{G}(j\omega_s)\|}{\|\overline{G}(j\omega_s)\|}, \tag{4.2}$$

  subject to

  $$\forall \epsilon \in \boldsymbol{e},\ \epsilon > 0.$$

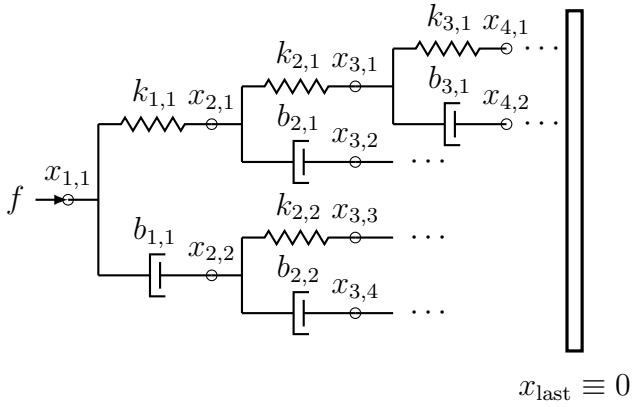


Figure 4.1. Mechanical tree network from Chapter 2.

For every $\boldsymbol{l} \in \mathcal{L}$, its minimizer $\boldsymbol{e}^*$ and the corresponding minimal $J^*$ are stored, where $J^*$ is called the *mismatch* of the candidate damage case $(\boldsymbol{l}, \boldsymbol{e}^*)$.

- Step 3: Among all $\boldsymbol{l} \in \mathcal{L}$, the one with the globally smallest mismatch is returned, and its corresponding damage case $(\boldsymbol{l}, \boldsymbol{e}^*)$ is regarded as the most likely one among all candidates.

The above three-step procedure is tested on an infinitely large mechanical tree network as shown in Figure 4.1, which is same as that in Chapter 2. From the modeling results in Chapter 2, we know that for all damage cases, its transfer function can always be represented by

$$G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s) = G_{\infty,\varnothing}(s)\Delta_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s),$$

where the undamaged transfer function is always

$$G_{\infty,\varnothing}(s) = \frac{1}{\sqrt{kbs}}.$$

Therefore, the $\Delta_{\infty,(\boldsymbol{l},\boldsymbol{e})}(s)$ part is the focus in this section for the damage detection purpose.

As a concrete example of the aforementioned three-step procedure, consider a

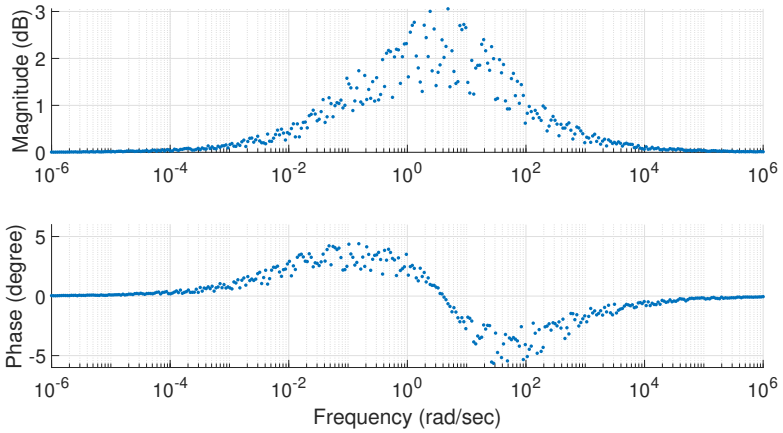


Figure 4.2. A noisy frequency response measurement $\overline{\Delta}(j\omega_s)$ for a damaged mechanical tree whose damage case $(\boldsymbol{l}, \boldsymbol{e}) = ([b_{1,1}, k_{3,1}], [0.45, 0.65])$. (50% noise added).

noisy frequency response measurement $\overline{\Delta}(j\omega_s)$ as shown in Figure 4.2 whose actual damage case is

$$(\boldsymbol{l}, \boldsymbol{e}) = ([b_{1,1}, k_{3,1}], [0.45, 0.65]),$$

with an additional 50% noise. The level of additional noise is quantified as follows. A noise with the level of $n\%$ means that if for the actual damage case $(\boldsymbol{l}, \boldsymbol{e})$, the modeling algorithms from Chapter 2 return its frequency response at the sampling frequency $\omega_s$ is $\Delta_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega_s)$, what the damage detection procedure can see is its corresponding noisy value of $\overline{\Delta}(j\omega_s)$ that

$$\|\overline{\Delta}(j\omega_s)\| = r_1 \|\Delta_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega_s)\|,$$
$$\angle\overline{\Delta}(j\omega_s) = r_2 \angle\Delta_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega_s),$$

where $r_1$ and $r_2$ are uniformly distributed random numbers in the range $(1 - n\%, 1 + n\%)$.

At the first step, the set containing all candidate $\boldsymbol{l}$ is defined as any combination

TABLE 4.1

STORED RESULTS FOR EACH ELEMENT $\boldsymbol{l} \in \mathcal{L}$ IN EQUATION (4.3) AFTER SOLVING THE MINIMIZATION PROBLEM IN EQUATION (4.2).

| Candidate damage cases | | |
|---|---|---|
| Candidate $\boldsymbol{l}$ | Locally best $\boldsymbol{e}^*$ | Locally smallest mismatch $J^*$ |
| $[k_{1,1}, b_{1,1}]$ | $[0.814, 0.517]$ | 26.2 |
| $[k_{1,1}, k_{2,1}]$ | $[0.481, 1.000]$ | 37.0 |
| $\vdots$ | $\vdots$ | $\vdots$ |
| $[b_{1,1}, k_{3,1}]$ | $[0.437, 0.665]$ | 23.8 |
| $\vdots$ | $\vdots$ | $\vdots$ |
| $[k_{4,8}, b_{4,8}]$ | $[0.001, 0.686]$ | 73.0 |

of two components in the first four generations. That is,

$$\mathcal{L} = \{[k_{1,1}, b_{1,1}], [k_{1,1}, k_{2,1}], \cdots, [k_{4,8}, b_{4,8}]\}. \tag{4.3}$$

At the second step, the optimization problem, Equation (4.2), is solved iteratively for each candidate $\boldsymbol{l} \in \mathcal{L}$. For the first element $\boldsymbol{l} = [k_{1,1}, b_{1,1}]$, the optimization solver returns that $\boldsymbol{e}^* = [0.814, 0.517]$ gives the locally smallest mismatch $J^*$, where $J^* = 26.2$. That result as well as the results for all the other $\boldsymbol{l} \in \mathcal{L}$ are given in Table 4.1. Finally, at the third step, it is determined that $J^* = 23.8$ is the globally smallest mismatch. Therefore, its corresponding damage case

$$(\boldsymbol{l}, \boldsymbol{e}^*) = ([b_{1,1}, k_{3,1}], [0.437, 0.665])$$

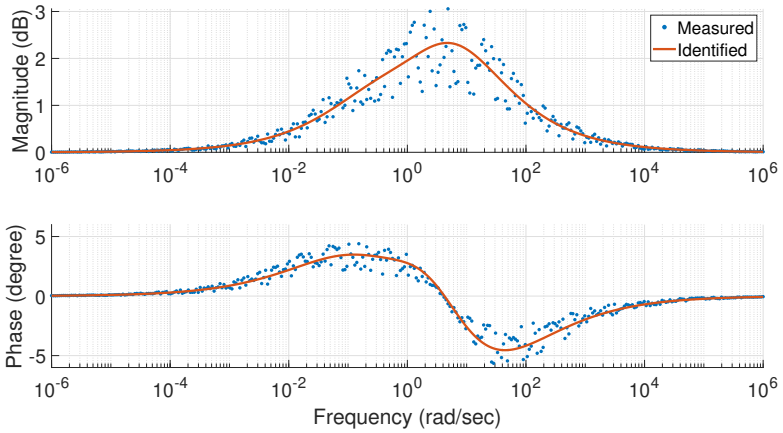


Figure 4.3. The noisy measurement from Figure 4.2 is plotted along with its identified $\Delta_{\infty,(\boldsymbol{l},\boldsymbol{e}^*)}(j\omega_s)$.

is returned as the final result and is regarded as the most likely damage case. The identified frequency response $\Delta_{\infty,(\boldsymbol{l},\boldsymbol{e}^*)}(j\omega_s)$ is plotted in Figure 4.3 along with the same noisy measurement from Figure 4.2. Note that in this example, the damage detection method correctly identifies the damage because the identified damaged components in $\boldsymbol{l}$ are correct, and the identified damage amounts in $\boldsymbol{e}^*$ are close to their actual values.

### 4.1.1 Performance Test

The performance of the proposed three-step damage detection method is tested for all damage cases where at most two components are damaged within the first three generations of an infinitely large mechanical tree network in Figure 4.1. The optimization problem, Equation (4.2), is solved by the `fmincon()` function in `MATLAB`. For each actual damaged component, ten different damage amounts from 0.05 to 0.95 with an increment of 0.1 are tested. From Figure 4.1, we can conclude that there are $2 + 2^2 + 2^3 = 14$ components in the first three generations for a mechanical tree

network. Therefore, the number of actual damage cases is

$$\binom{14}{1} \times 10 + \binom{14}{2} \times 10^2 = 9,240.$$

To imitate real measurements, 21 different levels of noise from 0% to 100% with a increment of 5% are also added to the frequency response measurements $\overline{\Delta}(j\omega_s)$. Therefore, the proposed damage detection procedure goes through $21 \times 9,240 = 194,040$ tests in total.

Since the damages are limited to the first three generations, at the first step of each test, the set $\mathcal{L}$ is selected to cover all combinations of at most two components in the first four generations. That is,

$$\mathcal{L} = \{k_{1,1}, b_{1,1}, \cdots, b_{4,8}, [k_{1,1}, b_{1,1}], [k_{1,1}, k_{2,1}], \cdots, [k_{4,8}, b_{4,8}]\}.$$

The results of the above tests are concluded as follows. When no noise presenting in the frequency response measurement $\overline{\Delta}(j\omega_s)$, the proposed damage detection method does very well. Out of 9240 damage cases, the method only misidentifies the following two damage cases:

1. It misidentifies $([k_{3,2}, k_{3,3}], [0.95, 0.85])$ as $([b_{3,2}, k_{3,3}], [0.9622, 0.8395])$;
2. It misidentifies $([b_{3,2}, b_{3,3}], [0.85, 0.95])$ as $([b_{3,2}, k_{3,3}], [0.8395, 0.9622])$.

The above two misidentified cases reveal the nature of how difficult it is to identify damages within large networks given only their overall response. Ideally, when the measurements are perfect, our algorithm should identify all damages exactly, since it uses the knowledge from computing a damage case's frequency response. However, that is not the case as indicated by the existence of the aforementioned two misidentified cases even when all actual damage cases are limited to the first three generations. The main reason is that the mapping from a damage case to its frequency response

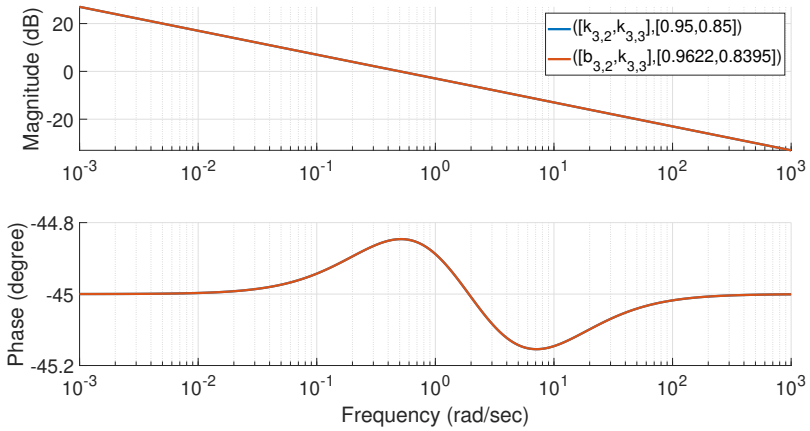


Figure 4.4. The frequency response $G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega)$ of two different damage cases almost overlap each other, which reveals the fact that the mapping from a damage case to its frequency response is not completely one-to-one.

is not completely one-to-one. Therefore, not all inverse problems can be solved exactly. For instance, Figure 4.4 shows the Bode plot for those two different damage cases in the first misidentified case above, from which we can confirm that their frequency responses $G_{\infty,(\boldsymbol{l},\boldsymbol{e})}(j\omega)$ are almost the same. Except for the aforementioned two misidentified cases, all the other damage cases are correctly identified, with the maximum absolute error for the list of damage amounts $\boldsymbol{e}$ being $1.26 \times 10^{-4}$, which happens at the actual damage case $([b_{2,2}, k_{3,4}], [0.05, 0.45])$.

When noise presents in the frequency response measurements, the number of misidentified cases increases with the level of additional noise in the frequency response measurements. However, the performance of the proposed damage detection method is still reasonably good. Figure 4.5 plots the percentage of misidentified cases out of 9240 total damage cases *versus* the level of additional noise in the frequency response measurements. From Figure 4.5, we can observe that, for example, even when 50% noise is added to the frequency response measurements, which should be as noisy as that shown in Figure 4.2, only 35% of total 9240 damage cases are

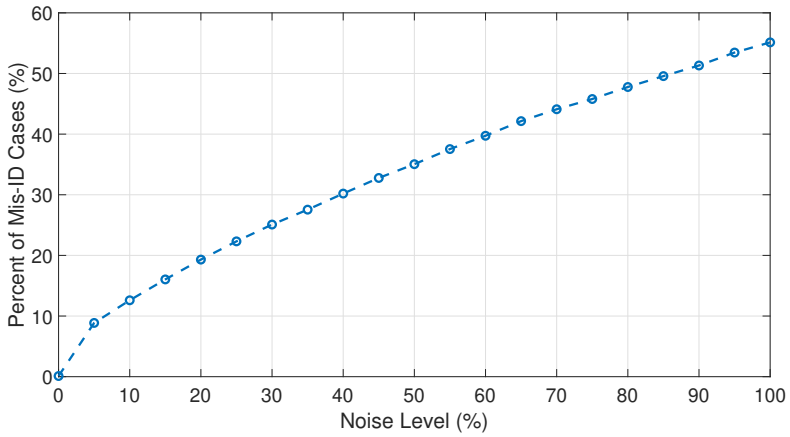


Figure 4.5. Percentage of misidentified cases during the test *versus* the level of noise added to the measurement.

misidentified.

Based on the test results, there also exist the following two observations when measurement noise presents:

1. The possibility of misidentification increases as the damage goes deeper.
2. At the same generation, damages happening on the inner components are more likely to be misidentified compared to those occurring on the outer components.

Both of the above observations are intuitive. The first observation can be seen from Figure 4.6, which compares the percentage of misidentified cases for those damage cases happening purely on the second and the third generation. From Figure 4.6, we notice that throughout all levels of additional noise, the damages purely happening on the second generation always have less possibility of misidentification as opposed to those on the third generation. The reason for that trend is the fact that the discrepancy between two frequency responses becomes less distinguishable when damages reside deeper in a network.

The second observation can be spotted from the damage cases occurring on the third generation, where $k_{3,1}$, $b_{3,1}$, $k_{3,4}$, $b_{3,4}$ are called outer components, and $k_{3,2}$, $b_{3,2}$,

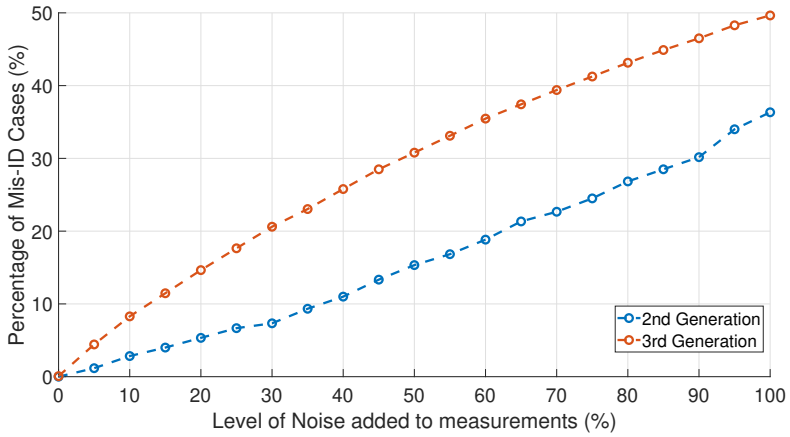


Figure 4.6. Percentage of misidentified cases *versus* level of additional noise for those damage cases which are purely on the second and the third generation.

$k_{3,3}$, $b_{3,3}$ are called inner components. During the test, among all damage cases within the above four outer components, 19% are misidentified. However, among all damage cases within the above four inner components, 42% are misidentified, which is more than twice larger than the previous one.

## 4.2 Determining A Set of Possible Damage Cases

For the basic damage detection procedure in Section 4.1, only the candidate with the globally smallest mismatch is returned. However, that returned result may not coincide with the actual damaged components as proved by the performance tests in Section 4.1.1. In addition, the information regarding all the other candidates is wasted since the user would have no idea about them after the damage detection is done. To overcome the above two disadvantages of the basic damage detection procedure, in this section and the next one, Section 4.3, two revised procedures are provided to make that damage detection method more useful in real health monitoring applications. In this section, a modified procedure leverages the agglomerative hierarchical clustering

method to group all candidates into two sets. One set contains all components that are possibly damaged, and the other set includes all the other components that are unlikely to be damaged. That two-set result offers more information to the user as opposed to only providing the component that is most likely to be damaged. Moreover, that result acts to narrow down the shortlist of candidates for the locations where damages reside, so it has a potential to save the time required for hardware inspections. Before presenting the details of that modification on the basic damage detection method, agglomerative hierarchical clustering is briefly reviewed in the following Section 4.2.1.

### 4.2.1 Agglomerative Hierarchical Clustering

Agglomerative hierarchical clustering is a method to classify unlabeled data based on their mutual similarities. The basic idea is to construct a tree-shaped *dendrogram* according to the distances, or dissimilarities, among the datapoints which displays a multilevel hierarchical relation. Then, the user could decide the level of clustering that is most appropriate [101].

As a concrete example, suppose the following five datapoints on the two-dimensional space need to be classified.

- Object 1 at $O_1 = (1, 2)$;
- Object 2 at $O_2 = (2, 2)$;
- Object 3 at $O_3 = (4, 1)$;
- Object 4 at $O_4 = (4, 3)$;
- Object 5 at $O_5 = (3, 6)$.

The above five points are plotted in Figure 4.7. Before constructing the dendrogram, two quantities need to be defined. The first one is the metric of distances, where the Euclidean distance is used in this example. The second one is the distance between

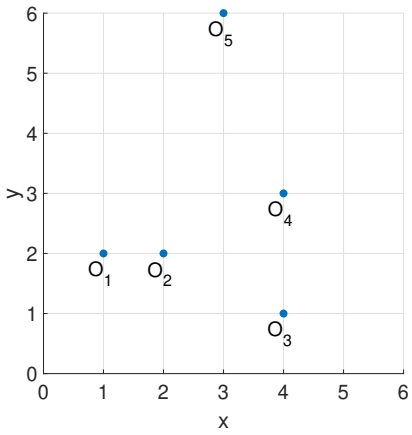


Figure 4.7. Five datapoints that need to be classified.

two clusters. Here, the shortest distance is used, *i.e.*, for two clusters $A$ and $B$, the distance between them $d(A, B)$ is defined as

$$d(A, B) = \min_{a \in A, b \in B} d(a, b),$$

where $d(a, b)$ is the distance metric defined previously (*i.e.*, Euclidean distance).

The construction of dendrogram starts by considering each object as one cluster and is built up iteratively. In each iteration, the two closest clusters, or two most similar clusters, are combined into one. That iteration terminates when there is only one cluster left. Specifically, for this example, at the first iteration, there are five initial clusters where $C_i = O_i$ for all $i = 1, 2, \ldots, 5$. Therefore, there are $\binom{5}{2} = 10$ pairwise distances as listed in Table 4.2, from which we can see that the closest clusters are $C_1$ and $C_2$. Therefore, they are combined into a new cluster $C_6$, *i.e.*, $C_6 = C_1 \cup C_2$, and the rows in Table 4.2 regarding $C_1$ or $C_2$ are revised accordingly in the second iteration, which results in Table 4.3. Note that the dissimilarity between two clusters is defined by the aforementioned shortest distance. For example, the distance between $C_6$ and $C_3$ equals the Euclidean distance between $O_2$ and $O_3$.

TABLE 4.2

PAIRWISE DISTANCES AT THE FIRST ITERATION OF BUILDING THE DENDROGRAM.

| Clusters | | Distance |
|---|---|---|
| $C_1$ | $C_2$ | 1 |
| $C_1$ | $C_3$ | $\sqrt{10}$ |
| $C_1$ | $C_4$ | $\sqrt{10}$ |
| $C_1$ | $C_5$ | $\sqrt{20}$ |
| $\vdots$ | $\vdots$ | $\vdots$ |
| $C_4$ | $C_5$ | $\sqrt{10}$ |

TABLE 4.3

PAIRWISE DISTANCES AT THE SECOND ITERATION OF BUILDING THE DENDROGRAM.

| Clusters | | Distance |
|---|---|---|
| $C_6$ | $C_3$ | $\sqrt{5}$ |
| $C_6$ | $C_4$ | $\sqrt{5}$ |
| $C_6$ | $C_5$ | $\sqrt{17}$ |
| $C_3$ | $C_4$ | 2 |
| $C_3$ | $C_5$ | $\sqrt{26}$ |
| $C_4$ | $C_5$ | $\sqrt{10}$ |

TABLE 4.4

PAIRWISE DISTANCES AT THE THIRD ITERATION OF BUILDING THE DENDROGRAM.

| Clusters | | Distance |
|---|---|---|
| $C_6$ | $C_7$ | $\sqrt{5}$ |
| $C_6$ | $C_5$ | $\sqrt{17}$ |
| $C_7$ | $C_5$ | $\sqrt{10}$ |

From Table 4.3, we see that the distance between $C_3$ and $C_4$ is the smallest, so they are combined to form a new cluster again, *i.e.*, $C_7 = C_3 \cup C_4$. Then, the rows regarding $C_3$ or $C_4$ in Table 4.3 are revised appropriately, which gives Table 4.4. Again Table 4.4 shows that $C_6$ and $C_7$ should be merged to a new cluster $C_8$, and the one last remaining pairwise distance is between $C_8$ and $C_5$, which is equal to $\sqrt{10}$. Finally, the iteration terminates with merging $C_8$ with $C_5$ to form a new and only cluster $C_9$. The dendrogram built from the above iterations is plotted in Figure 4.8 where the $x$-axis indicates the objects and the $y$-axis is the distance when two clusters are merged together. The resultant clusters are also shown in Figure 4.9.

Given a dendrogram in Figure 4.8, the user can decide where to cut it. For instance, if a user needs two clusters, the dendrogram in Figure 4.8 provides $C_8 = \{O_1, O_2, O_3, O_4\}$ and $O_5$. Instead, if that user wants three clusters, the dendrogram in Figure 4.8 provides $C_6 = \{O_1, O_2\}$, $C_7 = \{O_3, O_4\}$ and $O_5$. There also exist algorithms which aim at finding natural divisions in data given their dendrogram instead of the aforementioned user-specified divisions.

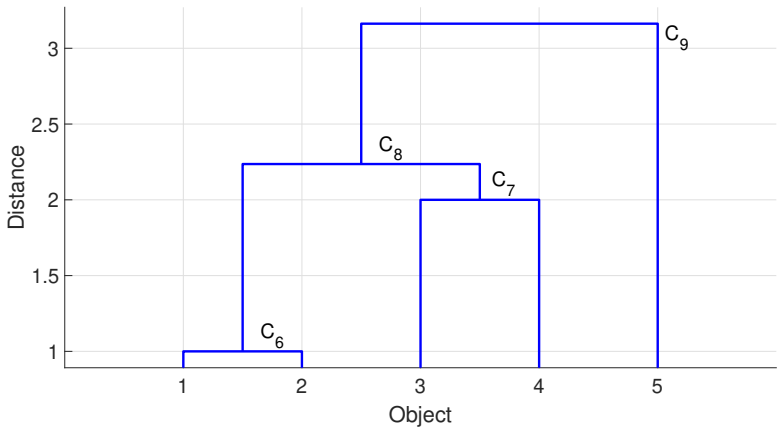


Figure 4.8. Dendrogram for the agglomerative hierarchical clustering example.

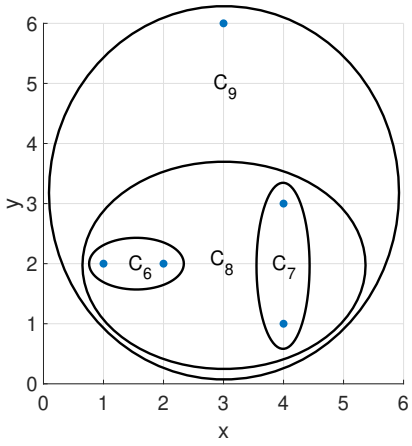


Figure 4.9. Resultant clusters of the agglomerative hierarchical clustering example.

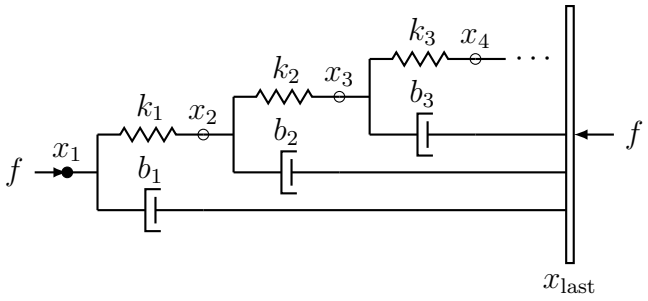


Figure 4.10. Infinite mechanical ladder network.

### 4.2.2 A Revised Damage Detection Procedure

This section provides a revised damage detection procedure from the basic one discussed in Section 4.1. Specifically, the first two steps in the original three-step procedure in Section 4.1 are not modified, while the last step is revised as follows.

- Step 3: Use agglomerative hierarchical clustering to partition all $\boldsymbol{l} \in \mathcal{L}$ into two sets based on their mismatch after solving the optimization problem in Equation (4.2). All $\boldsymbol{l}$ that belong to the same set as the one with the globally smallest mismatch are considered as possible damaged components. All $\boldsymbol{l}$ in the other set are regarded as components that are unlikely to be damaged.

By doing that, the information conveyed by the mismatch of all candidate $\boldsymbol{l}$ is not wasted. Instead, it guides the revised damage detection method to narrow down the shortlist of candidates.

Here, an infinitely large mechanical ladder network plotted in Figure 4.10 is employed as a concrete example. The undamaged constants are the same as the mechanical tree network in Chapter 2 where $k = 2N/m$ and $b = 1Ns/m$. The frequency response measurement $\overline{G}(j\omega_s)$ used in this example is shown in Figure 4.11, where the actual damage case is $(\boldsymbol{l}, \boldsymbol{e}) = ([k_1, b_1], [0.15, 0.25])$ whose frequency response is then perturbed by additional noise at the level of 30%.

The first step is the same as the original damage detection method where a set of potential damaged components, denoted by $\mathcal{L}$, is chosen. For the concrete example

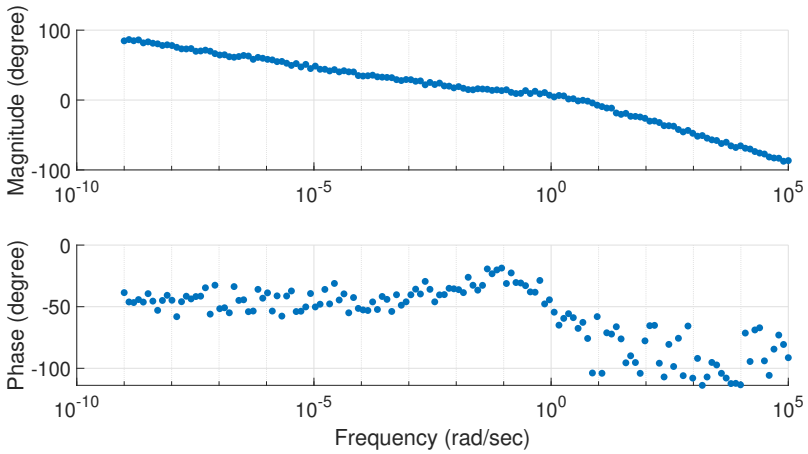


Figure 4.11. Frequency response measurements for the damage case $(\boldsymbol{l}, \boldsymbol{e}) = ([k_1, b_1], [0.15, 0.25])$ with additional 30% noises.

here, $\mathcal{L}$ is selected to cover any combination of two components in the first five generations. There are ten components in total in the first five generations. Hence, the cardinality $|\mathcal{L}| = \binom{10}{2} = 45$ with

$$\mathcal{L} = \{[k_1, b_1], [k_1, k_2], \ldots, [k_1, b_5], \ldots, [k_5, b_5]\}. \tag{4.4}$$

The second step is also the same as the original method, where the optimization problem in Equation (4.2) is solved for all $\boldsymbol{l} \in \mathcal{L}$, and every resultant minimizer $\boldsymbol{e}^*$ and minimal $J^*$ are stored. For the concrete example, the results are listed in Table 4.5.

In the modified third step, those mismatch $J^*$ listed in Table 4.5 are regarded as the feature to classify all those candidate damage cases into two groups through agglomerative hierarchical clustering. The resultant dendrogram for the example is plotted in Figure 4.12. By cutting the dendrogram in Figure 4.12 into two clusters, we can observe that the candidates with the first nine smallest mismatches belong to the same cluster. Therefore, they are considered as possible damage cases, while all

TABLE 4.5

STORED RESULTS FOR EACH ELEMENT $\boldsymbol{l} \in \mathcal{L}$ IN EQUATION (4.4) AFTER SOLVING THE MINIMIZATION PROBLEM IN EQUATION (4.2).

| Candidate damage cases | | |
|---|---|---|
| Damaged components $\boldsymbol{l}$ | Identified damage amounts $\boldsymbol{e}^*$ | Mismatch $J^*$ |
| $[k_1, b_1]$ | $[0.1901, 0.2698]$ | 32.0424 |
| $[k_1, k_2]$ | $[0.1649, 1.0000]$ | 58.8518 |
| $[k_1, b_2]$ | $[0.1924, 0.0185]$ | 58.8599 |
| $[k_1, k_3]$ | $[0.1649, 1.0000]$ | 58.8518 |
| $\vdots$ | $\vdots$ | $\vdots$ |

other 36 candidates are regarded as unlikely. The corresponding decision boundary between those two clusters is shown in Figure 4.13. From Figure 4.13, we can confirm that the leftmost blue dot is the only result returned by the original damage detection method. However, for this revised method, the information of all other candidates' mismatches is utilized to guide the inspections to determine which components require more efforts in the hardware examination.

The final result returned by the revised damage detection method for the example is listed in Table 4.6. As indicated by Table 4.6, the most likely damage case with the globally smallest mismatch is $([k_1, b_1], [0.1901, 0.2698])$ which are the same as the actual damaged components with the averaged absolute error in the damaged amounts

$$\frac{1}{2}(|0.1901 - 0.15| + |0.2698 - 0.25|) = 0.0299.$$

The performance test result of the original damage detection method in Section 4.1.1

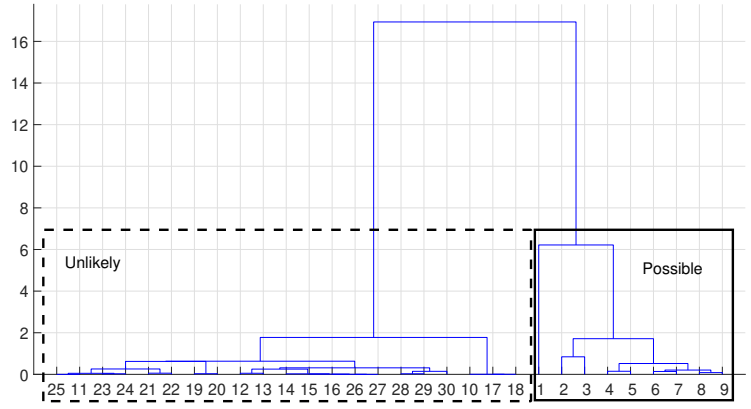


Figure 4.12. Dendrogram for all candidate damage cases listed in Table 4.5 by using their corresponding mismatch $J^*$ as the feature.

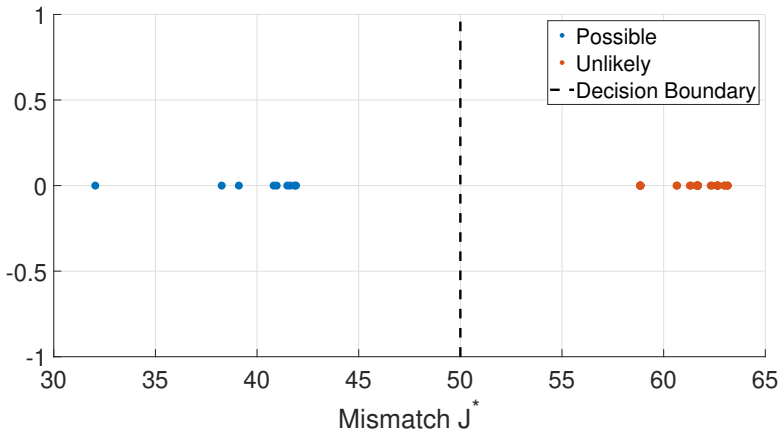


Figure 4.13. Decision boundary when the dendrogram in Figure 4.12 is cut into two clusters. The blue dots represent mismatches of the candidates that are considered as possible damage cases, while the red dots are for the candidates that are unlikely to be damaged. The leftmost blue dot is for the candidate with the globally smallest mismatch ($Y$-axis is meaningless in this plot).

TABLE 4.6

THE FINAL RESULT RETURNED BY THE REVISED DAMAGE DETECTION METHOD GIVEN THE MEASURED FREQUENCY RESPONSE IN FIGURE 4.11 AND THE SELECTED SET OF CANDIDATE DAMAGED COMPONENTS IN EQUATION (4.4).

| | Damage cases | Mismatch |
|---|---|---|
| Possible | $([k_1, b_1], [0.1901, 0.2698])$ | 32.0424 |
| | $([b_1, b_2], [0.2658, 0.0001])$ | 38.2606 |
| | $([b_1, k_2], [0.2675, 0.1924])$ | 39.1040 |
| | $\vdots$ | $\vdots$ |
| | $([b_1, b_5], [0.2699, 0.0001])$ | 41.9176 |
| Unlikely | $([k_1, k_2], [0.1649, 1.0000])$ | 58.8518 |
| | $\vdots$ | $\vdots$ |
| | $([k_4, b_5], [0.2688, 0.0011])$ | 63.1487 |

has proved that the damaged components with the globally smallest mismatch are not always correct. However, they have a much larger probability to be included in the set of possible damage cases given by the revised method here. That way of determining whether a candidate damage case is possible or unlikely narrows down the shortlist of damaged components and thus has potential to save the maintenance time as opposed to inspecting every component inside a large network.

### 4.2.3 Performance Test

Similar to the original damage detection method in Section 4.1, the revised one's performance is also tested here with respect to that infinitely large mechanical ladder network in Figure 4.10. In total, 94,770 different actual damage cases are trialed. Each actual damage case is constituted by three aspects. The first one is the damaged components which are assumed to be any combination of two components in the first five generations. As a result, there are a total of $\binom{10}{2} = 45$ different combinations of damaged components. The second aspect is the damage amounts where each component's damage amount ranges from 0.05 to 0.85 with an increment of 0.1. Therefore, there are a total of $9^2 = 81$ different combinations of damage amounts for each combination of damaged components. The third aspect is the additional noise where 26 different noise levels are tested extending from 0% to 70%. Hence, those three aspects result in

$$45 \times 81 \times 26 = 94,770$$

tests in total.

The result for each test case can be ranked by three levels as follows from the best to the worst.

- Rank One: The damaged components with the globally smallest mismatch are the same as the actual ones.
- Rank Two: The actual damaged components are different from the ones with

the globally smallest mismatch, while they still belong to the set of possible damage cases.

- Rank Three: The actual damaged components are in the set of unlikely damage cases.

For instance, the result of the example in Section 4.2.2 belongs to the best case. For the figures presented in this section, blue, red and yellow are used to indicate rank one, two and three results, respectively.

There exist the following three observations from the performance test results, which also follow intuitions.

1. Test cases with actual damaged components at shallow generations have better identification results.
2. Test cases with severe damages (small $\boldsymbol{e}$) have better identification results.
3. Test cases with less measurement noise have better identification results.

The above three observations can be spotted from Figures 4.14 to 4.16. Overall, the revised damage detection method works reasonably well because in the worst case, rank three results only take less than 10% of the total test cases.

## 4.3 Cooperation with Hardware Inspections

Both the original damage detection method in Section 4.1 and the revised one in Section 4.2.2 are separated from hardware inspections in the sense that they could provide identification results once a frequency response measurement is provided. Built upon those two methods, this section showcases a damage detection procedure which discovers damaged components iteratively and interacts with hardware inspections. The basic idea is that at every iteration, the damage detection procedure suggests the most suspicious components. Then, an inspection is conduct at those components, whose result is then fed to the damage detection procedure at the next

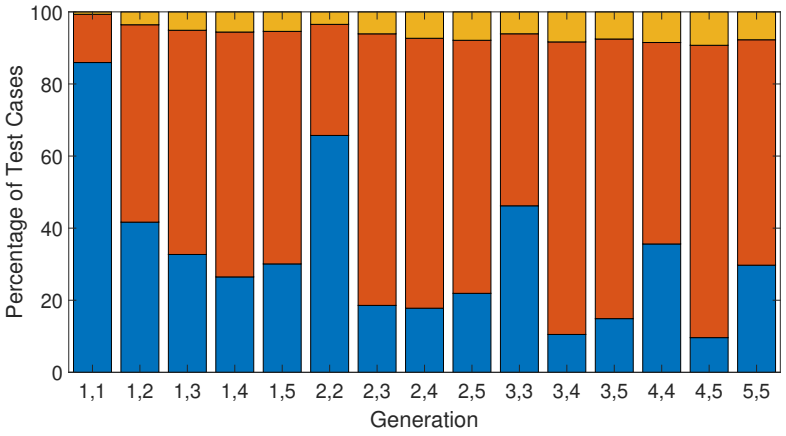


Figure 4.14. Percentage of test cases *versus* the generations where the actual damaged components locate. For example, $k_2$ and $b_4$ belong to the second and the fourth generation respectively. Hence, the list of damaged components $\boldsymbol{l} = [k_2, b_4]$ is a member of the $2, 4$ column in this figure. Blue: Rank one results. Red: Rank two results. Yellow: Rank three results.

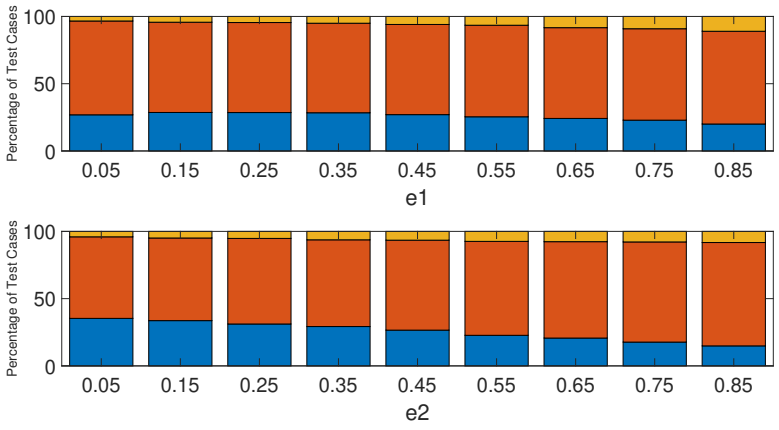


Figure 4.15. Percentage of test cases *versus* actual damage amounts where $[e_1, e_2] = \boldsymbol{e}$. Blue: Rank one results. Red: Rank two results. Yellow: Rank three results.

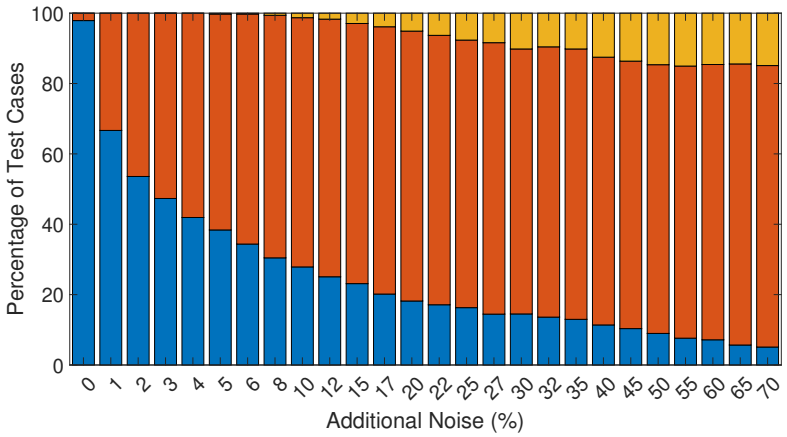


Figure 4.16. Percentage of test cases *versus* additional measurement noise. Blue: Rank one results. Red: Rank two results. Yellow: Rank three results.

iteration. That iterative process takes advantage of the fact that the set of candidate damaged components $\mathcal{L}$ is adaptive and subject to user's selection. The main motivation of that iterative method is to deal with more intricate damage cases and efficiently guide hardware inspections so that examinations can be performed more purposefully instead of testing all the components inside a large network.

Here, another specific example is given, where four damaged components are assumed in the first five generations of the infinite mechanical ladder network in Figure 4.10. Precisely, the actual damage case is

$$(\boldsymbol{l}, \boldsymbol{e}) = ([b_1, b_2, k_3, k_4], [0.05, 0.1, 0.15, 0.2]). \tag{4.5}$$

The corresponding noisy frequency response measurements are illustrated in Figure 4.17.

At the first iteration, the set $\mathcal{L}_1$ is picked to contain all single components in the first five generations, *i.e.*

$$\mathcal{L}_1 = \{[k_1], [b_1], [k_2], \ldots, [b_5]\}.$$

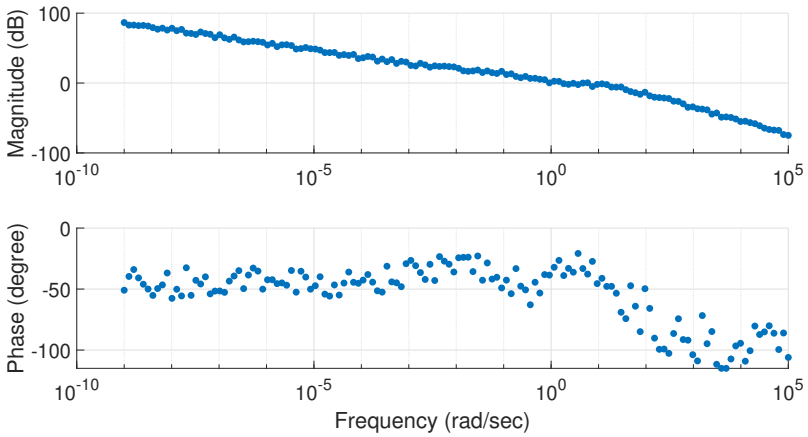


Figure 4.17. Frequency response measurements for the actual damage case in Equation (4.5) with additional 30% noise.

The original damage detection method in Section 4.1, provided the above set $\mathcal{L}_1$, returns that the damage case with the globally smallest mismatch is $([b_1], [0.0567])$. That damaged component is the most suspicious one at this iteration, so the user would inspect it on the hardware and obtain that its actual damage amount is 0.05. Then, after this iteration, the knowledge concerning the first discovered damaged component $([b_1], [0.05])$ is going to be supplied to the next iteration.

At the second iteration, the known component $b_1$ is excluded from the set of candidate damaged components. As a result,

$$\mathcal{L}_2 = \{[k_1], [k_2], [b_2], \ldots, [b_5]\}.$$

Each element in $\mathcal{L}_2$ would be grouped with the known component $b_1$ to form a candidate when solving the optimization problem in Equation (4.2). After going through all members in $\mathcal{L}_2$, the health monitoring process advises that the most

suspicious damage case at this iteration is

$$([b_1, k_2], [0.05, 0.1397]).$$

However, the new component $k_2$ is actually undamaged. Therefore, after the second inspection, the user would obtain the knowledge that the actual damage case is $([b_1, k_2], [0.05, 1])$ where the damage amount 1 indicates that $k_2$ is in fact undamaged.

At the third iteration, similarly, the set $\mathcal{L}_3$ is same as $\mathcal{L}_2$ except for the additional known component $k_2$, *i.e.*,

$$\mathcal{L}_3 = \{[k_1], [b_2], [k_3] \ldots, [b_5]\}.$$

This time, each element in $\mathcal{L}_3$ is combined with both known components $b_1$ and $k_2$ to form a candidate for identification. The result for this iteration and all the rest is concluded in Table 4.7.

After six iterations, we can observe that all actually damaged components are successfully discovered. That fact can be perceived from two perspectives. First, if we proceed to the seventh iteration and input the corresponding

$$\mathcal{L}_7 = \{[k_1], [b_3], [b_4], [b_5]\}$$

to the damage detection procedure, the returned damage amounts for them are in fact all 1.0000. That is to say, the optimization solver can no longer suggest any suspicious components at this iteration. The second observation can be made from how the globally smallest mismatch of each iteration varies, which is plotted in Figure 4.18. In Figure 4.18, we see that the globally smallest mismatch reaches a plateau after four iterations, which is another indication that either all damages have been discovered or the remaining undiscovered damages have little contribution to the

TABLE 4.7

THE RESULTS AFTER EACH ITERATION FOR THE EXAMPLE WHOSE ACTUAL DAMAGE CASE IS IN EQUATION (4.5) GIVEN ITS FREQUENCY RESPONSE MEASUREMENT IN FIGURE 4.17.

| Iter. | Suspicious Component | Inspection Result |
|---|---|---|
| 1 | $b_1$ | $([b_1], [0.05])$ |
| 2 | $k_2$ | $([b_1, k_2], [0.05, 1])$ |
| 3 | $k_3$ | $([b_1, k_2, k_3], [0.05, 1, 0.15])$ |
| 4 | $b_2$ | $([b_1, k_2, b_2, k_3], [0.05, 1, 0.1, 0.15])$ |
| 5 | $k_5$ | $([b_1, k_2, b_2, k_3, k_5], [0.05, 1, 0.1, 0.15, 1])$ |
| 6 | $k_4$ | $([b_1, k_2, b_2, k_3, k_4, k_5], [0.05, 1, 0.1, 0.15, 0.2, 1])$ |

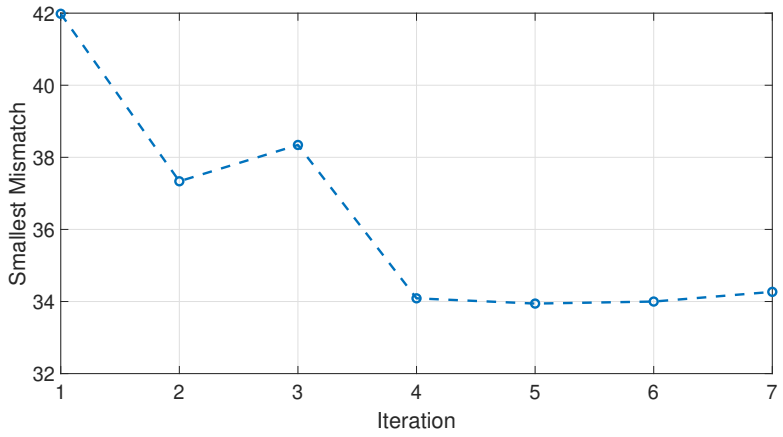


Figure 4.18. The globally smallest mismatch of each iteration for the example whose actual damage case is Equation (4.5).

overall frequency response and thus are hard to locate by only providing the frequency response measurements in Figure 4.17. In the latter case, more frequency response measurements within some smaller portion of the entire network should be provided. The aforementioned two evidences can be set as the stopping criterion for the iterative procedure discussed in this section.

Finally, the identified frequency response after each iteration is graphed in Figure 4.19. In conclusion, with the help of interaction between the proposed damage detection method and hardware inspections, only six investigations are necessary to discover all four damaged components instead of examining all ten elements, which has potentials to save the maintenance time.

## 4.4 Concluding Remarks

This chapter presents using the knowledge of frequency response of self-similar networks provided by Chapter 2 to monitor their health. The proposed damage detection method tries to identify existence, location and extent of damage. The input to that damage detection method is a measurement of a dynamic network's frequency

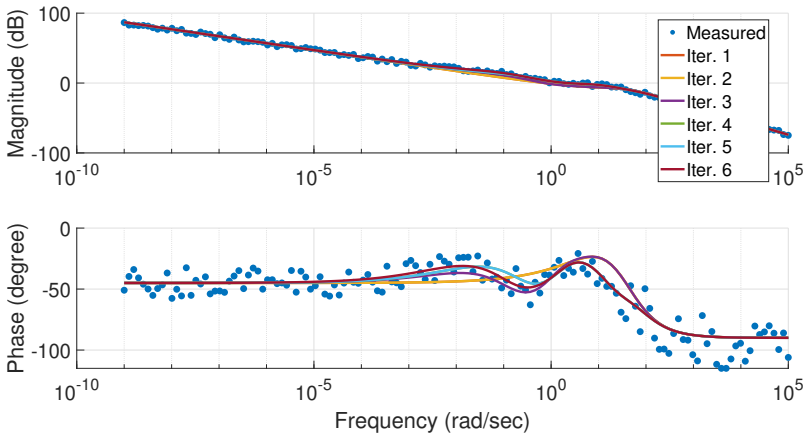


Figure 4.19. The resultant identified frequency response after each iteration for the specific example discussed in this section. The measurements are the same as those in Figure 4.17.

response $\overline{G}(j\omega_s)$. That method includes three steps. First, users pick a set $\mathcal{L}$ of candidate locations where damages could happen. Second, for each candidate location $\boldsymbol{l} \in \mathcal{L}$, an optimization solver finds the damage amount which best matches that frequency response measurement of the damaged network. Third, the one candidate which has the globally smallest mismatch among all $\boldsymbol{l} \in \mathcal{L}$ is returned. However, by doing that, the identification results of all the other $\boldsymbol{l} \in \mathcal{L}$ are wasted, and the returned result is not always consistent with the components which are actually damaged. Therefore, to overcome those two disadvantages, the third step is further modified to classifying all candidate $\boldsymbol{l}$ into two groups based on their mismatches obtained in the second step by using agglomerative hierarchical clustering. The cluster including the candidate with the globally smallest mismatch is considered as a collection of components where damages probably occur, while the other cluster is regarded as a collection of components which are unlikely to be damaged. In addition, this chapter also showcases how the proposed damage detection method could guide the hardware inspections in order to save the overall time of examination. In the next chapter, we

are going to see how that knowledge regarding dynamic networks' frequency response could also be leveraged to control their behavior.

# CHAPTER 5

# DEVELOPING UNIFIED CONTROLLERS FOR DYNAMIC NETWORKS WITH AN EXAMPLE OF VIBRATION CONTROL

Dynamic networks are more prone to different operating conditions due to their large numbers of components. In the previous chapter, that is the motivation of monitoring their health. The viewpoint of this chapter is more active, where it strives for a unified compensator to control their behavior even though their statuses are varying. From the knowledge of dynamic networks' frequency response and transfer functions brought by Chapter 2, it can be shown that their frequency response often form a set of neighboring plants under various conditions, which offers possibilities of designing a unified controller for that set of uncertain systems by using robust control methods.

In the literature of network control problems, especially for multi-agent systems, control strategies often render them to achieve a global task in all situations. However, the idea provided by this chapter tries to relax that requirement where a controller is required to give a desired performance to a dynamic network undergoing only certain extents of variations. Before designing controllers, the expected variations on a dynamic network's frequency response could be computed. Then, a unified controller can be designed specifically for those variations. That controller should be easier to obtain as it is not asked to work in any case, which may offer more design freedom when handling more intricate problems.

That idea is exemplified by an active vibration control problem of a multi-story building [42], where a unified controller is acquired by using loop shaping techniques

for a various number of floors. In an abstract point of view, that can be regarded as an example of dynamic networks whose nodes and edges disappear and reappear during service. A similar idea may also be applicable to the cases where the weights or transfer function blocks of edges inside a dynamic network keep varying.

## 5.1 Frequency Response and Transfer Function of A Multi-Story Building

This chapter models a multi-story building as a shear-frame structure with a lumped-mass planar network, as shown in Figure 5.1, where $u(t)$ is the force exerted by an external actuator (not included in Figure 5.1) installed on the top level, and $w(t)$ is the absolute displacement of the ground pointing to the right. Similar models are also utilized in other vibration control papers, *e.g.*, [53, 75, 64]. For a building with $n$ floors, its top level's response is thus represented by

$$X_n(s) = G_{n,u}(s)U(s) + G_{n,w}(s)W(s),$$

where $X_n$ is the absolute displacement pointing to the right. The frequency response and transfer functions of $G_{n,u}$ and $G_{n,w}$ are obtained by the two algorithms for finite networks from Chapter 2, which are adapted to the specific application here in Algorithms 6 and 7. Note that the `save()` function is the same as that in Chapter 3 which stores the intermediate results externally, so that the frequency response and transfer functions of $G_{i,u}$ and $G_{i,w}$ can be obtained for all $i$ from 1 to `nG` in one run.

The derivation of the one-generation transfer functions are straightforward. When the building only has one floor, it can be easily shown that

$$\begin{aligned} G_{1,u}(s) &= \frac{1}{m_1 s^2 + b_1 s + k_1}, \\ G_{1,w}(s) &= \frac{b_1 s + k_1}{m_1 s^2 + b_1 s + k_1}, \end{aligned} \tag{5.1}$$

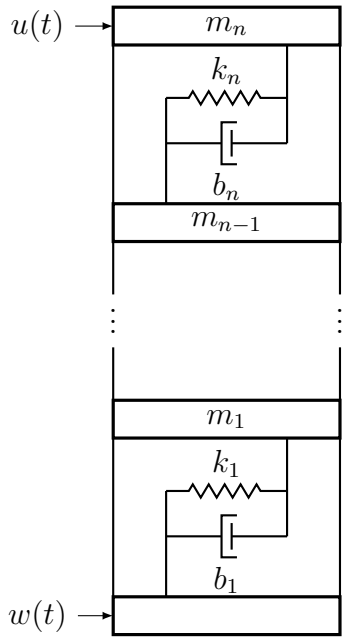


Figure 5.1. A multi-story building model.

**Algorithm 6** Pseudocode of computing the frequency response `Gu` and `Gw` at the angular frequency `omega` for a building with `nG` floors given three lists of constants at each floor `m`, `b` and `k`.

```
function [Gu,Gw] = freq(m,b,k,omega,nG)
s = j*omega;
[mn,bn,kn,mS,bS,kS] = partition(m,b,k);
if nG == 1 then
   [Gu,Gw] = G1(mn,bn,kn,s);
   save(Gu,Gw,nG);
else
   [GuS,GwS] = freq(mS,bS,kS,omega,nG-1);
   [Gu,Gw] = Gr(mn,bn,kn,GuS,GwS,s);
   save(Gu,Gw,nG);
end if
```

**Algorithm 7** Pseudocode of computing the coefficient vectors `cNu`, `cDu` of $G_{n,u(s)}$ and `cNw`, `cDw` of $G_{n,w}(s)$ for a building with `nG` floors given three lists of constants at each floor `m`, `b` and `k`.

```
1: function [cNu,cDu,cNw,cDw] = tran(m,b,k,nG)
2: [mn,bn,kn,mS,bS,kS] = partition(m,b,k);
3: if nG == 1 then
4:    [cNu,cDu,cNw,cDw] = C1(mn,bn,kn);
5:    save(cNu,cDu,cNw,cDw,nG);
6: else
7:    [cNuS,cDuS,cNwS,cDwS] = tran(m,b,k,nG-1);
8:    [cNu,cDu,cNw,cDw] = Cr(mn,bn,kn,cNuS,cDuS,cNwS,cDwS);
9:    [cNu,cDu] = simplify(cNu,cDu);
10:   [cNw,cDw] = simplify(cNw,cDw);
11:   save(cNu,cDu,cNw,cDw,nG);
12: end if
```

where $m_1$, $b_1$ and $k_1$ correspond to `mn`, `bn` and `kn` in the `G1()` and `C1()` functions' input arguments, respectively. Therefore, for the coefficient vectors in the base case, the `C1()` function returns that

$$\begin{aligned}
\texttt{cNu} &= \begin{bmatrix} 1 \end{bmatrix}, \\
\texttt{cDu} &= \begin{bmatrix} m_1 & b_1 & k_1 \end{bmatrix}, \\
\texttt{cNw} &= \begin{bmatrix} b_1 & k_1 \end{bmatrix}, \\
\texttt{cDw} &= \begin{bmatrix} m_1 & b_1 & k_1 \end{bmatrix}.
\end{aligned}$$

To derive the recurrence formulas, assume the response of the subnetwork is known,

$$X_{n-1}(s) = G_{n-1,u}(s)\overline{U}(s) + G_{n-1,w}(s)W(s). \tag{5.2}$$

When a new floor is added on the top of that, the previously external force $\overline{U}(s)$ is now determined by the internal relation

$$\overline{U}(s) = (k_n + b_n s)(X_n(s) - X_{n-1}(s)). \tag{5.3}$$

Substituting Equation (5.3) into Equation (5.2) leads to

$$X_{n-1}(s) = \frac{G_{n-1,u}(s)(k_n + b_n s)}{1 + G_{n-1,u}(s)(k_n + b_n s)} X_n(s) + \frac{G_{n-1,w}(s)}{1 + G_{n-1,u}(s)(k_n + b_n s)} W(s). \quad (5.4)$$

In addition, at level $n$, Newton's second law results in

$$(m_n s^2 + b_n s + k_n) X_n(s) = U(s) + (b_n s + k_n) X_{n-1}(s). \quad (5.5)$$

Substituting Equation (5.4) into Equation (5.5) gives

$$\begin{aligned} &[m_n s^2 + b_n s + k_n + G_{n-1,u}(s) m_n s^2 (b_n s + k_n)] X_n(s) \\ &= [1 + G_{n-1,u}(s)(b_n s + k_n)] U(s) + G_{n-1,w}(s)(b_n s + k_n) W(s). \end{aligned}$$

As a result, the following two recurrence formulas can be concluded,

$$\begin{aligned} G_{r,u}(s) = G_{n,u}(s) &= \frac{1 + G_{n-1,u}(s)(b_n s + k_n)}{m_n s^2 + b_n s + k_n + G_{n-1,u}(s) m_n s^2 (b_n s + k_n)}, \\ G_{r,w}(s) = G_{n,w}(s) &= \frac{G_{n-1,w}(s)(b_n s + k_n)}{m_n s^2 + b_n s + k_n + G_{n-1,u}(s) m_n s^2 (b_n s + k_n)}, \end{aligned} \quad (5.6)$$

which are implemented in the `Gr()` function in Algorithm 6. To obtain the computations on coefficient vectors in the `Cr()` function in Algorithm 7, assume that

$$G_{n-1,u}(s) = \frac{N_{n-1,u}(s)}{D_{n-1,u}(s)} \text{ and } G_{n-1,w}(s) = \frac{N_{n-1,w}(s)}{D_{n-1,w}(s)}.$$

The aforementioned recurrence formulas in Equation (5.6) become

$$
\begin{aligned}
&G_{r,u}(s) = G_{n,u}(s) \\
&= \frac{D_{n-1,u}(s)D_{n-1,w}(s) + N_{n-1,u}(s)D_{n-1,w}(s)(b_n s + k_n)}{(m_n s^2 + b_n s + k_n)D_{n-1,u}(s)D_{n-1,w}(s) + N_{n-1,u}(s)D_{n-1,w}(s)m_n s^2(b_n s + k_n)}, \\
&G_{r,w}(s) = G_{n,w}(s) \\
&= \frac{N_{n-1,w}(s)D_{n-1,u}(s)(b_n s + k_n)}{(m_n s^2 + b_n s + k_n)D_{n-1,u}(s)D_{n-1,w}(s) + N_{n-1,u}(s)D_{n-1,w}(s)m_n s^2(b_n s + k_n)}.
\end{aligned}
$$

Therefore, the computations in the `Cr()` function are

$$
\begin{aligned}
&\texttt{cNu} = \texttt{cDuS} * \texttt{cDwS} \oplus \texttt{cNuS} * \texttt{cDwS} * \begin{bmatrix} \texttt{bn} & \texttt{kn} \end{bmatrix}, \\
&\texttt{cDu} = \begin{bmatrix} \texttt{mn} & \texttt{bn} & \texttt{kn} \end{bmatrix} * \texttt{cDuS} * \texttt{cDwS} \oplus \texttt{cNuS} * \texttt{cDwS} * \begin{bmatrix} \texttt{mn} \times \texttt{bn} & \texttt{mn} \times \texttt{kn} & 0 & 0 \end{bmatrix}, \\
&\texttt{cNw} = \texttt{cNwS} * \texttt{cDuS} * \begin{bmatrix} \texttt{bn} & \texttt{kn} \end{bmatrix}, \\
&\texttt{cDw} = \begin{bmatrix} \texttt{mn} & \texttt{bn} & \texttt{kn} \end{bmatrix} * \texttt{cDuS} * \texttt{cDwS} \oplus \texttt{cNuS} * \texttt{cDwS} * \begin{bmatrix} \texttt{mn} \times \texttt{bn} & \texttt{mn} \times \texttt{kn} & 0 & 0 \end{bmatrix},
\end{aligned}
$$

where $*$ is vector convolution and $\oplus$ is an operator for adding two coefficient vectors as defined in Section 2.3.2.1.

The undamaged constants in this chapter are assumed to be $m = 11.34 \times 10^3 kg$, $k = 560.4 \times 10^6 N/m$ with a damping ratio of 5%, *i.e.*,

$$b = 0.05 \times 2\sqrt{km} = 252.1 \times 10^3 Ns/m.$$

The above constants are from [53]. Note that the buildings in this chapter are all undamaged despite that their numbers of floors may vary.

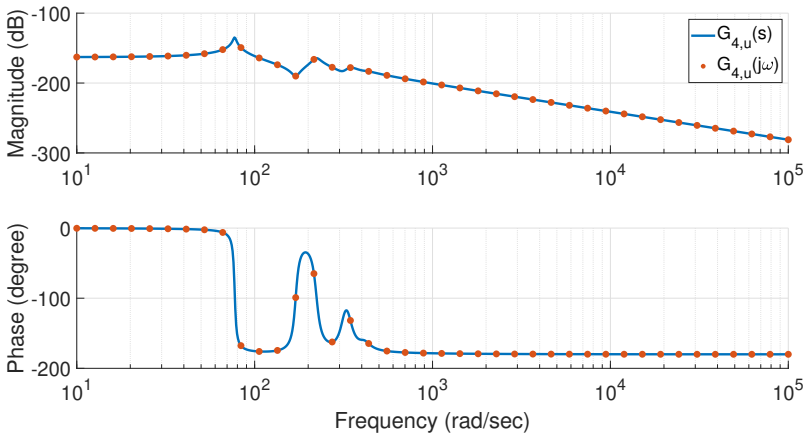


Figure 5.2. The consistency between the frequency response $G_{4,u}(j\omega)$ from Algorithm 6 and the transfer function $G_{4,u}(s)$ from Algorithm 7.

### 5.1.1 Correctness Verification

The verification is the same as that for finite networks in Section 2.3.3, whose goal is twofold. First, the transfer functions obtained from Algorithm 7 should be consistent with their corresponding frequency response given by Algorithm 6. That is confirmed in Figures 5.2 and 5.3 for a four-story building.

The second goal is to show the consistency between the time-domain response given by the transfer function from Algorithm 7 and that given by numerical integration. The time-domain response from the transfer function is obtained by using the `lsim()` function in `MATLAB`, while numerical integrations are performed by the `ode45()` function. The system of differential equations describing an $n$-story building's dynamics is

$$M\ddot{X} + B\dot{X} + KX = T \begin{bmatrix} u \\ w \end{bmatrix}, \tag{5.7}$$

where

$$X = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^\top,$$

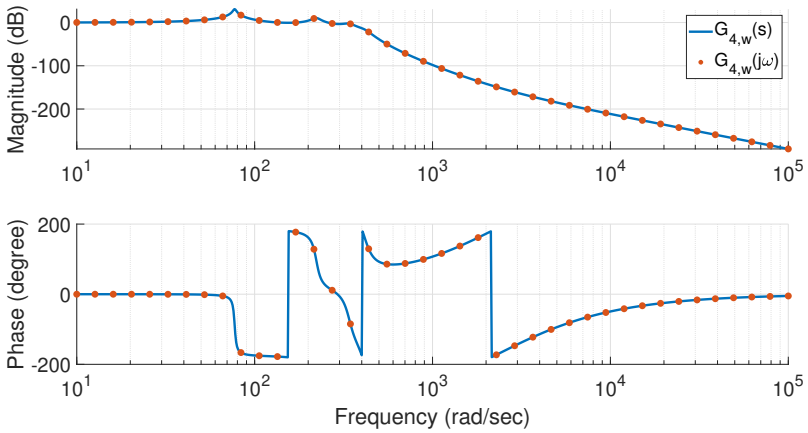


Figure 5.3. The consistency between the frequency response $G_{4,w}(j\omega)$ from Algorithm 6 and the transfer function $G_{4,w}(s)$ from Algorithm 7.

$$M = \begin{bmatrix} m_1 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & m_n \end{bmatrix},$$

$$B = \begin{bmatrix} b_1 + b_2 & -b_2 & 0 & 0 & 0 \\ -b_2 & b_2 + b_3 & -b_3 & \ddots & 0 \\ 0 & -b_3 & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & b_{n-1} + b_n & -b_n \\ 0 & 0 & 0 & -b_n & b_n \end{bmatrix},$$

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 & 0 & 0 & 0 \\ -k_2 & k_2 + k_3 & -k_3 & \ddots & 0 \\ 0 & -k_3 & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & k_{n-1} + k_n & -k_n \\ 0 & 0 & 0 & -k_n & k_n \end{bmatrix},$$

$$T = \begin{bmatrix} 0 & k_1 \\ 0_{(n-1)\times 1} & 0_{(n-1)\times 1} \\ 1 & 0 \end{bmatrix}.$$

That consistency is showcased by a four-story building here, with two groups of input signals:

$$\begin{cases} u(t) = \dfrac{10^6}{1+e^{-50(t-0.2)}}, \\ w(t) = 0, \end{cases} \quad \text{and} \quad \begin{cases} u(t) = 0, \\ w(t) = \dfrac{1}{1+e^{-50(t-0.2)}}. \end{cases}$$

Those nonzero input signals are logistic functions which are leveraged to avoid the sudden jump from 0 to 1 at time $t = 0$ of the classical unit-step input, so that the verification is mathematically valid. In addition, the input signal $u(t)$ is augmented by a factor of $10^6$ because the magnitude of the transfer function $G_{n,u}(s)$ is quite small, which can be observed from Figure 5.2. Finally, the consistency between the results from `lsim()` and `ode45()` functions is verified by Figures 5.4 and 5.5, which infers that frequency response and transfer functions from Algorithms 6 and 7 are correct.

## 5.2 A Robust Control Problem

This section puts forward and then solves a robust control problem regarding that multi-story building model in order to demonstrate that it is possible to find a unified controller for a dynamic network undergoing some extent of variations through its frequency response.

The block diagram of the controlled system is shown in Figure 5.6, where a unified controller $K$ is the design target for all $n$ in a predefined set $N$. Here, two closed-loop transfer functions are focused on. One transfer function is from the input $w$ of ground movement to the output $x_n$ of the displacement at the top floor,

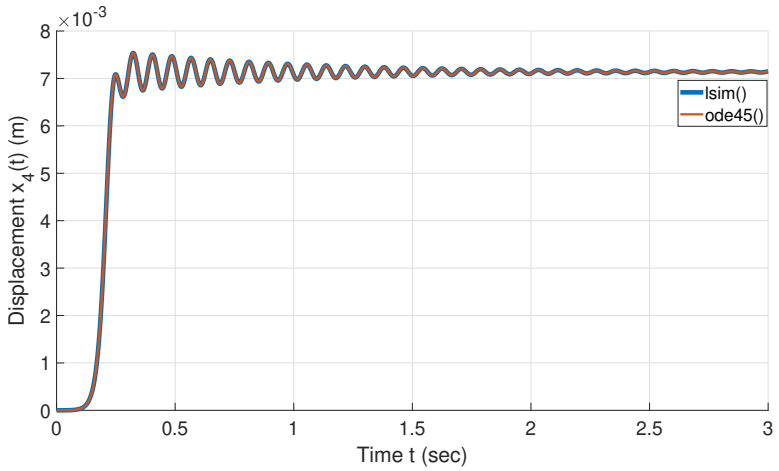


Figure 5.4. Step-like response of $x_4(t)$ for a four-story building when the inputs are $u(t) = 10^6/(1 + e^{-50(t-0.2)})$ and $w(t) = 0$.

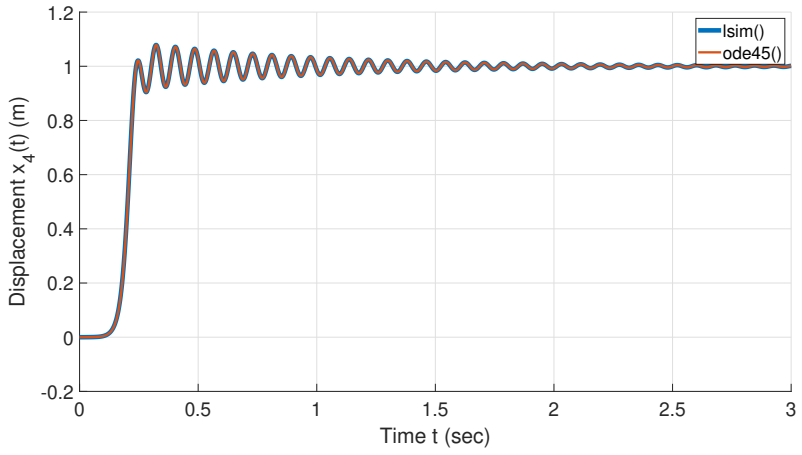


Figure 5.5. Step-like response of $x_4(t)$ for a four-story building when the inputs are $u(t) = 0$ and $w(t) = 1/(1 + e^{-50(t-0.2)})$.

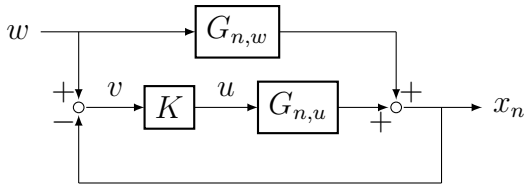


Figure 5.6. The block diagram of the controlled system.

$$X_n(s) = \frac{G_{n,w}(s) + G_{n,u}(s)K(s)}{1 + G_{n,u}(s)K(s)} W(s).$$

The other is the transfer function from the input $w$ to the error signal $v$,

$$V(s) = W(s) - X_n(s) = \frac{1 - G_{n,w}(s)}{1 + G_{n,u}(s)K(s)} W(s).$$

### 5.2.1 Problem Definition

A unified controller is required for that multi-story building model up to 10 stories. The Bode magnitude plots in Figure 5.7 confirm that for all $n \in N = \begin{bmatrix} 1 & 2 & \cdots & 10 \end{bmatrix}$, frequency response $G_{n,u}(j\omega)$ and $G_{n,w}(j\omega)$ form two sets of neighboring plants. Therefore, robust control methods can be applied here. Two requirements for that unified controller $K$ are established as follows.

1. Robust stability: For all $n \in N = \begin{bmatrix} 1 & 2 & \cdots & 10 \end{bmatrix}$, the closed-loop transfer function from the input $w$ to the output $x_n$,

$$\frac{X_n(s)}{W(s)} = \frac{G_{n,w}(s) + G_{n,u}(s)K(s)}{1 + G_{n,u}(s)K(s)}, \tag{5.8}$$

   is always stable.

2. Robust performance: For all $n \in N = \begin{bmatrix} 1 & 2 & \cdots & 10 \end{bmatrix}$ and $\omega \in \mathbb{R}^+$, the gain of the closed-loop transfer function from the input $w$ to the error $v$ always satisfies

$$\left| \frac{V(s)}{W(s)} \right| = \left| \frac{1 - G_{n,w}(s)}{1 + G_{n,u}(s)K(s)} \right| < 1.$$

Since this is an easy single-input single-output control problem with stable open-

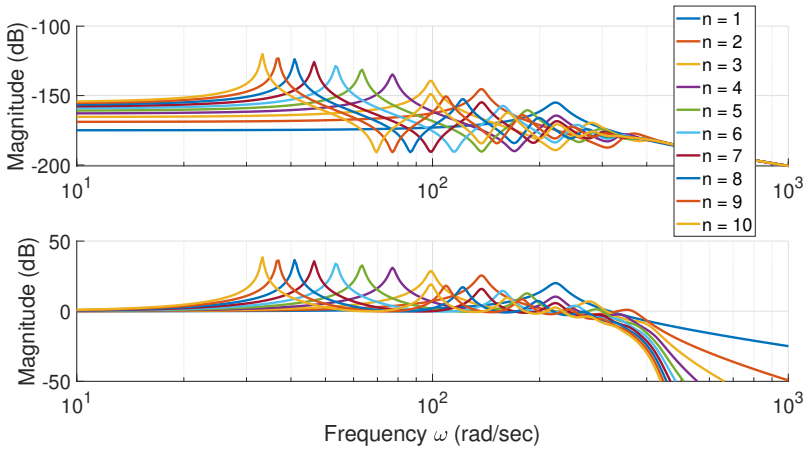


Figure 5.7. For all $n \in N = \begin{bmatrix} 1 & 2 & \cdots & 10 \end{bmatrix}$, frequency response $G_{n,u}(j\omega)$ and $G_{n,w}(j\omega)$ form two sets of neighboring plants. Upper: $G_{n,u}(j\omega)$. Lower: $G_{n,w}(j\omega)$.

loop plants, loop shaping techniques are going to be employed to achieve the goal. For the convenience of loop shaping, the above two requirements are recast more directly concerning the controller $K$.

First, for the robust stability requirement, the transfer functions $G_{n,u}(s)$, $G_{n,w}(s)$ and $K(s)$ can be rewritten as rational expressions where

$$G_{n,u}(s) = \frac{N_{n,u}(s)}{D_{n,u}(s)}, \; G_{n,w}(s) = \frac{N_{n,w}(s)}{D_{n,w}(s)}, \text{ and } K(s) = \frac{N_K(s)}{D_K(s)}. \tag{5.9}$$

Substituting Equation (5.9) into Equation (5.8) leads to

$$\frac{X_n(s)}{W(s)} = \frac{N_{n,w}(s)D_{n,u}(s)D_K(s) + N_{n,u}(s)N_K(s)D_{n,w}(s)}{D_{n,w}(s)(D_{n,u}(s)D_K(s) + N_{n,u}(s)N_K(s))}.$$

The above equation leads to the following equivalence.

Robust Stability

$$\Leftrightarrow \forall\, n \in N,\ \forall\, \mathrm{Re}[s] > 0,\ D_{n,w}(s)(D_{n,u}(s)D_K(s) + N_{n,u}(s)N_K(s)) \neq 0.$$

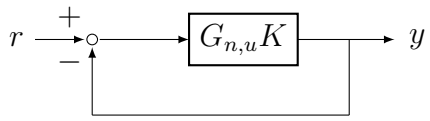


Figure 5.8. The block diagram of unity negative feedback where the open-loop transfer function is $G_{n,u}(s)K(s)$.

Moreover, note that the open-loop transfer function $G_{n,w}(s) = N_{n,w}(s)/D_{n,w}(s)$ is always stable for all $n \in N$, that is $\forall\ n \in N$, $\forall\ \mathrm{Re}[s] > 0$, $D_{n,w}(s) \neq 0$. Then, the above equivalence can be further simplified as

$$\text{Robust Stability} \Leftrightarrow \forall\ n \in N,\ \forall\ \mathrm{Re}[s] > 0,\ D_{n,u}(s)D_K(s) + N_{n,u}(s)N_K(s) \neq 0.$$

Note that the right-hand side is equivalent to requiring the transfer function

$$T(s) = \frac{G_{n,u}(s)K(s)}{1 + G_{n,u}(s)K(s)}$$

is always stable because $D_{n,u}(s)D_K(s) + N_{n,u}(s)N_K(s) = 0$ is the characteristic equation of $T(s)$. Furthermore, because that transfer function $T(s)$ is the complementary sensitivity function for the open-loop transfer function $G_{n,u}(s)K(s)$ under unity negative feedback as shown in Figure 5.8, it can be concluded that

$$\text{Robust Stability} \Leftrightarrow \forall\ n \in N,\ G_{n,u}(s)K(s) \text{ has sufficient gain and phase margins.}$$

For the robust performance requirement, the Cauchy-Schwarz inequality yields that

$$\left| \frac{1 - G_{n,w}(s)}{1 + G_{n,u}(s)K(s)} \right| \leq \frac{|1 - G_{n,w}(s)|}{|1 + G_{n,u}(s)K(s)|}.$$

Therefore, the robust performance requirement can be enforced by one of its sufficient

conditions, where

$$|1 + G_{n,u}(s)K(s)| > |1 - G_{n,w}(s)|.$$

In conclusion, that controller $K$ should satisfy the following three requirements.

1. Robust stability: For all $n \in N$, $G_{n,u}(s)K(s)$ has sufficient phase and gain margins.
2. Robust performance: For all $n \in N$ and $\omega \in \mathbb{R}^+$,

$$|1 + G_{n,u}(j\omega)K(j\omega)| > |1 - G_{n,w}(j\omega)|. \tag{5.10}$$

3. Because $U(s) = K(s)V(s)$, the magnitude of $K(s)$ should not be unnecessarily large in order to avoid large control inputs.

### 5.2.2 Loop Shaping

By comparing the magnitude of $G_{n,u}(j\omega)$ in Figure 5.9 to that of $1 - G_{n,w}(j\omega)$ in Figure 5.10, we see that a quite large gain needs to be added in order to fulfill the robust performance requirement in Equation (5.10). To make the loop shaping procedure convenient, an upper bound $H(j\omega)$ of $|1 - G_{n,w}(j\omega)|$ for all $n \in N$ and $\omega \in \mathbb{R}^+$ can be set, where

$$H(j\omega) = \frac{(j\omega + 1)^2(j\omega/2000 + 1)^2}{(j\omega/10 + 1)^2(j\omega/200 + 1)^2},$$

as illustrated by the red curve in Figure 5.10. As a result, the first candidate controller is selected as

$$K_1(s) = 10^{11.5}.$$

The corresponding Bode magnitude plot for $1 - G_{n,u}(j\omega)K_1(j\omega)$ is shown in Figure 5.11, where a drop exists around $5 \times 10^3 rad/sec$ violating the robust performance requirement in Equation (5.10). That can be repaired by adding a zero in that region. Another benefit brought by adding a zero in the high-frequency region can be seen from the Bode plot of $G_{n,u}(j\omega)$ in Figure 5.9. A high-gain controller like $K_1$ would

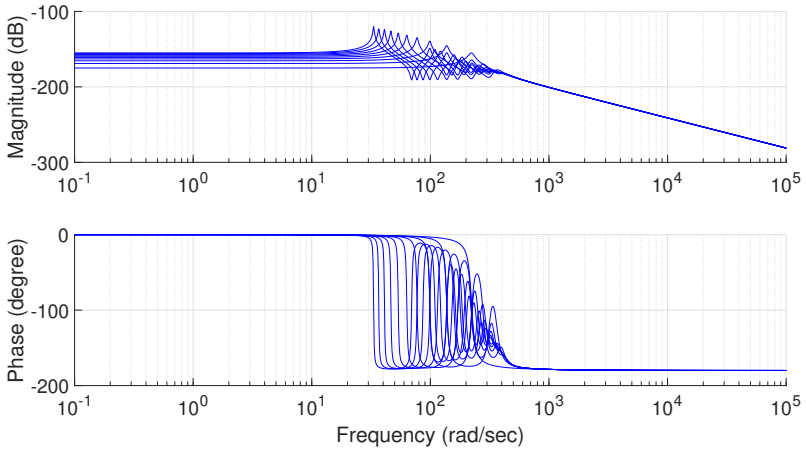


Figure 5.9. Bode plot of $G_{n,u}(j\omega)$ for all $n \in N$.

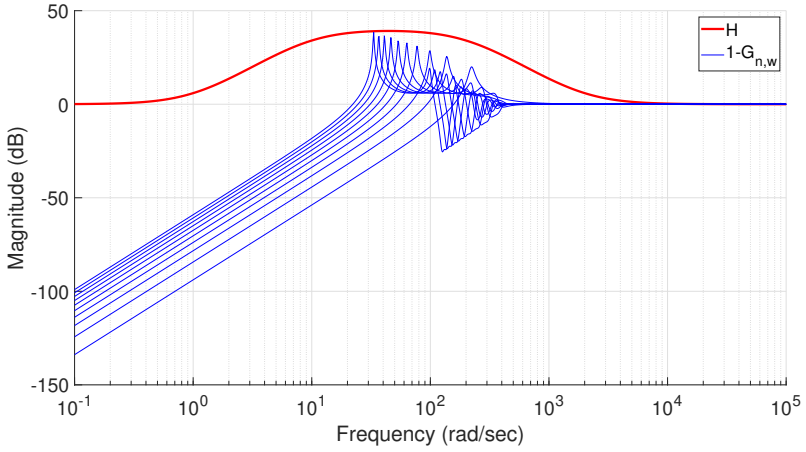


Figure 5.10. Bode magnitude plot of $1 - G_{n,w}(j\omega)$ for all $n \in N$. The red curve for $H(j\omega)$ is their upper bound.

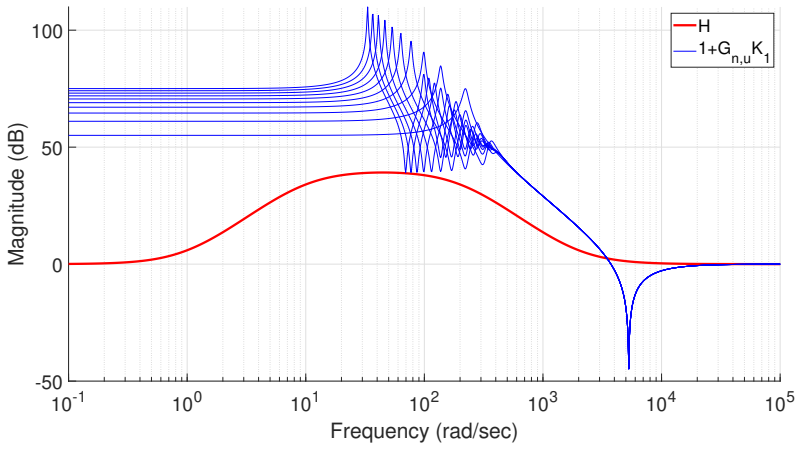


Figure 5.11. Bode magnitude plot of $1 + G_{n,u}(j\omega)K_1(j\omega)$ for all $n \in N$. The red curve for $H(j\omega)$ is same as the one in Figure 5.10.

push the gain crossover frequency to the high-frequency region where the phase of $G_{n,u}$ is about $-180^o$. Therefore, a zero in that region could bring a much larger phase margin to satisfy the robust stability requirement. Hence, a zero at $2000 rad/sec$ is added, so the next candidate controller is

$$K(s) = 10^{11.5}(s/2000 + 1), \tag{5.11}$$

which turns out to be the final design fulfilling all three requirements.

First, the robust stability criterion is illustrated in Figure 5.12, where all $G_{n,u}K$ have a phase margin about $80^o$. In addition, the phase of $G_{n,u}K$ never goes below $-180^o$, so the gain margin is always $\infty$. Second, for the robust performance requirement, Equation (5.10) is also fulfilled by the controller $K(s)$ as illustrated in Figure 5.13. Third, the small-control-input requirement is also satisfied since $|1 + G_{n,u}(s)K(s)|$ is tightly above $|H(s)|$ around the frequency where $|1 - G_{n,w}(s)|$ is just below $|H(s)|$. Any other controller whose magnitude is smaller than $|K(s)|$ is very likely to violate the robust performance requirement in Equation (5.10).

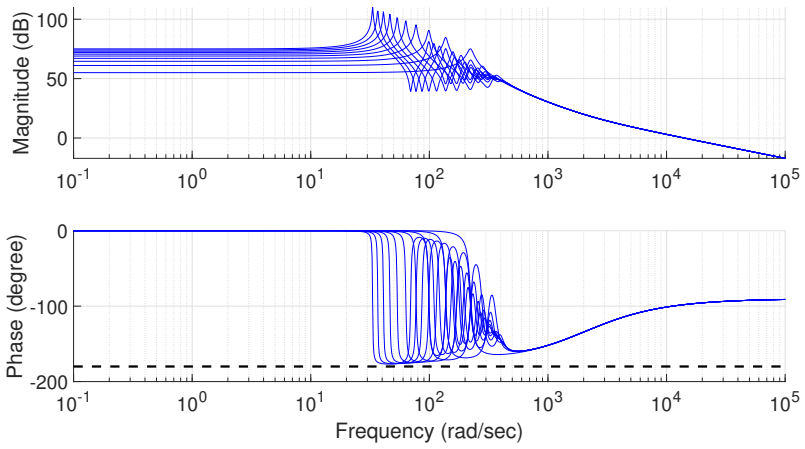


Figure 5.12. Bode plot of $G_{n,u}(j\omega)K(j\omega)$ for all $n \in N$.

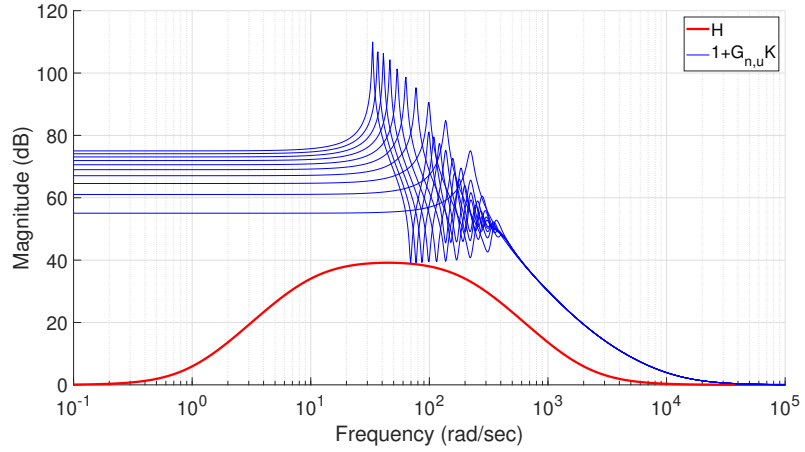


Figure 5.13. Bode magnitude plot of $1 + G_{n,u}(j\omega)K(j\omega)$ for all $n \in N$. The red curve for $H(j\omega)$ is same as the one in Figure 5.10.

## 5.3 Time-Domain Responses of The Controlled System

This section checks the time-domain responses given by that unified controller $K(s)$ in Equation (5.11). The confirmations of all $n \in N$ were conducted offline, whereas only the case of the 10-story building is displayed here. All time-domain responses in this section are obtained from the `ode45()`'s integrations of the equations of motion in Equation (5.7) (adapted to the 10-story case) with the control input $u(t)$ resulting from Equation (5.11),

$$u(t) = \frac{10^{11.5}}{2000}(\dot{w}(t) - \dot{x}_n(t)) + 10^{11.5}(w(t) - x_n(t)).$$

The first time-domain response is again excited by a logistic function $w(t)$ to form a continuous step response where

$$w(t) = \frac{1}{1 + e^{-50(t-0.2)}}. \tag{5.12}$$

A better performance given by the controlled system is shown in Figure 5.14, where the displacement of the top floor $x_{10}(t)$ follows the input ground movement $w(t)$ tightly, which leads to a much smaller overall vibration of the entire building as opposed to the uncontrolled system.

The second time-domain input imitates seismic waves, where the ground acceleration, as illustrated in Figure 5.15, is

$$\ddot{w}(t) = \left(1 - \frac{t}{50}\right) r, \tag{5.13}$$

in which $r$ is a normally distributed random number. A similar better performance can be observed in the response shown in Figure 5.16, where the controlled system has a smaller overall vibration compared to the uncontrolled one.

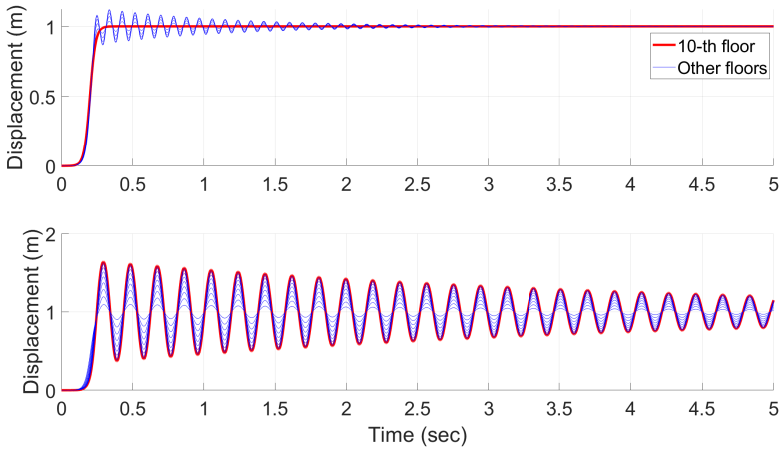


Figure 5.14. Step-like response to $w(t)$ in Equation (5.12) where the thick red curve is for the top floor $x_{10}(t)$ and all the other thin blue curves are for intermediate floors $x_1(t)$ to $x_9(t)$. Upper: Controlled. Lower: Uncontrolled.

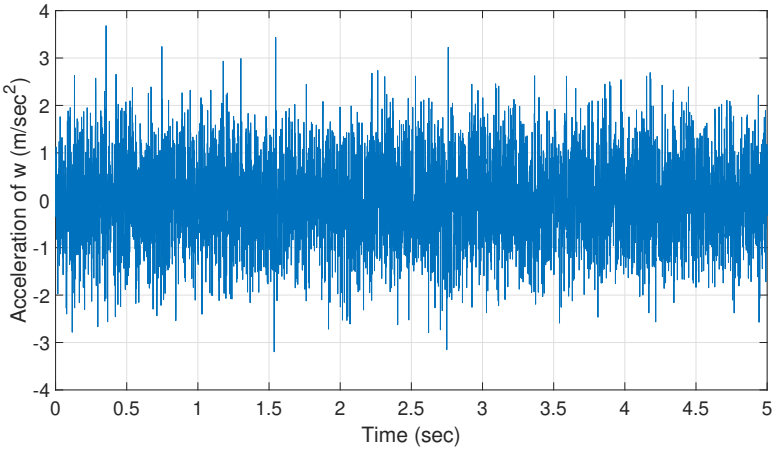


Figure 5.15. Simulated seismic wave $\ddot{w}(t)$ in Equation (5.13).

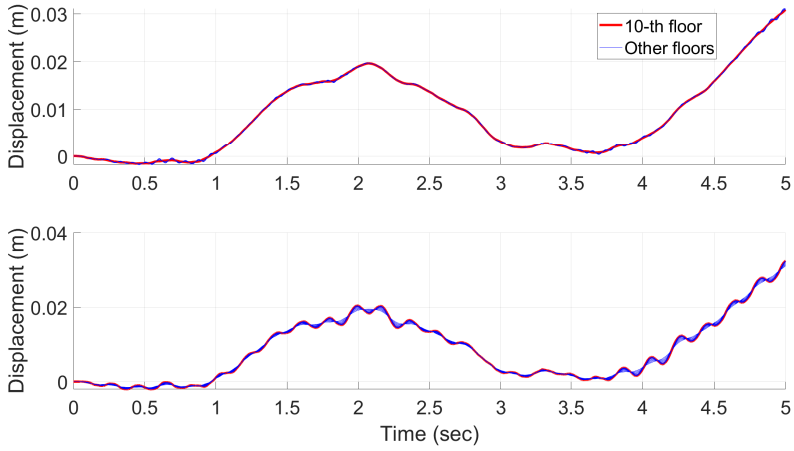


Figure 5.16. Seismic response to $\ddot{w}(t)$ in Equation (5.13) where the thick red curve is for the top floor $x_{10}(t)$ and all the other thin blue curves are for intermediate floors $x_1(t)$ to $x_9(t)$. Upper: Controlled. Lower: Uncontrolled.

## 5.4 Concluding Remarks

This chapter promotes the idea that dynamic networks may be viewed as a set of neighboring plants to which robust control methods can be applied. That neighboring effect is illustrated by their frequency responses which are obtained by Algorithm 6. Indeed, from an abstract viewpoint, the example presented in this chapter, a building with different numbers of floor, is an instance where a dynamic network's nodes and edge disappear and reappear during operation. That is one of main differences between dynamic networks and static ones. Another way that a network becomes dynamic could happen on the weights or the dynamics along the edges. In the context of buildings, that means the parametric variations occurring on masses, dampers and springs. As a concrete example, for the network model in Figure 5.1 with 4 floors, the spring's degradation on the second floor ($k_2$) happens while the building is in service, where its constant is multiplied by an uncertain factor ranging from 0.1 to 10. The effects of that parametric drift on the frequency response $G_{4,u}(j\omega)$ and $G_{4,w}(j\omega)$ are shown in Figure 5.17, from which we can observe that they also form a

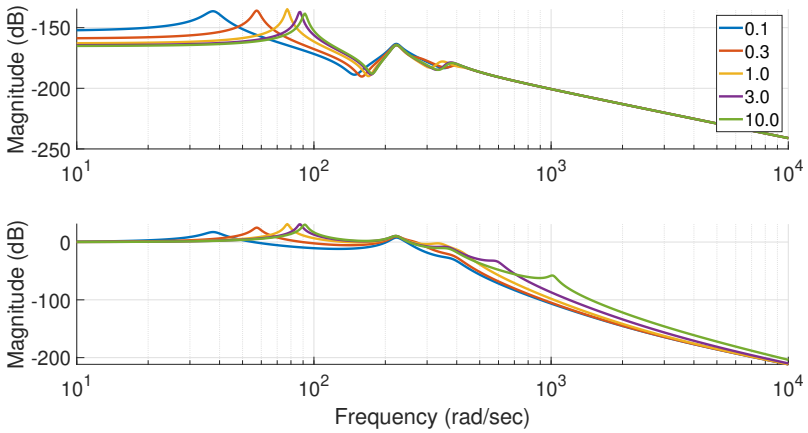


Figure 5.17. Effects of $k_2$'s parametric drifting from $0.1 \times 560.4 \times 10^6 N/m$ to $10 \times 560.4 \times 10^6 N/m$ on the frequency response. Upper: $G_{4,u}(j\omega)$. Lower: $G_{4,w}(j\omega)$.

set of neighboring plants.

That idea proposed in this chapter essentially relaxes the requirements imposed on dynamic network controllers from global stability to a stability only with respect to bounded variations. It is hoped that such relaxations can be leveraged to tackle more intricate dynamic networks' control problems in the future.

# CHAPTER 6

# RATIONAL APPROXIMATION

This chapter illustrates the application of the knowledge regarding transfer functions of dynamic networks to rational approximations, where some irrational functions are approximated by rational expressions. The idea is based on the fact that finite network transfer functions are always rational, while infinite network transfer functions are very likely to be fractional or irrational. Because a finite network's dynamics converge to its infinite variant's, that finite network's rational transfer function can be considered as an approximation of that infinite network's fractional or irrational transfer function.

The rational approximation results in this chapter are compared to those given by the Padé approximation, which is briefly reviewed in Section 6.1. Note that the rational approximation in this chapter has much greater restrictions as opposed to the Padé approximation. The main limitation is that the irrational expression that needs to be approximated must be a part of an infinite network's transfer function. It is also worth emphasizing that there exist algorithms for rational interpolation, which return a rational function interpolating a sequence of discrete datapoints.

## 6.1 Padé Approximation

For a function $f(x)$, if its Taylor series exists at $x = x_0$, it can be approximated by a finite power series whose highest order is assumed to be $L + M$, where $L,\ M \in \mathbb{N}$.

That is,

$$f(x) \approx A(x) = \sum_{i=0}^{L+M} a_i x^i.$$

Then, the Padé approximation with order $L/M$ of $f(x)$ at $x = x_0$ is defined as

$$R_{L/M}(x) = \frac{\sum_{i=0}^{L} p_i x^i}{1 + \sum_{i=1}^{M} q_i x^i} = A(x) = \sum_{i=0}^{L+M} a_i x^i, \tag{6.1}$$

where $p_i$ and $q_i$ are found by equating coefficients, which leads to a system of $L+M+1$ linear equations with $L+M+1$ unknowns. Therefore, all $p_i$ and $q_i$ should be obtained uniquely unless there exist some linearly dependent equations.

For example, $f(x) = \sqrt{x+1}$ needs to be estimated by Padé approximation with order $2/2$ at $x = 0$. First, $f(x)$ is approximated by a Taylor series up to the fourth order at $x = 0$ where

$$f(x) \approx 1 + \frac{1}{2}x - \frac{1}{8}x^2 + \frac{1}{16}x^3 - \frac{5}{128}x^4.$$

By equating coefficients, when $L = M = 2$, Equation (6.1) leads to a system of five linear equations

$$\begin{aligned}
p_0 &= a_0, \\
p_1 &= a_1 + a_0 q_1, \\
p_2 &= a_2 + a_1 q_1 + a_0 q_2, \\
0 &= a_3 + a_2 q_1 + a_1 q_2, \\
0 &= a_4 + a_3 q_1 + a_2 q_2.
\end{aligned}$$

Specifically for the example $f(x) = \sqrt{x+1}$,

$$\begin{aligned} p_0 &= 1, \\ p_1 &= \frac{1}{2} + q_1, \\ p_2 &= -\frac{1}{8} + \frac{1}{2}q_1 + q_2, \\ 0 &= \frac{1}{16} - \frac{1}{8}q_1 + \frac{1}{2}q_2, \\ 0 &= -\frac{5}{128} + \frac{1}{16}q_1 - \frac{1}{8}q_2. \end{aligned}$$

The solution to above system of equations is

$$p_0 = 1,\ p_1 = \frac{5}{4},\ p_2 = \frac{5}{16},\ q_1 = \frac{3}{4},\ q_2 = \frac{1}{16}.$$

Therefore, the Padé approximation with the order 2/2 of $f(x) = \sqrt{x+1}$ at $x = 0$ is

$$R_{2/2}(x) = \frac{1 + \frac{5}{4}x + \frac{5}{16}x^2}{1 + \frac{3}{4}x + \frac{1}{16}x^2} = \frac{16 + 20x + 5x^2}{16 + 12x + x^2}.$$

The comparison of $f(x) = \sqrt{x+1}$ and $R_{2/2}(x)$ is shown in Figure 6.1. In addition, the comparison when $x$ is an imaginary number, *i.e.*, $x = j\omega$ is plotted in Figure 6.2.

## 6.2 Rational Approximations through Dynamic Networks

This section illustrates how to use dynamic network transfer functions to obtain rational approximations for some irrational expressions. Recall that Chapter 2 shows that the transfer function of an infinite undamaged electrical ladder network, as

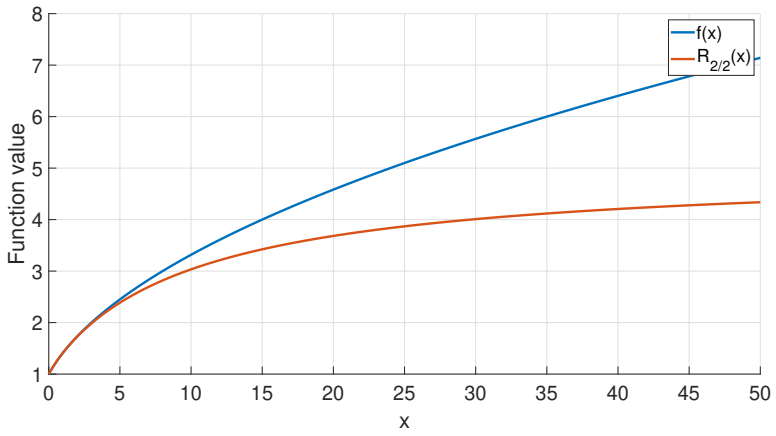


Figure 6.1. The comparison of an irrational function $f(x) = \sqrt{x+1}$ to its 2/2-order Padé approximation result at $x = 0$, $R_{2,2}(x)$.

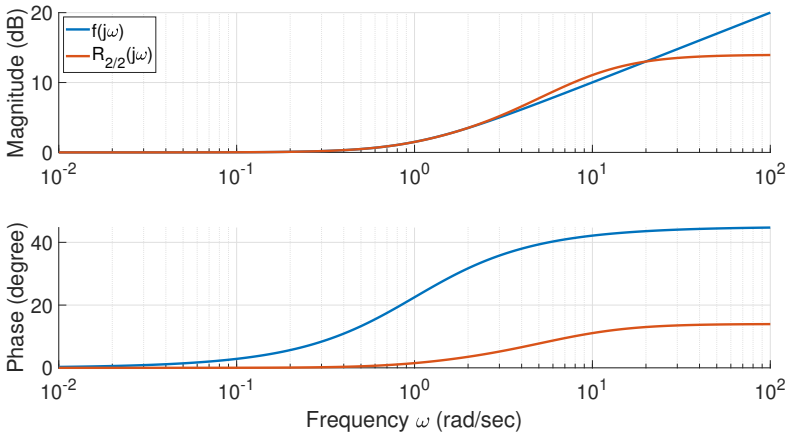


Figure 6.2. The comparison of an irrational function $f(j\omega) = \sqrt{j\omega+1}$ to its 2/2-order Padé approximation result at $j\omega = 0$, $R_{2,2}(j\omega)$, where $j\omega$ is an imaginary number.

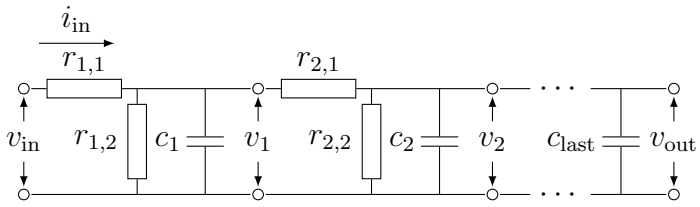


Figure 6.3. Electrical ladder network from Chapter 2.

shown in Figure 6.3, is

$$G_{\infty,\varnothing}(s) = \frac{s + \dfrac{1}{r_2 c} + \sqrt{s^2 + \dfrac{2r_1 + 4r_2}{r_1 r_2 c}s + \dfrac{r_1 + 4r_2}{r_1 r_2^2 c^2}}}{\dfrac{2}{r_1}s + \dfrac{2}{r_1 r_2 c}}.$$

Suppose now the irrational function $f(s) = \sqrt{s^2 + 100s + 100}$ needs to be estimated by rational functions, which is a part of the above $G_{\infty,\varnothing}(s)$ when

$$\frac{2r_1 + 4r_2}{r_1 r_2 c} = 100, \quad \text{and} \quad \frac{r_1 + 4r_2}{r_1 r_2^2 c^2} = 100.$$

To satisfy the above two equations, those three undamaged constants can be chosen as

$$r_1 = 1,\ r_2 = 5\sqrt{6} + 12,\ \text{and}\ c = \frac{5 + 2\sqrt{6}}{120 + 50\sqrt{6}}.$$

If the above undamaged constants are used to call the algorithm for finite network transfer functions in Section 2.3.2, the following convergence of finite undamaged

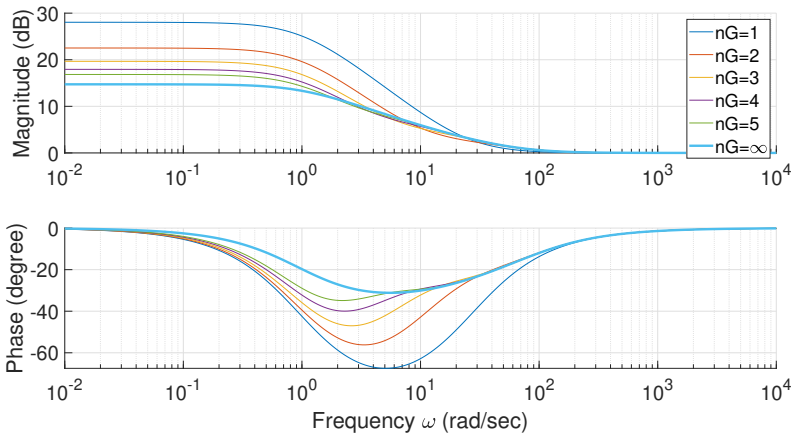


Figure 6.4. Convergence of finite undamaged electrical ladder networks' frequency response $G_{1,\varnothing}(j\omega), \ldots, G_{5,\varnothing}(j\omega)$ to its infinite variant's $G_{\infty,\varnothing}(j\omega)$.

electrical ladder networks' transfer functions can be obtained.

$$
\begin{aligned}
G_{1,\varnothing}(s) &= \frac{0.9899s + 25.247}{0.9899s + 1}, \\
G_{2,\varnothing}(s) &= \frac{s^2 + 75.505s + 675.26}{s^2 + 51.010s + 50.510}, \\
G_{3,\varnothing}(s) &= \frac{s^3 + 125.51s^2 + 3.9 \times 10^3 s + 1.8 \times 10^4}{s^3 + 101.01s^2 + 2.0 \times 10^3 s + 1.9 \times 10^3}, \\
G_{4,\varnothing}(s) &= \frac{s^4 + 175.51s^3 + 9.5 \times 10^3 s^2 + 1.7 \times 10^5 s + 5.2 \times 10^5}{s^4 + 151.01s^3 + 6.5 \times 10^3 s^2 + 7.1 \times 10^4 s + 6.6 \times 10^4}, \\
G_{5,\varnothing}(s) &= \frac{s^5 + 225.51s^4 + 1.8 \times 10^4 s^3 + 5.7 \times 10^5 s^2 + 6.5 \times 10^6 s + 1.5 \times 10^7}{s^5 + 201.01s^4 + 1.3 \times 10^4 s^3 + 3.3 \times 10^5 s^2 + 2.4 \times 10^6 s + 2.1 \times 10^6}, \\
&\vdots \\
G_{\infty,\varnothing}(s) &= \frac{s + 1.0102 + \sqrt{s^2 + 100s + 100}}{2s + 2.0204}.
\end{aligned}
$$

The convergence of the above transfer functions can be seen in Figure 6.4. As a result, an approximation of $f(s) = \sqrt{s^2 + 100s + 100}$ can be obtained from the above transfer functions,

$$H_g(s) = (2s + 2.0204)G_{g,\varnothing}(s) - (s + 1.0102),$$

where $g$ is a finite positive integer indicating the number of generations in the finite electrical ladder network that is used for approximation. Therefore, the following are the results of rational approximations of the irrational function $f(s) = \sqrt{s^2 + s + 100}$ given by undamaged electrical ladder networks' transfer functions.

$$
\begin{aligned}
H_1(s) &= \frac{0.9899s^2 + 50.495s + 50}{0.9899s + 1}, \\
H_2(s) &= \frac{s^3 + 101.01s^2 + 1.4 \times 10^3 s + 1.3 \times 10^3}{s^2 + 51.010s + 50.510}, \\
H_3(s) &= \frac{s^4 + 151.01s^3 + 5.9 \times 10^3 s^2 + 4.1 \times 10^4 s + 3.5 \times 10^4}{s^3 + 101.01s^2 + 2.0 \times 10^3 s + 1.9 \times 10^3}, \\
H_4(s) &= \frac{s^5 + 201.01s^4 + 1.3 \times 10^3 s^3 + 2.7 \times 10^5 s^2 + 1.2 \times 10^6 s + 9.8 \times 10^5}{s^4 + 151.01s^3 + 6.5 \times 10^3 s^2 + 7.1 \times 10^4 s + 6.6 \times 10^4}, \\
&\vdots \\
f(s) &= \sqrt{s^2 + 100s + 100}.
\end{aligned}
$$

The convergence of the above rational expressions $H_g(s)$ to the irrational expression $f(s)$ is plotted in Figure 6.5. In addition, that convergence when the independent variable $s = x$ is a real number is shown in Figure 6.6.

As a comparison, the following are the Padé approximations from 2/1-order to 6/5-order at $s = 0$ of the irrational expression $f(s) = \sqrt{s^2 + 100s + 100}$.

$$
\begin{aligned}
R_{2/1}(s) &= \frac{13s^2 + 100s + 100}{5s + 10}, \\
R_{3/2}(s) &= \frac{35s^3 + 570s^2 + 1500s + 1000}{19s^2 + 100s + 100}, \\
R_{4/3}(s) &= \frac{97s^4 + 2600s^3 + 12600s^2 + 2 \times 10^4 s + 10^4}{65s^3 + 630s^2 + 1500s + 10^3}, \\
R_{5/4}(s) &= \frac{275s^5 + 1.1 \times 1064s^4 + 8 \times 10^4 s^3 + 2.2 \times 10^5 s^2 + 2.5 \times 10^5 s + 10^5}{211s^4 + 3200s^3 + 13200s^2 + 2 \times 10^4 s + 10^4}, \\
&\vdots \\
f(s) &= \sqrt{s^2 + 100s + 100}.
\end{aligned}
$$

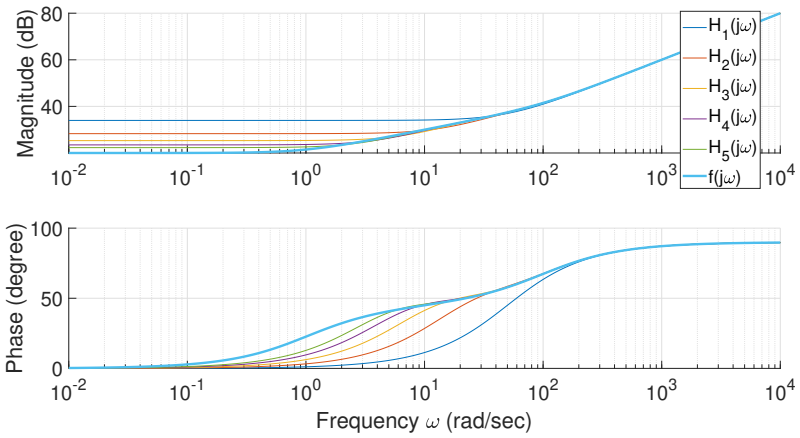


Figure 6.5. Rational approximations of $f(s) = \sqrt{s^2 + 100s + 100}$ given by undamaged electrical ladder networks' transfer functions when $s = j\omega$ is an imaginary number.

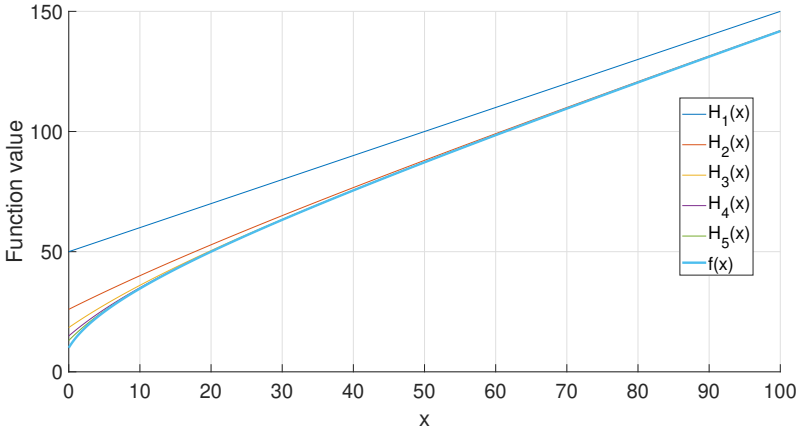


Figure 6.6. Rational approximations of $f(s) = \sqrt{s^2 + 100s + 100}$ given by undamaged electrical ladder networks' transfer functions when $s = x$ is a real number.

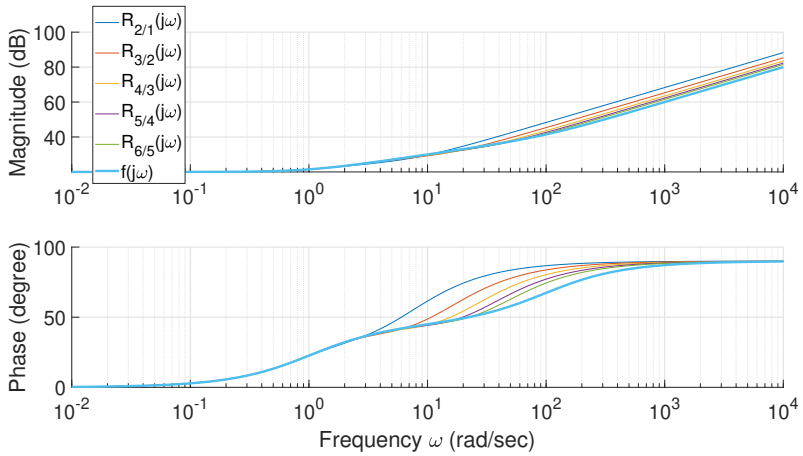


Figure 6.7. Padé approximations of $f(s) = \sqrt{s^2 + 100s + 100}$ at $s = 0$ when $s = j\omega$ is an imaginary number.

The convergence of the above Padé approximations $R_{L/M}(s)$ to the irrational expression $f(s)$ is plotted in Figure 6.7. In addition, that convergence when the independent variable $s = x$ is a real number is shown in Figure 6.8.

## 6.3 Concluding Remarks

This chapter explains how to use the knowledge regarding transfer functions of dynamic networks that satisfy assumptions (A-1) to (A-6) in Section 2.1 to obtain rational approximations of some irrational expressions. The idea leverages the fact that finite network transfer functions are always rational expressions, while infinite networks have a high possibility of being fractional or irrational. Then, because a finite network's dynamics converge to its infinite variant's, that finite network transfer function can be employed to estimate some fractional or irrational expressions that appear in the infinite network transfer function.

The idea is demonstrated by a concrete example, where the undamaged electrical ladder networks are utilized to find rational approximations of the irrational function $f(s) = \sqrt{s^2 + 100s + 100}$. In addition, the results are compared to those given by

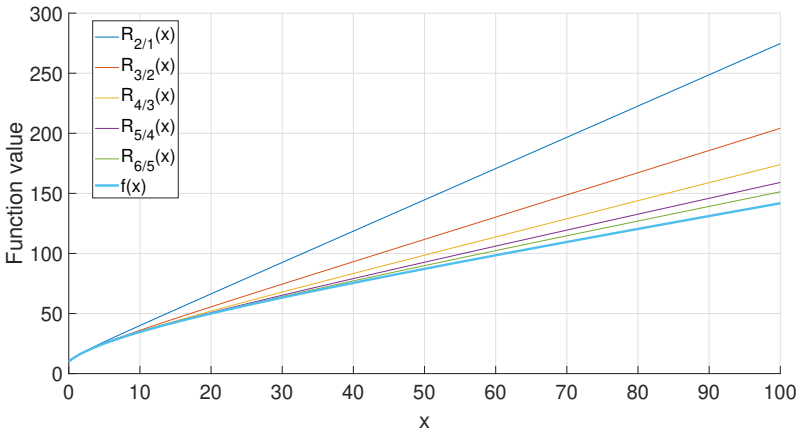


Figure 6.8. Padé approximations of $f(s) = \sqrt{s^2 + 100s + 100}$ at $s = 0$ when $s = x$ is a real number.

Padé approximations at $s = 0$ with the same orders. Both results are proved to be reasonable estimations of that irrational $f(s)$. However, there exist at least three drawbacks of the method using dynamic networks when compared to Padé approximations. First, the irrational expressions that can be approximated by the proposed method must appear in some infinite network transfer functions. That is a very demanding requirement because it eliminates lots of irrational functions. For example, currently, the author has no idea which infinite network transfer function could include exponential or trigonometric functions, while its finite version is still a rational expression. Second, the region where rational approximations converge to irrational functions is determined by the networks in use, and it cannot be directly specified by users. Third, the theoretical value of rate of convergence is also determined by the networks in use and seems hard to be derived rigorously through some mathematical methods. Nevertheless, that idea of using dynamic network transfer functions to find rational approximations is still mentioned here because that is a viable route and, to the author's knowledge, there are no similar methods in literature. Therefore, this method may have potential when studies regarding dynamic network frequency

response and transfer functions grow in the future.

# CHAPTER 7

# CONCLUSIONS AND SUGGESTED FUTURE WORK

Knowledge about networked dynamical systems' frequency response can be a new breakthrough in research about understanding and simulating complex systems' behavior, monitoring their health and controlling their functioning thanks to plentiful frequency-domain tools available. Chapter 2 lays the foundation of this dissertation and proposes recursive algorithms of exactly computing frequency response and transfer functions for a general class of self-similar dynamic networks, which leads to the four applications described by the following chapters. Chapter 3 illustrates the utility of that knowledge to simulating voltage and current along a transmission line whose electrical properties are unevenly distributed. That is then applied to railway track circuits to assess how voltage and current would vary when degradations occur and when a train is passing by. Chapter 4 presents health monitoring methods for dynamic networks through their frequency response, which aims at identifying existence, location and extent of the damage state. That method could return the components which are the most likely to be damaged, or it could classify all components into two groups where one group includes all candidates that are likely to be damaged, and the other contains components that are more likely to be intact. Chapter 5 leverages robust control methods to design a unified controller for a dynamic network undergoing some degree of variations, which is based on the observation from Chapter 2 that a dynamic network's frequency response often forms a set of neighboring plants under various conditions. That is exemplified by devising a state-feedback control strategy for a multi-story building to alleviate its vibration in the face of ground movements.

Chapter 6 offers a new rational approximation method of some irrational expressions through dynamic networks' transfer functions.

The main contribution of this dissertation is to introduce the idea of simulating, monitoring and controlling networked dynamical systems through their frequency response. In literature, there exists little work which exactly computes frequency response and transfer functions for a general class of dynamic networks, mostly due to their complexity. As a result, there is a gap between abundant frequency-domain tools available and the fact that those frequency-domain methods are infrequently applied to dynamic networks. However, by taking advantage of self-similarity, this dissertation exhibits that problem indeed becomes tractable. That assumption regarding self-similarity is far from restrictive, because it only requires the structure of a network being invariant across all generations, while the dynamics of its individual components are still allowed to vary. In fact, many real networks are self-similar, especially those built with the purpose of imitating complex systems. For any self-similar dynamic networks that satisfy the assumptions (A-1) to (A-6) in Section 2.1, their frequency response and transfer functions can be evaluated by the algorithms proposed in this dissertation whether their sizes are finite or not and whether their components are identical or not. As a result, those dynamic networks can be further simulated, monitored and controlled by using some frequency-domain tools.

Another contribution of this dissertation is that it offers a bridge between complex systems and fractional or irrational transfer functions. This dissertation shows that infinite dynamic networks' transfer functions are very likely to be fractional or irrational, which is consistent with the existing literature regarding infinite dimensional systems. That might explain why complex system behavior is often difficult to be modeled by linear integer-order dynamics, because complex systems usually have a great number of internal nodes interacting with each other, which are equivalent to nearly infinite networks. In addition, this dissertation enables the potential

to physically realize fractional-order compensators by some finite electrical networks whose transfer functions converge to those fractional-order compensators' as their sizes grow. Finally, this dissertation may also help researchers to understand the physical meaning of non-integer-order transfer functions, especially the meaning of irrational transfer functions such as $G(s) = \sqrt{s+1}/\sqrt{s}$.

The rest of this chapter offers prospective future work following this dissertation. The ultimate goal is to make this work more applicable to real implementations. First, the modeling algorithms in this dissertation should be extended to higher-dimensional networks, which could lead to applications of simulating, monitoring and controlling an increasing number of real networks, like structures and materials, and multi-agent systems. Second, the simulation results and the performance of health monitoring procedures should be verified by experiments on hardware. In addition, the proposed health monitoring procedures could be incorporated with existing methods to improve the quality of their damage detection results. Third, the computation of networks' frequency response can collaborate with existing control strategy design routines to devise controllers for multi-agent systems through their frequency response.

## 7.1 Extending to 2D Networks

One crucial limitation of the algorithms computing frequency response and transfer functions in Chapter 2 is that they can only be applied to one-dimensional networks. For mechanical networks, that means all nodes can only move back and forth along one direction. For electrical networks, that means the current leaving the input end would eventually come back to the input end. That is, no currents could leak to the exterior through other ports except for the input end. That assumption is restrictive and inhibits the idea of promoting frequency-domain methods to more real-life networks. Hence, extending the modeling methods introduced in this dissertation to high-dimensional networks is important.

When those methods are extended to high-dimensional networks, one challenge is that those systems inevitably become multi-input multi-output, which results in that the number of outputs could grow with the size of those networks. Therefore, a proper definition of system outputs should be carefully determined so that the number of outputs is invariant with respect to a network's size. For a constant multi-output network, the route of deriving one-dimensional networks' frequency response and transfer functions used in this work is still valuable. The main concept is leveraging recursive algorithms, assuming all required results of subnetworks are available, and focusing on how those results would vary if an extra generation is added to those subnetworks. That way of thinking should to a great extent alleviate the burden of coding for high-dimensional networks' frequency response and transfer functions.

In the author's opinion, extending the modeling methods in this dissertation to merely two-dimensional networks would have already included a great number of real applications. For example, that could be employed to simulate the universal dielectric response of composite materials and to reproduce viscoelastic or aeroelastic behavior as mentioned in Section 1.2.1. As another example, recall that in Chapter 3, a one-dimensional electrical network is utilized to estimate voltage and current along a transmission line with unevenly distributed properties. The accurate values of that voltage and current are given by a partial differential equation where its spatial variable $x$ is one dimensional. Once the proposed modeling method is extend to two-dimensional, it should be applicable to approximating solutions to a partial differential equation where its spatial variable $x$ is two dimensional, for instance, a two-dimensional heat equation,

$$\frac{\partial u}{\partial t} = \alpha \left( \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} \right).$$

Then, that can be used to simulate how heat diffuses across a plane where thermal

conductivity is unevenly distributed.

Once the modeling methods in this dissertation are extended to two-dimensional cases, they can also be employed in controlling some multi-agent systems, such as a group of automated guided vehicles moving on the ground. For each topology that this group of agents exhibits, it should have a corresponding frequency response. Then, the next question is whether those frequency responses also form a set of neighboring plants. If that is the case, robust control methods can be applied as well to design control strategies for those agents when their formation is varying. From a high-level viewpoint, that could be a concrete example of incorporating multi-agent system control strategy with their topology, which is one of current research focuses regarding multi-agent systems.

## 7.2 Hardware Experiments

All results and examples in this dissertation are based on simulations, so hardware experiments are another necessary direction for future work. The directions of hardware experiments could be at least fourfold. From the most basic aspect, frequency response measurements from hardware should be consistent with those obtained by the algorithms from this dissertation. Perhaps the simplest way is using electrical networks whose frequency response can be measured by spectrum analyzers. It is worth pointing out that the correctness of frequency response and transfer functions given by the algorithms in this dissertation has already been verified. For instance, finite networks' frequency response converge to infinite networks', time-domain response given by those frequency response agree with the results from numerical integrations, and the discrete approximations of voltage and current along a transmission line are compatible with the continuous results derived from telegrapher's equations.

The second goal is more related to mechanical networks, which verifies if a mechanical network's frequency response given by the algorithms in this dissertation is

consonant with its experimental modal analysis. For instance, a structure can be approximated by a mechanical network as illustrated by Chapter 5, where $G(j\omega)$ is the frequency response of that network obtained by the algorithms in this dissertation. On the other hand, a modal analysis experiment can be conducted on that structure where the input can be excited by impact hammers or electrodynamic shakers and the output can be measured by accelerometers, lasers and string pots. That modal analysis experiment would provide quantities like resonances, damping and mode shapes regarding that structure's response. Those measurements should be consistent with the prediction given by that mechanical network's frequency response $G(j\omega)$.

The third purpose is to test the health monitoring procedure proposed in this dissertation on hardware. Recall that in Chapter 4, the measurement noise is added to the actual frequency response by a random level of relative differences where that randomness is uniformly distributed. That may not coincide with how actual measurement noise would present. In literature, real frequency response measurements often have less noise around the peaks of the large magnitude, while they are extremely noisy when the magnitude is small. Additionally, each spectrum analyzer has the smallest signal level that can be measured. Therefore, whether the proposed health monitoring still works with real measurements is of great interest.

The last motivation is to cooperate with existing health monitoring methods. As discussed in Section 1.2.3, from an abstract viewpoint, most health monitoring methods select features from data and use them to identify the damage state. Each type of hardware systems have their own features that are frequently used in their research domain. For example, the aforementioned resonances, damping and mode shapes are often extracted as features when monitoring structural health. For detecting anomalies in dynamic graphs, some time series of measurements like nodes and edge statuses or community groups are often employed. In this dissertation, a new feature is leveraged for the health monitoring purpose, that is the mismatch between

the measured frequency response and the computed one of a network. Therefore, it is possible that including that feature with other domain features used by existing health monitoring methods could improve the quality of their damage detection results.

## 7.3 Combine Controller Analysis with Controller Synthesis

Section 1.2.1 states that the computation of frequency response in this dissertation is equivalent to a controller analysis problem where the control strategy has already been defined as opposed to a controller synthesis problem. However, that does not necessarily mean the study of this work cannot contribute to controller synthesis. One idea which offers that possibility is similar to the D-K iteration where a robust control problem is solved by switching between controller synthesis and controller analysis iteratively. The eventual goal of D-K iteration is to minimize a value called $\mu$, which is a quantification of an uncertain plant's robust stability. In the K iteration, a controller synthesis problem is solved, which provides a candidate controller K. Then, that K is supplied to a controller analysis problem which offers a weighting matrix D so that the $\mu$ keeps decreasing for that given K.

That organic blend of controller synthesis and analysis can also be adopted here. For instance, some frequency-domain controller synthesis method could present a candidate controller. Then, the modeling methods proposed in this dissertation could analyze that controller's performance and suggest how that could be improved. The research regarding this topic could start with simple systems, such as single-input single-output systems, in which case, empirical or heuristic tuning is possible. Built upon that experience, the next task is probably quantification of a controller's performance and the rule of improving that. The aim of this stage is to achieve auto-tuning. Then, that would lead to further implementations on complex networks, such as multi-input multi-output systems. Those discoveries in whole have potentials

to introduce a new controller design strategy to multi-agent systems by using their frequency response.

## 7.4 Concluding Remarks

This dissertation has advocated concepts about simulating, monitoring and controlling networked dynamical systems through their frequency response and transfer functions. One mathematical novelty of this work is it provides specific examples, that is infinite dynamic networks, where fractional and irrational transfer functions naturally come to light. That offers possibilities for understanding the physical meaning of fractional-order derivatives and implicit operators in the future.

The other advantages come from all of three perspectives in simulation, monitoring and control. First, the proposed method is similar to finite element analysis, which could efficiently simulate some quantities spreading over complex systems that are typically described by partial differential equations. In addition to that efficiency, this method could also handle the situation where the physical properties of those complex systems are not distributed evenly.

For health monitoring, this dissertation suggests a new feature for candidate damage cases, that is the mismatch between their computed frequency response and a measured one. That new feature can identify the existence, location and extent of the damage state, and save the inspection team from examining all components inside a large network. Moreover, that feature has potentials to be included with other existing features in the health monitoring research field to improve the quality of damage detection results.

For controlling dynamic networks, this work creates a bridge between them and available frequency-domain tools. That connection is unclear before this dissertation due to the complexity of computing dynamic networks' frequency response, which is shown to be manageable when self-similarities are leveraged in this dissertation.

Moreover, when networks grow into higher dimensions, their frequency response inevitably depend on their topology. Hence, this study also has potentials to link multi-agent systems' formation with their controller design.

Perhaps some unexpected benefits could also result from this work because it is one of few initial studies touching the area of frequency response for a general class of dynamic networks. Hopefully, this study could help advance the intellectual frontier of humankind.

# BIBLIOGRAPHY


1. E. M. E. Ahmed and A. S. Elgazzar. On fractional order differential equations model for nonlocal epidemics. *Physica A: Statistical Mechanics and its Applications*, 379(2):607 – 614, 2007. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2007.01.010. URL http://www.sciencedirect.com/science/article/pii/S0378437107000623.

2. R. Albert, H. Jeong, and A.-L. Barabási. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, Jul 2000. ISSN 1476-4687. doi: 10.1038/35019019. URL https://doi.org/10.1038/35019019.

3. D. P. Almond, C. J. Budd, M. A. Freitag, G. W. Hunt, N. J. McCullen, and N. D. Smith. The origin of power-law emergent scaling in large binary networks. *Physica A: Statistical Mechanics and its Applications*, 392(4):1004 – 1027, 2013. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2012.10.035. URL http://www.sciencedirect.com/science/article/pii/S0378437112009363.

4. T. M. Apostol. *Calculus*. John Wiley & Sons, 1991.

5. R. Aragues, J. Cortes, and C. Sagues. Distributed consensus on robot networks for dynamically merging feature-based maps. *IEEE Transactions on Robotics*, 28(4):840–854, 2012. doi: 10.1109/TRO.2012.2192012.

6. T. M. Atanacković, S. Konjik, S. Pilipović, and D. Zorica. Complex order fractional derivatives in viscoelasticity. *Mechanics of Time-Dependent Materials*, 20 (2):175–195, Jun 2016. ISSN 1573-2738. doi: 10.1007/s11043-016-9290-3. URL https://doi.org/10.1007/s11043-016-9290-3.

7. M. J. Balas. Direct velocity feedback control of large space structures. *Journal of Guidance and Control*, 2(3):252–253, 1979. doi: 10.2514/3.55869. URL https://doi.org/10.2514/3.55869.

8. J. Bang-Jensen and G. Z. Gutin. *Digraphs: theory, algorithms and applications*. Springer-Verlag London, 2009. ISBN 978-1-84800-998-1. doi: 10.1007/978-1-84800-998-1.

9. A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999. ISSN 0036-8075. doi: 10.1126/science.286.5439.509. URL https://science.sciencemag.org/content/286/5439/509.

10. A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1):69 – 77, 2000. ISSN 0378-4371. doi: https://doi.org/10.1016/S0378-4371(00)00018-2. URL http://www.sciencedirect.com/science/article/pii/S0378437100000182.

11. A. C. Bartlett, C. V. Hollot, and H. Lin. Root locations of an entire polytope of polynomials: It suffices to check the edges. *Mathematics of Control, Signals and Systems*, 1(1):61–71, 1988.

12. A. Bejan. Constructal theory of pattern formation. *Hydrology and Earth System Sciences Discussions*, 11(2):753–768, Jan. 2007. URL https://hal.archives-ouvertes.fr/hal-00305050.

13. A. Bejan. *Entropy generation minimization: the method of thermodynamic optimization of finite-size systems and finite-time processes*. CRC press, 2013.

14. A. Bejan. *Advanced engineering thermodynamics*. Wiley, Newark, 4th ed.. edition, 2016. ISBN 9781119245964.

15. A. Bejan and S. Lorente. Constructal theory of generation of configuration in nature and engineering. *Journal of Applied Physics*, 100(4):041301, 2006. doi: 10.1063/1.2221896. URL https://doi.org/10.1063/1.2221896.

16. A. Bejan and G. W. Merkx. *Constructal theory of social dynamics*. Springer, 2007. doi: 10.1007/978-0-387-47681-0.

17. N. L. Biggs, E. K. Lloyd, and R. J. Wilson. *Graph theory 1736-1936*. Clarendon Press, Oxford, 1976. ISBN 0198539010.

18. G. Borino, M. Di Paola, and M. Zingales. A non-local model of fractional heat conduction in rigid bodies. *The European Physical Journal Special Topics*, 193 (1):173, Apr 2011. ISSN 1951-6401. doi: 10.1140/epjst/e2011-01389-y. URL https://doi.org/10.1140/epjst/e2011-01389-y.

19. S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, 1994.

20. C.-Y. Chen, Y. Zhao, J. Gao, and H. E. Stanley. Nonlinear model of cascade failure in weighted complex networks considering overloaded edges. *Scientific Reports*, 10(1):13428, Aug 2020. ISSN 2045-2322. doi: 10.1038/s41598-020-69775-5. URL https://doi.org/10.1038/s41598-020-69775-5.

21. F. Chen and W. Ren. On the control of multi-agent systems: A survey. *Foundations and Trends® in Systems and Control*, 6(4):339–499, 2019. ISSN 2325-6818. doi: 10.1561/2600000019. URL http://dx.doi.org/10.1561/2600000019.

22. J. Chen, C. Roberts, and P. Weston. Fault detection and diagnosis for railway track circuits using neuro-fuzzy systems. *Control Engineering Practice*, 16(5): 585 – 596, 2008. ISSN 0967-0661. doi: https://doi.org/10.1016/j.conengprac.2007.06.007. URL http://www.sciencedirect.com/science/article/pii/S0967066107001244.

23. L. Chen. Progress in study on constructal theory and its applications. *Science China Technological Sciences*, 55(3):802–820, Mar 2012. ISSN 1869-1900. doi: 10.1007/s11431-011-4701-9. URL https://doi.org/10.1007/s11431-011-4701-9.

24. S. Chen, F. Cerda, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovačević. Semi-supervised multiresolution classification using adaptive graph filtering with application to indirect bridge structural health monitoring. *IEEE Transactions on Signal Processing*, 62(11):2879–2893, 2014. doi: 10.1109/TSP.2014.2313528.

25. Y. Chen, I. Petráš, and D. Xue. Fractional order control - a tutorial. In *2009 American Control Conference*, pages 1397–1411, 2009. doi: 10.1109/ACC.2009.5160719.

26. Z. Chen, W. Hendrix, H. Guan, I. K. Tetteh, A. Choudhary, F. Semazzi, and N. F. Samatova. Discovery of extreme events-related communities in contrasting groups of physical system networks. *Data Mining and Knowledge Discovery*, 27 (2):225–258, Sep 2013. ISSN 1573-756X. doi: 10.1007/s10618-012-0289-3. URL https://doi.org/10.1007/s10618-012-0289-3.

27. H. Cheng, P.-N. Tan, C. Potter, and S. Klooster. Detection and characterization of anomalies in multivariate time series. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 413–424, 2009. doi: 10.1137/1.9781611972795.36. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611972795.36.

28. A. Chiuso and G. Pillonetto. A bayesian approach to sparse dynamic network identification. *Automatica*, 48(8):1553 – 1565, 2012. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2012.05.054. URL http://www.sciencedirect.com/science/article/pii/S0005109812002270.

29. H.-L. Choi, L. Brunet, and J. P. How. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 25(4):912–926, 2009. doi: 10.1109/TRO.2009.2022423.

30. F. Chung and L. Lu. The average distances in random graphs with given expected degrees. *Proceedings of the National Academy of Sciences*, 99(25): 15879–15882, 2002. ISSN 0027-8424. doi: 10.1073/pnas.252631999. URL https://www.pnas.org/content/99/25/15879.

31. J. Cortés, S. Martínez, T. Karataş, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004. doi: 10.1109/TRA.2004.824698.

32. G. Cottone, M. Di Paola, and M. Zingales. Fractional mechanical model for the dynamics of non-local continuum. In *Advances in numerical methods*, pages 389–423. Springer, 2009.

33. P. Crucitti, V. Latora, and M. Marchiori. Model for cascading failures in complex networks. *Physical Review E*, 69:045104, Apr 2004. doi: 10.1103/PhysRevE.69.045104. URL `https://link.aps.org/doi/10.1103/PhysRevE.69.045104`.

34. S. Das. *Functional Fractional Calculus for System Identification and Controls*. Springer-Verlag Berlin Heidelberg, 2008. ISBN 978-3-540-72703-3. doi: 10.1007/978-3-540-72703-3.

35. T. de Bruin, K. Verbert, and R. Babuška. Railway track circuit fault diagnosis using recurrent neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 28(3):523–533, 2017. doi: 10.1109/TNNLS.2016.2551940.

36. M. Di Paola, G. Failla, and M. Zingales. Physically-based approach to the mechanics of strong non-local linear elasticity theory. *Journal of Elasticity*, 97 (2):103–130, 2009.

37. T. C. Doehring, A. D. Freed, E. O. Carew, and I. Vesely. Fractional Order Viscoelasticity of the Aortic Valve Cusp: An Alternative to Quasilinear Viscoelasticity. *Journal of Biomechanical Engineering*, 127(4):700–708, 01 2005. ISSN 0148-0731. doi: 10.1115/1.1933900. URL `https://doi.org/10.1115/1.1933900`.

38. J. Doyle and G. Stein. Multivariable feedback design: Concepts for a classical/modern synthesis. *IEEE Transactions on Automatic Control*, 26(1):4–16, 1981. doi: 10.1109/TAC.1981.1102555.

39. J. Doyle, K. Glover, P. Khargonekar, and B. Francis. State-space solutions to standard H2 and H$\infty$ control problems. In *1988 American Control Conference*, pages 1691–1696, 1988. doi: 10.23919/ACC.1988.4789992.

40. J. C. Doyle. Analysis of feedback systems with structured uncertainties. *IEE Proceedings D - Control Theory and Applications*, 129(6):242–250, 1982. doi: 10.1049/ip-d.1982.0053.

41. G. E. Dullerud and F. Paganini. *A course in robust control theory: a convex approach*, volume 36. Springer Science & Business Media, 2013.

42. O. El-Khoury and H. Adeli. Recent advances on vibration control of structures under dynamic loading. *Archives of Computational Methods in Engineering*, 20 (4):353–360, 2013.

43. A. Entezami and H. Shariatmadar. An unsupervised learning approach by novel damage indices in structural health monitoring for damage localization and quantification. *Structural Health Monitoring*, 17(2):325–345, 2018. doi: 10.1177/1475921717693572. URL https://doi.org/10.1177/1475921717693572.

44. A. Entezami, H. Shariatmadar, and S. Mariani. Fast unsupervised learning methods for structural health monitoring with large vibration data from dense sensor networks. *Structural Health Monitoring*, 19(6):1685–1710, 2020. doi: 10.1177/1475921719894186. URL https://doi.org/10.1177/1475921719894186.

45. P. Erdős and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

46. N. Everitt, G. Bottegal, and H. Hjalmarsson. An empirical bayes approach to identification of modules in dynamic networks. *Automatica*, 91: 144 – 151, 2018. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2018.01.011. URL http://www.sciencedirect.com/science/article/pii/S0005109818300189.

47. C. R. Farrar and K. Worden. An introduction to structural health monitoring. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, 365(1851):303–315, 2007. ISSN 1364503X. URL http://www.jstor.org/stable/25190443.

48. C. R. Farrar and K. Worden. *Structural health monitoring: a machine learning perspective*. John Wiley & Sons, 2012.

49. J. A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004. doi: 10.1109/TAC.2004.834433.

50. P. A. Forero, A. Cano, and G. B. Giannakis. Consensus-based distributed support vector machines. *Journal of Machine Learning Research*, 11(5), 2010.

51. O. Frank and D. Strauss. Markov graphs. *Journal of the American Statistical Association*, 81(395):832–842, 1986. doi: 10.1080/01621459.1986.10478342. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1986.10478342.

52. M. I. Friswell. Damage identification using inverse methods. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):393–410, 2007. doi: 10.1098/rsta.2006.1930. URL https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2006.1930.

53. T. S. Fu and E. A. Johnson. Distributed mass damper system for integrating structural and environmental controls in buildings. *Journal of Engineering Mechanics*, 137(3):205–213, 2011. doi: 10.1061/(ASCE)EM.1943-7889.

0000211. URL https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29EM.1943-7889.0000211.

54. J. Goncalves and S. Warnick. Necessary and sufficient conditions for dynamical structure reconstruction of lti networks. *IEEE Transactions on Automatic Control*, 53(7):1670–1674, 2008. doi: 10.1109/TAC.2008.928114.

55. B. Goodwine. Modeling a multi-robot system with fractional-order differential equations. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1763–1768, 2014. doi: 10.1109/ICRA.2014.6907089.

56. M. Green. H$\infty$ controller synthesis by j-lossless coprime factorization. *SIAM Journal on Control and Optimization*, 30(3):522–547, 1992. doi: 10.1137/0330031. URL https://doi.org/10.1137/0330031.

57. M. Gudarzi. $\mu$-synthesis controller design for seismic alleviation of structures with parametric uncertainties. *Journal of Low Frequency Noise, Vibration and Active Control*, 34(4):491–511, 2015. doi: 10.1260/0263-0923.34.4.491. URL https://doi.org/10.1260/0263-0923.34.4.491.

58. C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden. Distributed regression: An efficient framework for modeling sensor network data. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, IPSN '04, page 1–10, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138466. doi: 10.1145/984622.984624. URL https://doi.org/10.1145/984622.984624.

59. A. Haber and M. Verhaegen. Subspace identification of large-scale interconnected systems. *IEEE Transactions on Automatic Control*, 59(10):2754–2759, 2014. doi: 10.1109/TAC.2014.2310375.

60. T. Hatanaka, X. Zhang, W. Shi, M. Zhu, and N. Li. Physics-integrated hierarchical/distributed hvac optimization for multiple buildings with robustness against time delays. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 6573–6579, 2017. doi: 10.1109/CDC.2017.8264650.

61. N. A. Heard, D. J. Weston, K. Platanioti, and D. J. Hand. Bayesian anomaly detection methods for social networks. *Ann. Appl. Stat.*, 4(2):645–662, 06 2010. doi: 10.1214/10-AOAS329. URL https://doi.org/10.1214/10-AOAS329.

62. N. Heymans and J.-C. Bauwens. Fractal rheological models and fractional differential equations for viscoelastic behavior. *Rheologica Acta*, 33(3):210–219, 1994. ISSN 1435-1528. doi: 10.1007/BF00437306. URL https://doi.org/10.1007/BF00437306.

63. R. J. Hill, D. C. Carpenter, and T. Tasar. Railway track admittance, earth-leakage effects and track circuit operation. In *Proceedings., Technical Papers Presented at the IEEE/ASME Joint Railroad Conference*, pages 55–62, 1989.

64. N. Hoang, Y. Fujino, and P. Warnitchai. Optimal tuned mass damper for seismic applications and practical design formulas. *Engineering Structures*, 30 (3):707 – 715, 2008. ISSN 0141-0296. doi: https://doi.org/10.1016/j.engstruct.2007.05.007. URL http://www.sciencedirect.com/science/article/pii/S0141029607001927.

65. Z. Huang, Y. Hwang, and R. Radermacher. Review of nature-inspired heat exchanger technology. *International Journal of Refrigeration*, 78:1 – 17, 2017. ISSN 0140-7007. doi: https://doi.org/10.1016/j.ijrefrig.2017.03.006. URL http://www.sciencedirect.com/science/article/pii/S0140700717301019.

66. B. A. Huberman. *The laws of the Web : patterns in the ecology of information*. MIT Press, Cambridge, Mass., 2001. ISBN 0262083035.

67. C. Ionescu and J. F. Kelly. Fractional calculus for respiratory mechanics: Power law impedance, viscoelasticity, and tissue heterogeneity. *Chaos, Solitons & Fractals*, 102:433 – 440, 2017. ISSN 0960-0779. doi: https://doi.org/10.1016/j.chaos.2017.03.054. URL http://www.sciencedirect.com/science/article/pii/S0960077917301157. Future Directions in Fractional Calculus Research and Applications.

68. A. Jadbabaie, Jie Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003. doi: 10.1109/TAC.2003.812781.

69. A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000. doi: 10.1109/34.824819.

70. J. C. Jensen, D. H. Chang, and E. A. Lee. A model-based design methodology for cyber-physical systems. In *2011 7th International Wireless Communications and Mobile Computing Conference*, pages 1666–1671, 2011. doi: 10.1109/IWCMC.2011.5982785.

71. A. K. Jonscher. The 'universal' dielectric response. *Nature*, 267(5613):673–679, Jun 1977. ISSN 1476-4687. doi: 10.1038/267673a0. URL https://doi.org/10.1038/267673a0.

72. A. K. Jonscher. Dielectric relaxation in solids. *Journal of Physics D: Applied Physics*, 32(14):R57–R70, jan 1999. doi: 10.1088/0022-3727/32/14/201. URL https://doi.org/10.1088%2F0022-3727%2F32%2F14%2F201.

73. D. Kagan, Y. Elovichi, and M. Fire. Generic anomalous vertices detection utilizing a link prediction algorithm. *Social Network Analysis and Mining*, 8 (1):27, Apr 2018. ISSN 1869-5469. doi: 10.1007/s13278-018-0503-4. URL https://doi.org/10.1007/s13278-018-0503-4.

74. S. Kar, J. M. F. Moura, and K. Ramanan. Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication. *IEEE Transactions on Information Theory*, 58(6):3575–3605, 2012. doi: 10.1109/TIT.2012.2191450.

75. H. R. Karimi, F. Palacios-Quiñonero, J. M. Rossell, and J. Rubió-Massegú. Sequential design of multioverlapping controllers for structural vibration control of tall buildings under seismic excitation. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 227(2):176–183, 2013. doi: 10.1177/0959651812464026. URL `https://doi.org/10.1177/0959651812464026`.

76. J. F. Kelly and R. J. McGough. Fractal ladder models and power law wave equations. *The Journal of the Acoustical Society of America*, 126(4):2072–2081, 2009. doi: 10.1121/1.3204304. URL `https://asa.scitation.org/doi/abs/10.1121/1.3204304`.

77. P. P. Khargonekar, I. R. Petersen, and K. Zhou. Robust stabilization of uncertain linear systems: quadratic stabilizability and H$\infty$ control theory. *IEEE Transactions on Automatic Control*, 35(3):356–361, 1990. doi: 10.1109/9.50357.

78. D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

79. V. Koltchinskii, C. T. Abdallah, M. Ariola, P. Dorato, and D. Panchenko. Improved sample complexity estimates for statistical learning control of uncertain systems. *IEEE Transactions on Automatic Control*, 45(12):2383–2388, 2000. doi: 10.1109/9.895579.

80. D. Koutra, E. E. Papalexakis, and C. Faloutsos. Tensorsplat: Spotting latent anomalies in time. In *2012 16th Panhellenic Conference on Informatics*, pages 144–149, 2012. doi: 10.1109/PCi.2012.60.

81. F. Kuhn and R. Oshman. Dynamic networks: models and algorithms. *ACM SIGACT News*, 42(1):82–96, 2011.

82. H. Kwakernaak. Robust control and H$\infty$-optimization—tutorial paper. *Automatica*, 29(2):255–273, 1993.

83. A. Lanzon and I. R. Petersen. Stability robustness of a feedback interconnection of systems with negative imaginary frequency response. *IEEE Transactions on Automatic Control*, 53(4):1042–1046, 2008. doi: 10.1109/TAC.2008.919567.

84. K. Leyden and B. Goodwine. Using fractional-order differential equations for health monitoring of a system of cooperating robots. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 366–371, 2016. doi: 10.1109/ICRA.2016.7487154.

85. K. Leyden and B. Goodwine. Fractional-order trajectory-following control for two-legged dynamic walking. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 699–704, 2018. doi: 10.1109/IROS.2018.8593749.

86. K. Leyden, M. Sen, and B. Goodwine. Models from an implicit operator describing a large mass-spring-damper network. *IFAC-PapersOnLine*, 51 (2):831 – 836, 2018. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2018.04.017. URL `http://www.sciencedirect.com/science/article/pii/S2405896318301459`. 9th Vienna International Conference on Mathematical Modelling.

87. K. Leyden, M. Sen, and B. Goodwine. Large and infinite mass–spring–damper networks. *Journal of Dynamic Systems, Measurement, and Control*, 141(6), Feb 2019. ISSN 0022-0434. doi: 10.1115/1.4042466. URL `https://doi.org/10.1115/1.4042466`. 061005.

88. C.-H. Loh, C. H. Mao, S.-H. Chao, and J.-H. Weng. Feature extraction and system identification of reinforced concrete structures considering degrading hysteresis. *Structural Control and Health Monitoring*, 17(7):712–729, 2010. doi: https://doi.org/10.1002/stc.405. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/stc.405`.

89. E. R. Love. Fractional derivatives of imaginary order. *Journal of the London Mathematical Society*, s2-3(2):241–259, 1971. doi: https://doi.org/10.1112/jlms/s2-3.2.241. URL `https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/jlms/s2-3.2.241`.

90. J. T. Machado, V. Kiryakova, and F. Mainardi. Recent history of fractional calculus. *Communications in Nonlinear Science and Numerical Simulation*, 16 (3):1140 – 1153, 2011. ISSN 1007-5704. doi: https://doi.org/10.1016/j.cnsns.2010.05.027. URL `http://www.sciencedirect.com/science/article/pii/S1007570410003205`.

91. R. L. Magin. *Fractional calculus in bioengineering*, volume 2. Begell House Redding, 2006.

92. B. B. Mandelbrot. *The fractal geometry of nature*, volume 480. WH freeman New York, 1982.

93. C. A. Marin and M. R. Errera. A comparison between random and deterministic tree networks for river drainage basins. In *2009 3rd Southern Conference on Computational Modeling*, pages 18–23, Nov 2009. doi: 10.1109/MCSUL.2009.11.

94. J. Mayes. *Reduction and approximation in large and infinite potential-driven flow networks*. PhD thesis, University of Notre Dame, 2012.

95. N. J. McCullen, D. P. Almond, C. J. Budd, and G. W. Hunt. The robustness of the emergent scaling property of random rc network models of complex materials. *Journal of Physics D: Applied Physics*, 42(6):064001, mar 2009. doi: 10.1088/0022-3727/42/6/064001. URL https://doi.org/10.1088%2F0022-3727%2F42%2F6%2F064001.

96. D. McFarlane and K. Glover. A loop-shaping design procedure using h/sub infinity / synthesis. *IEEE Transactions on Automatic Control*, 37(6):759–769, 1992. doi: 10.1109/9.256330.

97. A. Megretski and A. Rantzer. System analysis via integral quadratic constraints. *IEEE Transactions on Automatic Control*, 42(6):819–830, 1997. doi: 10.1109/9.587335.

98. F. Merrikh-Bayat and M. Afshar. Extending the root-locus method to fractional-order systems. *Journal of Applied Mathematics*, 2008:528934, Jun 2008. ISSN 1110-757X. doi: 10.1155/2008/528934. URL https://doi.org/10.1155/2008/528934.

99. S. Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.

100. R. J. Minnichelli, J. J. Anagnost, and C. A. Desoer. An elementary proof of kharitonov's stability theorem with extensions. *IEEE Transactions on Automatic Control*, 34(9):995–998, 1989. doi: 10.1109/9.35816.

101. B. Mirkin. *Clustering: a data recovery approach*. CRC Press, 2012.

102. P. Mlynek, J. Misurec, M. Koutny, and P. Silhavy. Two-port network transfer function for power line topology modeling. *Radioengineering*, 21(1), 2012.

103. C. A. Monje, B. M. Vinagre, V. Feliu, and Y. Chen. Tuning and auto-tuning of fractional order controllers for industry applications. *Control Engineering Practice*, 16(7):798 – 812, 2008. ISSN 0967-0661. doi: https://doi.org/10.1016/j.conengprac.2007.08.006. URL http://www.sciencedirect.com/science/article/pii/S0967066107001566.

104. D. A. Murio. Stable numerical evaluation of grünwald–letnikov fractional derivatives applied to a fractional ihcp. *Inverse Problems in Science and Engineering*, 17(2):229–243, 2009. doi: 10.1080/17415970802082872. URL https://doi.org/10.1080/17415970802082872.

105. K. D. Murphy, G. W. Hunt, and D. P. Almond. Evidence of emergent scaling in mechanical systems. *Philosophical Magazine*, 86(21-22):3325–3338, 2006. doi: 10.1080/14786430500197934. URL https://doi.org/10.1080/14786430500197934.

106. A. Nedic, A. Ozdaglar, and P. A. Parrilo. Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55(4):922–938, 2010. doi: 10.1109/TAC.2010.2041686.

107. M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003. doi: 10.1137/S003614450342480. URL https://doi.org/10.1137/S003614450342480.

108. M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Random graphs with arbitrary degree distributions and their applications. *Phys. Rev. E*, 64:026118, Jul 2001. doi: 10.1103/PhysRevE.64.026118. URL https://link.aps.org/doi/10.1103/PhysRevE.64.026118.

109. X. Ni and B. Goodwine. Damage modeling for the tree-like network with fractional-order calculus, 2020, arXiv preprint, 2012.09212 (*Accepted by MTNS 2020*).

110. X. Ni and B. Goodwine. Damage identification for the tree-like network through frequency-domain modeling, 2020, arXiv preprint, 2012.09234 (*Accepted by IFAC World Congress 2020*).

111. X. Ni and B. Goodwine. Damage modeling and detection for a tree network using fractional-order calculus. *Nonlinear Dynamics*, 101(2):875–891, Jul 2020. ISSN 1573-269X. doi: 10.1007/s11071-020-05847-5. URL https://doi.org/10.1007/s11071-020-05847-5.

112. X. Ni and B. Goodwine. Frequency response and transfer functions of large self-similar networks, 2020, arXiv preprint, 2010.11015.

113. X. Ni and B. Goodwine. Frequency response of transmission lines with unevenly distributed properties with application to railway safety monitoring, 2020, arXiv preprint, 2012.09247 (*Accepted by ECC 2021*).

114. W. Nick, J. Shelton, K. Asamene, and A. C. Esterline. A study of supervised machine learning techniques for structural health monitoring. *MAICS*, 1353:36, 2015.

115. A. Noordergraaf. *Circulatory system dynamics*. Biophysics and bioengineering series ; v. 1. Academic Press, New York, 1978. ISBN 0125209509.

116. R. Olfati-Saber. Distributed kalman filter with embedded consensus filters. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 8179–8184, 2005. doi: 10.1109/CDC.2005.1583486.

117. R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 2006. doi: 10.1109/TAC.2005.864190.

118. M. D. Ortigueira. Introduction to fractional linear systems. part 1: Continuous-time case. *IEE Proceedings - Vision, Image and Signal Processing*, 147:62–70(8), February 2000. ISSN 1350-245X. URL https://digital-library.theiet.org/content/journals/10.1049/ip-vis_20000272.

119. M. D. Ortigueira. Introduction to fractional linear systems. part 2: Discrete-time case. *IEE Proceedings - Vision, Image and Signal Processing*, 147:71–78(7), February 2000. ISSN 1350-245X. URL https://digital-library.theiet.org/content/journals/10.1049/ip-vis_20000273.

120. R. Pastor-Satorras and A. Vespignani. Immunization of complex networks. *Phys. Rev. E*, 65:036104, Feb 2002. doi: 10.1103/PhysRevE.65.036104. URL https://link.aps.org/doi/10.1103/PhysRevE.65.036104.

121. R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani. Epidemic processes in complex networks. *Rev. Mod. Phys.*, 87:925–979, Aug 2015. doi: 10.1103/RevModPhys.87.925. URL https://link.aps.org/doi/10.1103/RevModPhys.87.925.

122. I. R. Petersen. A stabilization algorithm for a class of uncertain linear systems. *Systems & Control Letters*, 8(4):351 – 357, 1987. ISSN 0167-6911. doi: https://doi.org/10.1016/0167-6911(87)90102-2. URL http://www.sciencedirect.com/science/article/pii/0167691187901022.

123. I. R. Petersen and D. C. McFarlane. Optimal guaranteed cost control and filtering for uncertain linear systems. *IEEE Transactions on Automatic Control*, 39(9):1971–1977, 1994. doi: 10.1109/9.317138.

124. I. R. Petersen and R. Tempo. Robust control of uncertain systems: Classical results and recent developments. *Automatica*, 50(5):1315 – 1335, 2014. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2014.02.042. URL http://www.sciencedirect.com/science/article/pii/S0005109814000806.

125. I. R. Petersen, B. D. O. Anderson, and E. A. Jonckheere. A first principles solution to the non-singular H$\infty$ control problem. *International Journal of Robust and Nonlinear Control*, 1(3):171–185, 1991. doi: https://doi.org/10.1002/rnc.4590010304. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.4590010304.

126. I. R. Petersen, V. A. Ugrinovskii, and A. V. Savkin. *Robust Control Design Using H$\infty$ Methods*. Springer Science & Business Media, 2012.

127. I. Podlubny. Fractional-order systems and PI$\lambda$D$\mu$-controllers. *IEEE Transactions on Automatic Control*, 44(1):208–214, 1999. doi: 10.1109/9.739144.

128. C. E. Priebe, J. M. Conroy, D. J. Marchette, and Y. Park. Scan statistics on enron graphs. *Computational & Mathematical Organization Theory*, 11(3): 229–247, 2005.

129. L. Qiu, B. Bernhardsson, A. Rantzer, E. J. Davison, and J. C. Doyle. A formula for computation of the real stability radius. *Institute for Mathematics and Its Applications Preprint Series # 1160*, 1993.

130. S. Rahili and W. Ren. Distributed continuous-time convex optimization with time-varying cost functions. *IEEE Transactions on Automatic Control*, 62(4): 1590–1605, 2017. doi: 10.1109/TAC.2016.2593899.

131. A. Ramanathan, P. K. Agarwal, M. Kurnikova, and C. J. Langmead. An online approach for mining collective behaviors from molecular dynamics simulations. *Journal of Computational Biology*, 17(3):309–324, 2010. doi: 10.1089/cmb.2009.0167. URL https://doi.org/10.1089/cmb.2009.0167.

132. S. Ranshous, S. Shen, D. Koutra, S. Harenberg, C. Faloutsos, and N. F. Samatova. Anomaly detection in dynamic networks: a survey. *WIREs Computational Statistics*, 7(3):223–247, 2015. doi: https://doi.org/10.1002/wics.1347. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.1347.

133. H. F. Raynaud and A. Zergaïnoh. State-space representation for fractional order controllers. *Automatica*, 36(7):1017 – 1021, 2000. ISSN 0005-1098. doi: https://doi.org/10.1016/S0005-1098(00)00011-X. URL http://www.sciencedirect.com/science/article/pii/S000510980000011X.

134. A. H. Reis. Constructal view of scaling laws of river basins. *Geomorphology*, 78 (3):201 – 206, 2006. ISSN 0169-555X. doi: https://doi.org/10.1016/j.geomorph.2006.01.015. URL http://www.sciencedirect.com/science/article/pii/S0169555X06000225.

135. W. Ren, R. W. Beard, and E. M. Atkins. A survey of consensus problems in multi-agent coordination. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 1859–1864 vol. 3, 2005. doi: 10.1109/ACC.2005.1470239.

136. G. Rossetti and R. Cazabet. Community discovery in dynamic networks: A survey. *ACM Comput. Surv.*, 51(2), Feb. 2018. ISSN 0360-0300. doi: 10.1145/3172867. URL https://doi.org/10.1145/3172867.

137. M. Rubenstein, A. Cabrera, J. Werfel, G. Habibi, J. McLurkin, and R. Nagpal. Collective transport of complex objects by simple robots: theory and experiments. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 47–54, 2013.

138. A. Rytter. *Vibrational Based Inspection of Civil Engineering Structures*. PhD thesis, Aalborg University, Denmark, 1993.

139. M. G. Safonov. Stability margins of diagonally perturbed multivariable feedback systems. *IEE Proceedings D - Control Theory and Applications*, 129(6):251–256, 1982.

140. B. M. Sanandaji, T. L. Vincent, and M. B. Wakin. A review of sufficient conditions for structure identification in interconnected systems*. *IFAC Proceedings Volumes*, 45(16):1623 – 1628, 2012. ISSN 1474-6670. doi: https://doi.org/10.3182/20120711-3-BE-2027.00254. URL http://www.sciencedirect.com/

science/article/pii/S147466701538188X. 16th IFAC Symposium on System Identification.

141. M. A. Sandidzadeh and M. Dehghani. Intelligent condition monitoring of railway signaling in train detection subsystems. *Journal of Intelligent & Fuzzy Systems*, 24:859–869, 2013. doi: 10.3233/IFS-2012-0604. URL https://doi.org/10.3233/IFS-2012-0604. 4.

142. L. Schenato and F. Fiorentin. Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor networks. *Automatica*, 47(9): 1878 – 1886, 2011. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2011.06.012. URL http://www.sciencedirect.com/science/article/pii/S0005109811003116.

143. P. Sen, S. Dasgupta, A. Chatterjee, P. A. Sreeram, G. Mukherjee, and S. S. Manna. Small-world properties of the indian railway network. *Phys. Rev. E*, 67:036106, Mar 2003. doi: 10.1103/PhysRevE.67.036106. URL https://link.aps.org/doi/10.1103/PhysRevE.67.036106.

144. D. Sierociuk and A. Dzieliński. Fractional kalman filter algorithm for the states, parameters and order of fractional system estimation. *Zielona Góra: Uniwersytet Zielonogórski*, 2006.

145. D. Sierociuk, A. Dzieliński, G. Sarwas, I. Petras, I. Podlubny, and T. Skovranek. Modelling heat transfer in heterogeneous media using fractional calculus. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1990):20120146, 2013. doi: 10.1098/rsta.2012.0146. URL https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2012.0146.

146. A. Simonetto and G. Leus. Distributed maximum likelihood sensor network localization. *IEEE Transactions on Signal Processing*, 62(6):1424–1437, 2014. doi: 10.1109/TSP.2014.2302746.

147. H. Sohn. Effects of environmental and operational variability on structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):539–560, 2007. doi: 10.1098/rsta.2006.1935. URL https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2006.1935.

148. D. P. Spanos, R. Olfati-Saber, and R. M. Murray. Distributed sensor fusion using dynamic consensus. In *IFAC World Congress*. Citeseer, 2005.

149. G. Stein and J. C. Doyle. Beyond singular values and loop shapes. *Journal of Guidance, Control, and Dynamics*, 14(1):5–16, 1991. doi: 10.2514/3.20598. URL https://doi.org/10.2514/3.20598.

150. J. Stelling, S. Klamt, K. Bettenbrock, S. Schuster, and E. D. Gilles. Metabolic network structure determines key aspects of functionality and regulation. *Nature*, 420(6912):190–193, Nov 2002. ISSN 1476-4687. doi: 10.1038/nature01166. URL https://doi.org/10.1038/nature01166.

151. D. Strauss. On a general class of models for interaction. *SIAM Review*, 28 (4):513–527, 1986. doi: 10.1137/1028156. URL https://doi.org/10.1137/1028156.

152. M. D. Todd, J. M. Nichols, S. T. Trickey, M. Seaver, C. J. Nichols, and L. N. Virgin. Bragg grating-based fibre optic sensors in structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):317–343, 2007. doi: 10.1098/rsta.2006.1937. URL https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2006.1937.

153. J. Toivola and J. Hollmén. Feature extraction and selection from vibration measurements for structural health monitoring. In N. M. Adams, C. Robardet, A. Siebes, and J.-F. Boulicaut, editors, *Advances in Intelligent Data Analysis VIII*, pages 213–224, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-03915-7.

154. N. W. Tschoegl. *The Phenomenological Theory of Linear Viscoelastic Behavior: An Introduction*. Springer-Verlag Berlin Heidelberg, 1989. ISBN 978-3-642-73602-5. doi: 10.1007/978-3-642-73602-5.

155. D. Valério and d. C. José Sá. *An introduction to fractional control*. IET control engineering series ; 91. Institution of Engineering and Technology, London, 2013. ISBN 9781849195461.

156. P. M. Van den Hof, A. Dankers, P. S. Heuberger, and X. Bombois. Identification of dynamic models in complex networks with prediction error methods—basic methods for consistent module estimates. *Automatica*, 49(10):2994 – 3006, 2013. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2013.07.011. URL http://www.sciencedirect.com/science/article/pii/S0005109813003592.

157. M. Vidyasagar. Randomized algorithms for robust controller synthesis using statistical learning theory. *Automatica*, 37(10):1515 – 1528, 2001. ISSN 0005-1098. doi: https://doi.org/10.1016/S0005-1098(01)00122-4. URL http://www.sciencedirect.com/science/article/pii/S0005109801001224.

158. V. A. Vyawahare and P. S. V. Nataraj. Fractional-order modeling of neutron transport in a nuclear reactor. *Applied Mathematical Modelling*, 37(23): 9747 – 9767, 2013. ISSN 0307-904X. doi: https://doi.org/10.1016/j.apm.2013.05.023. URL http://www.sciencedirect.com/science/article/pii/S0307904X13003442.

159. H. Wang, M. Tang, Y. Park, and C. E. Priebe. Locality statistics for anomaly detection in time series of graphs. *IEEE Transactions on Signal Processing*, 62 (3):703–717, 2014. doi: 10.1109/TSP.2013.2294594.

160. J. C. Wang. Realizations of generalized warburg impedance with RC ladder networks and transmission lines. *Journal of The Electrochemical Society*, 134 (8):1915–1920, aug 1987. doi: 10.1149/1.2100789. URL `https://doi.org/10.1149/1.2100789`.

161. S. Wang and C. Ran. Rethinking cellular network planning and optimization. *IEEE Wireless Communications*, 23(2):118–125, 2016. doi: 10.1109/MWC.2016.7462493.

162. Z. Wang, J. Guo, Y. Zhang, and R. Luo. Fault diagnosis for jointless track circuit based on intrinsic mode function energy moment and optimized ls-svm. In *2016 IEEE International Conference on High Voltage Engineering and Application (ICHVE)*, pages 1–4, 2016.

163. D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, Jun 1998. ISSN 1476-4687. doi: 10.1038/30918. URL `https://doi.org/10.1038/30918`.

164. H. Weerts. *Identifiability and identification methods for dynamic networks*. PhD thesis, Technische Universiteit Eindhoven, 2018.

165. K. Worden and G. Manson. The application of machine learning to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):515–537, 2007. doi: 10.1098/rsta.2006.1938. URL `https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2006.1938`.

166. J. Xu and C. R. Shelton. Intrusion detection using continuous time bayesian networks. *Journal of Artificial Intelligence Research*, 39:745–774, 2010.

167. C. Zhai, D. Hanaor, and Y. Gan. Universality of the emergent scaling in finite random binary percolation networks. *PLOS ONE*, 12(2):1–11, 02 2017. doi: 10.1371/journal.pone.0172298. URL `https://doi.org/10.1371/journal.pone.0172298`.

168. C. Zhao, D. Xue, and Y. Chen. A fractional order pid tuning algorithm for a class of fractional order plants. In *IEEE International Conference Mechatronics and Automation, 2005*, volume 1, pages 216–221 Vol. 1, 2005. doi: 10.1109/ICMA.2005.1626550.

169. L. Zhao, J. Guo, H. Li, and W. Liu. The simulation analysis of influence on jointless track circuit signal transmission from compensation capacitor based on transmission-line theory. In *2009 3rd IEEE International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications*, pages 1113–1118, Oct 2009. doi: 10.1109/MAPE.2009.5355926.

170. M. Zhao and V. Saligrama. Anomaly detection with score functions based on nearest neighbor graphs. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22, pages 2250–2258. Curran Associates, Inc., 2009. URL `https://proceedings.neurips.cc/paper/2009/file/996a7fa078cc36c46d02f9af3bef918b-Paper.pdf`.

171. K. Zhou, J. C. Doyle, and K. Glover. *Robust and optimal control*, volume 40. Prentice hall New Jersey, 1996.

*This document was prepared & typeset with* pdfLaTeX*, and formatted with* NDdiss2$_\varepsilon$ *classfile (v3.2017.2[2017/05/09])*