國立交通大學

電子工程學系電子研究所

博士論文

應用在圖形辨識的自發性比例式記憶體細胞非線性網路的穩定度分析

Stability Analysis of Autonomous Ratio-Memory Cellular Nonlinear Network for Pattern Recognition

研究生 ： 蔡夙勇
指導教授 ： 吳重雨

中華民國一百年七月

應用在圖形辨識的自發性比例式記憶體細胞非線性網路的穩定度分析

Stability Analysis of Autonomous Ratio-Memory Cellular Nonlinear Network for Pattern Recognition

研究生 ： 蔡夙勇 Student : Su-Yung Tsai

指導教授 ： 吳重雨 Advisor : Chung-Yu Wu

國立交通大學

電機學院 電子工程學系 電子研究所

博士論文

A Dissertation

Submitted to Department of Electronics Engineering
and Institute of Electronics
College of Electrical and Computer Engineering
National Chiao-Tung University
in partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

in

Electronics Engineering

July 2011

Hsin-Chu, Taiwan, Republic of China

中華民國一百年七月

應用在圖形辨識的自發性比例式記憶體細胞非線性網路的穩定度分析

學生 ： 蔡夙勇　　　　　　指導教授 ： 吳重雨

國立交通大學

電機學院　　　電子工程學系　　　電子研究所

摘要

人腦可以處理對於一些用現今的數位影像方法仍然無法有令人滿意的影像問題，例如影像分割與辨識。一些研究人員相信大腦新皮質以自體聯結的方式回憶出形態。他們指出人腦可快速處理影像問題的原因乃由於人腦直接從記憶中取出答案。

為了處理影像辨識，我們提出自發性比例式記憶體細胞非線性網路 (以下簡稱比例式記憶體網路) 來製作自體聯結的記憶體。比例式記憶體網路是一個非線性系統，它的每個細胞與附近的鄰居細胞彼此連接，這些彼此連接的結構對於在每個細胞上的類比輸入即時訊號一起同時處理。人類視網膜亦具有這兩種特質: 每個細胞與其附近的鄰居細胞彼此連接以及類比輸入即時訊號一起同時處理。這兩種特質使得比例式記憶體網路適合以類比式超大型積體電路來製作。

對於影像辨識，瞭解穩態值及暫態吸引區對於穩定度分析是很重要的事。一開始的影像點將會依其所在的暫態吸引區來收斂到他所對應的穩態值。藉由李雅普諾夫定理，本論文提出一個輸入影像的保守的暫態吸引區。並且藉由李雅普諾夫

定理，本論文提出一個圖形法來建構此暫態吸引區。

從電腦模擬及離散電路執行的觀點來看，找到數值積分法的最大時間步而不引起數值不穩定是極其重要的事。除了數值不穩定外，另外一個議題就是錯誤的記憶點，一些並非黑或白的記憶點可能存在。這些錯誤的記憶點會降低圖形的辨識率。本論文使用擴散模式及特性空間分解法來探討在三種情況下的一維度比例式記憶體網路的歐拉數值積分的最大時間步。這三種情況分別是所有細胞處在線性區，一端細胞進入飽和區，兩端細胞進入飽和區。每一種情況有其對應的擴散模式的邊界條件。從這三種情況的特性空間分解法，我們推導出神經元增益的公式，來降低錯誤記憶點的數目。

此外，由此特性空間分解法可推論出在一個充份條件下，使用歐拉數值積分的比例式記憶體網路的穩態輸出等於類比連續式的比例式記憶體網路的穩態輸出。雖然本論文並未推導二維度的比例式記憶體網路的特性空間分解，我們建議類似的設計原則存在，本論文以一維度及二維度的比例式記憶體網路的例子來支持本論文的設計方程式。

Stability Analysis of Autonomous Ratio-Memory Cellular Nonlinear Network for Pattern Recognition

Student : Su-Yung Tsai Advisor : Chung-Yu Wu

Department of Electronics Engineering
and Institute of Electronics
National Chiao-Tung University

Abstract

Humans brains can resolve many complex image tasks such as pattern recognition and segmentation which are still difficult for digital image processing algorithms to have a satisfying result. Some researchers believe that the neocortex recalls patterns auto-associatively. They pointed out the reason for the brain to efficiently resolve these image tasks is that the brain retrieves the answer from memory.

We propose the ARMCNN (Autonomous Ratio-Memory Cellular Nonlinear Network) structure to implement the associative memory for pattern recognition. The ARMCNN is a nonlinear system with each cell locally coupled to its neighbors. This locally connected structure processes the input real-time analog signals at each cell simultaneously. Human retinas also have these two characterictics : the locally connected structure and the ability to process the input real-time analog signal. These two characterictics make ARMCNNs suitable to implement in analog VLSI.

For pattern recognition, stability analysis is essential to understand the steady state values and the domain of attraction. The initial image point will converge to its corresponing steady state value depending on which corresponding domain of attraction the initial point belong. This thesis aims to provide a conservative domain of attraction for the input image by Lyapunov stability analysis. From this Lyapunov stability analysis, a graphical method is proposed to construct the domain of attraction.

From the view point of computer simulation and discrete-time circuit implementation, it is crucial to obtain the maximum time step in the numerical integration algorithm without causing numerical instability. Besides the numerical instability, another issue is the spurious memory points. Some non-binary equilibrium points may exist. These spurious memory points lower the recognition rate. This thesis uses the diffusion model and the eigenspace decomposition method to examine the maximum forward Euler time step for one-dimensional ARMCNN in three cases: all neurons in linear regions, one end neuron entering the saturation region, and two end neurons entering the saturation region. Each case has a distinct boundary condition for its corresponding diffusion model. From the eigenspace decomposition in these three cases, an analytic neuron gain is derived to lower the amount of spurious memory points.

In addition, the eigenspace decomposition implied a sufficient condition to guarantee that the forward Euler ARMCNN has the same steady state output as the continuous time ARMCNN. Although the eigenspace decomposition for 2-D ARMCNN is not derived, this thesis suggests a similar design principle exists for 2-D ARMCNN. 1-D and 2-D examples are given in this thesis to support the ARMCNN design equations.

誌謝

首先我要對我的指導教授吳重雨教授致上最誠摯的謝意與敬意，感謝他在我博士班期間不厭其煩地給我指導與協助。感謝王啟旭教授，在數學理論上給與我的指導，並指導我正確的研究的態度。感謝王聖智教授，讓我旁聽影像處理的課程，並不時地給我建議，讓我學習到數位影像處理的基本原理。感謝清大陳新教授、鄭桂忠教授，啟發我對仿生電路及金氧半導體次臨界電路的興趣。感謝林大衛教授、桑梓賢教授，在通訊原理及隨機程序上的教導。感謝陳福川教授、蔡中教授常常鼓勵我。感謝王偉彥教授、黃弘一教授、林泓均教授給與論文寶貴的意見。此外我也要感謝交大307實驗室的吳介琮教授、柯明道教授、陳巍仁教授讓我有一個美好的工作環境，感謝307實驗室的學長姐、同學以及學弟妹，你們在這幾年間給與我許多的幫忙與關照，特別是在電路軟體的模擬，在實驗電路的架設，在網球場上，謝謝你們。我也要感謝助理卓慧貞小姐、林明霓小姐、鄭炎莉小姐在行政業務上的幫忙。因為有你們，我的生命更美好。

此外我要特別感謝我的妻子 Debbie 、以及我的家人，因為你們的支持與鼓勵，使我有動力完成這份研究，在此衷心地向上天感恩。最後僅將我的論文獻給我摯愛的父母。

蔡夙勇

國立交通大學
中華民國一百年七月

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Some basic image tasks such as image segementation, recognition and associatvie memory [1, 2] are difficult to have a satisfying result using digital image processing algorithms, while these basic image tasks are considered easy for huamn beings to perform. For example, if a cat is hidden partially in a picture, and one is asked to push the button to answer whether the cat is in the picture, it takes less than half a second to identify this cat. To quote Jeff Hawkins from [3], "... From the time light enters your eye to the time you press the button, a chain no longer than one hundred neurons could be involved...". On the other hand, it is difficult for a digital computer to excute one hundred steps to identify which kind of animal is in a picture. Even with many parallel computers, each computer still need billions of steps to accomplish its own task. According to Jeff Hawkins, this is because the brain "retrieves the answers from memory" [3]. Therefore this thesis focuses on biology-inspired circuits such as cellular neural (nonlinear) network (CNN) [4, 5] to perform the task of associative memory. The locally connected structure in CNN, also observed in the human retina, reduces the wire connections in analog VLSI.

From the mathematical point of view, CNN are coupled nolinear differential equations. Given initial conditions and a set of input image data, each trajectory must converge and propagate to some equilibrium point based on the coupled nolinear differential equations. To properly perform an image task, such as pattern recognition, CNNs are required to be completely stable. A circuit is completely stable if every trajectory tends to some equilibrium point [6]. Many equilibrium points in a CNN are possible. As Chua stated [7]: “Patterns are associated with complete stable circuits where each trajectory corresponding to a given initial condition must necessarily converge to some dc equilibrium point, usually among many others, as in cellular neural networks.” Therefore, understanding steady state values and the stability property is essential. However, previous studies in [8, 9, 10, 11, 12] do not guarantee that neuron states will converge to the equilibrium states. In [13, 14], the analysis of equilibrium points is mainly based on the standard activation function [4] and the space-invariant templates whose center element is larger than 1. Hence, this thesis seeks to find a domain of attraction for ARMCNN (Autonomous Ratio-Memory Cellular Nonlinear Network) solving the tasks of associative memory and pattern recognition.

Discrete time CNN is equivalent to Euler-integrated CNNs [15, 16]. A key question to discrete time implementation and computer simulation is numerical stability. Hence, from the digital emulation and the discrete implememtation view points, it is crucial to know the maximum time step in the numerical integration algorithm without causing numerical instability. In literature, locally regular (LR) CNN is well known to have a maximum forward Euler (FE) time step equal to the internal time constant of the CNN cell [17]. In addition, the equilibrim point is not changed after this time step. An interesting question arises as whether other CNNs have this property or not. Hence, this thesis aims to obtain the maximum time step by linking the ARMCNN dynamic equations into a diffusion model and using eigenspace decomposition for the ARMCNN functioned as associative memory. From the eigenspace decomposition, an analytic gain is derived to lower the amount of spurious memory points. Furthermore, the decomposition implied a sufficient condition to guarantee that the forward Euler ARMCNN has the same binary output as the continuous time ARMCNN. Although we do not derive the eigenspace decomposition for 2-D ARMCNN, this thesis suggests that a similar design principle also exists for 2-D ARMCNN. 1-D and 2-D examples are given to support the ARMCNN design concepts.

1.2 Organization

The organization of the thesis is described as follows.

Chapter 2 is the overview of the standard CNN. Relevant definitions are discussed.

Chapter 3 describes the operation, algorithm and dynamic state space equations of ARMCNNs.

Chapter 4 deals with the Lyapunov stability analysis and the conservative domain of attraction. From this analysis, the activation function and the ratio weight can vary without affecting the stability. The amount of the ratio weight variation is also discussed in this chapter.

Chapter 5 analyzes the maximum time step in the simulation without causing the numerical instability or oscillation. The analysis is based on the eigenspace decomposition of the initial state vector. From this decomposition, the amount of spurious equilibrium points can be reduced by enhancing the low frequency component of the initial state vector. Furthermore, the decomposition implied a sufficient condition to guarantee that the forward Euler ARMCNN has the same binary output as the continuous time ARMCNN.

Finally, conclusions and recommendations for future works are given in Chapter 6.

Chapter 2

Standard CNN

2.1 Introduction

Cellular neural/nonlinear networks (CNNs) are nonlinear continuous time operating structures composed of locally connected cells in an array. Locally connected structures are also found in the silicon retina which has been modeled as a resistive grid by Mead [18]. Each CNN cell receives the input image simultaneously and processs the image signal in the continuous time domain. The processed image signal propagates away from the locally connected cell. The connected weight between each neighboring cell is govered by simple templates.

Locally connected structures and the continouous time signal processing capability make CNNs suitable for analog VLSI implementation and image processing tasks. [19]. Among various image processing tasks, linear and nonlinear image filtering applications were demonstrated using CNNs as resistive grids in [20]. Many other CNN image processing examples such as connected component detector are available in [21, 22].

This chapter is organized as follows. Section 2.2 provides the mathematical definitions for the standard CNN. Image processing examples are illustrated by simple templates in Section 2.3.

2.2 Standard CNN State Equations

A standard CNN [4] consists of an $M \times N$ rectangular array with the cell or neuron $C(i,j)$ located at the ith row and jth column. The dynamics of the standard CNN [4] is governed by

$$\begin{aligned}
&C\frac{dv_{xij}}{dt} = -\frac{v_{xij}}{R_x} + \sum_{C(k,l)\in N_{Ra}(i,j)} A(i,j;k,l)v_{ykl} + \sum_{C(k,l)\in N_{Ra}(i,j)} B(i,j;k,l)v_{ukl} + z \\
&R_xC\frac{dv_{xij}}{dt} = -v_{xij} + \sum_{C(k,l)\in N_{Ra}(i,j)} R_xA(i,j;k,l)v_{ykl} + \sum_{C(k,l)\in N_{Ra}(i,j)} R_xB(i,j;k,l)v_{ukl} + R_xz \\
&\tau = R_xC
\end{aligned} \tag{2.1}$$

where $v_{xij}, v_{yij}, v_{ukl}, z$ are the state, output, input, and threshold of cell $C(i,j)$, respectively, and τ is the cell time constant. The units for $v_{xij}, v_{yij}, v_{ukl}, z$ are Volt, Volt, Volt and Ampere, respectively. $R_xA(i,j;k,l)$, $R_xB(i,j;k,l)$ are unitless, and R_xz has the unit of Volt. The constraint conditions are

$$|v_{xij}(0)| \leq 1, \quad v_{uij} \leq 1 \tag{2.2}$$

where "black" is coded as $+1$, "white" is coded as -1 and the gray level is between $+1$ and -1. The image can be applied to the CNN in the form of the state $v_{xij}(0)$ or the input v_{uij} of the cell $C(i,j)$, or both. Several definitions regarding (2.1) are listed below:

- Output function (also called activation fucntion)

  The only nonlinear element in Equation 2.1 is the output function, which is also called the activation function, and it has the form of

  $$v_{yij} = \frac{1}{2}(|v_{xij}+1| - |v_{xij}-1|). \tag{2.3}$$

  Figure 2.1 shows the piecewise-linear (PWL) characteristic in (2.3), and the range for v_{yij} is $-1 \leq v_{yij} \leq 1$.

- Sphere of influence of cell $C(i,j)$

  The sphere of influence $N_{Ra}(i,j)$ of cell $C(i,j)$ is defined as

  $$N_{Ra}(i,j) = \{C(k,l)|1 \leq k \leq M, 1 \leq l \leq N, \max(|k-i|,|l-j|) \leq R_a\} \tag{2.4}$$

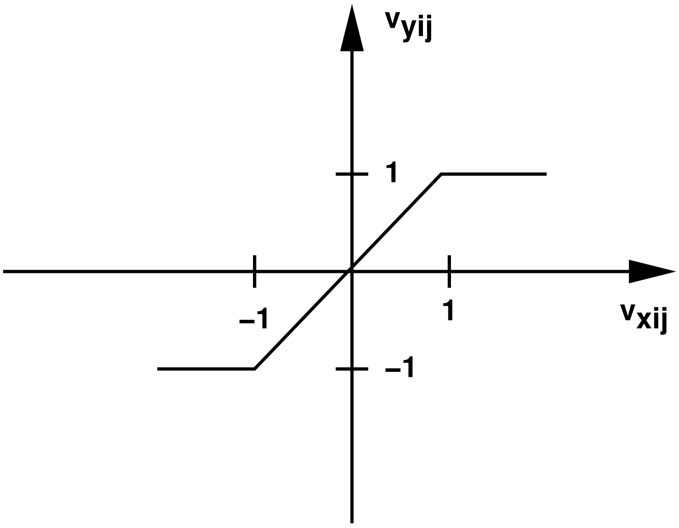


Figure 2.1: The PWL output or activation function.

where R_a is a positive integer. For $R_a = 1$, $C(i, j)$ is connected to its eight nearest neighbor cells $C(k, l)$.

- Space invariant

  A CNN is space invariant if synaptic operators $A(i, j; k, l)$, $B(i, j; k, l)$ and the threshold z do not vary with space. In this case, $A(i, j; k, l)$ and $B(i, j; k, l)$ are written as

$$A(i, j; k, l) = A(k - i, l - j), \ B(i, j; k, l) = B(k - i, l - j).$$

- Virtual cell

  Any cell $C(k, l)$, with $|k - i| \leq R_a, |l - j| \leq R_a$, and $k, l \notin \{1, 2, \ldots, M\}$ is called a virtual cell, and the associated x_{kl}, y_{kl} are called virtual state and virtual output. Virtual values are assigned to these virtual cells.

For example, Figure 2.2 shows a CNN consisting of a two-dimensional (2-D) $M \times N$ rectangulary array with $M = 4, N = 4$ and cells $C(i, j), i = 1, 2, 3, 4, j = 1, 2, 3, 4$. Total 16 cells are in this CNN array. $R_a = 1$ means only one layer of neighboring neurons is connected. For simplicity, only the virtual cells connceted to cell $C(4, 1)$ are shown in the dotted line.

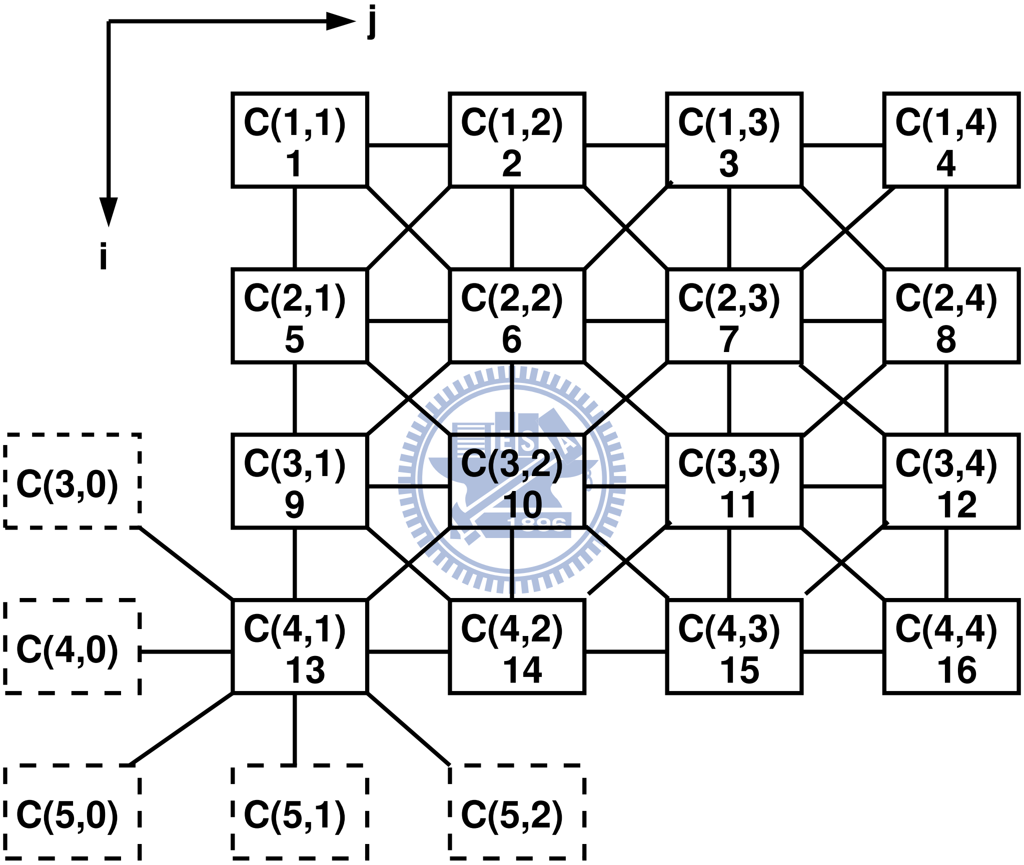


Figure 2.2: A 2-D CNN consisting of a 4 × 4 rectangular array with 4 rows counted in the i direction, and 4 columns counted in the j direction. The radius of the sphere of influence is $R_a = 1$. Virtual cells connceted to cell $C(4, 1)$ are shown in the dotted line

(2.1) implies that the input and the output of the activation fuction are voltage signals. Hence, $A(i, j; k, l)$ and $B(i, j; k, l)$ have the unit of Ω^{-1}. A voltage mode operation example uses several op amps to implement (2.1) in [4]. However, the voltage mode operation has been challenged recently in [23]. Therefore this thesis modifies the original CNN equation, and adopts the current mode approach as shown in chapter 3 for ARM-CNN. Nevertheless, these two approaches have the same normalization form as shown in chapter 3.

2.3 CNN Image Processing Examples

This section shows several CNN processing examples. Although CNN is locally connected, it can have several global image processing abilities. In this section, the test images use the bit map files with eight-bit resolution. The transformation between the read-in byte and the CNN state in (2.1) is detailed in Appendix A. These image processing examples include connected component detection (CCD) and global connectivity detection (GCD). These two tasks can be solved by locally regular (LR) CNNs [17], and can be numerically integrated with a step size τ [24, 17]. Many binary image tasks can be processed by LR-CNNs.

Connected component detection (CCD) and global connectivity detection (GCD) are simulated in this chapter with a forward Euler step size of τ, where $\tau = R_x C$. More CNN templates for various image tasks are available in [25].

2.3.1 Connected Component Detection (CCD)

CCD counts the number of contiguous blocks in the horizontal direction. Each contiguous block is denoted as a black pixel, and shifted to the right. The A,B and z templates for the CCD task are shown as,

$$AR_x = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix}, \; BR_x = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \; zR_x = 0, \qquad (2.5)$$

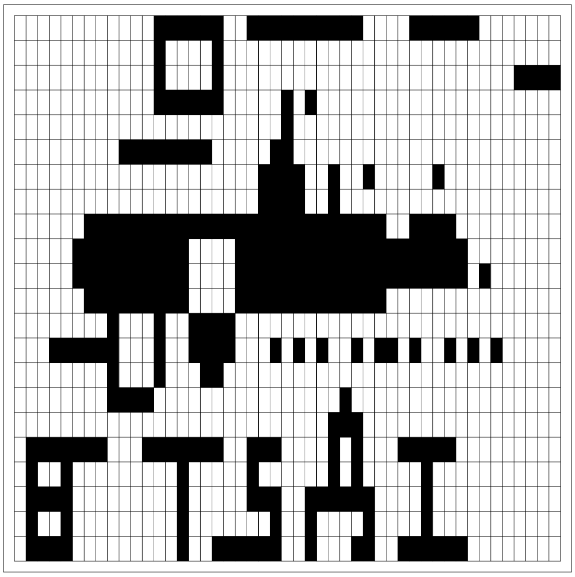


Figure 2.3: Input image applied as $v_{xij}(0)$ before CCD operation.

and the boundary condition for each vitual cell is the fixed type with $y_{kl} = 0$. The test image state $v_{xij}(0)$ for CCD is shown in Figure 2.3. Total simulation time is $T_{sim} = 200\tau$. Forward Euler integration method is applied with each time step equal to τ.

The output image after CCD operation is shown in Figure 2.4. In the second row of Figure 2.3, two blacks are seperated by white pixels. So in the second row of Figure 2.4, two black pixels are found in the right-hand side.

2.3.2 Global Connectivity Dectection (GCD)

GCD finds the contiguous part. Two static input image patterns applied at the input v_{uij} and the state $v_{xij}(0)$ are given. The initial state $v_{xij}(0)$ contains almost the same image as the input v_{uij} except some black pixels in $v_{xij}(0)$ are modified into white ones. These modified pixels can be thought as the igniters, whereas the other unchanged ones are considered as ignitable objects [26]. Therefore, at the steady state, or the end of the fire, only the disconnected part remains. In other words, the global rule is that all connected black pixels of the modified pixles are changed to white or burned. The local rule is that if an output black pixle $C(i, j)$ have a neighboring output white pixel $\hat{C}(i, j)$, and the pixel $\hat{C}(i, j)$ has a black input, then the original output black pixle $C(i, j)$ turns into

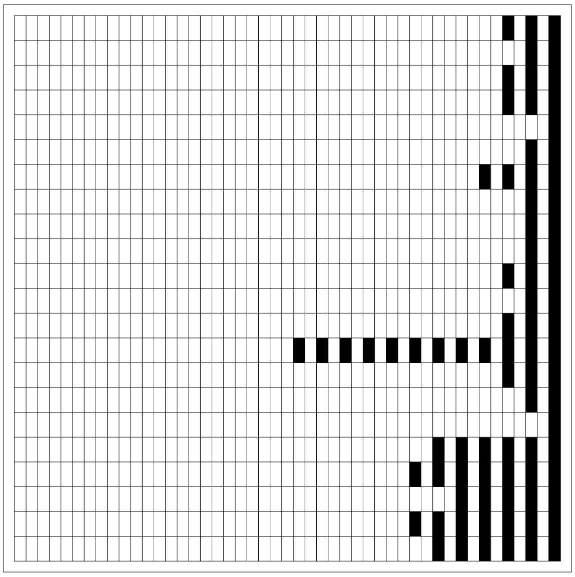


Figure 2.4: Output image, $v_{yij}(200\tau)$, after CCD operation. $T_{sim} = 200\tau$.

a output white pixel. For example, in Figure 2.6, the state at location $(i, j) = (4, 3)$ is black. Consequently, the output pixle $C(4, 3)$ is black. The neighboring cell state at $(3, 3)$ is white. The output is therefore white at $(3, 3)$. From Figure 2.5, the input is black at $(3, 3)$. According to the local rule, the original output black pixle $C(4, 3)$ turns into a output white pixel. It is like the original output black pixle $C(i, j) = C(4, 3)$ gets a fire from the neighboring pixel $\hat{C}(i, j) = C(3, 3)$. This is an interesting result because locally connected CNNs can solve a global connectivity problem [27].

The A,B and z templates for the GCD task are shown as,

$$AR_x = \begin{bmatrix} 0 & 0.5 & 0 \\ 0.5 & 3 & 0.5 \\ 0 & 0.5 & 0 \end{bmatrix}, \; BR_x = \begin{bmatrix} 0 & -0.5 & 0 \\ -0.5 & 3 & -0.5 \\ 0 & -0.5 & 0 \end{bmatrix}, \; zR_x = -4.5\text{V}, \qquad (2.6)$$

and the boundary condition for each vitual cell is the fixed type, $y_{ij} = -1, u_{ij} = -1$.

The input image for the GCD example is shown in Figure 2.5. Two black pixels are modified from Figure 2.5 at location (3,3) and (3,21) and stored as white state values shown in Figure 2.6.

After 50τ, the output is shown in Figure 2.7.

After 150τ, the output is shown in Figure 2.8.

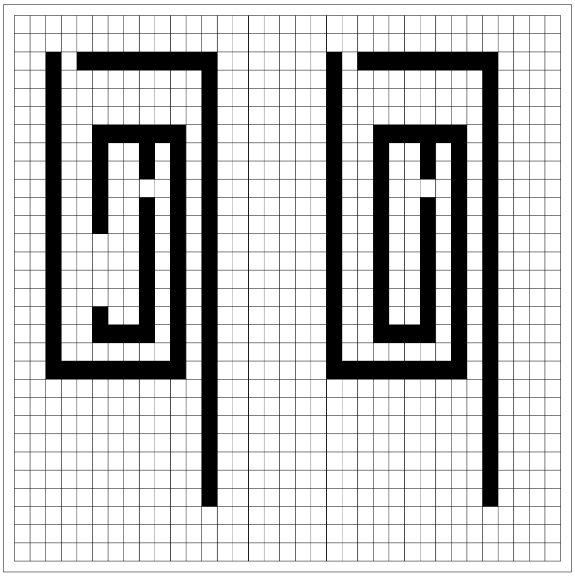


Figure 2.5: The input image v_{uij} for the GCD task.

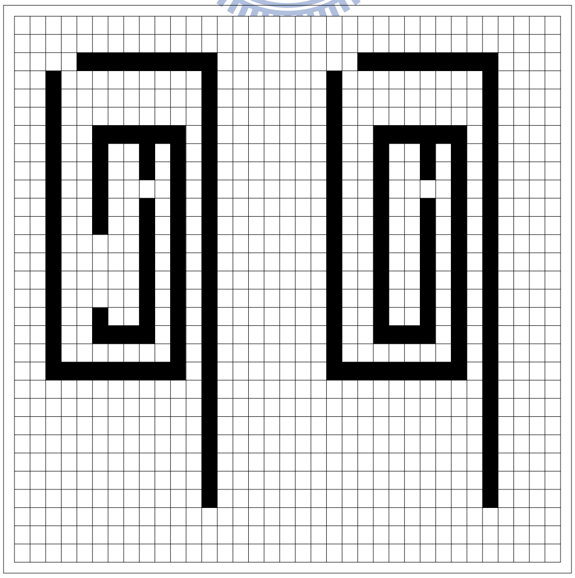


Figure 2.6: The state $v_{xij}(0)$ for the GCD task. Compared with Figure 2.5, the state at $(3,3)$ and $(3,21)$ are white. The coordinates follow the definition in Figure 2.2.

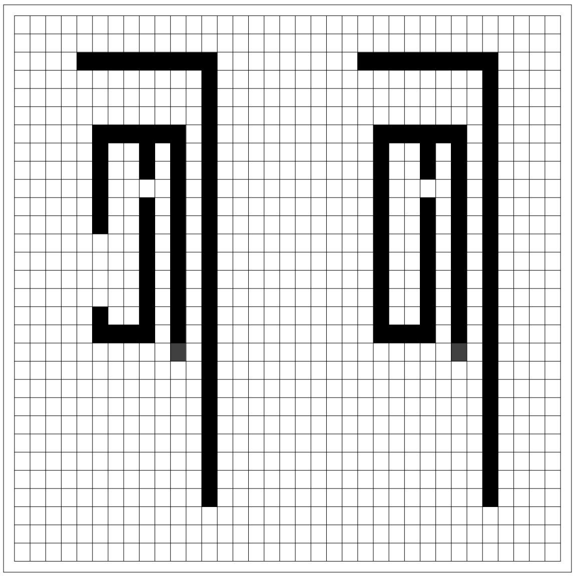


Figure 2.7: The output image, $v_{yij}(50\tau)$, after 50τ.

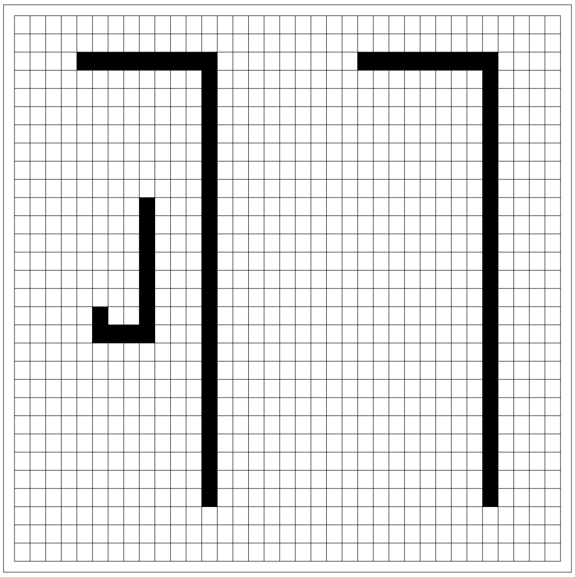


Figure 2.8: The output image, $v_{yij}(150\tau)$, after 150τ.

Chapter 3

ARMCNN

3.1 Introduction

Information storage is called associative memory if it has the capability to recall a partial information [28]. The associative memory can retrieve a stored pattern from the input noisy messed-up pattern. Some researchers believe that our memories correspond to attractors in the brain's huge phase space, since the human brain has more than 10^{12} neurons [29]. From this viewpoint, convergence to an appropriate attractor is called recognition. Hopfield proposed a model of a large fully connected network with symmetrical weights [30] that functions as an associative memory with the capability to recognize patterns. However, the limitation of Hopfield's approach is that it requires fully connected symmetrical weights, which makes it difficult to connect wires in integrated circuits.

Because of its connectivity, researchers consider the Cellular Nonlinear Network (CNN) [4] to be a potential architecture in future nano-electronic systems. To implement the associative memory by CNN, two key aspects of Hebb's postulate, locality and cooperativity [31], are fully exploited in the Ratio Memory CNN (RMCNN) or Autonomous-RMCNN (ARMCNN) [8, 9, 10, 11, 12]. Locality means that the change of synaptic efficacy only depends on local variables, which is the main point of CNN. Cooperativiy [31] means that the presynaptic and postsynaptic neurons must be active simultaneously for a synaptic weight to change, which is the learning rule in RMCNNs or ARMCNNs. This agrees with the correlation property in image data. Pixels located next to each other are more correlated than pixels that are in different regions of the image. The RM weight (or ratio weight) can be determined either through the elapsed time, as in [10, 11], or can be determined without the elapsed time, as in [9, 32].

While previous studies have proposed several different CNNs on associative memory [33, 34, 35, 36, 37], few have actually been implemented in analog VLSI [38], mainly due to complex mathematics required to make this possible. In literature, [33, 34] proposed a design method for the realization of associative memories through singular value decomposition technique (SVD). Another study [35, 36] showed the weights through solving linear matrix inequality (LMI) and generalized eigenvalue (GE) problems, whereas the work in [37] computed the CNN parameters by solving a set of linear equations via pseudoinversion techniques. In particular, the CNNs in [35, 36, 37, 33, 34] require mathematical operations such as SVD, LMI, and pseudoinversion. These operations are more complex than the simple methods proposed in this chapter and in [39], and more difficult to implement in analog VLSI [38].

ARMCNN has two periods of operation: the learning period and the recognition period. In the learning period, ratio weights are learned from the learning patterns. In the subsequent recognition period, the output pattern evolves from the noisy input pattern based on the learned ratio weights.

3.2 ARMCNN Learning Period

This section discusses the ARMCNN consisting of an $M \times N$ rectangular array. Cell $C(i, j)$ is located in the ith row and jth column. Two types of learning are distinguished in ARMCNN, learning with an elapsed time and learning without an elapsed time.

3.2.1 Operation with Elapsed Time

In the learning period, assume that ARMCNN must learn m patterns. The learned weight z_{ijkl} from neuron $C(k, l)$ to neuron $C(i, j)$ can be determined by the Hebbian learning rule [31]

$$z_{ijkl} = \sum_{p=1}^{m} u_{ij}^{p} u_{kl}^{p} \tag{3.1}$$

where u_{ij}^{p} is the learning pixel image at the ith row and jth column of the pth pattern out of m input patterns, and u_{kl}^{p} is the learning pixel image at the kth row and lth column in the set of $N_{Ra}^{0}(i,j)$. $N_{Ra}^{0}(i,j)$ is the set of a Ra-neighborhood system without the neuron $C(i,j)$. The Ra-neighborhood system $N_{Ra}(i,j)$ of the neuron $C(i,j)$ is defined as the set of all neurons including $C(i,j)$ and its neighboring neurons. $N_{Ra}(i,j) = \{C(k,l) | 1 \leq k \leq M, 1 \leq l \leq N, \max(|k-i|, |l-j|) \leq Ra\}$ where Ra is an integer called the radius of the sphere of influence. Besides, the correlation between two neighboring neurons at $C(i,j)$ and $C(k,l)$ is positively correlative if

$$u_{ij}^{p} = u_{kl}^{p}, for p = 1, \ldots, m \tag{3.2}$$

or negatively correlative if

$$u_{ij}^{p} = -u_{kl}^{p}, for p = 1, \ldots, m. \tag{3.3}$$

For example, for a ARMCNN without self-feedback, assume four learning patterns with $+1$ for a black pixel image and -1 for a white pixel image. All possible learned weights z_{ijkl} are $\{+4, +2, 0, -2, 0, -4\}$. The $+4$ occurs when the input image pair at $C(k,l)$ and $C(i,j)$ are positively correlative for all four input patterns. The -4 occurs when the input image pair at $C(k,l)$ and $C(i,j)$ are negatively correlative for all four input patterns. After learning all input patterns, the learned weight z_{ijkl} is transformed into the ratio weight w_{ijkl} [8, 9, 10, 11, 12]

$$w_{ijkl} = \frac{z_{ijkl}}{\sum_{C(k,l) \in N_{Ra}^{0}(i,j)} |z_{ijkl}|}.$$

For a ARMCNN with self-feedback (SARMCNN), simply replace $N_{Ra}^{0}(i,j)$ with $N_{Ra}(i,j)$. If $k = i, l = j$, then z_{ijij} is the self-feedback weight and w_{ijij} is the self-feedback ratio weight. The sum of the absolute values of the ratio weights for a SARMCNN is also equal to one.

In the elapsed period, the inevitable leakage current [11, 40] associated with the stored weight reduces the absolute value of z_{ijkl}. However, this leakage enhances the ratio weights whose absolute values are larger than the average of the absolute values of w_{ijkl} connected to $C(i,j)$ [12]. This feature enhancement effect was shown in [12, 41, 42, 43] as a result of storing weights as ratios.

3.2.2 Operation without Elapsed Time

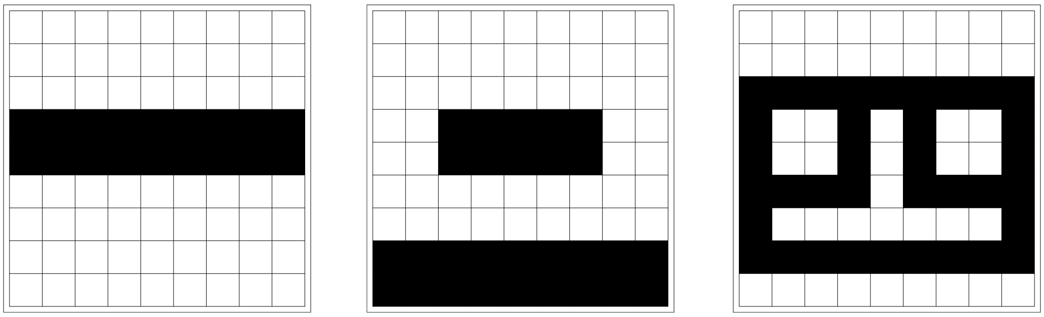
The learning algorithm for determining the ratio weights in ARMCNNs (or SARMCNNs), which requires no elapsed time, is summarized as follows: [32, 39]

- Obtain the weights z_{ijkl} of (3.1) after m patterns are learned.
- Set
  $$G_{max} = \sum_{p=1}^{m} |u^p_{ij}||u^p_{kl}|,$$
  where u^p_{ij} and u^p_{kl} are defined in (3.1).
- Compare $|z_{ijkl}|$ with G_{max}. The weight z_{ijkl} with its absolute value equal to G_{max} will be kept. On the contrary, the weight with its absolute value less than G_{max} will be set to zero.
- Transform the connecting weights of neuron $C(i, j)$ as
  $$w_{ijkl} = \frac{\mathbf{Sgn}(z_{ijkl})}{PN_{N^0_{Ra}}(i, j)} \quad \text{if } |z_{ijkl}| = G_{max} \tag{3.4}$$
  where $PN_{N^0_{Ra}}(ij)$ is the number of preserved weights in $N^0_{Ra}(i, j)$ with respect to neuron $C(i, j)$. $w_{ijkl} = 0$ if $|z_{ijkl}| < G_{max}$.

  For SARMCNNs, simply replace $N^0_{Ra}(i, j)$ with $N_{Ra}(i, j)$ and if $k = i, l = j$, then z_{ijij} is the self-feedback weight.

To simplify wire connections, only those neighboring neurons located on horizontal and vertical positions are connected, that is, up, down, left, right. Neighboring neurons located on diagonal sites are not connected. The algorithm without the elapsed time is used in this thesis. Therefore, all possible ratio weights are $\pm 1/4, \pm 1/3, \pm 1/2, \pm 1$ for a ARMCNN without self-feedback, whereas for a SARMCNN, all possible ratio weights are $\pm 1/5, \pm 1/4, \pm 1/3, \pm 1/2, \pm 1$.

For example, after learning the three Chinese characters in Figure 3.1, the survived ratio weight connection is shown in Figure 3.2. There are one 2-D 18N subsystem (top two rows), one 2-D 25N subsystem (indicated with an arrow), six 1-D 2N subsystems, one 2-D 8N subsystem and two 1-D 9N systems (bottom two rows). The thick bond in



(a) (b) (c)

Figure 3.1: (a) Chinese character ONE. (b) Chinese character TWO. (c) Chinese character FOUR.

Figure 3.2 is due to a negatively correlative input image at the two connected neurons and the thin bond is due to a positively correlative input image at the two connected neurons.

3.3 ARMCNN Recognition Period

3.3.1 ARCMNN State Equations

In the ARMCNN recognition period, the neuron or cell dynamic equation is

$$\frac{dx_{ij}(t)}{dt} = \frac{-x_{ij}(t)}{R_{ij}C_{ij}} + \sum_{C(k,l)\in N_{Ra}(i,j)} \frac{w_{ijkl}y_{kl}(t)}{C_{ij}} \tag{3.5}$$

$$y_{kl}(t) = \sigma(x_{kl}(t))$$

$$\sum_{C(k,l)\in N_R(i,j)} |w_{ijkl}| = 1$$

as shown in Figure 3.3, where ARMCNN variables, x_{ij}, y_{ij}, I_{sat} are represented as electrical signals such as voltages and currents [32, 40] for hardware implementation. x_{ij} is the neuron state voltage, y_{ij} is the neuron output current, I_{sat} is the output saturation current, g is the neuron gain, and R_{ij}, C_{ij} are the resistor and capacitor associated with the neuron $C(i, j)$ respectively. w_{ijkl} is the ratio weight from neuron $C(k, l)$ to neuron $C(i, j)$ and is implemented by current mirrors in integrated circuits [32]. The activation function, $\sigma(\cdot)$ is actually a voltage to current converter (V/I). Activation functions of different types are possible. For example, the piecewise linear (PWL) activation function, having

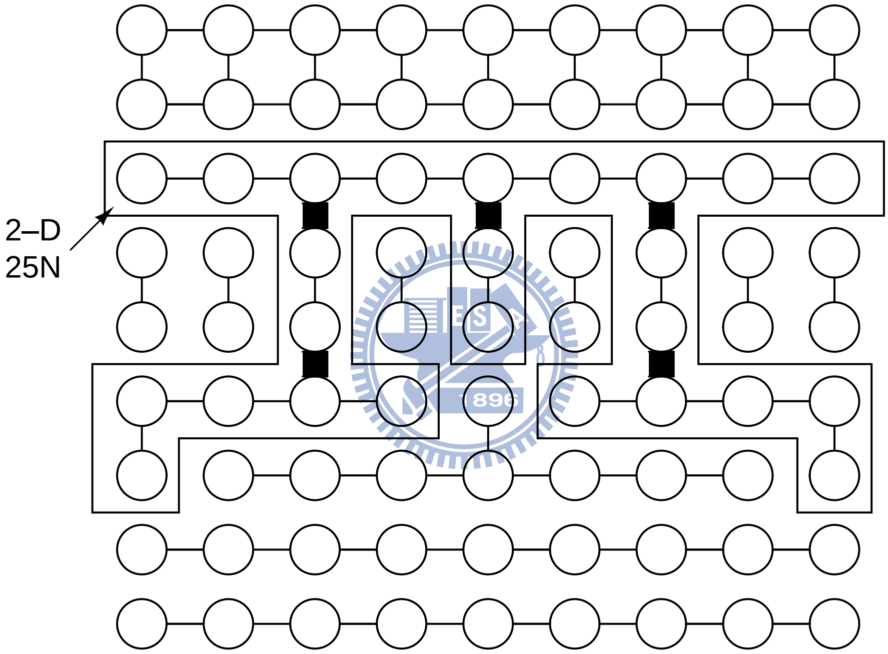


Figure 3.2: The survived ratio weights in the 9×9 ARMCNN after learning the three Chinese characters ONE, TWO, and FOUR of Figure 3.1

a transconductance value G_m of g/R_{ij}, is

$$\begin{aligned}\sigma_p(x_{kl}) &= \frac{g}{2R_{ij}}\left(\left|x_{kl}+\frac{I_{sat}R_{ij}}{g}\right|-\left|x_{kl}-\frac{I_{sat}R_{ij}}{g}\right|\right)\\ &= \frac{G_m}{2}\left(\left|x_{kl}+\frac{I_{sat}}{G_m}\right|-\left|x_{kl}-\frac{I_{sat}}{G_m}\right|\right)\\ &= \frac{gI_{sat}}{2}\left(\left|\frac{x_{kl}}{I_{sat}R_{ij}}+\frac{1}{g}\right|-\left|\frac{x_{kl}}{I_{sat}R_{ij}}-\frac{1}{g}\right|\right) \qquad (3.6)\end{aligned}$$

as illustrated in Figure 3.4. The PWL type is the mostly commonly used one. For PWL activation function, a state x_{ij} is in the linear region if

$$\frac{-I_{sat}R_{ij}}{g} \le x_{ij} \le \frac{I_{sat}R_{ij}}{g},\ y_{ij} = \sigma_p(x_{ij}) = \frac{gx_{ij}}{R_{ij}}. \qquad (3.7)$$

and a state x_{ij} is in the saturation region if

$$\begin{aligned} & x_{ij} > \frac{I_{sat}R_{ij}}{g},\ y_{ij} = I_{sat}\\ \text{or}\quad & x_{ij} < -\frac{I_{sat}R_{ij}}{g},\ y_{ij} = -I_{sat}.\end{aligned}$$

The input initial states (input image) are defined as

$$-I_{sat}R_{ij} \le x_{ij}(0) \le I_{sat}R_{ij}. \qquad (3.8)$$

To perform the ARMCNN stability analysis in chapter 4, an alternative row-wise ordering of the state variables instead of (3.5) is as follows,

$$\begin{aligned}\dot{x} &= -Cx + T\sigma(x)\\ y &= \sigma(x) \qquad (3.9)\end{aligned}$$

where $x \in \mathbf{R}^n$ is the state vector, y is the output vector, and $T = [T_{ij}] \in \mathbf{R}^{n\times n}$ is the connection matrix. $C = \mathbf{diag}[(R_1C_1)^{-1}, ..., (R_nC_n)^{-1}]$, with $R_i, C_i > 0$ for $i = 1, ..., n$, and $\sigma(x) = [\sigma(x_1), ..., \sigma(x_n)]^T$ represents the activation function. The input image, that is, the initial state, satisfies $|x_i(0)| \le I_{sat}R_i$ for $i = 1, \ldots, n$. $n = M \times N$ for a $M \times N$ pattern. For the ratio memory requirement,

$$\sum_{k=1}^{n}|T_{ik}| = \frac{1}{C_i},\ i = 1, ..., n \qquad (3.10)$$

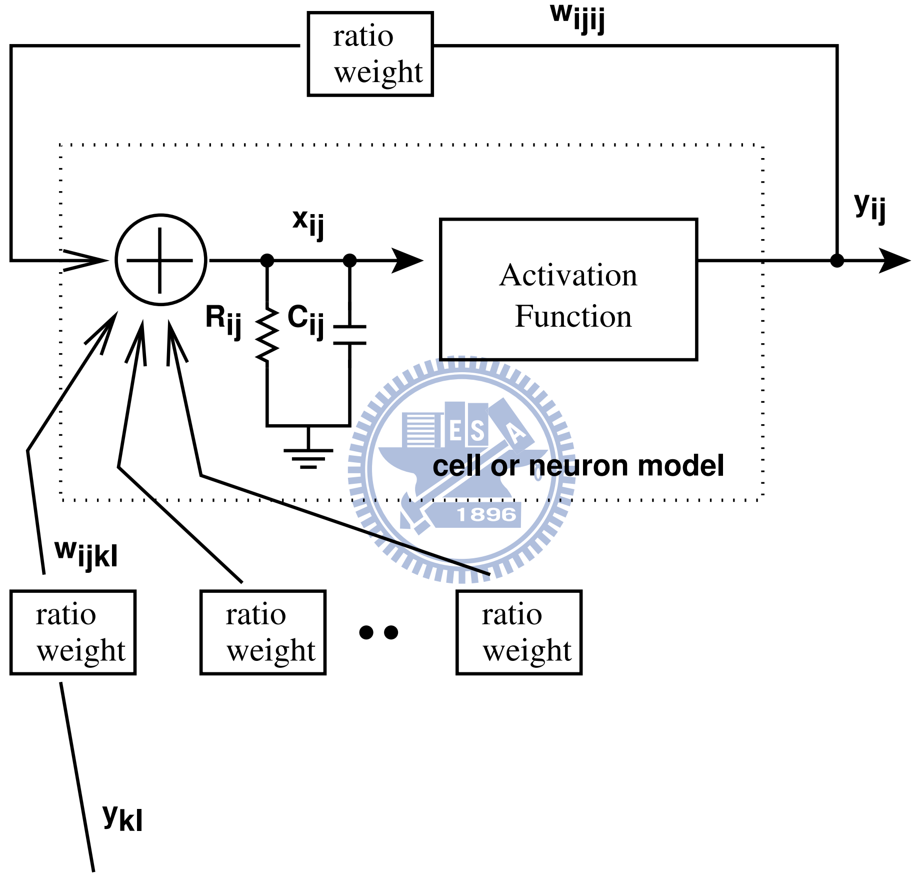


Figure 3.3: The neuron model of ARMCNN located at the ith row and jth column. If without self-feedback, $w_{ijij} = 0$.

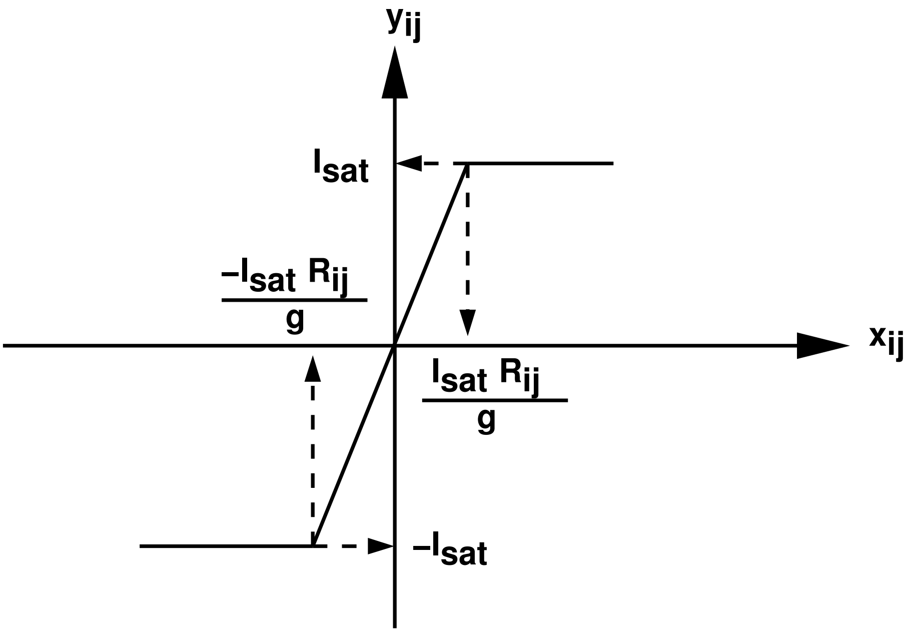


Figure 3.4: The transfer characteristic of the PWL activation function, $\sigma_p(\cdot)$.

where T_{ik} is the ratio weight from neuron k to neuron i multipled by $1/C_i$.

Many other ordering methods for the neurons are possible such as diagonal ordering and column-wise ordeing [44]. The row-wise odering in this chapter represents the two dimensional indices of C_{ab} and R_{ab} by one dimensional C_i and R_i using the following rule:

$$i = (a-1) \times N + b.$$

For example, row-wise ordering of state variables of a 4 × 4 array is shown in Figure 2.2. As seen, neuron (2, 1) in Figure 2.2 is ordered as 5, and the corresponding T is 16 × 16.

3.3.2 Equilibrium Point

The equilibrium points of (3.9) can be obtained by solving

$$Cx_{eq} = T\sigma(x_{eq}),\ x_{eq} \in \mathbf{R}^n. \tag{3.11}$$

In general, we can have q equilibrium points, $x_{eq}^1, \ldots, x_{eq}^q$. The resultant ratio weights generate many small subsystems. Each subsystem has two binary equilibrium points in the form of $(\pm I_{sat}R, \ldots, \pm I_{sat}R)$. The choice of the sign is determined as follows. Set the

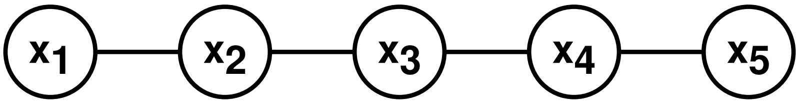


Figure 3.5: A 1-D five-neuron (5N) system having two binary equilibrium points $(x_1(\infty), x_2(\infty), x_3(\infty), x_4(\infty), x_5(\infty))$ as $\pm(I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R)$.

equilibrium state of the first neuron $C(i, j)$ to $I_{sat}R$ $(-I_{sat}R)$. If a negatively correlative input image is found between $C(k, l)$ and $C(i, j)$, then set the equilibrium state of the connected neuron $C(k, l)$ to $-I_{sat}R$ $(I_{sat}R)$. Otherwise, if positively correlative, then set the equilibrium state of the connected neuron $C(k, l)$ to $I_{sat}R$ $(-I_{sat}R)$. This is true for any number of neurons [8]. Further, the ratio memory requirement shown in (3.10) confines the ranges of all possible equilibrium states as:

$$|x_{eq,i}^{mul}| \leq I_{sat}R_i \text{ for } i = 1, ..., n, \qquad mul = 1, ..., q$$

where $x_{eq,i}^{mul}$ is the ith equilibrium neuron state for the multh equilibrium point. This is because, for the ith neuron $(i = 1, \ldots, n)$

$$\frac{1}{R_iC_i}x_{eq,i}^{mul} = \sum_{k=1}^{n} T_{ik}\sigma(x_{eq,k}^{mul}) \leq I_{sat}\sum_{k=1}^{n}|T_{ik}| = \frac{I_{sat}}{C_i}$$
$$\Rightarrow |x_{eq,i}^{mul}| \leq I_{sat}R_i \quad \text{for} \quad mul = 1, \ldots, q.$$

Because q is possible to be greater than 2, some undesired equilibrium points, that is, spurious memory points, may exist besides the two binary equilibrium points. To show spurious memory points, consider a 1-D 5N system shown in Figure 3.5. The neuron resistor $(R_i = R, i = 1, 2, \ldots, 5)$, capacitor $(C_i = C, i = 1, 2, \ldots, 5)$ and voltage to current converter (V/I) are as defined in Figure 3.3 and Figure 3.4.

For the 1-D 5N system in Figure 3.5 under $R_i = 1, C_i = 1,\ i = 1, 2, \ldots, 5$, we see that

$$\begin{aligned}
\dot{x}_1 &= -x_1 + \sigma(x_2) \\
\dot{x}_2 &= -x_2 + 0.5\sigma(x_1) + 0.5\sigma(x_3) \\
\dot{x}_3 &= -x_3 + 0.5\sigma(x_2) + 0.5\sigma(x_4) \\
\dot{x}_4 &= -x_4 + 0.5\sigma(x_3) + 0.5\sigma(x_5) \\
\dot{x}_5 &= -x_5 + \sigma(x_4). \qquad (3.12)
\end{aligned}$$

The thin bond is due to a positively correlative input learning image pair at the two connected neurons. For the PWL activation function with $g > 1$, $(1, 1, 1, 1, 1)$ and $(-1, -1, -1, -1, -1)$ are two equilibrium points as can be verified by substituting these equilibrium points into (3.12). If $g = 2$, then one possible spurious memory point is at $x_{eq}^{1} = [0.877, 0.4389, -0.0611, -0.5611, -1]^T$. Another possible spurious memory point is at $x_{eq}^{2} = [1, 0.5, 0, -0.5, -1]^T$. In chapter 5, these two spurious memory points are eliminated by reducing the neuron gain g. Notice, in this example, $|x_{eq,i}^{mul}| \leq 1$ for $mul = 1, 2$ and $i = 1, 2, \ldots, 5$.

3.3.3 Relationship to Standard CNN

(3.5) is suitable for the current mode approach with the w_{ijkl} implemented using replicas of current mirrors. In addition, if x_{ij} is normalized with respect to $I_{sat}R_{ij}$ and y_{ij} is normalized with respect to I_{sat}, then (3.5) is expressed as,

$$
\begin{aligned}
C_{ij}\frac{d}{dt}x_{ij} &= -\frac{x_{ij}}{R_{ij}} + \sum_{C(k,l)\in N_R(i,j)} w_{ijkl}\sigma_p(x_{kl}) \\
\Rightarrow C_{ij}\frac{d}{dt}x_{ij} &= -\frac{x_{ij}}{R_{ij}} + \sum_{C(k,l)\in N_R(i,j)} w_{ijkl}\left(\frac{gI_{sat}}{2}(|\frac{x_{kl}(t)}{I_{sat}R_{ij}} + \frac{1}{g}| - |\frac{x_{kl}(t)}{I_{sat}R_{ij}} - \frac{1}{g}|)\right) \\
\Rightarrow \tau\frac{d}{dt}(\frac{x_{ij}(t)}{I_{sat}R_{ij}}) &= -\frac{x_{ij}(t)}{I_{sat}R_{ij}} + \sum_{C(k,l)\in N_R(i,j)} w_{ijkl}\left(\frac{g}{2}(|\frac{x_{kl}(t)}{I_{sat}R_{ij}} + \frac{1}{g}| - |\frac{x_{kl}(t)}{I_{sat}R_{ij}} - \frac{1}{g}|)\right) \\
\Rightarrow \tau\frac{d\tilde{x}_{ij}(t)}{dt} &= -\tilde{x}_{ij}(t) + \sum_{C(k,l)\in N_R(i,j)} w_{ijkl}\tilde{\sigma}_p(\tilde{x}_{kl}) \qquad (3.13)
\end{aligned}
$$

where

$$
\begin{aligned}
\tilde{x}_{ij}(t) &= \frac{x_{ij}(t)}{I_{sat}R_{ij}}, \; \tau = R_{ij}C_{ij} \\
\tilde{\sigma}_p(\tilde{x}_{kl}) &= \frac{g}{2}(|\tilde{x}_{kl} + \frac{1}{g}| - |\tilde{x}_{kl} - \frac{1}{g}|) = \frac{\sigma_p(x_{kl})}{I_{sat}} \\
-1 \leq & \tilde{x}_{ij}(0) \leq 1. \qquad (3.14)
\end{aligned}
$$

$\sigma_p(x_{kl})$ is defined in (3.6). Obviously, $\tilde{x}_{ij}$ and w_{ijkl} play the same role as the v_{xij} and $R_xA(i, j; k, l)$ of (2.1) do respectively. The constraint condition on the initial state is also equivalent as seen from (2.2) and (3.14). Hence, it is convinent to use the normalized variables as $\frac{x_{ij}}{I_{sat}R_{ij}}$ and $\frac{\sigma_p(x_{ij})}{I_{sat}}$ in the simulation. Table 3.1 compares the equivalence between

Table 3.1: Equivalent representations between the N-ARMCNN and the S-CNN

	N-ARMCNN	S-CNN
state	$\frac{x_{ij}(t)}{I_{sat}R_{ij}}$	$v_{xij}(t)$
output	$\frac{y_{ij}(t)}{I_{sat}}$	$v_{yij}(t)$
weight	w_{ij}	$R_x A(i,j;k,l)$

the normalized ARMCNN (N-ARMCNN) in (3.13) and the standard CNN (S-CNN) in (2.1).

Chapter 4

Lyapunov Stability Analysis

4.1 Introduction

For pattern recognition, stability analysis is important to understand the behavior near the steady state values. Whether or not initial states can successfully converge to the desired equilibrium point determines recognition performance. However, previous studies in [8, 9, 10, 11, 12] do not guarantee that neuron states will converge to the equilibrium states.

Usually, obtaining the range in which the state converges or obtainging the convergence rate is more meaningful than knowing the stability type of the equilibrium point. Hence, this chapter aims to exploit a Lyapunov theory to obtain the domain of attraction (DOA).

4.2 Stability Definitions

For linear systems, there is no ambiguous definition about stablity. The eigenvalues of the linear system are required to be negative to ensure that the trajectory asymptotically conveges to the equilibrium point. However, when the system is nonlinear, various definitions are available such as stability, unform stability, exponential stability [45].

We consider a nonlinear autonomous system

$$\dot{x} = f(x), \text{ where } f : \mathbf{R}^n \rightarrow \mathbf{R}^n, \; f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_n(x) \end{bmatrix}, \; x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \; \dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \end{bmatrix} \tag{4.1}$$

where x_i is the state variable, and $\dot{x}_i$ denotes the derivative of x_i with respect to the time variable t. Because f does not depend explicitly on time, the system defined by (4.1) is called autonomous. A point or state vector $x_e \in R^n$ is an equilibrium point or an equilibrium state vector of the system if $f(x_e) = 0$. Several relevant definitions of stability in this thesis are as follows [46].

The equilibrium point x_e is

- stable (stability in the sense of Lyapunov)

  if for each ε, there is $\delta = \delta(\varepsilon) > 0$, such that $\|x(0) - x_e\| \leq \delta \Rightarrow \|x(t) - x_e\| < \varepsilon, \forall t \geq 0$

- unstable

  if it is not stable

- locally asymptotically stable (LAS) at x_e

  if it is stable and an $\delta > 0$ exists such that $\|x(0) - x_e\| \leq \delta \Rightarrow x(t) \rightarrow x_e$ as $t \rightarrow \infty$

The above definition of stability implies that the trajectory can be kept arbitrarily close to the equilibrium point by starting sufficiently close to the equilibrium point. This kind of stability is also called stability in the sense of Lyapunov.

For locally asymptotically stability, we need two conditions to hold. The equilibrium point is stable and the trajectories approach to the equilibrium point. When the equilibrium poin is locally asymptotically stable, the next interesting question is how far from the equilibrium point the initial state vector can be and the trajectory from that inital state vector still converges to the same equilibrium point. A domain of attraction of the equilibrium point is the set such that if the initial state vector is located within this set, then the trajectory converges to the equilibrium point. The proof process of the stability about the equilibrium point usually implies a conservative DOA as in Section 4.4.

4.3 Lyapunov Theory

Without actually solving (4.1), Lyapunov theory can be used to decide the stability of the equilibrium point. For (4.1), a typical Lyapunov theorem has the following expression [47],

- if there exists a function $V : \mathbf{R}^n \rightarrow \mathbf{R}$ that satisfies some condition on V and $\dot{V}$, where

$$\dot{V}(x) = \frac{dV(x)}{dt} = \sum_{j=1}^{n} \frac{\partial V}{\partial x_j} \dot{x}_j = \sum_{j=1}^{n} \frac{\partial V}{\partial x_j} f_j(x) = \nabla V(x)^T f(x) \tag{4.2}$$

- then conclusions can be made about the trajectories of (4.1).

Such a function V is called a Lyapunov function, and $\dot{V}(x)$ is a rate of decrease or increase of $V(x)$ along the trajectory.

4.3.1 A Lyapunov Local Asymptotic Theorem

Suppose $x = 0$ is an equilibrium point for (4.1). If a function V satisfies the following,

- $V(0) = 0, V(x) > 0$ in $D - 0$

- $\dot{V}(0) = 0, \dot{V}(x) < 0$ in $D - 0$,

and if $\Omega_c = \{x \in \mathbf{R}^n | V(x) \leq c\}$ is bounded and contained in D, then every point starting in Ω_c remains in Ω_c and approaches the origin. Hence, Ω_c is invariant and is a conservative domain of attraction (DOA) [48]. For example, Figure 4.1 shows an ellipsoid surface to represent $V(x) = x^T Px = c$, where c is a constant. D is assumed to be a sphere as $\|x\| \leq r$. To maximize the DOA, c is chosen as,

$$c = \min_{\|x\|=r} V(x) = \min_{\|x\|=r} x^T Px.$$

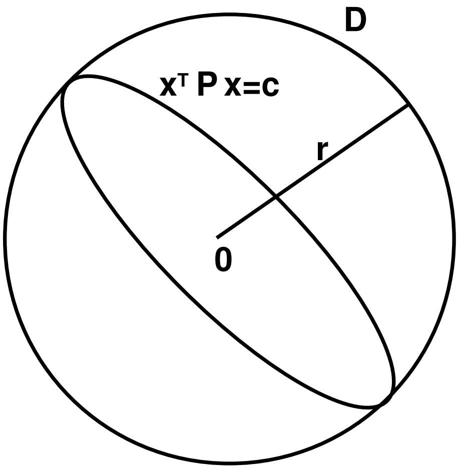


Figure 4.1: $\Omega_c = \{x \in \mathbf{R}^n | V(x) = x^T P x \leq c\}$ is bounded and contained in D.

4.4 ARMCNN Stability Analysis

Without loss of generality, R_i, C_i and I_{sat} in (3.9) are normalized to unities. This is because these parameters only affect the convergent time which is proportional to $1/(R_i C_i)$, and the state values and output currents can be normalized as $x_i/(I_{sat} R_i)$ and y_i/I_{sat}. The normalizations of state values and output currents are shown in (3.13). The normalized binary equilibrium point is expressed as:

$$Eqp = \begin{pmatrix} eq_1 & eq_2 & \cdots & eq_n \end{pmatrix}$$

where eq_i is in the form of ±1.

Consider the behavior around one normalized binary equilibrium point using new state variable z_i. The new state variable z_i with respect to the normalized binary equilibrium point is defined as

$$z_i = x_i - eq_i \qquad \text{for } i = 1, ..., n. \tag{4.3}$$

For instance, if the normalized equilibrium state in the i^{th} neuron is ±1, then

$$z_i = x_i \mp 1.$$

After this transformation from x to z, the new system has an equilibrium point at the origin and is represented as

$$\dot{z} = -z + g(z) \tag{4.4}$$

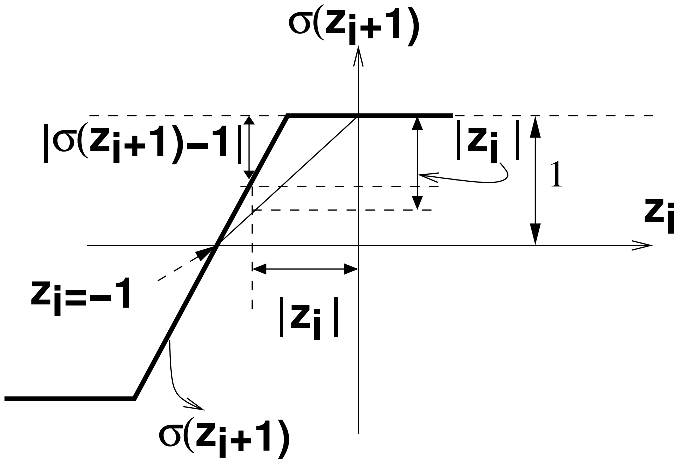


Figure 4.2: The shifted PWL activation function, $\sigma(z_i + 1)$.

where

$$\begin{aligned} g(z) &= [g_1(z) \cdots g_n(z)]^T \\ g_i(z) &= \left[\sum_{k=1}^{n} T_{ik}\sigma(z_k + eq_k) \right] - eq_i. \end{aligned} \tag{4.5}$$

Figure 4.2 shows the shifted PWL activation function of $\sigma(z_i + 1)$, and Figure 4.3 shows the shifted PWL activation function of $\sigma(z_i - 1)$. For the PWL activation, (4.6) and (4.7) must be satisfied in the following stability analysis.

$$\begin{aligned} & |\sigma(z_i + 1) - 1| \leq |z_i| \text{ for } |z_i| < 1,\ \sigma(1) = 1 \\ \text{and} \quad & \lim_{z_i \to 0} \tfrac{|\sigma(z_i+1)-1|}{|z_i|} = 0 \text{ for } i = 1, \ldots, n \end{aligned} \tag{4.6}$$

$$\begin{aligned} & |\sigma(z_i - 1) + 1| \leq |z_i| \text{ for } |z_i| < 1,\ \sigma(-1) = -1 \\ \text{and} \quad & \lim_{z_i \to 0} \tfrac{|\sigma(z_i-1)+1|}{|z_i|} = 0 \text{ for } i = 1, \ldots, n. \end{aligned} \tag{4.7}$$

Theorem 1 *For the PWL activation function defined in Figure 3.4, (4.6) and (4.7) are satisfied if and only if the slope g is greater than* 1.

Proof: From Figure 3.4, it is obvious that $\sigma(1/g) = 1$ for the normalized ARMCNN. The fact that g must be greater than 1 can be argued from the following analysis.

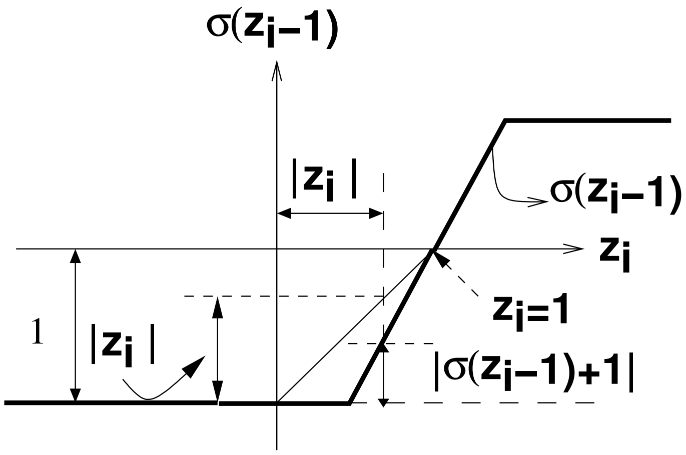


Figure 4.3: The shifted PWL activation function, $\sigma(z_i - 1)$.

1. If $g < 1$, then $1/g > 1$. The PWL activation function in Figure 3.4 is monotonically increasing from $x = 0$ to $x = 1/g$ >1. Therefore,

$$\sigma(0) = 0 < \sigma(1) < \sigma(1/g) = 1$$

which does not meet (4.6), that is, $\sigma(1) = 1$.

2. If $g = 1$, then

$$|\sigma(z_i + 1) - 1| = |z_i + 1 - 1| = |z_i| \text{ for } \quad -1 \le z_i \le 0$$

therefore,

$$\lim_{z_i \to 0^-} \frac{|\sigma(z_i + 1) - 1|}{|z_i|} = \lim_{z_i \to 0^-} \frac{|z_i|}{|z_i|} = 1, \text{ for } i = 1, \ldots, n,$$

which does not meet (4.6), that is,

$$\lim_{z_i \to 0} \frac{|\sigma(z_i + 1) - 1|}{|z_i|} = 0, \text{ for } i = 1, ..., n.$$

3. If $g > 1$, then $|z_i| < 1$ is divided into the following two ranges to show the PWL activation function defined in Figure 3.4 satisfies (4.6).

Range 1: if $-1 < z_i < -1 + 1/g < 0$, then

$$\begin{aligned} -g < gz_i &< -g + 1 \\ |\sigma(z_i + 1) - 1| &= |g \times (z_i + 1) - 1| \\ &= 1 - gz_i - g > 0 \\ |z_i| &= -z_i \\ |z_i| - |\sigma(z_i + 1) - 1| &= -z_i - 1 + gz_i + g \\ &= (g - 1) + z_i(g - 1) \\ &= (g - 1)(z_i + 1) > 0 \end{aligned}$$

$\Rightarrow |\sigma(z_i + 1) - 1| < |z_i|$ for $-1 < z_i < -1 + 1/g < 0$.

Range 2: if $-1 + 1/g \leq z_i < 1$, then

$$\begin{aligned} &\Rightarrow 1/g \leq z_i + 1 < 2 \\ &\Rightarrow \sigma(z_i + 1) = 1 \\ &\Rightarrow |\sigma(z_i + 1) - 1| = 0 \leq |z_i| \\ &\Rightarrow \lim_{z_i \to 0} \frac{|\sigma(z_i + 1) - 1|}{|z_i|} = \lim_{z_i \to 0} \frac{0}{|z_i|} = 0 \end{aligned}$$

In a similar way, $|z_i| < 1$ is divided into the following two ranges to show that the PWL activation function defined in Figure 3.4 satisfies (4.7).

Range 3: if $0 < 1 - 1/g < z_i < 1$, then

$$\begin{aligned} g - 1 < gz_i &< g \\ |\sigma(z_i - 1) + 1| &= |g \times (z_i - 1) + 1| \\ &= gz_i - g + 1 > 0 \\ |z_i| &= z_i \\ |z_i| - |\sigma(z_i - 1) + 1| &= z_i - 1 - gz_i + g \\ &= z_i(1 - g) + (g - 1) \\ &= (g - 1)(1 - z_i) > 0 \end{aligned}$$

$\Rightarrow |\sigma(z_i - 1) + 1| < |z_i|$ for $0 < 1 - 1/g < z_i < 1$.

Range 4: if $-1 < z_i \leq 1 - 1/g$, then

$$\begin{aligned}
&\Rightarrow -2 < z_i - 1 \leq -1/g \\
&\Rightarrow \sigma(z_i - 1) = -1 \\
&\Rightarrow |\sigma(z_i - 1) + 1| = 0 \leq |z_i| \\
&\Rightarrow \lim_{z_i \to 0} \frac{|\sigma(z_i - 1) + 1|}{|z_i|} = \lim_{z_i \to 0} \frac{0}{|z_i|} = 0
\end{aligned}$$

For the PWL activation function, the above case (3) shows that if $g > 1$ then (4.6) and (4.7) are satisfied. And the above case (1) and case (2) show that for the PWL activation function, if (4.6) and (4.7) are satisfied, then $g < 1$ and $g = 1$ are not possible. Therefore, for the PWL activation function defined in Figure 3.4, (4.6) and (4.7) are satisfied if and only if the slope g is greater than 1.

The ARMCNN defined in (3.9) is generated after performing the algorithm in Section 3.2 without elapsed time. The ratio weights of this ARMCNN satisfy (3.10), and the activation functions of each neuron are required to satisfy (4.6) and (4.7). Theorem 2 shows that the resultant ARMCNN converges to one of the binary equilibrium points.

Theorem 2 *ARMCNN stability analysis with a conservative Domain of Attraction (DOA). For the normalized ARMCNN ($R_i = R = 1, C_i = C = 1, I_{sat} = 1$) defined in (4.4) with the activation function satisfying (4.6) and (4.7), there exists a conservative domain of attraction (DOA), i.e., $\|z\| < r$, so that the ARMCNN will converge to one of the normalized binary equilibrium points for $\|z\| < r$.*

Proof: We need to first prove the asymptotic stability of (4.4), that is,

$$\dot{z} = -z + g(z).$$

Define a positive quadratic Lyapunov function

$$V(z) = z^T P z.$$

Choosing $P = 0.5\mathbf{1}_n$, where $\mathbf{1}_n$ is the $n \times n$ identity matrix, we have from (4.2),

$$\begin{aligned}\dot{V}(z) &= \nabla V(z)^T(-z + g(z)) \\ &= 2z^T P(-z + g(z)) \\ &= -2z^T Pz + 2z^T Pg(z) = -z^T z + z^T g(z) \\ &= -\|z\|^2 + z^T g(z) \leq -\|z\|^2 + \|z\|\|g(z)\|.\end{aligned}$$

It is obvious that $\|g(z)\|$ determines the sign of the derivative of $V(z)$. To find $\|g(z)\|$, the range of $g_i(z)$ is analyzed first from (4.5) as

$$\begin{aligned}g_i(z) &= \left[\sum_{k=1}^{n} T_{ik}\sigma(z_k + eq_k)\right] - eq_i \\ &= \sum_{k=1}^{n} T_{ik}\left[\sigma(z_k + eq_k) - eq_k\right] \\ &= \sum_{k=1}^{n} T_{ik} I_k \qquad (4.8)\end{aligned}$$

where

$$I_k \equiv \sigma(z_k + eq_k) - eq_k.$$

The reason for the validity of (4.8) is as follows. Denoting these two opposite equilibrium states as S^1 and S^{-1}: the index set of all neurons with the normalized equilibrium states equal to 1 is S^1 and the index set of all neurons with the normalized equilibrium states equal to -1 is S^{-1}, then

$$\begin{aligned}&\sum_{k=1}^{n} T_{ik}\left[\sigma(z_k + eq_k) - eq_k\right] \\ &= \sum_{k\in S^1} T_{ik}\left[\sigma(z_k + 1) - 1\right] + \sum_{k\in S^{-1}} T_{ik}\left[\sigma(z_k - 1) + 1\right] \\ &= \sum_{k\in S^1} T_{ik}\sigma(z_k + 1) - \sum_{k\in S^1} T_{ik} \\ &+ \sum_{k\in S^{-1}} T_{ik}\sigma(z_k - 1) + \sum_{k\in S^{-1}} T_{ik}. \qquad (4.9)\end{aligned}$$

If the equilibrium state in the ith neuron is 1, then

$$\begin{aligned}&T_{ik} > 0, \;\text{ for }\; k \in S^1 \;\text{ (positivelycorrelative)} \\ &T_{ik} < 0, \;\text{ for }\; k \in S^{-1} \;\text{ (negativelycorrelative)}\end{aligned}$$

and from (3.10), we have

$$\sum_{k\in S^1} -T_{ik} + \sum_{k\in S^{-1}} T_{ik} = -1,$$

so (4.9) is expressed as

$$\begin{aligned}&\sum_{k=1}^{n} T_{ik}\left[\sigma(z_k+eq_k)-eq_k\right]\\ &= \sum_{k\in S^1} T_{ik}\sigma(z_k+1) + \sum_{k\in S^{-1}} T_{ik}\sigma(z_k-1) - 1\end{aligned}$$

which is (4.5).

If the equilibrium state in the ith neuron is -1, then

$$\begin{aligned}T_{ik} &< 0, \ \text{ for } k\in S^1 \ \text{(positively correlative)}\\ T_{ik} &> 0, \ \text{ for } k\in S^{-1} \ \text{(negatively correlative)}\end{aligned}$$

and from (3.10), we have

$$\sum_{k\in S^1} -T_{ik} + \sum_{k\in S^{-1}} T_{ik} = 1,$$

so (4.9) is expressed as

$$\begin{aligned}&\sum_{k=1}^{n} T_{ik}\left[\sigma(z_k+eq_k)-eq_k\right]\\ &= \sum_{k\in S^1} T_{ik}\sigma(z_k+1) + \sum_{k\in S^{-1}} T_{ik}\sigma(z_k-1) + 1\end{aligned}$$

which is (4.5).

Therefore, (4.8) is proved. From (4.8) and equations of (4.6) and (4.7), we have

$$g_i(z) = \sum_{k=1}^{n} T_{ik} I_k, \ \text{ with } \lim_{z_k\to 0}\frac{|I_k|}{|z_k|} = 0 \text{ for } k = 1,\ldots,n. \tag{4.10}$$

This implies that for any ε $(0<\varepsilon<1)$, there exists an r_k, such that

$$|I_k| < \varepsilon|z_k| \quad \text{for} \quad |z_k| < r_k, \quad k = 1,\ldots,n.$$

In fact r_k can be found from Figure 4.2 and Figure 4.3 given the ε value. Let

$$r = \min(r_1, r_2, \ldots, r_n) \tag{4.11}$$

then

$$\text{if } |z_k| < r \qquad \Rightarrow \qquad |I_k| < \varepsilon |z_k| \text{ for } k = 1, ..., n.$$

Or,

$$\text{if } \|z\| < r \Rightarrow |z_k| < r \Rightarrow |I_k| < \varepsilon |z_k| \text{ for } k = 1, ..., n.$$

From (3.10), we can see that

$$\sum_{k=1}^{n} |T_{ik} z_k| \leq \sqrt{|z_1|^2 + |z_2|^2 + \ldots + |z_n|^2} = \|z\|. \tag{4.12}$$

From (4.10), (4.11) and (4.12) and let

$$I_i^* \equiv \sum_{k=1}^{n} |T_{ik} I_k|,$$

the upper bound of the norm of $g(z)$ can be found as

$$\begin{aligned}
\|g(z)\| &= \sqrt{g_1(z)^2 + \cdots + g_n(z)^2} \\
&\leq \sqrt{(I_1^*)^2 + \cdots + (I_n^*)^2} \\
&= \sqrt{(\sum_{k=1}^{n} |T_{1k} I_k|)^2 + \cdots + (\sum_{k=1}^{n} |T_{nk} I_k|)^2} \\
&< \sqrt{(\sum_{k=1}^{n} |T_{1k} \varepsilon z_k|)^2 + \cdots + (\sum_{k=1}^{n} |T_{nk} \varepsilon z_k|)^2} \\
&= \sqrt{\varepsilon^2 (\sum_{k=1}^{n} |T_{1k} z_k|)^2 + \cdots + \varepsilon^2 (\sum_{k=1}^{n} |T_{nk} z_k|)^2} \\
&\leq \sqrt{\varepsilon^2 \|z\|^2 n} = \varepsilon \sqrt{n} \|z\| \ = \gamma \|z\| \qquad \text{for } \|z\| < r
\end{aligned} \tag{4.13}$$

where $\gamma \equiv \varepsilon \sqrt{n}$. Using (4.13), we have

$$\begin{aligned}
\dot{V}(z) &\leq -\|z\|^2 + \|z\| \|g(z)\| \\
&\leq (\varepsilon \sqrt{n} - 1) \|z\|^2 = (\gamma - 1) \|z\|^2 \ \text{ for } \ \|z\| < r.
\end{aligned}$$

If $\gamma < 1$, then $\dot{V}(z) < 0$, which yields an asymptotic stable ARMCNN defined in (4.4). Since

$$\gamma \equiv \varepsilon \sqrt{n},$$

we have

$$0 < \gamma = \varepsilon\sqrt{n} < 1 \Rightarrow \varepsilon < \frac{1}{\sqrt{n}}. \tag{4.14}$$

Therefore, by choosing a proper ε from the above (4.14), we can find r from (4.11) such that

$$\begin{aligned} \|z\| < r \Rightarrow \dot{V}(z) &\leq (\gamma - 1)\|z\|^2 \\ \|z\| < r, \|z\| \neq 0 \Rightarrow \dot{V}(z) &< 0 \\ \|z\| = 0 \Rightarrow \dot{V}(z) &= 0 \end{aligned}$$

Finally, apply the Lyapunov local asymptotic theorem in Section 4.3.1, we have

$$V(z) = z^T P z = 0.5\|z\|^2 < 0.5r^2 \Rightarrow \|z\| < r \Rightarrow \dot{V}(z) < 0, \dot{V}(0) = 0 \tag{4.15}$$

that is, $\|z\| < r$ is the conservative DOA.

The above derivation is applicable to any activation function satisfying (4.6) and (4.7) that includes the PWL function of gain greater than unity, as Theorem 1 shows. Furthermore, only the unity sum of the absolute values of the ratio weights is required. The ratio weights can vary a lot as long as the signs of the corresponding ratio weights in the above proof are kept. Therefore, the ARMCNN can converge to the correct equilibrium states even when there are deviations in the ratio weights due to different VLSI processes.

4.5 Examples

Example 1. Ratio weight variations do not change the equilibrium points. For the 1-D 3N system in (4.16) of Figure 4.4 under $R_i = 1, C_i = 1, I_{sat} = 1, \ \ i = 1, 2, 3$, we have

$$\begin{aligned} \dot{x}_1 &= -x_1 + \sigma(x_2) \\ \dot{x}_2 &= -x_2 + 0.5\sigma(x_1) - 0.5\sigma(x_3) \\ \dot{x}_3 &= -x_3 - \sigma(x_2) \end{aligned} \tag{4.16}$$

where the activaion function $\sigma(\cdot)$ satisfies (4.6) and (4.7). Hence, $\sigma(1) = 1, \ \sigma(-1) = -1$. The thick bond in Figure 4.4 is due to a negatively correlative input image at the two

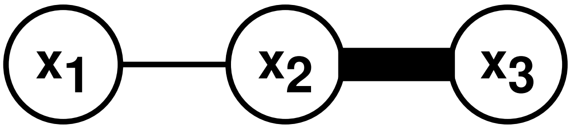


Figure 4.4: A 1-D three-neuron (3N) subsystem with two binary equilibrium points of black-black-white and white-white-black, or $(x_1(\infty), x_2(\infty), x_3(\infty)) = \pm(I_{sat}R, I_{sat}R, -I_{sat}R)$.

connected neurons and the thin bond is due to a positively correlative input image at the two connected neurons.

$(1, 1, -1)$ is an equilibrium point as can be verified by substituting $(1, 1, -1)$ into (4.16). If the process variation perturbs the system into

$$\begin{aligned}\dot{x}_1 &= -x_1 + \sigma(x_2)\\ \dot{x}_2 &= -x_2 + 0.7\sigma(x_1) - 0.3\sigma(x_3)\\ \dot{x}_3 &= -x_3 - \sigma(x_2),\end{aligned} \tag{4.17}$$

then after substitution of the original equilibrium point of $(1, 1, -1)$ into the right hand side of (4.17), we have

$$\begin{aligned}-1 + 1 &= 0\\ -1 + 0.7 - 0.3(-1) &= 0\\ -(-1) - 1 &= 0.\end{aligned}$$

Therefore $(1, 1, -1)$ is still an equilibrium point.

Example 2. Finding a DOA for a three-neuron system in Figure 4.4. The new state variable z_i with respect to the normalized equilibrium point $(1, 1, -1)$ is defined as

$$\begin{aligned}z_1 &= x_1 - 1\\ z_2 &= x_2 - 1\\ z_3 &= x_3 + 1.\end{aligned}$$

After this transformation from x to z, the new system exhibits equilibrium at the origin and is described as

$$\dot{z} = -z + g(z) \tag{4.18}$$

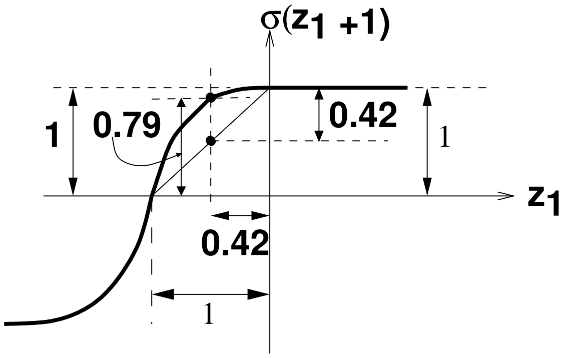


Figure 4.5: The shifted sinusoidal activation function of $\sigma(z_1+1)$. The sinusoidal activation function is defined in (4.19).

where

$$
\begin{aligned}
g(z) &= [g_1(z) \quad g_2(z) \quad g_3(z)]^T \\
g_1(z) &= \sigma(z_2+1) - 1 = I_2 \\
g_2(z) &= 0.5\sigma(z_1+1) - 0.5\sigma(z_3-1) - 1 \\
&= 0.5\,(\sigma(z_1+1)-1) - 0.5(\sigma(z_3-1)+1) \\
&= 0.5I_1 - 0.5I_3 \\
g_3(z) &= -(\sigma(z_2+1)-1) = -I_2.
\end{aligned}
$$

Assume that the activation function is the sinusoidal type defined as

$$
\begin{aligned}
\sigma(z_i) &= \sin(\frac{\pi}{2}z_i) \text{ if } |z_i| \leq 1 \\
\sigma(z_i) &= 1 \text{ if } z_i \geq 1 \\
\sigma(z_i) &= -1 \text{ if } z_i \leq -1.
\end{aligned} \qquad (4.19)
$$

Also assume that each neuron has the same activation function of (4.19). Figure 4.5 shows the shifted sinusoidal activation function of $\sigma(z_1+1)$. (4.19) satisfies the activation function requirement in (4.6) and (4.7). To show this, consider first $0 < z_1 < 1$. It follows that,

$$
\begin{aligned}
&\Rightarrow z_1 + 1 > 1\\
&\Rightarrow \sigma(z_1 + 1) = 1\\
&\Rightarrow |\sigma(z_1 + 1) - 1| = |I_1| = 0 \leq |z_1| && (4.20)\\
&\Rightarrow \lim_{z_1 \to 0^+} \frac{|\sigma(z_1 + 1) - 1|}{|z_1|} = \lim_{z_1 \to 0^+} \frac{0}{|z_1|} = 0. && (4.21)
\end{aligned}
$$

Next consider $-1 < z_1 \leq 0$. It follows that,

$$
\begin{aligned}
&\Rightarrow \sigma(z_1) = \sin(\frac{\pi}{2} z_1)\text{and}0 < \frac{z_1 + 1}{2} \leq \frac{1}{2}\\
&\Rightarrow 0 < \sigma(z_1 + 1) = \sin(\pi \frac{z_1 + 1}{2}) \leq 1.
\end{aligned}
$$

The sinc function of $\mathrm{sinc}(z) \equiv \frac{\sin \pi z}{\pi z}$ is monotonically decreasing from $z = 0$ to $z = 1$ [49]. So we have the following,

$$
\begin{aligned}
&\Rightarrow \mathrm{sinc}(\frac{1}{2}) \leq \frac{\sin(\pi \frac{z_1+1}{2})}{\pi \frac{z_1+1}{2}} = \mathrm{sinc}(\frac{z_1 + 1}{2}) < \mathrm{sinc}(0)\\
&\Rightarrow \frac{2}{\pi} \leq \frac{\sin(\pi \frac{z_1+1}{2})}{\pi \frac{z_1+1}{2}} < 1\\
&\Rightarrow z_1 + 1 \leq \sin(\pi \frac{z_1 + 1}{2}) = \sigma(z_1 + 1)\\
&\Rightarrow z_1 \leq \sigma(z_1 + 1) - 1 = I_1\\
&\Rightarrow |z_1| = -z_1 \geq 1 - \sigma(z_1 + 1) = |\sigma(z_1 + 1) - 1| \geq 0 && (4.22)
\end{aligned}
$$

Hence the limit as z_1 approaches from 0^- is

$$
\begin{aligned}
&\lim_{z_1 \to 0^-} \frac{|\sigma(z_1 + 1) - 1|}{|z_1|} = \lim_{z_1 \to 0^-} \frac{1 - \sigma(z_1 + 1)}{-z_1}\\
&= \lim_{z_1 \to 0^-} \frac{1 - \sin(\pi \frac{z_1+1}{2})}{-z_1} = \frac{-(\pi/2)\cos(\pi \frac{z_1+1}{2})}{-1}\Big|_{z_1=0} = 0. && (4.23)
\end{aligned}
$$

(4.23) is from L'Hospital's rule. From (4.20), (4.21), (4.22) and (4.23), we have shown that (4.19) satisfies the activation function requirement in (4.6). In a similar way, (4.19) can be shown to satisfy (4.7).

From Theorem 2, the finding of the conservative DOA is to find the r such that

$$
\text{if } |z_k| < r \Rightarrow |I_k| < \varepsilon |z_k| \text{ for } k = 1, 2, 3 \qquad (4.24)
$$

where

$$\varepsilon < 1/\sqrt{3} = 0.577$$

for a three-neuron system. Because of the odd symmetry property of (4.19), it suffices to use z_1 and I_1 to find the r. Figure 4.5 illustrates this process in finding the r such that if $|z_1| < r$, then $|I_1| \leq 0.5|z_1| < 0.577|z_1|$, which can be solved iteratively in the following:

1. Try $r = 0.5$, we have

$$\sin(\frac{\pi}{2}(-0.5+1)) = 0.707, \;\; z_1 = -0.5$$
$$|I_1/z_1| = (1-0.707)/(0.5) = 0.586 > 0.5.$$

2. Then select $r = 0.45$, we have

$$\sin(\frac{\pi}{2}(-0.45+1)) = 0.76, \;\; z_1 = -0.45$$
$$|I_1/z_1| = (1-0.76)/(0.45) = 0.53 > 0.5.$$

3. Finally, the following converged solution is

$$\sin(\frac{\pi}{2}(-0.42+1)) = 0.79, \;\; z_1 = -0.42$$
$$|I_1/z_1| = (1-0.79)/(0.42) = 0.5.$$

(4.19) has the property that

$$\left|\frac{I_1}{z_1}\right|_{z_1=z_{1a}} < \left|\frac{I_1}{z_1}\right|_{z_1=z_{1b}}, \quad \text{for} \;\; -1 < z_{1b} < z_{1a} < 0,$$

or

$$\frac{d}{dz_1}\left|\frac{I_1}{z_1}\right| = \frac{d}{dz_1}\left(\frac{\sin(\pi\frac{z_1+1}{2}) - 1}{z_1}\right) < 0, \quad \text{for} \;\; -1 < z_1 < 0.$$

So it follows that $\|z\| < r = 0.42$ is a conservative DOA for this three-neuron system.

The trajectories at three different initial points are shown in Figure 4.6. These three initial points $x(0)$ are $(0,1,0), (0.5,0,-1), (1.2,0.9,-1.5)$. The projection onto the plane $x_3 = -1$ is also shown. Clearly, these three points converge to the equilibrium point of $(1,1,-1)$, even these three initial points have the initial norm $\|z(0)\| > 0.42$. This implies that $\|z\| < r = 0.42$ is a conservative DOA for this three-neuron system.

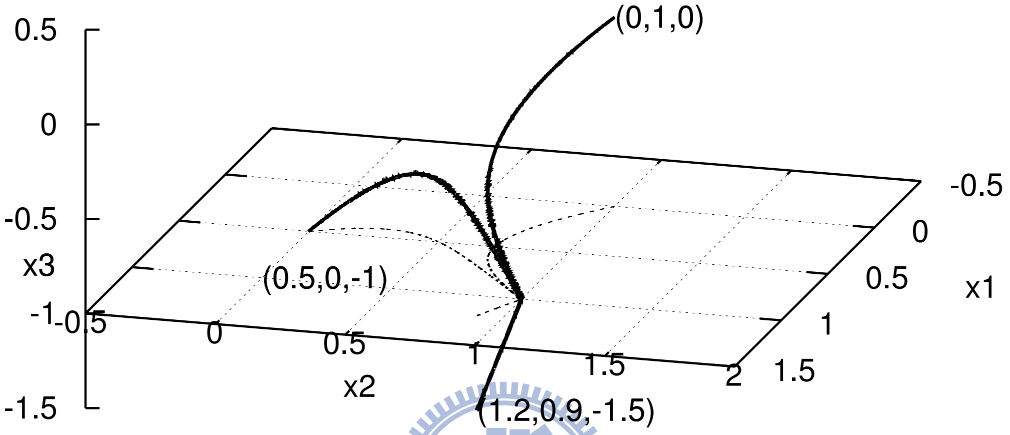


Figure 4.6: The trajectories in thick lines at three different initial states converge to the $(1, 1, -1)$. The projection onto the plane $x_3 = -1$ is shown in the thin dotted lines.

Figure 4.7 shows that the simulated norm of the trajectory $\|z(t)\|$ satisfies

$$\|z(t)\| \leq e^{-0.2t}\|z(0)\|,$$

where

$$\|z(t)\| = \sqrt{z_1^2 + z_2^2 + z_3^2} = \sqrt{(x_1 - 1)^2 + (x_1 - 1)^2 + (x_1 + 1)^2}.$$

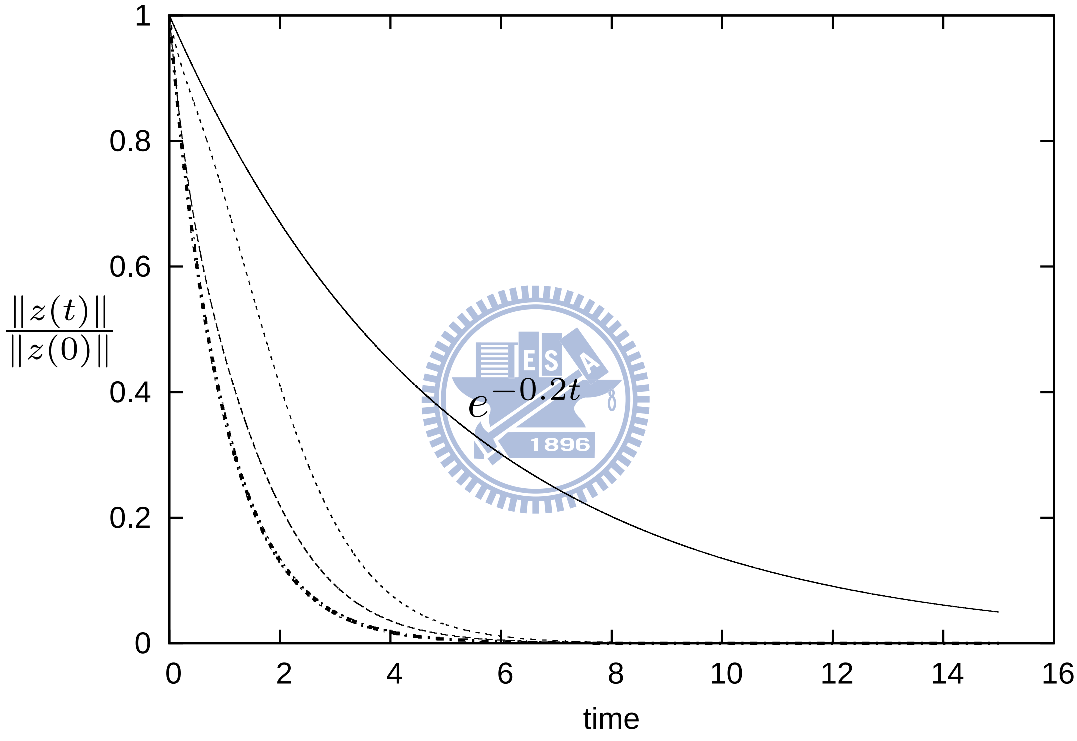


Figure 4.7: The normalized trajectories $\frac{\|z(t)\|}{\|z(0)\|}$ at three different initial points converge faster than $e^{-0.2t}$ towards the $(0, 0, 0)$. $e^{-0.2t}$ is plotted in the solid line.

Chapter 5

Numerical Stability Analysis

5.1 Introduction

In recent years, there has been an increasing concern to the choice of integration methods and time steps for the discrete time CNN application and the simulation [50, 51, 52]. To efficiently solve CNNs consisted of coupled nonlinear equations, several one-step numerical integration methods are available. Among them, the Runge-Kutta method is noted for its higher accuracy compared to the Forward Euler (FE) method because several weighted derivatives are used in the RK method [53]. On the other hand, the Forward Euler method, using single derivative, has a higher computation speed than the Runge-Kutta method. Hence, in a large CNN array emulation such as the emulation of a multi-layer retina model in [54, 55], the FE method is usually adopted. However, choosing an optimum time step for the FE method is usually difficult because trade-off exists among the time step, accuracy, and stability [17, 56]. A large time step enhances the computation speed. However, the FE method becomes unstable as a threshold time step is exceeded. Therefore, the time step is chosen as large as possible under the constraints of the accuracy and the numerical stability. Press, Teukolsky, Vetterling, and Flannery even considered obtaining a stable and efficient finite difference method is “an art as much as a science” [57].

Hanggi (2001) provided in-depth analysis for the important locally regular (LR) CNN, showing the optimum FE time step exists for the LR CNN [17]. In addition, the correct equilibrium state is not changed using the optimum FE time step. This raises an interesting question as to whether other binary CNNs such as ARMCNN have this property or not. If not, what is the maximum FE time step to avoid numerical instability?

This chapter aims at deriving a maximum time step for the ARMCNN simulated via the forward Euler method without inducing the numerical instability problem. To do this, this chapter associated ARMCNNs with the heat diffusion equation and used the eigenspace decomposition for the ARMCNN initial states. From the eigenspace decomposition, an analytic neuron gain is derived to lower the amount of spurious memory points. Further, because only binary equilibrium point is concerned instead of the complete transient trajectory, this chapter focuses on deriving a sufficient condition such that the FE simulated ARMCNNs (FE-ARMCNN) have the same binary equilibrium point as the continuous time ARMCNN (CT-ARMCNN).

5.2 Diffusion Model

The ARMCNN diffusion model is easily understood through a one-dimensional (1-D) network. Consider a one-dimensional ARMCNN having $P = 5$ neuron states $x_i, i = 0, 1, \ldots, P-1$. Figure 5.1 shows the circuit implementation. Neuron states x_1, x_2, x_3 are interior neurons, each with two neighboring neurons. However, x_0 is only connected to x_1, and x_4 is only connected to x_3. To approximate a 1-D heat diffusion on a finite interval using ARMCNN, boundary conditions are required for x_0 and x_4 [58]. Hence, we rewrite the 1-D ARMCNN dynamic equation into a diffusion form to obtain the the boundary conditions for x_0 and x_4.

Assume neuron states x_0, x_1, x_2, x_3, x_4 are in the linear region. The numering for the state x_i goes from 0 to $P-1$, $P = 5$. From the definition in (3.9) and (3.7), the set of

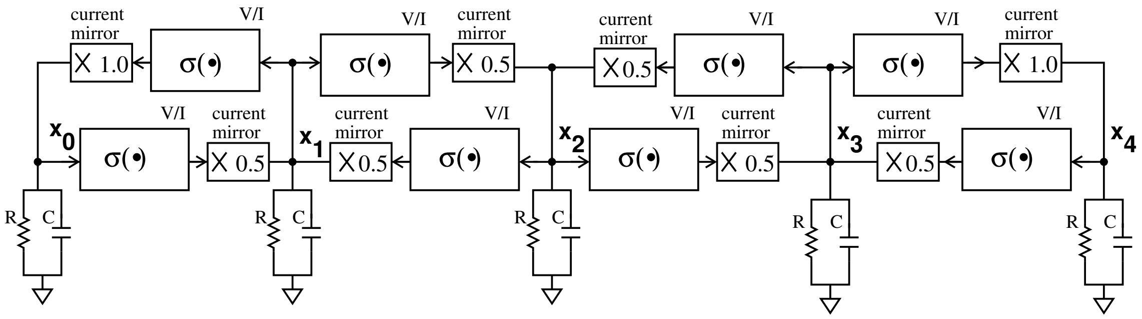


Figure 5.1: A circuit implementation for a 1-D 5N ARMCNN. For each synapse connection, two voltage to current converters (V/I) and two current mirrors are required.

dynamic equations is rewritten as

$$
\begin{aligned}
\tau\dot{x}_0 &= -x_0 + gx_1 = -x_0 + 0.5gx_1 + 0.5gx_{-1},\ x_{-1} = x_1\\
&= 0.5g(x_{-1} + x_1 - 2x_0) + x_0(g-1)\\
\tau\dot{x}_1 &= -x_1 + 0.5gx_0 + 0.5gx_2\\
&= 0.5g(x_0 + x_2 - 2x_1) + x_1(g-1)\\
\tau\dot{x}_2 &= -x_2 + 0.5gx_1 + 0.5gx_3\\
&= 0.5g(x_1 + x_3 - 2x_2) + x_2(g-1)\\
\tau\dot{x}_3 &= -x_3 + 0.5gx_2 + 0.5gx_4\\
&= 0.5g(x_2 + x_4 - 2x_3) + x_3(g-1)\\
\tau\dot{x}_4 &= -x_4 + gx_3 = -x_4 + 0.5gx_3 + 0.5gx_5,\ x_5 = x_3(x_P = x_{P-2})\\
&= 0.5g(x_3 + x_5 - 2x_4) + x_4(g-1)
\end{aligned}
\tag{5.1}
$$

where $\tau = RC$. x_{-1} has the symmetry at the $j = 0$ in that $x_{-1} = x_1$, while x_5 has the symmetry at the $j = 4$ in that $x_5 = x_3$. The voltage at x_{-1} is required to be equal to the voltage at x_1, and the voltage at x_5 is required to be equal to the voltage at x_3. The symmetry property is consistent with the boundary condition assignment of the Discrete Cosine Transform type 1 (DCT-1) [59]. Figure 5.2 illustrates the diffusion circuit representing (5.1) [58]. The connecting resistor between x_i and x_{i+1} $(i = -1, \ldots, 4)$ has a conduction value of $0.5g/R$. Each state $(x_0, \ldots, x_4)$ also connects to ground through a conduction value of $(1-g)/R$. Voltage controlled voltage sources (VCVS) are at the two ends with $x_{-1} = x_1, x_5 = x_3$.

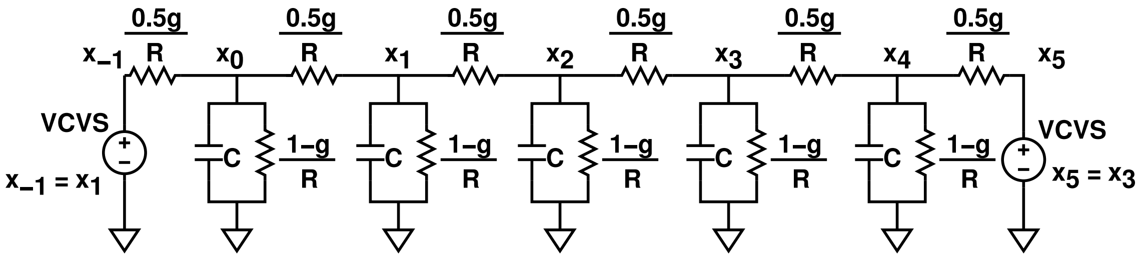


Figure 5.2: An equivalent diffusion circuit for (5.1).

To create finite difference equations for (5.1), we use a forward Euler (FE) difference with step Δt:

$$\begin{aligned} \frac{x_{j,n} - x_{j,n-1}}{\Delta t} &= (\frac{1}{h^2})(x_{j-1,n-1} + x_{j+1,n-1} - 2x_{j,n-1}) + \frac{g-1}{\tau}x_{j,n-1}, \ j = 0, 1, 2, 3, 4 \\ x_{j,n} &= \frac{0.5g\Delta t}{\tau}(x_{j-1,n-1} + x_{j+1,n-1}) + x_{j,n-1}(1 - \frac{\Delta t}{\tau}) \end{aligned} \tag{5.2}$$

where $x_{j,n}$ is a discrete approximation of $x_j(n\Delta t)$ and

$$\begin{aligned} x_{-1} = x_1, x_5 = x_3 \\ \frac{1}{h^2} = \frac{0.5g}{\tau} \end{aligned} \tag{5.3}$$

The first index $j = 0$ involves x_{-1}, and the last index $j = 4$ involves x_5. The initial conditon is

$$x_{j,0} = x_j(0).$$

The equation set of (5.2) corresponds to a numerical approximation of a modified heating rod equation

$$\frac{\partial x}{\partial t} = \frac{\partial^2 x}{\partial z^2} + \frac{g-1}{\tau}x \tag{5.4}$$

where $x(z, t)$ is the state at location z and time t and $x_{j,n}$ is the discrete approximation of $x(jh, n\Delta t)$ on the space-time grid with $\Delta z = x_{j,n} - x_{j-1,n} = h$. From (5.3), the two ends are the Neumann type boundary condition. Physically, this means no heat flows in or out at the boundary.

(5.2) also implies that if at $t = (n-1)\Delta t$ all states are positive (negative) and the time step $\frac{\Delta t}{\tau}$ is less than unity, then all states remain to be positive (negative) at the next time step of $t = n\Delta t$.

5.2.1 Exact Solution of the Diffusion Model

To obtain the exact solution of this 1-D 5N ARMCNN in Figure 5.2, express (5.1) in matrix notation as

$$\tau \dot{x} = Ax, \ \dot{x} = \frac{1}{\tau} Ax$$

where the state vector x, the A matrix and the $\dot{x}$ are

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, \quad A = \begin{bmatrix} -1 & g & 0 & 0 & 0 \\ 0.5g & -1 & 0.5g & 0 & 0 \\ 0 & 0.5g & -1 & 0.5g & 0 \\ 0 & 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0 & g & -1 \end{bmatrix}, \quad \dot{x} = \begin{bmatrix} \dot{x}_0 \\ \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} \tag{5.5}$$

and the initial condition is given as

$$x(0) = \begin{bmatrix} x_0(0) \\ x_1(0) \\ x_2(0) \\ x_3(0) \\ x_4(0) \end{bmatrix}.$$

$\dot{x}_0$ and $\dot{x}_4$ are determined by the boundary rows (rows 0 and $P-1$) of the A matrix. The interior rows (rows $1, \ldots, P-2$) of the A matrix in (5.5) have the nonzero entries of $0.5g, -1, 0.5g$. The jth component of Ax is $0.5gx_{j-1} - x_j + 0.5gx_{j+1}$ for $j = 1, \ldots, P-2$. From Appendix B, A is diagonalizable, and $r_0, \ldots, r_4$ is a linearly independent set of eigenvectors of A satisfying $Ar_k = \lambda_k r_k, k = 0, \ldots, 4$ with $\lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_4$. Therefore, we can write the diagonalization of A as

$$AR = R\Lambda, \ A = R\Lambda R^{-1}, \ \Lambda = R^{-1}AR, \ \Lambda = \mathbf{diag}(\lambda_0, \ldots, \lambda_4), \ R^{-1} = \begin{bmatrix} w_0^T \\ \vdots \\ w_4^T \end{bmatrix},$$

where the eigenvectors are put into the columns of R as

$$R = [r_0 \cdots r_4].$$

To simplify the cross coupled complexity in the differential equations of (5.1), let $\tilde{x} = R^{-1}x$ [60],

$$\tilde{x} = \begin{bmatrix} \tilde{x}_0 \\ \vdots \\ \tilde{x}_4 \end{bmatrix} = R^{-1}x = \begin{bmatrix} w_0^T \\ \vdots \\ w_4^T \end{bmatrix} x.$$

Hence,

$$\begin{aligned} \tilde{x}_k &= w_k^T x, \ k = 0, \ldots, 4 \\ \dot{\tilde{x}} &= R^{-1}\dot{x} = R^{-1}(\frac{1}{\tau}Ax) = R^{-1}\frac{1}{\tau}ARR^{-1}x = \frac{1}{\tau}\Lambda(R^{-1}x) = \frac{1}{\tau}\Lambda\tilde{x}. \end{aligned} \tag{5.6}$$

The kth component of $\dot{\tilde{x}}$ in (5.6) is

$$\dot{\tilde{x}}_k = \frac{1}{\tau}\lambda_k\tilde{x}_k, \quad \tilde{x}_k(0) = w_k^T x(0) \tag{5.7}$$

and the exact solution for (5.7) is

$$\tilde{x}_k = e^{\frac{\lambda_k}{\tau}t} w_k^T x(0). \tag{5.8}$$

Transform back to the x variable using $x = R\tilde{x}$ to obtain the exact trajectory solution as

$$\begin{aligned} x = [r_0 \cdots r_4]\tilde{x} &= [r_0 \cdots r_4] \begin{bmatrix} \tilde{x}_0 \\ \vdots \\ \tilde{x}_4 \end{bmatrix} \\ &= \tilde{x}_0 r_0 + \cdots + \tilde{x}_4 r_4 \\ &= \sum_{k=0}^{4} \left(e^{\lambda_k \frac{t}{\tau}} w_k^T x(0) \right) r_k. \end{aligned} \tag{5.9}$$

We summarize three steps to obtain the exact trajectory:

- Expand $x(0) = Ra = a_0 r_0 + a_1 r_1 + \cdots + a_{P-1} r_{P-1}$, where

$$a = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{P-1} \end{bmatrix} = R^{-1}x(0) = \begin{bmatrix} w_0^T \\ \vdots \\ w_4^T \end{bmatrix} x(0), \quad a_k = w_k^T x(0), \quad k = 0, \cdots, P-1.$$

- Multiply each a_k by $e^{\lambda_k \frac{t}{\tau}}$.

- Recombine $x = \sum_{k=0}^{4} \left(e^{\lambda_k \frac{t}{\tau}} w_k^T x(0) \right) r_k$.

5.2.2 Computational Method

The dynamic behavior can be analyzed from the exact solution in (5.9). To compute the trajectory, we begin with equal time steps Δt and consider the forward Euler's (FE) method due to its high computational speed and simplicity. Apply the FE method to (5.7), we obtain

$$\begin{aligned} \tilde{x}_{k,1} &= \tilde{x}_{k,0} + \frac{\lambda_k}{\tau}\Delta t \tilde{x}_{k,0} = (1 + \frac{\lambda_k}{\tau}\Delta t)\tilde{x}_{k,0} \\ \tilde{x}_{k,n} &= (1 + \frac{\lambda_k}{\tau}\Delta t)^n \tilde{x}_{k,0} = (1 + \frac{\lambda_k}{\tau}\Delta t)^n w_k^T x(0) \end{aligned} \tag{5.10}$$

where $\tilde{x}_{k,n}$ is the discrete approximation of $\tilde{x}_k(n\Delta t)$.

Using (5.10) and $x = R\tilde{x}$, the FE discrete trajectory is

$$\sum_{k=0}^{4} (1 + \frac{\lambda_k}{\tau}\Delta t)^{t/\Delta t} \left(w_k^T x(0) \right) r_k, \quad t = n\Delta t \tag{5.11}$$

This discrete trajectory of (5.11) is consistent with the exact solution in (5.9) in that:

$$(1 + \frac{\lambda_k}{\tau}\Delta t)^{t/\Delta t} \text{ approaches } e^{\frac{\lambda_k}{\tau} t} \quad \text{as} \quad \Delta t \to 0. \tag{5.12}$$

Numerical instability for (5.11) occurs if

$$1 + \frac{\lambda_k}{\tau}\Delta t < -1. \tag{5.13}$$

The above convergence requirement in (5.12) and (5.13) have been proved in [61, 62].

The jth state of (5.11) at time $n\Delta t$, $x_{j,n}$, is a discrete approximation of $x_j(n\Delta t)$. By defining

$$G(\frac{\Delta t}{\tau}, k) = 1 + \frac{\lambda_k}{\tau}\Delta t, \tag{5.14}$$

$x_{j,n}$ can be written from (5.11) as

$$\begin{aligned} x_{j,n} &= \sum_{k=0}^{4}(1+\frac{\Delta t}{\tau}\lambda_k)^n \left(w_k^T x(0)\right) r_{k,j}, \quad n=\frac{t}{\Delta t}, \quad j=0,\ldots,4 \\ &= \sum_{k=0}^{4} G(\frac{\Delta t}{\tau},k)^n a_k r_{k,j} \\ &= G(\frac{\Delta t}{\tau},0)^n \left(w_0^T x(0)\right) + \sum_{k=1}^{4} G(\frac{\Delta t}{\tau},k)^n \left(w_k^T x(0)\right) r_{k,j} \end{aligned} \tag{5.15}$$

where the jth component of the kth eigenvector r_k is $r_{k,j}$ and $r_{0,j}=1$ for $j=0,1,\ldots,4$. Appendix B shows the derivation of the eigenvalues λ_k and the eigenvectors r_k. The DC amplitude for all $x_j,\ j=0,\ldots,P-1$ at $t=n\Delta t$ is defined as

$$\left| G(\frac{\Delta t}{\tau},0)^n \left(w_0^T x(0)\right) \right|, \tag{5.16}$$

and the AC amplitude at $x_{j,n}$ is defined as

$$\left| \sum_{k=1}^{4} G(\frac{\Delta t}{\tau},k)^n \left(w_k^T x(0)\right) r_{k,j} \right|.$$

(5.11) has a physical meaning: the input initial state vector is represented as a combination of various eigenvector r_k. Each eigenvector r_k has a corresponding coefficient $a_k G(\frac{\Delta t}{\tau},k)^n$. The $a_k G(\frac{\Delta t}{\tau},k)^n$ grows by a growth factor $G(\frac{\Delta t}{\tau},k)$ at each time step. In the following sections, we derive the maximum time step to avoid the numerical instability through this decomposition. Furthermore, by enhancing the low frequency part of the input initial state vecor, we obtain the neuron gain range to decrease the amount of spurious equilibrium points.

5.2.3 Maximum Time Step

Consider the eigenspace decomposition of initial states $x_{j,0}$ in (5.15)

$$n=0,\ x_{j,0}=x_j(0)=\sum_{k=0}^{P-1} a_k r_{k,j} = \left(w_0^T x(0)\right) + \sum_{k=1}^{4}\left(w_k^T x(0)\right) r_{k,j}. \tag{5.17}$$

r_k are from the eigenvectors of the A matrix in (5.5). At each time step, the coefficient a_k grows by a growth facotr $G(\frac{\Delta t}{\tau},k)=1+\frac{\Delta t}{\tau}\lambda_k$ from (5.15). Hence, it is essential to obtain the eigenvalues and eigenvectors of the A matrix in (5.5). Figure 5.3 illustrates the decomposition where r_0 is the zero frequency all-ones DC eigenvector and from (5.16), the DC amplitude of (5.17) is

$$|a_0 = w_0^T x(0)|. \tag{5.18}$$

Figure 5.3 also illustrates the AC amplitude at x_1.

The eigenvalues of the A matrix in (5.5) are derived in Appendix B as

$$\begin{aligned} \lambda_k &= g\cos(\frac{k\pi}{P-1}) - 1, \quad k = 0, \dots, P-1 \\ \lambda_0 &= g - 1 > \lambda_1 > \lambda_2 > \lambda_{P-1} = -g - 1 \end{aligned} \tag{5.19}$$

and the jth component of the kth eigenvector r_k is

$$r_{k,j} = \cos(j\frac{k\pi}{P-1}) = \cos(j\theta_k), \quad j = 0, \dots, P-1, \;\; k = 0, \dots, P-1. \tag{5.20}$$

Hence (5.17) can be expressed as

$$\begin{aligned} x(0) &= \begin{bmatrix} x_0(0) \\ x_1(0) \\ x_2(0) \\ \vdots \\ x_{P-1}(0) \end{bmatrix} = Ra = \begin{bmatrix} r_0 & r_1 & r_2 & \cdots & r_{P-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{P-1} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \cos\theta & \cos 2\theta & \cdots & \cos((P-1)\theta) \\ 1 & \cos 2\theta & \cos 4\theta & \cdots & \cos(2(P-1)\theta) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \cos(P-1)\theta & \cos 2(P-1)\theta & \cdots & \cos((P-1)^2\theta) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{P-1} \end{bmatrix} \end{aligned} \tag{5.21}$$

where

$$\theta = \frac{\pi}{P-1}, \quad a = R^{-1}x(0) = \begin{bmatrix} w_0^T \\ w_1^T \\ w_2^T \\ \vdots \\ w_{P-1}^T \end{bmatrix} x(0).$$

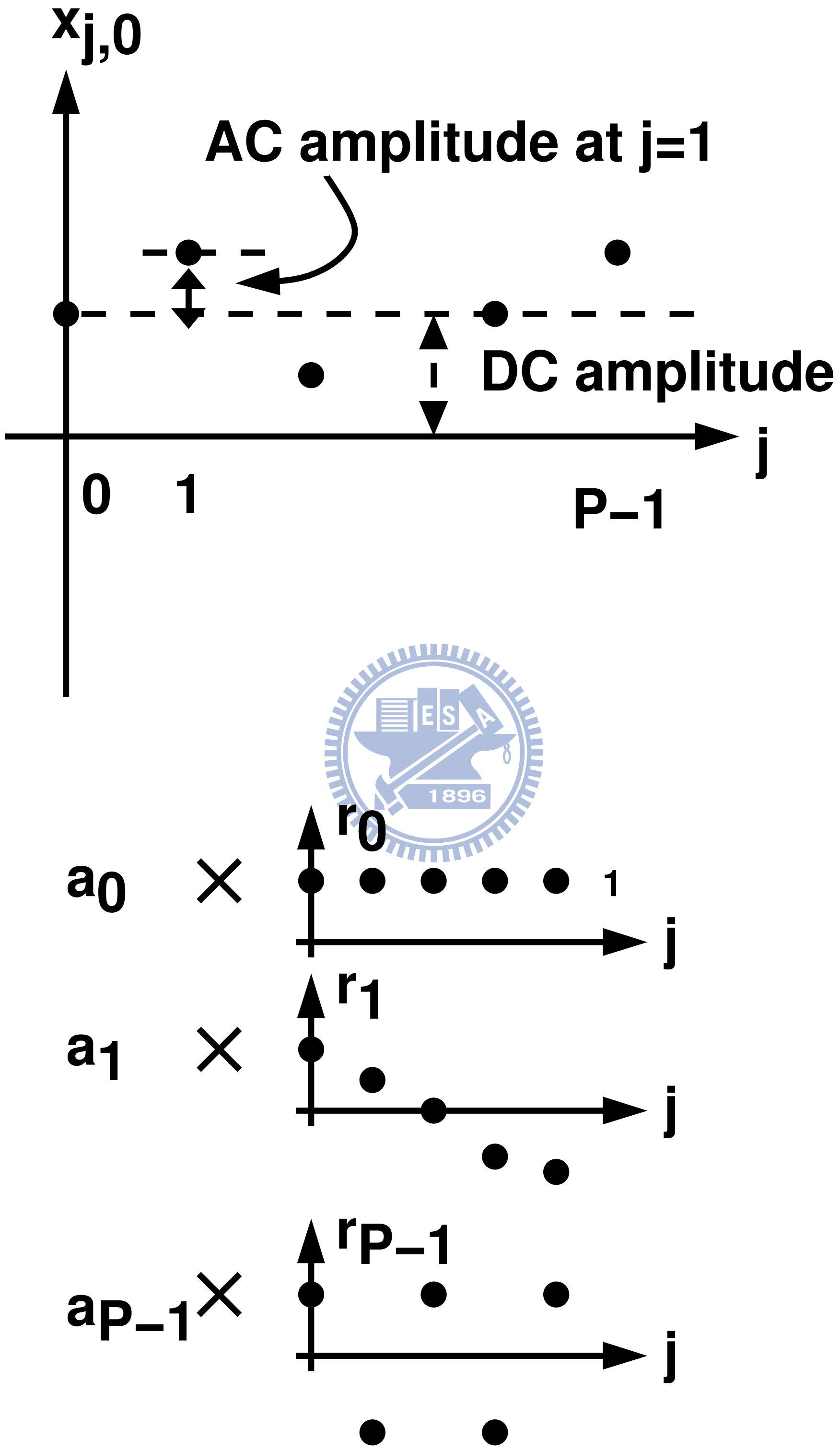


Figure 5.3: Decompostion of the initial state vector. Each r_k is multiplied by a corresponding coefficient a_k.

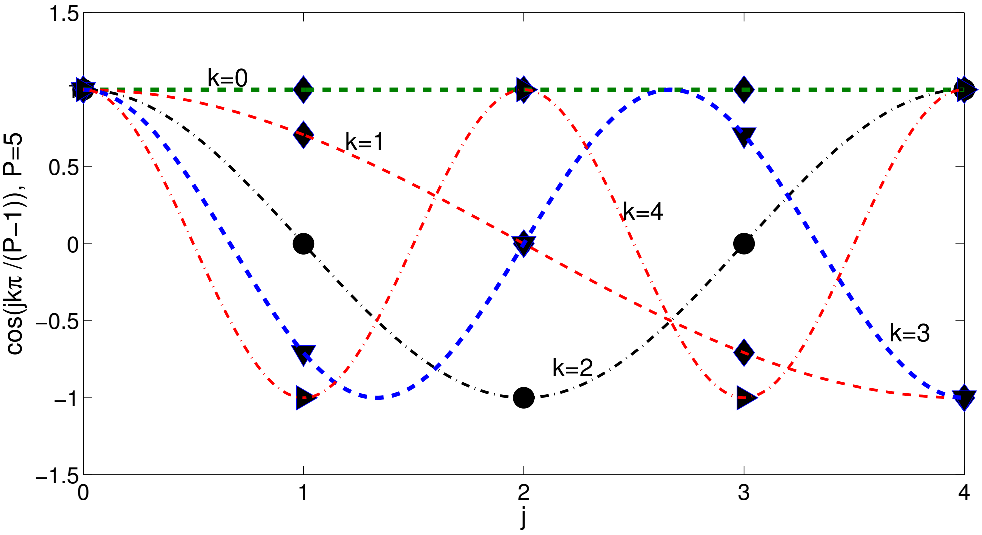


Figure 5.4: Eigenvectors r_k fall on the continuous cosine functions. As k increases, the corresponding eigenvector sequence ($j = 0, \ldots, P-1$) oscillates more rapidly.

The kth column of R, r_k, is the kth eigenvector of A matrix in (5.5). (5.21) is the matrix representation of the type 1 discrete cosine transform (DCT-1). Efficient computation algorithms and the analytic inverse matrix for R exist for DCT-1 [63]. Notice that r_0 gives the zero frequency all-ones DC eigenvector. The r_k ($k = 0, \ldots, P-1$) are illustrated in Figure 5.4. Because $|r_{k,j}| \le 1$, the maximum AC amplitude of (5.17) is bounded above as:

$$\text{Max AC amplitude} \le \sum_{k=1}^{P-1} |a_k| = \sum_{k=1}^{P-1} \left| w_k^T x(0) \right| . \tag{5.22}$$

Because each frequency component of the initial state vector is amplified by the growth factor as in (5.15), it is essential to analyze the growth factor. From (5.14) and (5.19), the growth factor is

$$G(\frac{\Delta t}{\tau}, k) = 1 + \frac{\Delta t}{\tau}\lambda_k = 1 + \frac{\Delta t}{\tau}\left(g\cos(\frac{k\pi}{P-1}) - 1 \right) \tag{5.23}$$

with

$$G(\frac{\Delta t}{\tau}, 0) > G(\frac{\Delta t}{\tau}, 1) > \cdots > G(\frac{\Delta t}{\tau}, P-1). \tag{5.24}$$

From (5.12) and (5.13), numerical instability occurs if $G(\frac{\Delta t}{\tau}, k) < -1$. So we require

$G(\frac{\Delta t}{\tau}, P-1) > -1$, or

$$G(\frac{\Delta t}{\tau}, P-1) = 1-(g+1)\frac{\Delta t}{\tau} > -1 \tag{5.25}$$

$$\Rightarrow \frac{\Delta t}{\tau} < \frac{2}{g+1}. \tag{5.26}$$

This sets the upper bound of the FE time step to avoid numerical instability.

5.2.4 Single Mode Operation

The binary ouput is only dominated by the zeroth term, $k = 0$, in (5.11) if we filter out high-frequency components ($k = 1, 2, \ldots, P-1$) and enhance the zero frequency component (k=0). To achieve this goal, we notice from (5.23) that

$$G(\frac{\Delta t}{\tau}, 0) = 1 + \frac{\Delta t}{\tau}(g-1) > 1$$

and require that

$$-1 < G(\frac{\Delta t}{\tau}, k) = 1 + \frac{\Delta t}{\tau}\left(g\cos(\frac{k\pi}{P-1}) - 1\right) < 1, \text{for} \quad k = 1, 2, \ldots, P-1.$$

We already have from (5.26) that if

$$\frac{\Delta t}{\tau} < \frac{2}{g+1},$$

then

$$-1 < 1 + \frac{\Delta t}{\tau}\left(g\cos(\frac{k\pi}{P-1}) - 1\right), k = P-1.$$

From (5.24), we only need to choose g such that

$$1 + \frac{\Delta t}{\tau}\left(g\cos(\frac{k\pi}{P-1}) - 1\right) < 1, \ k = 1$$

which implies

$$g\cos(\frac{\pi}{P-1}) - 1 < 0, \quad \text{or} \quad 1 < g < \frac{1}{\cos(\frac{\pi}{P-1})}, \ P \geq 4.$$

Note if $P = 2, 3$, then for all $g > 1$, $g\cos(\frac{\pi}{P-1}) - 1 < 0$.

Consequently, we have shown that if

$$
\begin{aligned}
&1 < g < g_{th} = \frac{1}{\cos(\frac{\pi}{P-1})}, \quad \text{for} \quad P \geq 4 \\
&g > 1 \quad \text{for} \quad P = 2, 3
\end{aligned}
\tag{5.27}
$$

and

$$\frac{\Delta t}{\tau} < \frac{2}{g+1} \tag{5.28}$$

then no numerical instability occurs and only the DC term $G(\frac{\Delta t}{\tau}, 0)^n w_0^T x(0) r_0$ in (5.15) grows because $G(\frac{\Delta t}{\tau}, 0) > 1$ and $|G(\frac{\Delta t}{\tau}, k)| < 1, \ k = 1, \ldots, P-1$. Appendix C derives w_0^T as:

$$w_0^T = [\frac{1}{8} \ \frac{1}{4} \ \frac{1}{4} \ \frac{1}{4} \ \frac{1}{8}].$$

Therefore, the coefficient $w_0^T x(0) = a_0$ is

$$w_0^T x(0) = a_0 = \frac{1}{8}\left(x_0(0) + 2x_1(0) + 2x_2(0) + 2x_3(0) + x_4(0)\right).$$

Hence, one possible condition to generate spurious equlibrium points is

$$w_0^T x(0) = a_0 = x_0(0) + 2x_1(0) + 2x_2(0) + 2x_3(0) + x_4(0) = 0. \tag{5.29}$$

However, in a practical circuit implementation, any thermal noise perturbs (5.29). Equation (5.29) can only hold in a noiseless case. Therefore, the growth factor $G(\frac{\Delta t}{\tau}, 0)$ amplifies the low frequency part of the initial states if both (5.27) and (5.28) hold.

5.2.5 One End Neuron Enters the Saturation Region

If one end neuron enters the sarutation region, say x_4 in (5.1), then we consider four ($N = 4$) neurons, x_0, x_1, x_2, x_3. The numbering for the state x_i goes from 0 to $N-1, \ N = 4$.

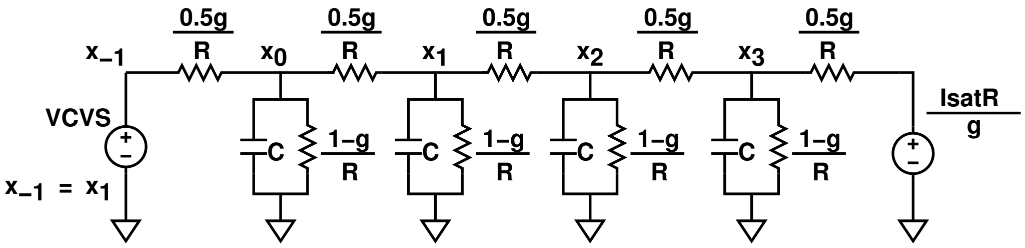


Figure 5.5: An equivalent diffusion circuit for (5.30)

The set of dynamic equations is expressed as

$$\begin{aligned}
\tau\dot{x}_0 &= -x_0 + gx_1 = -x_0 + 0.5gx_1 + 0.5gx_{-1},\ x_{-1} = x_1 \\
&= 0.5g(x_{-1} + x_1 - 2x_0) + x_0(g-1) \\
\tau\dot{x}_1 &= -x_1 + 0.5gx_0 + 0.5gx_2 \\
&= 0.5g(x_0 + x_2 - 2x_1) + x_1(g-1) \\
\tau\dot{x}_2 &= -x_2 + 0.5gx_1 + 0.5gx_3 \\
&= 0.5g(x_1 + x_3 - 2x_2) + x_2(g-1) \\
\tau\dot{x}_3 &= -x_3 + 0.5gx_2 + 0.5I_{sat}R \\
&= 0.5g(x_2 + \frac{I_{sat}R}{g} - 2x_3) + x_3(g-1). \qquad (5.30)
\end{aligned}$$

where $\tau = RC$. Figure 5.5 illustrates the diffusion circuit representing (5.30). The connecting resistor between x_i and x_{i+1} $(i = -1, \ldots, 2)$ has a conduction value of $0.5g/R$. Each state $(x_0, \ldots, x_3)$ also connects to ground through a conduction value of $(1-g)/R$. The left end boundary is a VCVS with $x_{-1} = x_1$, while the right end boundary is a fixed voltage source with a value of $\frac{I_{sat}R}{g}$.

Rewrite (5.30) in matrix notation as $\tau\dot{x} = Ax + u,\ \dot{x} = \frac{1}{\tau}Ax + \frac{1}{\tau}u$ where

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad u = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.5I_{sat}R \end{bmatrix}, \quad A = \begin{bmatrix} -1 & g & 0 & 0 \\ 0.5g & -1 & 0.5g & 0 \\ 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0.5g & -1 \end{bmatrix}. \qquad (5.31)$$

The interior rows of the A matrix in (5.31) are the same as the interior rows of the A matrix in (5.5). Only the boundary rows are different. The jth component of Ax is

$0.5gx_{j-1} - x_j + 0.5gx_{j+1}$ for $j = 1, \ldots, N-2$. The eigenvalues and eigenvectors of A in (5.31) are derived in Appendix B as

$$\begin{aligned}
\lambda_k &= g\cos\left((k+\frac{1}{2})\frac{\pi}{N}\right) - 1, \quad k = 0, \ldots, N-1 \\
\lambda_0 &= g\cos\frac{\pi}{2N} - 1 > \lambda_1 > \lambda_2 > \lambda_{N-1} = g\cos(\frac{2N-1}{2N}\pi) - 1 \qquad (5.32)
\end{aligned}$$

and the jth component of the kth eigenvector r_k is

$$r_{k,j} = \cos\left(j(k+\frac{1}{2})\frac{\pi}{N}\right) = \cos(j\theta_k), \quad j = 0, \ldots, N-1, \ \ k = 0, \ldots, N-1.$$

Therefore, we assume that $Ar_k = \lambda_k r_k, k = 0, \ldots, N-1$, and have

$$AR = R\Lambda, \ A = R\Lambda R^{-1}, \ \Lambda = R^{-1}AR,$$

where for $N = 4$,

$$R = [r_0 \cdots r_3] = \begin{bmatrix} \cos 0 & \cos 0 & \cos 0 & \cos 0 \\ \cos\theta_0 & \cos\theta_1 & \cos\theta_2 & \cos\theta_3 \\ \cos 2\theta_0 & \cos 2\theta_1 & \cos 2\theta_2 & \cos 2\theta_3 \\ \cos 3\theta_0 & \cos 3\theta_1 & \cos 3\theta_2 & \cos 3\theta_3 \end{bmatrix} \qquad (5.33)$$

$$\Lambda = \mathbf{diag}(\lambda_0, \ldots, \lambda_3), \ R^{-1} = \begin{bmatrix} w_0^T \\ \vdots \\ w_3^T \end{bmatrix}, \ \theta_0 = \frac{\pi}{2N}, \ \theta_1 = \frac{3\pi}{2N}, \theta_2 = \frac{5\pi}{2N}, \ \theta_3 = \frac{7\pi}{2N}.$$

The inverse matrix of (5.33) is derived in Appendix C. Let

$$\tilde{x} = R^{-1}x = \begin{bmatrix} w_0^T \\ \vdots \\ w_3^T \end{bmatrix} x, \quad \tilde{x} = \begin{bmatrix} \tilde{x}_0 \\ \vdots \\ \tilde{x}_3 \end{bmatrix},$$

we have

$$\begin{aligned}
\tilde{x}_k &= w_k^T x, \ k = 0, \ldots, 3 \\
\dot{\tilde{x}} &= R^{-1}\dot{x} = R^{-1}(\frac{1}{\tau}Ax + \frac{1}{\tau}u) = R^{-1}\frac{1}{\tau}ARR^{-1}x + \frac{1}{\tau}R^{-1}u = \frac{1}{\tau}\Lambda\tilde{x} + \frac{1}{\tau}R^{-1}u.
\end{aligned}$$

The kth component of $\dot{\tilde{x}}$ is

$$\dot{\tilde{x}}_k = \frac{1}{\tau}\lambda_k \tilde{x}_k + \frac{1}{\tau} w_k^T u = \alpha \tilde{x}_k + \beta, \quad \tilde{x}_k(0) = w_k^T x(0), \quad \alpha = \frac{1}{\tau}\lambda_k, \quad \beta = \frac{1}{\tau} w_k^T u \tag{5.34}$$

and the exact solution for (5.34) is

$$\tilde{x}_k = e^{\alpha t}\left(w_k^T x(0) + \frac{\beta}{\alpha}\right) - \frac{\beta}{\alpha} = e^{\frac{\lambda_k}{\tau}t}\left(w_k^T x(0) + \frac{w_k^T u}{\lambda_k}\right) - \frac{w_k^T u}{\lambda_k}. \tag{5.35}$$

Transform back to the x variable using $x = R\tilde{x}$ to obtain the exact trajectory solution as

$$\begin{aligned} x &= [r_0 \cdots r_3]\tilde{x} \\ &= \tilde{x}_0 r_0 + \cdots + \tilde{x}_3 r_3 \\ &= \sum_{k=0}^{3}\left[e^{\frac{\lambda_k}{\tau}t}\left(w_k^T x(0) + \frac{w_k^T u}{\lambda_k}\right) - \frac{w_k^T u}{\lambda_k}\right] r_k. \end{aligned} \tag{5.36}$$

The normalization condition of $I_{sat} = 1, R = 1, C = 1$ yields

$$u = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.5 \end{bmatrix}, \quad \tau = RC = 1, \quad w_k^T u = 0.5 w_{k,3}. \tag{5.37}$$

So

$$x(t) = \sum_{k=0}^{3}\left[e^{\lambda_k t}\left(w_k^T x(0) + \frac{0.5 w_{k,3}}{\lambda_k}\right) - \frac{0.5 w_{k,3}}{\lambda_k}\right] r_k. \tag{5.38}$$

The numerical FE method for (5.34) under the above normalization condition yields

$$\tilde{x}_{k,n} = \tilde{x}_{k,0}(1 + \lambda_k \Delta t)^n + \frac{0.5 w_{k,3}}{\lambda_k}(1 + \lambda_k \Delta t)^n - \frac{0.5 w_{k,3}}{\lambda_k}$$

where $\tilde{x}_{k,n}$ is the discrete approximation of $\tilde{x}_k(n\Delta t)$. Using $x = R\tilde{x}$, the FE discrete trajectory is

$$\begin{aligned} &\sum_{k=0}^{3}\left[(1 + \frac{\lambda_k}{\tau}\Delta t)^{t/\Delta t}\left(w_k^T x(0) + \frac{0.5 w_{k,3}}{\lambda_k}\right) - \frac{0.5 w_{k,3}}{\lambda_k}\right] r_k, \\ &= \sum_{k=0}^{3}\left[G(\frac{\Delta t}{\tau}, k)^{t/\Delta t}\left(w_k^T x(0) + \frac{0.5 w_{k,3}}{\lambda_k}\right) - \frac{0.5 w_{k,3}}{\lambda_k}\right] r_k, \quad t = n\Delta t. \end{aligned} \tag{5.39}$$

From (5.32), the growth factor $G(\frac{\Delta t}{\tau}, k)$ is

$$G(\frac{\Delta t}{\tau}, k) = 1 + \frac{\lambda_k}{\tau}\Delta t = 1 + \frac{\Delta t}{\tau}\left[g\cos\left((k+\frac{1}{2})\frac{\pi}{N}\right) - 1\right],\ k = 0, \dots, N-1$$

with

$$G(\frac{\Delta t}{\tau}, 0) = 1 + \frac{\Delta t}{\tau}\left[g\cos(\frac{\pi}{2N}) - 1\right] > G(\frac{\Delta t}{\tau}, 1) > \cdots > G(\frac{\Delta t}{\tau}, N-1).$$

To avoid numerical instability, require

$$G(\frac{\Delta t}{\tau}, N-1) = 1 + \frac{\Delta t}{\tau}\left[g\cos(\frac{(2N-1)\pi}{2N}) - 1\right] > -1.$$

This sets the constraint on the maximum time step as

$$\frac{\Delta t}{\tau} < \frac{2}{1 - g\cos(\frac{(2N-1)\pi}{2N})},\quad N \geq 2. \tag{5.40}$$

Note if $\frac{\Delta t}{\tau}$ satisfies (5.28), then (5.40) holds.

The growth factor is largest at $k = 0$ with the value:

$$G(\frac{\Delta t}{\tau}, 0) = 1 + \frac{\Delta t}{\tau}\left[g\cos(\frac{\pi}{2N}) - 1\right].$$

If

$$1 < g < \frac{1}{\cos(\frac{\pi}{2N})},\quad \text{for} \quad N \geq 2 \tag{5.41}$$

then

$$G(\frac{\Delta t}{\tau}, 0) < 1.$$

Hence, if (5.40) and (5.41) hold, then no numerical instability occurs and the magnitudes of all growth factors are less than 1,

$$-1 < G(\frac{\Delta t}{\tau}, k) < 1, \quad k = 0, \dots, N-1.$$

Consequently, the discrete trajectory converges to

$$
\begin{aligned}
&\sum_{k=0}^{3}\left(-\frac{0.5w_{k,3}}{\lambda_k}\right)r_k \\
&=\left(-\frac{0.5\frac{\cos 3\theta_0}{2}}{g\cos\frac{\pi}{2N}-1}\right)\begin{bmatrix}\cos 0\\ \cos\theta_0\\ \cos 2\theta_0\\ \cos 3\theta_0\end{bmatrix}+\left(-\frac{0.5\frac{\cos 3\theta_1}{2}}{g\cos\frac{3\pi}{2N}-1}\right)\begin{bmatrix}\cos 0\\ \cos\theta_1\\ \cos 2\theta_1\\ \cos 3\theta_1\end{bmatrix}+\left(-\frac{0.5\frac{\cos 3\theta_2}{2}}{g\cos\frac{5\pi}{2N}-1}\right)\begin{bmatrix}\cos 0\\ \cos\theta_2\\ \cos 2\theta_2\\ \cos 3\theta_2\end{bmatrix} \\
&+\left(-\frac{0.5\frac{\cos 3\theta_3}{2}}{g\cos\frac{7\pi}{2N}-1}\right)\begin{bmatrix}\cos 0\\ \cos\theta_3\\ \cos 2\theta_3\\ \cos 3\theta_3\end{bmatrix}
\end{aligned}
\tag{5.42}
$$

(5.42) implies the discrete trajectory converges to all ones independent of the initial state values.

On the other hand, if

$$
\begin{aligned}
&\frac{1}{\cos(\frac{\pi}{2N})}<g<g_{th}=\frac{1}{\cos(\frac{\pi}{N})}=\frac{1}{\cos(\frac{2\pi}{2N})},\quad N\geq 3\\
&g>\frac{1}{\cos(\frac{\pi}{2N})},\quad N=2
\end{aligned}
\tag{5.43}
$$

then

$$
\begin{aligned}
G(\frac{\Delta t}{\tau},0)&=1+\frac{\Delta t}{\tau}\left[g\cos(\frac{\pi}{2N})-1\right]>1,\\
G(\frac{\Delta t}{\tau},1)&=1+\frac{\Delta t}{\tau}\left[g\cos(\frac{3\pi}{2N})-1\right]<1.
\end{aligned}
$$

Hence, if (5.40) and (5.43) hold, then no numerical instability occurs and

$$
G(\frac{\Delta t}{\tau},0)>1,\quad -1<G(\frac{\Delta t}{\tau},k)<1,\quad k=1,\ldots,N-1.
$$

Consequently, for $N = 4$, the discrete trajectory converges to

$$
\begin{aligned}
& G(\frac{\Delta t}{\tau}, 0)^n \left(w_0^T x(0) + \frac{0.5 w_{0,3}}{\lambda_0} \right) r_0, \quad n = \frac{t}{\Delta t} \\
&= G(\frac{\Delta t}{\tau}, 0)^n \left(\frac{1}{4} x_0(0) + \frac{\cos\theta_0}{2} x_1(0) + \frac{\cos 2\theta_0}{2} x_2(0) + \frac{\cos 3\theta_0}{2} x_3(0) + \frac{0.5 \frac{\cos 3\theta_0}{2}}{\lambda_0} \right) r_0 \\
&= G(\frac{\Delta t}{\tau}, 0)^n \left(\frac{1}{4} x_0(0) + \frac{\cos\theta_0}{2} x_1(0) + \frac{\cos 2\theta_0}{2} x_2(0) + \frac{\cos 3\theta_0}{2} x_3(0) + \frac{0.5 \frac{\cos 3\theta_0}{2}}{g \cos \frac{\pi}{2N} - 1} \right) r_0, \\
&= G(\frac{\Delta t}{\tau}, 0)^n \left(0.25 x_0(0) + 0.46 x_1(0) + 0.35 x_2(0) + 0.19 x_3(0) + \frac{0.5 \times 0.19}{g \cos \frac{\pi}{2N} - 1} \right) r_0,
\end{aligned}
\tag{5.44}
$$

where from Appendix C, the r_0 and w_0^T are

$$
\begin{aligned}
r_0 &= \begin{bmatrix} \cos 0 \\ \cos\theta_0 \\ \cos 2\theta_0 \\ \cos 3\theta_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.92 \\ 0.70 \\ 0.38 \end{bmatrix} \\
w_0^T &= [\frac{1}{4} \quad \frac{\cos\theta_0}{2} \quad \frac{\cos 2\theta_0}{2} \quad \frac{\cos 3\theta_0}{2}] = [0.25 \quad 0.46 \quad 0.35 \quad 0.19] \\
\theta_0 &= \frac{\pi}{2N} = \frac{\pi}{8}.
\end{aligned}
$$

(5.44) also implies if all initial states are positive, then the discrete trajectory converges to all ones.

5.2.6 Both End Neurons Enter the Saturation Regions with the Same Sign

If both end neuron enter the sarutation region, say x_0, x_4 in (5.1), then we consider the three ($Q = P - 2 = 3$) neurons, x_1, x_2, x_3. The numbering for the states x_i goes from 1 to

Q. The set of dynamic equations in this case is expressed as

$$\begin{aligned}
\tau\dot{x}_1 &= -x_1 + 0.5I_{sat}R + 0.5gx_2 \\
&= 0.5g(\frac{I_{sat}R}{g} + x_2 - 2x_1) + x_1(g-1) \\
\tau\dot{x}_2 &= -x_2 + 0.5gx_1 + 0.5gx_3 \\
&= 0.5g(x_1 + x_3 - 2x_2) + x_2(g-1) \\
\tau\dot{x}_3 &= -x_3 + 0.5gx_2 + 0.5I_{sat}R \\
&= 0.5g(x_2 + \frac{I_{sat}R}{g} - 2x_3) + x_3(g-1).
\end{aligned} \tag{5.45}$$

Express (5.45) as $\tau\dot{x} = Ax + u,\ \dot{x} = \frac{1}{\tau}Ax + \frac{1}{\tau}u$ where

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad u = \begin{bmatrix} 0.5I_{sat}R \\ 0 \\ 0.5I_{sat}R \end{bmatrix}, \quad A = \begin{bmatrix} -1 & .5g & 0 \\ 0.5g & -1 & 0.5g \\ 0 & 0.5g & -1 \end{bmatrix}. \tag{5.46}$$

The eigenvalues and eigenvectors of A in (5.46) are derived in Appendix B as

$$\begin{aligned}
\lambda_k &= g\cos\left(\frac{k\pi}{Q+1}\right) - 1, \quad k = 1, \ldots, Q \\
\lambda_1 &= g\cos\frac{\pi}{Q+1} - 1 > \lambda_2 > \lambda_Q = g\cos(\frac{Q\pi}{Q+1}) - 1
\end{aligned} \tag{5.47}$$

and the jth component of the kth eigenvector r_k is

$$r_{k,j} = \sin\left(j\frac{k\pi}{Q+1}\right) = \sin(j\theta_k), \quad j = 1, \ldots, Q, \quad k = 1, \ldots, Q.$$

Because A is symmetric, an orthogonal set of eigenvectors is available. Therefore, we assume that $Ar_k = \lambda_k r_k, k = 1, \ldots, Q,\ \ Q = 3$, and have

$$AR = R\Lambda,\ A = R\Lambda R^{-1},\ \Lambda = R^{-1}AR,$$

where for $Q = 3$

$$R = [r_1\ \ r_2\ \ r_3] = \begin{bmatrix} \sin\frac{\pi}{4} & \sin\frac{2\pi}{4} & \sin\frac{3\pi}{4} \\ \sin\frac{2\pi}{4} & \sin\frac{4\pi}{4} & \sin\frac{6\pi}{4} \\ \sin\frac{3\pi}{4} & \sin\frac{6\pi}{4} & \sin\frac{9\pi}{4} \end{bmatrix}, \quad R^TR = \frac{Q+1}{2}I$$

$$\Lambda = \mathbf{diag}(\lambda_1, \lambda_2, \lambda_3),\ R^{-1} = \begin{bmatrix} w_1^T \\ w_2^T \\ w_3^T \end{bmatrix} = \frac{2}{Q+1}R^T = \frac{2}{Q+1}R.$$

Using the same approaches as in (5.34), (5.35), (5.36), and the normalization condition of $I_{sat} = 1, R = 1, C = 1$, we have

$$u = \begin{bmatrix} 0.5 \\ 0 \\ 0.5 \end{bmatrix}, \tau = RC = 1,$$

and the FE discrete trajectory is

$$\begin{aligned}
&\sum_{k=1}^{3}\left[(1+\frac{\lambda_k}{\tau}\Delta t)^{t/\Delta t}\left(w_k^T x(0)+\frac{0.5w_{k,1}+0.5w_{k,3}}{\lambda_k}\right)-\frac{0.5w_{k,1}+0.5w_{k,3}}{\lambda_k}\right]r_k \\
&=\sum_{k=1}^{3}\left[G(\frac{\Delta t}{\tau},k)^{t/\Delta t}\left(w_k^T x(0)+\frac{0.5w_{k,1}+0.5w_{k,3}}{\lambda_k}\right)-\frac{0.5w_{k,1}+0.5w_{k,3}}{\lambda_k}\right]r_k. \qquad (5.48)
\end{aligned}$$

From (5.47), the growth factor $G(\frac{\Delta t}{\tau}, k)$ is

$$G(\frac{\Delta t}{\tau},k) = 1+\frac{\lambda_k}{\tau}\Delta t = 1+\frac{\Delta t}{\tau}\left[g\cos\left(k\frac{\pi}{Q+1}\right)-1\right],\ k = 1,\dots,Q$$

with

$$G(\frac{\Delta t}{\tau},1) = 1+\frac{\Delta t}{\tau}\left[g\cos(\frac{\pi}{Q+1})-1\right] > G(\frac{\Delta t}{\tau},2) > \cdots > G(\frac{\Delta t}{\tau},Q).$$

To avoid numerical instability, require

$$G(\frac{\Delta t}{\tau},Q) = 1+\frac{\Delta t}{\tau}\left[g\cos(\frac{Q\pi}{Q+1})-1\right] > -1.$$

This sets the constraint on the maximum time step as

$$\frac{\Delta t}{\tau} < \frac{2}{1-g\cos(\frac{Q\pi}{Q+1})},\quad Q \geq 2. \qquad (5.49)$$

Note if $\frac{\Delta t}{\tau}$ satisfies (5.28), then (5.49) holds.

To analyze the growth factor at $k = 1$, consider

$$G(\frac{\Delta t}{\tau},1) = 1+\frac{\Delta t}{\tau}\left[g\cos(\frac{\pi}{Q+1})-1\right].$$

If

$$1 < g < \frac{1}{\cos(\frac{\pi}{Q+1})},\quad \text{for}\quad Q \geq 2 \qquad (5.50)$$

then

$$G(\frac{\Delta t}{\tau}, 1) < 1.$$

Note from (5.27) and $N = P - 1,\ Q = N - 1$

$$g_{th} = \frac{1}{\cos\frac{\pi}{N}} = \frac{1}{\cos\frac{\pi}{P-1}} = \frac{1}{\cos\frac{\pi}{Q+1}}.$$

Hence, if (5.49) and (5.50) hold, then no numerical instability occurs and the maginitude of all growth factors are less than 1,

$$-1 < G(\frac{\Delta t}{\tau}, k) < 1, \quad k = 1, \ldots, Q.$$

Consequently, the discrete trajectory converges to

$$\begin{aligned}
&\sum_{k=1}^{3}\left(-\frac{0.5w_{k,1} + 0.5w_{k,3}}{\lambda_k}\right) r_k \\
&= 0.5\left(-\frac{w_{1,1} + w_{1,3}}{\lambda_1}\right) r_1 + 0.5\left(-\frac{w_{2,1} + w_{2,3}}{\lambda_2}\right) r_2 + 0.5\left(-\frac{w_{3,1} + w_{3,3}}{\lambda_3}\right) r_3 \\
&= \frac{-1}{Q+1}\left(\frac{\sin\frac{\pi}{4} + \sin\frac{3\pi}{4}}{g\cos\frac{\pi}{4} - 1}\right) r_1 + \frac{-1}{Q+1}\left(\frac{\sin\frac{2\pi}{4} + \sin\frac{6\pi}{4}}{g\cos\frac{2\pi}{4} - 1}\right) r_2 + \frac{-1}{Q+1}\left(\frac{\sin\frac{3\pi}{4} + \sin\frac{9\pi}{4}}{g\cos\frac{3\pi}{4} - 1}\right) r_3 \\
&= \frac{-1}{4}\left(\frac{\sin\frac{\pi}{4} + \sin\frac{3\pi}{4}}{g\cos\frac{\pi}{4} - 1}\right)\begin{bmatrix}\sin\frac{\pi}{4}\\ \sin\frac{2\pi}{4}\\ \sin\frac{3\pi}{4}\end{bmatrix} + \frac{-1}{4}\left(\frac{\sin\frac{2\pi}{4} + \sin\frac{6\pi}{4}}{g\cos\frac{2\pi}{4} - 1}\right)\begin{bmatrix}\sin\frac{2\pi}{4}\\ \sin\frac{4\pi}{4}\\ \sin\frac{6\pi}{4}\end{bmatrix} \\
&+ \frac{-1}{4}\left(\frac{\sin\frac{3\pi}{4} + \sin\frac{9\pi}{4}}{g\cos\frac{3\pi}{4} - 1}\right)\begin{bmatrix}\sin\frac{3\pi}{4}\\ \sin\frac{6\pi}{4}\\ \sin\frac{9\pi}{4}\end{bmatrix} \qquad (5.51)
\end{aligned}$$

(5.51) implies the discrete trajectory converges to all ones independent of the initial state values.

5.3 ARMCNN Convergence Analysis

Even the time step and the neuron gain are designed using (5.27) and (5.28) such that the single mode operation ($k = 0$) holds without numerical instability, the output of FE-ARMCNN may not be equal to the output of CT-ARMCNN. The purpose of this section is to derive a sufficient condition to ensure that the convergent binary output of FE-ARMCNN is equal to the output of the continuous time ARMCNN (CT-ARMCNN). Assume all initial states are in the linear regions. From (5.27) and (5.28), we have shown that if the neuron gain and the normalized time step satisfy

$$1 < g < g_{th} = \frac{1}{\cos(\frac{\pi}{P-1})}, \quad \text{for} \quad P \geq 4$$
$$g > 1 \quad \text{for} \quad P = 2, 3$$

and

$$\frac{\Delta t}{\tau} < \frac{2}{g+1},$$

then no numerical instability occurs and the single mode operation holds in that

$$G(\frac{\Delta t}{\tau}, 0) > 1, \quad \text{and} \ \ |G(\frac{\Delta t}{\tau}, k)| < 1, \ \ k \geq 1. \tag{5.52}$$

Furthermore, if we constrain the initial states such that

$$\left| w_0^T x(0) \right| > \sum_{k=1}^{P-1} \left| w_k^T x(0) \right| \tag{5.53}$$

then from (5.52),

$$\left| G(\frac{\Delta t}{\tau}, 0)^n w_0^T x(0) \right| > \sum_{k=1}^{P-1} \left| G(\frac{\Delta t}{\tau}, k)^n w_k^T x(0) \right| \quad n \geq 0. \tag{5.54}$$

(5.54) implies that the maximum AC amplitude at each jth state is less than the DC amplitude. The reason is as follows. The discrete trajectory from (5.11) and (5.15) is

$$\sum_{k=0}^{P-1} G(\frac{\Delta t}{\tau}, k)^{t/\Delta t} \left(w_k^T x(0) \right) r_k, \quad t = n\Delta t \tag{5.55}$$

with the jth state of the discrete trajectory at $t = n\Delta t$ as

$$\begin{aligned} x_{j,n} &= \sum_{k=0}^{P-1} G(\frac{\Delta t}{\tau}, k)^n \left(w_k^T x(0) \right) r_{k,j}, \quad j = 0, \ldots, P-1 \\ &= G(\frac{\Delta t}{\tau}, 0)^n \left(w_0^T x(0) \right) r_{0,j} + \sum_{k=1}^{P-1} G(\frac{\Delta t}{\tau}, k)^n \left(w_k^T x(0) \right) r_{k,j} \\ &= G(\frac{\Delta t}{\tau}, 0)^n \left(w_0^T x(0) \right) + \sum_{k=1}^{P-1} G(\frac{\Delta t}{\tau}, k)^n \left(w_k^T x(0) \right) r_{k,j}, \quad |r_{k,j}| \leq 1, r_{0,j} = 1 \end{aligned}$$

From (5.54), we have

$$\left|G(\frac{\Delta t}{\tau},0)^n w_0^T x(0)\right| > \sum_{k=1}^{P-1}\left|G(\frac{\Delta t}{\tau},k)^n w_k^T x(0)\right| > \left|\sum_{k=1}^{P-1} G(\frac{\Delta t}{\tau},k)^n w_k^T x(0) r_{k,j}\right| .$$

Therefore, the DC amplitude,

$$\left|G(\frac{\Delta t}{\tau},0)^n \left(w_0^T x(0)\right)\right|$$

is larger than the maximum AC amplitude and

$$x_{j,n}\left(w_0^T x(0)\right) > 0, j = 0, \ldots, P-1$$

which means $x_{j,n}$ has the same sign as $w_0^T x(0)$. As n gets larger, $x_{j,n}$ approaches to

$$G(\frac{\Delta t}{\tau},0)^n \left(w_0^T x(0)\right) .$$

If a particular x_j enters the saturation region, for example, $x_{j'} > \frac{I_{sat}R}{g}$, then replace $x_{j'}$ with $\frac{I_{sat}R}{g}$ for the FE computation of other linear state values in the next time instant as shown in Section 5.2.5. All states keep the same sign as $w_0^T x(0)$. Hence all states grow to the binary outputs as shown in Section 5.2.5 and Section 5.2.6.

We have thus shown that if (5.27), (5.28), and (5.53) hold, then the FE ARMCNN has the convergent binary outputs determined by the sign of $w_0^T x(0)$ for all time steps satisfying

$$\frac{\Delta t}{\tau} < \frac{2}{g+1}.$$

In this case, the convergent binary outputs of the FE ARMCNN are the same as the convergent binary outputs of the continuous time ARMCNN.

5.3.1 Simulation Verification

For example, consider the same 1-D 5N ARMCNN, $P = 5$, with the dynamic equations of (5.1). Applying (5.27) to filter out high frequency components, the neuron gain range is:

$$g < g_{th} = \frac{1}{\cos\frac{\pi}{P-1}} = 1.41.$$

We choose $g = 1.4$. To avoid the numerical instability, the maximum time step from (5.28) is

$$\frac{\Delta t}{\tau} < \frac{2}{g+1} = \frac{2}{1.4+1} = 0.833.$$

Therefore, we choose the normalized time step $\frac{\Delta t}{\tau} = 0.83$. Based on the designed $g = 1.4$ and the $\frac{\Delta t}{\tau} = 0.83$, the corresponding growth factors from (5.23) are

$$\begin{aligned} &G(\frac{\Delta t}{\tau}, 0) = 1.33,\ G(\frac{\Delta t}{\tau}, 1) = 0.9917,\ G(\frac{\Delta t}{\tau}, 2) = 0.17 \\ &G(\frac{\Delta t}{\tau}, 3) = -0.65,\ G(\frac{\Delta t}{\tau}, 4) = -0.9920. \end{aligned}$$

As observed, only $G(\frac{\Delta t}{\tau}, 0)$ is greater than 1, and $-1 < G(\frac{\Delta t}{\tau}, k) < 1,\ k = 1, \ldots, P-1$.

To check whether (5.53) holds or not, the row vectors of the matrix R^{-1} are required. The R^{-1} with $P = 5$ is derived in Appendix C as

$$R^{-1} = \begin{bmatrix} 0.125 & 0.25 & 0.25 & 0.25 & 0.125 \\ 0.25 & 0.3536 & 0 & -0.3536 & -0.25 \\ 0.25 & 0 & -0.5 & 0 & 0.25 \\ 0.25 & -0.3536 & 0 & 0.3536 & -0.25 \\ 0.125 & -0.25 & 0.25 & -0.25 & 0.125 \end{bmatrix} = \begin{bmatrix} w_0^T \\ w_1^T \\ w_2^T \\ w_3^T \\ w_4^T \end{bmatrix}. \tag{5.56}$$

For an initial point $x(0)$

$$x(0) = \begin{bmatrix} x_0(0) \\ x_1(0) \\ x_2(0) \\ x_3(0) \\ x_4(0) \end{bmatrix} = \begin{bmatrix} 0.05 \\ -0.67 \\ 0.14 \\ 0.31 \\ 0.37 \end{bmatrix},$$

we have

$$\left| w_0^T x(0) \right| = 0.0025 < \left| w_1^T x(0) \right| = 0.4265,$$

which imples (5.53) is not satisfied. Hence, if simulated at a smaller time step, the convergent output with this initial point may change. The time domain simulations of the 1-D 5N FE-ARMCNN at two different time steps $\frac{\Delta t}{\tau} = 0.83$ and $\frac{\Delta t}{\tau} = 0.02$ for the same initial point are shown in Figure 5.6 and Figure 5.7, respectively. The larger time step generates all normalized neuron states converged at $(1, 1, 1, 1, 1)$ in Figure 5.6, while the smaller

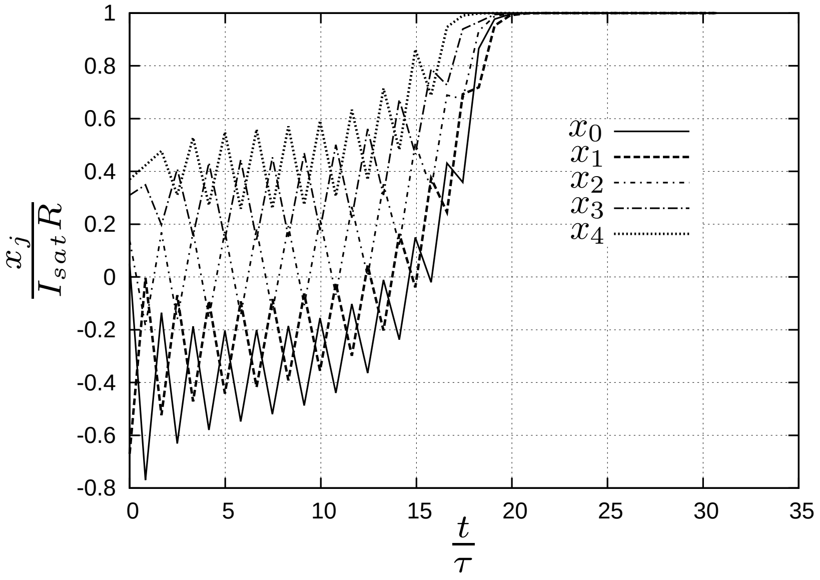


Figure 5.6: The initial point is $(0.05, -0.67, 0.14, 0.31, 0.37)$. $g = 1.4$ and $\frac{\Delta t}{\tau} = 0.83$. The x-axis variable is $\frac{t}{\tau}$ with $t = n\Delta t,\ n = 0, 1, 2, \ldots$.

time step generates all normalized neuron states converged at $(-1, -1, -1, -1, -1)$ in Figure 5.7.

To illustrate the effect of the time step and the neuron gain to the spurious points and numerical instability, five thousand Gaussian noisy input patterns are applied to the 1-D ARMCNN in Figure 5.1 at two different neuron gains with one smaller than g_{th}, $g = 1.4 < g_{th}$, and the other larger than than g_{th}, $g = 2.0 > g_{th}$. Each state x_i is $I_{sat}R$ added with $I_{ni}R$ where I_{ni} is a Gaussian noise with a standard deviation (STD) of $\sqrt{20} \times I_{sat}$. Hence, the input pattern consists of five states as a state vector and is written as:

$$x_i(0) = (I_{sat} + I_{ni}) \times R,\ i = 0, \ldots, 4 \tag{5.57}$$

where

$$\text{if}\ \ (I_{sat} + I_{ni}) \times R > I_{sat}R_i\ \ \text{then}\ \ x_i(0) = I_{sat}R_i$$

$$\text{if}\ \ (I_{sat} + I_{ni}) \times R < -I_{sat}R_i\ \ \text{then}\ \ x_i(0) = -I_{sat}R_i.$$

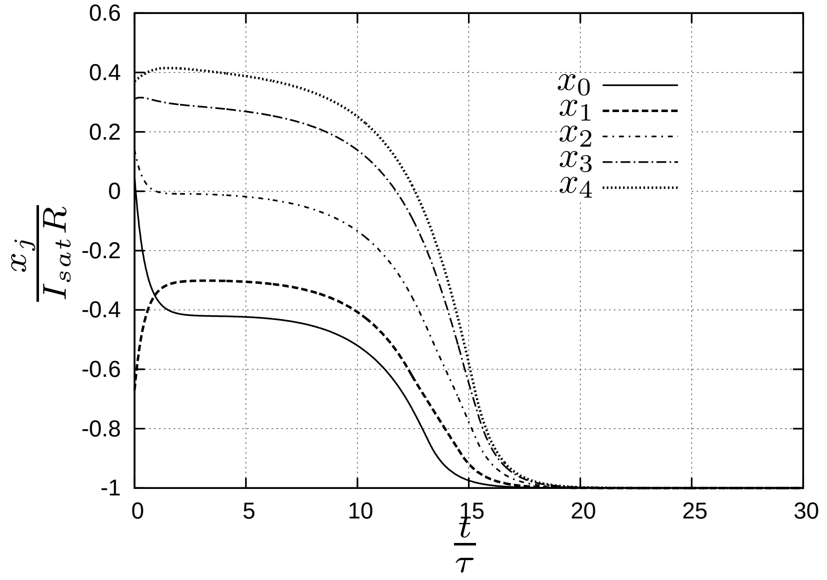


Figure 5.7: The initial point is $(0.05, -0.67, 0.14, 0.31, 0.37)$. $g = 1.4$ and $\frac{\Delta t}{\tau} = 0.02$. The x-axis variable is $\frac{t}{\tau}$ with $t = n\Delta t,\ n = 0, 1, 2, \ldots$.

Let $Eq_1 = (I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R)$, and $Eq_2 = -(I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R)$. Six different sets of initial state vectors are classified in Figure 5.8 as

- set A: these state vectors are closer to Eq_1 than Eq_2 and converge to point Eq_1
- set B: these state vectors are closer to Eq_2 than Eq_1 and converge to Eq_2
- set C: these state vectors are closer to Eq_1 than Eq_2 and converge to Eq_2
- set D: these state vectors are closer to Eq_2 than Eq_1 and converge to Eq_1
- set E: these state vectors cause the numerical instability, and can not converge
- set F: these state vectors converge to non-binary points, or spurious points.

Figure 5.9 shows the number of state vectors in set A,B,C,D as the normalized FE time step, $\frac{\Delta t}{\tau}$, varies from 0.1 to 2.0 under a fixed neuron gain of 1.4 and a fixed noise STD of $\sqrt{20} \times I_{sat}$ in (5.57). This noise STD of $\sqrt{20} \times I_{sat}$ is based on the normalize distance between Eq1 and Eq2. The state vectors in set A,B,C,D converge to

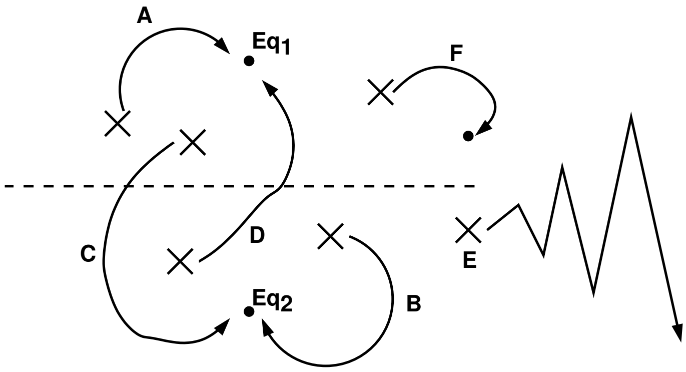


Figure 5.8: Different sets of initial states are shown. The optimal decision boundary line is shown in the dotted line.

one of the two binary points, Eq1 or Eq2. Because the mean of the input pattern is $Eq_1 = (I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R, I_{sat}R)$, Figure 5.9 shows that set A has a larger amount of points than set B. As $\frac{\Delta t}{\tau}$ increases over 0.83, the numbers of state vectors in set A,B,C,D decreases significantly. This is because the amount of points in set E starts to increase, or numerical instability kicks in as shown in Figure 5.10. In addition, only a spurious point $(0, 0, 0, 0, 0)$ was observed from our simulation results. This is consistent with (5.29).

If the neuron gain is 2.0, then the single mode operation is not valid. Figure 5.11 shows the number of state vectors in set A,B,C,D as the normalized FE time step, $\frac{\Delta t}{\tau}$, varies under a fixed neuron gain of 2.0 and a fixed noise variance of 20 in (5.57). Compared with Figure 5.9, the number of points in set A,B,C,D decrease, and more spurious points were observed. Numerical instability kicks in as the normalized time step increases over $\frac{2}{g+1} = \frac{2}{3}$ as shown in Figure 5.12.

We summarize key results for 1-D ARMCNN as

- For the matrice in (5.5), eigenvectors are not not dependent on the neuron gain, whereas eigenvalues depend on the neuron gain. Each eigenvector sequence represents different frequency component as shown in Figure 5.4. The eigenvector sequence of $k = 0$ is the all-ones DC component.

- Normally, it is hard to compute the eigenvector in (5.5) and obtain the frequency components. Interestingly, these eigenvectors are exactly the columns of DCT-1 matrix. Fast FFT algorithm can be applied to compute matrix multiplication in

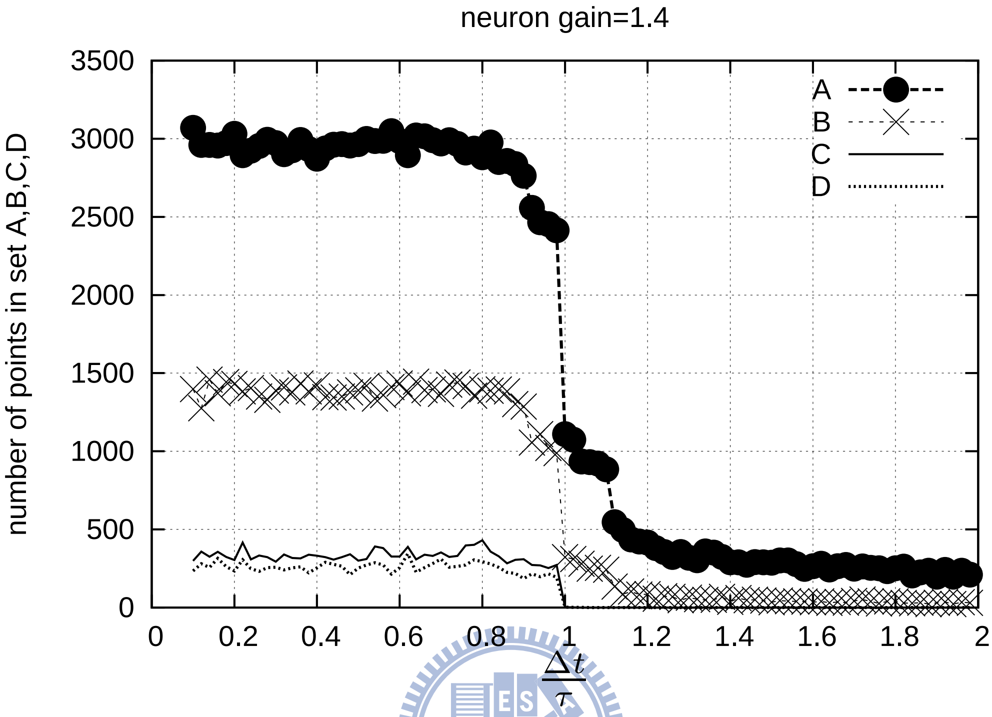


Figure 5.9: The numbers of initial state vectors in set A,B,C,D are plotted at various time steps. The neuron gain is 1.4. The simulation time is 200τ.

(5.21).

- To reduce the amount of spurious points, filter out high frequency components of the initial state vector by making the corresponding growth factor maginitudes smaller than 1. This low pass filtering is done by (5.27). To avoid numerical instability, choose $\frac{\Delta t}{\tau}$ to satisfy (5.28).

Recognition Rate of a 2-D 9 × 9 *ARMCNN*

This example performs behavior simulations for 9 × 9 ARMCNNs with PWL activation functions at different neuron gains. The input patterns to be learned and recognized are three Chinese characters of ONE, TWO, and FOUR, as Figure 3.1 shows. Figure 3.2 shows the survived ratio weights. One hundred noisy patterns for each Chinese character

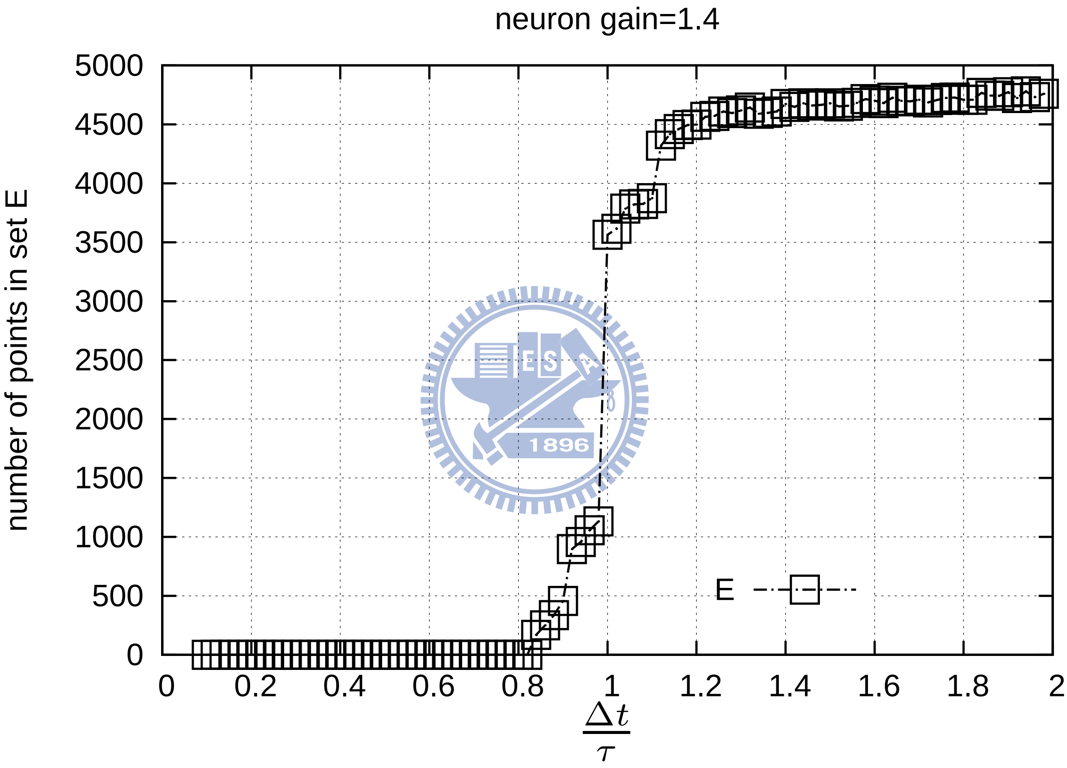


Figure 5.10: The number of state vectors in set E increases as $\frac{\Delta t}{\tau}$ is larger than 0.83.

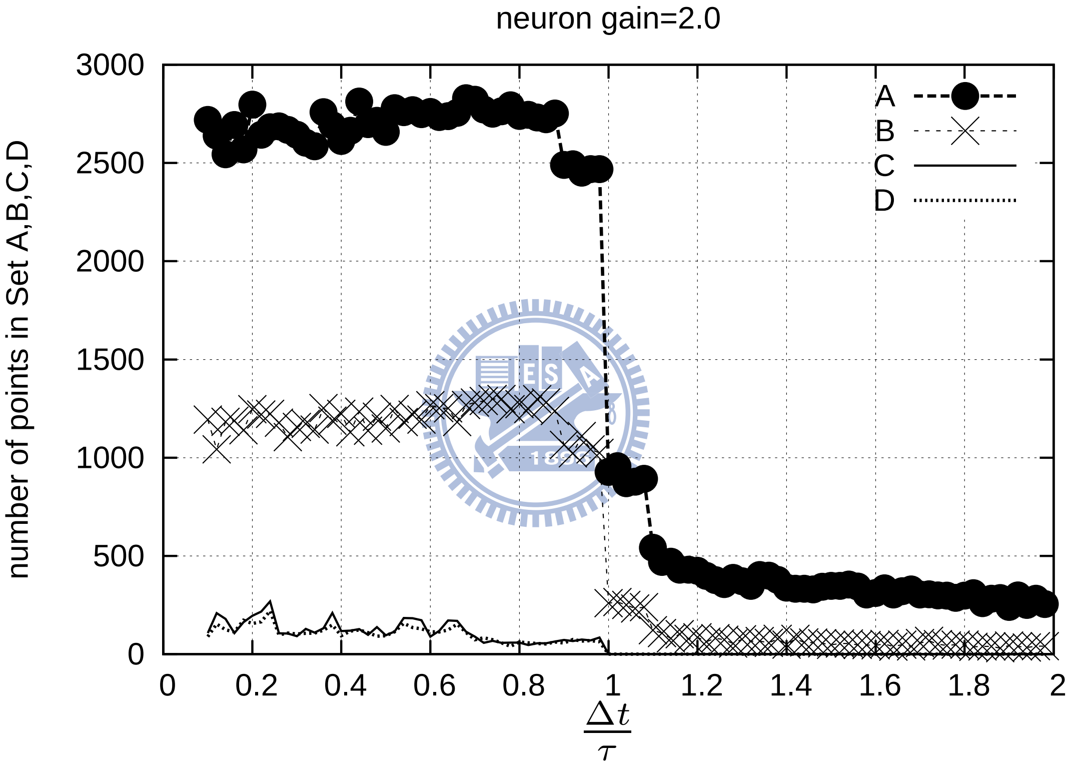


Figure 5.11: The numbers of initial state vectors in set A,B,C,D are plotted at various time steps. The neuron gain is 2.0. The simulation time is 200τ.

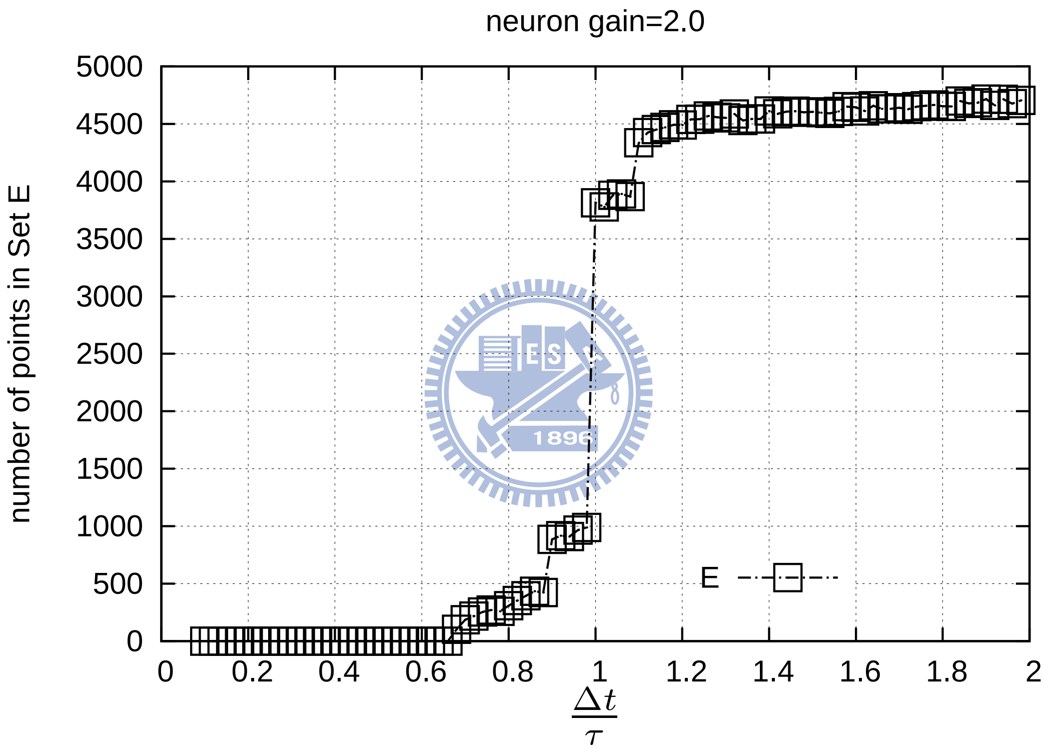


Figure 5.12: The number of state vectors in set E increases as $\frac{\Delta t}{\tau}$ is larger than 0.66.

were generated as

$$x_{ij}(0) = (I_{sat} + I_{n,1}) \times R_{ij} \quad \text{for a black pixel}$$

$$x_{ij}(0) = (-I_{sat} + I_{n,2}) \times R_{ij} \quad \text{for a white pixel}$$

where I_{ni} is a Gaussian noise and

$$\text{if } \ (I_{sat} + I_{n,1}) \times R_{ij} > I_{sat}R_{ij} \ \text{ then } \ x_{ij}(0) = I_{sat}R_{ij}$$

$$\text{if } \ (-I_{sat} + I_{n,1}) \times R_{ij} < -I_{sat}R_{ij} \ \text{ then } \ x_{ij}(0) = -I_{sat}R_{ij}.$$

The RR of a group of m patterns is the number of successful recognitions divided by $100 \times m$. The output variable y/I_{sat} was compared with the correct one. All the 81 pixels must be correct to produce a successful recognition. The simulations used the forward Euler method at different neuron gains with the simulation time of 100τ. Figure 5.13 shows that when the standard deviation of the input noise level is smaller than $0.4 \times I_{sat}$, the RR is almost unity. No numerical instability was found using these time steps. In addition, as the neuron gain increases, more spurious points are generated to deteriorate the RR. In the optimal case of no spurious points, the RR has a upper bound constrained by the six 1-D 2N system in Figure 3.2. The six 1-D 2N system has a recognition rate of RR_{2N}^{6} [8]. Similar results were also found in SARMCNNs [39].

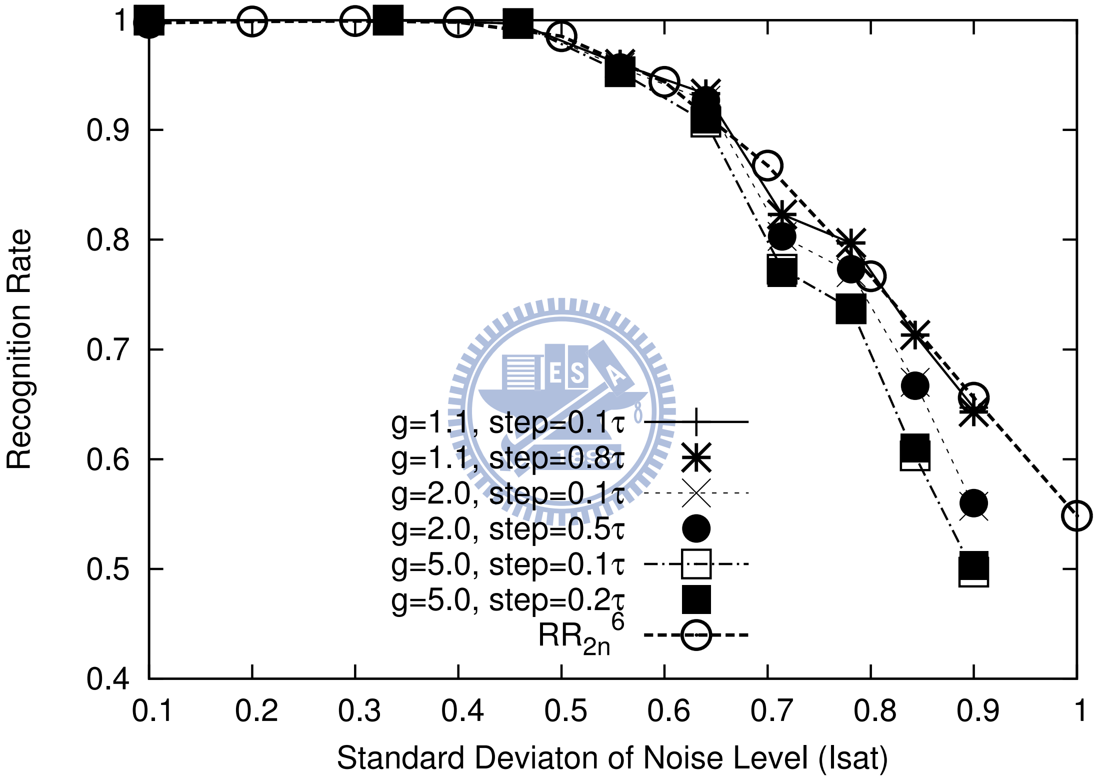


Figure 5.13: The RR of recognizing three 9 × 9 Chinese characters ONE, TWO, and FOUR by ARMCNNs.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

This thesis examines the stability analysis of the ARMCNN via the Lyapunov theorem. Results show that the ARMCNN can tolerate large ratio weight variations and the activation function can vary as long as (4.6) and (4.7) are satisfied. In addition to the robustness of ARMCNNs, the learning rule is simple and therefore is suitable for analog VLSI implementation. Further, a conservative DOA can easily be determined from the stability proof using a simple graphical method. Table 6.1 shows the comparison between ARMCNNs and other CNNs on associative memory. The signal range ratio (SR ratio) is defined as $x/(I_{sat}R)$ divided by y/I_{Isat}. As shown in Table 6.1, for ARMCNNs, the generation of weights is from Hebbian learning. This biology-like Hebbian learning depends on the correlation between neighboring neurons. If after learning, many isolated neurons are generated, then the RR is dominated by these isolated neurons. For other CNNs, the generations of weights are from SVD, LMI, GEVM, and pseudoinverse operations. Integrating these operations into each CNN cell as analog VLSI is more difficult.

The primary problem for ARMCNNs without self-feedback is the occurrence of isolated neurons due to low correlation between neighboring neurons. SARMCNN solves the problem of isolated neurons by self-feedback. Further, for each subsystem, spurious memory points may exist besides the two binary equilibrium points. The occurence of spurious memory points will reduce the RR. To reduce the amount of spurious memory

Table 6.1: Comparison of various CNN works on associative memory.

	This work	[36, 35]	[33, 34]	[37]
$\sigma(\bullet)$	Theorem 1	PWL	PWL	PWL
DOA	Theorem 2	LMI, GEVM	-	-
Analog VLSI	[32]	-	-	-
Synapse Circuit	V/Is,current mirrors	-	-	-
SR Ratio ≤ 1	yes	no	no	no
Weight	Hebbian learning	LMI, GEVM	SVD	pseudoinverse

points, this thesis suggests to lower the neuron gain so that only low frequency components of initial state vectors dominate. An analytical neuron gain range is developed for 1-D ARMCNN.

From the computer simulation and discrete implementation view poins, this thesis obtains the maximum forward Euler (FE) time step to avoid numerical instability. In addition, a sufficent condition is found to ensure that the output from the FE method is the same as the output from the continuous time CNN.

6.2 Recommendations for Future Investigation

This section presents several suggestions for future investigations in circuit design and computer simulation for ARMCNNs.

- The proposed domain of attraction of ARMCNN ensures that the ARMCNN is stable. However the DOA is conservative. Future investigation are needed to enlarge the DOA.

- ARMCNN can only store and recall static patterns. However, the human neocortex can process sequences of dynamic patterns. Future research should be undertaken to modify the simple CNN cell to store dynamic patterns.

- For the continuous time current mode CNN, the synapse weight circuit between neighboring neurons is composed of two V/Is and two current mirrors. The layout area is still too large for a high density CNN array. It would be interesting to assess the discrete-time ARMCNN or the FPGA ARMCNN.

- We expand the initial state vector of 1-D ARMCNN with an analytic form to obtain the corresponding growth factor. However, for SARMCNNs and 2-D ARMCNNs, boundary conditions of the corrsponding diffusion models are not known. It is still difficult to have the analytic eigenvector decomposition for SARMCNNs and 2-D ARMCNNs.

Appendix A

Bitmap File format

This thesis assumes an eight-bit grey levels per pixel in the image BMP file format. An all zero byte (00H) corrsponds to a black pixel, meaning no light at all. An all one byte (FFH) corresponds to a white pixel. In the CNN state representation in (2.1), x state value of 1 means a black pixel, whereas x state value of -1 means a white pixel. Therefore the transforamtion formula between the BMP file format and the read-in CNN state value is as,

$$x(0) = \frac{255 - 2Bt}{255} \tag{A.1}$$

where Bt is the read-in byte from the BMP file from 00h to FFh. From this formula, a byte of 00h (a black pixel) from the BMP transforms to CNN state value of 1. A byte of FFh (255, a wite pixel) from the BMP transforms to CNN state value of -1. Similarly, a byte of 7Fh (127) transforms to a CNN state value of $\frac{1}{255}$, and a byte of 7Eh (126) transforms to a CNN state value of $\frac{3}{255}$. The transformation result is illustrated in Figure A.1.

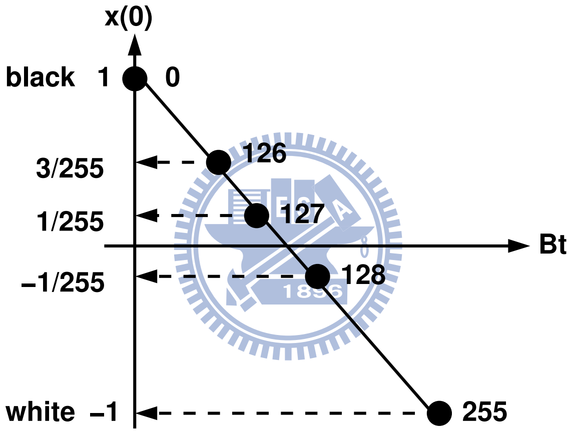


Figure A.1: Transformation between the read-in byte from the BMP file and the CNN state value. Bt denotes the read-in byte.

Appendix B

Eigenvalues of 1-D ARMCNN

B.1 All Neuros in Linear Regions

It is well known that all the eigenvalues of a symmetric matrix are real. The A matrix in (5.5) is almost symmetric. To obtain the eigvalues of the A matrix in (5.5), first consider the interior rows. The coefficients are $0.5g, -1, 0.5g$. From the interior components ($j = 1, \dots, P-2$) of $Ar_k = \lambda_k r_k$, we have

$$\begin{aligned}
&0.5g\cos\left((j+1)\theta_k\right) - \cos(j\theta_k) + 0.5g\cos\left((j-1)\theta_k\right) \\
&= 2 \times 0.5g\left(\cos(j\theta_k)\cos(\theta_k)\right) - \cos(j\theta_k) \\
&= \left(g\cos(\theta_k) - 1\right)\cos(j\theta_k) \\
&= \lambda_k \cos(j\theta_k)
\end{aligned}$$

where $\theta_k = \frac{k\pi}{P-1}, \quad r_{k,j} = \cos(j\theta_k)$.

It remains to show the first ($j = 0$) and last ($j = P-1$) components of $Ar_k = \lambda_k r_k$.

Using trigonometric identity of $\cos(\alpha-\beta)=\cos\alpha\cos\beta+\sin\alpha\sin\beta$, we have

$$\begin{aligned}
j=0: &\quad -\cos 0 + g\cos\theta_k = (g\cos(\theta_k)-1)\cos(0\theta_k) = \lambda_k\cos(0\theta_k) = \lambda_k r_{k,j} \\
j=P-1: &\quad -\cos\left((P-1)\theta_k\right) + g\cos\left((P-2)\theta_k\right) \\
&\quad = -\cos(k\pi) + g\cos(\frac{(P-2)k\pi}{P-1}) \\
&\quad = -\cos(k\pi) + g\cos(k\pi - \frac{k\pi}{P-1}) \\
&\quad = -\cos(k\pi) + g\cos(k\pi)\cos(\frac{k\pi}{P-1}) \\
&\quad = (-1 + g\cos(\frac{k\pi}{P-1}))\cos(k\pi) \\
&\quad = (g\cos(\theta_k)-1)\cos\left((P-1)\theta_k\right) = \lambda_k\cos\left((P-1)\theta_k\right) = \lambda_k r_{k,j}
\end{aligned}$$

Therefore, we have obtained the eigenvalues and eigenvectors of the A matrix in (5.5). The eigenvalues and eigenvectors are shown in (5.19) and (5.20), respectively.

B.2 One End Neuron in Saturation Region

To show the eigenvalues of the A matrix in (5.31), first consider the interior rows. The coefficients are $0.5g, -1, 0.5g$. From the interior components ($j = 1, \ldots, N-2$) of $Ar_k = \lambda_k r_k$, we have

$$\begin{aligned}
&0.5g\cos\left((j+1)\theta_k\right) - \cos(j\theta_k) + 0.5g\cos\left((j-1)\theta_k\right) \\
&= 2\times 0.5g\left(\cos(j\theta_k)\cos(\theta_k)\right) - \cos(j\theta_k) \\
&= (g\cos(\theta_k)-1)\cos(j\theta_k) \\
&= \lambda_k\cos(j\theta_k)
\end{aligned}$$

where $\theta_k = (k+\frac{1}{2})\frac{\pi}{N},\ N\theta_k = \pi(k+\frac{1}{2})$.

It remains to show the first ($j=0$) and last ($j=N-1$) components of $Ar_k = \lambda_k r_k$. Using trigonometric identity of $\cos(\alpha-\beta)=\cos\alpha\cos\beta+\sin\alpha\sin\beta$ and $\cos(N\theta_k)=0$, we have

$$\begin{aligned}
j = 0: \quad & -\cos 0 + g\cos\theta_k = (g\cos(\theta_k) - 1)\cos(0\theta_k) = \lambda_k \cos(0\theta_k) \\
j = N-1: \quad & -\cos\left((N-1)\theta_k\right) + 0.5g\cos\left((N-2)\theta_k\right) \\
& = -\cos\left((N-1)\theta_k\right) + 0.5g(\cos(N\theta_k)\cos(2\theta_k) + \sin(N\theta_k)\sin(2\theta_k)) \\
& = -\cos\left((N-1)\theta_k\right) + 0.5g\sin(N\theta_k)\sin(2\theta_k) \\
& = -\cos\left((N-1)\theta_k\right) + g\sin(N\theta_k)\sin(\theta_k)\cos(\theta_k) \qquad \text{(B.1)}
\end{aligned}$$

On the other hand,

$$\begin{aligned}
& \lambda_k r_{k,N-1} = \lambda_k \cos\left((N-1)\theta_k\right) \\
& = (g\cos(\theta_k) - 1)\cos\left((N-1)\theta_k\right) \\
& = g\cos(\theta_k)\cos\left((N-1)\theta_k\right) - \cos\left((N-1)\theta_k\right) \\
& = g\cos(\theta_k)(\cos(N\theta_k)\cos(\theta_k) + \sin(N\theta_k)\sin(\theta_k)) - \cos\left((N-1)\theta_k\right) \\
& = g\cos(\theta_k)\sin(N\theta_k)\sin(\theta_k) - \cos\left((N-1)\theta_k\right) \qquad \text{(B.2)}
\end{aligned}$$

(B.1) is equal to (B.2), hence, we have obtained the eigenvalues and eigenvectors of the A matrix in (5.31). Hnce, the eigenvalues and eigenvectors of A in (5.31) are

$$\begin{aligned}
& \lambda_k = g\cos\left((k+\frac{1}{2})\frac{\pi}{N}\right) - 1, \quad k = 0, \ldots, N-1 \\
& \lambda_0 = g\cos\frac{\pi}{2N} - 1 > \lambda_1 > \lambda_2 > \lambda_{N-1} = g\cos(\frac{2N-1}{2N}\pi) - 1
\end{aligned}$$

and the jth component of the kth eigenvector r_k is

$$r_{k,j} = \cos\left(j(k+\frac{1}{2})\frac{\pi}{N}\right) = \cos(j\theta_k), \quad j = 0, \ldots, N-1, \quad k = 0, \ldots, N-1.$$

B.3 Both End Neurons Enter the Saturation Region

To show the eigvalues of the A matrix in (5.46), first consider the interior rows. The coefficients are $0.5g, -1, 0.5g$. From the interior components ($j = 2, \ldots, Q-1$) of $Ar_k =$

$\lambda_k r_k$, we have

$$\begin{aligned}
&0.5g\sin\left((j-1)\theta_k\right) - \sin(j\theta_k) + 0.5g\sin\left((j+1)\theta_k\right)\\
&= 2\times 0.5g\left(\sin(j\theta_k)\cos(\theta_k)\right) - \sin(j\theta_k)\\
&= \left(g\cos(\theta_k)-1\right)\sin(j\theta_k)\\
&= \lambda_k\sin(j\theta_k)
\end{aligned}$$

where $\theta_k = \frac{k\pi}{Q+1}$, $(Q+1)\theta_k = k\pi$.

It remains to show the first ($j=1$) and last ($j=Q$) components of $Ar_k = \lambda_k r_k$. Using trigonometric identity of $\sin(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta$ and $\sin((Q+1)\theta_k) = 0$, we have

$$\begin{aligned}
j=1: \quad & -\sin\theta_k + 0.5g\sin(2\theta_k) = -\sin\theta_k + g\sin\theta_k\cos\theta_k\\
&= \left(g\cos(\theta_k)-1\right)\sin\theta_k\\
&= \lambda_k\sin\theta_k = \lambda_k r_{k,j}\\
j=Q: \quad & 0.5g\sin((Q-1)\theta_k) - \sin(Q\theta_k) = 0.5g(\sin(Q\theta_k)\cos\theta_k - \cos(Q\theta_k)\sin\theta_k) - \sin(Q\theta_k)\\
&= g(\sin(Q\theta_k)\cos\theta_k) - \sin(Q\theta_k)\\
&= \left(g\cos(\theta_k)-1\right)\sin(Q\theta_k)\\
&= \lambda_k\sin(Q\theta_k) = \lambda_k r_{k,j}
\end{aligned}$$

Hence, the eigenvalues and eigenvectors of A in (5.46) are

$$\begin{aligned}
\lambda_k &= g\cos\left(\frac{k\pi}{Q+1}\right) - 1, \quad k = 1,\ldots,Q\\
\lambda_1 &= g\cos\frac{\pi}{Q+1} - 1 > \lambda_2 > \lambda_Q = g\cos(\frac{Q\pi}{Q+1}) - 1
\end{aligned}$$

and the jth component of the kth eigenvector r_k is

$$r_{k,j} = \sin\left(j\frac{k\pi}{Q+1}\right) = \sin(j\theta_k), \quad j = 1,\ldots,Q, \;\; k = 1,\ldots,Q.$$

Appendix C

Discrete cosine transform

C.1 All Neurons in the Linear Regions

Appendix C.1 examines the decomposition of the initial states if all neurons are in the linear regions. From (5.21), the decomposition of the initial states are as follows.

$$x(0) = \begin{bmatrix} x_0(0) \\ x_1(0) \\ x_2(0) \\ \vdots \\ x_{P-1}(0) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \cos\theta & \cos 2\theta & \cdots & \cos((P-1)\theta) \\ 1 & \cos 2\theta & \cos 4\theta & \cdots & \cos(2(P-1)\theta) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \cos(P-1)\theta & \cos 2(P-1)\theta & \cdots & \cos((P-1)^2\theta) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{P-1} \end{bmatrix}$$

$$= \begin{bmatrix} r_0 & r_1 & r_2 & \cdots & r_{P-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{P-1} \end{bmatrix} = Ra, \quad a = R^{-1}x(0) = \begin{bmatrix} w_0^T \\ w_1^T \\ w_2^T \\ \vdots \\ w_{P-1}^T \end{bmatrix} x(0), \;\; \theta = \frac{\pi}{P-1}. \tag{C.1}$$

As already proved in Apendix B.1, the kth column of the matrix R is the kth eigenvector of the A matrix in (5.5), $k = 0, \ldots, P-1$. Here, we show how to derive the inverse of the matrix R based on Strang's approach [59]. Applying the similarity transformation

$$\hat{A} = D_1^{-1} A D_1,$$

with $D_1 = \textbf{diag}(\sqrt{2}, 1, \ldots, 1, \sqrt{2})$, and $D_1^{-1} = \textbf{diag}(1/\sqrt{2}, 1, \ldots, 1, 1/\sqrt{2})$, we obtain a symmetric $\hat{A}$ as follows:

$$
\begin{aligned}
\hat{A} &= D_1^{-1} A D_1 \\
&= \begin{bmatrix} 1/\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} -1 & g & 0 & 0 & 0 \\ 0.5g & -1 & 0.5g & 0 & 0 \\ 0 & 0.5g & -1 & 0.5g & 0 \\ 0 & 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0 & g & -1 \end{bmatrix} D_1 \\
&= \begin{bmatrix} -1/\sqrt{2} & g/\sqrt{2} & 0 & 0 & 0 \\ 0.5g & -1 & 0.5g & 0 & 0 \\ 0 & 0.5g & -1 & 0.5g & 0 \\ 0 & 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0 & g/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} \end{bmatrix} \\
&= \begin{bmatrix} -1 & g/\sqrt{2} & 0 & 0 & 0 \\ 0.5g\sqrt{2} & -1 & 0.5g & 0 & 0 \\ 0 & 0.5g & -1 & 0.5g & 0 \\ 0 & 0 & 0.5g & -1 & 0.5g\sqrt{2} \\ 0 & 0 & 0 & g/\sqrt{2} & -1 \end{bmatrix} = \begin{bmatrix} -1 & g/\sqrt{2} & 0 & 0 & 0 \\ g/\sqrt{2} & -1 & 0.5g & 0 & 0 \\ 0 & 0.5g & -1 & 0.5g & 0 \\ 0 & 0 & 0.5g & -1 & g/\sqrt{2} \\ 0 & 0 & 0 & g/\sqrt{2} & -1 \end{bmatrix}.
\end{aligned}
$$

The eigenvalues of $\hat{A}$ are not changed because from $Ax = \lambda x$, we have

$$\hat{A}(D_1^{-1}x) = (D_1^{-1}AD)D_1^{-1}x = D_1^{-1}Ax = \lambda D_1^{-1}x,$$

meaning that the eigenvalue is λ and the eigenvector is $D_1^{-1}x$. Hence, the diagonalization of $\hat{A}$ is

$$\hat{A} = \hat{R}\Lambda\hat{R}^{-1}$$

where

$$\hat{R} = \begin{bmatrix} \hat{r}_0 & \hat{r}_1 & \dots & \hat{r}_{P-1} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \cdots & \frac{1}{\sqrt{2}} \\ 1 & \cos\theta & \cos 2\theta & \cdots & \cos((P-1)\theta) \\ 1 & \cos 2\theta & \cos 4\theta & \cdots & \cos(2(P-1)\theta) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \cos(P-2)\theta & \cos 2(P-2)\theta & \cdots & \cos((P-2)^2\theta) \\ \frac{1}{\sqrt{2}} & \frac{\cos(P-1)\theta}{\sqrt{2}} & \frac{\cos 2(P-1)\theta}{\sqrt{2}} & \cdots & \frac{\cos((P-1)^2\theta)}{\sqrt{2}} \end{bmatrix}$$

$$= D_1^{-1} R.$$

The orthogonal eigenvectors of $\hat{A}$ do not have equal length. $\hat{r}_0$ and $\hat{r}_{P-1}$ have length $\sqrt{P-1}$, while $\hat{r}_k, k = 1, \dots, P-2$, have length $\sqrt{(P-1)/2}$. From the orthogonality of $\hat{r}_k$, we have

$$D_c \hat{R}^T \hat{R} = D_c (D_1^{-1} R)^T (D_1^{-1} R) = D_c R^T (D_1^{-1})^T D_1^{-1} R = I, \tag{C.2}$$

where the D_c compensate the length of $\hat{r}_k$ and is

$$D_c = \mathbf{diag}(\frac{1}{P-1}, \frac{2}{P-1}, \dots, \frac{2}{P-1}, \frac{1}{P-1}).$$

Finally, from (C.2) and $R^T = R, (D_1^{-1})^T = D_1^{-1}$, we obtain

$$\begin{aligned} R^{-1} =& D_c R^T (D_1^{-1})^T D_1^{-1} = D_c R D_1^{-1} D_1^{-1} \\ =& \begin{bmatrix} \frac{1}{2(P-1)} & \frac{1}{P-1} & \frac{1}{P-1} & \cdots & \frac{1}{2(P-1)} \\ \frac{1}{(P-1)} & \frac{2\cos\theta}{P-1} & \frac{2\cos 2\theta}{P-1} & \cdots & \frac{\cos((P-1)\theta)}{P-1} \\ \frac{1}{(P-1)} & \frac{2\cos 2\theta}{P-1} & \frac{2\cos 4\theta}{P-1} & \cdots & \frac{\cos(2(P-1)\theta)}{P-1} \\ \vdots & \vdots & \vdots & & \vdots \\ \frac{1}{2(P-1)} & \frac{\cos(P-1)\theta}{P-1} & \frac{\cos 2(P-1)\theta}{P-1} & \cdots & \frac{\cos((P-1)^2\theta)}{2(P-1)} \end{bmatrix}, \end{aligned} \tag{C.3}$$

where $\theta = \frac{\pi}{P-1}$. For example, in the case of 1-D 5N ARMCNN, $P = 5$, the R^{-1} is

$$R^{-1} = \begin{bmatrix} 0.125 & 0.25 & 0.25 & 0.25 & 0.125 \\ 0.25 & 0.3536 & 0 & -0.3536 & -0.25 \\ 0.25 & 0 & -0.5 & 0 & 0.25 \\ 0.25 & -0.3536 & 0 & 0.3536 & -0.25 \\ 0.125 & -0.25 & 0.25 & -0.25 & 0.125 \end{bmatrix} = \begin{bmatrix} w_0^T \\ w_1^T \\ w_2^T \\ w_3^T \\ w_4^T \end{bmatrix}. \tag{C.4}$$

C.2 One End Neuron Enters the Saturation Region

Appendix C.2 examines the decomposition of states if one end neuron enters the saturation region. From (5.39), the FE discrete trajectory is

$$\sum_{k=0}^{3}\left[(1+\frac{\lambda_k}{\tau}\Delta t)^{t/\Delta t}\left(w_k^T x(0)+\frac{0.5w_{k,3}}{\lambda_k}\right)-\frac{0.5w_{k,3}}{\lambda_k}\right]r_k, \tag{C.5}$$

where r_k is the eigenvector of the A matrix in (5.31). As shown in Appendix B.2 and section 5.2.5, the diagonalization of the A matrix in (5.31) is written as

$$AR=R\Lambda,\ A=R\Lambda R^{-1},\ \Lambda=R^{-1}AR,$$

where for $N=4$,

$$R=[r_0\cdots r_3]=\begin{bmatrix}\cos 0 & \cos 0 & \cos 0 & \cos 0\\ \cos\theta_0 & \cos\theta_1 & \cos\theta_2 & \cos\theta_3\\ \cos 2\theta_0 & \cos 2\theta_1 & \cos 2\theta_2 & \cos 2\theta_3\\ \cos 3\theta_0 & \cos 3\theta_1 & \cos 3\theta_2 & \cos 3\theta_3\end{bmatrix} \tag{C.6}$$

$$\Lambda=\mathbf{diag}(\lambda_0,\ldots,\lambda_3),\ R^{-1}=\begin{bmatrix}w_0^T\\ \vdots\\ w_3^T\end{bmatrix},\ \theta_0=\frac{\pi}{2N},\ \theta_1=\frac{3\pi}{2N},\theta_2=\frac{5\pi}{2N},\ \theta_3=\frac{7\pi}{2N}.$$

The goal of this section is to derive the inverse matrix of the R matrix in (C.6). Applying the similarity transformation

$$\hat{A}=D_1^{-1}AD_1,$$

with $D_1 = \mathbf{diag}(\sqrt{2}, 1, 1, 1)$, and $D_1^{-1} = \mathbf{diag}(1/\sqrt{2}, 1, 1, 1)$, we obtain a symmetric $\hat{A}$ as follows:

$$\begin{aligned}
\hat{A} &= D_1^{-1} A D_1 \\
&= \begin{bmatrix} 1/\sqrt{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & g & 0 & 0 \\ 0.5g & -1 & 0.5g & 0 \\ 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0.5g & -1 \end{bmatrix} D_1 \\
&= \begin{bmatrix} -1/\sqrt{2} & g/\sqrt{2} & 0 & 0 \\ 0.5g & -1 & 0.5g & 0 \\ 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0.5g & -1 \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} -1 & g/\sqrt{2} & 0 & 0 \\ 0.5g\sqrt{2} & -1 & 0.5g & 0 \\ 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0.5g & -1 \end{bmatrix} = \begin{bmatrix} -1 & g/\sqrt{2} & 0 & 0 \\ g/\sqrt{2} & -1 & 0.5g & 0 \\ 0 & 0.5g & -1 & 0.5g \\ 0 & 0 & 0.5g & -1 \end{bmatrix}. \qquad \text{(C.7)}
\end{aligned}$$

Hence, the diagonalization of the symmetric $\hat{A}$ is

$$\hat{A} = \hat{R} \Lambda \hat{R}^{-1}$$

where

$$\hat{R} = \begin{bmatrix} \hat{r}_0 & \hat{r}_1 & \hat{r}_2 & \hat{r}_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \cos\theta_0 & \cos\theta_1 & \cos\theta_2 & \cos\theta_3 \\ \cos 2\theta_0 & \cos 2\theta_1 & \cos 2\theta_2 & \cos 2\theta_3 \\ \cos 3\theta_0 & \cos 3\theta_1 & \cos 3\theta_2 & \cos 3\theta_3 \end{bmatrix} = D_1^{-1} R$$

$$\theta_0 = \frac{\pi}{2N}, \ \theta_1 = \frac{3\pi}{2N}, \theta_2 = \frac{5\pi}{2N}, \ \theta_3 = \frac{7\pi}{2N}. \qquad \text{(C.8)}$$

The orthogonal eigenvectors of $\hat{A}$ have equal length of $\sqrt{N/2} = \sqrt{2}$. From the orthogonality of $\hat{r}_k$, we have

$$D_c \hat{R}^T \hat{R} = D_c (D_1^{-1} R)^T (D_1^{-1} R) = D_c R^T (D_1^{-1})^T D_1^{-1} R = I, \qquad \text{(C.9)}$$

where the D_c compensate the length of $\hat{r}_k$ and is

$$D_c = \mathbf{diag}(\frac{2}{N}, \frac{2}{N}, \frac{2}{N}, \frac{2}{N}).$$

Finally, from (C.9) and $(D_1^{-1})^T = D_1^{-1}$, we obtain

$$\begin{aligned} R^{-1} &= D_c R^T (D_1^{-1})^T D_1^{-1} = D_c R^T D_1^{-1} D_1^{-1} \\ &= \begin{bmatrix} \frac{1}{4} & \frac{\cos\theta_0}{2} & \frac{\cos 2\theta_0}{2} & \frac{\cos 3\theta_0}{2} \\ \frac{1}{4} & \frac{\cos\theta_1}{2} & \frac{\cos 2\theta_1}{2} & \frac{\cos 3\theta_1}{2} \\ \frac{1}{4} & \frac{\cos\theta_2}{2} & \frac{\cos 2\theta_2}{2} & \frac{\cos 3\theta_2}{2} \\ \frac{1}{4} & \frac{\cos\theta_3}{2} & \frac{\cos 2\theta_3}{2} & \frac{\cos 3\theta_3}{2} \end{bmatrix} \end{aligned} \tag{C.10}$$

where $\theta_0 = \frac{\pi}{2N}$, $\theta_1 = \frac{3\pi}{2N}$, $\theta_2 = \frac{5\pi}{2N}$, $\theta_3 = \frac{7\pi}{2N}$.

For example, in Section 5.2.5, $N = 4$, the R and R^{-1} are:

$$R = [r_0 \cdots r_3] = \begin{bmatrix} \cos 0 & \cos 0 & \cos 0 & \cos 0 \\ \cos\theta_0 & \cos\theta_1 & \cos\theta_2 & \cos\theta_3 \\ \cos 2\theta_0 & \cos 2\theta_1 & \cos 2\theta_2 & \cos 2\theta_3 \\ \cos 3\theta_0 & \cos 3\theta_1 & \cos 3\theta_2 & \cos 3\theta_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0.92 & 0.38 & -0.38 & -0.92 \\ 0.70 & -0.70 & -0.70 & 0.70 \\ 0.38 & -0.92 & 0.92 & -0.38 \end{bmatrix}$$

$$R^{-1} = \begin{bmatrix} w_0^T \\ w_1^T \\ w_2^T \\ w_3^T \end{bmatrix} = \begin{bmatrix} \frac{1}{4} & \frac{\cos\theta_0}{2} & \frac{\cos 2\theta_0}{2} & \frac{\cos 3\theta_0}{2} \\ \frac{1}{4} & \frac{\cos\theta_1}{2} & \frac{\cos 2\theta_1}{2} & \frac{\cos 3\theta_1}{2} \\ \frac{1}{4} & \frac{\cos\theta_2}{2} & \frac{\cos 2\theta_2}{2} & \frac{\cos 3\theta_2}{2} \\ \frac{1}{4} & \frac{\cos\theta_3}{2} & \frac{\cos 2\theta_3}{2} & \frac{\cos 3\theta_3}{2} \end{bmatrix} = \begin{bmatrix} 0.25 & 0.46 & 0.35 & 0.19 \\ 0.25 & 0.19 & -0.35 & -0.46 \\ 0.25 & -0.19 & -0.35 & 0.46 \\ 0.25 & -0.46 & 0.35 & -0.19 \end{bmatrix} \tag{C.11}$$

Bibliography

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Pearson Prentice Hall, 2010.

[2] P. Arena, A. Basile, M. Bucolo, and L. Fortuna, "An object oriented segmentation on anlog CNN chip," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, pp. 837–846, 2003.

[3] J. Hawkins, *On Intelligence*. Henry Holt and Company, LLC, 2004, p. 66.

[4] L. O. Chua and L. Yang, "Cellular Neural Networks: Theory," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1257–1272, October 1988.

[5] K. R. Crounse and L. Chua, "Methods for image processing and pattern formation in cellular neural networks: A tutorial," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 42, no. 10, pp. 583–601, October 1995.

[6] L. O. Chua and T. Roska, "Stability of a class of nonreciprocal Cellular Neural Networks," *IEEE Transactions on Circuits and Systems*, vol. 37, pp. 1520–1527, December 1990.

[7] L. O. Chua, "Guest editorial," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 42, pp. 557–558, October 1995.

[8] C.-Y. Wu and S.-Y. Tsai, "Autonomous ratio-memory cellular nonlinear network (ARMCNN) for pattern learning and recognition," in *CNNA*, 2006, pp. 137–141.

[9] Y. Wu and C.-Y. Wu, "The design of CMOS non-self-feedback ratio memory for cellular neural network without elapsed operation for pattern learning and recognition," in *CNNA*, 2005, pp. 282–285.

[10] C.-H. Cheng and C.-Y. Wu, "The design of cellular neural network with ratio memory for pattern learning and recognition," in *CNNA*, 2000, pp. 301–307.

[11] C.-Y. Wu and C.-H. Cheng, "A learnable cellular neural network structure with ratio memory for image processing," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, pp. 1713–1723, Dec. 2002.

[12] C.-Y. Wu and J.-F. Lan, "CMOS current-mode neural associative memory design with on-chip learning," *IEEE Trans. Neural Netw.*, vol. 1, pp. 167–181, Jan. 1996.

[13] M. Gilli, M. Biey, and P. Checco, "Equilibrium analysis of cellular neural networks," *IEEE Transactions on Circuits and Systems I*, vol. 51, pp. 903–912, May 2004.

[14] S. Arik and V. Tavsanoglu, "Equilibrium analysis of non-symmetric cnns," *International Journal of Circuit Theory and Applications*, vol. 24, pp. 269–274, 1996.

[15] M. Hanggi, H. C. Reddy, and G. S. Moschytz, "The CNN sampling theorem," in *Euro. Conf. Circuit Theory Design*, 1999, pp. 995–998.

[16] K. Kayaer and V. Tavsanoglu, "A new approach to emulate CNN on FPGAs for real time video processing," in *International Workshop on Cellular Neural Networks and their Applications (CNNA)*, Spain, 2008.

[17] M. Hanggi, "On locally regular cellular neural networks," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, no. 5, pp. 513–520, May 2001.

[18] C. Mead, *Analog VLSI and Neural Systems*. Addison-Wesley, 1989, p. 287.

[19] L. O. Chua and L. Yang, "Cellular Neural Networks: Applications," *IEEE Transactions on Circuits and Systems*, vol. 35, pp. 1273–1290, October 1988.

[20] B. E. Shi and L. Chua, “Resistive grid image filtering: Input/output analysis via the cnn framework,” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 39, no. 7, pp. 531–548, July 1992.

[21] L. O. Chua, *CNN: A Paradigm for Complexity*. World Scientific Publishing, 1998.

[22] L. O. Chua and T. Roska, *Cellular neural networks and visual computing*. Cambridge University Press, 2002.

[23] A. Rodriguez-Vazquez, S. Espejo, R. Dominguez-Caetro, J. L. Huertas, and E. Sanchez-Sinencio, “Current-mode techniques for the implementation of continuous- and discrete-time cellular neural networks,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 40, no. 3, pp. 132–16, March 1993.

[24] M. Hanggi, H. C. Reddy, and G. S. Moschytz, “A sampling theorem in numerical integration,” in *ISCAS*, 1999.

[25] Software library for cellular wave computing engines. [Online]. Available: http://cnn-technology.itk.ppke.hu/

[26] L. O. Chua, *CNN: A Paradigm for Complexity*. World Scientific Publishing, 1998, p. 92.

[27] A. Zarandy, “The art of cnn template design,” *International Journal of Circuit Theory and Applications*, vol. 27, pp. 5–23, 1999.

[28] M. Itoh and L. Chua, “Star Cellular Neural Netowrks for Associative and Dynamic Memories,” *International Journal of Bifurcation and Chaos*, vol. 14, pp. 1725–1772, 2004.

[29] F. C. Hoppensteadt and E. M. Izhikevich, *Weakly Connected Neural Network*. Springer, 1997, p. 108.

[30] J. J. Hopfield, “Neurons with graded response have collective computational properties like those of two-state neurons,” in *Proc. Natl. Acad. Sci. USA*, vol. 8, 1984, pp. 3088–3092.

[31] W. Gerstner and W. Kistler, *Spiking Neuron Models*. New York: Cambridge UNiversity Press, 2002, pp. 356–357.

[32] W.-T. Chou, “The design of the autonomous ratio memory cellular nonlinear network without elapsed operation for pattern learning and recognition,” Master’s thesis, National Chiao-Tung University, Taiwan, Dec. 2007.

[33] D. Liu and A. Michel, “Sparsely Interconnected Neural Networks for Associative Memories With Applications to Cellular Neural Networks,” *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 41, pp. 295–307, Apr. 1994.

[34] M. Namba, “Estimating learner’s comprehension with cellular neural network for associative memory,” in *CNNA*, 2008, pp. 150–153.

[35] J. Park, H.-Y. Kim, Y. Park, and S.-W. Lee, “A synthesis procedure for associative memories based on space-varying cellular neural networks,” *Neural Networks*, vol. 14, pp. 107–113, Jan. 2001.

[36] R. Bise, N. Takahashi, and T. Nishi, “An improvement of the design method of cellular neural networks based on generalized eigenvalue minimization,” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, pp. 1569–1574, Dec. 2003.

[37] G. Grassi, “A new approach to design cellular neural networks for associative memories,” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 44, pp. 835–838, Sept. 1997.

[38] S.-C. Liu, J. Kramer, G. Indiveri, T. Delbruck, and R. Douglas, *Analog VLSI: circuits and principles*. The MIT press, 2002.

[39] S.-Y. Tsai, C.-H. Wang, and C.-Y. Wu, “Stability analysis of autonomous ratio-memory cellular nonlinear networks for pattern recognition,” *IEEE Transactions on Circuits and Systems I*, vol. 57, pp. 2156–2167, Aug. 2010.

[40] G. Cauwenberghs and M. A. Bayoumi, *Learning on Silicon: Adapative VLSI Neural Systems*. Kluwer Acadamic Publishers, 1999, p. 316.

[41] J.-F. Lan and C.-Y. Wu, "CMOS current-mode outstar neural networks with long period analog ratio memory," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 3, 1995, pp. 1676–1679.

[42] ——, "Analog CMOS current-mode implementation of the feedforward neural network with on-chip learning and storage," in *Proc. of 1995 IEEE International Conf. on Neural Networks*, vol. 1, 1995, pp. 645–650.

[43] C.-Y. Wu and J.-F. Lan, "A new neural associative memory with learning," in *IJCNN*, vol. 1, 1992, pp. 487–492.

[44] L. O. Chua and T. Roska, *Cellular neural network and visual computing*. Cambridge University Press, 2002.

[45] S. Sastry, *Nonlinear Systems*. NY: Springer-Verlag, 1999, pp. 185–187.

[46] H. K. Khalil, *Nonlinear Systems*, 3rd ed. NJ: Prentice Hall, 2002, pp. 112–113.

[47] S. Boyd. Basic Lyapunov theory. [Online]. Available: http://www.stanford.edu/class/ee363

[48] H. K. Khalil, *Nonlinear Systems*, 3rd ed. NJ: Prentice Hall, 2002, p. 122.

[49] R. Ziemer and W. Tranter, *Principles of Communication: systems, modulation and noise*, 5th ed. John Wiley and sons, 2002.

[50] K.-A. Wen, J.-Y. Su, and C.-Y. Lu, "VLSI design of digital cellular neural networks for image processing," *Journal of Visual Communication and Image Representation*, vol. 5, pp. 117–126, 1994.

[51] H. Harrer, A. Schuler, and E. Amelunxen, "Comparison of different numerical integraton methods for simulating cellular neural networks," in *CNNA*, 1990, pp. 151–159.

[52] F. Pozas-Flores, R. Carmona-Galan, and A. Rodriguez-Vazquez, “Simplified state update calculation for fast and accurate digital emulation of cnn dynamics,” in *International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*, Berkeley, USA, 2010.

[53] W. E. Boyce and R. C. Diprima, *Elementary Differential Equations and Boundary Value Problems*, 4th ed. John Wiley and Sons, 1986, p. 464.

[54] Z. Nagy and P. Szolgay, “Configurable Multilayer CNN-UM Emulator on FPGA,” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, pp. 774–778, June 2003.

[55] Z. Nagy, Z. Voroshazi, and P. Szolgay, “Emulated digital CNN-UM solution of partial differential equations,” *International Journal of Circuit Theory and Applications*, vol. 34, pp. 445–470, 2006.

[56] G. Strang, *Computational Science and Engineering*. Wellesley-Cambridge Press, 2007, p. 461.

[57] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes*, 3rd ed. Wellesley-Cambridge Press, 2007, p. 1035.

[58] S.-Y. Tsai, C.-H. Wang, and C.-Y. Wu, “On the diffusion model of autonomous ratio-memory cellular nonlinear network,” in *International Workshop on Cellular Nanoscale Networks and their Applications (CNNA)*, Berkeley, USA, 2010.

[59] G. Strang, “The discrete cosine transform,” *SIAM Review*, vol. 41, pp. 135–147, 1999.

[60] R. J. Leveque, *Finite Difference Methods for Ordinary and Partial Differential Equations*. Wellesley-Cambridge Press, 2007, p. 158.

[61] G. Strang, *Computational Science and Engineering*. Wellesley-Cambridge Press, 2007, pp. 464–465.

[62] R. J. Leveque, *Finite Difference Methods for Ordinary and Partial Differential Equations*. Wellesley-Cambridge Press, 2007, pp. 138–140.

[63] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, 2nd ed. Prentice Hall, 1999, pp. 591–593.

自傳

Su-Yung Tsai was born in Yi-Lan, Taiwan. He received the B.S. degree in control engineering and the M.S. degree in electronics engineering from National Chiao Tung University, HsinChu, Taiwan in 1993 and 1995, respectively. From 1995 to 1997, he served as an army officer for the obligatory military service in Taiwan. He is currently working towards the Ph.D. degree in National Chiao Tung University, Taiwan.

His current research interests include VLSI implementation of cellular nonlinear networks, neuromorphic VLSI that mimics the biological system, and CMOS image sensors with a pulsed laser or LED to obtain the 3D image.

住址: 國立交通大學工程四館 307 室

本論文使用 LaTeX[1] 系統排版.

[1] LaTeX 是 TeX 之下的 macros 集. TeX 是 American Mathematical Society 的註冊商標. 本論文 macros 的原始作者是 Dinesh Das, Department of Computer Sciences, The University of Texas at Austin. 交大中文版的作者是吳介琮, 交通大學電子工程學系, 新竹, 台灣.

Publication List

- ## Journal Paper

  - **S.-Y. Tsai**, C.-H. Wang, and C.-Y. Wu, “Stability Analysis of Autonomous Ratio-Memory Cellular Nonlinear Networks for Pattern Recognition,” *IEEE Transactions on Circuits and Systems I*, vol. 57, no. 8, pp. 2156–2167, Aug. 2010

- ## Conference Paper

  - C.-Y. Wu and **S.-Y. Tsai**, “ Autonomous Ratio-Memory Cellular Nonlinear Network (ARMCNN) for Pattern Learning and Recognition,” in *IEEE International Workshop on Cellular Neural Networks and their Applicaions (CNNA)*, Istanbul, Turkey, 2006, pp. 137–141.
  - **S.-Y. Tsai**, C.-H. Wang, and C.-Y. Wu, “On the Diffusion Model for Autonomous Ratio-Memory Cellular Nonlinear Network for Pattern Recognition,” in *IEEE International Workshop on Cellular Nanocale Networks and their Applicaions (CNNA)*, Berkeley, USA, 2010.
  - **S.-Y. Tsai**, W.-C. Yang, L.-J. Lin, C. -Y. Wu, and H. Chiueh, “The Development of 2-Dimensional Focal Plane CMOS 3D Imager,” in 96 年度國防科技學術合作計畫成果發表會論文集, 桃園龍潭, 臺灣, 中華民國九十六年十一月.