Chapter 1

Introduction

1.1 Background

An artificial neural network is a computational model designed to mimic the behavior of biological neural networks found in the human brain. It operates based on the fundamental principles observed in the function of biological neurons. This thesis explores the stability, synchronization, quasi-projective synchronization, function-projective Mittag-Leffler synchronization, and Lagrange synchronization of different integer and fractional order neural network models. Understanding the dynamics of neural networks is crucial for their practical application to real-world problems, such as associative memory, artificial intelligence, secure communication, pattern recognition, signal processing, and optimization problems [2, 3, 4, 5]. For instance, in the chaos synchronization of neural networks, the concept is applied in secure communication to enhance the confidentiality of signals transmitted between a sender and a receiver. This technique ensures robust encryption by exploiting the inherent chaotic dynamics of neural networks, thereby safeguarding sensitive information from unauthorized access or interception. In addressing synchronization issues

within neural networks, utilizing controllers derived from control theory becomes essential to stabilize error systems. Thus far, various controllers have been devised, such as integral sliding mode control [6], adaptive control [7], linear feedback control [8], and intermittent control [9]. Among these effective controllers, the linear feedback controller offers ease of design and cost-effective control calculation. However, nonlinear control techniques provide advantages over their linear counterparts by accommodating neural network systems' complex relationships and dynamics. In this thesis, alongside exploring linear feedback control, emphasis will be placed on investigating nonlinear control methodologies. The primary objective of this research is to delve into the dynamics of neural networks within different domains, including real, complex, quaternion, and hypercomplex domains, while employing different control strategies.

The introductory chapter starts by elucidating the core inspiration driving artificial neural networks, rooted in the biological structure of neurons. We will explore the structure of neurons, the transmission of neural signals, and the methodologies for modeling communication among these neurons. Following this exploration, the subsequent section introduces artificial neural networks and establishes essential mathematical principles that underpin the examination of neural network dynamics in the following chapters. In the final segment of this chapter, we introduced different definitions of synchronization and delineated its various forms.

1.2 Biological neurons

The human brain and nervous system comprise a vast network of interconnected cellular units, including nerve cells (neurons) and glial cells. Glial cells play a crucial role by providing physical and functional support to neurons. Despite neurons'

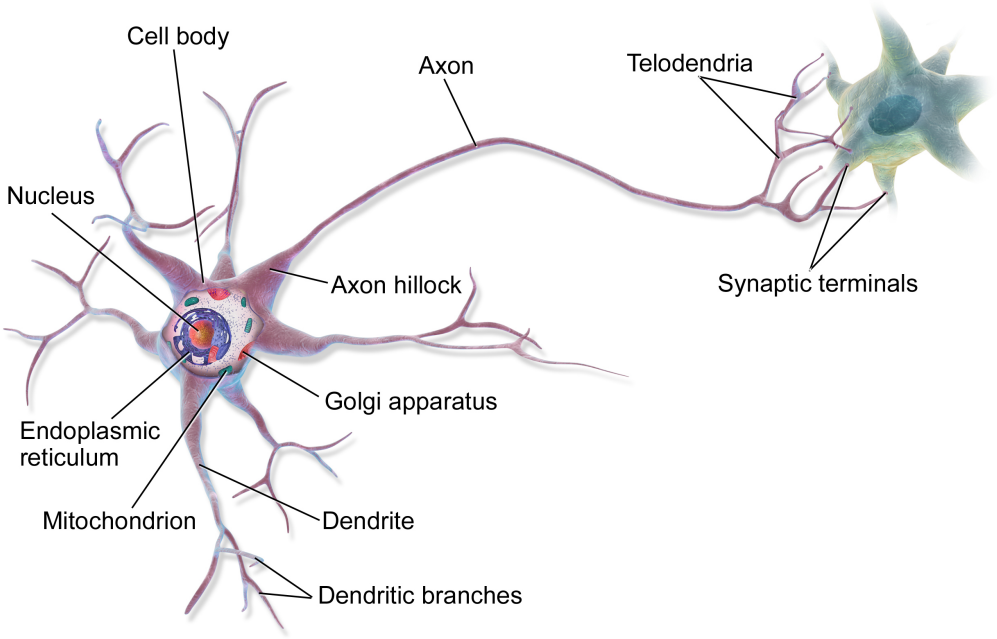

FIGURE 1.1: Schematic structure of a typical neuron.

diverse shapes, sizes, and locations, most share uniform structures and operate according to fundamental principles in transmitting neural signals. The essence of the brain lies in the interconnection of neurons and the transmission of electrochemical signals within them. These basic mechanisms underlie the brain's ability to perform complex tasks.

Figure 1.1 depicts a prototypical neuron [10]. The central region called the cell body or soma, contains the nucleus and various organelles essential for cellular function. Numerous dendrites extend from the cell body, which are root-like extensions and a single tubular fiber known as the axon. The axon typically branches into more minor extensions at its end. Dendrites, originating from the neuron cell body, are branch-like extensions that often exhibit extensive branching. Synapses, responsible for receiving signals, impulses, or information, are located on both the cell body and dendrites. Some neurons feature spines on their dendrites, creating specialized receiving sites. Since neurons integrate information from other neurons, the number

of inputs each receives significantly influences neural function. The axon, an elongated fiber-like extension, conducts signals, facilitating the transmission of impulses to other neurons or muscle fibers. This process, known as the action potential, involves the transmission of electrochemical signals through the axon. Neurons utilize two signaling mechanisms: electrical signals prevail within the neuron's interior, while chemical signals are employed at the synaptic terminals.

We will now delve into the process by which an electric current is generated across the axon membrane, its propagation, and the various mathematical models used to describe it. The neuron membrane consists of two thin lipid molecule layers, separating the axon's cytoplasm (interior) from the extracellular fluid. This membrane exhibits selectivity in allowing the diffusion of specific molecules, a property that can vary over time and along the length of the axon. The selective permeability of membranes is primarily facilitated by ion channels, which only allow specific types of ions to traverse the membrane according to their concentration and electrochemical gradients. The process responsible for transmitting signals within the axon is known as the action potential. Ion concentrations of potassium (K^+), sodium (Na^+), chloride (Cl^-), and calcium (Ca^{2+}), and the differences in these concentrations across the axon membrane, contribute to the generation of electrical potential within neurons. Under equilibrium conditions, the potential difference across the axon membrane typically measures around -70mV, a state referred to as the resting potential. During a neuron's inactive state, the distribution of ions across the membrane's thin layers results in the interior or cytoplasm of the neuron being negatively charged compared to the extracellular fluid. This polarization occurs due to the activity of a biological ion pump that becomes active when the interior of the axon becomes more positively charged. Introducing an electric pulse into the neuron's axon disrupts the membrane's resting potential, leading to ion concentration alterations and

subsequently changing the potential difference.

Therefore, the axon membrane demonstrates capacitance, which involves separating charges. Its operation resembles a resistor-capacitor (RC) circuit, a concept we will explore in the upcoming section. The membrane's permeability allows sodium ions (Na^+) to enter the neuron's interior through ion channels. This influx elevates the membrane voltage, and upon surpassing the threshold, the injected current generates a single pulse that propagates through the axon terminals. As the pulse travels along the axon terminals, it halts at the synaptic ends due to the synaptic gap. The signal is then relayed to the target neuron through a unique chemical process called synaptic transmission. During synaptic transmission, specific neurotransmitters are released from the synaptic ends and interact with the target neurons across the synaptic gap. Within the target neurons, the signal pulse acts as a current pulse, following the exact mechanism observed in the pre-synaptic neuron, thereby influencing other neurons.

1.2.1 Biological model

As mentioned above, neurons in the human brain communicate by transmitting electrical impulses along their axons to interact with other neurons. This inter-neuronal signaling is crucial for the brain to perform various cognitive tasks. To gain insights into brain function, it is essential to develop mathematical models that capture the behavior of biological neural networks. In this section, the task will be to delve into the general mathematical model of biological neural networks, initially proposed by Robert L. Harvey in his book "Neural Network Principles," published in 1994 [11]. Let's consider a network containing n neurons, depicted in Figure 1.2. These neurons are labelled as $u_1, u_2, ..., u_n$. Let us denote the state of the i-th

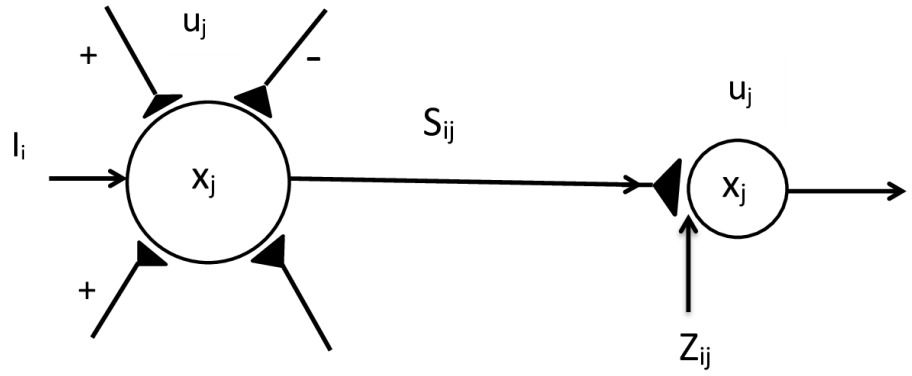

FIGURE 1.2: Schematic illustration of a neuron and a network: Neuron u_i, with a potential $x_i(t)$ relative to equilibrium and transmits a signal S_{ij} along its axon to a target neuron u_j. This signal influences the target neuron with a coupling strength Z_{ij}. The signal can have an excitatory effect ($Z_{ij} > 0$) or an inhibitory effect ($Z_{ij} < 0$). The external inputs I_i are stimuli for the neuron model.

neuron as $x_i(t)$ and represent the coupling strength between neurons u_i and u_j with the variable $Z_{ij}(t)$. More specifically,

$$x_i(t) = \text{the deviation of the } i\text{-th neuron from the resting potential.}$$

The variable $x_i(t)$ denotes the activation level of the i-th neuron, often referred to as axon potential or short-term memory (STM) trace. The interaction strength between neurons u_i and u_j is denoted by the variable $Z_{ij}(t)$, which can be either negative or positive, indicating inhibition or excitation of neurons to fire, respectively. When $Z_{ij}(t) > 0$, it signifies that the i-th neuron is being stimulated to transmit a signal to the j-th neuron. Conversely, if $Z_{ij}(t) < 0$, it suggests that the i-th neuron is being restrained from transmitting a signal to the j-th neuron.

$Z_{ij}(t) =$ the average release rate of neurotransmitter per unit axonal signal frequency.

This parameter is known as the synaptic coupling coefficient or long-term memory (LTM) trace. If there exists a variation in a neuron's potential from its equilibrium state due to internal and external processes within the neural network, the differential equation governing the rate of change in the neuron's potential is as follows.

$$\frac{dx_i(t)}{dt} = \left(\frac{dx_i(t)}{dt}\right)_{external} + \left(\frac{dx_i(t)}{dt}\right)_{internal}, \quad \forall i. \tag{1.1}$$

Suppose the inputs from other neurons and stimuli are additive. Then, we have

$$\begin{aligned}\frac{dx_i(t)}{dt} =& \left(\frac{dx_i(t)}{dt}\right)_{external} + \left(\frac{dx_i(t)}{dt}\right)_{excitatory} - \left(\frac{dx_i(t)}{dt}\right)_{inhibitory} \\ &+ \left(\frac{dx_i(t)}{dt}\right)_{internal}, \forall i.\end{aligned} \tag{1.2}$$

Moreover, assuming that the neuron's potential is exponentially decaying toward its equilibrium state in the absence of external processes, this can be expressed as follows:

$$\left(\frac{dx_i(t)}{dt}\right)_{internal} = -\alpha_i(x_i(t))x_i(t), \text{where} \quad \alpha_i(x_i(t)) > 0, \quad \forall i. \tag{1.3}$$

Suppose additive synaptic excitation is directly proportional to the pulse train frequency. Then we have

$$\left(\frac{dx_i(t)}{dt}\right)_{excitatory} \propto \sum_{other\ neurons} (\text{frequency of signal})(\text{synaptic coupling strengths}), \tag{1.4}$$

which can be expressed as

$$\left(\frac{dx_i(t)}{dt}\right)_{excitatory} = \sum_{\substack{l=1 \\ l\neq i}}^{n} S_{li}(t)Z_{li}(t), \quad \forall i. \tag{1.5}$$

Here, $S_{li}(t)$ denotes the average frequency of signals in the axon travelling from neuron u_l to u_i, evaluated at u_i. The average signal frequency $S_{li}(t)$ is influenced by two main factors: the propagation time delay τ_{li} for the signal to reach neuron u_i from u_l and the threshold value Γ_l for the firing of neuron u_l. This connection can be articulated as follows:

$$S_{li}(t) = f_l(x_l(t - \tau_{li}) - \Gamma_l), \tag{1.6}$$

where, $f_l : \mathbb{R} \to [0, \infty)$ denotes a particular non-negative function termed the signal function. Different variations of signal functions are frequently utilized in neural networks, and we will delve deeper into their specifics in subsequent discussions. Substituting (1.6) in equation (1.69), we obtain

$$\left(\frac{dx_i(t)}{dt}\right)_{excitatory} = \sum_{\substack{l=1 \\ l \neq i}}^{n} Z_{li}(t) f_l(x_l(t - \tau_{li}) - \Gamma_l), \quad \forall i. \tag{1.7}$$

Let's suppose that the inhibitory inputs from other neurons are hardwired, implying that the coupling strengths between the inhibited neurons remain constant. Within this framework, it can be stated that

$$\left(\frac{dx_i(t)}{dt}\right)_{inhibitory} = \sum_{\substack{l=1 \\ l \neq i}}^{n} C_{li}, \quad \forall i, \tag{1.8}$$

where $C_{li} = b_{li} h_l(x_l(t - \tau_{li}) - \Gamma_l)$, where $b_{li} \geq 0$ represents a constant coupling strength. The function h_l denotes a signal function, and the threshold value Γ_l is typically same across all neurons within the network.

The stimuli are external sources capable of altering the neuron's potential. Consequently,

$$\left(\frac{dx_i(t)}{dt}\right)_{external} = I_i, \quad \forall i. \tag{1.9}$$

By substituting the equations (1.6), (1.7), (1.8), and (1.9) in the equation (1.2), we obtain the so-called additive STM trace equation for $i = 1, 2, ..., n$ as

$$\frac{dx_i(t)}{dt} = -\alpha_i(x_i(t))x_i(t) + \sum_{\substack{l=1 \\ l\neq i}}^{n} Z_{li}(t) f_l(x_l(t-\tau_{li}) - \Gamma_l) - \sum_{\substack{l=1 \\ l\neq i}}^{n} b_{li} h_l(x_l(t-\tau_{li}) - \Gamma_l) + I_i. \tag{1.10}$$

Since we've considered that the excitatory synaptic coupling varies with time, the phenomenon can be expressed through the following equation, which is derived from Hebb's law.

$$\frac{dZ_{ij}(t)}{dt} = -A_{ij}(Z_{ij}(t))Z_{ij}(t) + P_{ij}(t)[x_j(t)]^+, \quad A_{ij}(Z_{ij}(t)) > 0, \quad \forall i, j, \tag{1.11}$$

where $P_{ij}(t) = \beta_{ij} f_i(x_i(t-\tau_{ij}) - \Gamma_i)$, $\beta_{ij} \geq 0$, and

$$[x_j(t)]^+ = \begin{cases} x_j(t), & \text{if } x_j \geq 0, \\ 0, & \text{if } x_j < 0. \end{cases}$$

The second component of equation (1.11) suggests that for the enhancement of $Z_{ij}(t)$, it is necessary for the presynaptic neuron u_i to convey a signal $P_{ij}(t)$ to the post-synaptic neuron u_j. Concurrently, neuron u_j needs to be activated, which implies that $x_j(t) > 0$. Hebb's law, in this context, posits that the coupling strength $Z_{ij}(t)$ increases if the presynaptic neuron u_i is active, and it successfully influences the post-synaptic neuron u_j. This adaptation in coupling strength as a function of

the post-synaptic neuron's activity is a crucial aspect of the learning phase, reflecting how synaptic connections are strengthened based on the frequency and impact of neuronal interactions.

The equations (1.10) and (1.11) for short-term memory (STM) and long-term memory (LTM) traces, respectively, cannot be solved until the values of the coefficients $\alpha_i, A_{ij}, C_{li}, S_{li}, P_{ij}$ and the external stimuli I_i are provided.

1.3 Artificial neural network

An artificial neural network (ANN) is a computational framework designed to mimic the human brain's cognitive processes during specific tasks or functions. Typically, this network is realized using electronic components or is simulated through software on a digital computer based on the comprehensive work of Haykin (2004) [12]. It comprises computational units representing neurons and interconnections representing synapses. The design of artificial neurons and their synaptic connections is derived from the model of biological neurons, reflecting the brain's architecture. Table 1.1 illustrates the distinctions between an artificial neural network and its biological counterpart. The definition of an ANN, as articulated by Simon Haykin in his book "Neural Network: A Comprehensive Foundation" [12], is outlined as follows:

Definition 1.3.1. A neural network is a highly parallel distributed processor composed of simple processing units possessing an innate capacity for experiential learning and knowledge utilization. In two key aspects, it mirrors the functioning of the brain:

Parameters	Artificial neural networks	Biological neural networks
Structure	input hidden layer weight output Bias	dendrites synapse axon cell body external stimulus
Processing	Faster processing speed Serial processing	Slower processing speed massively parallel processing
Computing	centralized sequential stored programs	distributed parallel self-learning

TABLE 1.1: Comparison between artificial and biological neural network.

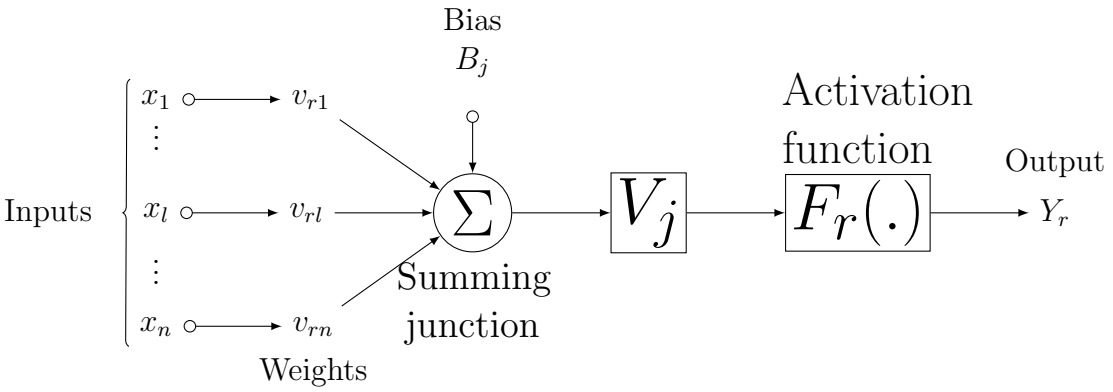

FIGURE 1.3: Fundamental diagram of an artificial neuron of the network.

(i) Neural networks assimilate knowledge from their environment through a dynamic process of learning.

(ii) Synaptic weights, representing the strengths of interconnections within the neural network, serve as repositories for acquired knowledge.

Neural networks are robust computing machines due to their distributed architecture and ability to learn and generalize. Generalization means providing reasonable outputs for inputs not encountered during learning. A neural network solves complicated problems in an integrated manner. It cannot function independently. Figure 1.3 depicts the fundamental structure of an artificial neuron. It is evident that the artificial neuron consists of seven fundamental components, which are given as

(a) The input signals $(x_1, ..., x_l, .., x_n)$ refer to the data or information received by the neuron from external sources or other neurons in the network. These input signals represent the values associated with specific variables relevant to the task or problem being addressed. Each input signal corresponds to a particular feature or aspect of the input data.

(b) Synaptic weights $(v_{r1}, ..., v_{rl}, ..., v_{rn})$ are numerical parameters assigned to each input connection. These weights represent the strength or significance of the respective input signals in influencing the neuron's output.

(c) The linear aggregator $(\sum)$ generates an activation voltage by combining all input signals weighted by their synaptic weight.

(d) The activation threshold or bias (B_r) is a crucial parameter in determining the threshold that the output of the linear aggregator must surpass to trigger the neuron's activation. In essence, it establishes the level of activation required for the neuron to produce an output signal. Adjusting this threshold enables fine-tuning the neuron's sensitivity to incoming signals, influencing its responsiveness and decision-making capabilities.

(e) The activation potential (V_r) is determined by the difference between the linear aggregator and the activation threshold. If this value is positive, indicating that $V_r > B_r$, the neuron produces an excitatory potential; otherwise, it produces an inhibitory potential.

(f) The activation function (F_r) is crafted to confine the neuron's output within a suitable range of values aligned with its intended functional behavior.

(g) The output signal (Y_r) represents the neuron's ultimate value depending on a specific collection of input signals. This output signal can also be input for other consecutively interconnected neurons.

The r-th neuron receives input signals, represented as $x_1, ..., x_l, ..., x_n$, which are connected via synaptic weights denoted as $v_{r1}, ..., v_{rl}, ..., v_{rn}$. These synaptic weights quantify the strength of connections between the input signals and the neuron. The sign of each synaptic weight indicates whether the connection is excitatory (positive) or inhibitory (negative). Notably, the notation for the subscripts in synaptic weights, such as v_{rl}, follows a specific convention: the first subscript identifies the neuron that receives the signals, and the second subscript refers to the input terminals of the synapse to which the weight is associated. In biological neurons, synaptic weights or connections are described with the pre-synaptic neuron (sending neuron) first and the post-synaptic neuron (receiving neuron) second. Unlike biological neurons, the synaptic weights in ANNs can take values within intervals of real numbers. The summing junction acts as a linear combiner of inputs, wherein each input signal x_l is multiplied by its synaptic weight v_{rl}. The bias B_r is an external stimulus that adjusts the input to the activation function $F_r(.)$, either increasing or decreasing it depending on the polarity of the bias. The mathematical expression for the linear combination of input signals, as proposed by McCulloch and Pitts [13], can be summarized as follows:

$$v_r = \sum_{l=1}^{n} x_l w_{rl} + B_r. \tag{1.12}$$

The output of the linear combiner v_r in equation (1.12) acts as an input or induced local field for the activation function. The activation function shapes the neuron's output to achieve the desired response. Commonly known as a "squashing function," it compresses the range of the neuron's output to a finite span. Figure 1.3 illustrates the output of a neuron using the activation function:

$$Y_r = F_r(v_r) = F_r(\sum_{l=1}^{n} x_l w_{rl} + B_r). \tag{1.13}$$

Therefore, the operation of an artificial neuron can be outlined through the following sequential steps:

(i) Input variables are presented to the neuron, constituting a set of values.

(ii) Each neuron input is multiplied by its corresponding synaptic weight.

(iii) The activation potential is derived by computing the weighted sum of the input signals and then subtracting the activation threshold.

(iv) It is imperative to employ an appropriate activation function to confine the neuron's output within a desired range.

(v) The output is calculated by applying the chosen neural activation function to the activation potential.

1.3.1 Activation function

Activation functions are crucial in determining the behavior and learning capability of the neural networks. Introducing non-linearity enables networks to capture and represent intricate relationships within data. The selection of activation functions substantially influences the network's training dynamics and overall performance. Here are several commonly used activation functions:

(a) **Sigmoid function.** The sigmoid function is strictly monotonically increasing, ensuring that for any two real numbers x and y, where $x < y$, then $F(x) < F(y)$. The sigmoid function is smooth. That is, its derivative is continuous for all real numbers. It is bounded within a finite range, with its output constrained to the interval $[0, 1]$. The logistic function stands as one prominent

example of a sigmoid function, defined by

$$F(u) = \frac{1}{1 + exp(-\beta u)}. \tag{1.14}$$

The symbol β represents the amplification factor, determining the slope parameter for the curve. Manipulating β allows for observing changes in the slopes of the curve, as illustrated in Figure 1.4(a). The mathematical expression in equation (1.14) indicates that as β approaches positive infinity, the sigmoid function tends toward values of 1, and as it approaches negative infinity, it tends toward 0. Consequently, the sigmoid function converges to a threshold function when the slope parameter β tends towards infinity. It is essential to highlight the distinction between threshold and sigmoid functions in ANNs. While a threshold function lacks smoothness, sigmoid functions exhibit smoothness, which is valuable in various applications. For instance, consider an all-or-none neuron model where the neuron fires only if the input exceeds a certain threshold. When the firing threshold is a random variable with a Gaussian normal distribution function, the expected output signal's value aligns with a sigmoid activity function [14]. This type of neuron model is referred to as the stochastic model. Due to its smoothness and versatility, the sigmoid activation function has gained popularity in designing ANN models. Beyond the sigmoid function, other commonly utilized examples of sigmoid activation functions include the inverse tangent and hyperbolic tangent functions. These functions offer advantages in terms of continuity and differentiability, contributing to the effectiveness of neural network architectures across a wide range of tasks and domains.

(b) **Hyperbolic tangent activation function.** The hyperbolic tangent activation function illustrated in Figure 1.4(b) differs from the logistic function by

consistently yielding real values within the range of -1 to 1, as defined by the following mathematical expression:

$$F(u) = \tanh(u). \tag{1.15}$$

(c) **Piecewise linear function.** The following mathematical expression characterizes the piecewise linear function depicted in Figure 1.4(c).

$$F(u) = \begin{cases} 0 & \text{if } u \leq -\frac{1}{\beta}, \\ u + \frac{1}{\beta} & \text{if } -\frac{1}{\beta} < u < \frac{1}{\beta}, \\ 1 & \text{if } u \geq \frac{1}{\beta}, \end{cases} \tag{1.16}$$

where β is often denoted as the amplification factor or the neural gain. This specific activation function has widespread application in cellular neural network architectures [15, 16]. The piecewise linear function depicted in Figure 1.4(c) corresponds to a value of β equal to 2. It is apparent from the mathematical expression 1.15 that as the amplification factor β approaches infinity, the piecewise linear function converges towards a threshold function.

(d) **Threshold function (step function).** For the activation function depicted in Figure 1.4(d), the output of a neuron from Figure 1.1 would be as follows:

$$Y_l = F_l(u) = \begin{cases} 1 & \text{if } u_l \geq 0, \\ 0 & \text{if } u_l < 0. \end{cases} \tag{1.17}$$

A threshold function is used to characterize the McCulloch-Pitts model [13] as its activation function. This model operates on a fundamental principle: a neuron emits a signal denoting "yes," represented as "1," if the induced local

field of that neuron is non-negative; otherwise, it emits a signal of "0." The activation function characterizes the all-or-none property of a neuron in the McCulloch-Pitts model.

(e) **RELU function** The Rectified Linear Unit (ReLU) is a popular nonlinear activation function extensively employed in neural networks [17]. Mathematically, it is defined as $f(x) = \max(0, x)$. The key advantage of ReLU lies in its ability to accelerate training speed and mitigate saturation issues commonly encountered in other activation functions. Several enhanced versions of ReLU have been developed to address its limitations. These include leaky ReLU, parametric ReLU, and randomized ReLU. These variations aim to overcome the tendency of standard ReLU to discard extreme values, which can lead to either information loss or explosive behavior within the neural network [18]. By introducing controlled modifications to the ReLU function, these variants offer improved stability and performance, enhancing the overall effectiveness of neural network training and operation.

In essence, activation functions play a pivotal role in defining the behavior and learning capabilities of neural networks. By introducing non-linearity, these functions enable networks to capture and represent complex relationships within data, thereby enhancing their capacity to learn and generalize. The selection of an appropriate activation function can significantly influence the training process and the overall performance of the network. While traditional activation functions are predefined and fall into the category of fixed-shape activation functions, ongoing research is exploring the potential of trainable activation functions [19]. These trainable functions offer the prospect of adapting their shape during the training process, potentially leading to improved network performance and enhanced flexibility in modeling diverse data distributions.

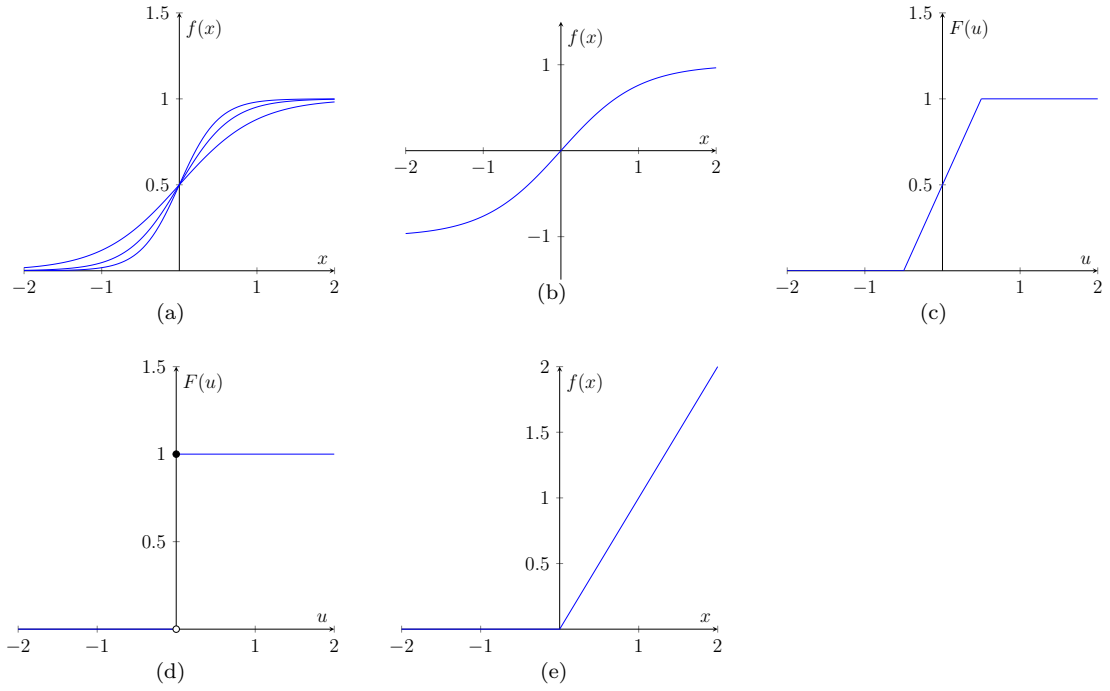

FIGURE 1.4: (a) Sigmoidal function, (b) tanh function, (c) Piecewise activation function, (d) Threshold function, (e) RELU function

1.3.2 Key architectures of ANNs.

(a) **Input Layer.** This layer serves as the initial recipient of information, encompassing data, signals, features, or measurements from the external environment. Within this layer, inputs, which could be samples or patterns, are frequently normalized to fit within the boundaries defined by the activation function. Normalization ensures that input values are appropriately scaled, enhancing the numerical precision required for subsequent mathematical operations the network executes. This preparatory step facilitates more accurate and efficient information processing as it traverses the neural network architecture.

(b) **Intermediate Layers.** These layers within a neural network are neurons that extract patterns relevant to the analyzed process or system. As the backbone

of internal processing, these layers play a pivotal role in transforming input data into meaningful representations, facilitating the network's ability to learn and generalize from the provided information.

(b) **Output Layer.** This layer, comprised of neurons like the preceding layers, serves the crucial role of producing and presenting the ultimate outputs of the network. These outputs are the culmination of the computational operations conducted by the neurons across the preceding layers.

The primary architectures of ANNs, taking into account the arrangement of neurons, their interconnections, and the composition of layers, can be categorized as follows: (i) single-layer feedforward ANNs, (ii) multilayer feedforward ANNs, (iii) recurrent ANNs, and (iv) mesh ANNs.

1.3.3 Architecture of a single-layer feedforward Network

The single-layer feedforward architecture of an ANN comprises a solitary input layer and a sole neural layer, which concurrently serves as the output layer. This architectural configuration, as depicted in Figure 1.5, encompasses n inputs and m outputs. Information flow within this architecture is unidirectional, consistently progressing from the input layer towards the output layer. Notably, the number of network outputs aligns with the number of neurons within the neural layer. This architecture finds frequent application in tasks such as pattern classification and linear filtering, as illustrated in Figure 1.5.

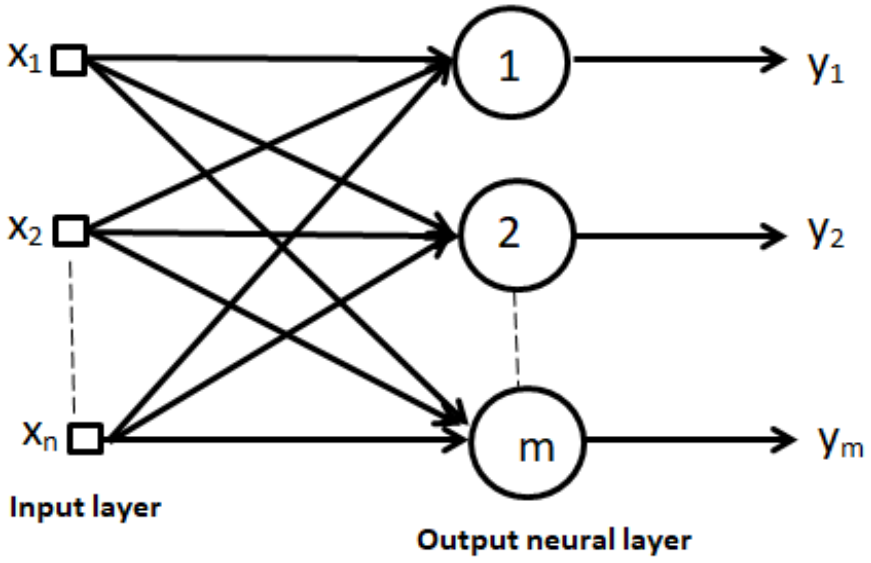

FIGURE 1.5: Single-layer feedforward network

1.3.4 Architectures of multi-layer feedforward networks

In contrast to the simpler single-layer network architecture, multi-layer feedforward networks are characterized by one or more hidden neural layers, as illustrated in Figure 1.6. These networks are employed across various applications, including function approximation, pattern classification, system identification, process control, optimization, robotics, and more. Figure 1.6 illustrates a typical configuration of a multi-layer feedforward network, comprising the following components: an input layer containing n sample signals, two hidden neural layers with n_1 and n_2 neurons, and a final output neural layer consisting of m neurons representing the corresponding output values of the analyzed problem. Observing Figure 1.6, it is apparent that the number of neurons in the first hidden layer often differs from the number of signals in the network's input layer. Determining the number of hidden layers and their respective neurons depends on various factors, including the nature and complexity of the problem being addressed and the quantity and quality of available

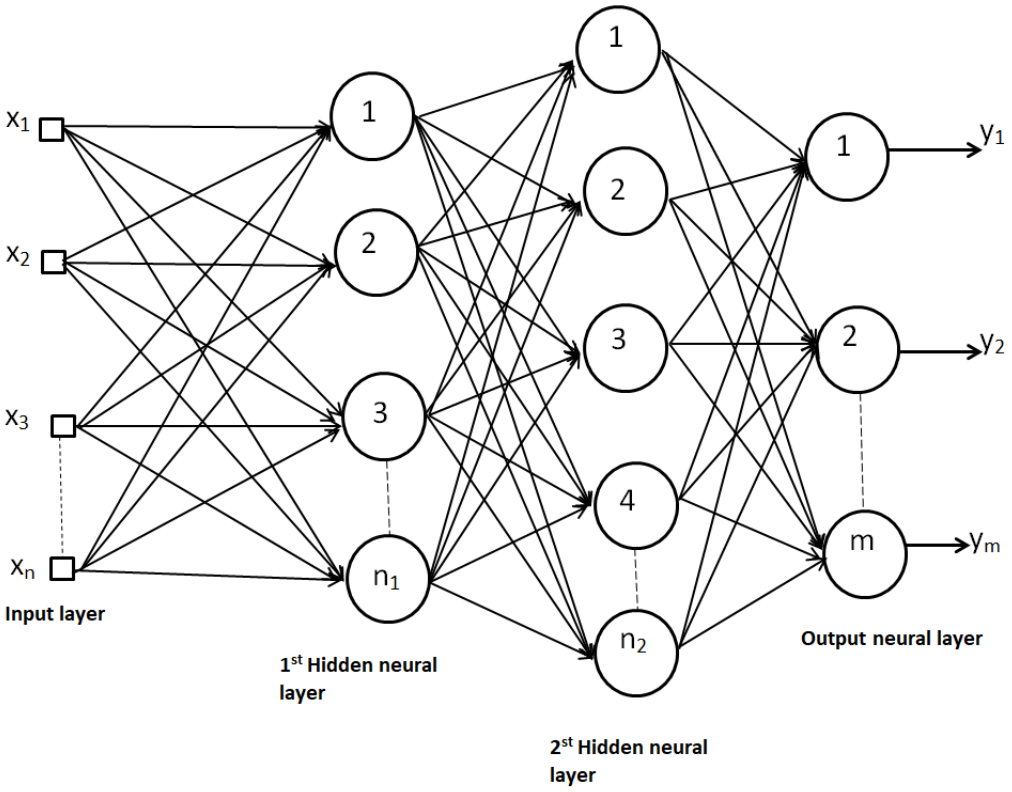

FIGURE 1.6: Multi-layer feedforward network

data associated with the problem. However, akin to single-layer feedforward networks, the number of output signals consistently aligns with the number of neurons in the respective output layer.

1.3.5 Architecture of recurrent or feedback networks

In recurrent or feedback networks, the outputs of neurons play a dual role, serving not only as outputs but also as feedback inputs for other neurons within the network. Incorporating feedback mechanisms endows these networks with the capability for dynamic information processing, making them well-suited for applications involving time-varying systems. Examples of such applications span a wide range, including time series prediction, system identification, optimization, process control, and more.

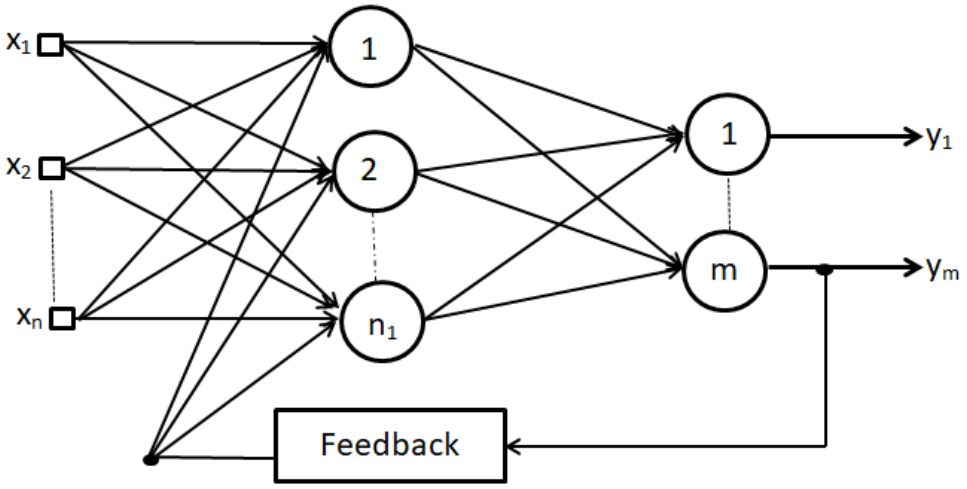

FIGURE 1.7: Recurrent feedforward network

Among the primary recurrent networks are the Hopfield and Perceptron, which incorporate feedback connections between neurons, often spanning distinct layers. The training processes of these networks typically involve learning algorithms based on principles such as energy function minimization and the generalized delta rule. As depicted in Figure 1.7, an example of a Perceptron network with feedback illustrates the feedback loop where one of its output signals is fed back to the middle layer. Through this feedback mechanism, networks with such architectures generate current outputs while considering previous output values, allowing them to effectively capture temporal dependencies and perform dynamic computations.

1.3.6 Mesh architectures

The defining feature of networks utilizing mesh structures is their focus on the spatial organization of neurons to extract patterns effectively. Within these networks, the spatial positioning of neurons directly influences the tuning of their synaptic weights and thresholds. Mesh architectures are deployed across diverse domains, encompassing applications such as data clustering, pattern recognition, system optimization,

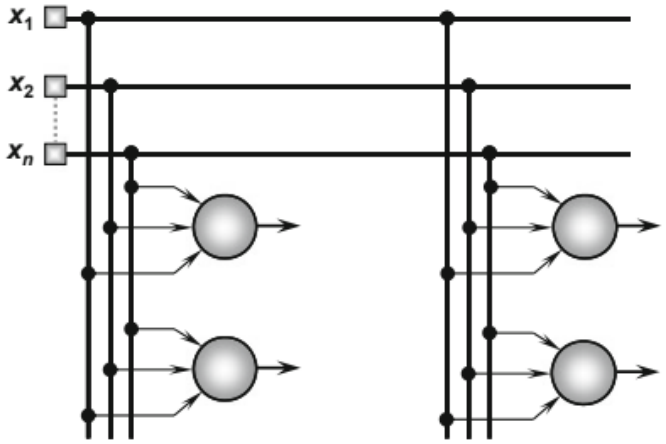

FIGURE 1.8: Structure of a Mesh network

and tasks related to graph analysis. This spatially aware approach facilitates enhanced pattern extraction and interpretation, contributing to the versatility and efficacy of mesh-based neural networks across a broad spectrum of tasks and disciplines. From the preceding discussions, it is evident that several crucial factors significantly impact the performance of a neural network. These factors include:

(1) External inputs.

(2) Internal decay rates.

(3) Synaptic weights.

(4) Activation functions.

(5) Propagation delays.

(6) Connections topology/Network architecture.

In deterministic models, a network's behavior is dynamically determined by its connection topology. By Considering the initial activation levels of neurons and the synaptic coupling coefficients weights, and treating other relevant factors as parameters, it is possible to compute subsequent activation levels and synaptic weights. This phenomenon is known as joint activation-weight dynamics. However, practical applications, the dynamics of neuron activation and weight adjustment are often treated separately. Various methods exist to dynamically adjust the synaptic weights

of a network to achieve specific objectives, such as pattern recognition or generating desired network outputs from a given set of inputs. These approaches typically involve defining a discrete dynamical system (a system of difference equations) or a continuous dynamical system (a system of differential equations) within the space of matrices representing synaptic coupling coefficients. This aspect is commonly referred to as weight dynamics. Once the connection topology and synaptic weights are established, future activation levels can be predicted by specifying the initial activation levels and treating other factors as parameters. With the foundational properties of ANNs discussed, the exploration now proceeds to introduce prominent neural network models, such as the Hopfield and Cohen-Grossberg models, which are pivotal for the current thesis. Understanding the RC circuit before introducing the Hopfield model is essential, as it provides the necessary background for grasping the dynamics of these neural network models.

1.3.7 Basics of electrical circuit

Understanding electrical circuits is essential for grasping the intricate processes underlying signal transmission among neurons in the human brain. As elucidated earlier, the membrane of a biological neuron behaves akin to a capacitor, facilitating the establishment of distinct ion concentrations. This segregation of charges culminates in forming an electric pulse, which propagates along the neuron's axon to impact the neighboring neuron. A resistor-capacitor (RC) circuit is a suitable model to replicate this phenomenon, as illustrated in Figure 1.9. In the depicted electrical circuit shown in Figure 1.9, a series arrangement of components comprising a resistor (with a resistance R), a capacitor (C), and a voltage source (battery) is illustrated. This configuration serves as a model for comprehending the dynamics associated with the transmission of electrical signals. More specifically, the battery represented

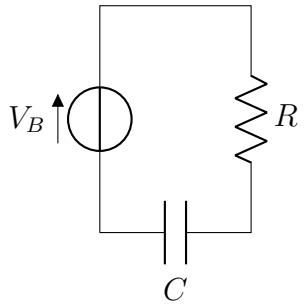

FIGURE 1.9: RC-circuit with a source of voltage.

in the circuit symbolizes the electromotive force resulting from disparities in ion concentrations inside and outside the cell membrane. Similarly, the resistor simulates ion channels, regulating the membrane's permeability to specific ions. The analysis of this circuit's behavior relies on applying Kirchhoff's laws, which govern the principles of current and voltage conservation in electrical circuits.

(a) *Kirchhoff's Current Law:* The total current flowing into a junction equals the total current flowing out of that junction.

(b) *Kirchhoff's Voltage Law:* The algebraic sum of voltage differences around any closed loop equates to zero.

Kirchhoff's voltage law states that the voltage drop across a capacitor equals the algebraic sum of the voltage drops across a battery and a resistor.

$$V_C = V_B + I_R R, \tag{1.18}$$

where V_C stands for the voltage across the capacitor, V_B denotes the voltage across the battery, and I_R signifies the current flowing through the resistor, as defined by

Ohm's law ($V_R = I_R R$). Therefore, we have

$$I_R(t) = \frac{V_C(t) - V_B(t)}{R}. \tag{1.19}$$

The neuron receives an external current input via its synapses, represented as I_{ext}. This current input adheres to Kirchhoff's current law, which can be articulated as follows:

$$I_{ext.} = I_C + I_R. \tag{1.20}$$

Equations (1.19) and (1.20) yield the following equation

$$I_{ext} = C\frac{dV_C}{dt} + \frac{V_C(t) - V_B(t)}{R}. \tag{1.21}$$

1.3.8 Additive model

Drawing inspiration from the behavior of biological neurons, modeled as RC circuits, the model described in equation (1.13) can be expanded to incorporate the temporal dynamics of input data. When dealing with inputs that fluctuate over time, one approach to address this temporal aspect is illustrated in Figure 1.10. Here, synaptic weights $w_{l1}, w_{l2}, ..., w_{ln}$ are analogously represented by conductance (the reciprocal of resistance), while the respective inputs $x_1(t), x_2(t), ..., x_n(t)$ are depicted as potentials (voltages). These inputs' products and corresponding synaptic weights are amalgamated at the current summing junction. A low input resistance, unity current gain, and high output resistance characterize this junction.

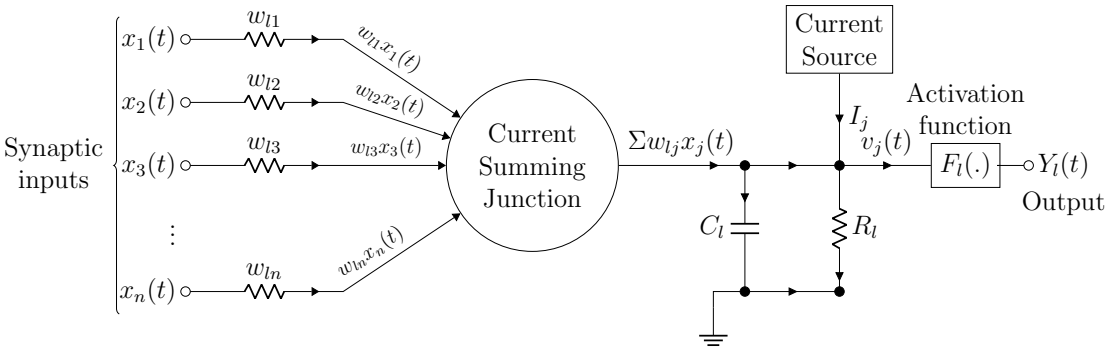

FIGURE 1.10: Additive model of a neuron.

The total current flowing to the input node of the activation function $F_l(.)$ is

$$\sum_{j=1}^{n} w_{lj}x_j(t) + I_l, \tag{1.22}$$

the first component originates from the input signals of the l-th neuron, while the second component arises from an external current source acting as a bias for the neuron. Denoting the induced local field at the input node of the activation function as $v_l(t)$, the total current departing from the input node of the activation function can be formulated as

$$\frac{v_l(t)}{R_l} + C_l\frac{dv_l(t)}{dt}, \tag{1.23}$$

where the initial term corresponds to the current flowing through the resistor R_l, while the second term denotes the current generated by the voltage drop across the capacitor C_l. Adhering to Kirchhoff's current law, the total current entering the input node of the activation function depicted in Figure 1.10 equals zero. Therefore, we can infer the following from equations (1.22) and (1.23).

$$\frac{v_l(t)}{R_l} + C_j\frac{dv_l(t)}{dt} = \sum_{j=1}^{n} w_{lj}x_j(t) + I_l. \tag{1.24}$$

The activation function determines the output of neuron l based on the induced input field $v_l(t)$ as

$$Y_l(t) = F_l(v_l(t)). \tag{1.25}$$

Usually, within the additive model, the activation functions are chosen to be both bounded and differentiable. They demonstrate asymptotic behavior, as depicted by functions like the logistic function, showcased in Figure 1.4(a).

$$F_l(v_l) = \frac{1}{1 + \exp(-v_l)}. \tag{1.26}$$

The differential equation (1.24) defines a neuron model known as additive model. The designation "additive" is utilized to differentiate it from multiplicative (shunting) models, which integrate state-dependent synaptic weights [20].

1.3.9 Hopfield neural network

Consider a fully connected recurrent network comprising n neurons; each structured similarly to the one depicted in Figure 1.10. The essential connectivity pattern is portrayed in Figure 1.11, wherein each neuron's output is fed back, via a unit delay element z^{-1}, to the inputs of other neurons through feedback loops. Notably, the network lacks self-feedback loops, enabling the analysis of network dynamics by considering the instantaneous propagation of signals between neurons and Equation (1.23). This yields

$$C_l \frac{dw_l(t)}{dt} = -\frac{w_l(t)}{R_l} + \sum_{i=1}^{n} v_{lj} F_j(w_j(t)) + I_l, \text{for } j = 1, 2, ..., n. \tag{1.27}$$

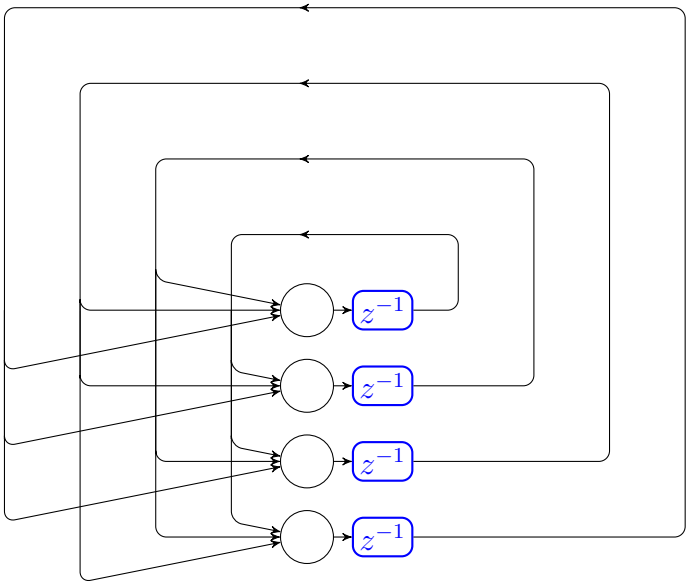

FIGURE 1.11: An architecture of Hopfield neural network consisting of $n = 4$ neurons.

Here, $x_j(t) = F_j(w_j(t))$ indicates that each neuron in the system has its activation function. Equation (1.27) characterizes the Hopfield neural network configuration without any time delay. The Hopfield model typically manifests in two distinct formulations: discrete and continuous flows. The solution to the differential equation (1.27) corresponds to the continuous flow representation of the Hopfield neural network. The discrete dynamics of the Hopfield network adhere to the McCulloch-Pitts model, where the state of each neuron is binary, taking values of either -1 or 1 based on the signs of the induced local field. In particular, $x_j(t) = -1$ if $w_j(t) > 0$, and $x_j(t) = 1$ if $w_j(t) < 0$, for all j. However, this thesis primarily emphasizes the continuous flow of neural networks rather than the discrete one.

The Hopfield neural network has received significant attention in various applications, particularly as an associative memory system. Associative memory, a variant of content-addressable memory, stores patterns within the fixed points of the network. Its primary function is retrieving a stored pattern from memory, even

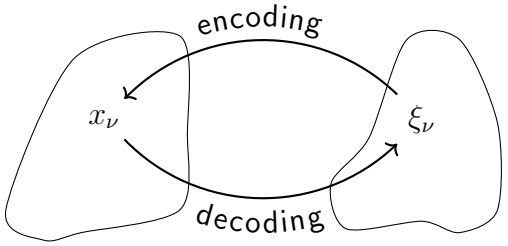

FIGURE 1.12: Encoding-Decoding illustration between the space of fundamental memories ξ_μ and the space of stored vectors x_μ.

when presented with incomplete or noisy information. A crucial aspect of content-addressable memory is its ability to recall the stored pattern using partial or subset information related to that pattern [21, 22]. In mathematical terms, the core concept of content-addressable memory entails mapping a fundamental memory ξ_ν to a stable fixed point x_ν within the network. The encoding process of a fundamental memory onto a fixed point is depicted by the arrow from right to left in Figure 1.12. Conversely, the arrow from left to right illustrates the decoding process used to retrieve the stored memory. Now, consider that the neural network contains a pattern that holds some, but not complete, information about one of its fundamental memories. This partial information serves as the starting point for the network's dynamics, residing within the basin of attraction of a stable fixed point. Mathematically, the question identifies the conditions under which the network's dynamics converge towards this fixed point, regardless of the initial data. In the context of content-addressable memory, the number of fixed points in the neural network plays a pivotal role in its applications. A more significant number of fixed points signifies a higher storage capacity within the network.

John Hopfield's 1984 paper, [23], delved into examining the dynamics of the Hopfield network. In this seminal work, he introduced an energy function for the network, establishing it based on certain underlying assumptions.

(i) The synaptic weights matrix exhibits symmetry, represented as $v_{lj} = v_{jl}$ for all l and j.

(ii) An inverse of the activation function described in equation (1.26) can be derived, enabling us to represent w as $F_l^{-1}(x)$.

Using equation (1.26), we have

$$F_l^{-1}(x) = -\ln\left(\frac{1-x}{1+x}\right). \tag{1.28}$$

The Lyapunov function associated with the Hopefield model (1.27) can be expressed as

$$L = -\frac{1}{2}\sum_{j=1}^{n}\sum_{l=1}^{n} v_{jl}x_j(t)x_l(t) + \sum_{l=1}^{n}\frac{1}{R_l}\int_0^{x_l} F_x^{-1}(l)dx - \sum_{l=1}^{n} I_l x_l. \tag{1.29}$$

Later on, the concept of a Lyapunov function will be introduced. The Lyapunov function described in equation (1.29) has multiple minima points, each corresponding to stable fixed points within the Hopfield model (1.27). The network dynamics are directed towards recognizing and converging to these minimum points. Hence, upon taking the derivative of L with respect to time and utilizing equations (1.27) and (1.28), the following result is obtained.

$$\frac{dL}{dt} = -\sum_{l=1}^{n} C_l\left(\frac{dx_l}{dt}\right)^2\left[\frac{dF_l^{-1}(x_l)}{dx_l}\right]. \tag{1.30}$$

Based on equation (1.28), it is clear that the inverse activation function consistently increases as the output increases. Therefore, it follows that

$$\frac{dF_l^{-1}(x_l)}{dx_l} \geq 0 \quad \forall\ x_l. \tag{1.31}$$

Therefore, from equation (1.30), we can get

$$\frac{dL}{dt} \leq 0. \tag{1.32}$$

In the context of Lyapunov's stability method, when considering the inequality (1.32), it signifies that the continuous Hopfield neural network, as depicted by equation (1.27), follows a trajectory aiming to reach the minima of the energy function L. This trajectory is directed towards converging to these minima, ultimately ceasing at these fixed points. Notably, the inequality (1.32) attains a value zero exclusively when

$$\frac{dx_l}{dt} = 0 \text{ for all } j. \tag{1.33}$$

Thus, we can state that

$$\frac{dL}{dt} < 0, \text{ except at fixed points.} \tag{1.34}$$

As a result, the continuous Hopfield neural network's trajectory gradually approaches and converges towards its fixed points over time.

1.3.10 Cohen-Grossberg neural network

The Cohen-Grossberg neural network, introduced by Cohen and Grossberg [24] in the year 1983, is widely acknowledged as a generalized iteration of the Hopfield network. In their work, M. A. Cohen and S. Grossberg established a fundamental principle for constructing content-addressable memory networks, showcasing the

formulation of these models in the following manner:

$$\frac{dx_j(t)}{dt} = a_j(x_j(t))\big[b_j(x_j) - \sum_{l=1}^{n} w_{jl}F_l(x_l)\big], \text{ for all } j = 1, 2, ..., n. \tag{1.35}$$

In the given system (1.35), where $x_j(t)$ represents the state of the $j-$th neuron, $a_j(x_j(t))$ denotes the amplification function, $b_j(x_j)$ stands for the self-signal function, and $F_l(x_l)$ signifies the standard signal function, it is notable that the rate of change in the neuron's activity diminishes only when the net input to the neuron exceeds a specific intrinsic function b_j of its activity, and when $a_j(x_j(t))$ is positive. The following Lyapunov function is applicable to the nonlinear system (1.35)

$$L(x) = -\sum_{j=1}^{n}\int_0^{x_j} b_j(\xi_j)F_j^{'}(\xi_j)d\xi_j + \frac{1}{2}\sum_{l,i=1}^{n} w_{li}F_l(x_l)F_i(x_i), \tag{1.36}$$

if the synaptic coefficients w_{li}, along with the other components a_j, b_j, and F_j of the system, satisfy the following conditions

(i) *Symmetric*: $w_{jl} = w_{lj}$;

(ii) *Continuity*: $a_j(x)$ and $b_j(x)$ are continuous for $x > 0$;

(iii) *Positivity*: $a_j(x) > 0$ if $x > 0$;

(iv) *Monotonicity*: $F_j(x)$ is continuously differentiable, and $F_j^{'}(x) \geq 0$ for $x \geq 0$.

After derivating L, one obtains the following result.

$$\frac{dL}{dt} = -\sum_{j=1}^{n} a_j F_j^{'}\big[b_j - \sum_{l=1}^{n} w_{jl}F_l\big]^2. \tag{1.37}$$

Taking into account conditions (iii) and (iv), it follows that $\frac{dL}{dt}$ is non-positive along the trajectories. In simpler terms, this means that the global trajectories of system (1.35) converge towards the equilibrium points.

M. A. Cohen and S. Grossberg highlighted in their work [25] that the differential equation (1.35) can depict the additive model (1.27) by utilizing coefficients interpreted in the standard electrical circuit sense.

$$a_j(x_j(t)) = \frac{1}{C_j}, \tag{1.38}$$

$$b_j(x_j(t)) = -\frac{1}{R_j} + I_j, \tag{1.39}$$

$$w_{jl} = -w_{jl}. \tag{1.40}$$

In the case of additive systems, the amplification function maintains a positive constant value, thereby fulfilling the condition of positivity. Additionally, the self-signal function exhibits linearity. By integrating equations (1.38), (1.39), and (1.40) into the Lyapunov function (1.36), we derive

$$L = -\frac{1}{2}\sum_{i,l=1}^{n} w_{il}F_i(x_i)F_l(x_l) + \sum_{j=1}^{n}\frac{1}{R_j}\int_0^{x_j}\xi_j F_j^{'}(\xi_i)d\xi_j - \sum_{j=1}^{n} I_j F_j(x_j). \tag{1.41}$$

In the Hopfield energy function, it was initially stipulated that the activation function must be invertible. However, Cohen and Grossberg introduced the concept of a non-decreasing activation function. Interestingly, Hopfield's seminal work emerged in 1984, a year after Cohen and Grossberg's findings were published. Despite this chronological sequence, physicists and engineers frequently referred to the additive model as the "Hopfield additive model" in their literature. In response to this convention, S. Grossberg authored a review article [25], which elucidated how the additive model can be viewed as a specialized instance of the work he had collaborated

on with M. A. Cohen [26].

1.4 A comprehensive exploration of mathematical concepts

This segment aims to offer insight into the mathematical principles utilized throughout the chapters of the current thesis, emphasizing stability theory, Lyapunov stability theory, Lagrange stability theory, delayed differential equations, and Fractional order systems.

1.4.1 Stability theory

Let us consider the autonomous system

$$\dot{y} = f(y), \tag{1.42}$$

here, $f : D \to \mathbb{R}^n$ be a local Lipschitz map from a domain $D \subset \mathbb{R}^n$ into $\mathbb{R}^n$. Assume $\bar{y}_e \in D$ is an equilibrium point of equation (1.42), i.e.,

$$f(\bar{y}_e) = 0.$$

For simplicity and without loss of generality, we define all concepts and theorems specifically for the case where the equilibrium point is positioned at the origin of $\mathbb{R}^n$, denoted as $\bar{y} = 0$. If another equilibrium point exists, we can transform variables by shifting that equilibrium point to the origin. Suppose $\bar{y} \neq 0$, and consider the

change of variables $z = y - \bar{y}$. The derivative of z is expressed as:

$$\dot{z} = \dot{y} = f(y) = f(z + \bar{y}) \stackrel{\text{def}}{=} g(z), \qquad \text{where } g(0) = 0$$

The system reaches equilibrium at the origin within the newly introduced variable, referred to as y. Moving forward, to maintain the validity of our analysis, we will consistently operate under the assumption that $f(y)$ adheres to the condition $f(0) = 0$. Subsequently, we will investigate the stability of the origin, specifically at $y = 0$.

Definition 1.4.1. The equilibrium point $y = 0$ of system (1.42) is

(i) **Stable**: If, for every $\epsilon > 0$, there exists a $\delta > 0$ such that, if $\|y(0) - y_e\| < \delta$, then for every $t \geq 0$ we have $\|y(t) - y_e\| < \epsilon$.

(ii) **Asymptotically Stable**: If it is stable and there exists $\delta > 0$ such that if $\|y(0) - y_e\| < \delta$, then $\lim_{t\to\infty} \|y(t) - y_e\| = 0$.

(iii) **Exponential Stable**: The equilibrium of the above system is said to be exponentially stable if it is asymptotically stable and there exists $N > 0, b > 0, \delta > 0$ such that if $\|y(0) - y_e\| = N < \delta$, then $\|y(t) - y_e\| \leq Ne^{-bt}$, for all $t \geq 0$.

(iv) **Unstable** If it is not stable.

In the context of the ϵ-δ framework, ensuring stability of the origin entails a challenge and its subsequent resolution. Demonstrating the stability involves proving that for any selected value of ϵ, a corresponding δ exists, which can vary based on ϵ. This δ guarantees that any trajectory starting within a δ-neighborhood of the origin will remain within an ϵ-neighborhood for its entire duration.

1.4.2 Lyapunov stability

In 1892, Lyapunov functions, named after Aleksandr Lyapunov, emerged as pivotal tools for assessing the stability of equilibria in differential equations [27]. These functions also referred to as Lyapunov's second method for stability, hold significant importance in the stability theory of dynamical systems and control theory. Consider a continuously differentiable function $L : D \to \mathbb{R}$ defined over a domain $D \subset \mathbb{R}^n$ containing the origin. The derivative of L along the trajectories of equation (1.42), symbolized as $\dot{L}(y)$, is expressed as follows:

$$\begin{aligned}\dot{L}(y) &= \sum_{j=1}^{n} \frac{\partial L}{\partial y_j}\dot{y}_j = \sum_{j=1}^{n} \frac{\partial L}{\partial y_j} f_j(y) \\ &= \begin{bmatrix} \frac{\partial L}{\partial y_1}, & \frac{\partial L}{\partial y_2}, & \cdots, & \frac{\partial L}{\partial y_n} \end{bmatrix} \begin{bmatrix} f_1(y) \\ f_2(y) \\ \vdots \\ f_n(y) \end{bmatrix} = \frac{\partial L}{\partial y} f(y)\end{aligned}$$

The rate of change of L along the trajectories of a system is contingent upon the specific equation governing the system. As a result, the expression for $\dot{L}(y)$ will demonstrate variability depending on the particular system in consideration. If we denote $\psi(t; y)$ as the solution of equation (1.42) with an initial state of y at time $t = 0$, then we can articulate the expression for $\dot{L}(y)$ as follows:

$$\dot{L}(y) = \frac{d}{dt} L(\psi(t; y))\Big|_{t=0}$$

Therefore, if $\dot{L}(x)$ is negative, the function L will decrease along the solution trajectory of equation (1.42).

Theorem 1.4.1. Assume that the point $y = 0$ is an equilibrium point for equation (1.42), and let $D \subset \mathbb{R}^n$ be a domain that includes $y = 0$. If there exists a continuously differentiable function $L : D \to \mathbb{R}$ such that:

$$L(0) = 0 \text{ and } L(y) > 0 \text{ in } D - \{0\}$$

$$\dot{L}(y) \leq 0 \text{ in } D$$

In this case, the equilibrium point at $y = 0$ exhibits stability. Moreover, if the condition $\dot{L}(x) < 0$ is satisfied throughout the domain $D - 0$, then the equilibrium point $y = 0$ is designated as asymptotically stable.

Proof. The theorem's proof can be readily located in [28]. □

Selecting an appropriate Lyapunov function is essential for demonstrating stability. In this thesis, the Lyapunov functions are specifically tailored to the unique dynamics and characteristics of the systems being studied. Generally, the process begins by identifying a Lyapunov function that represents the system's energy or a similar concept. It is crucial that this function is positive definite and that its derivative along the system's trajectories is negative definite. This approach ensures that the system's trajectories remain bounded and converge to the equilibrium point, thereby guaranteeing stability.

Theorem 1.4.2. [28] Suppose within every neighborhood of the equilibrium point $y = 0$, there exists at least one point where the function $L(y, t)$ is positive (negative), and along the trajectories of the system, the derivative $\frac{dL}{dt} > 0$(or < 0). In that case, system's origin is considered an unstable critical point (1.42).

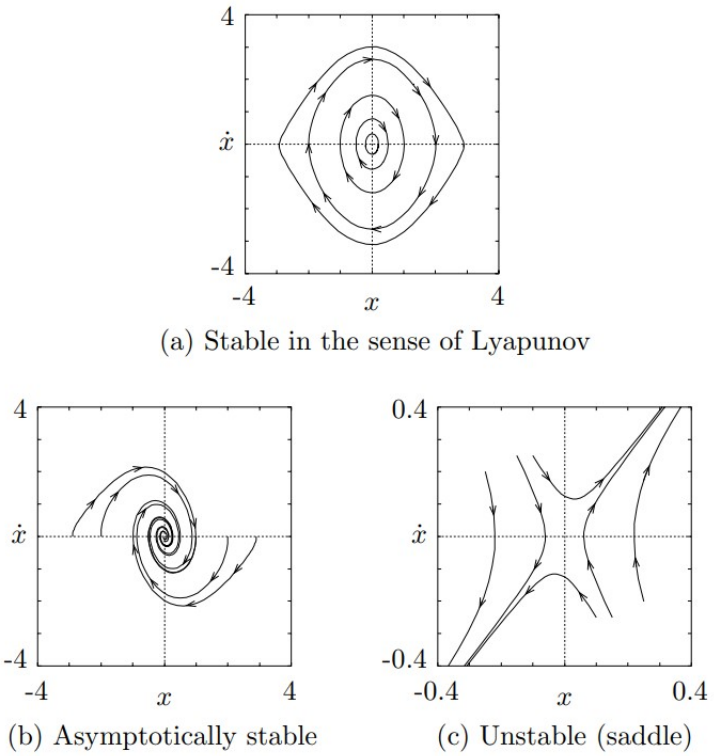

FIGURE 1.13: Phase portrait for stable and unstable equilibrium point

1.4.3 Lagrange stability

Lagrange stability extends the concept of Lyapunov stability to include the notion of the boundedness of the trajectories of a dynamical system. It focuses on the behavior of trajectories near an equilibrium point and their behavior over the entire state space. A system is Lagrange stable if its trajectories are both Lyapunov stable and remain bounded. In other words, not only do trajectories starting near the equilibrium point remain close to it, but trajectories starting from any initial condition do not escape to infinity. In Lagrange stability analysis, a global attractive set is often employed. A global attractive set is a region in the system's state space from which trajectories starting in its vicinity ultimately converge.

Theorem 1.4.3. [29] **Lagrange Stability Theorem**: Consider a bounded neighborhood D centered at the origin, with D^c denoting its complement (i.e., the set

of all points outside D). Let $L(y)$ be a scalar function with continuous first partial derivatives defined in D^c and satisfying the following conditions:

1 $L(y) > 0 \; \forall \; y \in D^c$,

2 $\dot{L}(y) \leq 0 \; \forall \; y \in D^c$,

3 $L(y) \longrightarrow \infty$ as $\|y\| \longrightarrow \infty$,

then, every solution of the (1.42) is bounded for all $t \geq 0$.

1.4.4 Finite and fixed time stability

Finite-time stability (FNTS) and fixed-time stability (FTS) are concepts in control theory that describe the behavior of dynamical systems in response to perturbations within a finite or fixed time duration, respectively. FNTS was introduced in the 1950s by P. Dorato [30] and ensures that a system will reach equilibrium within a finite time, depending on its initial conditions. Unlike asymptotic stability, which guarantees convergence to an equilibrium point as time approaches infinity, FNTS sets a bound on the time it takes for the system to reach equilibrium. This makes FNTS particularly useful for systems where time constraints are essential or where waiting for asymptotic stability is not feasible. FTS is a more potent form of FNTS, introduced by A. Polyakov [31] in the year 2012. It guarantees convergence to equilibrium within a finite time and ensures that this convergence time is independent of the system's initial conditions. In other words, FTS guarantees that the system will reach equilibrium within a fixed duration, regardless of where it starts. This property makes FTS particularly desirable for applications where predictability and robustness to initial conditions are crucial.

A dynamic system is also considered FNTS if a finite settling time T exists. For any initial state within a specific region of attraction, the system's state will reach the equilibrium point within T seconds. Although the settling time may vary based on the initial state, it remains finite. Furthermore, a dynamic system is deemed FTS if a finite settling time T exists independent of the system's initial state. This implies that for any initial state within a specific region of attraction, the system's state will reach the equilibrium point within the same fixed duration of time T.

1. **Finite Time Stability** The equilibrium point of a dynamical system is termed finite-time stable [31] if, for any solution $y(t)$ starting at y_0, it is asymptotically stable, and $\lim_{t \to T(y_0)} |y(t)| = 0$ holds, with $y(t) = 0$ for all $t \geq T$, where T is denoted as the settling time function.

2. **Fixed Time Stability** An equilibrium point is considered fixed-time stable [31] if it meets the criteria of finite-time stability and, furthermore, if the settling time is bounded, i.e., $T(y_0) < T_{\max}$ for all $y_0 \in \mathbb{R}^n$, where $T_{\max}$ is a constant.

An instance demonstrating finite-time convergence can be found in the bang-bang time-optimal feedback control [32]. The utility of both FNTS and FTS transcends various domains, including robotics, hypersonic missiles, power systems, engineering systems, and space technology [33, 35].

1.4.5 Delay differential Equations

A delay differential equation (DDE) is a differential equation in which the highest-order derivative at a certain time is a function of the dependent variable at a previous

time. For instance,

$$\dot{y}(t) = y^3(t-1) + y(t), \tag{1.43}$$

meets the criteria for a DDE because the highest-order derivative, $\dot{y}(t)$ depends on the value $y(t-3)$ but

$$\dot{y}(t-3) = y^3(t) + y(t) \tag{1.44}$$

does not satisfy the conditions required for a DDE. This is due to a violation of the characteristic structure of a DDE in which the terms $y^3(t)$ and $y(t)$ do not share the same delayed argument, resulting in the highest order derivative $\dot{y}(t-3)$ involving a delay of 3 units.

Let us define

$$C = C\left([-\tau, 0]; \mathbb{R}^n\right) = \{\psi; \psi : [-\tau, 0] \to \mathbb{R}^n \text{ is continuous } \}.$$

Equipping this space with the specified norm transforms it into a Banach space.

$$\|\psi\| = \sup_{s\in[-\tau,0]} |\psi(s)|, \quad \psi \in C,$$

where $|\cdot|$ represents the usual Euclidean norm. Let us define $y_t : [-\tau, 0] \to \mathbb{R}^n$ by

$$y_t(s) = y(t+s), \quad \text{for } s \in [-\tau, 0]. \tag{1.45}$$

The part of the graph of y that is restricted to the interval $[t-\tau, t]$ and then translated back to the original interval $[-\tau, 0]$ is represented geometrically by the variable y_t.

For $E \subset \mathbb{R}^n$, the space $\mathcal{C}_E = C([-\tau, 0], E)$ is referred to as Banach space. This space contains continuous functions that transfer the interval $[-\tau, 0]$ to the set E.

Definition 1.4.2. Let $A \subset \mathbb{R}$, $f : A \times \mathcal{C}_E \to \mathbb{R}^n$ is a given function, and $D^+(x(t))$ denotes the right-hand time derivative. Then, we say that the following relation

$$\dot{y}(t) = f(t, y_t), \tag{1.46}$$

is a delay differential equation on $A \times \mathcal{C}_E$.

The right-hand time derivative of a function $y(t) : \mathbb{R} \to \mathbb{R}$ is defined as

$$D^+(y(t)) = \lim_{h \to 0^+} \sup \frac{y(t+h) - y(t)}{h}. \tag{1.47}$$

For a given $t_0 \in I$ and $\psi_0 \in \mathcal{C}_E$, the initial value problem (IVP) related with the DDE (1.46) is given by:

$$\begin{cases} \dot{y}(t) = f(t, y_t), \ t > t_0, \\ y(s) = \psi_0(s - t_0), \ \forall \ s \in [t_0 - \tau, t_0]. \end{cases} \tag{1.48}$$

This implies that at the initial time t_0, the solution $x(t)$ needs to be specified for the entire past interval $[t_0 - \tau, t_0]$. However, the initial function or history ψ_0, which defines the state of the system at $t_0 - \tau$, is a continuous function but may not necessarily satisfy the delay differential equation (1.48). As a result, the solution might not exhibit differentiability at the initial instant t_0. For example, consider the simple model of a DDE given by

$$\begin{cases} \dot{y}(t) = y(t-1), \ t > 0, \\ y(s) = 1, \ \forall \ s \in [-1, 0], \end{cases} \tag{1.49}$$

The solution to the DDE (1.49) for the interval $t \in (0, 1]$ is $y(t) = t + 1$ with the initial condition $y(0) = 1$. It is clear from the solution that $\dot{y}(0^+) = 1 \neq \dot{x}(0^-) = 0$, indicating that the first derivative of $x(t)$ at $t_0 = 0$ is not continuous. The equation (1.48) provides a general framework encompassing various types of differential equations as

(i) If $\tau = 0$, it reduces to the ordinary differential equation:

$$\dot{y}(t) = f(t, y). \tag{1.50}$$

(ii) For discrete delays, the DDE takes the form as

$$\dot{y}(t) = f(t, y(t), y(t - \tau_1), ..., y(t - \tau_n)), \quad \text{where } \tau = \max_{1 \leq i \leq n} \tau_i. \tag{1.51}$$

(iii) The DDE for distributed delay is stated as

$$\dot{y}(t) = \int_{-\tau}^{0} f(t, s, y(t + s))\, ds. \tag{1.52}$$

In the third example, the delay can extend to infinity, yielding the expression:

$$\dot{y}(t) = \int_{-\infty}^{0} f(t, s, y(t + s))\, ds, \tag{1.53}$$

which is associated with the history function $y(s) = \psi_0(s - t_0)$ for all $s \in (-\infty, t_0]$. In understanding the dynamical behavior of neural networks, it is crucial to consider time delays. In previous sections, instantaneous signal transmission in neurons is assumed which is only sometimes realistic. Axons' diverse shapes and sizes can lead to non-instantaneous signal transmission between neurons [36]. To account for this, one can transform the additive model of Hopfield, represented by equation (1.27),

into the following DDE.

$$C_l \frac{dw_l(t)}{dt} = -\frac{w_l(t)}{R_l} + \sum_{j=1}^{n} v_{lj} F_j(w_j(t-\tau_{lj})) + I_l, \text{for } l = 1, 2, ..., n, \tag{1.54}$$

In this equation, τ_{lj} denotes the transmission time from the j-th to the l-th neurons in the network. Such a delay is commonly known as a discrete delay. However, due to the spatial organization and numerous parallel pathways among neurons, neural networks cannot be accurately depicted using discrete delays alone. Instead, there exists a distribution of propagation time delays. Therefore, for a more accurate representation of neural networks, it is preferable to incorporate continuously distributed delays [37, 38]. Equation (1.27) can be re-expressed as a DDE with distributed delays.

$$C_l \frac{dw_l(t)}{dt} = -\frac{w_l(t)}{R_l} + \sum_{j=1}^{n} v_{lj} F_j\left(\int_0^{\infty} w_j(t-s) g_{lj}(u) ds \right) + I_l, \text{for } l = 1, 2, ..., n, \tag{1.55}$$

In the previous equation, the variable s denotes a signal delay from the j-th to the l-th neuron, characterized by a probability distribution function $g_{lj}(u)$ and a mean delay $\tau_{lj} = \int_0^{\infty} s g_{lj}(s) ds$. A DDE integrating discrete and distributed delays is termed a DDE with mixed-time delays. The evolution of the state variable w_l is influenced by w_j over the entire past interval $(-\infty, t]$. In recent years, several publications [41, 42] have presented results and insights about delayed neural networks.

1.5 Synchronization

The term "synchronization" finds its roots in the Greek words "Syn", meaning together, and "Chronos", meaning time. In its literal interpretation, synchronization

denotes the sharing of standard time. However, within the realm of dynamical system theory, synchronization takes on a more nuanced meaning, referring to the alignment of oscillations concerning time. Chaos, a pivotal phenomenon in nonlinear dynamical systems, is characterized by its pronounced sensitivity to initial conditions. Extensive research, as evidenced by various articles such as [44, 45], has underscored the chaotic nature inherent in neural networks. Achieving synchronization in chaotic systems poses a formidable challenge due to their extreme sensitivity to initial conditions. Even minute discrepancies in initial states between identical systems lead to rapid divergence over time, resulting in the loss of practical synchronization. Despite these obstacles, recent endeavors have put forth strategies for attaining synchronized behavior in chaotic systems. Pioneering work by Pecora and Carroll (1990) [46] introduced the concept of a response system aligning itself with a driver system. While earlier studies primarily focused on single-driver systems steering a response system, the insights garnered from these rudimentary setups may not fully encapsulate the dynamics in systems featuring multiple independent driver systems vying for synchronization with a shared response system. Here are descriptions of some of the various types of synchronization reported in the literature [47, 48].

(1) **Exact Synchronization**: This type synchronization arises when systems are identical and coupled either unidirectionally or bidirectionally. Let us examine two coupled identical systems described as follows:

$$\dot{z}(t) = f(z(t)), \tag{1.56}$$

$$\dot{y}(t) = f(y(t)) + U(z(t), y(t)), \tag{1.57}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is a continuous vector field and $U(z, y)$ is a coupling

term. The systems (1.56) and (1.57) are said to be completely synchronized if $\|y(t) - z(t)\| \to 0$ as $t \to \infty$.

(2) **Approximate or Quasi Synchronization**: Consider the coupled systems as follows:

$$\dot{z}(t) = f(z(t)), \tag{1.58}$$

$$\dot{y}(t) = g(y(t)) + U(z(t), y(t)), \tag{1.59}$$

where $g : \mathbb{R}^n \to \mathbb{R}^n$ is a continuous vector field. The systems (1.58) and (1.59) are said to be synchronized in a quasi way if there exists $T > 0$ such that $e(t) \in D = \{e(t) : \|e(t)\| < \epsilon\}$ for all $t > T$, where $e(t) = y(t) - z(t)$ and ϵ is a synchronization error bound.

(3) **Projective Synchronization**: In projective synchronization, the response system is synchronized with the drive system up to a scaling factor. The error system is defined as

$$e(t) = y(t) - \beta z(t), \tag{1.60}$$

where $\beta \neq 0$ is a scaling factor, and $\|e(t)\| \to 0$ as $t \to \infty$.

- Generally, if β is replaced by a diagonal matrix $\Omega = \{\beta_1, \beta_2, ..., \beta_n\}$ with constant elements $\beta_j \neq 0 \ \ \forall j$, it is called modified projective synchronization.
- If β is replaced by a diagonal matrix $\Omega(t) = \text{diag}\{\beta_1(t), \beta_2(t), ..., \beta_n(t)\}$, where $\beta_i(t) \neq 0$ is a bounded and continuously differentiable function for all i, it is called modified function projective synchronization (MFPS).

- If $\beta_1(t) = \beta_2(t) = ... = \beta_n(t)$, then it is called function projective synchronization.
- If $\beta = -1$, then it denotes anti-synchronization, where the phases of the signals exhibit a 180-degree difference.

(4) **Quasi Projective Synchronization**: Quasi-projective synchronization is a type of synchronization found in dynamical systems where the systems involved are almost synchronized but with a persistent synchronization error. Unlike complete synchronization, where the systems would align completely, quasi-projective synchronization allows for a small, constant error between the systems. i.e, $\lim_{t\to\infty} \|y(t) - \beta z(t)\| \leq \epsilon$, then the master and drive systems are quasi projective synchronized where ϵ is a synchronization error bound. This error does not reduce over time and can even fluctuate.

Various synchronization schemes have been reported in the literature, including anticipated, phase, and generalized synchronization; see [49, 50, 51] and the references cited therein.

1.6 Hypercomplex neural networks

Research on hypercomplex neural networks (HCNNs) commenced as early as the 1970s, with notable contributions by Aizenberg [52]. However, significant advancements in this field began in the late 1980s and early 1990s, marked by pioneering works on phasor NNs [53] and complex-valued neural networks (CVNNs) [54, 53]. Hypercomplex numbers encompass various algebraic structures, including complex, hyperbolic, and dual numbers, as well as quaternions, tessarines, octonions, and

other higher algebras like Clifford and Cayley-Dickson algebras [55, 56]. N.N. Aizenberg emphasized that CVNNs, besides adequately handling phase information, offer greater flexibility and reliability compared to their real-valued counterparts [57]. Studies have demonstrated that CVNNs outperform real valued neural networks (RVNNs), particularly in dealing with multidimensional challenges by directly utilizing complex numbers [58]. Significant applications of CVNNs include computer vision, medical diagnosis, remote sensing, and adaptive signal processing [59]. The introduction of a multistate CVNN with Hebbian learning by Jankowski et al. in 1996 [60] laid the groundwork for numerous HCNN developments. Recent studies on multistate CVNNs have focused on networks with symmetric weights [61] and iterative learning rules [62]. Apart from CVNNs, hyperbolic neural networks and quaternion-valued neural networks (QVNNs) are noteworthy examples of HCNNs. In the late 1990s, hyperbolic neural networks exhibited high convergence rates and showed promise in understanding hyperbolic rotations [63]. In addition to CVNN models, Isokawa et al. [64] presented QVNNs utilizing either the projection rule or Hebbian learning. Parcollet et al. [65] have conducted a comprehensive survey of QVNNs, which find applications in various fields such as array processing, secure communication, signal processing, biological systems, 3-D wind prediction and color image processing.

Generally, there are two approaches for dealing with higher-dimensional neural networks: the first one involves splitting multi-dimensional NNs into multiple RVNNs. In contrast, the second one entails reducing higher-dimensional NNs to lower-dimensional neural networks. Many techniques, like the Lyapunov direct method and the energy function technique, which are used in CVNNs, QVNNs, and octonion-valued neural networks (OVNNs), cannot be directly applied to HCNNs due to their non-commutative and non-associative nature for dimensions greater than or equal to

eight. The dynamical features of these multi-dimensional models, related to CVNNs, QVNNs, and OVNNs, are highly complex and mathematically challenging to obtain due to their higher dimension. Across the real field, a hypercomplex number $\hat{u}$ is formulated as

$$\hat{u} = \hat{u}_0 + \hat{u}_1 i_1 + \hat{u}_2 i_2 + \ldots + \hat{u}_n i_n, \tag{1.61}$$

where $n > 0$ is an integer, $\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_n \in \mathbb{R}$ and $i_0, i_1, \ldots, i_n$ are known as hyperimaginary units [55]. It is worth noting that the $(n+1)-$ tuple $(\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_n)$ of real numbers can be used to identify a hypercomplex number $\hat{u} = \hat{u}_0 + \hat{u}_1 i_1 + \hat{u}_2 i_2 + \ldots + \hat{u}_n i_n$. Consequently, the dimension of $\mathbb{H}$ is $(n+1)$ i.e., $\dim(\mathbb{H}) = n+1$, denoting $\mathbb{H}$ as the algebra of hypercomplex numbers.

Consider $\hat{u}, \tilde{u} \in \mathbb{H}$, then addition of two hypercomplex numbers $\hat{u} = \hat{u}_0 + \hat{u}_1 i_1 + \hat{u}_2 i_2 + \ldots + \hat{u}_n i_n$, and $\tilde{u} = \tilde{u}_0 + \tilde{u}_1 i_1 + \tilde{u}_2 i_2 + \ldots + \tilde{u}_n i_n$ is defined as in a component-wise as

$$\hat{u} + \tilde{u} = (\hat{u}_0 + \tilde{u}_0) + (\hat{u}_1 + \tilde{u}_1) i_1 + \ldots + (\hat{u}_n + \tilde{u}_n) i_n. \tag{1.62}$$

Let us define the product of w and $\tilde{u}$, denoted by their juxtaposition, as follows: For each product of the two hyperimaginary units i_α and $i_\beta, \alpha, \beta \in \{1, 2, \ldots, n\}$ we establish a new hypercomplex number

$$i_\alpha i_\beta = \kappa_0 + \kappa_1 i_1 + \ldots + \kappa_n i_n, \tag{1.63}$$

here the selection of real numbers $\kappa_0, \kappa_1, \ldots, \kappa_n$ is exclusively defined by the choice of subscripts α, β. To underscore this dependence in equation (1.63), we denote κ_ρ by $\kappa_{\alpha\beta,\rho}$, where $\rho = 0, 1, \ldots n$.

$$i_\alpha i_\beta = \kappa_{\alpha\beta,0} + \kappa_{\alpha\beta,1} i_1 + \ldots + \kappa_{\alpha\beta,n} i_n. \tag{1.64}$$

The equation (1.63) generates a multiplication table where each entry represents the product of i_α and i_β, corresponding to the hypercomplex number found at the intersection of the $\alpha-$th row and the $\beta-$th column. For instance, Tables 1.2-1.6 provide visual representations of the multiplication tables for complex, hyperbolic, dual, quaternion, and octonion numbers. By applying the distributive law and

$\times$	i_1
i_1	-1

TABLE 1.2: Complex Numbers

$\times$	i_1
i_1	$+1$

TABLE 1.3: Hyperbolic numbers

$\times$	i_1
i_1	0

TABLE 1.4: Dual Numbers

$\times$	i_1	i_2	i_3
i_1	-1	i_3	$-i_2$
i_2	$-i_3$	-1	i_1
i_3	i_2	$-i_1$	-1

TABLE 1.5: Quaternions

$\times$	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
i_0	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
i_1	i_1	$-i_0$	i_3	$-i_2$	i_5	$-i_4$	$-i_7$	i_6
i_2	i_2	$-i_3$	$-i_0$	i_1	i_6	i_7	$-i_4$	$-i_5$
i_3	i_3	i_2	$-i_1$	$-i_0$	i_7	$-i_6$	i_5	$-i_4$
i_4	i_4	$-i_5$	$-i_6$	$-i_7$	$-i_0$	i_1	i_2	i_3
i_5	i_5	i_4	$-i_7$	i_6	$-i_1$	$-i_0$	$-i_3$	i_2
i_6	i_6	i_7	i_4	$-i_5$	$-i_2$	i_3	$-i_0$	$-i_1$
i_7	i_7	$-i_6$	i_5	i_4	$-i_3$	$-i_2$	i_1	$-i_0$

TABLE 1.6: Octonions

referencing the multiplication table, we obtain

$$\begin{aligned}\hat{u}\tilde{u} =&\Big(\hat{u}_0\tilde{u}_0 + \sum_{\alpha,\beta=1}^{n} \hat{u}_\alpha\tilde{u}_\beta\kappa_{\alpha\beta,0}\Big) + \Big(\hat{u}_0\tilde{u}_1 + \hat{u}_1\tilde{u}_0 + \sum_{\alpha,\beta=1}^{n} \hat{u}_\alpha\tilde{u}_\beta\kappa_{\alpha\beta,1}\Big)i_1 \\ &+ \ldots + \Big(\hat{u}_0\tilde{u}_n + \hat{u}_n\tilde{u}_0 + \sum_{\alpha,\beta=1}^{n} \hat{u}_\alpha\tilde{u}_\beta\kappa_{\alpha\beta,n}\Big)i_n. \end{aligned} \tag{1.65}$$

A real number, denoted as $\hat{u} \in \mathbb{R}$, can be represented as a hypercomplex number in the form $\hat{u} = \hat{u}+0i_1+0i_2+\ldots+0i_n \in \mathbb{H}$. Consequently, scalar multiplication can be viewed as a special instance of hypercomplex multiplication, as defined by equation (1.65). For any $w, v \in \mathbb{R}$ and $\hat{u}, a, b \in \mathbb{H}$, addition and multiplication, as defined by equations (1.62) and (1.65) respectively, adhere to the following properties.

(i) $w\hat{u} = w\hat{u}_0 + (w\hat{u}_1)i_1 + \ldots + (w\hat{u}_n)i_n = \hat{u}w.$

(ii) $(w\hat{u})(vx) = (wv)(\hat{u}x).$

(iii) $\hat{u}(a+b) = \hat{u}a + \hat{u}b;\ (a+b)\hat{u} = a\hat{u} + b\hat{u}.$

1.7 Fractional calculus

Fractional calculus extends the principles of differentiation and integration to non-integer orders, departing from the traditional integer-based framework. The phenomena characterized by memory effects, fractal behavior, and long-range interactions can be accurately modelled through fractional calculus. Its applications span diverse fields such as physics, engineering, signal processing, and finance [66, 67, 68]. Techniques like the Riemann-Liouville, Caputo, and Grünwald-Letnikov definitions enable the computation of fractional order derivatives and integrals, broadening the scope of classical differentiation and integration. This mathematical framework

enhances the description of complex systems, making fractional calculus an indispensable tool in scientific and engineering disciplines.

1.7.1 History

On September 30, 1695, a significant exchange occurred between G. W. Leibniz and L'Hopital, marking a pivotal moment in the history of calculus. L'Hopital inquired about a specific notation regarding the n-th derivative of the linear function $f(x) = x$, $\frac{D^n y}{Dy^n}$. In particular, he asked what the result would be if $n = 1/2$. Leibniz's response to this query was profound: "An apparent paradox, from which one day useful consequences will be drawn." This correspondence laid the groundwork for the birth of fractional calculus. Interestingly, this exchange indicates that the concepts of derivatives of both fractional and integer orders were almost simultaneously conceived.

In 1697, G. W. Leibniz addressed a letter to J. Bernoulli and J. Wallis, raising the intriguing prospect of fractional-order differentiation. This concept involves considering differentiation for non-integer values of n.

$$\frac{d^n e^{bx}}{dx^n} = b^n e^{bx}.$$

After Leibniz's passing, several subsequent scholars started to work in this field. Leonard Euler played a significant role in advancing fractional differential calculus within this lineage. Daniel Bernoulli extended the factorial approach $n!$ to encompass non-integer values, a development known as the Gamma function $\Gamma(\cdot)$. Between 1810 and 1819, the French mathematician Sylvestre Francois Lacroix utilized Euler's derivation in his textbook 'Traité du Calcul Différentiel et du Calcul Integral' [69]. Lacroix extended the concept of the derivative from integer order to arbitrary order

α of x^β as

$$\frac{d^\alpha x^\beta}{dx^\alpha} = \frac{\Gamma(\beta+1)}{\Gamma(\beta-\alpha+1)} x^{\beta-\alpha}. \tag{1.66}$$

Joseph Fourier introduced a generalization of differentiation for arbitrary functions in 1822 through his book [70]. Before this development, there had been no efforts to describe physical phenomena using generalized arbitrary derivatives; instead, the focus was primarily on establishing the foundations of fractional differential calculus. Niels Henrik Abel employed fractional differential calculus as early as 1823 to tackle the tautochrone problem [71]. Subsequently, in 1832, J. Liouville introduced two separate definitions of fractional derivatives, with the first definition relying on a series expansion of the function $f(x)$

$$f(x) = \sum_{n=0}^{\infty} C_n e^{b_n x}, \tag{1.67}$$

then, consider the derivative of arbitrary order, denoted by α

$$D^\alpha f(x) = \sum_{n=0}^{\infty} C_n \, b_n^\alpha \, e^{b_n x}, \tag{1.68}$$

A restricted series relies on the order of differentiation for convergence. The second definition of fractional derivative applies to functions in the form of x^{-b} where $b > 0$. In this context, the integral $I = \int_0^\infty u^{b-1} e^{-xu}, du$ is considered. Employing the transformation $xu = t$ yields

$$x^{-b} = \frac{1}{\Gamma(b)} I.$$

By applying the derivative operator D^{α} to both sides of the equation and utilizing equation (1.2), we obtain

$$D^{\alpha}x^{-b} = \frac{(-1)^{\alpha}\Gamma(b+\alpha)}{\Gamma(b)}x^{-b-\alpha}. \tag{1.69}$$

The second definition's drawback lies in its need for more suitability for a broad range of functions.

The fundamental discrepancy between the Lacroix and Liouville definitions of fractional derivatives lies in their treatment of constants. In Lacroix's definition, the fractional derivative of a constant yields a non-zero value, whereas in Liouville's definition, it results in zero [72]. This discrepancy ignited a fascinating debate during the 19th century, as mathematicians debated which definition was more accurate or helpful. B. Riemann, though emerging after this debate, further contributed to the discourse by presenting his definition of fractional derivatives. Liouville's ideas, particularly evident, influenced Riemann's work in his memoirs, where Liouville emphasized ordinary differential equations such as $\frac{d^n y(x)}{dx^n} = 0$, which led to the development of complementary solution given by

$$y(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1}x^{n-1}.$$

Riemann endeavored to determine the solution $y(x)$ of the equation $\frac{d^n y(x)}{dx^n} = f(x)$, where $f(x) \in C[d,e]$, by establishing the condition $y^{(j)}(c) = 0$ for $c \in (d,e)$ and $0 \leq j \leq n-1$. The resulting solution is unique and can be expressed as

$$y(x) = \frac{1}{(n-1)!}\int_c^x (x-t)^{n-1} f(t)dt.$$

By generalizing this concept to non-integer orders represented by α, we arrive at the Riemann-Liouville definition of fractional-order integrals as

$$y(x) = J_x^\alpha f(x) = \frac{1}{\Gamma(\alpha)} \int_c^x (x-t)^{\alpha-1} f(t) dt, \quad \alpha \geq 0,$$

and the fractional order derivative is also referred to as

$$cD_x^\alpha f(x) = \frac{d^n}{dx^n} cJ_x^{(n-\alpha)} f(x).$$

In 1829 Hadamard [73] introduced definitions for both fractional order integral and derivative, followed by Weyl [74] in the year 1917, who formulated a similar definition to the Riemann-Liouville approach, albeit with distinct Kernel function $(t-x)^{\alpha-1}$ and different integration limits. Grunwald [75] and Post [76] later developed a formulation of the fractional derivative as the limit of a sum, drawing upon the classical definition of a derivative. Additionally, Marchaud [77] proposed a fractional derivative approach applicable to any order in the year 1927.

Mittag-Leffler function has played a crucial role in generalizing e^x in fractional calculus given in 1903, while Erdelyi [78] and Kober [79] introduced the Erdelyi-Kober fractional integral, expanding upon Riemann and Weyl integrals. Riesz formulated another fractional integral that is useful in potential theory. M. Caputo coined the term "Caputo fractional derivative" in 1967, derived by first computing the fractional integral and then the ordinary derivative. Other definitions exist [80], and in 1927, H. T. Davis [81], showcased fractional calculus benefits for functional equations, later expanded upon by E. Love [82]. Love's work also extended fractional calculus into the complex order in 1971 [83]. Ross organized the first fractional calculus conference in 1974, preceding the publication of Campos's article on generalizing Weyl and Cauchy integrals [84]. Sanko, Kilbas, and Marichev's monograph, regarded as an "encyclopedia" of fractional calculus, further contributed to its dissemination.

Once seen as purely theoretical, fractional calculus has found applications across various domains, including control systems, finance, physics, and engineering materials like viscoelastic polymers and biological tissues, demonstrating its relevance and expanding scope in recent decades.

1.7.2 Mittag-Leffler function

The Mittag-Leffler function, frequently employed to describe solutions of fractional-order systems, bears similarity to the exponential function, which is standard for characterizing solutions of integer-order systems.
The Mittag-Leffler function with two parameters is defined as

$$\mathbf{E}_{\alpha,\beta}(z) = \sum_{r=0}^{\infty} \frac{z^r}{\Gamma(\alpha r + \beta)},$$

where $\alpha, \beta > 0$ and $z \in \mathbb{C}$. For $\beta = 1$, we have $\mathbf{E}_{\alpha}(z) = \mathbf{E}_{\alpha,1}(z)$ defined as

$$\mathbf{E}_{\alpha}(z) = \sum_{r=0}^{\infty} \frac{z^r}{\Gamma(\alpha r + 1)},$$

where $\alpha > 0$ and $z \in \mathbb{C}$. Particularly $\mathbf{E}_{1,1}(z) = e^z$.

1.7.3 Caputo fractional derivative

Fractional derivatives can be defined in different ways, leading to various results and interpretations. One important type is the Caputo fractional-order derivative, it allows to use initial conditions in terms of integer-order derivatives. This thesis

focuses on fractional-order systems using the Caputo fractional-order derivative because it effectively captures important physical properties, making it very useful for real-world applications.

Definition 1.7.1. [85] The definition of the Caputo fractional derivative of order α for the function $\hat{\phi}(t)$ is

$$ {}_a^C D_t^\alpha \hat{\phi}(t) = \frac{1}{\Gamma(n-\alpha)} \int_a^t \frac{\hat{\phi}^{(n)}(\tau)}{(t-\tau)^{\alpha+1-n}} d\tau, \ t > a, \tag{1.70} $$

where $\alpha \in \mathbb{R}^+$ on the half axis $\mathbb{R}^+$ and $n = \min\{k \in \mathbb{N} : k > \alpha\}$, $\alpha > 0$. If $n = 1$, then $\alpha \in (0, 1)$ and

$$ {}_a^C D_t^\alpha \hat{\phi}(t) = \frac{1}{\Gamma(1-\alpha)} \int_a^t \frac{\hat{\phi}^{(1)}(\tau)}{(\mathbf{t}-\tau)^{\alpha}} d\tau. $$

Property 1. The linearity condition of the Caputo derivative for arbitrary constants c_1 and c_2 is stated as

$$ {}_a^C D_t^\alpha (c_1 \hat{\psi}(t) + c_2 \hat{\phi}(t)) = c_1 \, {}_a^C D_t^\alpha \hat{\psi}(t) + c_2 \, {}_a^C D_t^\alpha \hat{\phi}(t). \tag{1.71} $$

Property 2. The fractional derivative $D_t^\alpha \hat{\phi}(t)$ of a function $\hat{\phi}(t)$ is integrable, i.e.,

$$ I_t^\alpha (D_t^\alpha \hat{\phi}(t)) = \hat{\phi}(t) - \sum_{\mu=1}^{\Omega} [D_t^{\alpha-\mu} \hat{\phi}(t)]_{t=0} \frac{t^{\alpha-\mu}}{\Gamma(\alpha-\mu+1)}. \tag{1.72} $$

****