Chapter 1

Introduction

An artificial neural network is a mathematical model in the form of an electronic device that is developed to design a model so that it exactly performs a specific operation as the brain functions in human body. It is designed on the maxims of the working mechanism of the nervous system of living organisms. In practical use, artificial neural networks with unique equilibrium point are mainly employed to solve optimization challenges. In contrast, artificial neural networks endowed with multiple equilibrium points find diverse applications in various domains, including associative memories, pattern recognition, pattern formation, signal processing, and many more [1, 2, 3]. When deploying neural networks for tasks such as associative memories, pattern recognition, pattern generation, and signal processing, it is essential to ensure that the designed system has tremendous storage capacity to deliver excellent performance. Due to the tremendous storage capacity of multi-dimensional neurons, the analysis of multistability and synchronization of artificial neural networks containing multi-dimensional neurons has become essential from the outlook of the application. In the present thesis, we focused on investigating the problems

of multistability and synchronization of artificial neural networks by selecting multidimentional neurons. Stability analysis is of two types depending on the number of stable equilibrium points. These are as follows:

(a) *Monostability analysis.* In monostability analysis, there exists a unique equilibrium point of the artificial neural network, and any state trajectory in the neighborhood of the equilibrium point converges to that point.

(b) *Multistability analysis.* The concept of "multistability" represents the coexistence of multiple stable equilibrium points. In which we analyze the dynamical behaviour of the system taking into account all the equilibrium points.

This thesis's primary purpose is to study the multistability analysis and synchronization of artificial neural networks with multidimentional neurons such as quaternion and octonion valued neurons. To give a glimpse, some important definitions along with synchronization are listed in this section, which are required to prove the main results. firstly, in this introductory chapter, we present the basic concepts of artificial neural networks from biological neurons. In the following section, we define artificial neural networks and their Architectures. At the end of this chapter we present some algebra of quaternions and octonions which will be of great use in the coming chapters.

1.1 Biological Neuron

The information processing inherent in the human brain is organized by biological processing components working in parallel. It is the parallel work of executing parallel thought processes and acquiring knowledge. Neurons are the basic functional

units of the nervous system that are capable of carrying information to and from different parts of the body in the form of electrical signals. The neuron can be be divided into the three main parts namely: the dendrite, the cell body which is also called the "soma", and the axon.

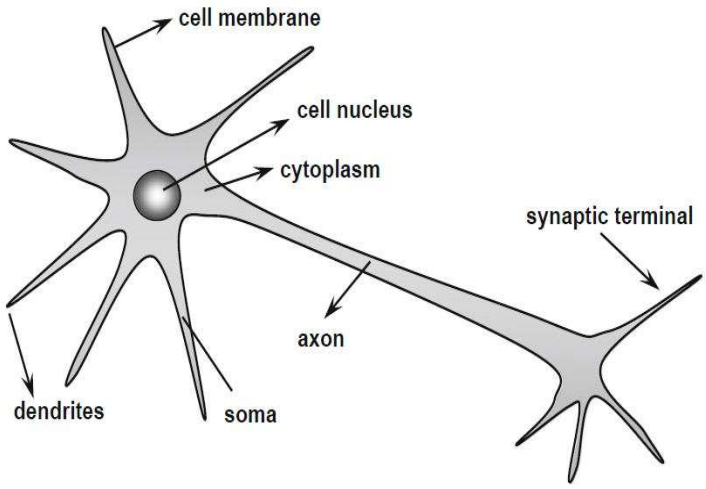


FIGURE 1.1: Structure of a biological neuron.

Dendrites consist of many slender extensions, which collectively form the dendritic tree (see Fig. 1.1). These dendrites serve as receivers, gathering information from other neurons. The nucleus, along with organelles important for cellular function, is located in the central region of the neuron known as the cell body. The cell body takes responsibility for processing all the information coming from the dendrite. The axon is a lengthy fiber-like extension from the cell body that plays a pivotal role in directing the transmission of electrical signals to neighboring neurons or muscle fibers. The terminating part of the axon is also made up of branches known as synaptic terminals. When two or more neurons are interlinked then there are specific connections are formed, these connections are known as synaptic connections. Synaptic connections enable the transmission of electrical impulses through the axon to the dendrites of other interconnected neurons (see Fig. 1.2). It's essential to em-

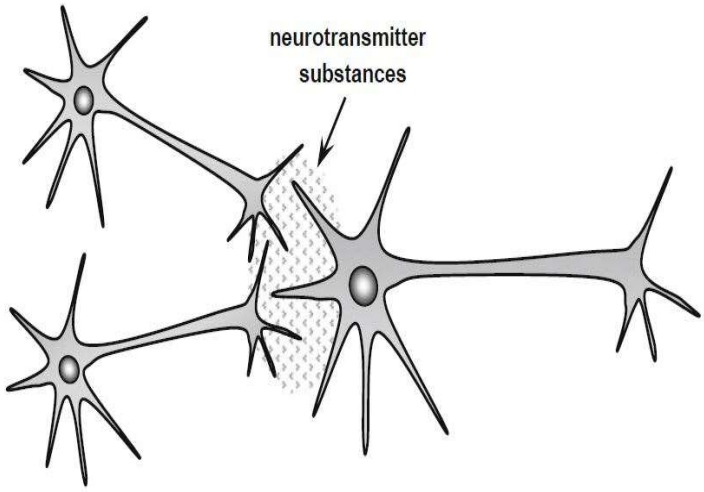


FIGURE 1.2: Visualization of synaptic connections of interconnected neurons.

phasize that there is no direct physical touch exists between the neurons involved in synaptic junctions. Rather, it's the chemical substances namely neurotransmitter elements discharged at these connections are in charge of weighting the transmission between neurons.

1.1.1 Artificial Neural Network

An artificial neural network or simply a neural network, is a biologically motivated device that mimics how biological brain networks perform specific tasks or functions of interest. Artificial neurons are the main processing components of artificial neural networks that are inspired by biological neurons. This model is inspired by how the cell membrane of a neuron transmits and spreading electrical impulses. The artificial neuron is shown in Fig. 1.3 which works on the principle of artificial proposed by Warren McCulloch and Walter Pitts (1943). From Fig. 1.3, we can see that the above artificial neural network consists of the following main elements:

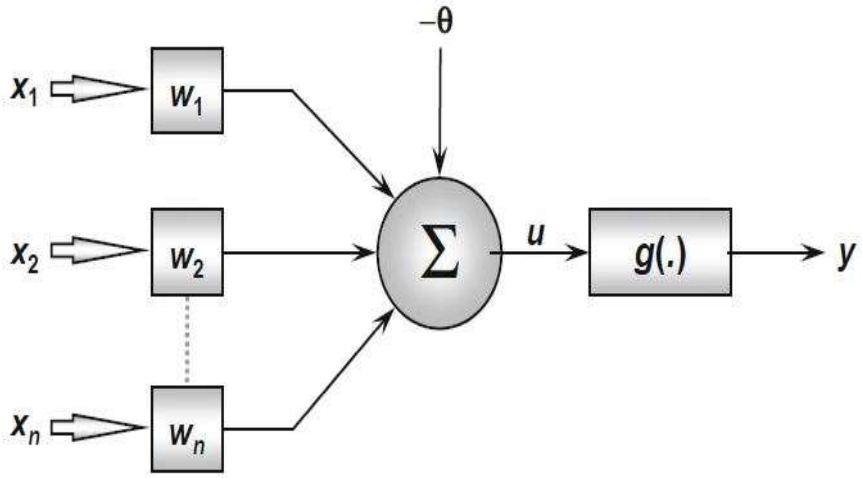


FIGURE 1.3: McCulloch-Pitts model of neural network.

(i): $(x_1, x_2, \ldots, x_n)$ are the Input signal, which are travelling from other neurons or other external resources.

(ii): The weights $(w_1, w_2, ..., w_n)$ are called the synaptic weights for the input variables.

(iii): "$\sum$" is called the linear aggregator, which acts as a linear combiner of inputs, i.e., input signal x_r is multiplied by its respective synaptic weight w_r.

(iv): The variable "θ" is called the bias, which is an external stimulus that can be used to excite the neurons so that the neurons produce outputs.

(v): "u" is the activation potential, which is obtained by subtracting the (θ) from the linear aggregator.

(vi): "g" is the activation function whose mission is to direct the neuron output within a certain range.

(vii): "y" is the final output value generated by the artificial neuron that can serve as input for the next interlinked neurons.

1.1.2 Activation Functions

Activation function is commonly used in artificial neural networks to convert the action potential of previous nodes into an output signal. Activation functions play an important role in the process of keeping the output value within the desired range. An activation function of a artificial neuron can be linear or nonlinear in nature. Now we discuss the following important activation functions which can be very useful in modeling artificial neural networks.

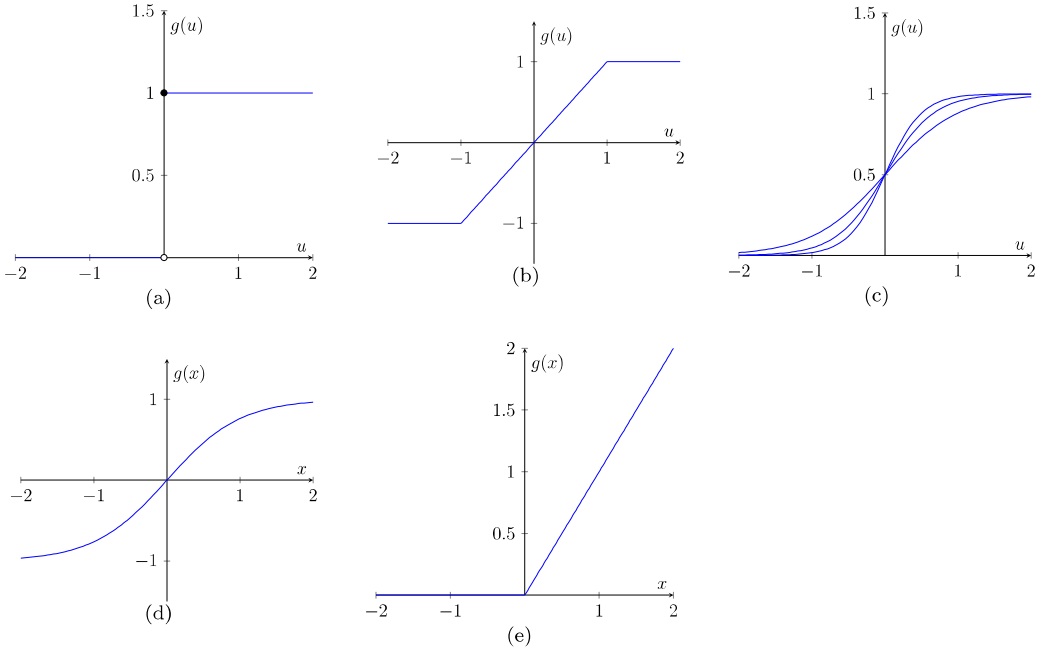


FIGURE 1.4: (a) A Unit step function, (b) Piecewise linear function, (c) Sigmoid activation function for $\beta = 2, 3$, and 4, (d) tanh function, and (e) Relu function.

(a) *Unit step function.* This type of activation function is shown in Fig. 1.4(a). The mathematical expression of this type of function is given

$$y = g(u) = \begin{cases} 1, & 0 \leq u < \infty, \\ 0, & -\infty < u < 0. \end{cases} \tag{1.1}$$

Initially devised for tackling differential equation challenges, the above function has found widespread utility. Presently this function is also being used very rapidly in the fields of biochemistry, neuroscience and in the unusual well testing problems [4].

(b) *Piecewise linear function.* The piecewise linear type function is defined by

$$y = g(u) = \begin{cases} -1 & , \quad u \in (-\infty, -1), \\ u & , \quad u \in [-1, 1], \\ 1 & , \quad u \in (1, +\infty). \end{cases} \tag{1.2}$$

From the 1.2, we can see that the function $g(\cdot)$ is 1 and -1 on the intervals $(-\infty, -1)$ and $(1, +\infty)$, respectively. Moreover, it becomes an identity function on $[-1, 1]$. The function's graph is displayed in Fig. 1.4(b). At present this type of functions are being used very rapidly in the artificial neural networks [5, 6, 7].

(c) *Sigmoid function.* Among artificial neural networks, the sigmoid function has emerged as a leading alternative to nonlinear activation functions. The curve of this type of activation function is S-shaped. This keeps the output signal within the range of 0 and 1. Such type of activation function has been widely implemented in artificial neural networks [8, 9]. The mathematical expression of this type of activation function is given by

$$y = g(u) = \frac{1}{1 + exp(-\beta u)}, \tag{1.3}$$

where, $\beta \in \mathbb{R}$ is a real constant associated with the function slope in its inflection point. It can be observed from the function 1.3 that the sigmoid function asymptotically approaches to 1 or 0 when $u \to \infty$ or $u \to -\infty$.

Typical graph of activation function 1.3 for C=2, 3 and 4 is shown in Fig. 1.4(c).

(d) *Hyperbolic tangent function.* This function falls in the category of hyperbolic functions, it is denoted and defined by

$$y = g(u) = tanh(u) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}}, \quad -\infty < u < \infty. \tag{1.4}$$

The output value given by this activation function falls between -1 and 1. This is similar to sigmoidal function but it is symmetric around the origin, whose graphical representation is as shown in Fig. 1.4(d).

(e) *ReLU function (Rectified linear unit function).* The ReUU function is as shown in Fig. 1.4(e) and it is defined by

$$y = g(u) = \max(0, u) = \frac{u + |u|}{2} = y = g(u) = \begin{cases} u & \text{if } u > 0, \\ 0 & \text{if otherwise.} \end{cases} \tag{1.5}$$

This function has many applications, which is widely used in computer vision [10], Phone recognition [11], and computational neuroscience [12].

1.2 Architectures of an Artificial Neural Network

The architecture of an artificial neural network refers to its structural design, including the arrangement and connectivity of its various components. It consists of three layers namely the input layer, hidden layer, and output layer. These three layers are the major part of neural network architecture. The input layer is the layer of the network whose function is to receive signals from the external environment. Hidden

layers are used to improve the quality of computations in the network, which are located between layers of input and output. All the neurons present in the hidden layers collect weighted inputs and generate outputs through activation functions. The number of hidden layers depends on the quantity and quality of data available to the network, additionally it also depends on the complexity of the proposed network that is designed for the computational task. Finally, the output layer is the final or last layer from where the final output signals comes out through the activation functions. Depending on the interconnection of neurons, there are mainly three types of architectures of artificial neural networks. Based on the interconnections of neurons, the main architectures of artificial neural networks can be classified into the following categories:

1. *Feedforward network with single layers.* Fig. 1.5 represents a feedforward neural network without hidden layers. It is a single layer network consisting of an input layer and an output layer. From the picture we can see that there are 4 inputs in the input layer and 2 outputs in the output layer. Inputs are the source of signals sending towards the neurons of output layer, and not vice versa. In this network the number of output signals is equal to number of neurons. Moreover, in feedforward neural networks, the output of any layer is never the feedback input of other neurons in the same layer.

2. *Multilayer feedforward neural network.* A more complex feedforward neural network is a multilayer feedforward network shown in Fig. 1.6. The network consists of an input layer of 5 neurons, a hidden layer of 3 neurons, and an output layer of a single neuron. The number of output signals in this network is equal to the number of neurons present in the output layer.

3. *Feedback or recurrent neural network.* The diagram of feedback neural network is shown in Fig. 1.7. It is a type of neural network in which output signals

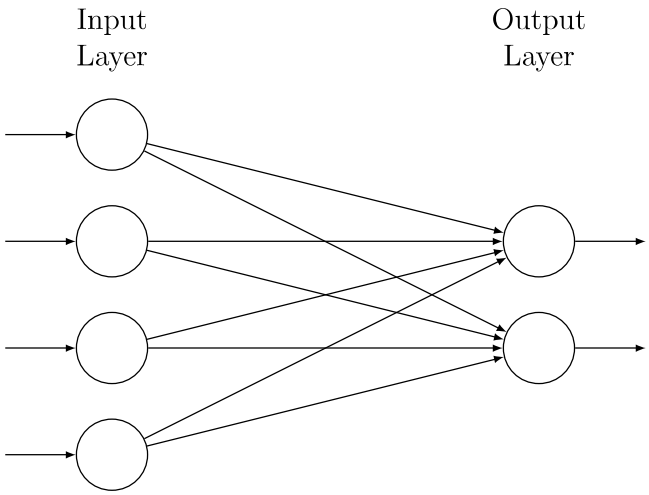


FIGURE 1.5: A feedforward network without hidden layer.

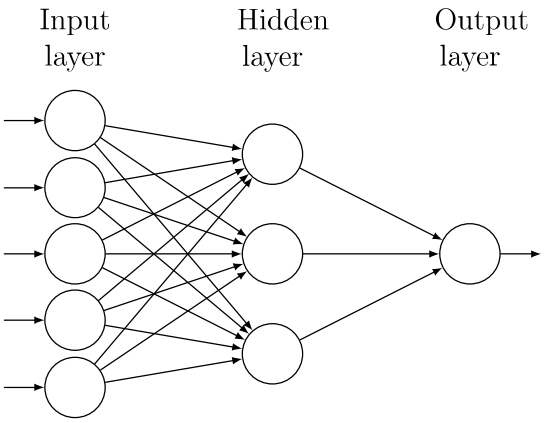


FIGURE 1.6: Multilayer feedforward neural network with one hidden layer.

serve as input signals to other neurons. In the network illustrated in Fig. 1.7, n-neurons are present, with the characteristic feature that each neuron's output signal is recursively fed back into the inputs for the rest of the other neurons. Hopfield neural networks are very popular examples of recurrent neural networks.

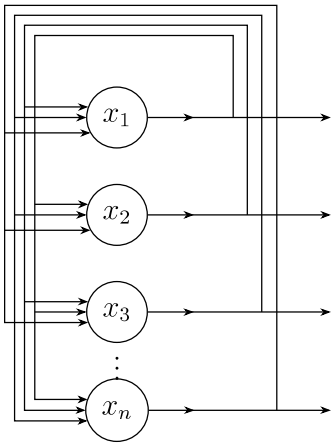


FIGURE 1.7: Feedback neural network with n neurons.

1.2.1 Hopfield Neural Networks.

Hopfield neural networks are a well-known example of recurrent neural networks that are primarily used in pattern recognition, associative memory, and optimization problems. The additive model 1.6 represents the Hopfield neural network without time delay in continuous time [13].

$$\begin{cases} \dot{u_i}(t) = -d \cdot u_i(t) + \sum_{j=1}^{n} W_{ij} \cdot v_j(t) + i_i^b, \text{ with } j = 1, 2, .., n \\ v_j(t) = g(u_j(t)), \end{cases} \tag{1.6}$$

where, $\dot{u}_j(t) = \frac{du_j}{dt}$; W_{ij} is the instantaneous synaptic connection strengths from jth to ith neurons; i_i^b is the bias applied to the ith neuron; $g(\cdot)$ is an activation function; $d \cdot u_j(t)$ is a passive decay term: and $x_i(t)$ is the input signal of ith neuron: $v_j(t)$ is the output signal of the jth neuron. Fig. 1.8 represents the diagram of Hopfield recurrent neural networks, In which all the output signals serve as input signals for each neuron. And besides this it also involves self feedback loop, which means that the output of the i-th neuron also becomes the input of that neuron.

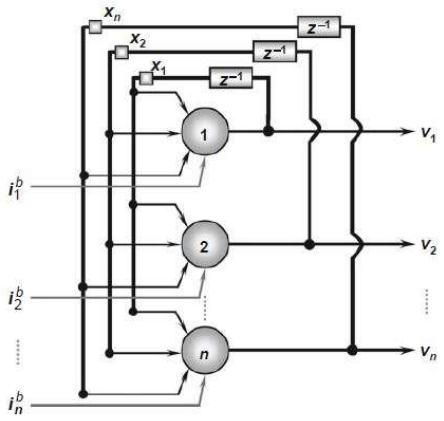
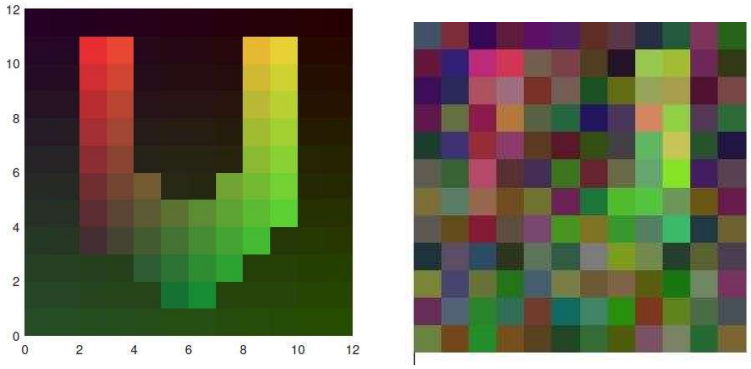


FIGURE 1.8: Diagram of Hopfield neural networks.



(a) (b)

FIGURE 1.9: (a): Original pattern of image the "V", (b): Noisy sample of the original image "V".

1.2.2 Associative Memories

Since the design of the Hopfield network [14], mathematical modeling of storage and retrieval has been a focus of attention in the fields of science and engineering. Associative memory has been a very popular application of Hopfield networks, which is often used to retrieve and remember original pattern with the help of noisy or incomplete information of this original pattern. Fig. 1.9 contains the original color image pattern of "V" and the noise information of this original image. In later chapters, we will discuss associative memory applications for remembering and retrieving

these types of color image patterns using Hopfeld neural networks.

1.3 Training Process or Learning Algorithm of Artificial Neural Network

The training process of artificial neural networks is a collection of general steps to refine the synaptic weights and thresholds of neurons, the purpose of readjustment is to tune the network so that the outputs generated are close to the desired or target values. And this type of refinement or adjustment process is known as learning algorithm. Upon completion of the learning process, the network demonstrates the capability to generate output closely resembling the expected or desired output for any provided inputs. The three main learning algorithms are given in the following subsections:

1.3.1 Supervised Learning

In supervised learning, we have a dataset consisting of input-output pairs. The input is the data we feed into the algorithm, and the output is the corresponding label or target variable. In supervised learning algorithms, we have knowledge about the desired output for a given set of input signals. Supervised learning used in to quantify uncertainties in turbulence and combustion modeling [15], to validation of damage detection [16], for target localization in wireless sensor network [17], and many more.

1.3.2 Unsupervised Learning

Unsupervised learning means "learning without a teacher." Unsupervised learning algorithms differ from supervised learning in that we have no knowledge of the desired output. In this learning algorithm, the system needs to organize itself. In which the system produces clusters of output signals for any given input and identifies clusters presenting similarities. Unsupervised learning used in intrusion detection system [18], anatomic pathology [19], decision making in internet of things [20], etc,.

1.3.3 Reinforcement Learning

Reinforcement learning is a learning algorithm that is a mix between supervised and unsupervised learning. In this algorithm the teacher is present but unable to indicate the desired output, the teacher only tells how close the calculated output is to the desired output.

1.4 Delay Differential Equations

Time delay is the period of time before an event occurs. It also indicates the period of time when two events occur and a pause between them. In the context of telecommunications, there may be a time delay between the sender sending a message and the recipient receiving it. Ordinary differential equations (ODEs) and delay differential equations (DDEs) are both types of differential equations, but they differ in the way they model the evolution of a system over time. Mathematically, DDEs are more complex than ODEs because they involve both current and past values of the

solution. A simple example of the ODE is given by

$$\frac{dx(t)}{dt} = -k \cdot x(t), \tag{1.7}$$

where, $x(t)$ is the unknown function depending upon time t; $\frac{dx(t)}{dt}$ is the derivative of $x(t)$ with respect to time t and k is a decay constant. In the other hand, a simple example of DDE is given by

$$\frac{dx(t)}{dt} = -k \cdot x(t) + a \cdot x(t - \tau), \tag{1.8}$$

where, $x(t)$ denotes the state variable at a time t; $\frac{dx(t)}{dt}$ is the derivative of $x(t)$ with respect to time t; -$k \cdot x(t)$ is a decay term; $a \cdot x(t - \tau)$ is the feedback with time delay, herein $x(t - \tau)$ is the value of the function $x(t)$ at a time τ unit before the current time t.

From equations (1.7) and (1.8), we can see that the derivative of $x(t)$ in the DDE (1.8) is not only dependent on x but it also depends on the previous value of the solution. Nowadays, many researchers have explored the combination of DDEs and neural networks in certain applications, particularly when dealing with systems exhibiting time delays or temporal dependencies. This integration is often done to enhance the ability of the neural network to model and predict time-varying processes.

1.4.1 Types of Delays

Some of the most common delays often used in neural networks depending on the nature of the system are as follows:

1. *Constant delay.* In constant delay, the time delay remains constant over time. In the DDE equation, constant delay is a fixed value τ, where $x(t-\tau)$ represents the state of the system at a time delay τ in the past.

2. *Time-varying delay.* In time-varying delay $\tau(t)$, the time delay is not fixed but it varies with time. This means, time-varying delay is a function of a time variable that varies dynamically with time when it is involved in the system.

3. *Distributed delay.* In distributed delay network, the delay is not concentrated at a single point in time but is distributed over a range. It is often formulated using an integral term, and represents a more continuous delay distribution.

1.5 Stability

Stability analysis of neural networks is an important aspect that can affect the training, performance, and generalization of the model. In this section we will discuss different types of stability definitions of a dynamical system. Before this it is important to know about the equilibrium points of the system, these points are also called fixed points.

1.5.1 Equilibrium Point

Let us consider a system

$$\frac{dx(t)}{dt} = f(x(t), t) \tag{1.9}$$

where, $f(x(t), t)$ is a function of $x(t)$ and t. Then x^* is said to be an equilibrium point (EP) of the system (1.9), if $f(x(t), t) = 0$. It is also a solution of the given

system which is independent of time. A dynamic system may have more than one EP.

An EP x^* is said to be stable if, for any small disturbance from x^*, the system returns to or towards x^* over time.

1.5.2 Stability and Attractivity

An EP x^* of the system is called stable [21] if for any $\epsilon > 0$, there exists a real number $\delta > 0$ such that $\|x(t) - x^*\| < \epsilon$ for $t \geq t_0$, whenever $\|x(\theta) - x^*\| < \delta$ for $\theta \in [t_0 - \tau, t_0]$. Furthermore, x^* is said to be attractive if there exists a time T such that $\lim_{t\to\infty} \|x(t) - x^*\| = 0$, for $t > T$.

1.5.3 Asymptotically Stability

An EP x^* of the system is called locally asymptotically stable [21] if it is stable and attractive.

1.5.4 Exponentially Stability

An EP x^* of the system is called locally exponentially stable [22], if there exists a $\lambda > 0$, and for any $\epsilon > 0$, there exists a $\delta > 0$ such that $\|x(t_0) - x^*\| \leq \delta$ implies that $\|x(t) - x^*\| \leq \epsilon e^{-\lambda t}$ for all $t \geq t_0$.

1.5.5 μ-Stability

Suppose that $\mu(t)$ is a positive real valued continuous function on $\mathbb{R}$ satisfying $\mu(t) \to +\infty$ as $t \to \infty$, and x^* is an EP of the system. Then x^* is called μ-stable [23] if there exist the scalars $\mathbf{M} > 0$ and $\mathbf{T} \geq 0$ such that

$$|x(t) - x^*| \leq \frac{\mathbf{M}}{\mu(t)}, \qquad t \geq \mathbf{T}.$$

We can establish the following stabilities by assuming an appropriate function of $\mu(t)$.

1. *Power stability [23].* An EP x^* of the system is called power stable if there exist real scalars $\mathbf{T} \geq 0$, $\mathbf{M} > 0$ and $\epsilon > 0$ such that

$$|x(t) - x^*| \leq \frac{\mathbf{M}}{t^{\epsilon}}, \qquad t \geq \mathbf{T}.$$

2. *Exponentially stability [23].* An EP x^* of the system is called exponentially stable if there exist real scalars $\mathbf{T} \geq 0$, $\mathbf{M} > 0$ and $\epsilon > 0$ such that

$$|x(t) - x^*| \leq \frac{\mathbf{M}}{e^{\epsilon t}}, \qquad t \geq \mathbf{T}.$$

3. *Log stability [23].* We can obtain the definition of log stability by selecting $\mu(t) = \ln(1 + t)$.

1.5.6 Positively Invariant Set

"positively invariant set" is a collection of points in the state space of a system that, once the system starts inside this set, remains inside the set for all future

times. Mathematically, Ω be a subset of the state space of a dynamical system, and let $x(t, t_0, \phi)$ be any solution with an initial condition ϕ. The set Ω is said to be positively invariant if, for any initial condition $\phi \in \Omega$ the solution $x(t, t_0, \phi) \in \Omega$ $\forall\ t \geq t_0$.

1.5.7 Finite Time Stability

Finite-time stability has become particularly relevant in the design of control systems where it is important to achieve stability and performance objectives within a limited time. It was first introduced in the 1950s. The concept has been used in a variety of fields including robotics, aerospace, and autonomous systems [24, 25, 26, 27].

The EP x^* of the system is said to be finite time stable [28] if for any solution $x(t)$ starting at x_0, it is asymptotically asymptotically stable and $\lim_{t \to T(x_0)} \|x(t)\| = x^*, x(t) = x^*\ \forall\ t \geq T(x_0)$, where the function $T : \mathbf{R}^n \to [0, \infty)$ is called settling time function.

1.5.8 Fixed Time Stability

Fixed time stability is a concept in control theory and dynamical systems that extends the idea of stability to a specific, fixed time period, rather than focusing on asymptotic behavior or convergence over time. The main goal of fixed time stability is to achieve stability within a predetermined, constant time, regardless of initial conditions. It has applications in control systems, robotics, and autonomous vehicles, where guarantees on convergence and stability within specified time intervals are required [29, 30].

The EP x^* of the system is called fixed time stable [28], if it is finite time stable and the settling time is bounded.

1.5.9 Stability Analysis by Lyapunov Functional Method

In control theory and dynamical systems theory, the Lyapunov functional method is one of the valuable techniques to analyze the stability and behavior of systems over time. Let x^* be an EP of the system. A continuously differentiable function $V : D \to \mathbb{R}$ is called Lyapunov functional if

1. $V(x^*) = 0$ and $V(x) > 0$ for all $x \in D - \{x^*\}$,
2. $\dot{V}(x) \leq 0$ for all $x \in D$.

Then $\mathbf{x}^*$ is stable in the sense of Lyapunov. With the above condition "1" if

2(i). $\dot{V}(x) < 0$ for all $x \in D - \{x^*\}$, then x^* is asymptotically stable.

2(ii). $\dot{V}(x) > 0$ for all $x \in D - \{x^*\}$, then x^* is unstable.

Example 1.1. *Let us consider a simple linear system:*

$$\dot{x(t)} = -bx(t),$$

where b is a positive constant. Let us select a Lyapunov functional $V(x) = \frac{1}{2}x^2$. *Clearly* $x^* = 0$ *is an EP of the considered system and* $\dot{V}(x) = -bx^2 < 0$. *By the concept of Lyapunov stability, the considered system is stable at* $x^* = 0$.

1.6 Synchronization

A dynamical system is called chaotic whenever its evolution sensitively depends on the initial conditions. Many researchers have demonstrated the chaotic behavior of neural networks in their articles [31, 32, 33]. The synchronization of chaotic systems is a difficult problem owing to their extremely sensitive dependence on initial conditions. It is a process in which two or several chaotic systems that can be either equivalent or non-equivalent adjust a given property of their motion to a common behavior due to a coupling or a forcing. In the sense of neural networks, it is a modern process of exchanging secret messages. Nowadays synchronization in neural networks is being used extensively in applications like cryptography, secure communication, encryption-decryption etc [34, 35, 36, 37]. Depending on the initial conditions and settlement time, fixed and finite time synchronization have attracted researchers to a great extent.

In finite time synchronization, emphasis is placed on achieving synchronization within a limited and predetermined time. Finite-time synchronization is related to given initial conditions, and different initial values lead to different settling times. On the other hand, fixed time synchronization is a more stringent condition where synchronization is guaranteed to occur within a fixed and predetermined time, regardless of the system's initial conditions. In the fixed time synchronization, the settling time has a uniform upper bound and is independent on the initial conditions.

1.7 Types of Neural Network

Neural networks are generally classified based on their architecture, connection patterns, and the values of their state variables. Based on the architecture and connection patterns, we have already discussed neural network models. Additionally, there are various types of neural network models that are classified based on the value of the state variable. These include Real Valued Neural Networks (RVNNs), Complex Valued Neural Network (CVNNs), Quaternion Valued Neural Network (QVNNs), Octonion Neural Network (OVNNs), and others. In the upcoming chapters of this thesis we will discuss multistability analysis and synchronization of QVNNs and OVNNs. Before discussing QVNNs and OVNNs models, we need to discuss the sets of quaternions and octonions and some of their algebra.

1.8 Quaternion Algebra

In this section, we would like to mention quaternion numbers and some operations on them. Here, the set of quaternions is defined by

$$\mathbb{H} = \left\{ q = q^{(0)} + q^{(1)}i + q^{(2)}j + q^{(3)}k \mid q^{(b)} \in \mathbb{R}, b = 0, 1, 2, 3 \right\}, \tag{1.10}$$

where $q^{(0)}$ is the real part of q and i, j and k are the three imaginary units, which satisfy the following conditions:

$$i^2 = j^2 = k^2 = -1; ij = k = -ji; jk = i = -kj; ki = -ik = j. \tag{1.11}$$

Now, for any $z, h \in \mathbb{H}$, where $z = z^{(0)} + z^{(1)}i + z^{(2)}j + z^{(3)}k, h = h^{(0)} + h^{(1)}i + h^{(2)}j + h^{(3)}k$, we define their addition as follows:

$$z + h = (z^{(0)} + h^{(0)}) + (z^{(1)} + h^{(1)})i + (z^{(2)} + h^{(2)})j + (z^{(3)} + h^{(3)})k. \tag{1.12}$$

According to multiplication rule (1.11) of quaternion units, the product of z and w is defined as

$$\begin{aligned} zh =& (z^{(0)}h^{(0)} - z^{(1)}h^{(1)} - z^{(2)}h^{(2)} - z^{(3)}h^{(3)}) + (z^{(0)}h^{(1)} + z^{(1)}h^{(0)} + z^{(2)}h^{(3)} - z^{(3)}h^{(2)})i \\ &+ (z^{(0)}h^{(2)} + z^{(2)}h^{(0)} - z^{(1)}h^{(3)} + z^{(3)}h^{(1)})j \\ &+ (z^{(0)}h^{(3)} + z^{(3)}h^{(0)} + z^{(1)}h^{(2)} - z^{(2)}h^{(1)})k. \end{aligned} \tag{1.13}$$

Scalar multiplication is defined as

$$\begin{aligned} aq &= a(q^{(0)} + q^{(1)}i + q^{(2)}j + q^{(3)}k) \\ &= aq^{(0)} + aq^{(1)}i + aq^{(2)}j + aq^{(3)}k, \ \forall a \in \mathbb{R}. \end{aligned} \tag{1.14}$$

The conjugate and magnitude of $q^{(0)}+q^{(1)}i+q^{(2)}j+q^{(3)}k \in \mathbb{H}$ is $q^{(0)}-q^{(1)}i-q^{(2)}j-q^{(3)}k$ and $\sqrt{(q^{(0)})^2 + (q^{(1)})^2 + (q^{(2)})^2 + (q^{(3)})^2}$, respectively. And, the derivative of $q(t) \in \mathbb{H}$ with respect to an independent variable t is defined as

$$\dot{q}(t) = \dot{q}^{(0)}(t) + \dot{q}^{(1)}(t)i + \dot{q}^{(2)}(t)j + \dot{q}^{(3)}(t)k \tag{1.15}$$

1.9 Octonion Algebra

Let us assume, the set $\mathbb{O}$ of all octonion numbers is defined as

$$\mathbb{O} = \left\{ y = \sum_{b=0}^{7} y^{(b)} \kappa_b \mid y^{(b)} \in \mathbb{R}, b = 0, 1, ..., 7 \right\},$$

where κ_b are octonion units, $b = 0, 1, 2, ..., 7$.

Let $y = \sum_{b=0}^{7} y^{(b)} \kappa_b \in \mathbb{O}$, and $z = \sum_{b=0}^{7} z^{(b)} \kappa_b \in \mathbb{O}$, then their addition define as:

$$y + z = \sum_{b=0}^{7} (y^{(b)} + z^{(b)}) \kappa_b.$$

Scalar multiplication is defined as

$$cy = \sum_{b=0}^{7} (cy^{(b)}) \kappa_b, \text{where } c \in \mathbb{R}.$$

$\times$	κ_0	κ_1	κ_2	κ_3	κ_4	κ_5	κ_6	κ_7
κ_0	κ_0	κ_1	κ_2	κ_3	κ_4	κ_5	κ_6	κ_7
κ_1	κ_1	$-\kappa_0$	κ_3	$-\kappa_2$	κ_5	$-\kappa_4$	$-\kappa_7$	κ_6
κ_2	κ_2	$-\kappa_3$	$-\kappa_0$	κ_1	κ_6	κ_7	$-\kappa_4$	$-\kappa_5$
κ_3	κ_3	κ_2	$-\kappa_1$	$-\kappa_0$	κ_7	$-\kappa_6$	κ_5	$-\kappa_4$
κ_4	κ_4	$-\kappa_5$	$-\kappa_6$	$-\kappa_7$	$-\kappa_0$	κ_1	κ_2	κ_3
κ_5	κ_5	κ_4	$-\kappa_7$	κ_6	$-\kappa_1$	$-\kappa_0$	$-\kappa_3$	κ_2
κ_6	κ_6	κ_7	κ_4	$-\kappa_5$	$-\kappa_2$	κ_3	$-\kappa_0$	$-\kappa_1$
κ_7	κ_7	$-\kappa_6$	κ_5	κ_4	$-\kappa_3$	$-\kappa_2$	κ_1	$-\kappa_0$

TABLE 1.1: Table of multiplication between octonion units.

Table 1.1 represents the multiplication table [38] between octonion units. With the help of the Table 1.1, the product of $y \in \mathbb{O}$ and $z \in \mathbb{O}$ is defined as

$$yz = (yz)^{(0)}\kappa_0 + (yz)^{(1)}\kappa_1 + (yz)^{(2)}\kappa_2 + (yz)^{(3)}\kappa_3 + ... + (yz)^{(7)}\kappa_7, \tag{1.16}$$

where

$$(yz)^{(0)} = y^{(0)}z^{(0)} - y^{(1)}z^{(1)} - y^{(2)}z^{(2)} - y^{(3)}z^{(3)} - y^{(4)}z^{(4)} - y^{(5)}z^{(5)} - y^{(6)}z^{(6)} - y^{(7)}z^{(7)}, \tag{1.17}$$

$$(yz)^{(1)} = y^{(0)}z^{(1)} + y^{(1)}z^{(0)} + y^{(2)}z^{(3)} - y^{(3)}z^{(2)} + y^{(4)}z^{(5)} - y^{(5)}z^{(4)} - y^{(6)}z^{(7)} + y^{(7)}z^{(6)}, \tag{1.18}$$

$$(yz)^{(2)} = y^{(0)}z^{(2)} + y^{(2)}z^{(0)} - y^{(1)}z^{(3)} + y^{(3)}z^{(1)} + y^{(4)}z^{(6)} - y^{(6)}z^{(4)} + y^{(5)}z^{(7)} - y^{(7)}z^{(5)}, \tag{1.19}$$

$$(yz)^{(3)} = y^{(0)}z^{(3)} + y^{(3)}z^{(0)} + y^{(1)}z^{(2)} - y^{(2)}z^{(1)} + y^{(4)}z^{(7)} - y^{(7)}z^{(4)} - y^{(5)}z^{(6)} + y^{(6)}z^{(5)}, \tag{1.20}$$

$$(yz)^{(4)} = y^{(0)}z^{(4)} + y^{(4)}z^{(0)} - y^{(1)}z^{(5)} + y^{(5)}z^{(1)} - y^{(2)}z^{(6)} + y^{(6)}z^{(2)} - y^{(3)}z^{(7)} + y^{(7)}z^{(3)}, \tag{1.21}$$

$$(yz)^{(5)} = y^{(0)}z^{(5)} + y^{(5)}z^{(0)} + y^{(1)}z^{(4)} - y^{(4)}z^{(1)} - y^{(2)}z^{(7)} + y^{(7)}z^{(2)} + y^{(3)}z^{(6)} - y^{(6)}z^{(3)}, \tag{1.22}$$

$$(yz)^{(6)} = y^{(0)}z^{(6)} + y^{(6)}z^{(0)} + y^{(2)}z^{(4)} - y^{(4)}z^{(2)} + y^{(1)}z^{(7)} - y^{(7)}z^{(1)} - y^{(3)}z^{(5)} + y^{(5)}z^{(3)}, \tag{1.23}$$

$$(yz)^{(7)} = y^{(0)}z^{(7)} + y^{(7)}z^{(0)} - y^{(1)}z^{(6)} + y^{(6)}z^{(1)} + y^{(2)}z^{(5)} - y^{(5)}z^{(2)} + y^{(3)}z^{(4)} - y^{(4)}z^{(3)}. \tag{1.24}$$

and the modulus of $y \in \mathbb{O}$, denoted by $|y|$, is defined as

$$|y| = \sqrt{\sum_{b=0}^{7} \left(y^{(b)}\right)^2}. \tag{1.25}$$

The definition of the derivative of y(t) belonging to the set of octonions $\mathbb{O}$ with respect to time t is as follows:

$\dot{y}(t) = \sum_{b=0}^{7} \frac{d}{dt}(y^{(b)}(t))\kappa_b$.

****