CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACANTENCO

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

SECCIÓN DE ELECTRÓNICA DEL ESTADO SÓLIDO

"EXPERIMENTACIÓN CON REDES NEURONALES PULSADAS: EVALUACIÓN DE CAPACIDADES COMPUTACIONALES"

TESIS QUE PRESENTA:

ING. GUILLERMO NIETO HERNÁNDEZ

PARA OBTENER EL GRADO DE:

MAESTRO EN CIENCIAS

EN LA ESPECIALIDAD DE:

INGENIERÍA ELÉCTRICA

DIRECTORES DE LA TESIS:

DR. FELIPE GÓMEZ CASTAÑEDA

DR. JOSÉ ANTONIO MORENO CADENAS

Ciudad de México

Agosto, 2018

AGRADECIMIENTOS

A mi familia por todo el apoyo y consuelo en los momentos difíciles. Agradezco su paciencia infinita, su motivación y sus palabras de aliento que fueron un aliciente cuando más lo necesité.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACYT) por la beca otorgada, que fue una de las piezas fundamentales para culminar este trabajo.

A mis asesores, el Dr. Felipe Gómez Castañeda y el Dr. José Antonio Moreno Cadenas por su invaluable dedicación, apoyo y enseñanza a lo largo de esta etapa.

Al M. en C. Luis Martín Flores Nava por tan admirable entrega a su profesión.

A mis revisores, el Dr. Mario Alfredo Reyes Barranca y el Dr. Ramón Peña Sierra por mostrar tanta disposición y por destinar su valioso tiempo para revisar este escrito.

DEDICATORIAS

A mi familia, que a pesar de las dificultades, siempre han demostrado ser un admirable ejemplo de perseverancia.

A mis profesores, por todo este tiempo dedicado en formar investigadores de gran nivel.

ÍNDICE

RESUMEN

El desarrollo de este proyecto plantea un método de análisis para una red neuronal pulsada usada en sistemas de procesamiento de información temporal, los cuales se basan en sistemas de reconocimiento de patrones dedicados a tareas propias de la inteligencia artificial. La motivación de este trabajo se origina del estudio de dos trabajos sobresalientes que estudian muy detalladamente las características computacionales de las redes neuronales pulsadas. El primer trabajo es un artículo escrito por *H. Herzog et al.* [14] que realiza una implementación de la una red neuronal pulsada para la clasificación de patrones monocromáticos horizontales y verticales. El segundo trabajo es una tesis de maestría elaborada por *M. Al-Shedivat* [17] que plantea el uso de memristores como elementos de memoria (sinapsis) en una red neuronal pulsada que presenta un comportamiento totalmente estocástico.

Las redes neuronales pulsadas son diseñadas para propósitos muy específicos, más concretamente, en ciertos procesos que requieran categorización o regresión; además al involucrar procesos de entrenamiento no supervisado, son adecuadas para obtener un modelo probabilístico generalizando la respuesta de la red. Esencialmente, estas redes son un paradigma de almacenamiento de la correlación que existe entre una respuesta predictiva (también llamada predicción probabilística) y un estímulo de entrada (*Gerstner 2002*) [8].

Con base en la información anterior, este trabajo tiene como objetivo presentar en un entorno de simulación computacional, las ventajas inherentes de las redes neuronales pulsadas, así como también los conceptos subyacentes en su desarrollo, con base en la codificación de los algoritmos en Matlab del modelo matemático de la *neurona pulsada*. Como parte complementaria de esto, se presenta una propuesta del modelo que muestra las distintas formas del *potencial de acción* tanto en neuronas de tipo excitatorias como de tipo inhibitorias. En conjunto con esto, se incluye una técnica de aprendizaje que

es completamente compatible con las neuronas pulsadas y que expone con precisión la evolución de las *sinapsis* a lo largo de todas las épocas de entrenamiento; esta técnica es conocida como *STDP*, de su nomenclatura en inglés: *Spike-Timing Dependent Plasticity.*

Como un elemento importante del proyecto, también se expone la arquitectura *WTA, (*de su nomenclatura en inglés: *Winner-Takes-All)*. Esta arquitectura WTA tiene cuatro neuronas, de las cuales dos son excitatorias y las otras dos son inhibitorias. La función del WTA es clasificar binariamente patrones monocromáticos posicionados horizontal o verticalmente en la red de entrada, definida como un arreglo ortogonal de un plano de neuronas.

Este trabajo tiene una componente tecnológica usando memristores, por lo que se desarrolló un ambiente de simulación codificado en Verilog-A y simulado a nivel componente través de Pyxis Layout de Mentor Graphics que incluye una introducción al uso de estos elementos eléctricos como memorias dinámicas entre las conexiones neuronas de la red; esto es, la utilización de memristores como sinapsis. Además, se muestran tres modelos matemáticos que caracterizan el comportamiento de la resistencia eléctrica de dichos elementos de manera estocástica, lo cual aproxima el comportamiento real de este tipo de redes neuronales pulsadas. Finalmente se presenta una aportación que consiste en una implementación con Simulink, la cual pretende en trabajos futuros, la traslación de la red neuronal pulsada de esta tesis hacia una realización en hardware basado en FPGA (Field-Programmable Gate Array).

ABSTRACT

The development of this project proposes a method of analysis for a spiking neural network for temporary information processing systems, based on pattern recognition systems dedicated to related tasks to artificial intelligence. The motivation of this work arises from the study of two outstanding works that study in detail the computational characteristics of the spiking neural networks.

The first work is an article [14] that performs an implementation of a spiking neural network for the classification of horizontal and vertical monochromatic patterns. The second work is a master degree thesis [17] that proposes the use of memristors as memory elements (synapses) in a spiking neural network that shows a totally stochastic behavior.

Spiking neural networks are designed for very specific purposes, i.e. in certain processes that require categorization or regression; in addition, by involving unsupervised training processes, they are appropriate to obtain a probabilistic model generalizing the response of the network. Essentially, these networks are a storage paradigm of the correlation that exists between a predictive response (also called probabilistic prediction) and an input stimulus [8].

Based on the above information, this work aims to present in a computational simulation environment, the inherent advantages of spiking neural networks, as well as the underlying concepts in their development, based on the coding of Matlab algorithms of the mathematical model of the *spiking neuron*. As a complementary part of this, a model proposal is presented that shows the different forms of the *action potential* in both *excitatory* and *inhibitory* type neurons. Related with this, a learning method is included that is completely compatible with the spiking neurons and accurately exposes the evolution of the synapses throughout all training periods; this method is known as *STDP*, from the english meaning: *Spike-Timing Dependent Plasticity*.

As an important element of the project, the *WTA* (from the english meaning: *Winner-Takes-All*) architecture is exposed. This *WTA* architecture has four neurons, of which two are excitatory and the other two are inhibitory. The function of the *WTA* is to binary classify monochromatic patterns placed horizontally or vertically in the input network, defined as an orthogonal arrangement of a plane of neurons.

This work has a technological component using memristors, a simulation environment coded in *Verilog-A* and simulated from the component level was developed using *Pyxis Layout* of *Mentor Graphics* including an introduction to

the use of these electrical elements as dynamic memories between the connections of the neurons in the network; that is, use memristors as synapses. In addition, three mathematical models are shown that stochastically characterize the behavior of the electrical resistance of these elements, which approximates the real behavior of this type of spiking neural networks.

Finally, a contribution that consists of an implementation with Simulink which intends in future work, the translation of the neural network pulsed from this thesis towards an implementation in hardware based on FPGA (Field-Programmable Gate Array).

INTRODUCCIÓN

La Inteligencia artificial es el campo técnico-científico de la informática que se relaciona con el diseño y estudio de programas y máquinas que pueden mostrar comportamientos considerados inteligentes. La inteligencia artificial ha demostrado ser una disciplina relacionada con diversos campos que presenta retos complejos, por eso se busca encontrar una relación que permita "enseñar" a una computadora a "pensar" como lo hacen algunos paradigmas que representan los diversos procesos llevados a cabo en el cerebro humano. Aunque el cerebro humano ha sido el objetivo principal de numerosos estudios, se entiende que los principios de procesamiento de información que utiliza el cerebro, son realmente complejos ya que la mayoría de ellos se generan, en esencia, a un nivel molecular. Por eso, una buena estrategia que se ha seguido por parte de la comunidad científica en la tarea de modelar el funcionamiento del cerebro, es considerando el papel que desempeña la neurona en forma colectiva. De esta forma, se puede observar que las neuronas conectadas en redes presentan una dinámica estocástica, lo cual implica escoger un análisis basado en variables aleatorias y con reglas de probabilidad condicional.

Este comportamiento estocástico representa una forma de "comunicación" que tienen las poblaciones de neuronas en el cerebro, enviando y recibiendo pulsos eléctricos o potenciales de acción, los cuales son generados con mayor o menor frecuencia de acuerdo a la intensidad del estímulo en forma de corriente. Con base a estos principios biofísicos, en este proyecto se usa un modelo matemático que aproxima dicho potencial de acción. El modelo es biorealista y puede ser codificado algorítmicamente en Matlab. Este modelo fue originalmente desarrollado por el matemático Eugene Izhikevich. Las redes neuronales pulsadas se ven como la próxima generación de redes neuronales. Estas redes se pueden emular de manera muy eficiente explotando la física de los dispositivos electrónicos. Además, en comparación con las redes neuronales artificiales convencionales, las redes pulsadas teóricamente

podrían generar un mayor poder computacional y podrían aprender y calcular de forma similar al cerebro biológico, lo que implica mayor velocidad de procesamiento y un entrenamiento no supervisado. A pesar de esto, las redes neuronales pulsadas son difíciles de modelar, ya que los algoritmos son aún muy complejos, lo que conlleva a un desarrollo más lento.

Las redes formadas por neuronas pulsadas producen patrones de actividad a través de las sinapsis que las interconectan, los cuales modulan la cantidad de corriente que pasa en ellas. Fisiológicamente, esto se localiza en las dendritas.

La sinapsis es un proceso electroquímico que define la razón de aprendizaje de la neurona, esto es, la intensidad a la que la neurona reacciona a un estímulo [8]. La fuerza, también llamado *peso sináptico*, con la que dos o más neuronas generan esta conexión sigue la regla de aprendizaje de *Hebb* o mejor conocida como *plasticidad de Hebb.* Esta regla define como una sinapsis es modificada y cómo cambia a lo largo del proceso de aprendizaje en el tiempo. Sabiendo esto, es necesario implementar una técnica que sea compatible en su totalidad con este comportamiento, es por eso que se optó por una técnica fácil de implementar computacionalmente y que permita visualizar la evolución de los pesos sinápticos a lo largo de cada época de entrenamiento. Este proceso es conocido como *Spike-Timing Dependent Plasticity o STDP.*

Una de las formas observadas en la que el cerebro procesa la información proveniente del exterior, está gobernada por un mecanismo en el cual algunas neuronas generan pulsos mientras que otras se mantienen inactivas. Este mecanismo se identifica como *El-Ganador-Toma-Todo o WTA* y es sumamente eficiente ya que está basado en la decisión perceptual, la cual determina qué estímulo de entrada tiene la mayor “importancia” y dicho estímulo es el que se “aprende” [24]. Este mecanismo se demuestra en este proyecto, usando una arquitectura con cuatro neuronas: dos de tipo excitatorio y dos de tipo inhibitorio, interconectadas lateralmente en parejas inhibitoria-

excitatoria y capaces de recibir varios estímulos simultáneos, para posteriormente generar una respuesta que indica el tipo de patrón (horizontal o vertical) ante el conjunto de estímulos de mayor peso.

El desarrollo del método computacional de aprendizaje/entrenamiento para la red neuronal del proyecto está limitado por la capacidad de procesamiento de la computadora, así como también por la cantidad de información que puede ser almacenada en la RAM. Estas limitantes son un factor fundamental para el desarrollo de estos procesos en sistemas FPGA, en los cuales las limitantes mencionadas no permiten usar recursos computacionales deliberadamente. Es por eso que se buscó generar algoritmos que procesen de manera eficiente la información, dicho de otra manera, se pretende que los algoritmos desarrollen un procesamiento biorealista pero óptimo en la demanda de recursos. Una implementación biorealista común de una técnica de aprendizaje neuronal y que ha demostrado generar resultados satisfactorios, incorpora el uso de dispositivos *memristores.* Un *memristor*, es un elemento pasivo de dos terminales, fabricado con materiales y métodos nanométricos el cual puede variar su resistencia eléctrica en función del voltaje o la corriente en sus terminales y conservar ese valor de resistencia incluso si ya se ha retirado dicho estímulo. Es por eso que los *memristores* se pretenden utilizar en proyectos futuros con redes neuronales pulsadas como elementos de memoria o lo que es equivalente, usarlos como sinapsis.

El memristor al ser un dispositivo tecnológicamente nuevo, es objeto de ambiciosos estudios de entre los cuales se tiene como objetivo desarrollar modelos matemáticos que representen su comportamiento real. Aunque actualmente no existe un modelo totalmente exacto, las aproximaciones existentes son bastante aceptables, lo que permite realizar simulaciones e implementaciones en sistemas computacionales; en este estudio se ha usado la plataforma de *Mentor Graphics* conocida como *Pyxis Layout* la cual ha permitido realizar simulaciones a nivel de circuito eléctrico a través de códigos descriptivos en *Verilog A* optimizados en módulos esquemáticos.

La mayoría de los modelos desarrollados para los memristores tienen un enfoque determinista, o sea que establecen valores nominales o promedios “esperados” para todas las variables y parámetros, sin embargo, tanto los memristores como la red tienen un comportamiento probabilístico [17]; para este trabajo se incluyen las modificaciones necesarias a algunos modelos deterministas para que presenten propiedades estocásticas. Para este fin, se hace una introducción al desarrollo de tres modelos matemáticos deterministas y como extensión, se presentan a los mismos, pero con comportamiento estocástico.

1 ELECTROFISIOLOGÍA DE LAS NEURONAS

1.1 Introducción

La investigación biológica ha acumulado una enorme cantidad de conocimiento detallado acerca de la estructura y funcionamiento del cerebro. Estas investigaciones han tomado nuevos rumbos y han centrado su atención en áreas específicas, mucho más detalladas, como es el caso del modelado computacional el cual tiene como objetivo principal, desarrollar algoritmos que permitan emular artificialmente y posteriormente imitar el comportamiento del cerebro. Dicho esto, la intención del presente capítulo es mostrar los conceptos básicos para entender la respuesta de una neurona a ciertos estímulos.

1.2 Neurona pulsada

Cuando se habla del cerebro humano, se da inicio a un valioso debate acerca de su funcionamiento y la forma en cómo "aprende" bajo ciertas circunstancias. Sin embargo, todos los puntos de vista de dicho debate concuerdan con una idea común: la neurona es el concepto más importante en la ciencia del cerebro.

En el cerebro humano existen alrededor de 10^{11} neuronas, muchas menos que la cantidad de células no neurales tales como la glía [5]. Sin embargo, las neuronas son únicas, en el sentido de que pueden transmitir señales eléctricas a lo largo de grandes distancias. Con todo esto se puede trabajar en distintos niveles de complejidad, desde un circuito neuronal hasta estructuras corticales e incluso hasta el cerebro completo, lo que finalmente nos lleva a entender el comportamiento del organismo.

Una neurona típica recibe estímulos de más de 10,000 neuronas a través de varios kilómetros de "cables" por centímetro cúbico [8]. Esta neurona puede ser dividida en tres distintas partes funcionales, llamadas dendritas, soma y axón, ver Fig. 1.1. En general, las dendritas juegan el papel de "dispositivos de entrada" que recogen las señales de otras neuronas y las transmiten al

soma. Los pulsos, también llamados potenciales de acción o potenciales de membrana, tienen una amplitud de alrededor 100 mV y una duración típica de 1-2 ms.

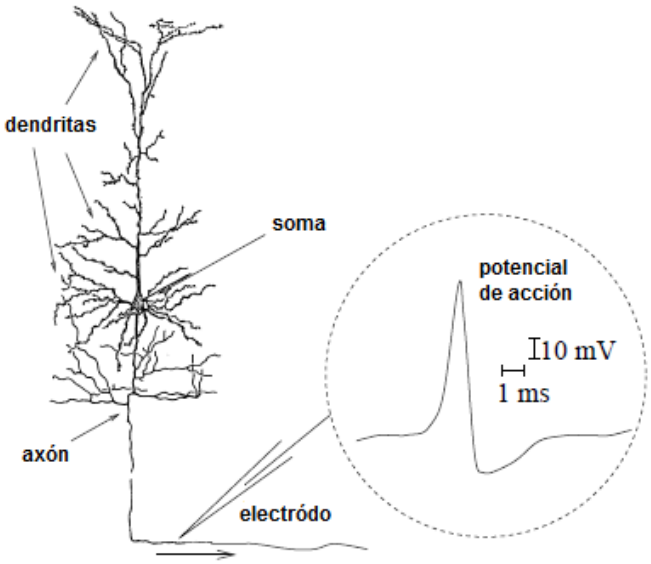


Fig. 1.1.Neurona simple. Dendritas, soma y axón se distinguen claramente.

La forma de los pulsos no cambia como el potencial de acción se propaga a través del axón. Una cadena de potenciales de acción emitidos por una sola neurona es llamada tren de pulsos, la cual es una secuencia de eventos repetidos que ocurren a intervalos regulares o irregulares. Dado que todos los pulsos de una neurona determinada se ven iguales, la forma del potencial de acción no transporta ninguna información. Más bien, es la cantidad y la frecuencia de los pulsos lo que importa. Es por esto que el potencial de acción es el medio principal de comunicación entre neuronas.

En general, las neuronas no disparan por si mismas; ellas disparan como resultado de los pulsos provenientes de otras neuronas. Lo que nos lleva a cuestionarnos una de las más fundamentales preguntas en neurociencia: *¿qué exactamente hace que una neurona dispare?* Por otra parte, ¿por qué pueden dos neuronas tener diferentes respuestas a exactamente la misma entrada e idénticas respuestas a entradas completamente diferentes? O ¿qué hay en los pulsos entrantes que provoca una respuesta en una neurona pero no en otra?

Para contestar estas preguntas, es necesario entender primero la dinámica del mecanismo de generación de pulsos en las neuronas.

La mayoría de los libros introductorios de neurociencias describen a las neuronas como integradores con umbral: la neurona suma los pulsos provenientes de otras neuronas y “comparan” los pulsos ya integrados con un cierto valor de voltaje, llamado umbral de disparo. Si el voltaje del pulso resultante está por debajo del umbral, la neurona permanece inactiva; cuando el voltaje del pulso está por encima del umbral, la neurona dispara un pulso “todo o nada” como se muestra en la Fig. 1.2, y posteriormente restablece su potencial de membrana. Para agregar plausibilidad teórica a dicho argumento, se puede revisar el modelo de Hodgkin-Huxley de la generación de pulsos en los axones gigantes de calamares [18]. Lo interesante del modelo de Hodgkin-Huxley es que no tiene un umbral bien definido, no dispara pulsos “todo o nada” y no es un integrador sino un resonador (es decir, las entradas tienen ciertas frecuencias que resuenan con la frecuencia de las oscilaciones de sub-umbral de la neurona). Más adelante se revisará con mayor detalle el modelo de Hodgkin-Huxley.

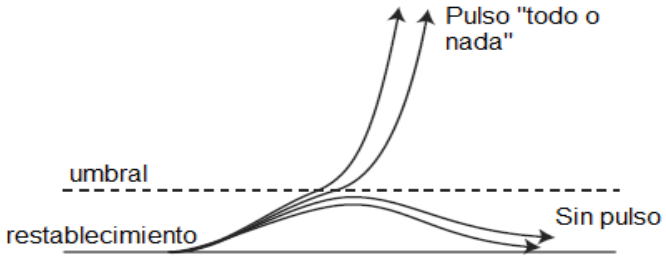


Fig. 1.2. Concepto de umbral de disparo.

1.3 Sinapsis

El sitio donde el axón de una neurona presináptica hace contacto con una dendrita (o soma) de una neurona postsináptica es la sinapsis. El tipo más común de sinapsis en el cerebro vertebrado es la sinapsis química. En la sinapsis química, el axón está colocado muy cerca de la neurona

postsináptica, dejando solo una pequeña separación entre la membrana presináptica y la membrana postsináptica, llamada hendidura sináptica. Cuando el potencial de acción llega a la sinapsis, se dispara una cadena compleja de procesos bioquímicos, que conducen a la liberación de neurotransmisores desde la terminal presináptica hacía la hendidura sináptica. Al instante en que las moléculas del neurotransmisor han alcanzado el extremo postsináptico, son detectadas por receptores químicos localizados en la membrana postsináptica y abren canales específicos para que los iones del fluido extracelular fluyan dentro de la célula. La respuesta de una neurona postsináptica generada por un potencial de acción proveniente de una neurona presináptica se le conoce como potencial postsináptico (o *PSP* por sus siglas en inglés *Postsynaptic Potential*).

1.4 Introducción a la dinámica neuronal

El modelo matemático que se toma como base para desarrollar el modelo de Izhikevich para una neurona pulsada, es el modelo de Hodgkin-Huxley. En general, dicho modelo describe la dinámica neuronal en términos la activación o inactivación de conductancias controladas por voltaje, lo que nos lleva a resaltar un resultado importante y es que las neuronas se comportan como sistemas dinámicos, por lo que es apropiado estudiarlas como tales.

Un sistema dinámico consiste en un conjunto de variables que describen su estado (del sistema) y una regla que describe la evolución de las variables de estado a lo largo del tiempo; esto es, la dependencia del estado del sistema en un momento posterior tras la interacción con la entrada, y de su estado (del sistema) en un momento anterior. El modelo de Hodgkin-Huxley es un sistema dinámico tetra-dimensional porque su estado se determina únicamente con el potencial de membrana, V, y las denominadas variables de activación n, m, y h para las corrientes persistentes y transitorias de los iones K^+y Na^+. La regla de evolución de estas variables, está dada por un sistema de cuatro dimensiones de ecuaciones diferenciales ordinarias.

Típicamente, todas las variables que están involucradas en describir la dinámica neuronal, pueden ser clasificadas de acuerdo a cuatro clases, con base en su función y su escala de tiempo (duración), así se definen:

1. **Potencial de membrana.**
2. **Variables de excitación:** como la activación de la corriente de los iones Na^+. Se considera que las variables de excitación son las responsables de la carrera ascendente de un pulso (*spike).*
3. **Variables de recuperación:** como la inactivación de la corriente de los iones Na^+y la activación de una corriente rápida de K^+. Estas variables son las responsables de la re-polarización (carrera descendente) de un pulso *(spike).*
4. **Variables de adaptación:** hace referencia a la activación de voltajes lentos o las corrientes de pendientes de los iones de Ca^{2+}. Las variables de adaptación presentan un fenómeno de acumulación durante pulsaciones prolongadas y pueden afectar la excitabilidad a largo plazo.

Cabe mencionar que el modelo de Hodgkin-Huxley no incorpora a las variables de adaptación, ya que no muestra una dinámica de descarga rápida *(bursting)*

Adicionalmente a esto, se requiere el análisis de dos conceptos que son fundamentales para entender el por qué algunas neuronas tienen umbrales bien definidos, pulsos “todo o nada”, pulsos post-inhibitorios, preferencias de frecuencia, histéresis, etc., y otras neuronas que no los tienen. Estos conceptos son los retratos de fase y las bifurcaciones.

1.4.1 Retratos de fase

Se entiende por retrato de fase a una representación geométrica de las trayectorias de un sistema dinámico en un plano de fase. Estos retratos deben incluir la trayectoria del sistema a través de flechas, los estados estables a través de puntos y los estados inestables a través de círculos. Además los ejes deben ser las variables de estado. (Ver Apéndice A para más información)

Como inicio, consideremos a una neurona inactiva, cuyo potencial de membrana está en reposo. Desde el punto de vista de un sistema dinámico, no hay algún cambio en las variables de estado de dicha neurona, por lo tanto se puede decir que existe un punto de equilibrio. Todas las corrientes de entrada que despolarizan a la neurona están balanceadas o equilibradas por las corrientes de salida que hiperpolarizan a esa neurona. Si la neurona continúa inactiva a pesar de esas pequeñas perturbaciones y ruidos en la membrana, como se muestra en la fig.1.3a (arriba), se puede concluir que el equilibrio es estable. Por ejemplo, el estado del modelo $I_{Na,p} + I_{K^-}$ está descrito por el potencial de membrana V, y la variable de activación n, de la corriente persistente de los iones K^+. Por lo tanto, su representación se puede generar usando un vector de dos dimensiones (V, n). La activación instantánea de la corriente de los iones Na^+ es una función de V, por lo tanto no se considera como una variable más. La forma como el modelo evoluciona se puede representar usando una trayectoria $\big(V(t), n(t)\big)$ en un plano $V \times n$; dependiendo del punto inicial, el sistema puede presentar varias trayectorias, tal como se muestra en la fig.1.3a (abajo). Todas las trayectorias en la figura mostrada, son llevadas al equilibrio estable, denotado por un punto negro llamado *atractor.*

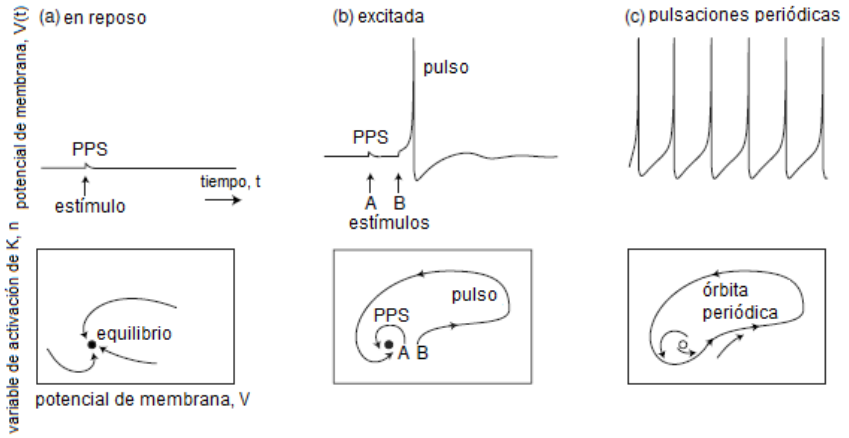


Fig. 1.3. Actividades de una neurona pulsada correspondientes a un equilibrio estable (a y b) o a un ciclo límite (c).

Una propiedad fundamental de las neuronas es la *excitabilidad,* ilustrada en la fig.1.3b. En este estado, la neurona se encuentra en reposo, esto es, su retrato de fase tiene un equilibrio estable. Cuando se presentan pequeñas perturbaciones como A, el resultado es una pequeña excursión desde el punto de equilibrio, denotada como PPS (potencial postsináptico). Cuando se presenta una perturbación grande, como B, ésta es amplificada por la dinámica intrínseca de la neurona y da como resultado una respuesta de una sola pulsación.

Si posteriormente se inyecta a la neurona una corriente lo suficientemente grande, ésta entra a un modo marcapasos, o lo que es lo mismo, muestra un comportamiento de pulsaciones periódicas, como en la fig.1.3c. Desde un punto de vista de los sistemas dinámicos, esta neurona presenta un estado de ciclo límite estable, también conocido como órbita periódica estable. Los detalles electro-físicos de la neurona (por ejemplo, el número y el tipo de sus corrientes, su cinética, etc.) determinan solo la localización, la forma, y el período del ciclo límite. Mientras exista el ciclo límite, la neurona presentará una actividad de pulsaciones periódicas. Es necesario aclarar que el equilibrio y los ciclos límite pueden coexistir, por lo tanto, una neurona puede cambiar de un estado a otro solamente con una entrada transitoria.

1.5 El modelo de Hodgkin-Huxley

Hodgkin y Huxley desarrollaron en el año 1952, algunos experimentos con los axones gigantes de calamares [18] y encontraron tres diferentes tipos de corrientes iónicas, la corriente de sodio, la de potasio y la corriente de fuga que es generada principalmente por los iones de Cl^-. Canales iónicos dependientes del voltaje, uno para sodio y otro para potasio, controlan el flujo de esos iones a través de la membrana celular. La corriente de fuga se encarga de otros tipos de canales los cuales no se describen explícitamente.

Este modelo puede ser más comprensible si se observa la fig.1.4. La membrana celular semi-permeable divide al interior de la célula del fluido extracelular que se encuentra en el exterior y también actúa como un capacitor [8]. Si en un momento dado se inyecta a la célula una corriente $I(t)$, ésta puede agregar carga al capacitor o puede fugarse a través de los canales en la membrana celular. Debido al transporte activo de iones a través de la membrana celular, la concentración de iones dentro de la célula es diferente al del fluido extracelular en el exterior. El potencial de Nernst generado por la diferencia de concentraciones de los iones está representado en la fig.1.4 por la batería.

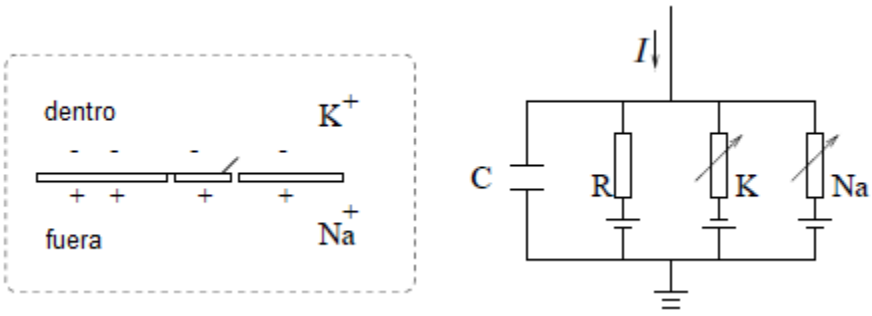


Fig. 1.4. Diagrama esquemático para el modelo de Hodgkin-Huxley.

La consideración anterior se puede exponer en términos de una ecuación matemática. La conservación de la carga eléctrica en una porción de la membrana implica que la corriente aplicada $I(t)$ se divide en una corriente capacitiva I_C que carga al capacitor C y una componente de corriente I_K que fluye a través de los canales iónicos, así:

$$I(t) = I_C(t) + \sum_k I_k(t) \tag{1.1}$$

donde la sumatoria se realiza con todos los canales iónicos. En el modelo estándar de Hodgkin-Huxley solo hay tres tipos de canales: un canal de sodio denotado como Na, uno de potasio denotado como K y un canal sin especificar de fuga con resistencia R. De acuerdo a la definición de capacitancia

$C = Q/u$, donde Q es la carga y u el voltaje en el capacitor, se puede expresar la corriente de carga como $I_C = C\ du/dt$. Por lo tanto de (1.1):

$$C\frac{du}{dt} = -\sum_k I_k(t) + I(t) \tag{1.2}$$

Presentando todo en términos biológicos se tiene que u es el voltaje en la membrana y $\sum_k I_k$ es la suma de las corrientes iónicas que fluyen a través de la membrana celular. Todos los canales iónicos se caracterizan de acuerdo a su resistencia o, de forma equivalente, a su conductancia. En el caso del canal de fuga, la conductancia es dependiente del voltaje y se denota como $g_L = 1/R$. Por otro lado, las conductancias de los otros canales iónicos, son dependientes del voltaje y del tiempo. Si los canales se encuentran totalmente abiertos entonces la corriente fluye a través de ellos con una máxima conductancia g_{Na} o g_K, respectivamente. Comúnmente, algunos de los canales se encuentran bloqueados por lo tanto, debido a la naturaleza del sistema, es necesario considerar la probabilidad de que dichos canales se encuentren abiertos. Esto se hace a través de tres variables adicionales m, n y h. Los canales Na^+ son controlados por un acción conjunta de m y h mientras que los canales K^+ son controlados por n. Con base en esto, Hodgkin y Huxley formularon una expresión que incorpora la acción de las tres variables anteriores. La expresión es la siguiente:

$$\sum_k I_k(t) = g_{Na}m^3h(u - E_{Na}) + g_K n^4(u - E_K) + g_L(u - E_L) \tag{1.3}$$

Los parámetros E_{Na}, E_K y E_L son denominados potenciales de inversión. Tanto los potenciales de inversión como las conductancias, son parámetros experimentales. La tabla 1.1 muestra los valores originales reportados por Hodgkin y Huxley. Estos valores están basados en una escala de voltaje en la cual el potencial de reposo es cero. Sin embargo, para que estos valores sean aceptados en las investigaciones actuales, deben ser corregidos a través de un factor de offset el cual es de -65 mV. Por ejemplo, el valor correcto del potencial de inversión del sodio es $E_{Na} = 50\,mV$ y el de potasio es

$E_K = -77\,mV$. Las tres variables antes mencionas, m, n y h son llamadas variables de control y cambian de acuerdo a las siguientes ecuaciones diferenciales:

$$\begin{aligned} \dot{m} &= \alpha_m(u)(1-m) - \beta_m(u)\,m \\ \dot{n} &= \alpha_n(u)(1-n) - \beta_n(u)\,n \\ \dot{h} &= \alpha_h(u)(1-h) - \beta_h(u)\,h \end{aligned} \tag{1.4}$$

sabiendo que $\dot{m} = dm/dt$ y así con las demás variables. En (1.4) se pueden observar varias funciones nuevas α y β, las cuales se proporcionan también en la tabla 1.1 y son funciones empíricas de u y se modificaron de acuerdo a los requerimientos de Hodgkin y Huxley para que se ajusten a los datos obtenidos del axón gigante del calamar.

$\boldsymbol{x}$	$\boldsymbol{E_x}$	$\boldsymbol{g_x}$
Na	$115\,mV$	$120\,mS/cm^2$
K	$-12\,mV$	$36\,mS/cm^2$
L	$10.6\,mV$	$0.3\,mS/cm^2$

$\boldsymbol{x}$	$\boldsymbol{\alpha_x(u/mV)}$	$\boldsymbol{\beta_x(u/mV)}$
n	$(0.1-0.01u)/[\exp(1-0.1u)-1]$	$0.125\exp(-u/80)$
m	$(2.5-0.1u)/[\exp(2.5-0.1u)-1]$	$4\exp(-u/18)$
h	$0.07\exp(-u/20)$	$1/[\exp(3-0.1u)+1]$

Tabla 1.1. Parámetros usados en las ecuaciones de Hodgkin-Huxley. La capacitancia de la membrana es $C = 1\ \mu F/cm^2$. La escala de voltaje se desplazó hasta que el potencial de reposo desapareció.

Para entender mejor las tres ecuaciones de (1.4) es conveniente rescribirlas de la siguiente forma:

$$\dot{x} = -\frac{1}{\tau_x(u)}[x - x_0(u)] \tag{1.5}$$

donde x representa m, n o h. Para un cierto valor de u, la variable x se aproxima al valor $x_0(u)$ con una constante de tiempo de $\tau_x(u)$. El valor asintótico $x_0(u)$ y la constante $\tau_x(u)$ se generan a partir de la transformación $x_0(u) = \alpha_x(u)/[\alpha_x(u) + \beta_x(u)]$ y $\tau_x(u) = [\alpha_x(u) + \beta_x(u)]^{-1}$. Usando los parámetros de la tabla 1.1 se pueden obtener las gráficas de las funciones $x_0(u)$ y $\tau_x(u)$. Estas gráficas se pueden observar en la fig.1.5.

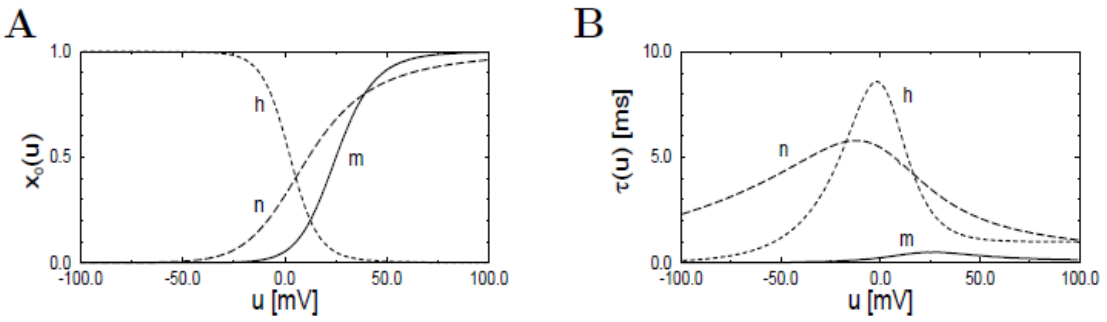


Fig. 1.5. Función de equilibrio (A) y constante de tiempo (B) para las tres variables m, n, h en el modelo de Hodgkin-Huxley. El potencial de reposo está a un valor de $u = 0$.

1.5.1 Bifurcaciones

Una bifurcación es un cambio en los puntos de equilibrio u órbitas periódicas, o en sus propiedades de estabilidad, como consecuencia de la variación de un parámetro [22]. (Ver Apéndice A para mayor información)

Se puede suponer el caso en el que la magnitud de la corriente inyectada a la neurona es un parámetro que se puede controlar, como por ejemplo una rampa de corriente o un escalón. La neurona está en reposo antes de encontrarse con dicho estímulo, entonces el retrato de fase muestra un equilibrio estable como el de la fig.1.6a (superior) o el de la fig.1.6a (inferior). Posteriormente la neurona comienza a disparar pulsos acentuados, entonces su retrato de fase tiene un atractor de ciclo límite y puede parecerse al de la fig.1.6.b (superior).

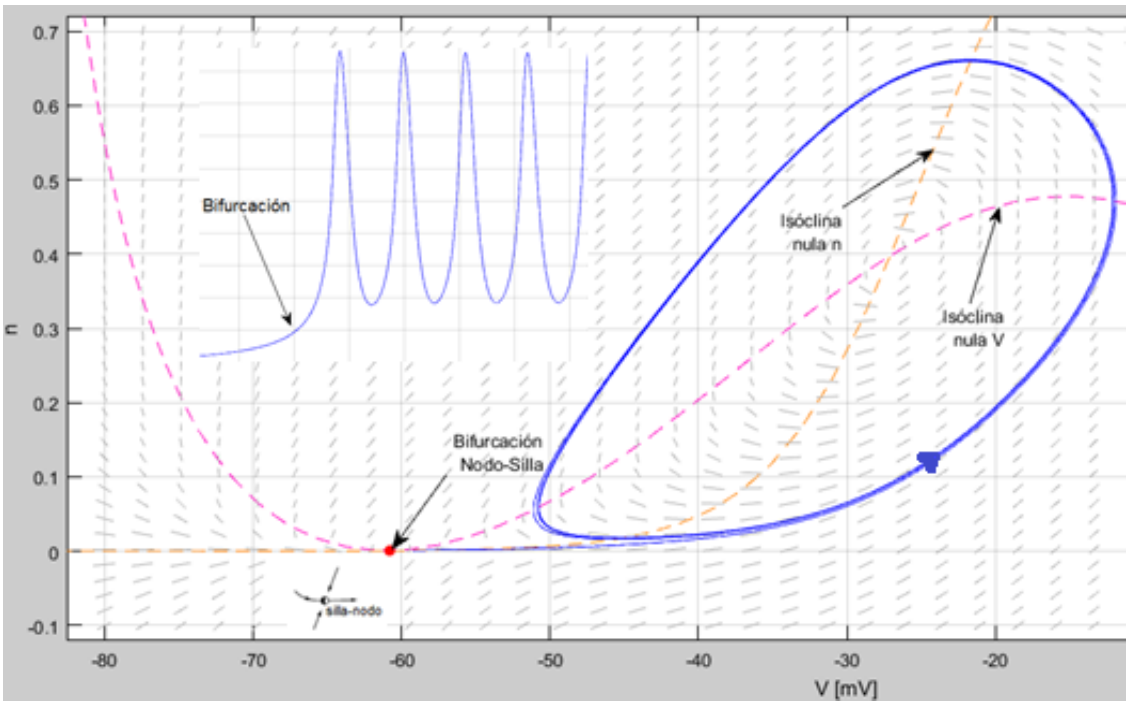
Tras estudiar detenidamente los retratos de fase de la fig.1.6a (inferior) y la fig.1.6b (superior), aparentemente existen algunos niveles intermedios de la corriente inyectada que corresponden a la transición desde el reposo hasta un pulso ininterrumpido. Estas transiciones, desde un punto de vista de sistemas dinámicos, corresponden a una bifurcación de la dinámica neuronal, esto es, a un cambio cualitativo en un retrato de fase del sistema.

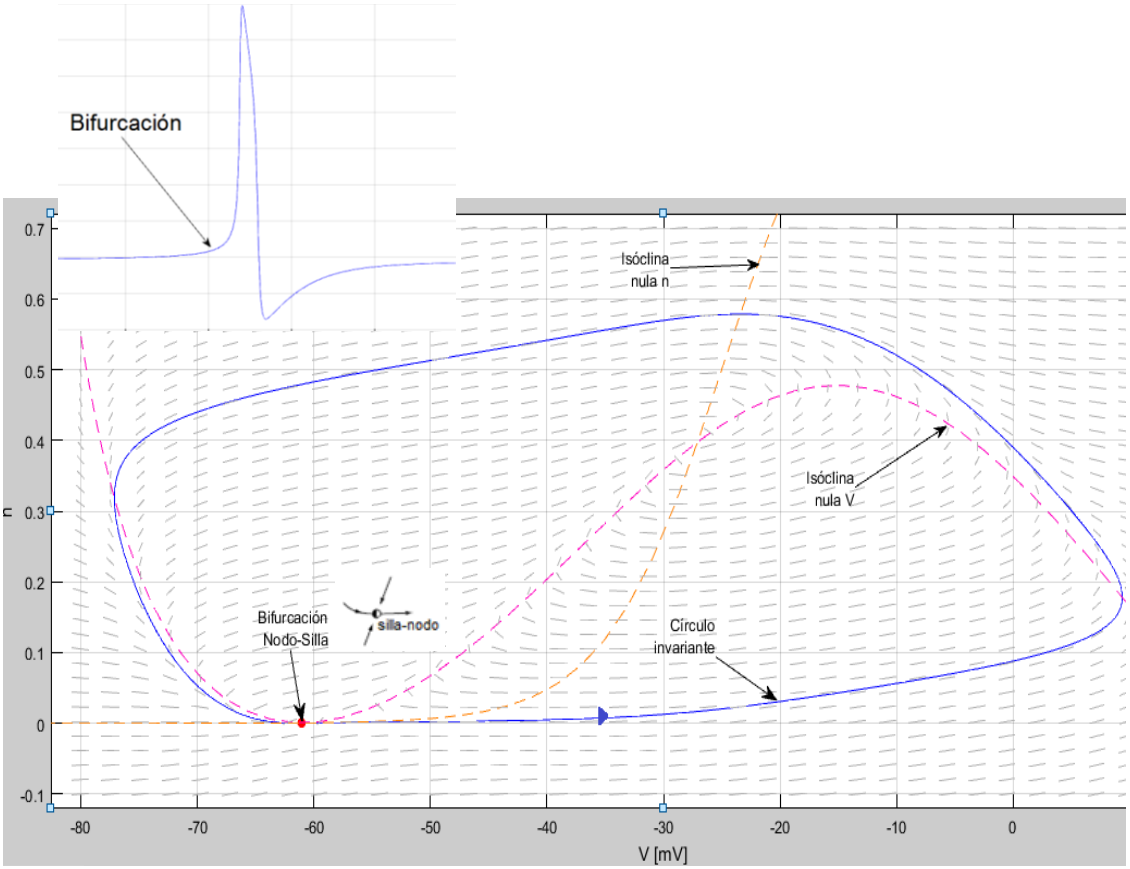
Esto nos da una idea más clara sobre cuándo una neurona es excitable y cuando no. Una neurona es excitable porque está cerca de una bifurcación

desde el reposo hasta el momento del disparo, por lo que el tipo de bifurcación determina las propiedades de excitabilidad de la neurona. Es muy importante aclarar que las bifurcaciones son determinadas por la electrofisiología de la neurona.

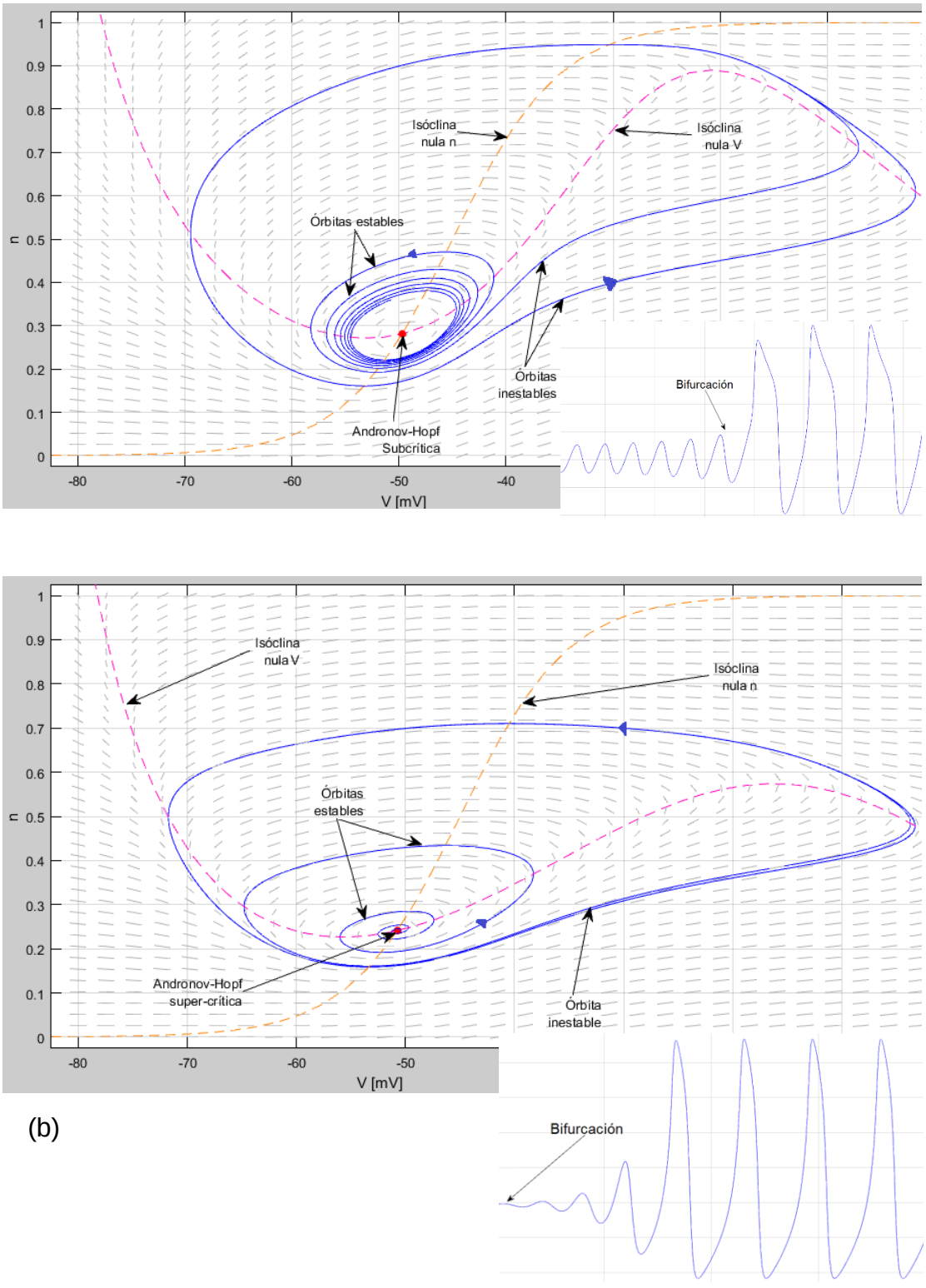
Aunque se ha observado que puede haber millones de diferentes mecanismos electrofisiológicos de excitabilidad, sólo existen cuatro tipos de bifurcaciones de equilibrio que el sistema puede experimentar sin ninguna constricción adicional, como la simetría.

Estas cuatro bifurcaciones son mostradas en la figura 1.6. Los trazos superiores (a) son los dos tipos de bifurcaciones que se presentan en un sistema dinámico neuronal. Así mismo, (b) muestra las dos bifurcaciones correspondientes del tipo Andronov-Hopf.


Bifurcación
Isóclina
nula n
Isóclina
nula V
Bifurcación
Nodo-Silla
silla-nodo
n
V [mV]


Bifurcación
Isóclina
nula n
Isóclina
nula V
Bifurcación
Nodo-Silla
silla-nodo
Círculo
invariante
n
V [mV]

(a)



(b)

Fig. 1.6. (a) Los dos tipos de bifurcaciones silla-nodo que se presentan en un sistema dinámico neuronal. (b) Los dos tipos de bifurcaciones Andronov-Hopf.

- **Bifurcación silla-nodo.** Como la magnitud de la corriente inyectada o cualquier otro parámetro de la bifurcación cambia, un estado estable correspondiente al estado de reposo es abordado por un equilibrio inestable, entonces se unen y se neutralizan (círculo rojo). Después de esto el estado de reposo ya no existe, entonces la trayectoria que describe la evolución del sistema experimenta una transición hacía el atractor de ciclo límite, lo que indica que la neurona comienza a disparar pulsos acentuados. Nótese que el ciclo límite, o cualquier otro atractor, debe coexistir con el estado de reposo para que la transición reposo-pulso ocurra (fig.1.6a superior).

- **Bifurcación silla-nodo en un círculo invariante.** Es similar a la bifurcación anterior, excepto que ahora hay un círculo invariante al momento de la bifurcación, el cual se convierte en el atractor de ciclo límite (fig.1.6a inferior)

- **Bifurcación subcrítica Andronov-Hopf.** En esta bifurcación un ciclo límite inestable pequeño se contrae a un equilibrio estable lo que genera que pierda estabilidad como se muestra en la fig.1.6b (superior). Debido a estas inestabilidades, la trayectoria diverge del equilibrio y se acerca a un ciclo límite de pulsación de gran amplitud o a algún otro atractor.

- **Bifurcación super-crítica Andronov-Hopf.** Aquí el equilibrio estable pierde estabilidad y da lugar a la creación de un atractor de ciclo límite de pequeña amplitud, como en la fig.1.6b (inferior). Debido a que la magnitud de la corriente inyectada se incrementa, la amplitud del ciclo límite también se incrementa y se transforma en un ciclo límite de pulsación de escala completa.

Es importante notar que el estado de reposo y el estado de pulsación coexisten en los casos de las bifurcaciones silla-nodo y la subcrítica Andronov-Hopf, pero no en las dos restantes. Tal coexistencia se revela a través de un

comportamiento de histéresis cuando la corriente que se inyecta a la neurona incrementa de manera lenta y posteriormente disminuye más allá del valor de bifurcación. La histéresis existe debido a que la transición reposo-pulsación y la transición pulsación-reposo suceden a diferentes valores de corriente.

Los sistemas sometidos a las bifurcaciones Andronov-Hopf, ya sean subcríticas o super-críticas, muestran ciertas oscilaciones amortiguadas del potencial de membrana, mientras que los sistemas cerca de las bifurcaciones silla-nodo, ya sea que estén dentro o no de un círculo invariante, no muestran esas oscilaciones. La existencia de oscilaciones de pequeña amplitud propias de un sistema amortiguado, originan la posibilidad de una resonancia a la frecuencia de los pulsos entrantes.

Con base en este comportamiento, se puede realizar una clasificación más completa de las neuronas de acuerdo a las bifurcaciones del estado estable. Las neuronas que presentan oscilaciones de sub-umbral amortiguadas se conocen como resonadoras y aquellas que no tienen esta propiedad, se les conoce como integradoras. Ahora, si las neuronas muestran una coexistencia entre los estados de reposo y pulsación, por lo menos cerca de la transición de reposo a pulsación, se les denomina biestables, y aquellas que presentan el comportamiento opuesto, se les denomina monoestables. De una manera más práctica, esta clasificación se puede ver en la fig. 1.7.

Coexistencia de los estados de reposo y pulsación

Oscilaciones de sub-umbral	Sí (biestable)	No (monoestable)
No (Integradora)	silla-nodo	silla-nodo en un círculo invariante
Sí (Resonadora)	Andronov-Hopf subcrítica	Andronov-Hopf super-crítica

Fig. 1.7. Clasificación de las neuronas de acuerdo a las bifurcaciones del estado de reposo.

1.6 Reducción del modelo de Hodgkin-Huxley al modelo simple de Izhikevich

Como ya se ha mencionado, el modelo de Hodgkin-Huxley es un modelo tetra-dimensional y biofísicamente preciso, lo que lo hace computacionalmente prohibitivo debido a que solo se puede simular una pequeña porción de células en tiempo real. Por otra parte, el uso de un modelo de integración y disparo permite visualizar resultados computacionalmente efectivos; sin embargo, el modelo es muy irreal y simple, incapaz de producir una dinámica rica en descargas rápidas como exhiben las neuronas corticales. Es por esta razón que el matemático Eugene M. Izhikevich propone una simplificación del modelo de Hodgkin-Huxley a través de una metodología de bifurcaciones la cual reduce al modelo a dos dimensiones, lo que lo hace más adecuado para realizar simulaciones a gran escala.

Primeramente es necesario considerar la expresión del modelo de Hodgkin-Huxley usando la ecuación (1.2) y sustituyendo la ecuación (1.3).

$$C\frac{du}{dt} = -g_{Na}m^3h(u - E_{Na}) - g_K n^4(u - E_K) - g_L(u - E_L) + I(t) \qquad (1.6)$$

Para comenzar la simplificación se debe incluir la relación observada en la simulación computacional de [20]. Esta relación involucra directamente a n y h:

$$n(t) + h(t) \approx 0.84 \qquad (1.7)$$

Como se muestra en la fig.1.8. De hecho, al graficar las variables en un plano (n, h), como en la fig.1.9, se puede observar que la relación entre n y h es casi una constante.

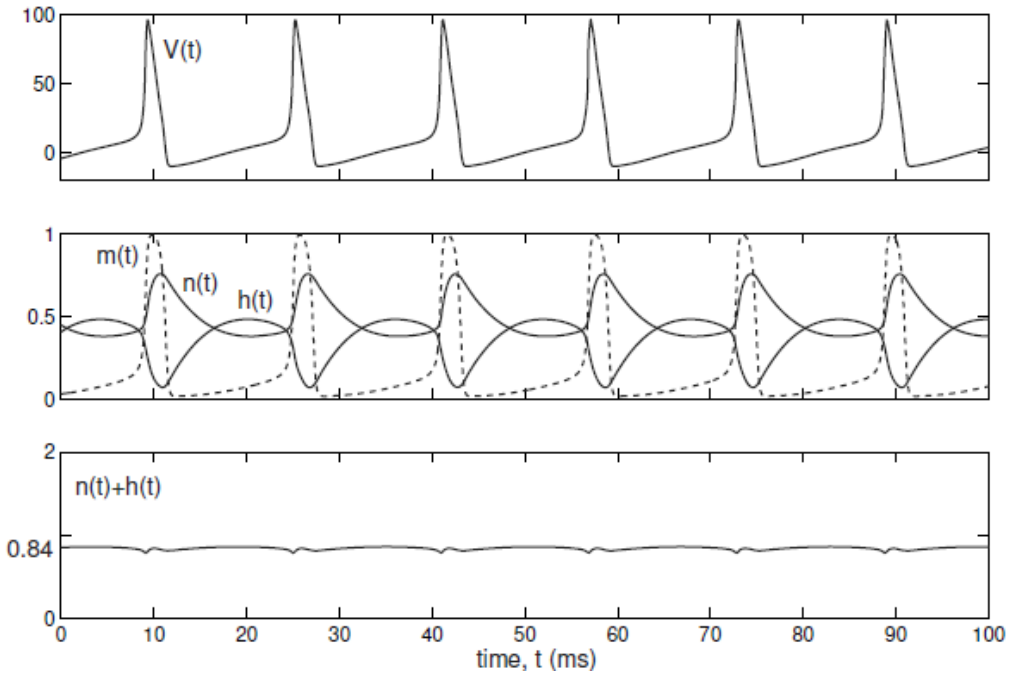


Fig. 1.8. La suma de $n(t) + h(t) \approx 0.84$ en el modelo de Hodgkin-Huxley.

$$h = 0.89 - 1.1n \tag{1.8}$$

Si se usa la ecuación (1.8) en (1.6) y además se asume que la cinética de activación de la corriente de Na^+ es instantánea, entonces $m = m_0(u)$, obteniendo así una reducción a un sistema bidimensional:

$$C\frac{du}{dt} = -g_{Na}m_0^3(0.89 - 1.1n)(u - E_{Na}) - g_K n^4(u - E_K) - g_L(u - E_L) + I(t)$$
$$\dot{n} = -\frac{1}{\tau_n(u)}[n - n_0(u)] \tag{1.9}$$

Cuyas soluciones mantienen un acuerdo cualitativo y en parte cuantitativo con el modelo original de cuatro dimensiones (ver fig.1.10).

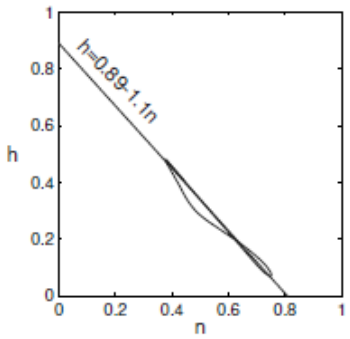


Fig. 1.9. Relación entre $n(t)$ y $h(t)$.

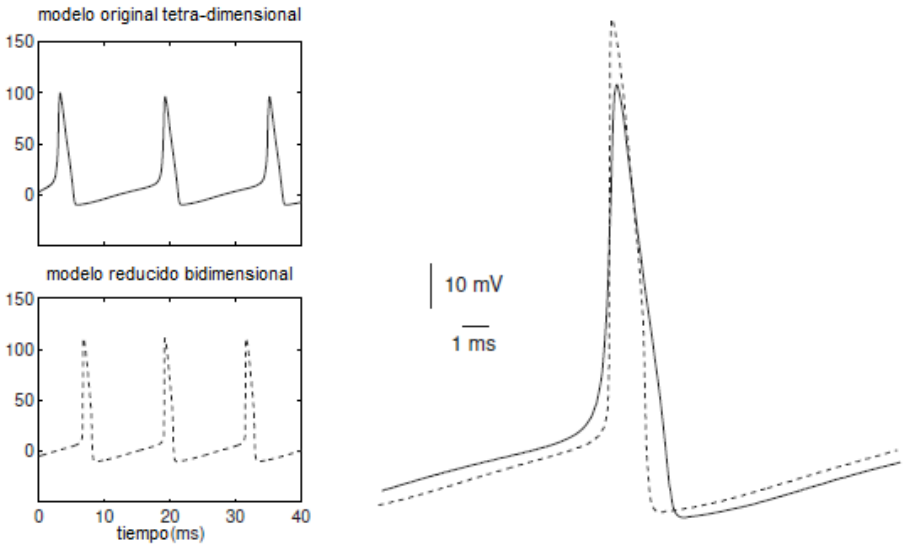


Fig. 1.10. Potencial de acción del modelo original (arriba) y del modelo reducido (abajo) con una densidad de corriente de $I(t) = 8\ \mu A/cm^2$

Para comenzar cualquier análisis de un sistema de dos dimensiones, lo primero que se necesita es encontrar todas sus isóclinas nulas, esto es, $\frac{du}{dt} = 0$ y $\frac{dn}{dt} = 0$. Para encontrar la isóclina nula de u basta con resolver numéricamente para n la siguiente ecuación:

$$I(t) - g_{Na}m_0^3(0.89 - 1.1n)(u - E_{Na}) - g_K n^4(u - E_K) - g_L(u - E_L) = 0 \quad (1.10)$$

Observar que la isóclina nula tiene una forma conocida como “N” (ver fig.1.11). Nótese que solo tiene una intersección con la isóclina nula de n que es $n = n_0(u)$, por lo tanto, solo hay un equilibrio, que es estable cuando $I = 0$. Cuando el parámetro I aumenta, el equilibrio pierde estabilidad (se vuelve inestable o asintóticamente inestable) por efecto de una bifurcación Andronov-Hopf. Cuando I es lo suficientemente grande (por ejemplo $I = 12\ \mu A/cm^2$, ver fig.1.11) se genera un atractor de ciclo límite que corresponde a una pulsación periódica, por lo que este proceso sucede desde el estado de reposo, el cual es el punto de intersección de ambas isóclinas nulas en el punto de inflexión

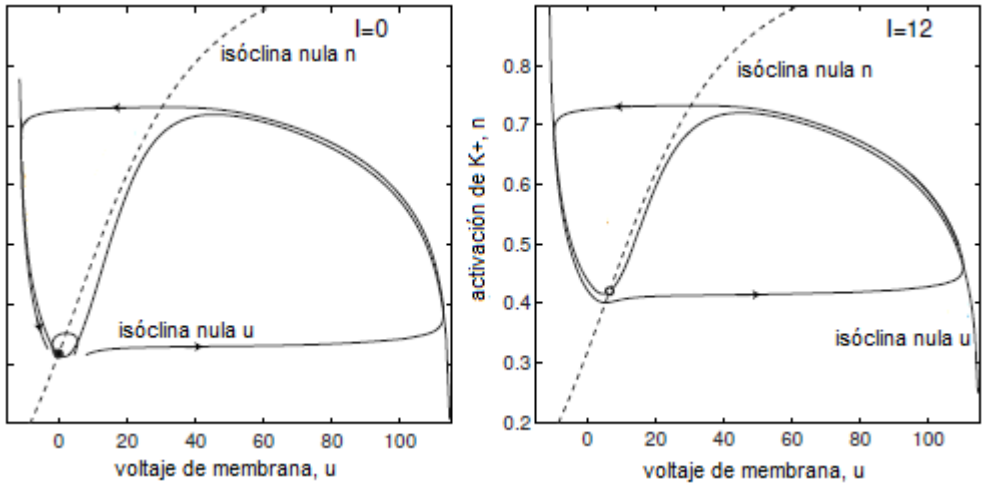


Fig. 1.11. Reducción del modelo de Hodgkin-Huxley para el plano de fase (u, n).

inferior por lo que ésta es la región que más información proporciona. Esta región está delimitada por un cuadro sombreado (ver fig.1.12a). A partir de dicha región se puede obtener un modelo más simple de la reducción a dos dimensiones. Primero consideremos que el segmento de la isóclina nula encerrada en el cuadro sombreado (isóclina rápida) se puede aproximar usando una parábola cuadrática:

$$v = v_{min} + p(u - u_{min})^2 \tag{1.11}$$

donde $(u_{min},\ v_{min})$ es la ubicación del punto de inflexión mínimo, y $p \geq 0$ es un coeficiente de escalamiento. De forma similar, la isóclina nula representada

con la línea punteada (isóclina lenta) se puede aproximar a través de una línea recta:

$$v = s(u - u_0) \tag{1.12}$$

donde s es la pendiente y u_0 es el cruce con el eje u (ver fig.1.12). Todos estos parámetros pueden ser determinados fácilmente, ya sea geométricamente o analíticamente. Con las expresiones anteriores, se puede aproximar la dinámica de la región sombreada a través del siguiente sistema:

$$\begin{aligned} \dot{u} &= \tau_f\{p(u - u_{min})^2 - (u - u_{min})\}, \\ \dot{v} &= \tau_s\{s(u - u_0) - u\} \end{aligned} \tag{1.13}$$

donde los parámetros τ_f y τ_s describen las escalas de tiempo de las isóclinas nulas. Debido al término $(u - u_{min})^2$, la variable u puede aproximarse al infinito en un tiempo finito. Por lo tanto, podemos concluir que esto corresponde a un disparo del potencial de acción, más precisamente, a su etapa ascendente. Para modelar la etapa descendente, se asume que el potencial u_{max}, es el valor pico de potencial de acción y en el instante siguiente, el sistema pasa al estado de restablecimiento.

$$(u, v) \leftarrow (u_{reset}, v + v_{reset}), \qquad donde\ u = v_{max} \tag{1.14}$$

Con un escalamiento apropiado de las variables, se puede obtener una forma equivalente del modelo:

$$\begin{aligned} \dot{u} &= I + u^2 - v \quad si\ u \geq 1,\ entonces \\ \dot{v} &= a(bu - v) \quad u \leftarrow c, v \leftarrow v + d \end{aligned} \tag{1.15}$$

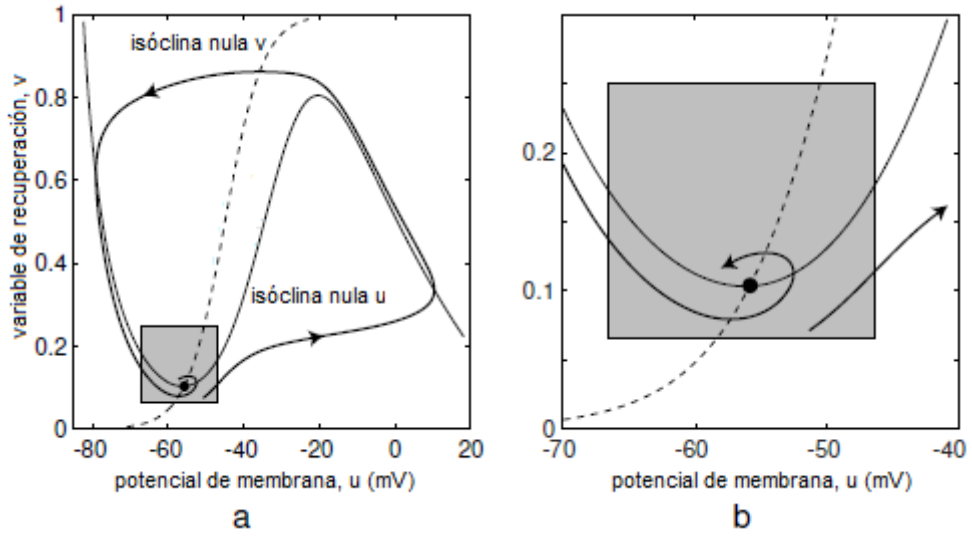


Fig. 1.12. Retrato de fase (a) y su amplificación (b) de un modelo neuronal típico con variable de voltaje u y variable de recuperación v.

Los parámetros del modelo simple pueden ser obtenidos usando relaciones instantáneas $(I-V)$ de estado estacionario [5]. De esta manera se puede representar al modelo simple de (1.15) con una forma equivalente:

$$\begin{aligned} C\dot{v} &= k(v-v_r)(v-v_t)-u+I \qquad si\ v \geq v_{pico}, entonces \\ \dot{u} &= a\{b(v-v_r)-u\} \qquad\qquad v \leftarrow c, u \leftarrow u+d \end{aligned} \tag{1.16}$$

donde C es la capacitancia de membrana. Es importante aclarar que se realizaron algunas modificaciones de nomenclatura de tal manera que las ecuaciones de la reducción del modelo de Hodgkin-Huxley al modelo de Izhikevich tengan coincidencia con las ecuaciones presentadas en [11]. Este cambio consiste en intercambiar las literales $v \rightarrow u$ y $u \rightarrow v$, por lo que ahora el potencial de membrana será representado por v y la variable de recuperación será u. Esto no afecta en ninguna medida el desarrollo presentado anteriormente.

En la fig.1.13 se presenta la relación entre los parámetros de modelo simple de la ecuación (1.16) y las relaciones instantáneas $(I-V)$ de estado estacionario. En la ecuación (1.16) se puede observar el polinomio cuadrático $-k(v-v_r)(v-v_t)$ el cual aproxima la parte de sub-umbral de la relación instantánea $(I-V)$, $I_0(V)$. En la expresión, v_r es el potencial de reposo de la membrana y v_t es potencial instantáneo de umbral (ver fig.1.13). Por otra parte, el polinomio $-k(v-v_r)(v-v_t)+b(v-v_r)$ aproxima la parte de sub-umbral de la relación $I_\infty(V)$. Cuando $b<0$, su máximo se aproxima a la ***reobase*** de la neurona que en neurociencias, es la amplitud mínima de la corriente que da como resultado el umbral de despolarización del potencial de membrana. Las constantes a, b, c y d que aparecen tanto en la ecuación (1.15) como en la (1.16) serán explicadas más adelante con más detalle ya que en función de sus valores, el potencial de membrana mostrará un comportamiento específico.

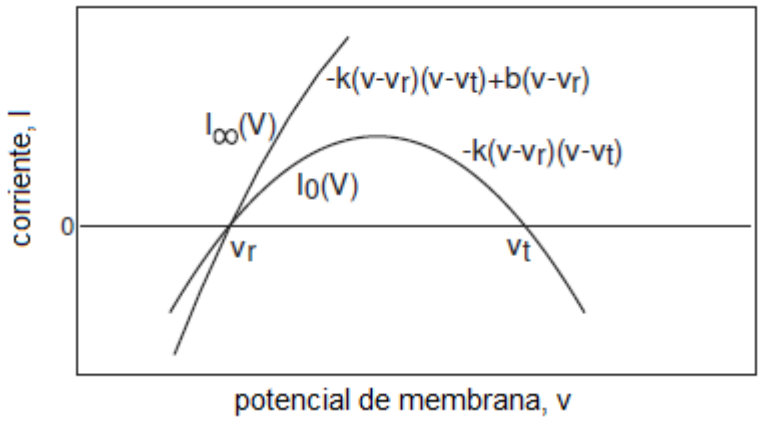


Fig. 1.13. Relación entre los parámetros de modelo simple de la ecuación (1.16) y las relaciones instantáneas $(I-V)$ de estado estacionario.

Los parámetros del modelo (1.16) se pueden obtener al realizar un ajuste de la dinámica de inicio de pulsación de una neurona cortical (también son viables otras opciones). De este modo el potencial de membrana tendrá escala de mV y el tiempo tendrá escala de ms [11]. Como la mayoría de las neuronas reales, este modelo no tiene un umbral definido. Dependiendo del historial del potencial de membrana antes de disparar un pulso, el umbral del potencial puede ser tan bajo como -55 mV o tan alto como -40 mV. Con estas

consideraciones el modelo de Izhikevich para el potencial de membrana de las neuronas pulsadas tiene la siguiente forma:

$$v' = 0.04^2 + 5v + 140 - u + I \tag{1.17}$$

Y con un restablecimiento auxiliar después de un pulso:

$$si\ v \geq 30\ mV, entonces \begin{cases} v \leftarrow c \\ u \leftarrow u + d. \end{cases} \tag{1.18}$$

Como ya se ha mencionado, v representa el potencial de membrana y u es la variable de recuperación de membrana, la cual explica la activación de las corrientes iónicas de K^+ y la inactivación de las corrientes iónicas de Na^+, provee también una retroalimentación negativa a v. Después de que el pulso alcanza su valor pico $(+30\ mV)$ es necesario restablecer el potencial de membrana y la variable de recuperación de acuerdo a (1.18). Las corrientes sinápticas y también las corrientes inyectadas de DC deben ser incluidas en la variable I. Los parámetros a, b, c y d requieren una explicación un poco más detallada:

- El parámetro a describe la escala de tiempo de la variable de recuperación, por lo que usar valores pequeños genera una recuperación lenta. Un valor típico es de 0.02.
- El parámetro b es la sensibilidad de la variable de recuperación ante las fluctuaciones de sub-umbral del potencial de membrana, por lo tanto valores grandes acoplan fuertemente a v y a u resultando en posibles oscilaciones de sub-umbral y dinámicas de pulsación de umbral bajo. Se escoge un valor típico de 0.2. El caso de $b < a$ corresponde a un bifurcación de estado estable de silla-nodo mientras que el caso $b > a$ corresponde a una bifurcación de Andronov-Hopf [19].

- El parámetro c es el valor de restablecimiento de v después de una pulsación. Su valor típico es de -65 mV.
- El parámetro d describe el valor de restablecimiento de u después de una pulsación. Su valor típico es de 2.

Diferentes valores de los parámetros resultan en diferentes patrones de pulsación. Estos patrones se muestran en la fig.1.14.

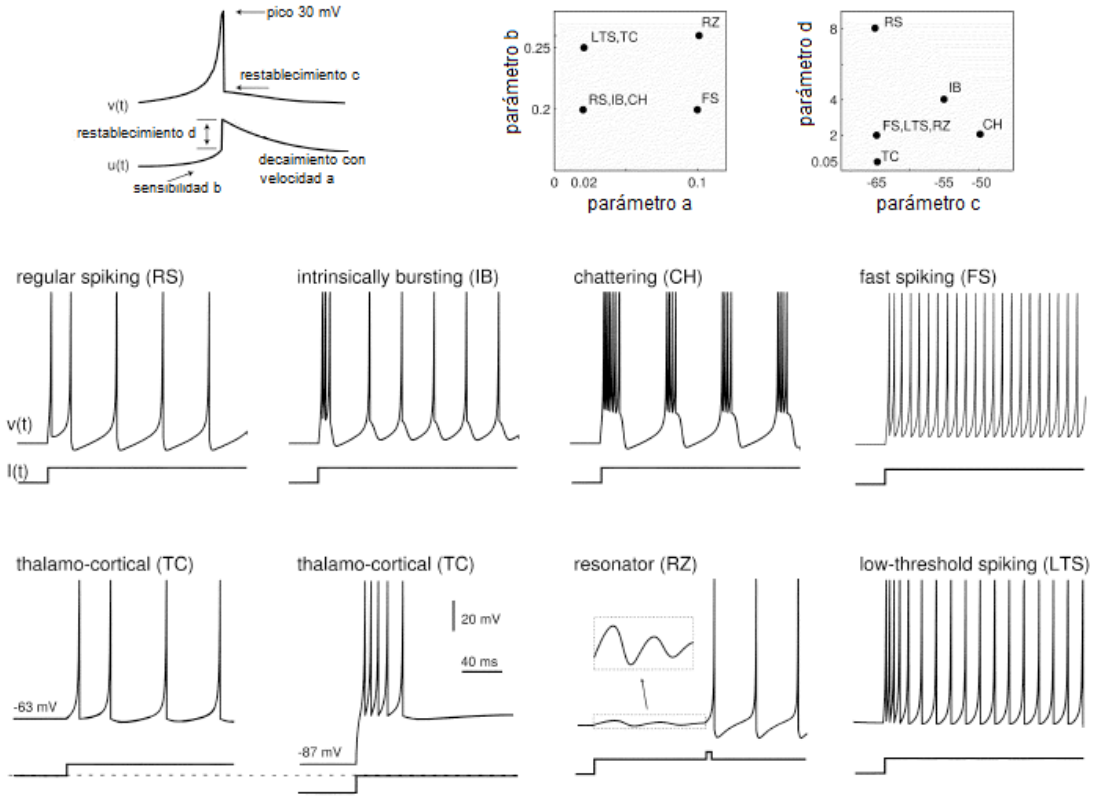


Fig. 1.14. Diferentes tipos de patrones de pulsación para diferentes valores de a, b, c y d.

1.7 Sinapsis química

Un esquema de los componentes fundamentales de una sinapsis química se muestra en la fig.1.15. La sinapsis química es un proceso complejo de transducción de señales que produce una respuesta postsináptica cuando un potencial de acción alcanza la terminal presináptica.

El cambio en el potencial de la membrana causado por la llegada de ese potencial de acción conduce a la apertura de los canales de calcio dependientes de voltaje localizados en la membrana presináptica. Debido al fuerte gradiente de concentración de Ca^{2+} a través de la membrana presináptica, la apertura de estos canales causa un rápido influjo de Ca^{2+} hacia la terminal presináptica, con el resultado de que la concentración de Ca^{2+} del citoplasma en la terminal, aumente transitoriamente a un valor mucho más alto.

La elevación de la concentración presináptica de Ca^{2+}, a su vez, permite que las vesículas sinápticas se fusionen con el plasma de la membrana de la neurona presináptica. Esta fusión provoca que su contenido, compuesto en su mayoría de neurotransmisores, sea liberado hacia la hendidura sináptica. Estos transmisores se difunden a lo largo de la hendidura y se combinan temporalmente con receptores específicos en la membrana de la neurona postsináptica. Esta unión provoca que los canales en la membrana postsináptica se abran (o a veces se cierren), cambiando así la capacidad de los iones de fluir hacia (o fuera de) las neuronas postsinápticas. Todo da como resultado una corriente inducida por neurotransmisores la cual altera la conductancia y el potencial de la membrana de la neurona postsináptica, incrementado o disminuyendo la probabilidad de que esta neurona dispare un potencial de acción. De esta forma, la información es transmitida de una neurona a otra.

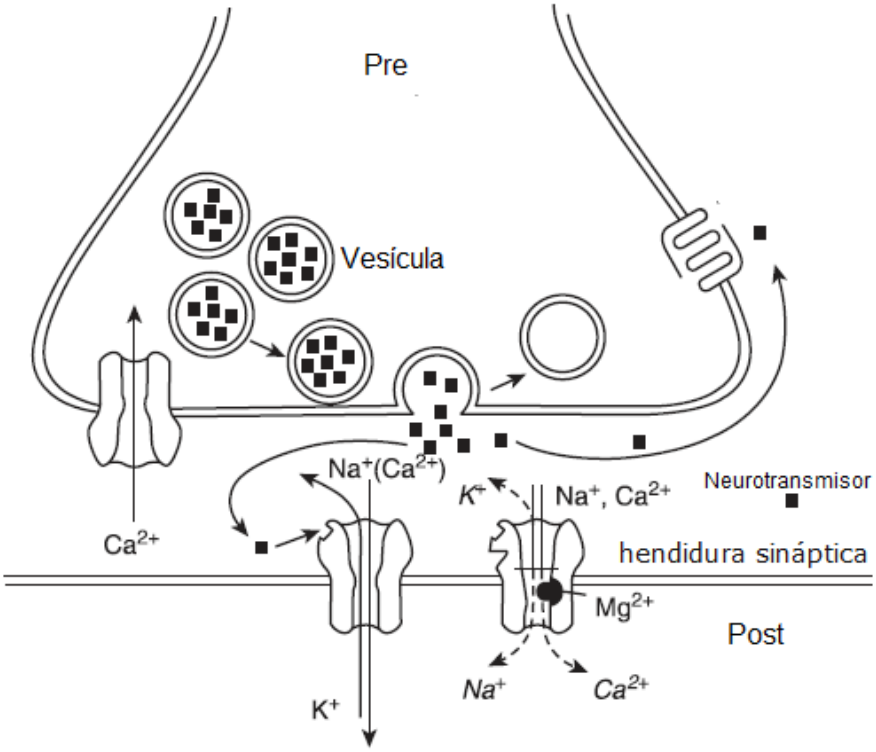


Fig. 1.15. Esquema que muestra los componentes fundamentales de una sinapsis química.

La conductancia $g_{syn}(t)$ al ser una variable transitoria que depende de la llegada de un pulso presináptico, puede ser modelada usando tres formas de onda simples las cuales se presentan en la fig.1.16. Estas tres formas de onda son: (a) exponencial, (b) alfa y (c) doble exponencial y están descritas por las siguientes expresiones:

$$g_{syn}(t) = \exp\left(-\frac{t}{\tau}\right) \tag{1.19}$$

$$g_{syn}(t) = \frac{t}{\tau}\exp\left(-\frac{t}{\tau}\right) \tag{1.20}$$

$$g_{syn}(t) = \frac{\tau_1\tau_2}{\tau_1 - \tau_2}\left(\exp\left(-\frac{t}{\tau_1}\right) - \exp\left(-\frac{t}{\tau_2}\right)\right) \tag{1.21}$$

Las formas de onda alfa y doble exponencial son las representaciones más realistas del cambio de conductancia de una sinapsis típica, aunque la

exponencial se usa con más frecuencia ya que requiere menos recursos computacionales y ofrece resultados bastante aceptables.

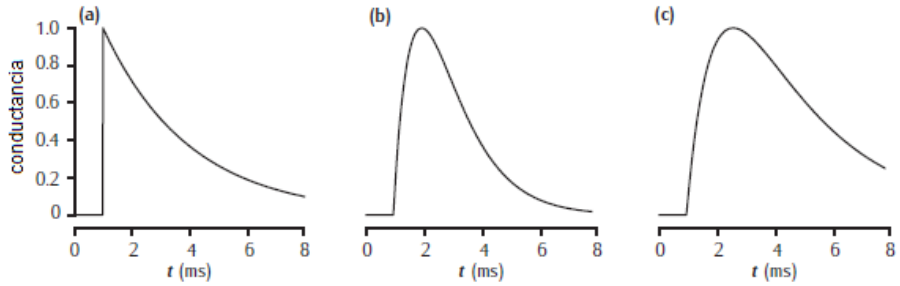


Fig. 1.16. Formas de onda para modelar la conductancia sináptica: (a) exponencial simple con un tiempo de decaimiento de $\tau = 3\,ms$, (b) función alfa con tiempo de decaimiento de $\tau = 1\,ms$, y (c) doble exponencial con tiempos de decaimiento de $\tau_1 = 3ms$ y $\tau_2 = 1ms$. Todas las respuestas se da con un pulso que llega en $t = 1\,ms$, además, todas las conductancias se escalaron a un máximo de 1 con unidades arbitrarias.

1.8 Potenciales postsinápticos excitatorios e inhibitorios

Como ya se mencionó antes, un potencial postsináptico es producido en la membrana celular de una neurona cuando a través de la sinapsis fluye una corriente lo suficientemente grande. Por esta razón se puede también entender que una mayor cantidad de potenciales postsinápticos aumentan la probabilidad de generar un potencial de acción en una neurona postsináptica. Sin embargo, también existen ciertos tipos de sinapsis que disminuyen la probabilidad de generar un potencial de acción en la neurona postsináptica. A los potenciales postsinápticos que ocurren en este tipo de sinapsis se les conoce como inhibitorios o *IPSP,* por sus siglas en inglés de *Inhibitory Postsynaptic Potential.* A los potenciales que aumentan dicha probabilidad se les conoce como excitatorios o *EPSP,* por sus siglas en inglés de *Excitatory Postsynaptic Potential.* Por lo general, se puede hacer referencia a estos potenciales como solamente una entidad neuronal, por lo que también es válido decir que una población de neuronas tiene neuronas excitatorias y neuronas inhibitorias.

Debido a que la mayoría de las neuronas interactúan tanto con sinapsis excitatorias como sinapsis inhibitorias, es importante entender de manera más precisa el mecanismo que determina qué tipo de sinapsis excita o inhibe a su

par postsináptico. La inhibición y la excitación, básicamente hacen referencia a la forma en cómo se comportan los neurotransmisores, esto es, cuando se unen a los receptores para cerrar o abrir los canales iónicos de la neurona postsináptica. Una respuesta postsináptica EPSP o IPSP depende del tipo de canal que está acoplado al receptor y de la concentración de iones permeantes dentro y fuera de la célula. De hecho, la única distinción entre una excitación o inhibición postsináptica es el potencial de inversión del potencial postsináptico en relación con el voltaje de umbral.

Para entender mejor esto, considérese una sinapsis que use glutamato como transmisor. Cuando los receptores de glutamato se activan, tanto los iones Na^+ como los K^+, fluyen a través de la membrana postsináptica, produciendo un potencial de inversión E_{syn} de aproximadamente $0\ mV$ para la corriente postsináptica resultante. Si el potencial de reposo de la neurona postsináptica es de $-60\ mV$, el EPSP resultante se despolarizará al llevar el potencial de la membrana postsináptica hacia $0\ mV$. Por eso, un EPSP inducido por glutamato incrementará la probabilidad que la neurona produzca un potencial de acción, definiendo la sinapsis como excitatoria.

Como ejemplo de una acción inhibitoria postsináptica, considérese una sinapsis que use GABA como transmisor. En tal sinapsis, los receptores de GABA abren los canales que son selectivamente permeables al ion Cl^-. La acción de GABA provoca que el Cl^- fluya hacia la neurona. En este caso, E_{Cl} es $-70\ mV$, que es un valor típico de muchas neuronas, de modo que el potencial de reposo postsináptico de $-60\ mV$ (valor típico) es menos negativo que E_{Cl}. La fuerza impulsora electroquímica positiva resultante $(v - E_{syn})$ causará que el Cl^- cargado negativamente fluya hacía la célula y produzca un IPSP hiperpolarizante. Este IPSP hiperpolarizante, apartará la membrana postsináptica del umbral de $-40\ mV$, claramente inhibiendo la neurona postsináptica.

Como se puede observar, los voltajes producidos por todos estos procesos, son relativamente pequeños, en un rango de unas cuantas decenas de milivolts, suficientes para poder transmitir información. Esto es posible ya que en una neurona existen miles de sinapsis, y los potenciales postsinápticos generados por cada sinapsis activa pueden sumarse, en espacio y en tiempo, para determinar el comportamiento de la neurona postsináptica.

Considérese el caso simplificado en el cual una neurona está inervada por dos sinapsis excitatorias, cada una generando un EPSP de sub-umbral, y una sinapsis inhibitoria que produce un IPSP (ver fig.1.17a). Mientras la activación por separado de cada de una de las sinapsis excitatorias ($E1$ o $E2$ de la fig.1.17b) produce un EPSP de sub-umbral, la activación de ambas sinapsis casi al mismo tiempo causa que los dos EPSP se sumen. Si esta suma ($E1$ + $E2$) despolariza la neurona postsináptica lo suficiente para alcanzar el umbral, se genera un potencial de acción.

La sumatoria permite, por lo tanto, que los EPSP de sub-umbral influyan en la producción de potenciales de acción. Del mismo modo, un IPSP generado por una sinapsis inhibitoria (I) puede sumarse (algebraicamente hablando) con un EPSP de sub-umbral para reducir su amplitud ($E1 + I$) o puede sumarse con los EPSP de supra-umbral para prevenir que la neurona postsináptica alcance el umbral ($E1 + I + E2$).

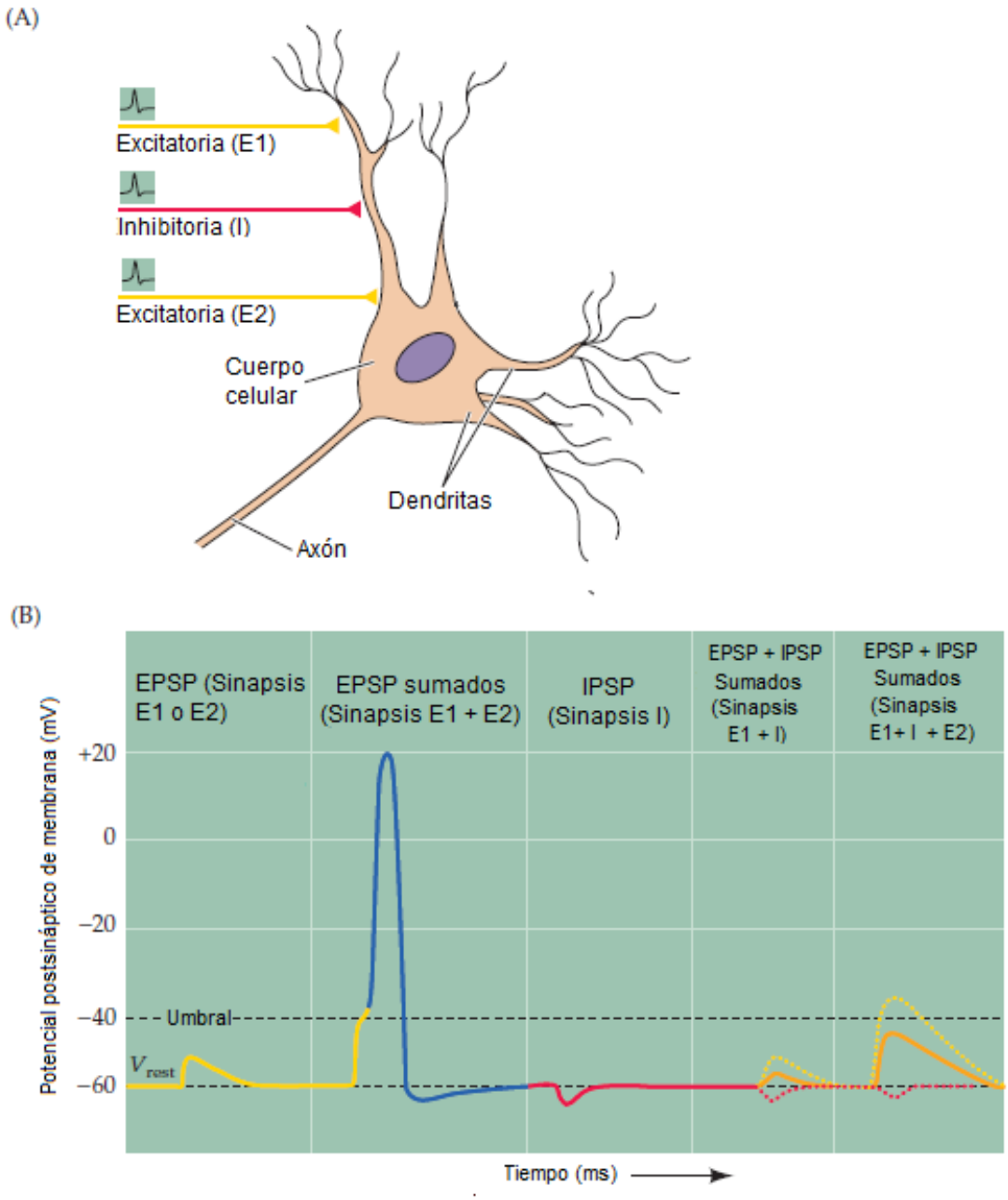


Fig. 1.17. Sumatoria de los potenciales postsinápticos.

1.9 Conclusiones

Para entender cómo trabaja el cerebro, es necesario combinar estudios experimentales con simulaciones numéricas de modelos neuronales a gran escala. A medida que se desarrollan tales modelos, se deben tomar ciertos puntos clave, como por ejemplo: los modelos para una neurona deben ser computacionalmente simples y capaces de producir estímulos como los exhibidos por las neuronas biológicas reales. Para desarrollar dichos modelos, es necesario entender a niveles adecuados, la electrofisiología de la neurona, y hacer énfasis en los fenómenos que interaccionan con la membrana celular, como es el caso de la transmisión de información a través de una sinapsis química. Una herramienta que ha demostrado ser fundamental para el modelado computacional de la neurona, es el retrato de fase ya que permite de una manera totalmente gráfica, representar un fenómeno en términos de su dinámica. De esta forma es posible obtener parámetros y aproximaciones que, de realizarlos de manera analítica, se volvería algebraicamente prohibitivo.

Una de las razones fundamentales para generar un modelo plausible, es reducir las falsas suposiciones de un fenómeno y llevarlo a un entorno controlable como es la simulación, además de que también un modelo bien hecho, remueve las ambigüedades de las teorías, ya que asegura que las suposiciones hechas por diversos observadores, se formalicen y sean entendibles de una misma forma, explícita y lógicamente consistente. Finalmente, los modelos ayudan formidablemente a probar hipótesis, reemplazando en algunos casos, a los experimentos.

2 REDES NEURONALES ARTIFICIALES PULSADAS

2.1 Introducción

En muchas áreas del cerebro, las neuronas están organizadas en poblaciones de unidades con propiedades similares. Ejemplos prominentes son las columnas en la corteza somato-sensorial y visual, así como los conjuntos de neuronas motoras. Dado este gran número de neuronas dentro de dichas columnas o conjuntos, es sensato describir la actividad media de la población neuronal en lugar de solamente un pulso individual de una sola neurona. Por esta razón, el propósito de este capítulo es describir cómo interactúan las neuronas cuando son agrupadas en poblaciones llamadas “redes neuronales” así como también sus características más sobresalientes.

2.2 Relación entre redes neuronales biológicas y artificiales

Una red neuronal es básicamente un arreglo estructurado de “procesadores” interconectados y organizados por niveles o capas. La más básica de las redes está organizada en tres capas: la primera capa se encarga de recibir los estímulos o información del exterior. Esta información no representa nada para la primera capa, es solo información “en bruto”. La segunda capa recibe la información proveniente de la primera capa, con características que permiten procesarla de forma que sea “comprensible” para esta segunda capa. A este proceso se le conoce como codificación y es el proceso fundamental de todas las redes neuronales.

Cuando la información ya ha sido codificada, se envía a la tercera y última capa, la cual se encarga de “traducir” la información codificada y generar una respuesta apropiada con algún nivel de correlación con la información de entrada. Aunque esta definición se ajusta mejor a una red neuronal artificial, el proceso de recepción-codificación-interpretación es aplicable también para las redes neuronales biológicas pero en un grado mayor de complejidad. Es necesario mencionar que la base de estudio de esta tesis son las redes neuronales artificiales, más específicamente las redes neuronales pulsadas.

Por esta razón se puede precisar una definición más formal para una red neuronal artificial, enunciándola de la siguiente manera: una red neuronal artificial es una serie de algoritmos que intentan identificar las relaciones subyacentes en un conjunto de datos. Con base en eso, se pueden enlistar una serie de características que ambas redes neuronales tienen en común mediante el uso de un proceso que imita la manera en la cual funciona el cerebro humano. Estas redes tienen la capacidad de adaptarse a los cambios de entrada para que la red produzca el mejor resultado posible sin la necesidad de rediseñar los criterios de salida. Toda esta información nos permite enlistar una serie de características que deben tener las redes neuronales artificiales para imitar fielmente a las redes biológicas [24]:

1. Representación analógica y procesamiento de información.
2. Habilidad de promediar condicionalmente conjuntos de datos.
3. Tolerancia a las fallas, así como degradación gradual y recuperación tras un error.
4. Adaptación a un cambio de entorno e implementación de funciones “inteligentes” de procesamiento de información a través de auto-organización en respuesta de los datos.

Para implementar una red artificial con estas características se pueden usar diferentes soluciones técnicas, que incluso no se encuentran por si misma de manera natural. Por lo tanto, las redes artificiales no necesitan ser modelos estrictamente fieles de ningún circuito biológico, sino que pueden utilizar, por ejemplo, propiedades específicas del silicio. Además, aunque la representación analógica parece implicar ciertas soluciones a través del uso de cierto tipo de arquitecturas, puede ser emulada digitalmente, especialmente si algún tipo de tecnología circuital es apropiada para ello.

El propósito de cualquier sistema nervioso biológico es centralizar el control de las funciones vitales tales como funciones sensomotoras, ritmos, circulación sanguínea, respiración, metabolismo, funciones de emergencia, etc. Para formas más desarrolladas de comportamiento, los animales

superiores (incluido el humano) poseen conciencia y la habilidad de planificar diversas acciones, haciendo referencia a la memoria y a la imaginación. En neurociencia es más habitual distinguir dos sistemas de control particulares en el cerebro, a saber, el de las funciones cognitivas y el de las reacciones emocionales, respectivamente. A menos que el objetivo del modelado sea la simulación de efectos psicológicos o fisiológicos, el alcance de las redes neuronales artificiales suele ser más restringido. Algunos investigadores tienden a pensar que el objetivo final de la investigación de las redes neuronales artificiales es desarrollar autómatas; en consecuencia, las funciones principales que se pueden implementar con estas redes son:

- Funciones sensoriales (o de forma más general, percepción artificial).
- Funciones motoras (manipulación, locomoción).
- Toma de decisiones (razonamiento, estimación, resolución de problemas).
- Comportamiento verbal (comprensión y producción del habla, lectura y escritura, preguntas-respuestas).

2.3 Evolución de las redes clásicas a redes neuronales pulsadas

La mayoría de los modelos de redes neuronales artificiales se basan en un modelo de neurona (llamado perceptrón o compuerta de umbral) que McCulloch y Pitts propusieron en 1943 como una abstracción de la función computacional de una neurona biológica.

Actualmente se sabe que existen una serie de diferencias fundamentales entre los cálculos en tales redes neuronales artificiales y los cálculos en los circuitos de las neuronas biológicas. Una de esas diferencias se vuelve obvia cuando se considera la salida de la neurona. Esta salida, en una red neuronal artificial es un bit (en el caso del perceptrón) o un número real (en el caso de una función sigmoidal) [25]. En el caso de una neurona biológica, la salida consiste en fuertes aumentos de potencial, que como ya se estudió con anterioridad, reciben el nombre de potenciales de acción o simplemente pulsos. La diferencia fundamental entre la red artificial y la red biológica consiste en que

las redes artificiales se manejan en un modo síncrono, mientras que una red biológica no tiene un reloj central. Los instantes cuando la neurona dispara dependen del valor actual y del valor posterior de la entrada. Por lo tanto, aunque en cualquier momento la decisión de pulsar de una neurona se asemeje a una operación perceptrón, y aunque el número promedio de pulsos (con un promedio tomado en el tiempo o en el espacio) emitidos por la neurona se asemeje a una salida evaluada como función sigmoidal, el tiempo juega un rol completamente diferente en el cálculo de un circuito neuronal biológico. Por esta razón, las neuronas biológicas solo codifican información esencial sobre la entrada en los instantes que se dispara. Esta capacidad puede ser usada, por ejemplo, para transmitir información de forma muy rápida a través de pequeñas diferencias temporales entre pulsos enviados por diferentes neuronas. Las redes neuronales artificiales integradas por neuronas pulsadas son clasificadas como la tercera generación de redes neuronales artificiales.

2.4 Arquitectura "El ganador toma todo (WTA, Winner-Takes-All)"

La arquitectura de una red neuronal artificial define cómo lo grupos de neuronas están organizadas o colocados entre sí. Estos arreglos están estructurados esencialmente por la forma que tienen las conexiones sinápticas de dichas neuronas. La topología de una red neuronal dada, dentro de una arquitectura particular, puede ser definida como las diferentes composiciones estructurales que puede asumir. En otras palabras, es posible tener dos topologías que pertenecen a la misma arquitectura.

Un mecanismo WTA es un dispositivo que determina la identidad, y principalmente la amplitud de su entrada más grande [25]. Tal mecanismo es necesario en los modelos de redes para hacer cumplir la competencia entre las diferentes posibles salidas de la red. Existen dos variantes importantes en un proceso de WTA, el primero es conocido como k-*WTA,* donde la entrada k más grande es identificada. El segundo es una versión simplificada, conocida como *softmax* [26], que consiste en asignar a cada entrada un peso estadístico de tal manera que todas las ponderaciones se sumen en una y la entrada más

grande reciba el mayor peso. Es muy importante aclarar que en ambas variantes, los pesos deben de ser positivos, esto es así porque los pesos positivos pueden ser implementados por conexiones excitatorias, mientras que los pesos negativos no pueden ser implementados de manera directa. En las redes con arquitectura WTA, existe un proceso conocido como *inhibición lateral,* el cual se define como la capacidad de una neurona excitada de reducir la actividad de las neuronas vecinas, esto es, impedir la propagación de potenciales de acción de la neurona pulsante hacía las neuronas vecinas, en una dirección lateral. Una red neuronal con arquitectura WTA puede observarse en la fig.2.1.

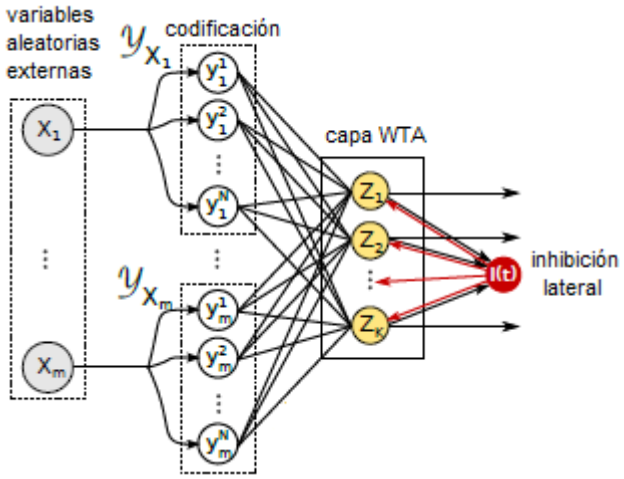


Fig. 2.1. Configuración de una arquitectura WTA con inhibición lateral.

2.5 Interpretación probabilística de la arquitectura WTA

Como ya se ha estudiado anteriormente, la aparición de un potencial de acción en la membrana de una neurona, depende tanto de la actividad eléctrica actual como de la actividad eléctrica posterior. Esto es fácil de interpretar cuando solo se está estudiando una neurona, ya que de alguna forma se puede tener control de ambos eventos. Sin embargo, cuando existe una interacción entre varias neuronas o varios grupos de neuronas interconectados, el análisis matemático se vuelve complejo debido a que las neuronas exhiben un comportamiento completamente probabilístico. De manera más específica, las

neuronas en una red WTA presentan un comportamiento de acuerdo a una distribución inhomogénea de Poisson [17] con razones de disparo que exponencialmente dependen del voltaje de membrana. La ecuación (2.1) expone lo anterior:

$$r_k(t) = \frac{1}{\tau_0} \exp\left(\frac{v_k(t) - I(t)}{V_0}\right) \tag{2.1}$$

donde τ_0 y V_0 son constantes. La ecuación (2.1) es también conocida como *modelo de la respuesta de pulsos estocásticamente generados* o *SRM* por sus siglas en inglés de *Spike Response Model* [8]. Curiosamente, se demostró que este mecanismo simple estaba en una relación muy estrecha con los datos registrados en vivo de los experimentos con neuronas piramidales de la corteza somato-sensorial [27]. La ecuación (2.1) puede usarse también para predecir la razón de disparo en una neurona que usa el modelo Izhikevich, solamente cambiando apropiadamente el valor de las constantes τ_0 y V_0.

Con las propiedades de la arquitectura WTA escritas anteriormente, se sabe que en cualquier instante solo una neurona-*Z* (ver fig.2.1) puede generar un pulso. Además, se sabe que todas las neuronas-*Z,* cuando no se han inhibido, se comportan como un proceso inhomogéneo de Poisson, esto implica que toda la capa-*Z* se comporta de manera similar y su razón de disparo es $\sum_k r_k(t)$. Si se sabe que la capa WTA produce un pulso al tiempo t, se puede escribir una expresión para la probabilidad condicional para dicho pulso emitido por la neurona Z_k:

$$P(Z_k | WTA\ dispara\ al\ tiempo\ t) = \frac{r_k(t)}{\sum_k r_k(t)} = \frac{\exp(v_k(t)\ /\ V_0)}{\sum_j \exp\big(u_j(t)\ /\ V_0\big)} \tag{2.2}$$

Ahora, supóngase que a la red se le presenta cierto patrón X, el cual se codifica por las neuronas de la capa Y, la cual estimula a las neuronas de salida Z. Ahora será necesario demostrar que los pulsos generados por la capa WTA provienen de la distribución posterior $P(k|X)$ bajo el modelo de mezcla multinominal de Naive Bayes [28] (ver fig.2.2).

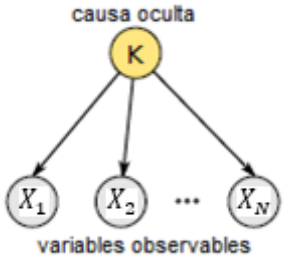


Fig. 2.2. Modelo de mezcla multinominal de Naive Bayes: las variables observables aleatorias, $X_1, \dots, X_m$, dependen de la causa oculta K. Las variables observables son condicionalmente independientes de la causa.

Para demostrar lo anterior, primero se escribe una relación para el estado interno de todas las neuronas Z, $v(t)$, como sigue

$$v(t) = b_k + \sum_{m=1}^{M} \sum_{j=1}^{N} w_{jmk} \cdot y\left(y - t_m^j\right) \equiv b_k + \sum_{i=1}^{n} w_{ik} \cdot y(t - t_i) \qquad (2.3)$$

donde b_k es una constante de excitabilidad intrínseca de la neurona Z_k, $w_{\#}$ son los pesos sinápticos, $y(t)$ son las formas temporales de los potenciales postsinápticos, y $t_{\#}$ son los tiempos de los pulsos propagados a través de la sinapsis. Nótese que la segunda parte de la ecuación (2.3) es una relación equivalente y es usada para simplificar la notación: en lugar de sumar en la población codificadora M y en cada una de las neuronas N, podemos simplemente sumar los n potenciales postsinápticos que actualmente contribuyen a $v_k(t)$. Se asume que los potenciales postsinápticos $y(t)$, en función del tiempo, son los mismos para cada sinapsis, tienen soporte limitado σ, y sus magnitudes están definidas por los correspondientes pesos sinápticos, $w_{\#.}$ Es posible resaltar algunas observaciones: la sinapsis w_{jmk} contribuye al potencial de membrana de la neurona Z_k en cualquier punto en el tiempo t, si y solo si, la neurona codificadora y_m^j ha emitido al menos un pulso durante el periodo $[t - \sigma, t]$. De acuerdo a la definición anterior, esto implica que el componente X_m de la entrada tiene un valor A_j en el tiempo t_m^j. Lo que significa que si el sistema se "congela" en cualquier instante, y se analizan los

potenciales postsinápticos que están actualmente contribuyendo al estado de la neurona Z_k, se puede inferir el patrón, X, que está siendo observado por la red.

Para seguir el análisis, supóngase que la capa WTA produce un pulso al tiempo t. Se puede fijar este tiempo y escribir el siguiente modelo generativo para la actividad de codificación Y durante $[t-\sigma, t]$, y por lo tanto el patrón X que causa el disparo de este pulso desde la capa WTA:

$$P(Y, Z; \boldsymbol{w}) = \frac{1}{\varsigma} \exp\left(\sum_k Z_k \left(b_k + \sum_i w_{ik} y(t - t_i)\right)\right) \tag{2.4}$$

donde Z_k es 1 para la neurona de la capa WTA que produce el pulso, y 0 para el resto, mientras que ς es un coeficiente de normalización. La distribución puede representarse como $P(Y, Z; \boldsymbol{w}) = P(Y|Z; \boldsymbol{w}) \cdot P(Z; \boldsymbol{w})$, y entonces, asumiendo $P(Z; \boldsymbol{w}) = \exp(\sum_k Z_k b_k)$ y usando la regla de Bayes se puede obtener la distribución posterior $P(Z|Y; \boldsymbol{w})$:

$$P(Z|Y; \boldsymbol{w}) = \frac{P(Z) \cdot P(Y|Z)}{\sum_Z P(Z) \cdot P(Y|Z)} = \frac{\exp(\sum_k Z_k b_k) \cdot \exp\left(\sum_{i,k} Z_k w_{ik} y(t - t_i)\right)}{\sum_k \exp(b_k + \sum_i w_{ik} y(t - t_i))} \tag{2.5}$$

Que coincide con la probabilidad condicional de (2.2). En otras palabras, cada pulso representa una muestra de la distribución posterior $P(Y, Z; \boldsymbol{w})$ del modelo de mezcla multinominal de Naive Bayes, parametrizado por los pesos, $w_{\#}$.

2.6 Plasticidad Sináptica: Regla de Hebb

El concepto de plasticidad sináptica fue propuesto por primera vez como un mecanismo de aprendizaje y memoria basado en los análisis teóricos de Donald Olding Hebb en 1949 [30]. La regla de plasticidad propuesta por Hebb postula que cuando cierta neurona impulsa la actividad de otra neurona, la conexión entre ambas neuronas es potenciada.

Los análisis teóricos indican que no solo potenciación sináptica hebbiana es necesaria sino también la depresión entre las dos neuronas, las cuales no están suficientemente coactivas. La depresión es necesaria por muchas

razones, entre ellas la de evitar que todas las sinapsis se saturen a sus valores máximos y perder así su selectividad y también para evitar un ciclo de retroalimentación positiva entre la actividad de la red y los pesos sinápticos.

Las correlaciones experimentales de estas formas teóricamente propuestas de la plasticidad sináptica son llamadas potenciación a largo plazo (*Long-Term Potentiation, LTP)* y depresión a largo plazo *(Long-Term Depressión, LTD)*.

2.7 Spike-Timing Dependent Plasticity (STDP)

STDP es una forma temporalmente asimétrica del aprendizaje hebbiano inducida por estrechas correlaciones temporales entre los pulsos de neuronas pre y postsinápticas. Al igual que con otras formas de plasticidad sináptica, se cree ampliamente que subyace el aprendizaje y el almacenamiento de información en el cerebro, así como el desarrollo y el refinamiento de los circuitos neuronales durante el desarrollo del cerebro. En STDP, la llegada repetida de pulsos unos pocos milisegundos **después** de los pulsos postsinápticos, conduce a muchos tipos de sinapsis a una potenciación de largo plazo, mientras que la llegada repetida de pulsos unos pocos milisegundos **antes** de los pulsos postsinápticos conduce a una depresión de largo plazo de la misma sinapsis. El cambio de la sinapsis trazada como una función del tiempo relativo de los potenciales pre y postsinápticos se denomina función STDP o ventana de aprendizaje y varía entre los tipos de sinapsis.

En términos de los resultados experimentales se han generado varios modelos para representar la función STDP. El más usado es el modelo básico y expresa que el cambio del peso Δw_j de una sinapsis desde una neurona presináptica j depende del tiempo relativo entre la llegada del pulso presináptico y los pulsos postsinápticos. Nombremos a los tiempos de llegada de los pulsos presinápticos a la sinapsis j como t_j^f donde $f = 1,2,3, \ldots$ y cuenta los pulsos presinápticos. Similarmente, t_i^n con $n = 1,2,3, \ldots$ denota los tiempos de disparo de la neurona postsináptica. El cambio total de peso sináptico Δw_j inducido por

el protocolo de estimulación con pares de pulsos pre y postsinápticos es entonces:

$$\Delta w_j = \sum_{f=1}^{N} \sum_{n=1}^{N} W\left(t_i^n - t_j^f\right) \tag{2.6}$$

donde $W(x)$ denota una de las funciones STDP, la cual se ilustra en la fig.2.3. Una elección común para esta función $W(x)$ es:

$$\begin{aligned} W(x) &= A_+ \exp\left(-\frac{x}{\tau_+}\right) \; para \; x > 0 \\ W(x) &= -A_- \exp\left(\frac{x}{\tau_-}\right) \; para \; x < 0 \end{aligned} \tag{2.7}$$

La cual se ha usado para ajustar los datos experimentales y los modelos. Los parámetros A_+ y A_- pueden depender del valor actual del peso sináptico w_j. Las constantes de tiempo τ_+ y τ_- están en el orden de decenas de milisegundos.

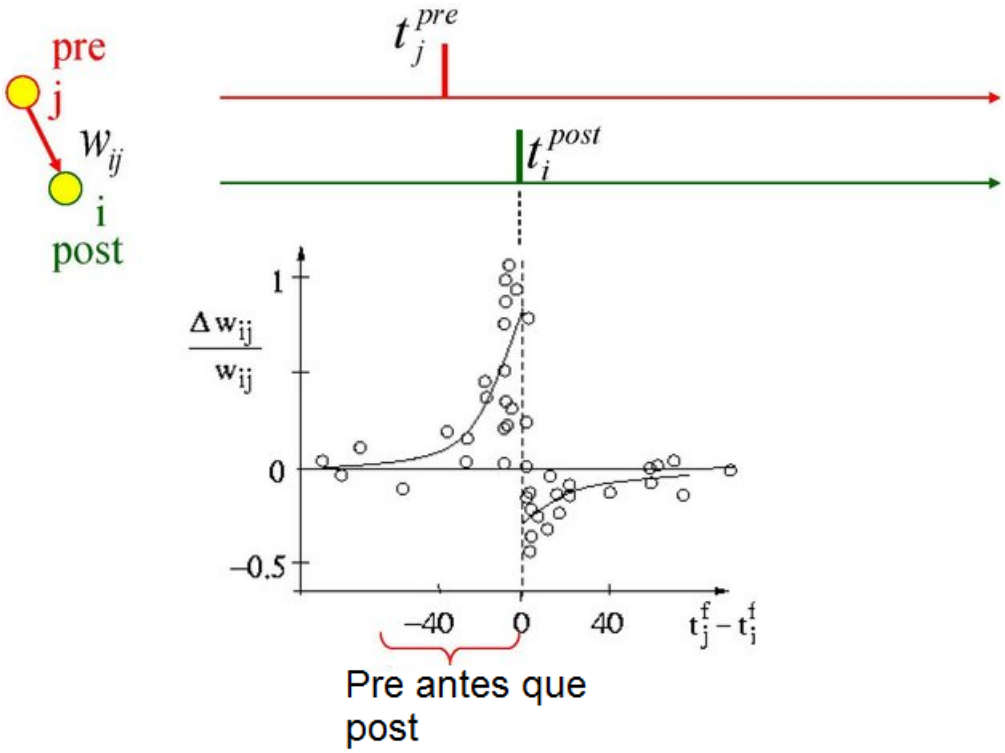


Fig. 2.3. Función STDP y su evolución de acuerdo al cambio relativo del tiempo entre los pulsos presinápticos y postsinápticos. Los círculos blancos muestran los datos experimentales.

2.8 Conclusiones

Un componente esencial en el la obtención de un modelo, es llevar a cabo las suposiciones adecuadas. Esto es particularmente importante cuando se modela una red neuronal. Generalmente, no es posible representar cada neurona de un sistema real en un solo modelo, y por eso se deben plantear varias preguntas de diseño. Las principales preguntas deben hacer referencia al número de neuronas en la arquitectura de la red, en cómo cada neurona debe ser modelada y en cómo las neuronas deben interactuar entre ellas. De acuerdo a esto, construir un modelo de la red neuronal requiere enfocarse en cada etapa del diseño del modelo, comenzando con la elección apropiada del nivel de descripción de las neuronas individuales dentro de la red. También se deben especificar los números relativos de neuronas dentro de las diferentes subpoblaciones y cómo se interconectan. Dentro de este proceso de diseño, algo sumamente importante es la elección de la regla de aprendizaje sináptico que permitirá a la red responder a los estímulos deseados. Una de las más populares y que muestra resultados sólidos, es la ya mencionada regla de Hebb la cual permite generar a largo plazo, una memoria asociativa además de que es posible simularla a través de una simulación basada en eventos o transitoria, como se describirá en la secciones posteriores.

3 MEMRISTORES

3.1 Introducción

Antes del año 1971, solo nueve tipos de elementos de dos terminales (cinco pasivos y cuatro activos) se utilizaban para modelar cualquier componente eléctrico o circuito. Cada elemento estaba definido por una relación entre las variables de estado, corriente I, voltaje V, carga Q y flujo φ incluyendo también una clasificación que depende de la linealidad del elemento. Con base en esta premisa, este capítulo presenta una introducción al estudio de un nuevo elemento pasivo de dos terminales, conocido como *memristor.* También se estudiarán tres diferentes tipos de modelos estocásticos para realizar una comparación con sus homólogos deterministas.

3.2 Definición básica del memristor

Un memristor es un dispositivo pasivo que surge de las relaciones de las cuatro variables fundamentales de los circuitos: corriente i, voltaje v, carga q y flujo φ. La relación entre estas variables se deduce a partir de la ley de inducción de Faraday. Un resistor está definido por la relación entre voltaje v y corriente i, ($dv = Rdi$), un capacitor está definido por la relación entre la carga q y el voltaje v ($dq = Cdv$) y el inductor está definido por la relación entre el flujo φ y la corriente i ($d\varphi = Ldi$). Además, la corriente i está definida como la derivada temporal de la carga q de acuerdo a la ley de Faraday y el voltaje v está definido por la derivada temporal del flujo φ. Estas relaciones se muestran en la fig.3.1.

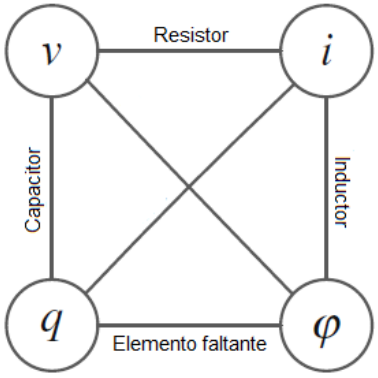


Fig. 3.1. Relación entre los elementos de un circuito.

Este elemento faltante fue propuesto por el profesor Leon Chua en 1971 y se nombró *memristor,* cuya características predichas se basan en la retención del valor de la resistencia cuando la fuente de polarización ha sido desconectada del elemento [33].

3.2.1 El memristor de Hewlett-Packard

En 2008, los laboratorios HP anunciaron que después de una serie de experimentos, habían encontrado el elemento faltante y publicaron sus resultados en Nature [31]. Además el equipo de HP introdujo el primer modelo básico del memristor el cual es gobernado por la formulación matemática de Chua de un sistema memristivo [32]. La estructura del memristor de HP está compuesto por dos capas químicas diferentes, la primera de dióxido de titanio TiO_2 y la segunda con dióxido de titanio con vacancias de oxígeno TiO_{2-x} (ver fig.3.2) por lo que se puede considerar como un material nano-estructurado. Las vacancias de oxígeno son donadoras de electrones, por lo tanto las vacancias están cargadas positivamente. Cuando se aplica un voltaje positivo al electrodo superior del dispositivo, las vacancias de oxígeno serán repelidas hacía la capa de TiO_2 puro.

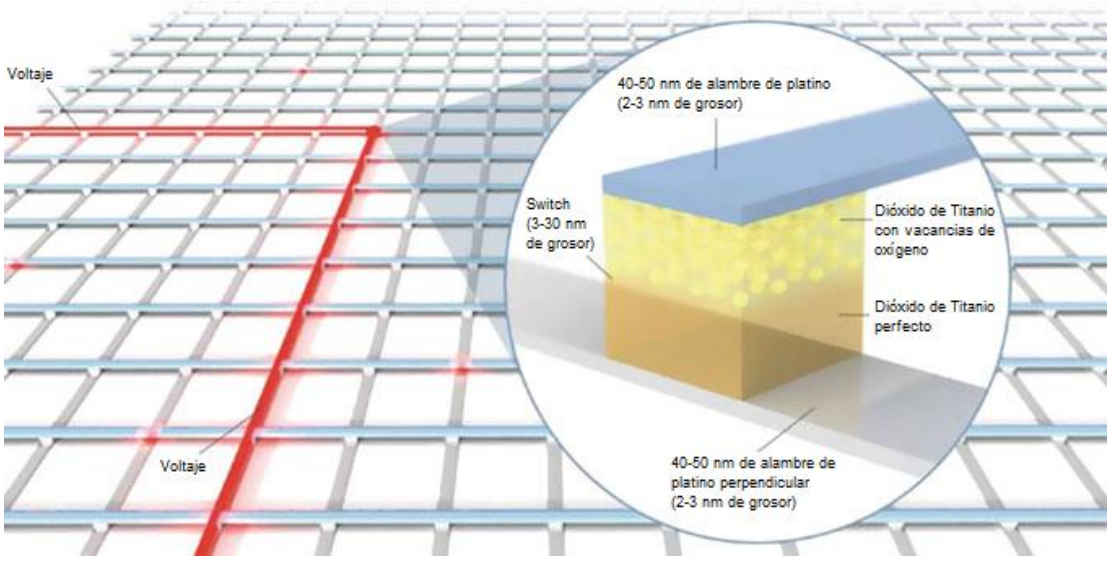


Fig. 3.2. Estructura del memristor de HP.

Esta transferencia de vacancias de oxígeno incrementa el ancho de la capa de TiO_{2-x} y reduce el ancho de la capa de TiO_2, pero al aplicar un voltaje negativo, se puede obtener el efecto inverso, de esta manera se incrementa el ancho de la capa de TiO_2 y se reduce la capa de TiO_{2-x} ya que ahora las vacancias de oxígeno son atraídas hacía el electrodo superior.

De acuerdo a la descripción previa, el equipo de HP presenta un modelo matemático para el memristor. Este modelo está basado en dos resistores en serie R_{on} y R_{off}, donde R_{on} y R_{off} son las resistencias de las regiones dopadas y no dopadas. Se asume que el dispositivo físico tiene un ancho D, y la región dopada tiene un ancho w, como se muestra en la fig.3.3. Nótese que la región dopada con ancho w, es la variable de estado que cambia en función de la carga:

$$M(t) = R_{on}\frac{w(t)}{D} + R_{off}\left(1 - \frac{w(t)}{D}\right) \tag{3.1}$$

$$\frac{dw}{dt} = \frac{\mu_v R_{on}}{D} i(t) \tag{3.2}$$

donde μ_v es la movilidad iónica promedio.

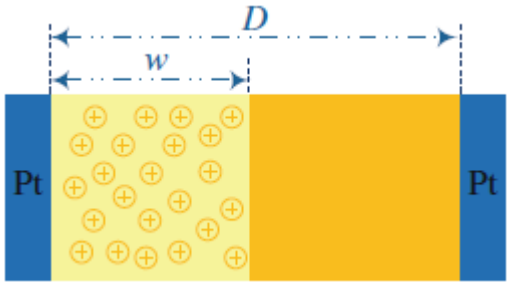


Fig. 3.3. Modelo del memristor de HP.

Después de que HP presentó su memristor, se han propuesto varios modelos a lo largo de los últimos años para explicar el comportamiento de los memristores. Los modelos varían en un rango de complejidad desde la descripción física de los mecanismos de funcionamiento hasta modelos matemáticos más generalizados. No obstante, la estocasticidad, un fenómeno ampliamente observado en todos los sistemas físicos reales, se ha pasado por

alto en sobremanera desde la perspectiva del modelado. Esta variabilidad inherente dentro de la operación del memristor es una característica vital para la integración de este elemento no lineal dentro del área de la nano-electrónica estocástica. Por esta razón, es necesario proponer un modelo que sea genérico y pueda incorporase en memristores que basan su funcionamiento en un valor de umbral, como es el caso de la mayoría de los memristores fabricados en la actualidad.

3.3 Modelado estocástico de memristores

La dinámica de los dispositivos memristivos ha sido minuciosamente investigada para modelar adecuadamente su comportamiento no lineal. Los modelos que se han propuesto a lo largo de varios dominios han establecido estándares en términos de complejidad, precisión y practicidad en la simulación y el diseño de circuitos. En general, las operaciones controladas por carga o flujo son las dos variantes que imponen una dependencia en la evolución en el tiempo de la corriente o el voltaje aplicados a través de las terminales de esta nano-estructura. Los modelos que se proponen a continuación son variantes impulsadas por corriente y voltaje con énfasis en la abstracción variable y el ajuste general del efecto de estocasticidad.

3.3.1 Memristor bipolar con umbral

En este modelo [37], los autores introducen un modelo general que exhibe una característica de memoria con una razón de cambio relacionado con el voltaje aplicado. En la versión determinista, ninguna respuesta es percibida por debajo del umbral. Su estado de conmutación comienza a incrementar en cierta razón β que actúa como la pendiente de la dinámica interna del memristor. La fig.3.4a muestra la forma general del dispositivo memristivo basado en un umbral. Su comportamiento está modelado por una relación lineal de $I-V$ y la forma de la función describe el proceso interno y la sensibilidad a los parámetros limitantes como se expone a continuación:

$$I = X^{-1}V_M \tag{3.3}$$

$$\frac{dX}{dt} = f(V_M)\left[\theta(V_M)\theta\left(R_{off} - X\right) + \theta(-V_M)\theta(X - R_{on})\right] \tag{3.4}$$

$$\begin{cases} f(V_M) = \beta V_M - 0.5\beta[|V_M + V_t| - |V_M - V_t|], \\ dV_T = \underbrace{\alpha \cdot \theta\left(V_{T_0} - V_T\right)dt}_{término\ determinista} + \underbrace{\left(|V| - \Delta V - V_{T_0}\right)dN(\tau)}_{término\ estocástico} \end{cases} \tag{3.5}$$

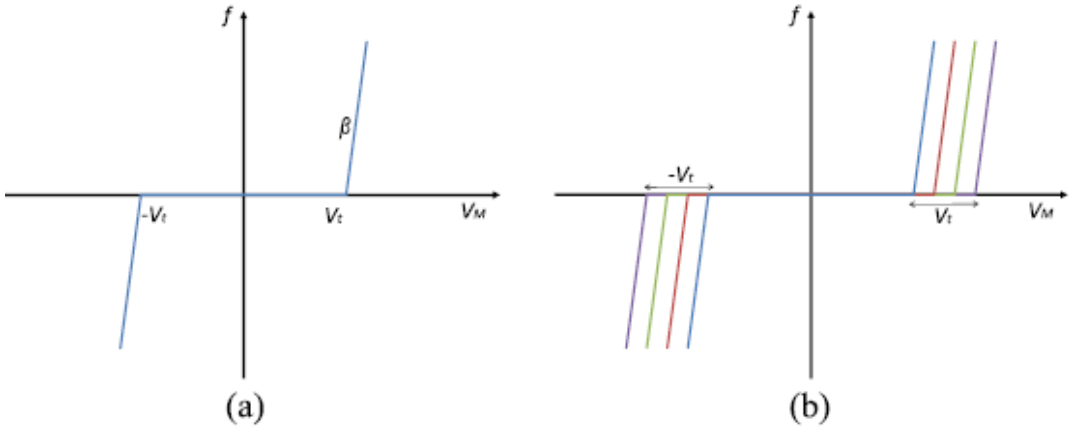


Fig. 3.4. (a) Dinámica interna del modelo controlado por voltaje. (b) Dinámica estocástica: variación resultante del voltaje de umbral con solo el cambio en la localización de la conmutación y la preservación de la función cinética original.

La memristancia se refleja dentro de la variable de estado X y está limitada por las condiciones de frontera superior e inferior R_{off} y R_{on}, respectivamente. La función θ es la función escalón responsable de hacer cumplir esta restricción. La incorporación de la variabilidad en este modelo sigue un enfoque directo donde el voltaje de umbral V_t se varía de alguna forma y acatando las medidas de estocasticidad explicadas anteriormente y expresadas en (3.5). Permite también una conmutación de sub-umbral de una forma no determinista donde el valor del punto de conmutación está relacionado con el tiempo y la amplitud de la entrada de voltaje. La implicación directa para la estocasticidad inherente es un voltaje de umbral variable que es particularmente susceptible a voltajes

cercanos a su valor fijo original donde la mayor probabilidad de conmutación ocurre.

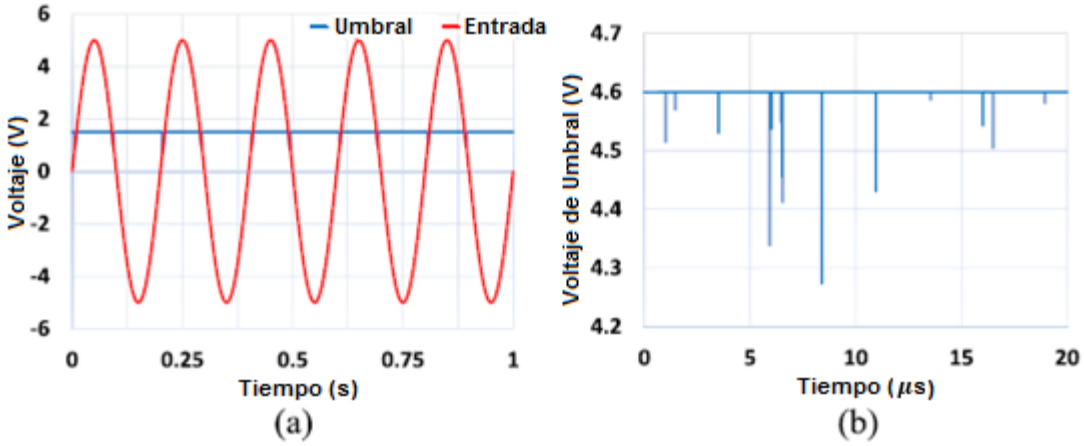


Fig. 3.5. (a) Respuesta a un voltaje de entrada sinusoidal que muestra la variación más grande, particularmente a voltajes cercanos al umbral original. (b) Una resolución más precisa de la variación del umbral.

La variación del umbral es aparente en la fig.3.5b la cual muestra la variación en respuesta del voltaje de entrada aplicado. Este comportamiento se refleja en una fluctuación en la función de conformación en el punto de conmutación mientras se conserva la cinética del sistema como se muestra en la fig.3.4. Además, la salida resultante para este comportamiento estocástico es evidente en la histéresis que muestra la relación entre el voltaje y la corriente.

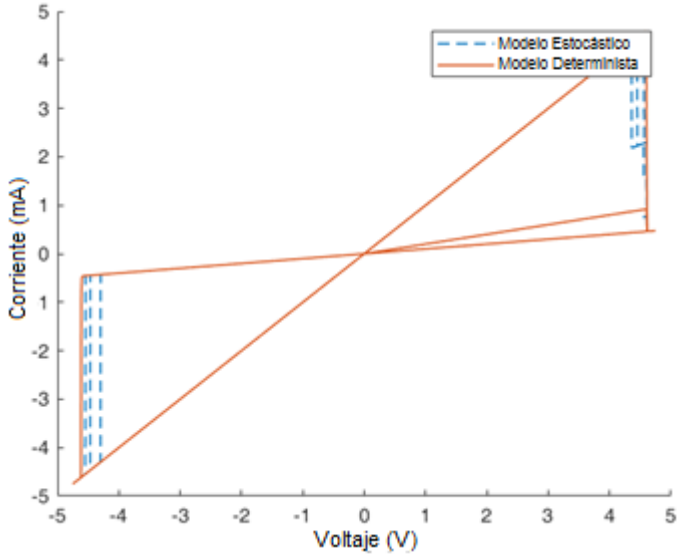


Fig. 3.6. La salida estocástica con la configuración de histéresis externa que limita la conmutación estocástica por debajo del umbral.

En lugar de tener una sola respuesta para una señal sinusoidal de entrada, la salida está contaminada con histéresis internas añadidas, encapsuladas dentro de límites externos establecidos por el umbral determinista de voltaje. La fig.3.6 muestra el comportamiento estocástico en comparación con el comportamiento determinista.

3.3.2 Modelo analítico del memristor

Una versión no lineal del modelo bipolar, particularmente adecuada para aplicaciones neuromórficas, es presentada en el modelo en [37]. Varios dispositivos fabricados han sido alineados con las ecuaciones del modelo propuesto en función de una modificación fina de los parámetros de ajuste. Este modelo se ajusta a una forma generalizada que es caracterizada por una relación $I-V$ no lineal como se expresa en (3.6), donde los parámetros a_1 y a_2 son usados como parámetros de amplitud para incluir la conductividad de las diferentes estructuras del dispositivo junto con la polaridad de la entrada aplicada. Además, se incluye un factor de control donde b calibra la intensidad del umbral en relación con la amplitud del voltaje.

$$I(t)=\begin{cases} a_1 x(t) senh\big(bV(t)\big), & V(t)\geq 0 \\ a_2 x(t) senh\big(bV(t)\big), & V(t)<0 \end{cases} \tag{3.6}$$

$$\frac{dx}{dt}=g\big(V(t)\big)f\big(x(t)\big) \tag{3.7}$$

La segunda ecuación característica (3.7) es el cambio de la variable de estado x con dos funciones responsables de controlar la operación: un factor de imposición de umbral y una función que define las fronteras. La función $g\big(V(t)\big)$ expresada en (3.8), introduce voltajes de umbral en las regiones positivas y negativas V_p y V_n, respectivamente, sin cambios permitidos entre estos dos puntos de límite. Además, la razón de cambio más allá del umbral establecido está controlado por las variables A_p y A_n.

$$g\big(V(t)\big) = \begin{cases} A_p\big(e^{V(t)} - e^{V_p}\big), & V(t) > V_p \\ -A_n\big(e^{-V(t)} - e^{V_n}\big), & V(t) < -V_n \\ 0, & -V_n \leq V(t) \leq V_p \end{cases} \tag{3.8}$$

$$dv_{p/n} = \underbrace{\alpha \cdot \theta\big(v_{p/n_0} - v_{p/n}\big)dt}_{término\ determinista} + \underbrace{\big(V - \delta\mathrm{V} - v_{p/n}\big)dN(\tau)}_{término\ estocástico} \tag{3.9}$$

La incorporación de la estocasticidad en este modelo le agrega un gran valor, ya que es una característica destacada tener un nivel razonable de variabilidad en el funcionamiento de los circuitos neuromórficos [38]. Esto permite tener una función de ruido inherente incorporada en un único elemento que exhibe agilidad e integración en sistemas a gran escala. La ecuación (3.9) muestra la forma matemática de la estocasticidad presente en los umbrales positivo y negativo.

Desde el punto de vista del modelado, abordar la aplicación de la estocasticidad en este modelo requiere un escalamiento del parámetro α_0 como respuesta de la modificación del voltaje de umbral. Como los parámetros de ajuste propuestos están basados en la descripción [39] del dispositivo, donde el voltaje de umbral positivo se ajusta a 1.5 V, que es tres veces menor que la base original. Similarmente, el umbral negativo se ajusta a 0.5 V, que es diez veces menor al valor original. Como se observa en la fig.3.5 los picos abruptos en el voltaje de umbral son una consecuencia directa de la estocasticidad, la cual modifica el umbral del dispositivo en función de la probabilidad de operación y conmutación. Además, esta variación se transmitió directamente en la relación $I - V$ donde el punto de operación es divergente de los deterministas debido al no determinismo inducido del comportamiento subyacente. La fig.3.7 muestra la respuesta determinista y no determinista del modelo.

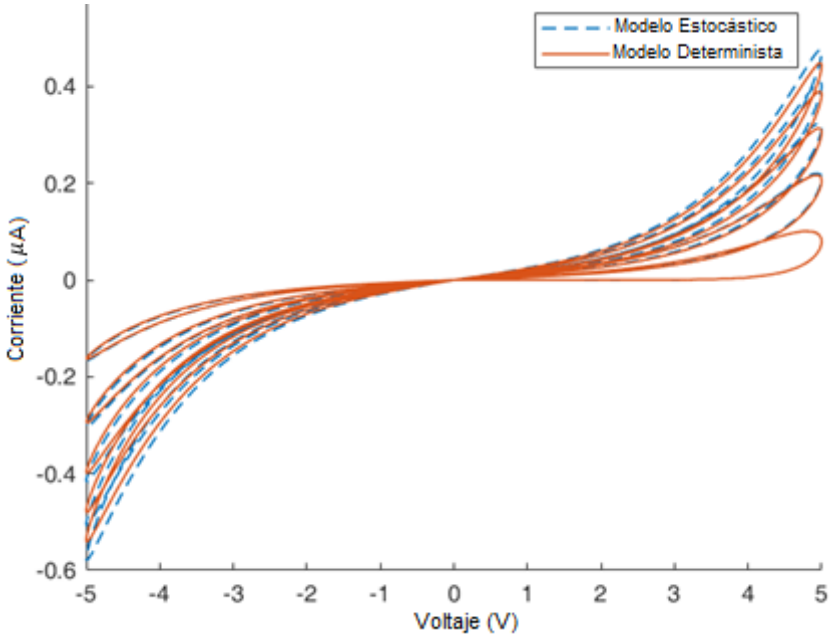


Fig. 3.7. Modelo determinista (rojo) con parámetros de ajuste para el modelo de [39] para una entrada sinusoidal, $V_p = 1.5\,V. V_n = 0.5\,V, A_p = 0.005, A_n = 0.08, x_p = 0.2, x_n = 0.5, \alpha_p = 1.2, \alpha_n = 3, \alpha_1 = 3.7 \times 10^{-7}, \alpha_2 = 4.35 \times 10^{-7}, b = 0.7$ *y* $x_0 = 0.1$. *El modelo estocástico (azul) conserva los mismos parámetros pero se modificó el parámetro de intensidad* α_0 *para que el ajuste de los umbrales positivo y negativo sea adecuado.*

3.3.3 Modelo de Simmons de barrera de túnel

Un modelo de ajuste preciso con las mediciones de datos experimentales, es el modelo de Pickett [40], el cual es una representación no lineal de la conmutación no lineal. El cambio de conductancia está basado en la barrera túnel de Simmons, de ancho *w,* que se encuentra en serie con un resistor interno que afecta de manera acumulativa las características generales del sistema. La corriente aplicada dispara el cambio en el tiempo de la variable de estado *w* debido a los efectos exponenciales sobre la velocidad de los impurificantes ionizados. Esta dependencia conduce a una conmutación asimétrica de encendido a apagado con una difusión iónica altamente susceptible a la polaridad de la entrada. Las ecuaciones generales (3.10) y (3.11) que retratan el comportamiento de este modelo están basadas en las realizaciones físicas logradas para los dispositivos fabricados y se ajustan a través de un extenso análisis de regresión.

$$Conmutación\ OFF\ (i > 0)$$
$$\frac{dw}{dt} = f_{off} senh\left(\frac{i}{i_{off}}\right) \exp\left[-\exp\left(\frac{w - a_{off}}{w_c} - \frac{|i|}{b}\right) - \frac{w}{w_c}\right] \tag{3.10}$$

$$Conmutación\ ON\ (i < 0)$$
$$\frac{dw}{dt} = f_{on} senh\left(\frac{i}{i_{on}}\right) \exp\left[-\exp\left(-\frac{w - a_{on}}{w_c} - \frac{|i|}{b}\right) - \frac{w}{w_c}\right] \tag{3.11}$$

Este modelo impulsado por corriente, tiene valores en las conmutaciones ON y OFF que actúan como factores de límite para el cambio de estado, el cual se considera insignificante por debajo de las condiciones de frontera asignadas. Por otra parte, la razón de cambio es modificada a través de los parámetros de ajuste f_{on} y f_{off}, y la variable de estado x está confinada dentro de sus fronteras a través de los parámetros a_{on} y a_{off} en ambos extremos. La relación $I - V$ no es explícita sino que se basa en el modelo túnel de Simmons [41]

$$i = \frac{j_0 A}{(\Delta w)^2}\left(\phi_1 e^{-B\sqrt{\phi_1}} - \left(\phi_1 + e\left|v_g\right|\right)e^{-B\sqrt{\phi_1 + e|v_g|}}\right) \tag{3.12}$$

$$da_{on/off} = \underbrace{\alpha \cdot \theta\left(a_0 - a_{on/off}\right)dt}_{término\ determinista} + \underbrace{(w - \delta w - a_0)dN(\tau)}_{término\ estocástico} \tag{3.13}$$

Los parámetros en (3.12) se ajustan de acuerdo al modelo de Spice provisto por Pickett [42] donde una resistencia en serie R_s de $215\ \Omega$ junto con la barrera del túnel constituyen la estructura del dispositivo. La estocasticidad en este caso, se incorpora en una manera similar que los modelos anteriores donde la corriente de umbral es el parámetro afectado. La ecuación (3.13) induce la dimensión estocástica en el modelo con una variación establecida en el parámetro a que gobierna los puntos de conmutación del modelo. En una escala más precisa con respecto a los parámetros de ajuste del modelo, la ecuación para la variabilidad es ajustada mediante la multiplicación de la resistencia instantánea y la corriente de entrada para obtener la probabilidad respectiva de conmutación por debajo de los límites de corriente establecidos.

Considerando la diferencia en los umbrales negativo y positivo, las probabilidades son también ajustadas para obtener el impacto de la polaridad. En una forma inversa de la relación $I-V$, el modelo de Simmons de barrera de túnel tiene corrientes de umbral en el rango de μA. Por lo tanto, la modificación instantánea del umbral no se observa fácilmente en la forma de la histéresis. La fig.3.8 muestra este mecanismo.

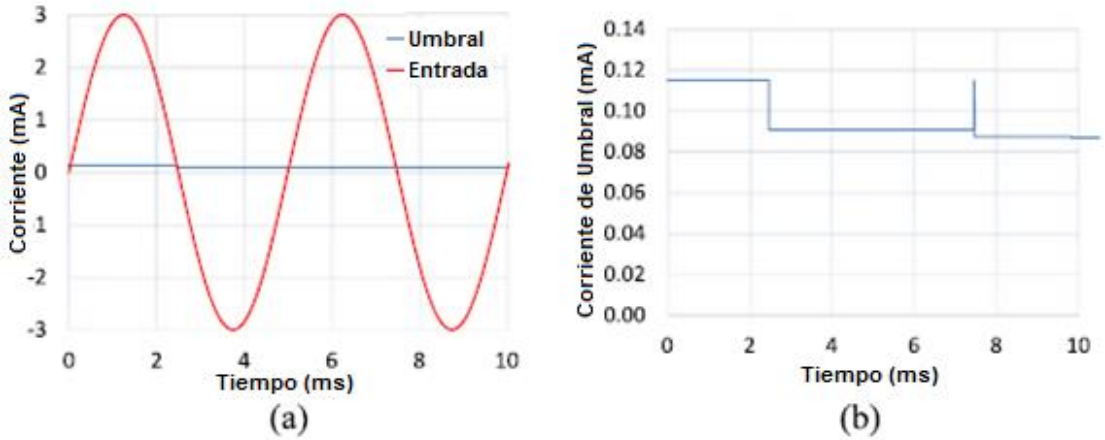


Fig. 3.8. (a) Variación de la corriente de umbral como respuesta a una corriente de entrada con estocasticidad incorporada. (b) Variación de los valores de umbral con una resolución más precisa.

Además, el impacto directo de esta variabilidad en el umbral es una modificación de histéresis añadida que refleja la configuración subyacente. La fig.3.9 muestra el comportamiento determinista y estocástico del modelo. La cinética interna es suave y lenta lo que resulta en una menor variación de la localización de los puntos de conmutación con la conservación del mecanismo general del sistema en términos de operación y dinámica.

Tabla 3.1.Modelos del memristor con estocasticidad incorporada.

Modelo	Ecuación de estado	Variable estocástica
Modelo Bipolar	$\frac{dX}{dt} = f(V_M)\left[\theta(V_M)\theta\left(R_{off} - X\right) + \theta(-V_M)\theta(X - R_{on})\right]$ $f(V_M) = \beta V_M - 0.5\beta[\lvert V_M + V_t\rvert - \lvert V_M - V_t\rvert]$ $\frac{dx}{dt} = g\left(V(t)\right)f\left(x(t)\right)$	Voltaje de umbral V_t
Modelo Analítico	$g\left(V(t)\right) = \begin{cases} A_p\left(e^{V(t)} - e^{V_p}\right), & V(t) > V_p \\ -A_n\left(e^{-V(t)} - e^{V_n}\right), & V(t) < -V_n \\ 0, & -V_n \leq V(t) \leq V_p \end{cases}$	Voltajes de umbral V_p y V_n
Barrera Túnel de Simmons	$\frac{dw}{dt} = f_{off} senh\left(\frac{i}{i_{off}}\right)\exp\left[-\exp\left(\frac{w - a_{off}}{w_c} - \frac{\lvert i\rvert}{b}\right) - \frac{w}{w_c}\right]$ $\frac{dw}{dt} = f_{on} senh\left(\frac{i}{i_{on}}\right)\exp\left[-\exp\left(-\frac{w - a_{on}}{w_c} - \frac{\lvert i\rvert}{b}\right) - \frac{w}{w_c}\right]$	Parámetros de umbral a_{off} y a_{on}

La tabla 3.1 muestra las ecuaciones de estado y las variables estocásticas de cada uno delos modelos presentados anteriormente

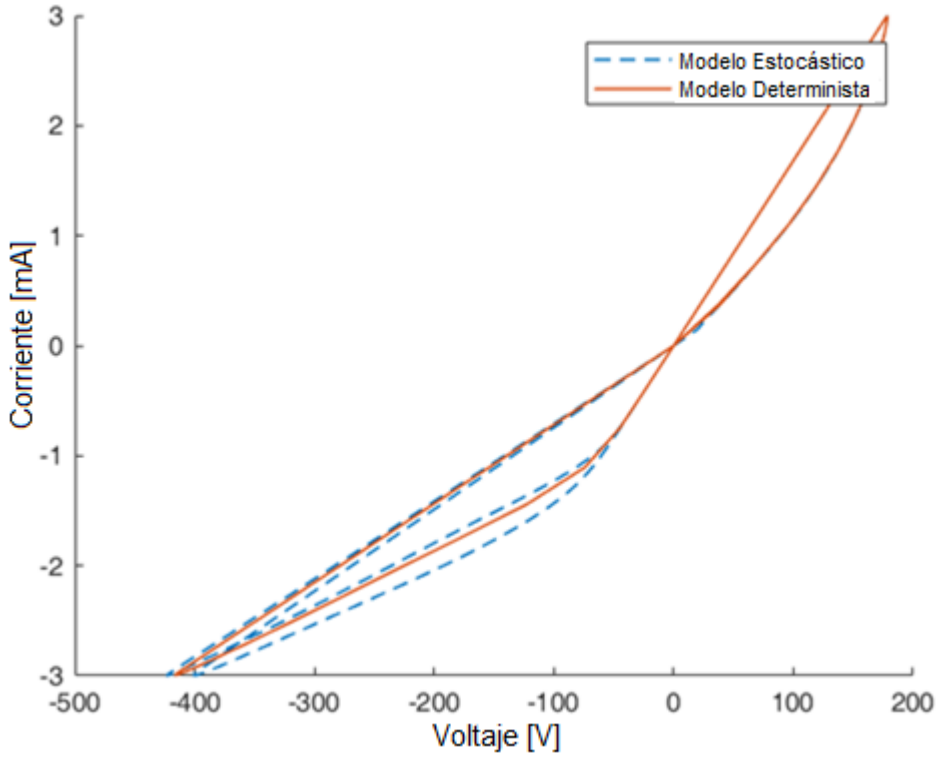


Fig. 3.9. Modelo determinista (rojo) con parámetros $i_{off} = 115\ \mu A, i_{on} = 8.9\ \mu A, a_{on} = 2 \times 10^{-9}, a_{off} = 1.2 \times 10^{-9}, f_{on} = 40 \times 10^{-6}, f_{off} = 3.5 \times 10^{-6}, b = 500 \times e^{-6}, w_c = 107 \times 10^{-12}, D = 3 \times 10^{-9}, R_{on} = 100\ \Omega, R_{off} = 20k\Omega$. *El modelo estocástico (azul) conserva los mismos parámetros pero se modificó el parámetro de intensidad* α_0 *para que el ajuste de los umbrales positivo y negativo sea adecuado.*

3.4 Resumen

En este capítulo se presentó un estudio introductorio de la estocasticidad inducida en una gama de modelos de memristores basados en umbral. Este estudio proporciona una cuantificación de la variabilidad de las características de conmutación entre los dos extremos de la resistencia. Además se presentó una comparación entre la versión determinista y la estocástica que permite observar a detalle el comportamiento de ambas versiones bajo las mismas condiciones de polarización [34].

La importancia de estudiar esta técnica de estocasticidad inducida radica principalmente en la implementación de los memristores en los circuitos neuromórficos como elementos de memoria, más específicamente, las sinapsis. Como será explicado en secciones posteriores, el objetivo de conocer el comportamiento estocástico de una conexión neuronal, es generar un algoritmo apropiado para entrenar la red neuronal y de esta manera modelar a detalle los aspectos más funcionales de la red.

4 DESARROLLO DE LA RED NEURONAL PULSADA

4.1 Introducción

A medida que las computadoras van ocupando todas las áreas del mundo moderno, el software requiere evolucionar en consecuencia, de tal manera que sea transferible a diferentes tipos de hardware, al grado de ser incluso bio-compatible. Por esta razón los esfuerzos recientes de la Inteligencia Artificial proponen que el software pueda ser entrenado y "enseñado" en lugar de solamente codificar secuencias. Con esta motivación, este capítulo expone el desarrollo de la codificación, simulación y prueba en Matlab de una red neuronal pulsada capaz de clasificar dos grupos formados por patrones verticales y patrones horizontales, usando el modelo de Izhikevich en una arquitectura WTA, compuesta por dos pares de neuronas excitatorias e inhibitorias interconectadas lateralmente. Se propone también un método de entrenamiento del tipo STDP para la red neuronal y como un aporte adicional, se formula la implementación en Simulink de dicho método.

4.2 Configuración de la red

La red consta de dos capas, una de entrada y la otra es la capa WTA (ver fig.4.2). La capa de entrada se usa para recibir y codificar los estímulos que se describirán posteriormente. Esta capa está compuesta por 64 neuronas excitatorias, que no tienen ninguna conexión lateral. La capa WTA consiste en dos neuronas excitatorias, que cumplen por un lado la función de integrar las entradas que recibe de la capa de entrada, y por otro lado se inhiben mediante dos neuronas inhibitorias conectadas lateralmente. Cada neurona de entrada i esta conectada con cada neurona excitatoria j de la capa WTA, con pesos sinápticos iniciales $c_{ij} = r^2/64$, donde r es seleccionado aleatoriamente de una distribución uniforme dentro de un intervalo de $[0,0.5]$. En la capa WTA todos los pesos sinápticos de las conexiones laterales se ajustaron a $c_{WTA} = 1$.

La capa de entrada recibe una estimulación proveniente del arreglo de 8x8 a través los patrones de entrada. Dichos patrones son imágenes monocromáticas, que muestran pixeles distribuidos de manera horizontal o vertical como se muestra en la fig.4.1.

Cada pixel puede estimular la neurona de entrada correspondiente, la cual envía un tren de pulsos con una frecuencia $f = 50\ Hz$, con una distribución uniforme.

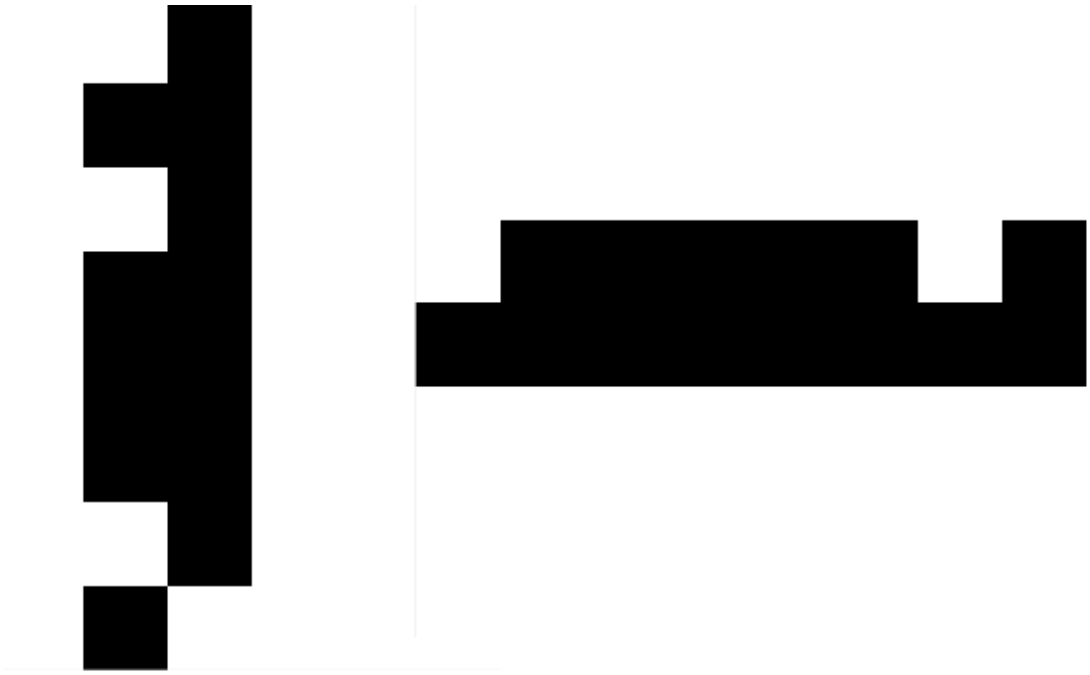


Fig. 4.1.Ejemplos de patrones horizontales y verticales usados para entrenar y posteriormente para probar la red.

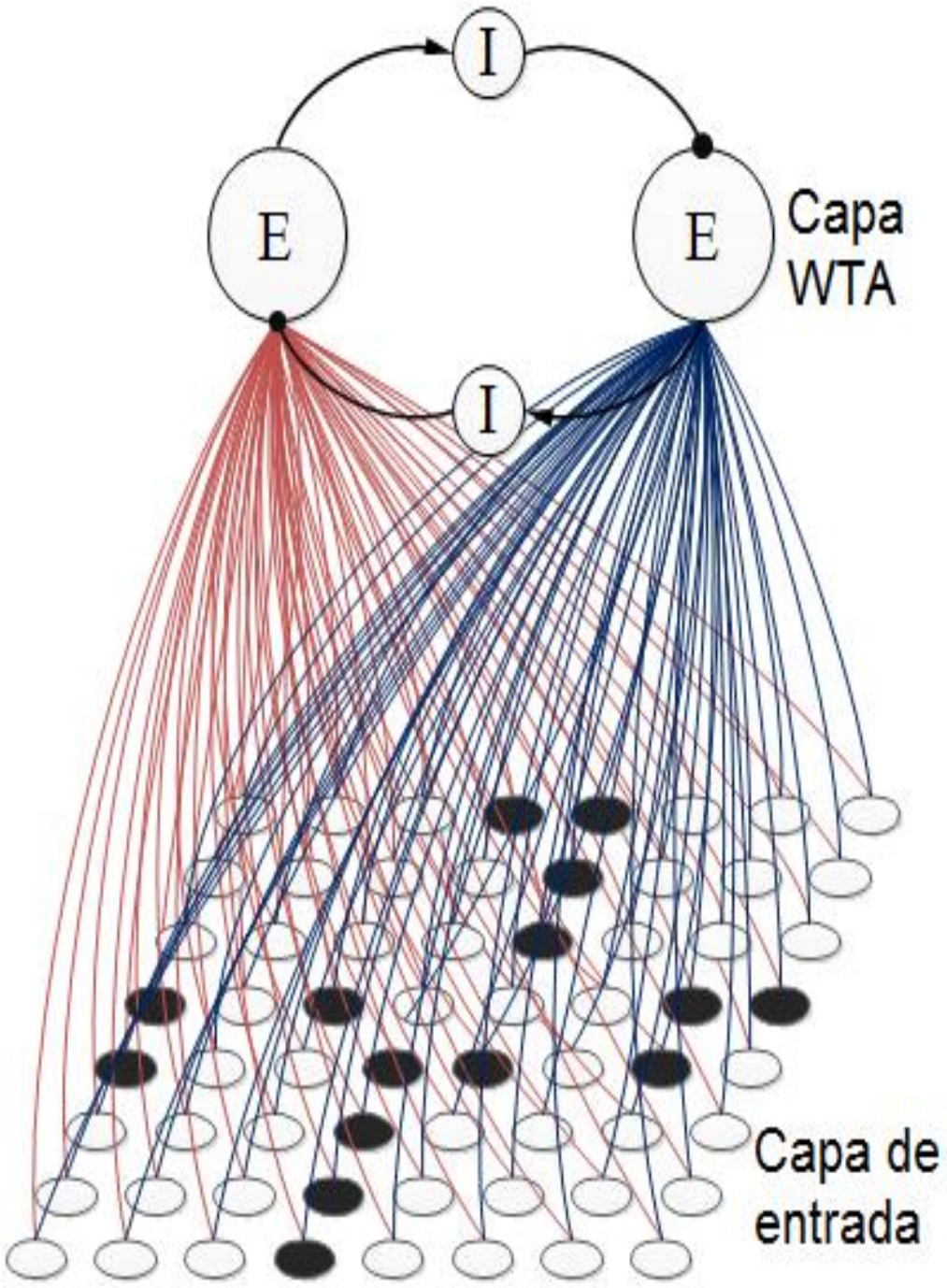


Fig. 4.2. Estructura de la arquitectura WTA propuesta. La capa de entrada es activada a través de un estímulo (círculos negros).

4.3 Simulación en Matlab del modelo de Izhikevich

Como ya se mencionó anteriormente, el modelo matemático usado para simular el comportamiento de las neuronas, es el propuesto por Izhikevich [11]. La decisión de usar dicho modelo radica en la posibilidad de obtener una rápida respuesta computacional así como también la configuración de diferentes comportamientos (familias de neuronas) a través de solo cuatro parámetros. Uno de los requerimientos más importantes para configurar la red con arquitectura WTA es modelar las neuronas excitatorias de tal manera que tengan características integradoras [5]. Las modificaciones a la ecuación (1.17) quedarían de la siguiente manera:

$$v' = 0.04v^2 + 4.1v + 108 - u - I_{syn} \tag{4.1}$$

Para cada paso en la simulación, el tiempo seleccionado fue de 0.1 ms. La fig.4.3 muestra el comportamiento de la ecuación (4.1).

Las constantes a, b, c y d se ajustaron de acuerdo a los criterios obtenidos en las simulaciones de $[12]$. Las neuronas inhibitorias de la capa WTA y las neuronas de la capa de entrada se modelaron de acuerdo a la ecuación (1.17); sin embargo, ambos tipos de neuronas tienen diferentes constantes a, b, c y d. debido a que se espera que cada una tenga un comportamiento específico.

De acuerdo con los requerimientos anteriores, estos fueron los valores seleccionados para las constantes a, b, c y d de los tres tipos de neuronas usadas en la red:

- **Neuronas de entrada:** Neuronas excitatorias (glutamato) con $(a, b) = (0.02, 0.2)$ y $(c, d) = (-65, 8) + (15, -6)r^2$, donde r se escoge aleatoriamente de una distribución uniforme en un intervalo de $[0, 0.5]$. Para la red propuesta se optó por escoger el valor de $r = 0$ ya que permite obtener un comportamiento de pulsaciones regulares (RS) con frecuencia configurable.

- **Neuronas inhibitorias WTA (GABA):** $(a, b) = (0.02{,}0.25)$ y $(c, d) = (-65{,}2)$ para obtener un comportamiento de pulsaciones de umbral bajo (LTS).
- **Neuronas excitatorias WTA:** Neuronas de entrada integradora con $(a, b) = (0.02, -0.1)$ y $(c, d) = (-55{,}6)$.

La fig.4.4 muestra la salida (superior) de una neurona excitatoria de entrada con una frecuencia fijada a 50 Hz. La imagen inferior muestra la salida de la neurona inhibitoria de la capa WTA en respuesta a un escalón de corriente de -13 pA. La neurona excitatoria de entrada tiene $(a, b) = (0.02, 0.2)$ y $(c, d) = (-65{,}8)$.

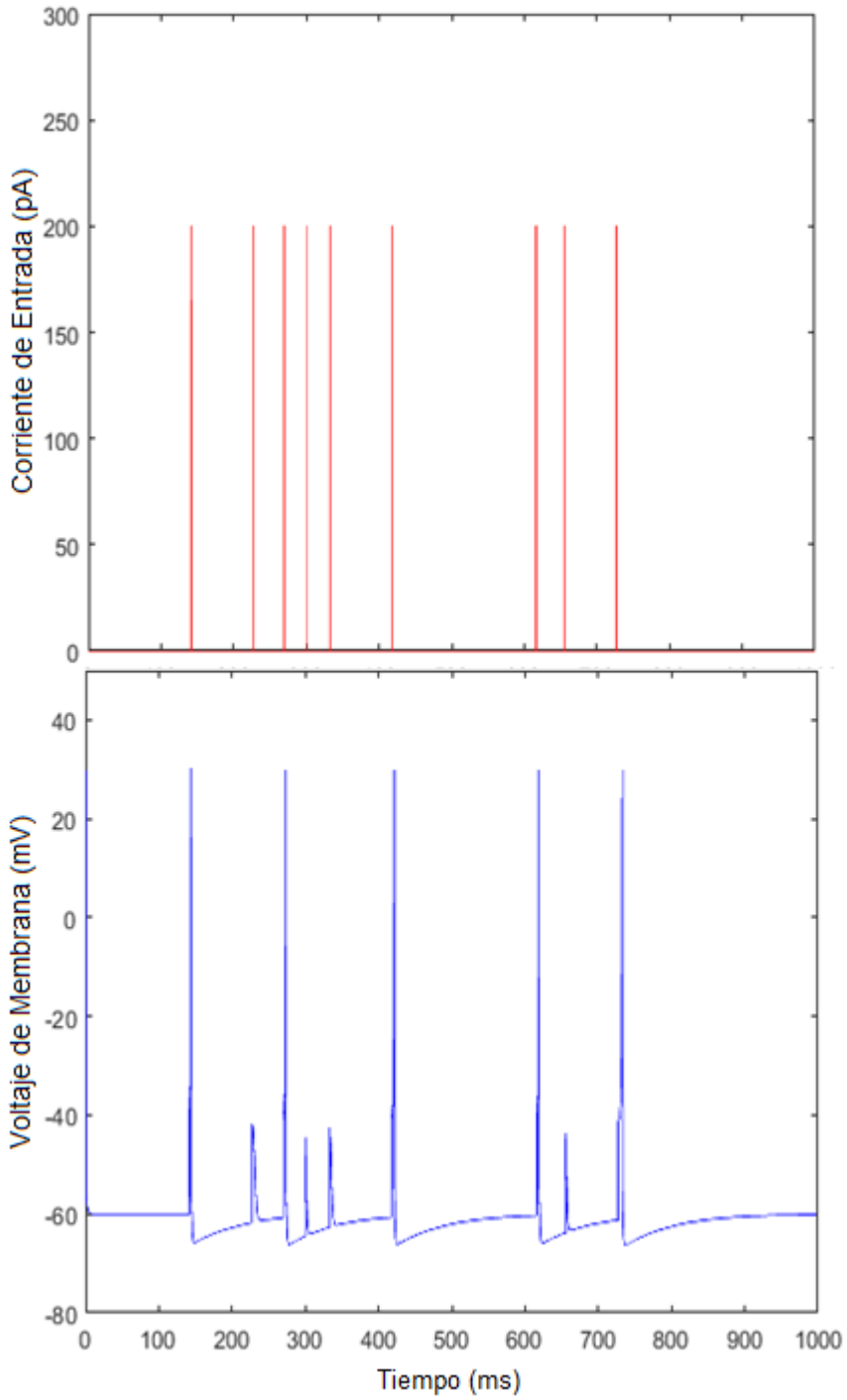


Fig. 4.3. Comportamiento de una neurona excitatoria de entrada integradora modelada con la ecuación (4.1). La gráfica con trazos en rojo muestra la forma de la corriente de entrada y la salida con trazos azules.

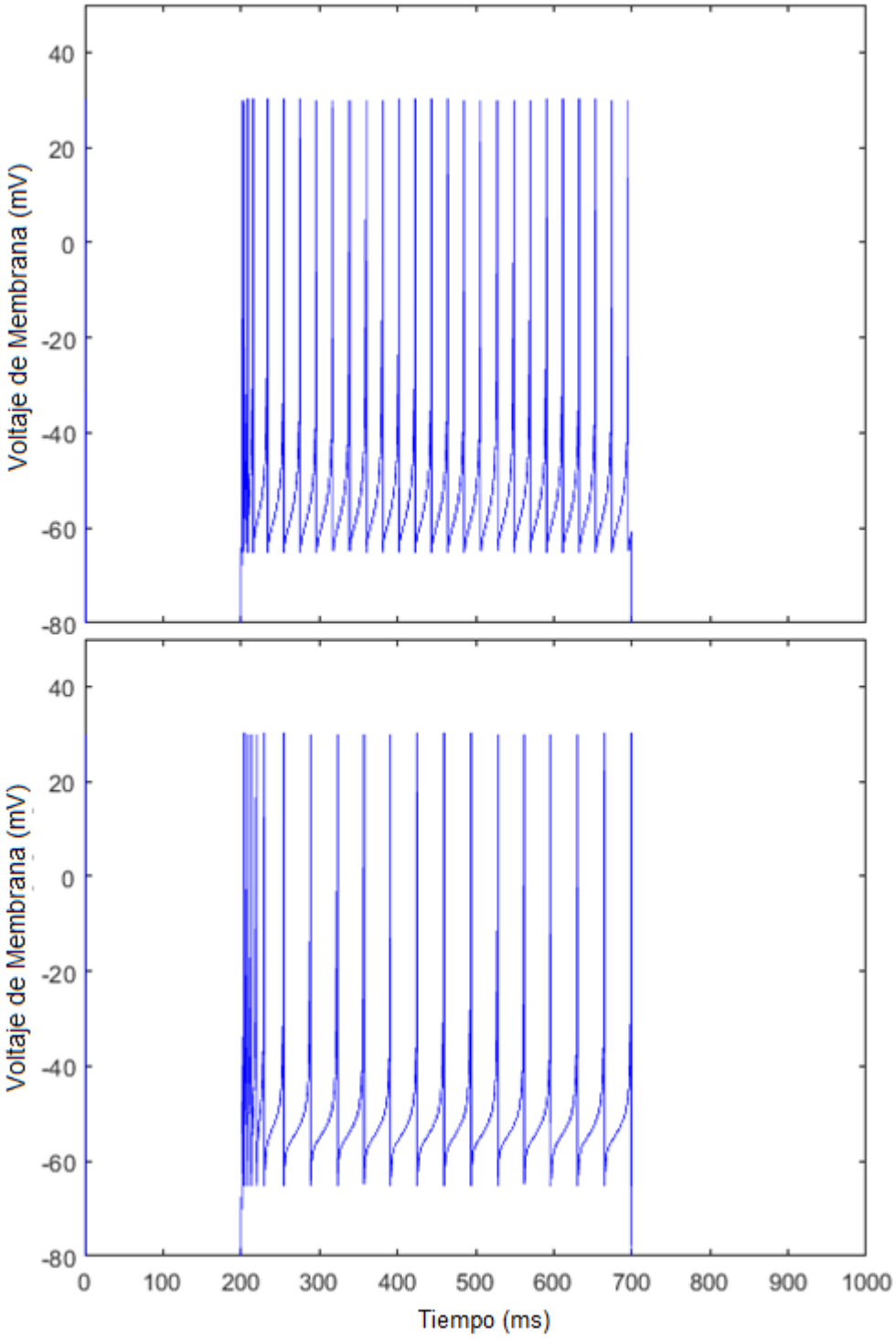


Fig. 4.4.Salida de la neurona excitatoria de entrada con frecuencia fijada a 50 Hz (Superior). Salida de una neurona inhibitoria de la capa WTA (inferior).

4.4 Simulación en Matlab del modelo de sinapsis

Para simular la red neuronal completa, fue necesario simular primero el modelo completo de sinapsis propuesto en [12]. De acuerdo a esta propuesta, la corriente de sinapsis de entrada de una neurona se calcula en cada paso de tiempo con la siguiente expresión:

$$I_{syn} = g_{AMPA}(v_{post} - v_{pre}) + g_{NDMA}\left(\frac{\left[(v_{post} + 80)/60\right]^2}{1 + [(v + 80)/60]^2}\right)(v_{post} - v_{pre}) + g_{GABA_A}(v_{post} + 70) + g_{GABA_B}(v_{post} + 90) \quad (4.2)$$

Con los voltajes medidos en mV y la corriente en pA. v_{post} y v_{pre} representan los voltajes de membrana de la neurona postsináptica y presináptica, respectivamente. Los subíndices que acompañan a g indican el tipo de receptor neuronal. De esta manera la proporción de receptores $NMDA$ a $AMPA$ se estableció para ser uniforme en un valor de 1 para todas las neuronas excitatorias mientras que la proporción de receptores $GABA_A$ a $GABA_B$ también tiene un valor de 1 para las neuronas inhibitorias.

Para simular el cambio de las conductancias se optó por utilizar el modelo de exponencial simple de la ecuación (1.19). La implementación computacional del modelo se hizo a través de una cinética lineal de primer orden:

$$g_k' = -\frac{g_k}{\tau_k} \quad (4.3)$$

La expresión anterior justifica el decaimiento exponencial a través del siguiente desarrollo:

- Si a la ecuación (1.19) se le aplica la primera derivada entonces

$$g_k(t) = e^{-\frac{t}{\tau_k}} \Rightarrow g_k'(t) = -\frac{1}{\tau_k} e^{-\frac{t}{\tau_k}} \Rightarrow g_k'(t) = -\frac{1}{\tau_k} g_k(t)$$

- Se sabe también que

$$g_k' = \frac{g_k(t + T) - g_k(t)}{T}$$

- Por lo tanto

$$\frac{g_k(t+T) - g_k(t)}{T} = -\frac{1}{\tau_k} g_k(t)$$

- Despejando a $g_k(t+T)$ obtenemos

$$g_k(t+T) = \left(1 - \frac{T}{\tau_k}\right) g_k(t) \tag{4.4}$$

Con la ecuación (4.4) se puede calcular el cambio de la conductancia en cada paso T de la ejecución, en este caso es de $0.1\ ms$. La simulación de la ecuación (4.4) se presenta en la fig.4.5. Las constates de tiempo son $\tau_k = 5, 150, 6$ y $150\ ms$ para los receptores $AMPA, NMDA, GABA_A$ y $GABA_B$ respectivamente [12]. Si un pulso se transmite desde una neurona presináptica i hacía una neurona postsináptica j, después de un tiempo de retardo, las conductancias se actualizan dependiendo del tipo de neurona presináptica y el peso sináptico c_{ij}:

$$g_k \leftarrow g_k + c_{ij} \tag{4.5}$$

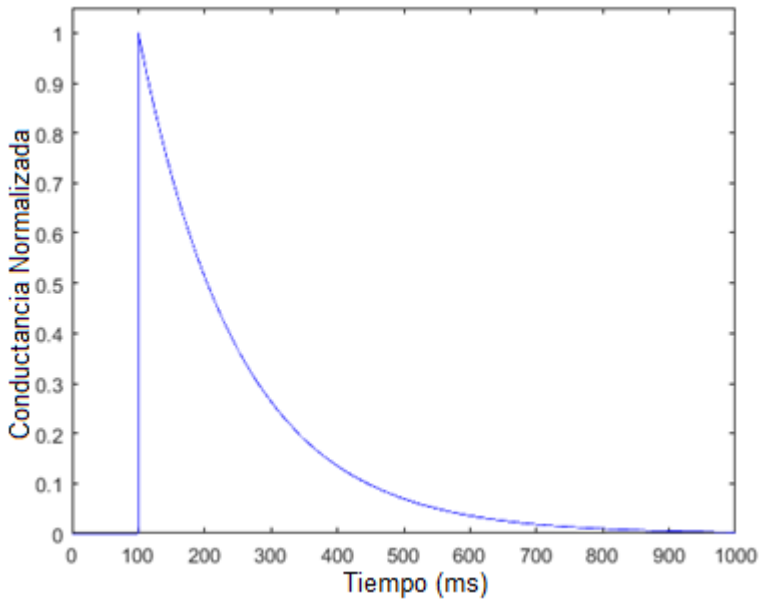


Fig. 4.5. Simulación del modelo de exponencial simple a través de la ecuación (4.4). El paso de la simulación es de $0.1\ ms$. y el valor de la conductancia se normalizó además el tiempo de decaimiento es de $\tau = 150\ ms$.

La simulación del modelo completo de sinapsis se realizó a través de una conexión entre una neurona excitatoria de la capa WTA y una neurona excitatoria de entrada que produce un tren de pulsos normalizados a una frecuencia de $50\ Hz$. Los resultados se presentan en la fig.4.6.

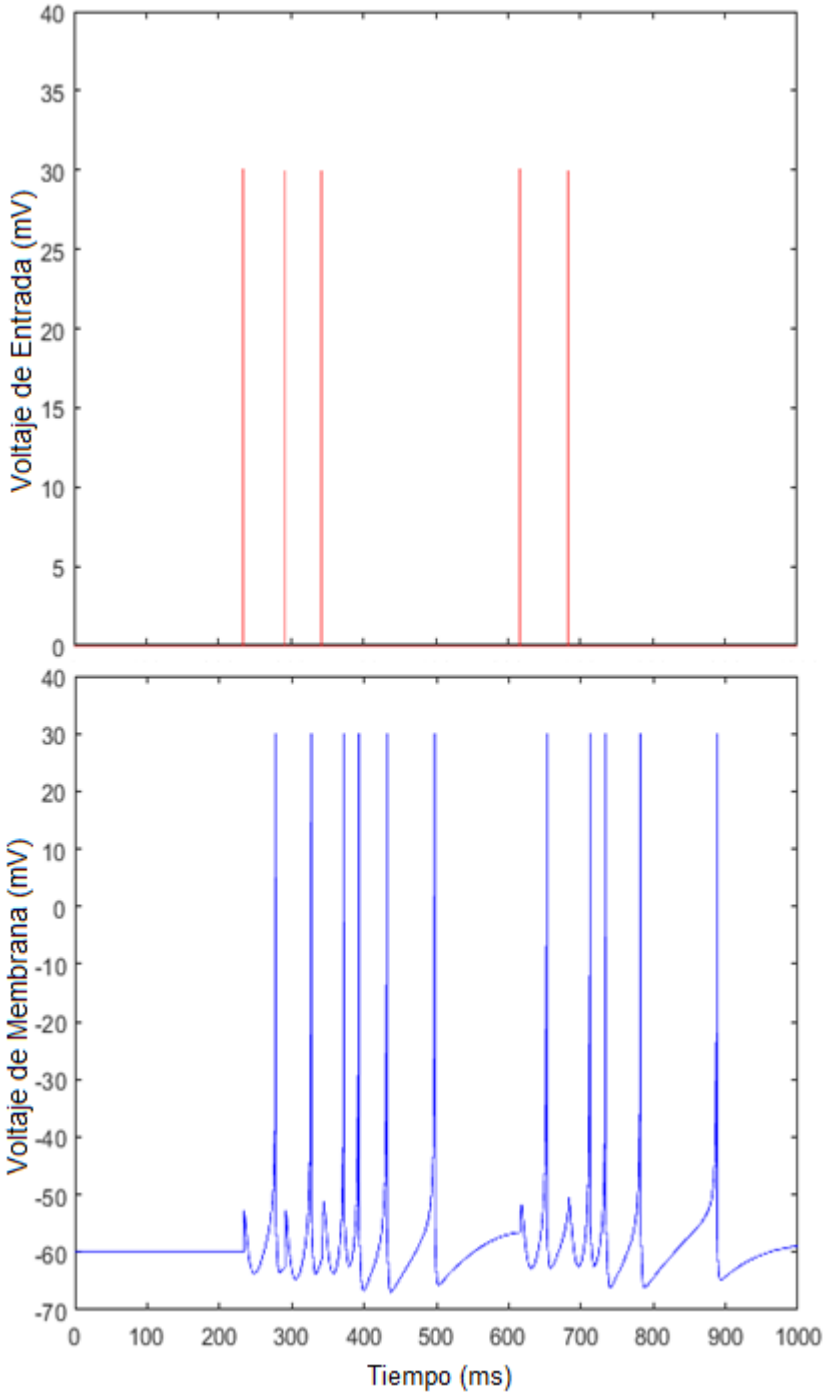


Fig. 4.6. Simulación de una conexión sináptica usando la ecuación (4.2). Los trazos rojos muestran la entrada presináptica y los trazos azules muestran la salida presináptica.

4.5 Implementación del modelo de entrenamiento STDP en Matlab

Como ya se mencionó en el capítulo 2, STDP es una forma temporal que deriva de la regla de Hebb, por lo que es posible usarla para generar un modelo de entrenamiento no supervisado para la red neuronal. La forma general de este modelo es la siguiente

$$F(\Delta t) = \begin{cases} A_+ \exp(\Delta t/\tau_+) \\ A_- \exp(-\Delta t/\tau_-) \end{cases} \tag{4.6}$$

La forma anterior se puede implementar en Matlab usando el método propuesto en [2] el cual tiene como base expresar la potenciación (LTP) y la depresión (LTD) como funciones exponenciales simples. Los cambios en la sinapsis se generan en el modelo a través de N funciones $M_a(t)$ y $P_a(t)$; para $a = 1,2,\dots,N$:

$$\tau_- \frac{dM_a}{dt} = -M_a \tag{4.7}$$

$$\tau_+ \frac{dP_a}{dt} = -P_a \tag{4.8}$$

Cada vez que la neurona postsináptica dispara un potencial de acción, $M_a(t)$ se reduce en una cantidad A_-, y cada vez que la sinapsis a recibe un potencial de acción, $P_a(t)$ se incrementa en una cantidad A_+. $M_t(t)$ se usa para reducir la fuerza sináptica. Si la sinapsis a recibe un potencial de acción presináptico al tiempo t, su parámetro de máxima conductancia se modifica de acuerdo a $\bar{g}_a = \bar{g}_a + M(t)\bar{g}_{max}$. Si esto hace que $\bar{g}_a < 0$, entonces $\bar{g}_a$ se ajusta a cero. $P_a(t)$ se usa para incrementar la fuerza de la sinapsis a. Si la neurona postsináptica dispara un potencial de acción al tiempo t, $\bar{g}_a$ se modifica de acuerdo a $\bar{g}_a \rightarrow \bar{g}_a + P_a(t)\bar{g}_{max}$. Si esto hace que $\bar{g}_a > \bar{g}_{max}$, $\bar{g}_a$ se ajusta a $\bar{g}_{max}$. El parámetro $\bar{g}_{max}$ es un valor de conductancia máximo que evita que la sinapsis se sature y pierda selectividad. Las ecuaciones (4.7) y (4.8) al tener cinética de primer orden, se pueden implementar computacionalmente usando la ecuación (4.4). Todo este procedimiento se simuló a través de dos trenes de pulsos, el primero realiza la función de potenciales de acción presinápticos

y el otro realiza la función de potenciales de acción postsinápticos. La idea principal es generar un barrido de frecuencia de tal manera que la diferencia $t_{pre} - t_{post}$ sea variable en incrementos de $10\,ms$ y de esta forma poder visualizar la curva característica de STDP. Este procedimiento se puede observar en la fig.4.7 y la curva característica obtenida con esta implementación se observa en la fig.4.8. Para esta simulación los valores de los parámetros usados fueron: $A_+ = 0.005,\ A_- = 0.009,\ \tau_+ = 15\,ms,\ \tau_- = 20\,ms$. Estos valores pueden elegirse sin ningún criterio específico, solamente se debe cumplir $A_+\tau_+ \leq A_-\tau_-$. Esto asegura que los pulsos pre- y postsinápticos no correlacionados conduzcan a un debilitamiento sináptico general sin llegar a la saturación.

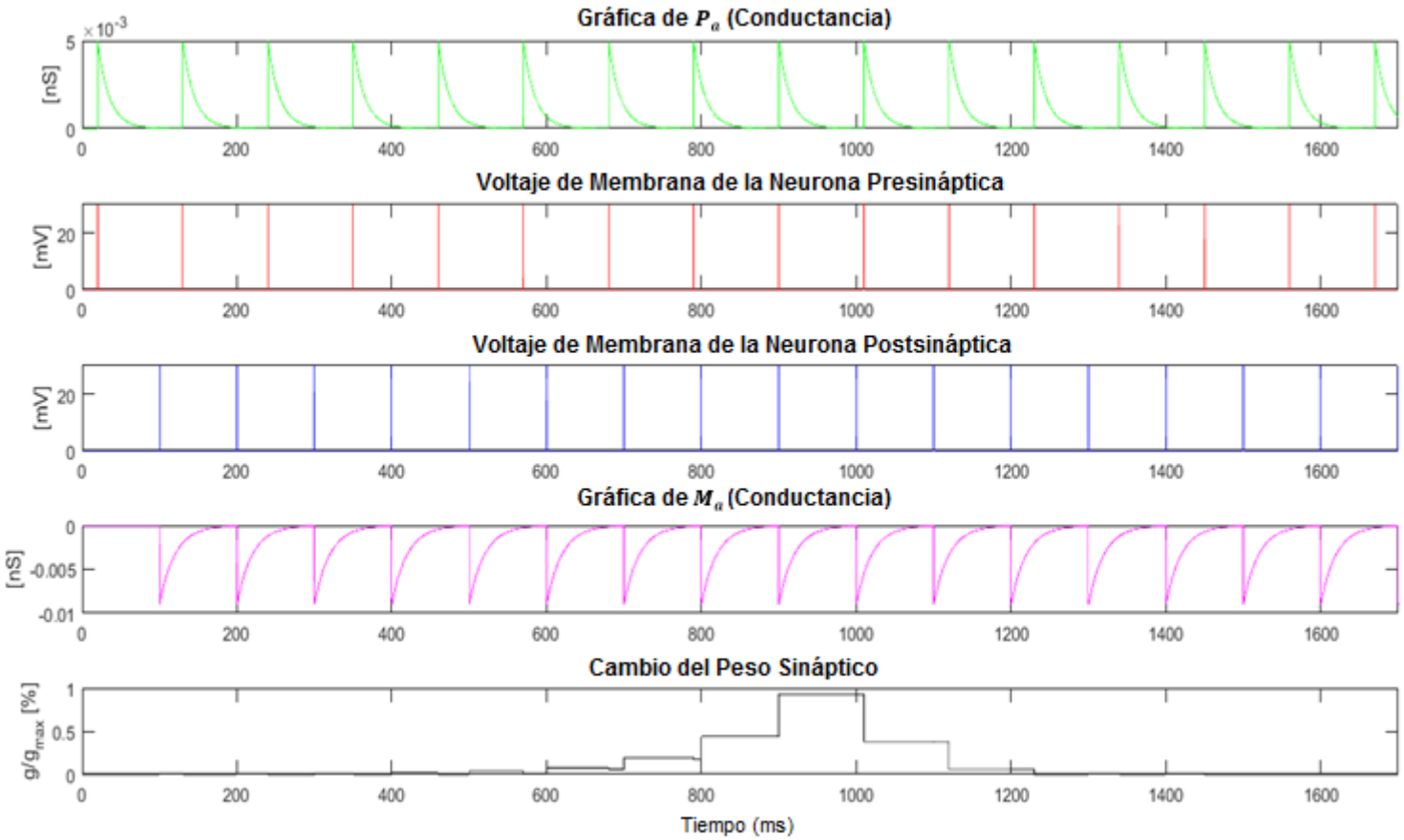


Fig. 4.7. Gráficas obtenidas de la implementación del entrenamiento STDP usando método propuesto en [2]. El trazo rojo muestra los pulsos generados por la neurona presináptica mientras que el trazo azul muestra los pulsos generados por la neurona postsináptica. La diferencia $t_{pre} - t_{post}$ es variable con incrementos de $10\ ms$. El cambio del peso sináptico aparece con un trazo negro en la gráfica inferior.

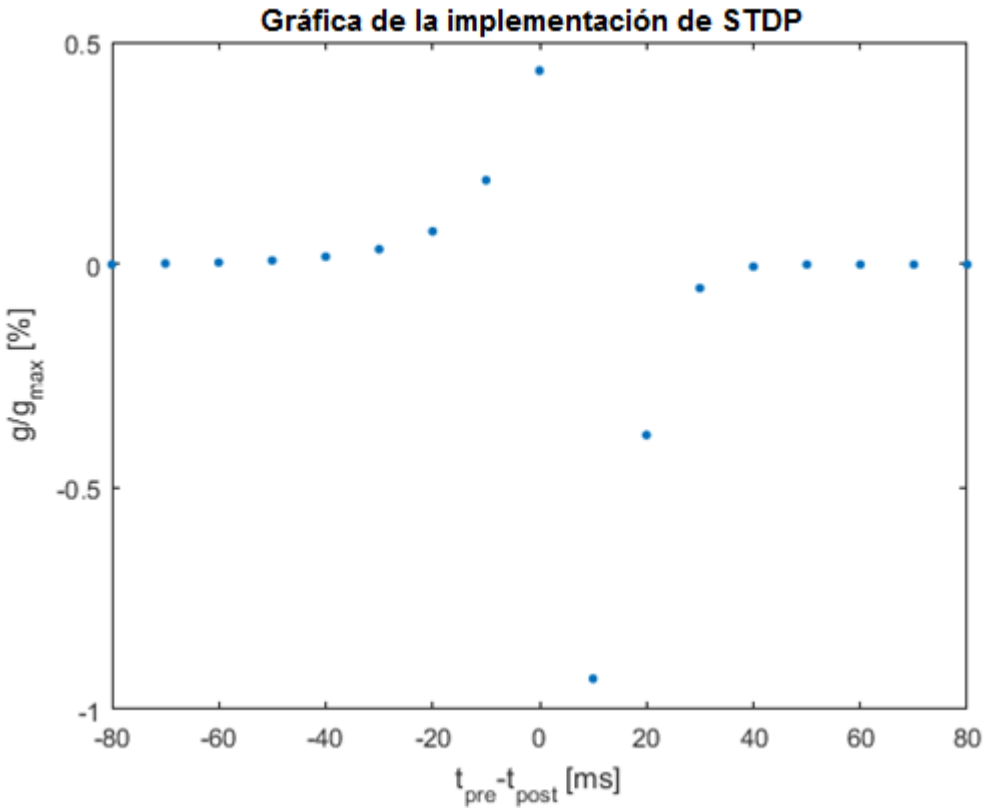


Fig. 4.8. Curva característica STDP generada por el método descrito anteriormente. Los puntos azules fueron obtenidos del cambio del peso sináptico que se muestra en la gráfica inferior de la fig.4.7.

4.6 Procedimiento para entrenar la red neuronal

Para entrenar la red neuronal a través de STDP fue necesario desarrollar un algoritmo que permitiera mostrar conjuntos de patrones de entrada, tanto horizontales como verticales y actualizar los pesos sinápticos de cada una de las sinapsis (64 por cada neurona excitatoria) a través de la función de la ecuación (4.6). Para generar lo patrones de entrada de tal manera que todos fueran diferentes entre sí, se implementó un algoritmo que genera dos vectores con valores binarios (1's y 0's) distribuidos aleatoriamente. Los 0's representan un pixel completamente negro y los 1's representan un pixel completamente blanco. Entonces por ejemplo si el vector 1 tiene la forma "00001110" y el vector 2 tiene la forma "01011000" el patrón generado será como alguno de los mostrados en la fig.4.9. Ambos vectores se pueden usar

para formar patrones horizontales y verticales por lo que solo bastó con generar una vez cada vector por cada patrón.

Fig. 4.9. Posibles patrones formados con el vector "00001110" *y el vector* "01011000"*.*

Estos vectores son irrepetibles y en el algoritmo se limitó a que la cantidad de ceros fueran 5, 6, 7 y 8, de esta manera la cantidad de combinaciones posibles por vector es de 92, lo que nos da un total de 8464 patrones posibles sin ninguna repetición. Al ser un número bastante grande, solamente se ocuparon 5000 patrones (5000 horizontales y 5000 verticales) para entrenar a la red.

Los patrones de entrenamiento se presentaban a la red de manera intercalada, esto es, primero un patrón horizontal y después un patrón vertical. Cada uno se presentó con una duración de 1 segundo, por lo que el tiempo total de entrenamiento fue de 10000 segundos (tiempo simulado, no tiempo real). Para probar la red se agruparon 35 patrones totalmente distintos a los generados en el algoritmo (ver fig.4.10) que incluían algunos patrones que no pertenecían a ninguna de los dos clases (horizontal y vertical) y servían para probar el desempeño de la red.

Otra configuración importante fue la inicialización de los pesos sinápticos de acuerdo a la regla $c_{ij} = r^2/64$, como ya se mencionó en la sección 4.2. Esta configuración se puede apreciar en la fig.4.11.

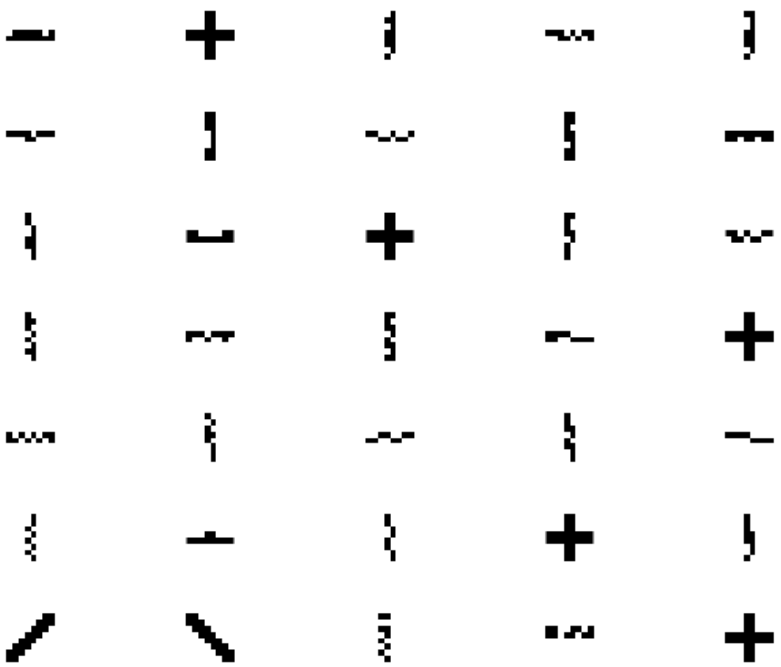


Fig. 4.10. Patrones de prueba para la red. Se pueden apreciar algunas configuraciones que no pertenecen a la clase horizontal o a la clase vertical.

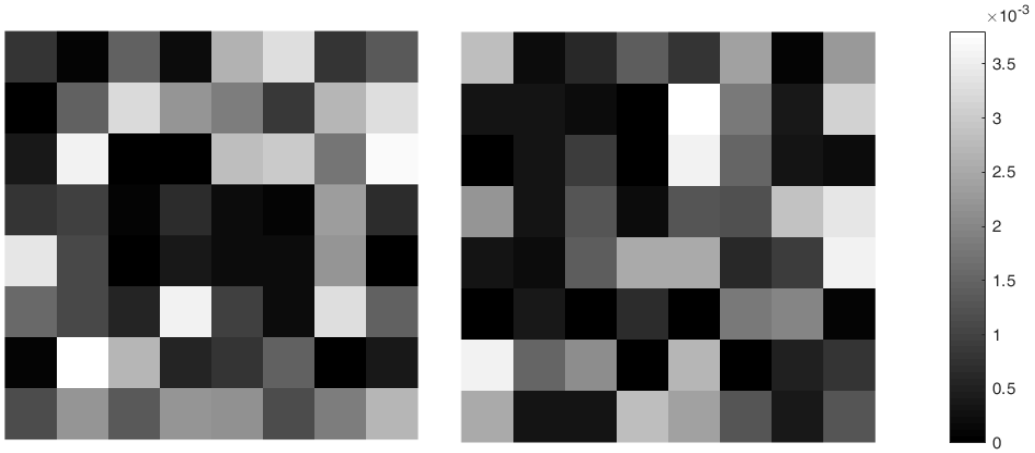


Fig. 4.11. Inicialización de los pesos sinápticos en $t = 0$ de acuerdo a la regla $c_{ij} = r^2/64$ donde r es un número aleatorio seleccionado de una distribución uniforme en un intervalo de $[0, 0.5]$

4.7 Implementación en Simulink del modelo de STDP

Simulink, al ser un programa de simulación basada en modelos, permite codificar en bloques funcionales las instrucciones de Matlab del método de la sección 4.5 usado para simular el modelo de STDP. El modelo de Simulink para esta simulación consta de cuatro secciones fundamentales. La primera sección es la entrada de datos, los cuales están almacenados en cinco vectores: pulsos presinápticos, pulsos postsinápticos, función M_a, función P_a y el cambio de peso $g\ /\ g_{max}$. Tanto los pulsos presinápticos como los pulsos postsinápticos, fueron generados en Matlab para ser usados como señal de entrada. Las demás señales fueron extraídas del modelo simulado en Matlab con la finalidad de poder comparar los resultados obtenidos en Simulink. La segunda sección genera las señales de las funciones M_a y P_a. La tercera sección contiene un módulo que se encarga de actualizar el cambio del peso sináptico en términos de las funciones M_a y P_a y la llegada de los pulsos presinápticos y postsinápticos. La cuarta y última sección muestra simultáneamente las señales generadas por el modelo de Simulink y el modelo de Matlab, esto con la finalidad de realizar una comparación entre las señales. La fig.4.12 muestra el diagrama a bloques del modelo, en conjunto con la fig.4.13 que muestra el diagrama a bloques del sub-módulo de actualización de peso sináptico y la fig.4.14 muestra las señales generadas por el modelo.

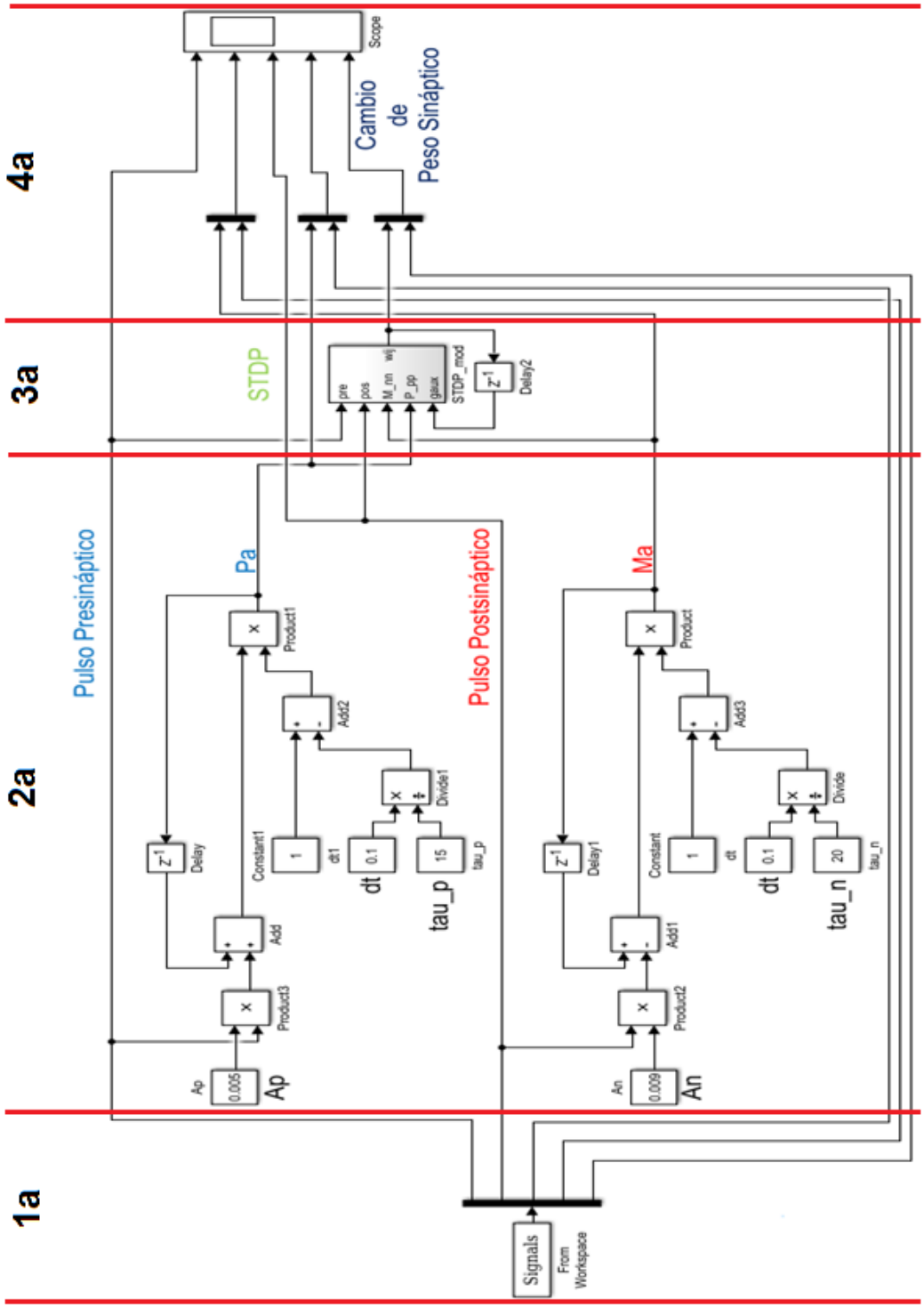


Fig. 4.12. Diagrama a bloques del modelo de STDP (leer de izquierda a derecha).

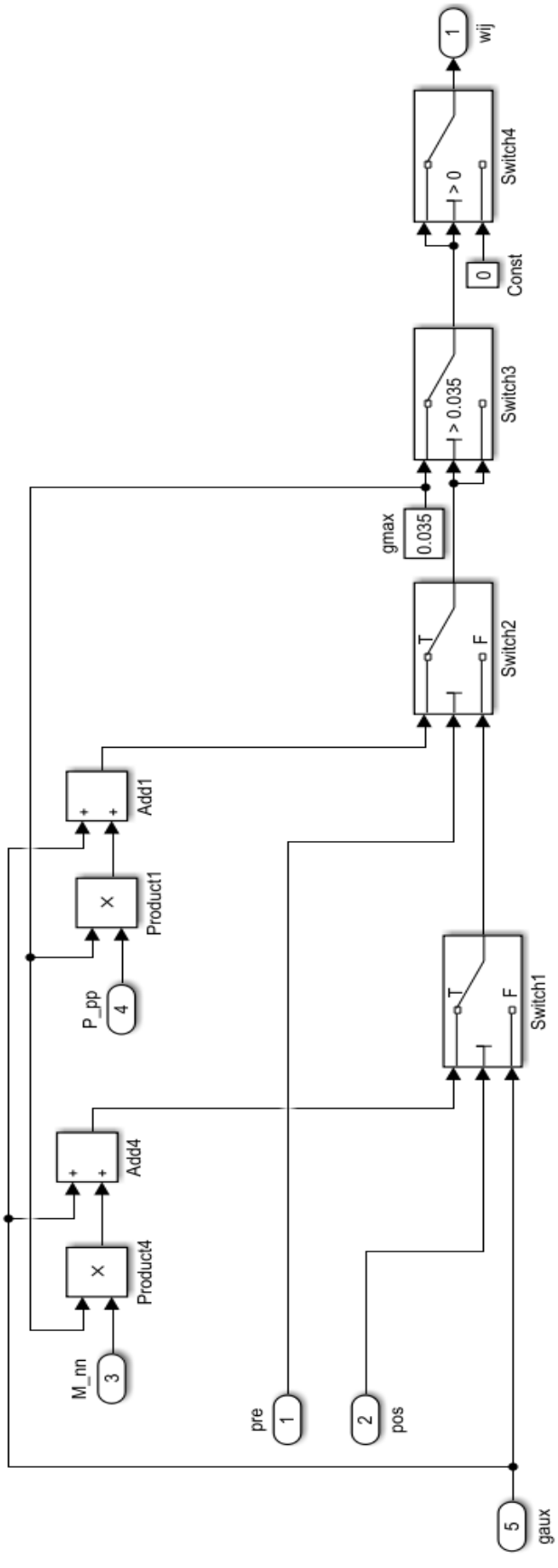


Fig. 4.13. Diagrama a bloques del módulo encargado de actualizar el cambio del peso sináptico en función de la diferencia $t_{pre} - t_{pos}$.

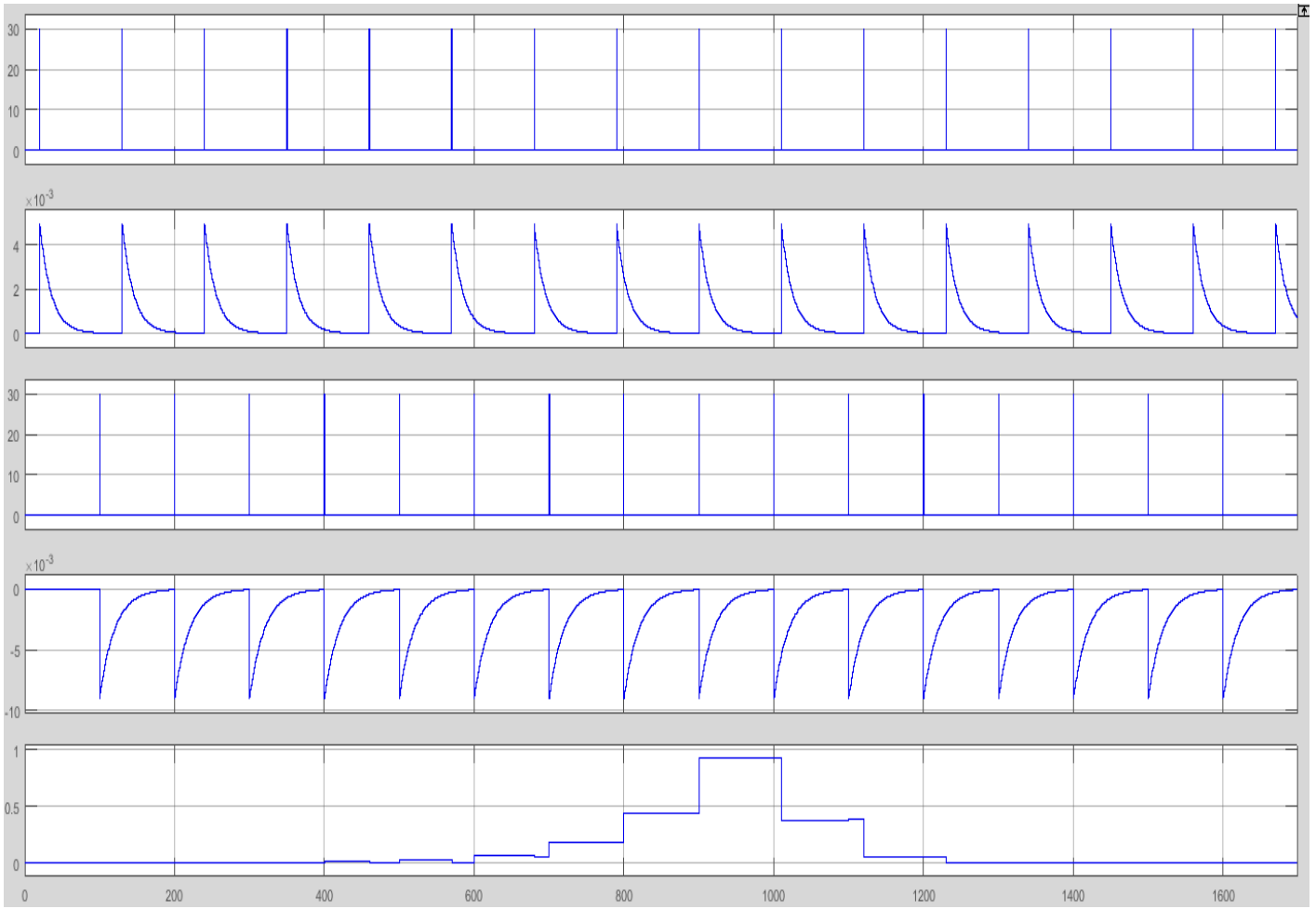


Fig. 4.14. Señales de salida generadas por el modelo de STDP en Simulink. La primera señal (superior) son los pulsos presinápticos, la segunda es la función P_a, la tercera son los pulsos postsinápticos, la cuarta es la función M_a y la quinta es el cambio del peso sináptico g/g_{max}.

4.8 Desarrollo de la simulación de los tres modelos estocásticos del memristor (Biolek, Yakopcic y Pickett)

La simulación de los tres modelos se realizó a través de la descripción de un circuito en Verilog-A. Esto permitió modelar la condición de estocasticidad, la cual es el parámetro más importante para la simulación de los tres modelos propuestos. Los módulos para la simulación se implementaron en la herramienta de diseño Pyxis Layout, desarrollada por Mentor Graphics. Pyxis Layout permite generar un esquemático a partir de un modelo descrito en Verilog-A. Se optó por usar este software debido a que en conjunto con otras herramientas adicionales, se podría comenzar el diseño *custom/semi-custom* de un circuito integrado a muy alta escala de integración (VLSI).

La idea central que se toma como punto de partida para crear el algoritmo de estocasticidad es que la variabilidad se basa en el cambio en el punto de conmutación del memristor, ya que su valor particular depende de la probabilidad calculada en cada paso de tiempo. Acumulativamente la operación completa sigue una distribución de Poisson/Log-Normal, mientras que los eventos de conmutación correspondientes, son aleatorios en el tiempo y se deciden en cada instante, como se presenta en la fig.4.15. Es decir, en cada instante de tiempo, la conmutación puede o no ocurrir arbitrariamente.

```
Seleccionar la distribución de la conmutación;
Calcular los parámetros característicos;
Para cada instante de tiempo hacer
        Calcular la probabilidad correspondiente P;
        Tomar una muestra de la distribución aleatoria uniforme R;
        Comparar;
        Si P>=R entonces
                El voltaje (o corriente) de umbral se ajusta al voltaje (o
                corriente) actual;
        de otro modo
                El voltaje (o corriente) de umbral no cambia;
Manda el voltaje umbral;
```

Fig. 4.15. Algoritmo de variación estadística del voltaje (o corriente) de umbral.

El concepto principal del algoritmo de la fig.4.15 radica en tomar una muestra de la distribución aleatoria uniforme y compararla con la probabilidad calculada de conmutación en cada punto de tiempo. En caso de que la probabilidad calculada sea mayor que la muestra escogida, el voltaje (o corriente) de umbral toma el valor del voltaje (o corriente) instantáneo y en consecuencia, se produce la conmutación. Si esto no ocurre, el voltaje (o corriente) de umbral permanece en el caso casi determinista establecido para el dispositivo particular. El algoritmo completo codificado en Verilog-A se puede encontrar en el apéndice B. Las simulaciones obtenidas en Pyxis, se pueden observar con detalle en la sección de “Resultados generales”.

Algunos parámetros de configuración, comunes en los tres modelos, se presentan en la tabla 4.1.

Modelo	**Tipo de fuente de alimentación**	**Amplitud**	**Frecuencia**	**Paso de tiempo para el análisis transitorio en la simulación**	**Duración de la simulación**	**Paso de tiempo para el análisis transitorio en el modelo**
Biolek	Voltaje Senoidal	$4.75\ V$	$200\ kHz$	$0.01\ ns$	$30\ \mu s$	$1\ ns$
Yakopcic	Voltaje Senoidal	$5\ V$	$5\ Hz$	$0.01\ s$	$1\ s$	$0.001\ s$
Pickett	Corriente Senoidal	$0.003\ A$	$200\ Hz$	$500\ ps$	$20\ ms$	$10\ ns$

Tabla 4.1. Parámetros de configuración para la simulación de los tres modelos estocásticos del memristor.

4.9 Conclusiones

La simulación numérica de redes de gran escala de neuronas pulsadas y la implementación en tiempo real de sistemas neuromórficos requieren una potencia de cálculo significativa. Por otra parte, los microprocesadores de propósito general existentes, aunque son extremadamente versátiles, se basan principalmente en procesamiento en serie de datos, lo que limita severamente su rendimiento computacional. Sin embargo, el uso de técnicas de programación especializadas permite sintetizar algoritmos que aprovechan esta deficiencia de procesamiento serial. Un ejemplo es el que se muestra en este capítulo, el cual describe una implementación de una red neuronal pulsada cuya principal característica, es que la velocidad de procesamiento de información depende únicamente de la frecuencia de la llegada de los estímulos, mas no de su intensidad o forma. Asimismo, se presenta un desarrollo computacional en Simulink que en conjunto con el modelado estocástico de memristores, se podrá realizar para una síntesis eficiente para sistemas embebidos usando FPGAs.

5 RESULTADOS GENERALES

5.1 Introducción

Este capítulo tiene como objetivo presentar los resultados obtenidos del entrenamiento, las pruebas y el rendimiento de la red neuronal pulsada en conjunto con las simulaciones en Pyxis Layout obtenidas de los tres modelos estocásticos del memristor descritos en el capítulo 3.

5.2 Entrenamiento de la red neuronal pulsada

Como ya se mencionó en la sección 4.6, la técnica de entrenamiento consistió en presentar a la entrada de la red, 5000 patrones horizontales diferentes y 5000 patrones verticales diferentes, de forma intercalada (uno vertical después de uno horizontal) en una ventana de tiempo de $1\,s$ por cada patrón. La modificación de los pesos sinápticos se hizo a través de STDP. Al $t = 0$ los pesos sinápticos se encontraban con los valores mostrados en la fig. 4.11.

Para verificar la evolución de los pesos sinápticos a lo largo de las épocas de entrenamiento, se tomaron tres muestras a diferentes tiempos: $500\,s, 2000s$ y $5000\,s,$ tanto de los pesos sinápticos para patrones horizontales, como de los pesos sinápticos para patrones verticales. Estos resultados se muestran de la fig.5.1 a la fig.5.6. Los pesos sinápticos mostrados en la fig.5.6 son los que se usaron en las etapas de prueba y rendimiento de la red.

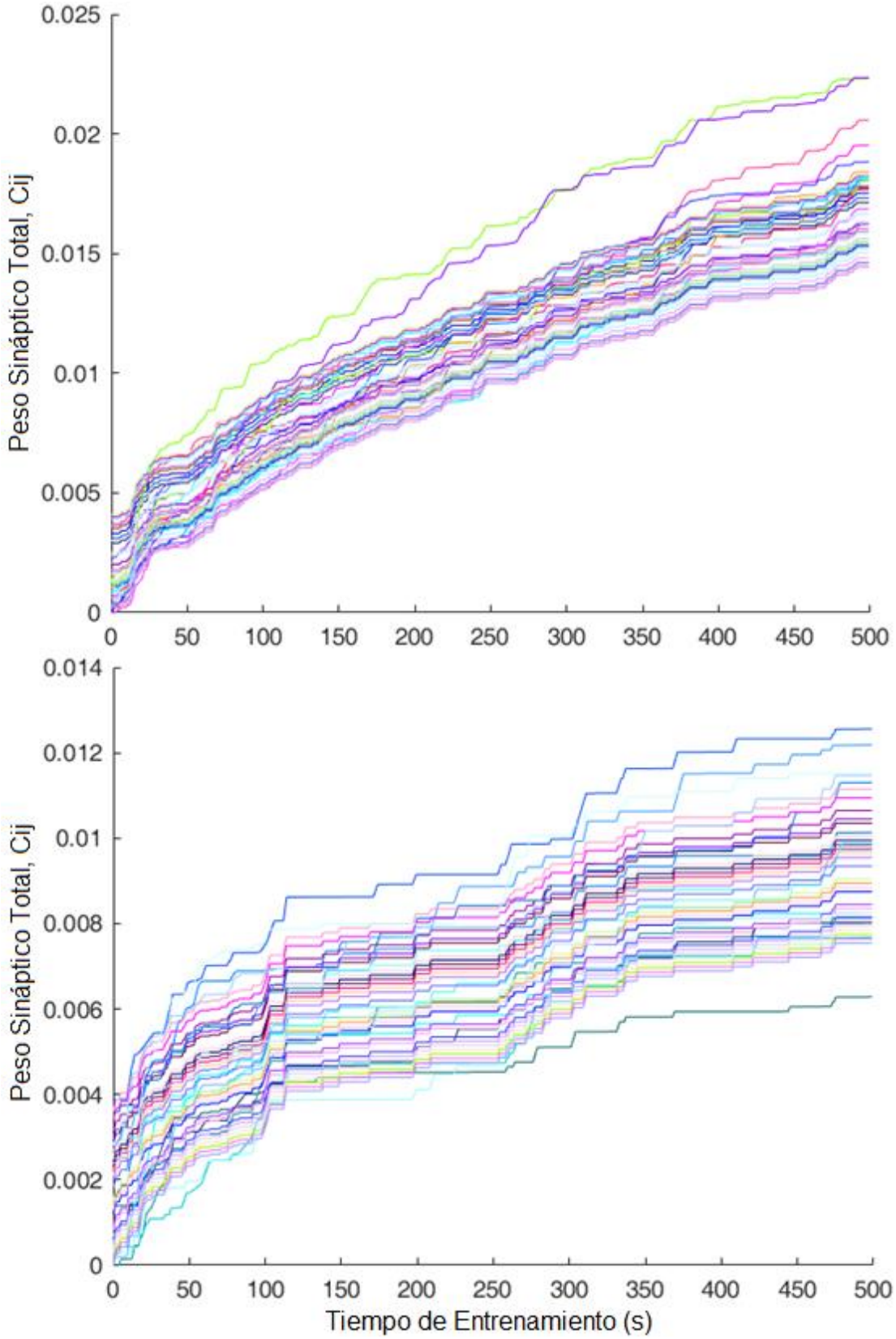
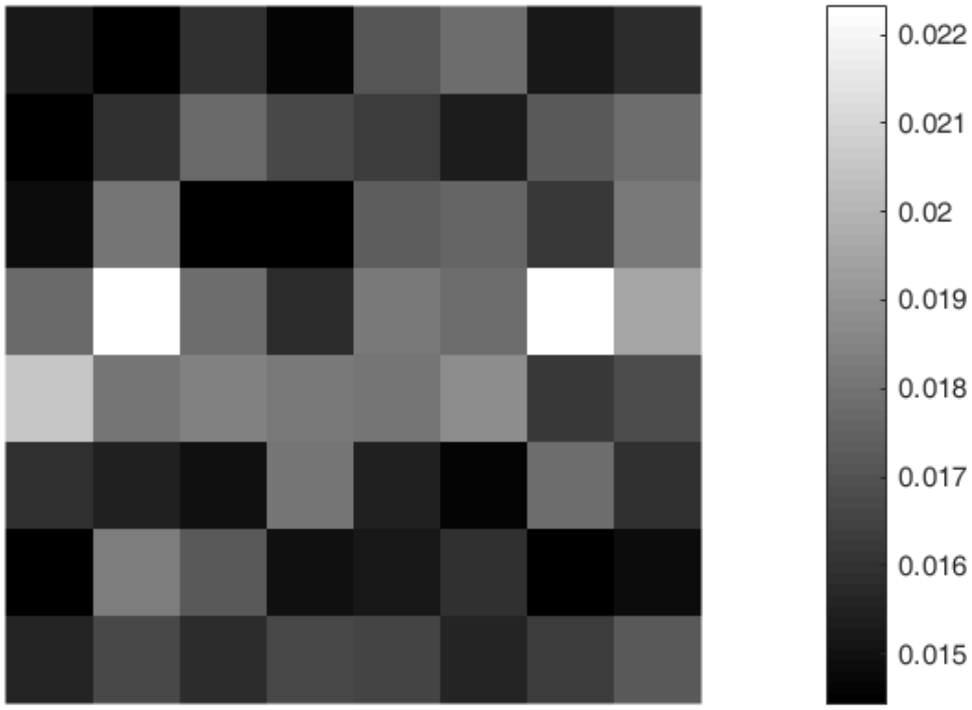


Fig. 5.1.Evolución de los pesos sinápticos para patrones horizontales (superior) y para patrones verticales (inferior) después de 500 segundos de entrenamiento.



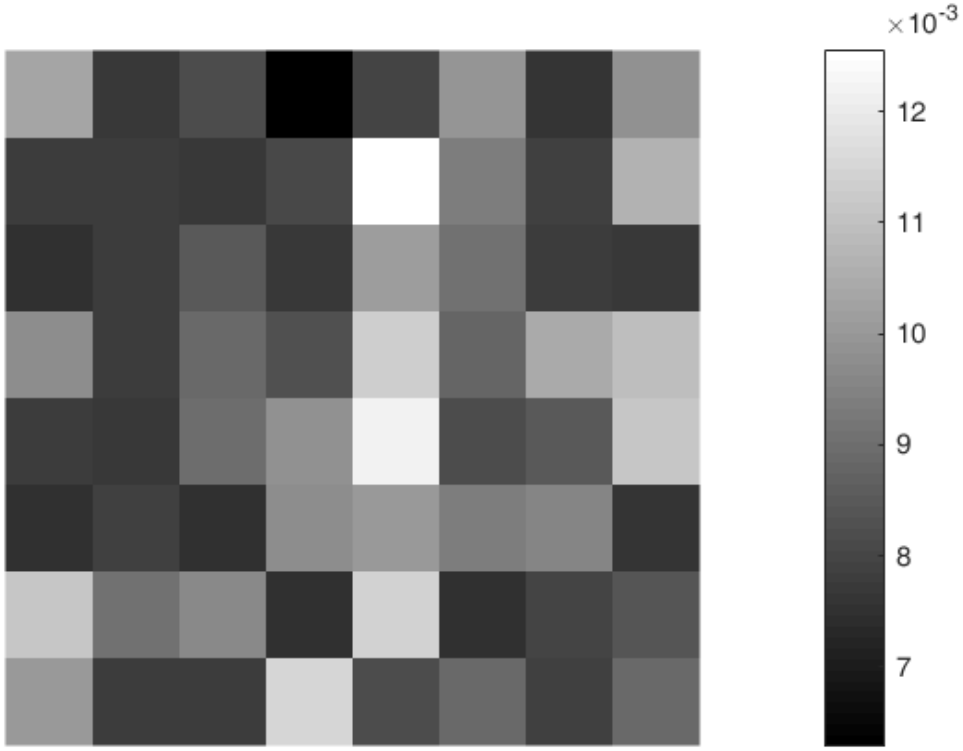


Fig. 5.2. Representación numérica de los pesos sinápticos para patrones horizontales (superior) y para patrones verticales (inferior) después de 500 segundos de entrenamiento.

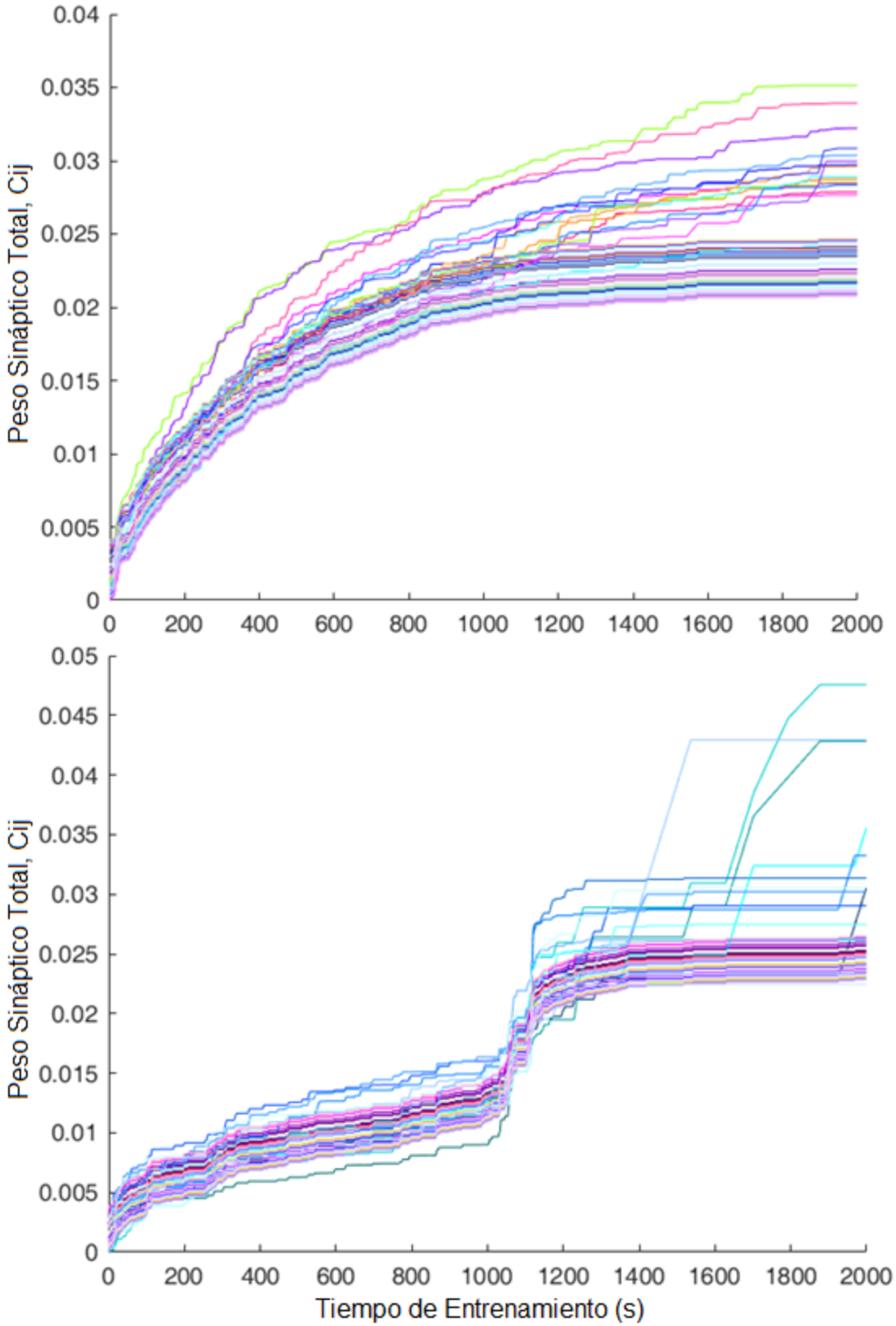


Fig. 5.3. Evolución de los pesos sinápticos para patrones horizontales (superior) y para patrones verticales (inferior) después de 2000 segundos de entrenamiento.

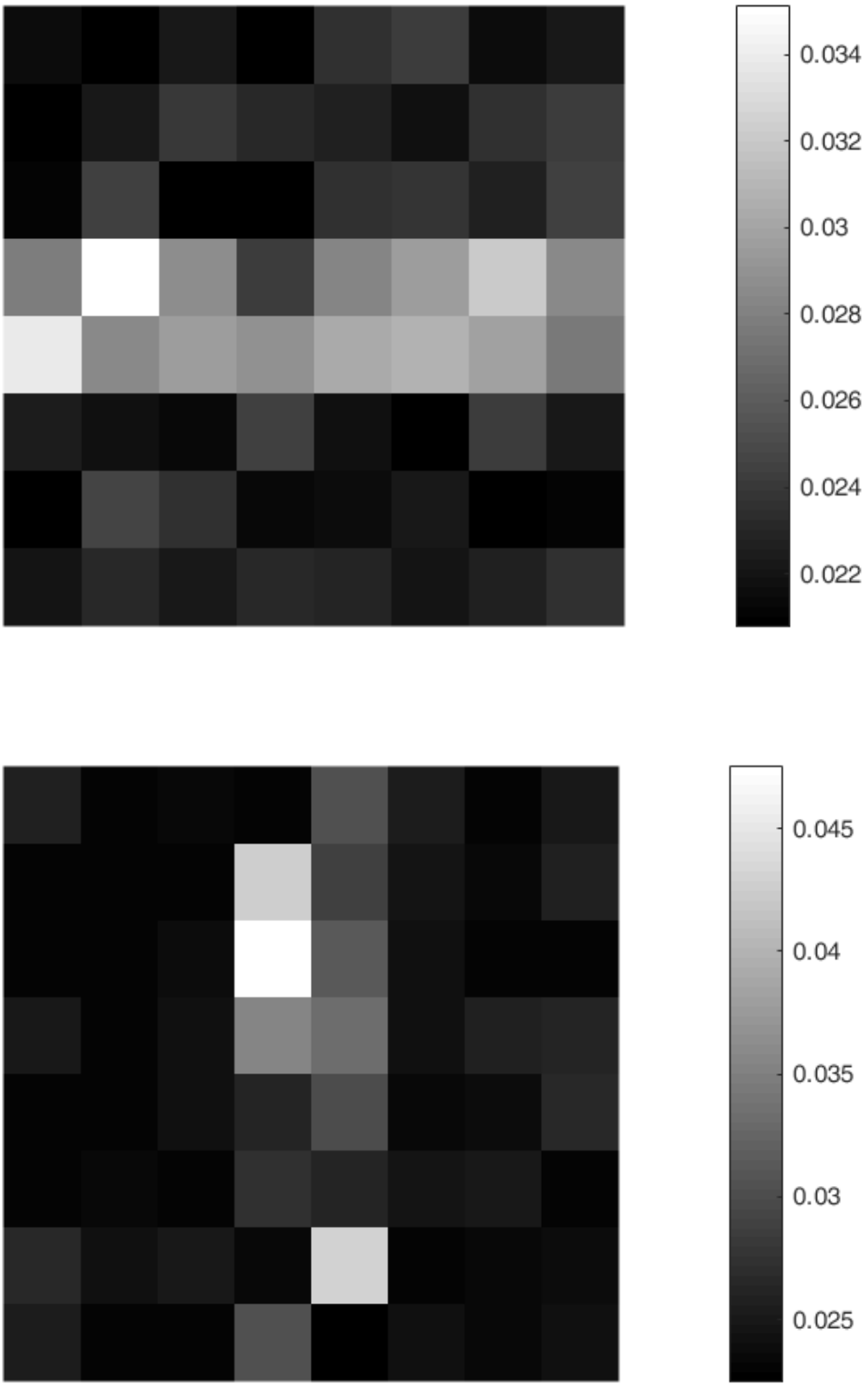


Fig. 5.4. Representación numérica de los pesos sinápticos para patrones horizontales (superior) y para patrones verticales (inferior) después de 2000 segundos de entrenamiento.

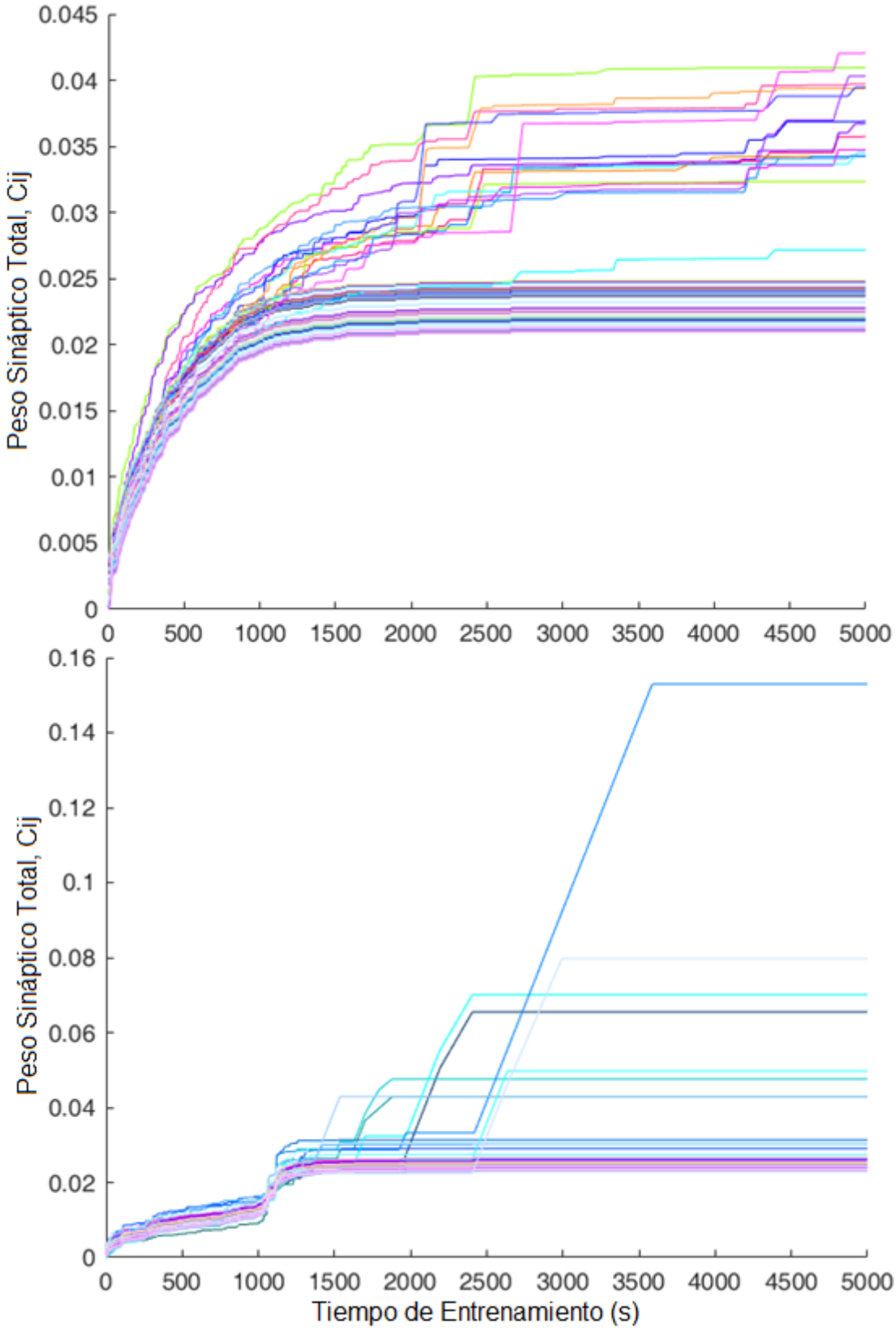


Fig. 5.5. Evolución de los pesos sinápticos para patrones horizontales (superior) y para patrones verticales (inferior) después de 5000 segundos de entrenamiento.

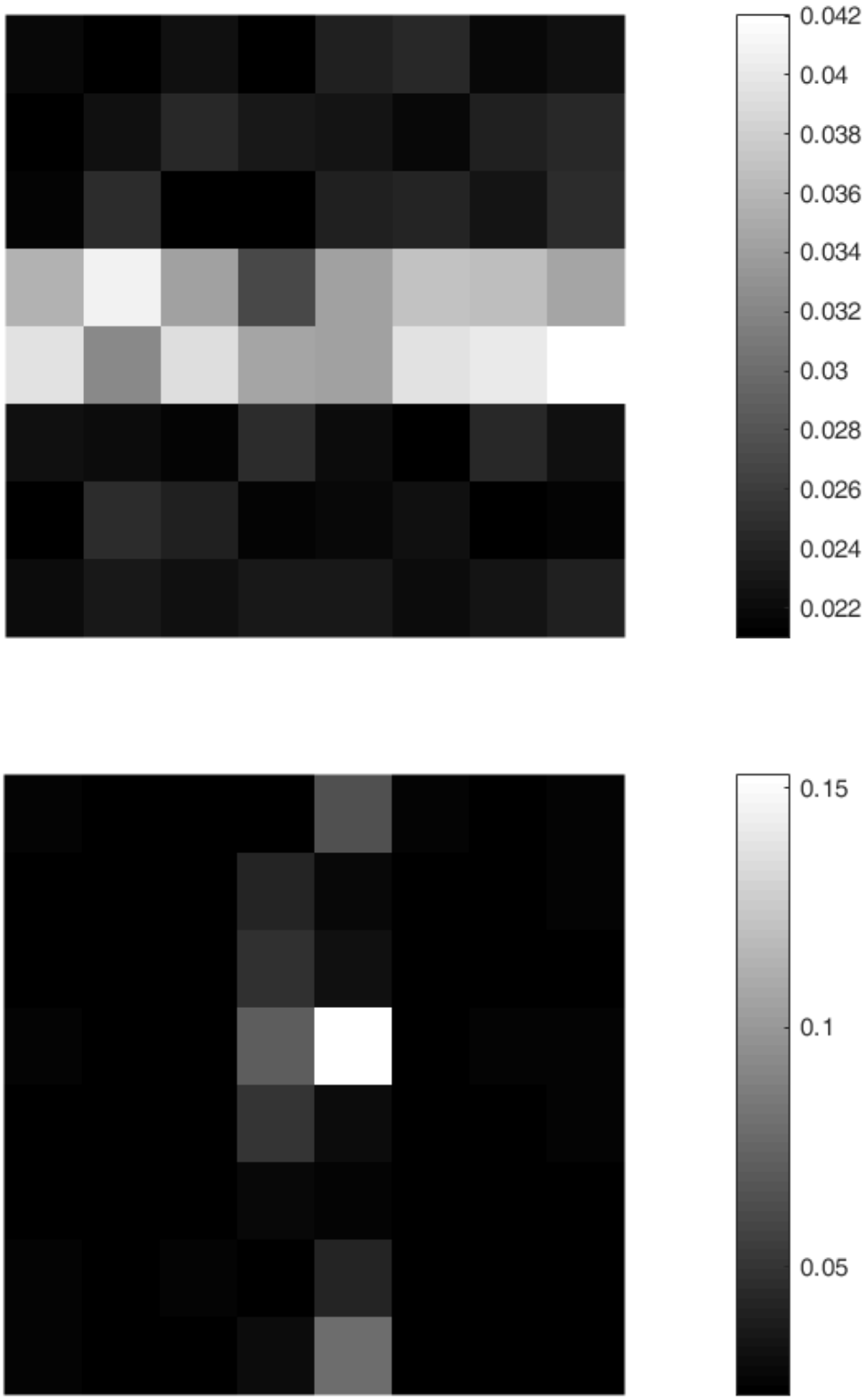


Fig. 5.6. Representación numérica de los pesos sinápticos para patrones horizontales (superior) y para patrones verticales (inferior) después de 5000 segundos de entrenamiento.

5.3 Prueba de funcionamiento de la red neuronal

La prueba consistió en presentar en la entrada de la red el conjunto de patrones de la fig.5.7, en el mismo orden que se observa. Dentro de este conjunto existen algunos patrones que no pertenecen estrictamente hablando, a ninguna de las clases de la red. Esto se hizo para conocer la capacidad de "inferir" de la red. Estos patrones de esta clase "desconocida" incluían "cruces", "diagonales" y arreglos que no seguían la regla de creación de los patrones descrita en la sección 4.6 la cual consistía en generar vectores con 5, 6, 7 u 8 ceros. Se muestran dos representaciones de los resultados obtenidos, una de ellas muestra las señales de salida generadas por las cuatro neuronas de la red, mientras que la otra muestra la representación discreta o de "tramas" de las señales de salida de las cuatro neuronas.

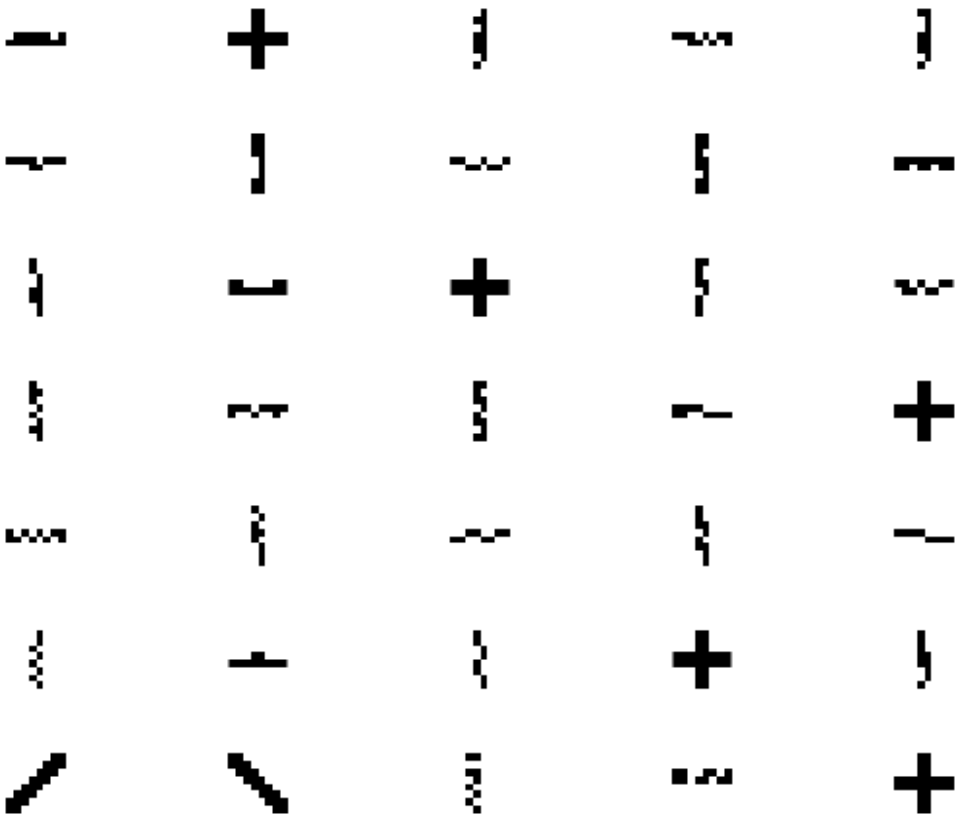


Fig. 5.7. Patrones de prueba para la red. Se pueden apreciar algunas configuraciones que no pertenecen a la clase horizontal o a la clase vertical y otras con menos cantidad de pixeles negros. La lectura del patrón se realiza de izquierda a derecha, comenzando por la esquina superior izquierda.

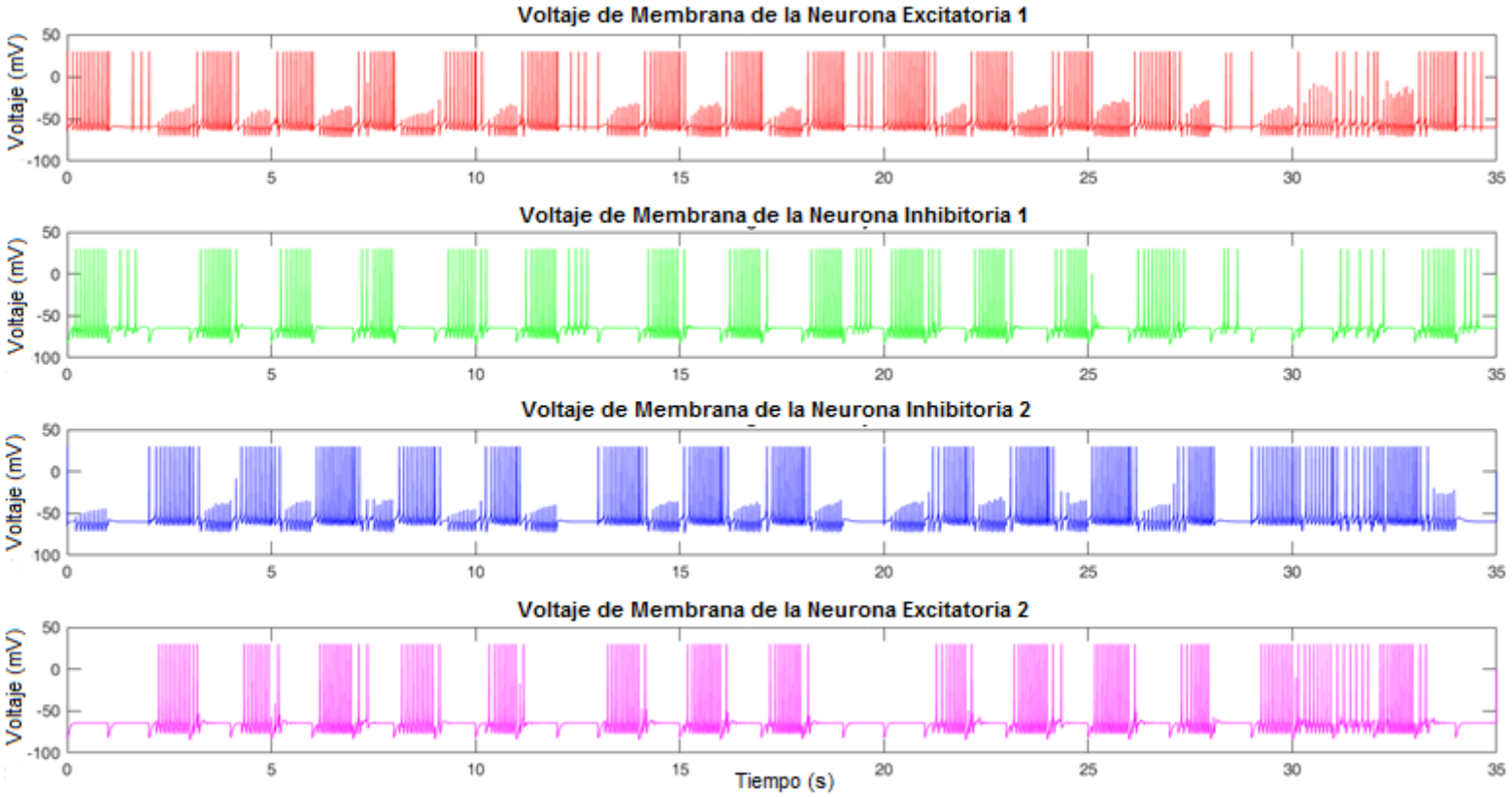


Fig. 5.8. Señales de salida generadas por la capa WTA de la red neuronal. La señal roja corresponde a la neurona excitatoria de los patrones horizontales mientras que la señal violeta corresponde a la neurona excitatoria de los patrones verticales. La neurona excitatoria 1 es inhibida por la neurona inhibitoria 2 y la neurona excitatoria 2, es inhibida por la neurona inhibitoria 1.

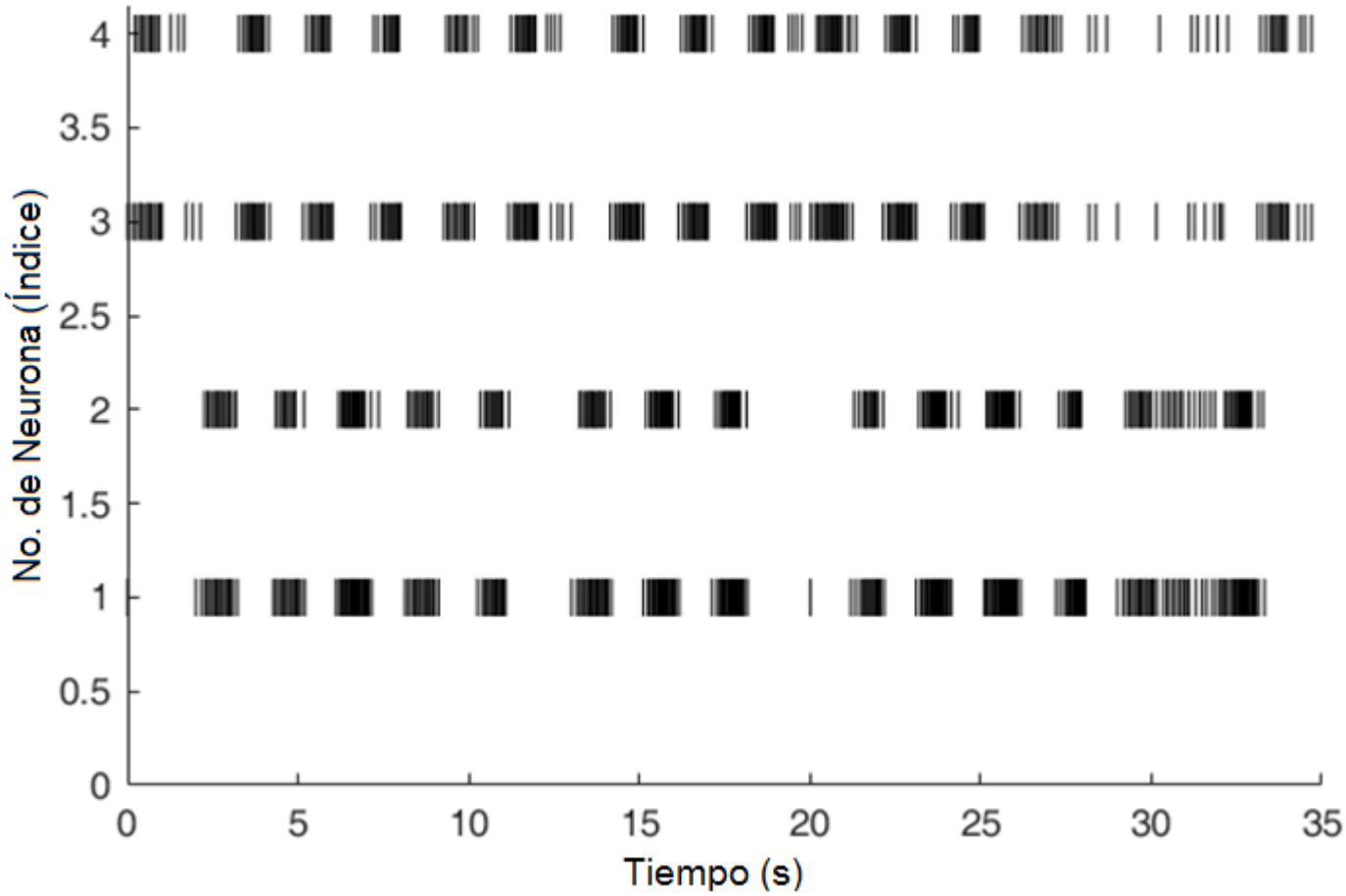


Fig. 5.9. Representación discreta de las señales de salida de la capa WTA. Los índices 3 y 1 son la neurona excitatoria 1 y la neurona excitatoria 2, respectivamente. Los índices 4 y 2 son las neurona inhibitoria 1 y la neurona inhibitoria 2, respectivamente.

5.4 Resultados de la prueba de desempeño de la red neuronal

Como ya se estudió en la sección 2.5, las redes neuronales pulsadas muestran un comportamiento completamente estocástico, por lo que el análisis del desempeño después del entrenamiento, debe realizarse de la misma manera. Esto se logra usando la ecuación (2.5), descrita previamente y un conjunto adicional de ecuaciones presentadas en [34]. La primera de ellas es la ecuación (5.1), la cual permite obtener la probabilidad de predicción de una clase específica incluida en el entrenamiento de la red:

$$predicción(x_i) = \arg\max_{clase} \sum_{k} P(clase|Z_k) N_{ki}^{prueba} \tag{5.1}$$

$$P(clase|Z_k) = \frac{N_k^{clase}}{N_k} \tag{5.2}$$

donde N_k^{clase} es el número de pulsos emitidos en cada clase (horizontal y vertical) después del entrenamiento y N_k es el total de pulsos emitidos por las neuronas de salida, en este caso son dos por lo que N_k se reduce a la suma de los pulsos de ambas clases. El término N_{ki}^{prueba} es el número de pulsos emitidos por cada neurona de salida al presentarse un patrón x_i del conjunto de prueba, $x_i \in X_{prueba.}$

Las fig.5.10 y fig.5.11 muestran los resultados de la ecuación (2.5) usando una secuencia de prueba de un patrón horizontal, luego ningún patrón y finalmente un patrón vertical. La fig.5.12 muestra las probabilidades de disparo de cada uno de los patrones del conjunto de prueba. Cabe mencionar que la ecuación (2.5) solo calcula la probabilidad de disparo de la red cuando se muestra un patrón a la entrada que puede o no pertenecer a la red. La probabilidad de disparo aumenta si el patrón pertenece a alguna clase de la red. Finalmente, la fig.5.13 muestra la predicción de cada uno de los patrones del conjunto de prueba.

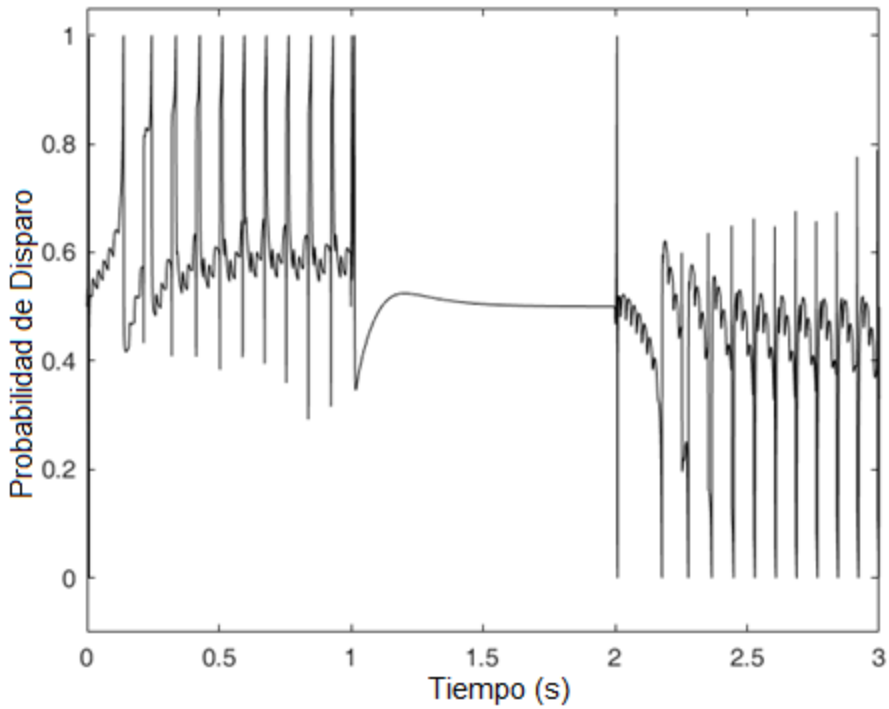


Fig. 5.10. Probabilidad de disparo de la neurona excitatoria 1 (horizontal). Nótese que al presentarse un patrón horizontal la probabilidad de disparo es de 1 mientras que al presentar un patrón vertical la probabilidad de disparo es de 0, el cual es un resultado esperado.

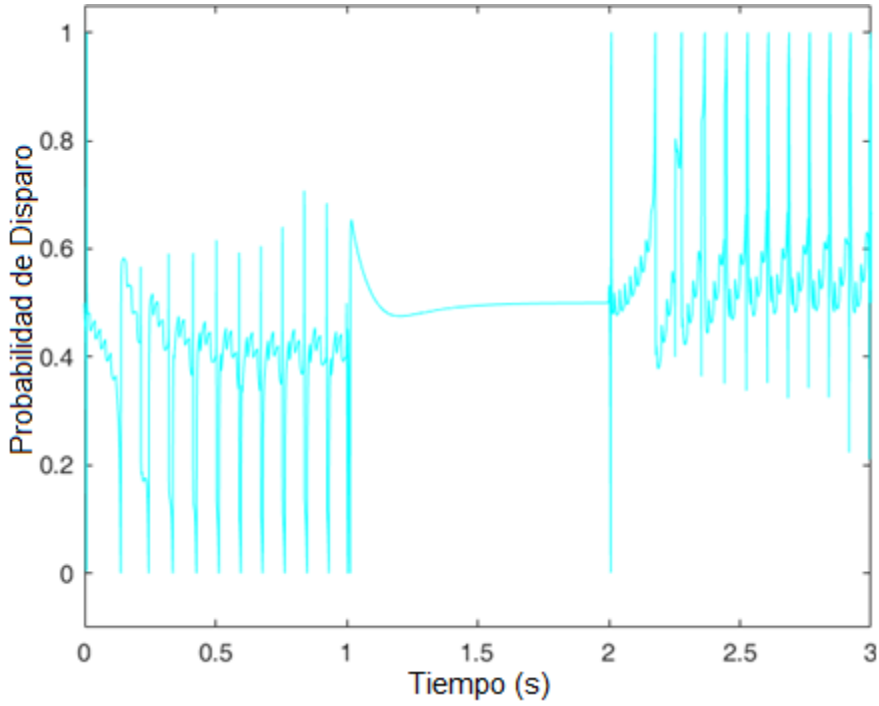


Fig. 5.11. Probabilidad de disparo de la neurona excitatoria 2 (vertical). Nótese que al presentarse un patrón vertical la probabilidad de disparo es de 1 mientras que al presentar un patrón horizontal la probabilidad de disparo es de 0, el cual es un resultado esperado.

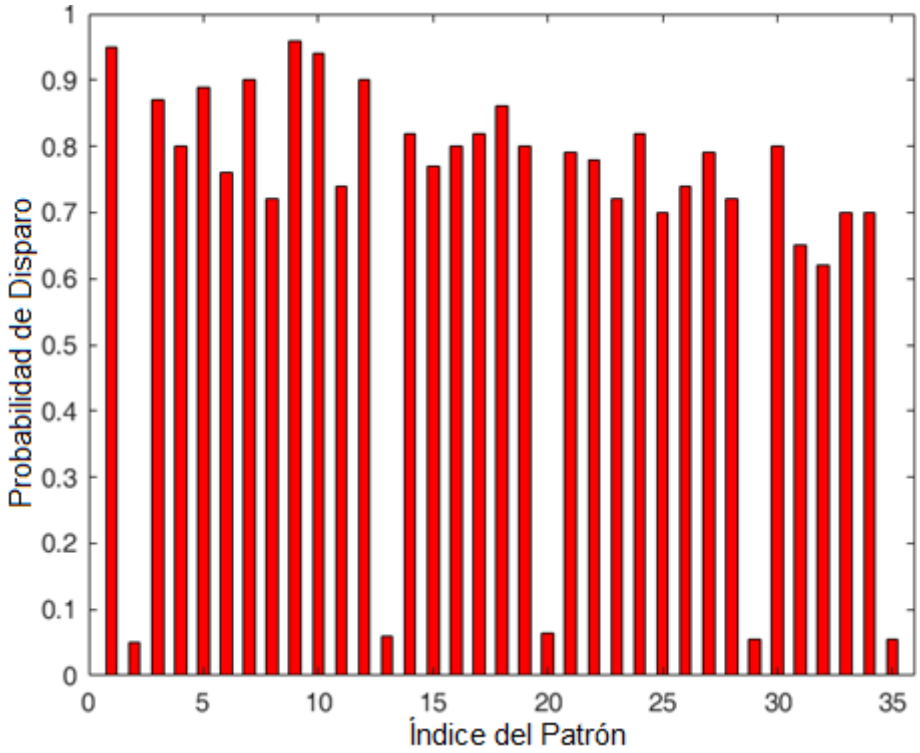


Fig. 5.122. Probabilidad de disparo de los 35 patrones de prueba. Nótese que las barras con menos probabilidad pertenecen a los patrones con forma de cruz.

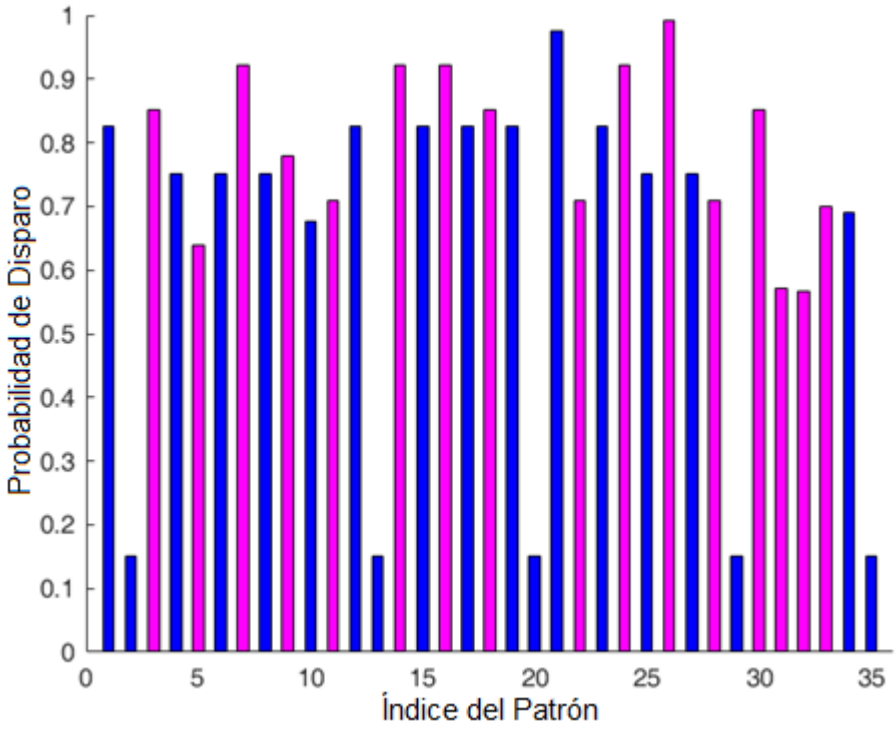


Fig. 5.133. Predicción de los 35 patrones de prueba. Las barras azules representan la predicción de los patrones horizontales mientras que las barras violetas representan la predicción de los patrones verticales. Debido a que la red neuronal es de clasificación binaria, todos los patrones presentados en la entrada deben ser clasificados en alguna de las dos clases.

5.5 Resultados de las simulaciones en Pyxis Layout de los modelos estocásticos de memristores.

Se realizaron una serie de simulaciones de los tres modelos estocásticos, de éstas, se capturaron los resultados más relevantes que son las formas de onda del voltaje y la corriente, la forma característica de la histéresis y la variable estocástica, en este caso el voltaje de umbral o la corriente de umbral (modelo de Pickett). La fig.5.14 muestra el circuito usado, el cual está constituido de una fuente de alimentación, de voltaje alterno senoidal para los modelos de Biolek y Yakopcic y corriente alterna senoidal para el modelo de Pickett y la representación esquemática del modelo respectivo codificado en Verilog A. Cada simulación se realizó con los parámetros de la tabla 4.1.

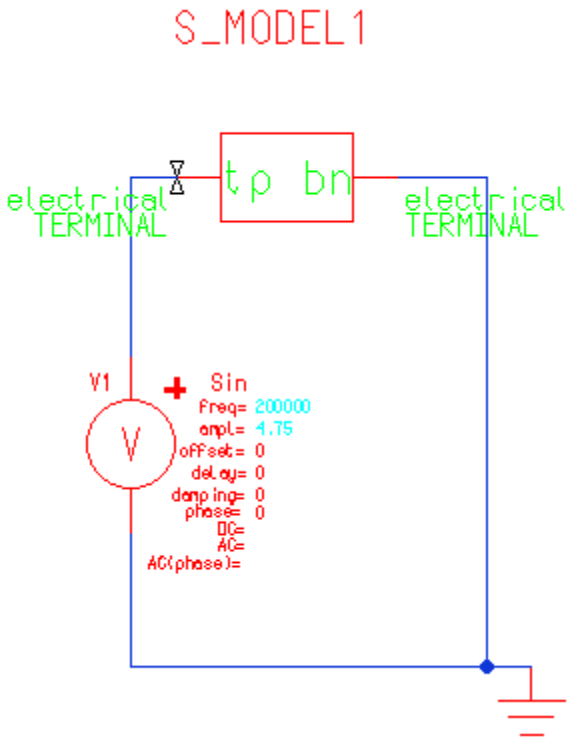


Fig. 5.14. Diagrama esquemático de prueba de los tres modelos estocásticos.

Las siguientes figuras muestran los resultados de las simulaciones, con sus respectivos parámetros (fig.5.15 a fig.5.24). Como parte complementaria, se simularon los mismos modelos pero en su versión determinista (fig.5.25 a fig.5.27) obteniendo la forma característica de la histéresis.

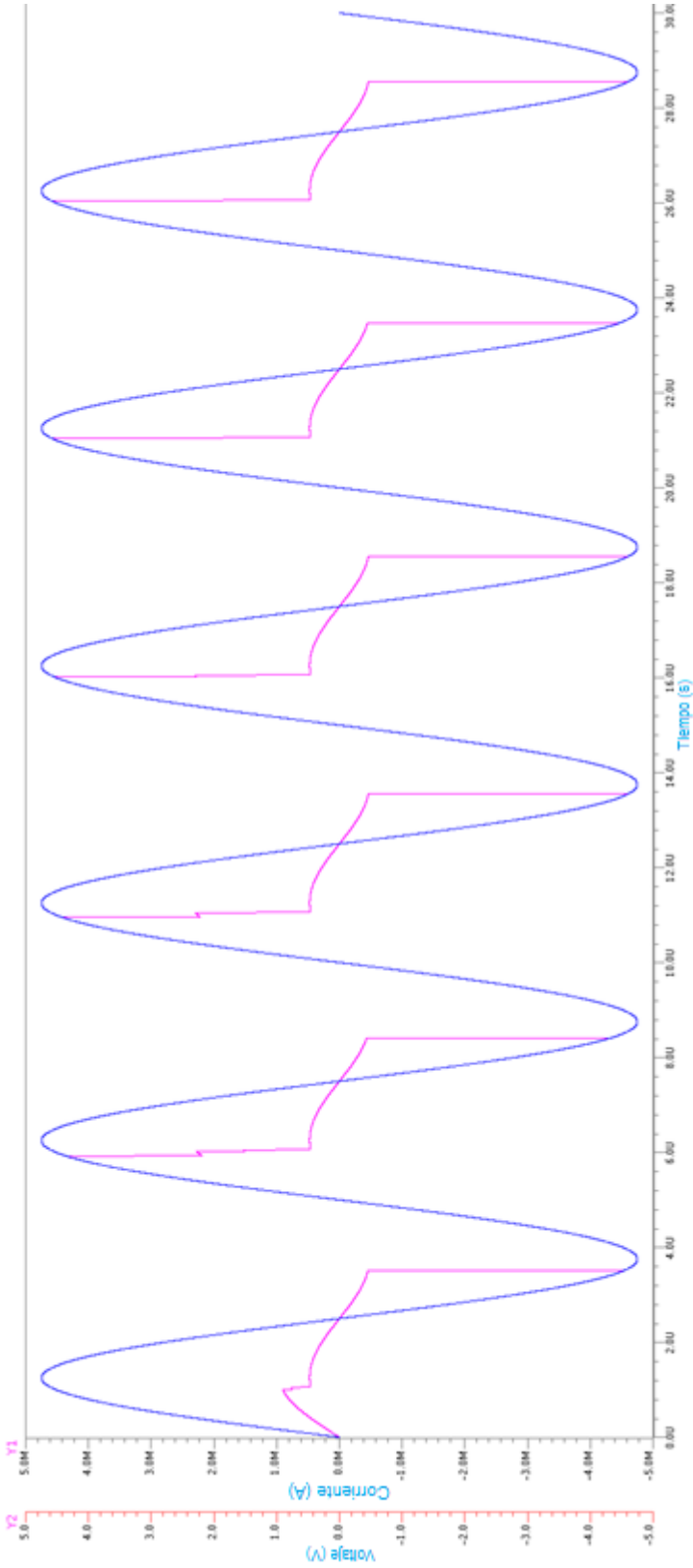


Fig. 5.15. Formas de onda del voltaje (azul) y la corriente (violeta) del modelo estocástico Biolek.

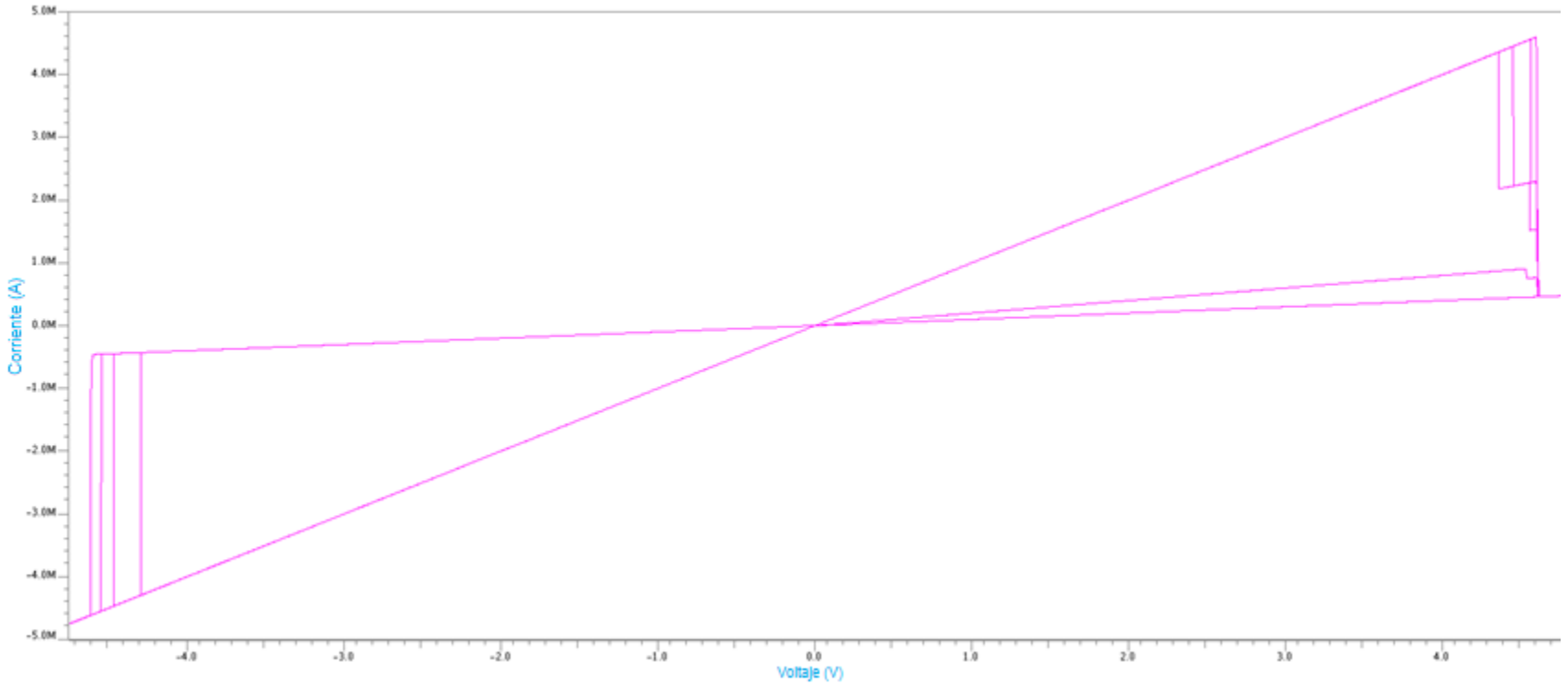


Fig. 5.16. Forma característica de la histéresis del modelo estocástico de Biolek. Nótese que existen varias señales que se anteponen a diferentes valores de voltaje, este es el comportamiento clásico de un modelo estocástico.

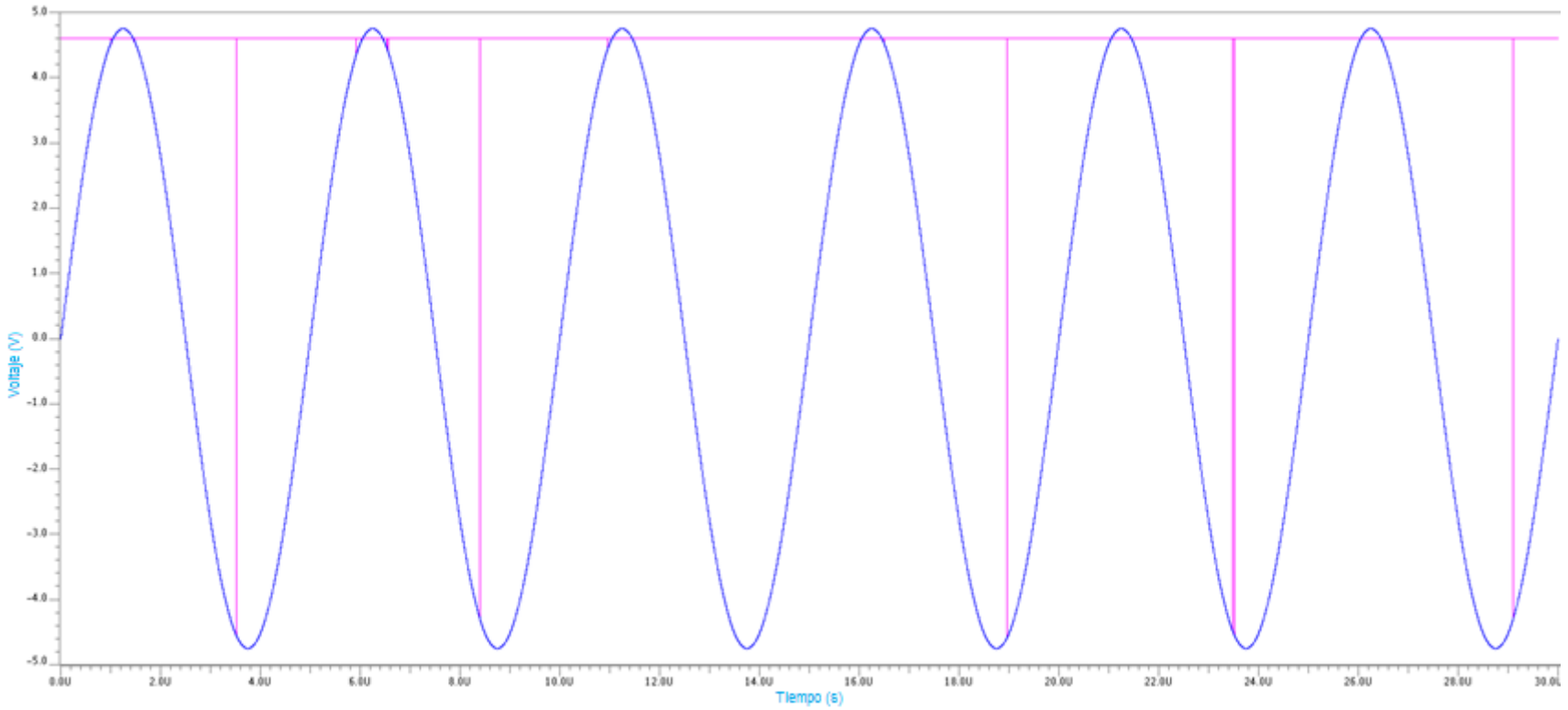


Fig. 5.17. Voltaje de umbral del modelo estocástico de Biolek. El voltaje de umbral tiene distintas amplitudes debido a la estocasticidad inducida en el algoritmo.

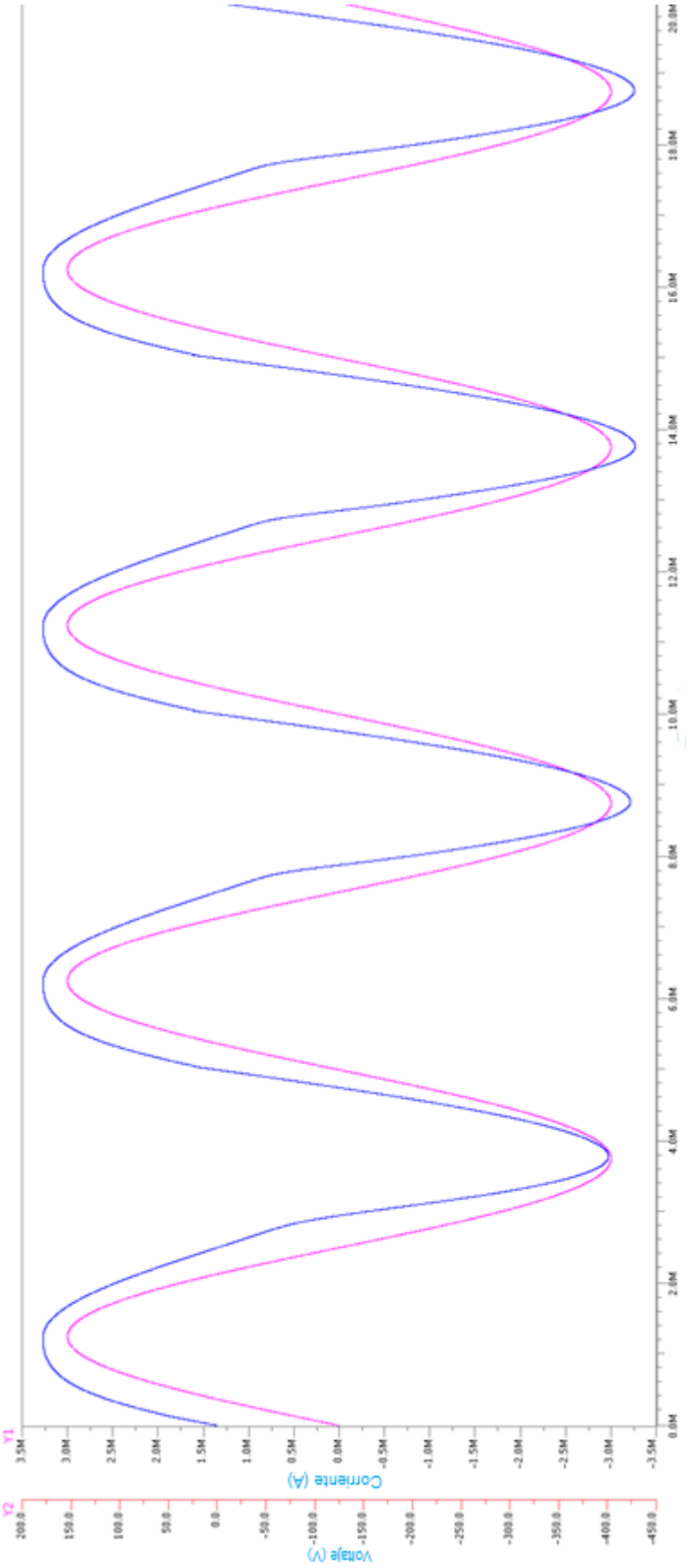


Fig. 5.18. Formas de onda del voltaje (azul) y la corriente (violeta) del modelo estocástico de Pickett.

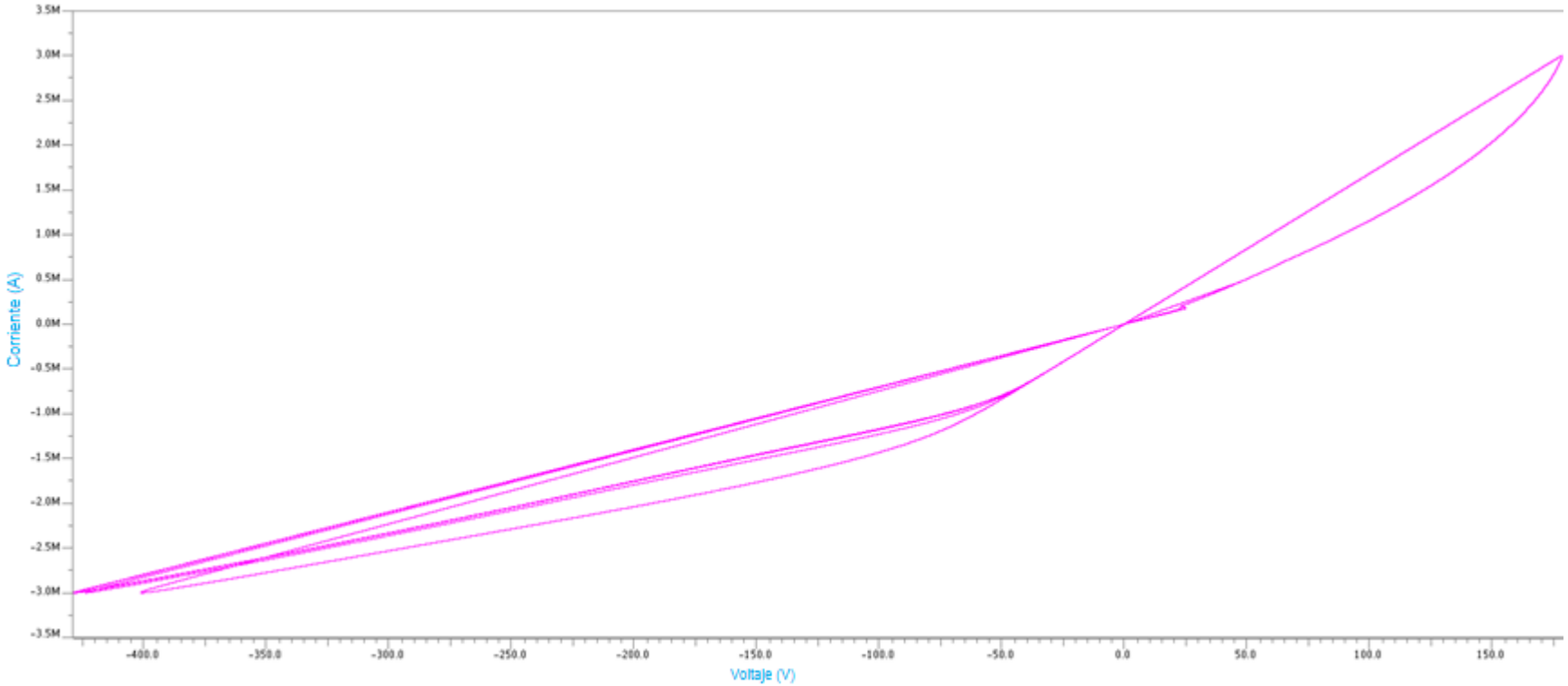


Fig. 5.19. Forma característica de la histéresis del modelo estocástico de Pickett. Nótese que existen varias señales que se anteponen a diferentes valores de corriente, este es el comportamiento clásico de un modelo estocástico.

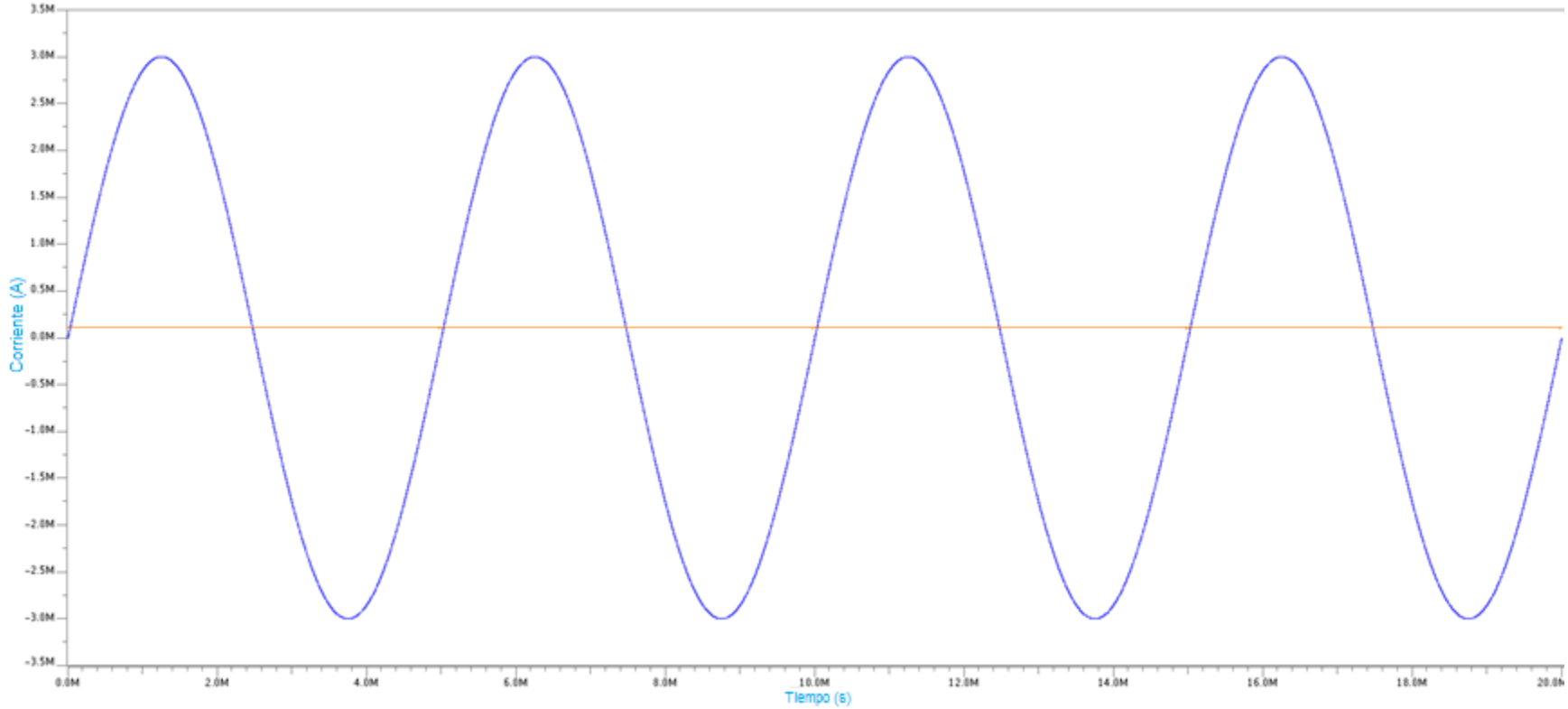


Fig. 5.20. Corriente de umbral del modelo estocástico de Pickett. La corriente de umbral tiene distintas amplitudes debido a la estocasticidad inducida en el algoritmo. Esto se puede apreciar mejor en la fig.5.21.

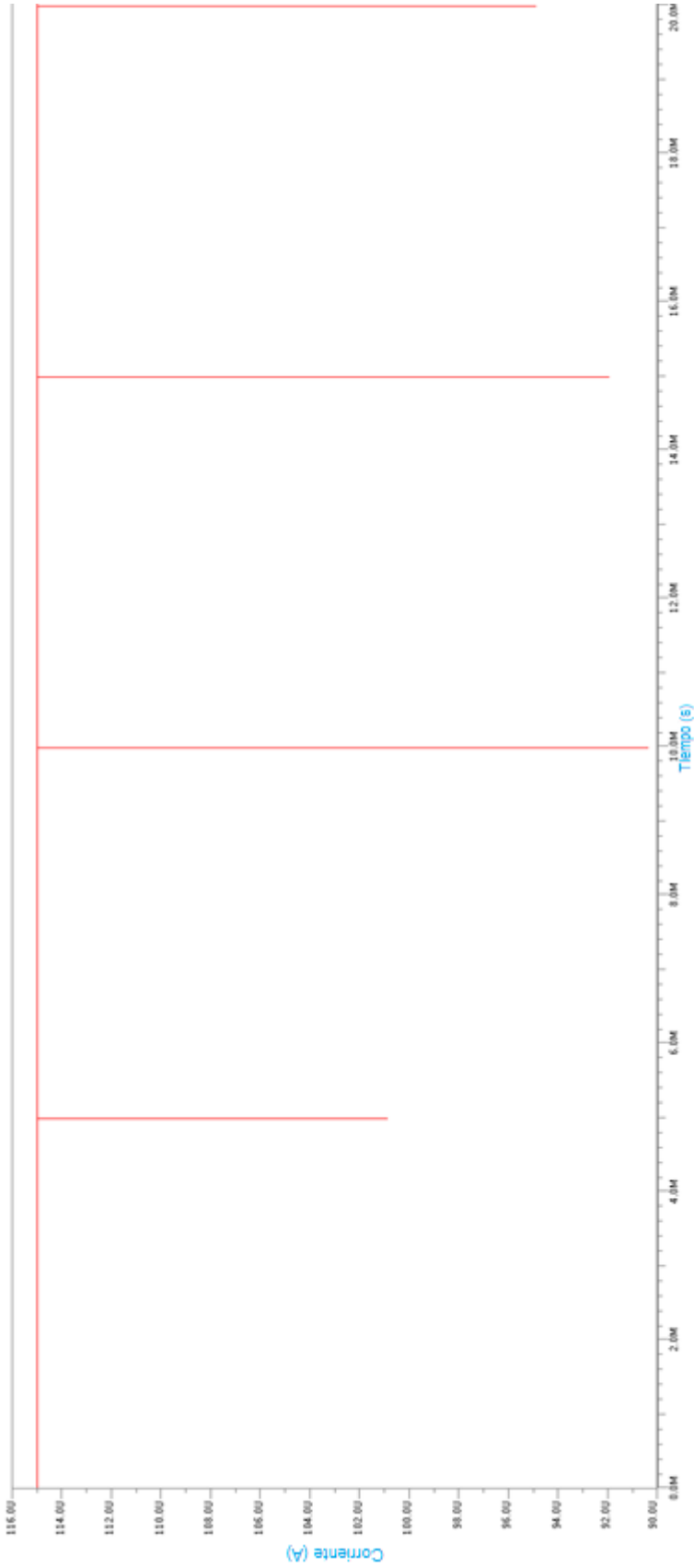


Fig. 5.21. Ampliación de la corriente de umbral de la fig.5.20 del modelo estocástico de Pickett.

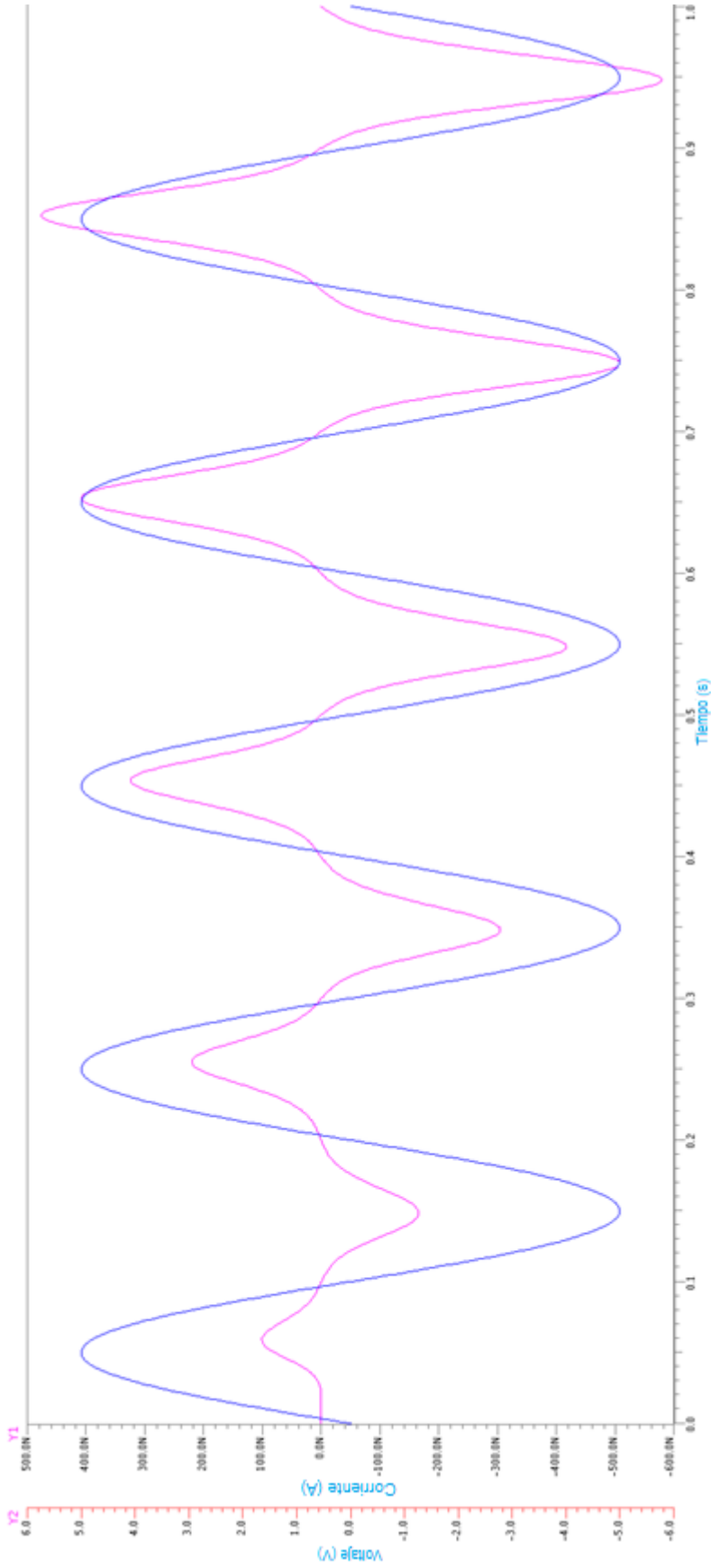


Fig. 5.22. Formas de onda del voltaje (azul) y la corriente (violeta) del modelo estocástico de Yakopcic.

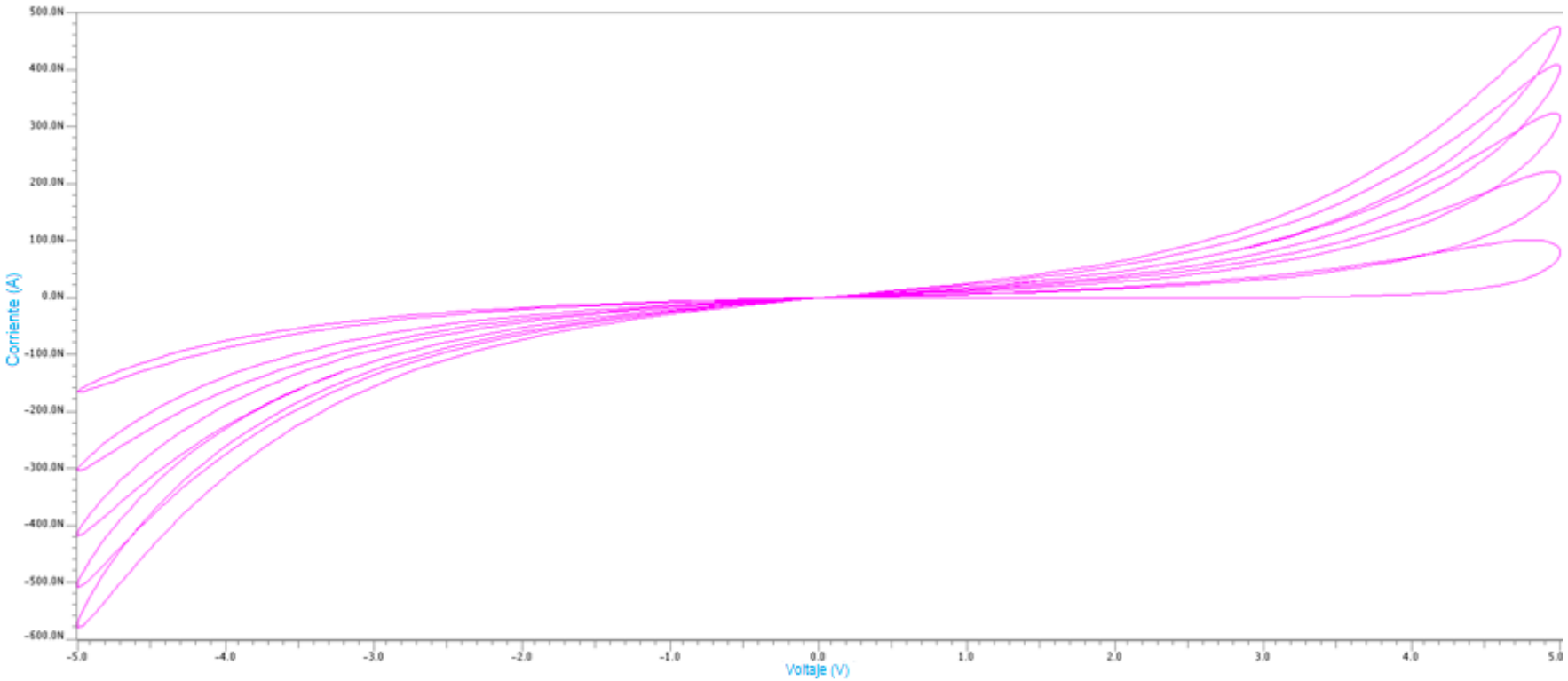


Fig. 5.23. Forma característica de la histéresis del modelo estocástico de Yakopcic. Nótese que existen varias trayectorias que cruzan por el mismo punto este es el comportamiento clásico de un modelo estocástico, sin embargo el modelo determinista también muestra este comportamiento aunque la única diferencia entre ambos modelos es la repetitividad de los resultados y las resistencias R_{ON} y R_{OFF}.

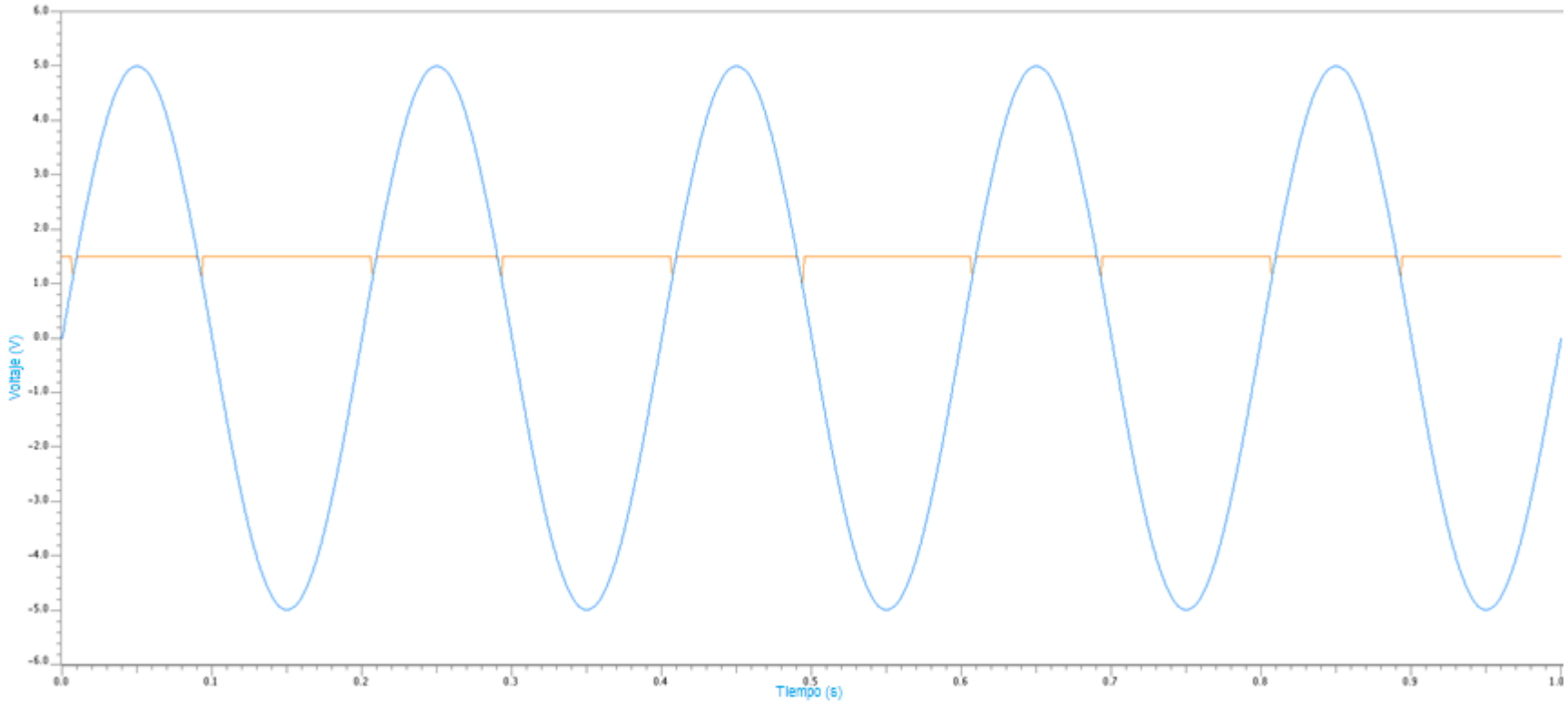


Fig. 5.24. Voltaje de umbral del modelo estocástico de Yakopcic. La corriente de umbral tiene distintas amplitudes debido a la estocasticidad inducida en el algoritmo.

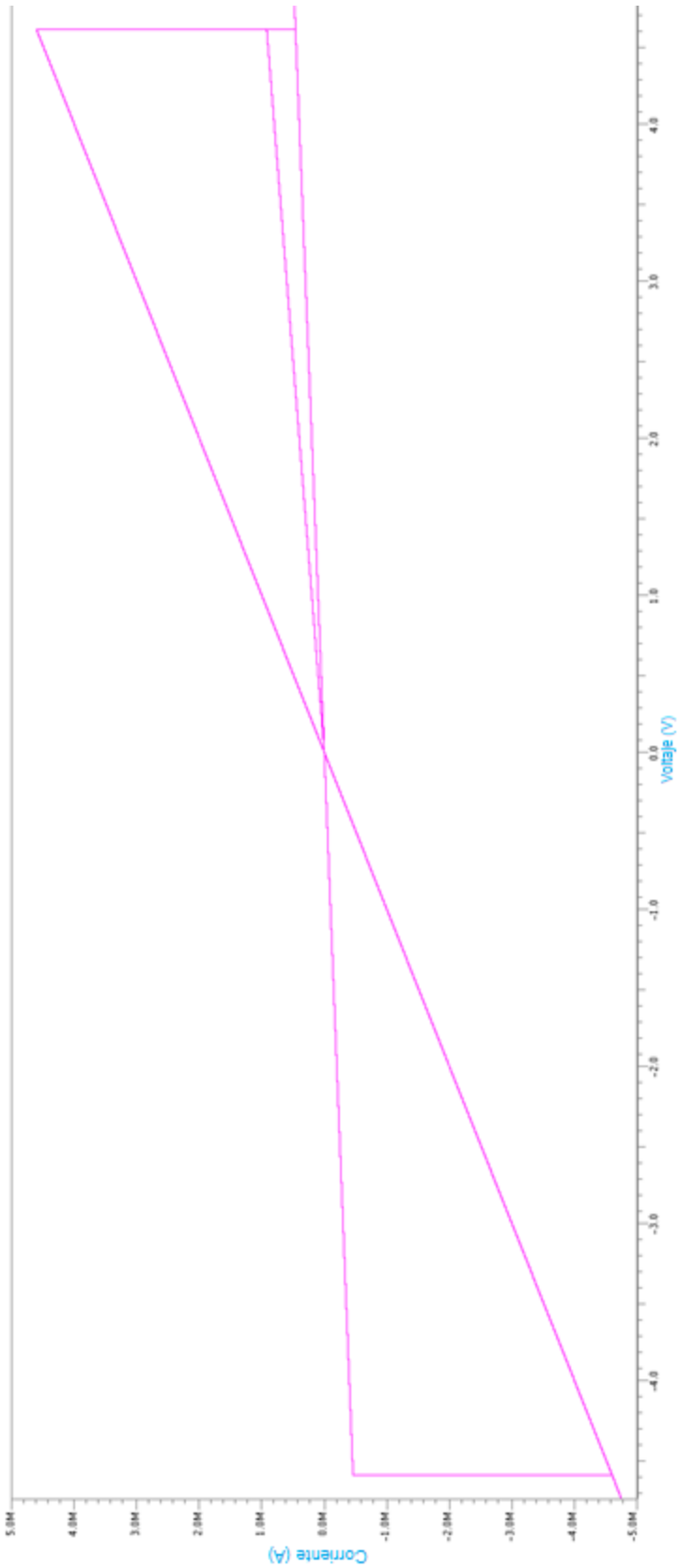


Fig. 5.25. Histéresis del modelo de Biolek en su versión determinista. Para más detalles consultar la sección 3.3.1.

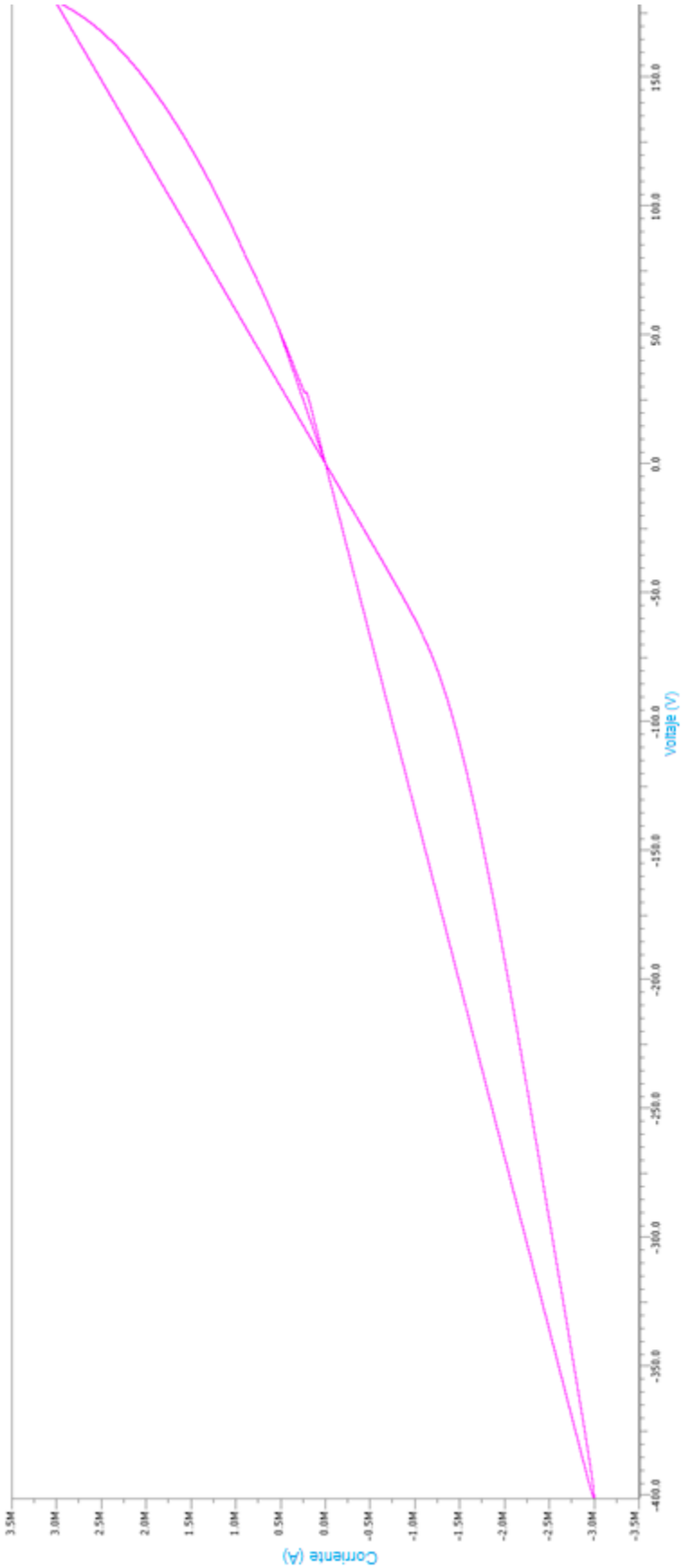


Fig. 5.26. Histéresis del modelo de Pickett en su versión determinista. Para más detalles consultar la sección 3.3.3.

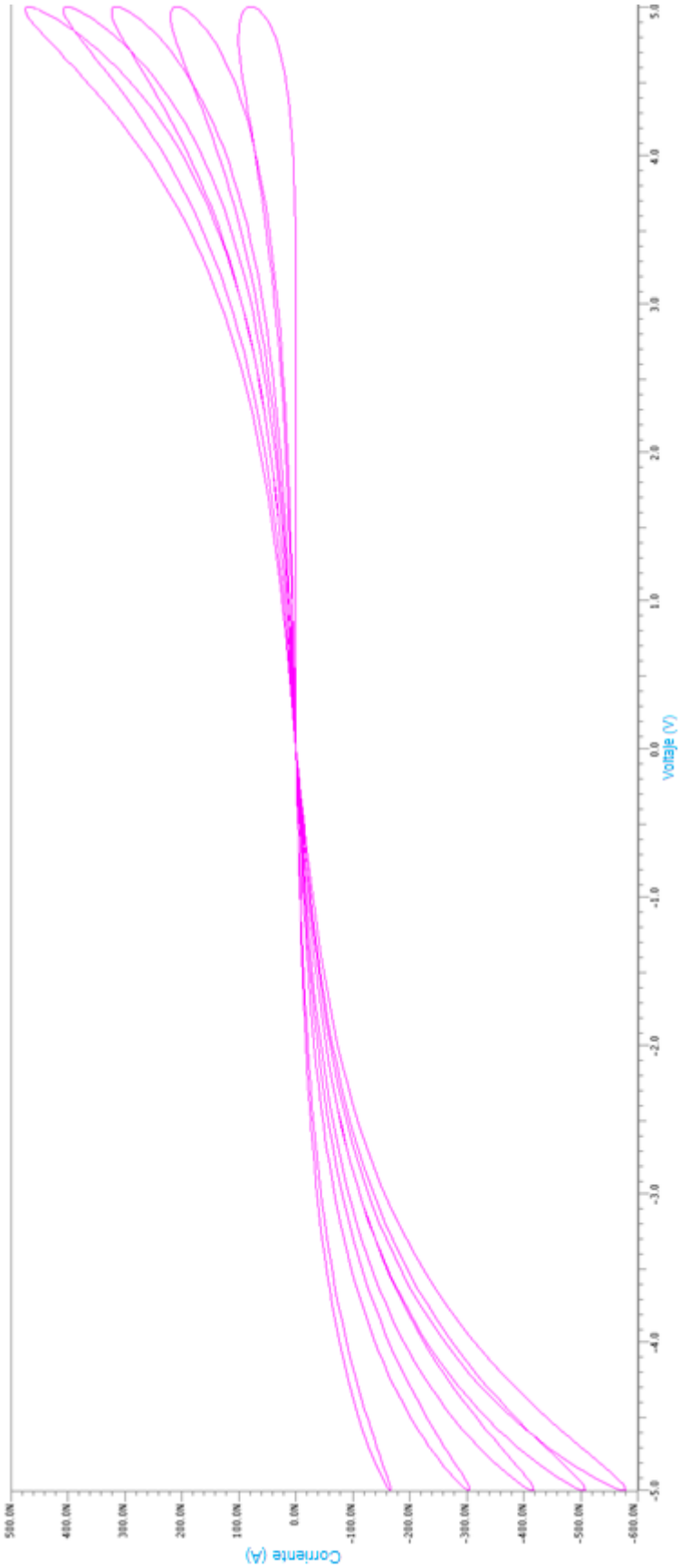


Fig. 5.27. Histéresis del modelo de Yakopcic en su versión determinista. Para más detalles consultar la sección 3.3.2.

5.6 Conclusiones

Los resultados anteriores muestran que los modelos propuestos se comportan de acuerdo a lo esperado. Incluso sin realizar un análisis detallado, por inspección de las curvas características de histéresis, se puede apreciar el comportamiento probabilístico, ya que bajo condiciones de polarización semejantes, las trayectorias cerradas aparecen en distintos niveles de corriente o de voltaje, y todas las transiciones se presentan en relación directa con los niveles de umbral tanto de corriente, como de voltaje. Otra característica que muestran estos resultados, es la importante similitud del modelo estocástico y el modelo determinista. Esto es fundamental ya que es posible estudiar los parámetros dinámicos como son $R_{ON,}\, R_{OFF}, V_T$ e I_T. Con estos resultados será posible plantear una propuesta de circuito neuromórfico para emular el comportamiento biológico de una neurona pulsada y analizar el modelo en poblaciones más numerosas.

6 CONCLUSIONES GENERALES

Los sistemas de neuronas biológicos son extremadamente eficientes tanto en el consumo de energía como en el procesamiento de información. El cerebro en particular, está compuesto de miles de millones de células neuronales y tan solo consume cerca 20 W de energía. A pesar de que las neuronas son ruidosas, hablando en términos eléctricos y son imprecisas, cuando se integran como una arquitectura en el cerebro e interactúan entre sí, pueden resolver tareas realmente complejas y exhiben un comportamiento complejo en tiempo real con una alta precisión mientras su consumo de energía es realmente bajo. Esta premisa permitió plantear un tema de estudio como el que se presentó en esta tesis el cual consiste en demostrar a través del modelado computacional las capacidades de procesamiento de una red neuronal pulsada, que es fundamentalmente una red bio-inspirada. Además, con la ayuda de técnicas más sofisticadas como es el modelado estocástico de memristores, se planteó la posibilidad de desarrollar un sistema que no dependa de una computadora y que aproveche las características de eficiencia que esta red pulsada promete.

En términos más precisos, se demostró que las redes neuronales pulsadas presentan un procesamiento más eficiente, ya que con tan solo cuatro neuronas, fue posible obtener un clasificador binario, con la posibilidad de extender a un clasificador multiclase sin aumentar significativamente la complejidad de la red. Sin embargo, el principal inconveniente al desarrollar la red neuronal, se presentó en el método de entrenamiento usando STDP ya que a pesar de ser un método completamente compatible con la arquitectura de la red neuronal, el conjunto de entrenamiento fue de gran tamaño, lo que incrementó considerablemente el tiempo de entrenamiento. Esto implica tiempos de entrenamiento poco prácticos si se tratará de un clasificador multiclase. Una propuesta para reducir estos tiempos, radica en la selección adecuada de los parámetros característicos de la función STDP, ya sea en los tiempos τ_- y τ_+ o en las constantes A_- y A_+.

7 TRABAJO FUTURO

Como trabajo futuro, se pretende imitar la capacidad inmensamente útil de procesamiento que presentan las redes neuronales pulsadas, a través de su implementación embebida, abarcando dos grandes ramas que son la electrónica analógica y la electrónica digital, culminando el desarrollo del proyecto en una integración a gran escala ya sea usando FPGAs o un circuito integrado de propósito específico. Fundamentalmente, esto se realizará con una primera propuesta, usando los algoritmos presentados en esta tesis pero recodificando en un lenguaje más enfocado a la descripción de sistemas neuromórficos, como es Simulink, VHDL y Verilog A.

El futuro de este proyecto es prometedor y de un gran impacto tecnológico, por lo que continuar con su desarrollo permitirá descubrir nuevas capacidades computacionales así como también nuevos campos de aplicación.

A APÉNDICE A

A.1 Espacio de fase: retratos de fase

Existen ecuaciones diferenciales que no pueden ser resueltas convenientemente mediante métodos analíticos. Debido a esto, resulta funcional considerar cierta información cualitativa que puede ser obtenida acerca de sus soluciones sin realmente resolver las ecuaciones. Los aspectos que serán considerados están asociados con la idea de estabilidad de una solución y los métodos empleados son esencialmente geométricos.

Considérese un sistema de ecuaciones diferenciales de primer orden, de las funciones x y y y la variable dependiente t, de la forma:

$$\begin{cases} \dfrac{dx}{dt} = f(x,y) \\ \dfrac{dy}{dt} = g(x,y) \end{cases} \tag{A.1}$$

Nótese que la variable independiente t no aparece en los términos de la derecha en (A.1). A este tipo de sistemas se les denomina **autónomos**. Un **punto de equilibrio** (también llamado punto crítico) del sistema (A.1) es un punto $(x_0, y_0) \in \mathbb{R}^2$ tal que $f(x_0, y_0) = 0 = g(x_0, y_0)$, y la solución correspondiente $x(t) = x_0, \quad y(t) = y_0$ se denomina **solución de equilibrio**. Las dos funciones $x(t)$ y $y(t)$ pueden considerarse como las dos componentes de una función vectorial que se denota como $\vec{X}(t)$. Esta última función está definida en un intervalo I de $\mathbb{R}$ y toma valores en $\mathbb{R}^2$. Al variar la variable independiente t, el punto $\big(x(t), y(t)\big)$ define una curva en plano xy (también llamado **plano de fase**), con t como parámetro. Esta curva planar se conoce como **trayectoria**. La gráfica de una (o varias) trayectorias dibujada como una curva paramétrica en el plano de fase, se le denomina **retrato de fase**.

Obsérvese que las trayectorias de las soluciones de equilibrio son precisamente puntos (los puntos de equilibrio), por lo que es posible obtener una representación gráfica más precisa de (A.1) usando la siguiente expresión

$$\frac{dy}{dx} = \frac{g(x,y)}{f(x,y)} \tag{A.2}$$

La idea de hacer esta representación es aprovechar toda la información que puede brindar el campo de pendientes de la ecuación anterior, asignándoles además una dirección a cada uno de los segmentos de recta, lo que permitirá intuir la dirección del "flujo" de las soluciones al realizar incrementos en t. De esta forma, se genera un campo direccional que es mucho más representativo.

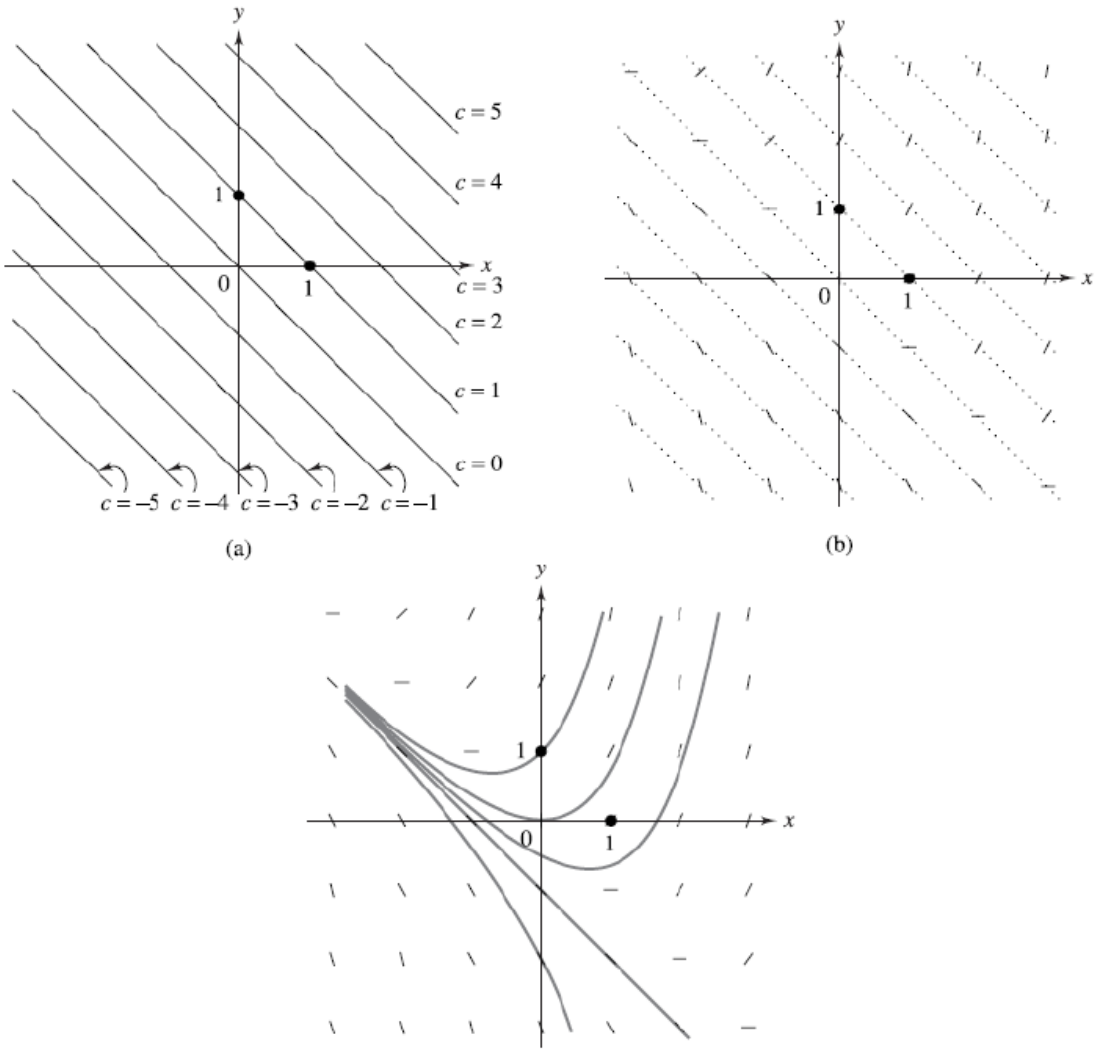


Fig. A.1. (a) Isóclinas para $y' = x + y$. (b) Campo direccional de $y' = x + y$. (c) Soluciones de $y' = x + y$

Un método típico para graficar dicho campo direccional, se realiza a través de uso de isóclinas. Una **isóclina** para la ecuación diferencial de la forma $y' = f(x, y)$ es el conjunto de puntos en el plano xy donde todas las soluciones tienen la misma pendiente dy/dx. Por ejemplo, si $y' = f(x, y) = x + y$, las isóclinas son las rectas $x + y = c$ o $y = c - x$, con c como una constante arbitraria. Nótese que c se puede interpretar como el valor numérico de la pendiente dy/dx de cada curva solución al cruzar la isóclina (es importante resaltar que c no corresponde a la pendiente de la isóclina, que en este caso es claramente -1). La Fig.A.1 muestra estas afirmaciones.

A.1.1 Estabilidad

La estabilidad es un concepto importante cuando se estudian los sistemas dinámicos. La estabilidad de una órbita de un sistema dinámico caracteriza si las orbitas cercanas (es decir, perturbadas) permanecerán alrededor de dicha órbita o serán repelidas lejos de ella.

Para definir matemáticamente la estabilidad de un sistema considérese un sistema autónomo $x'(t) = f(x(t))$ con f continuamente diferenciable en una región cualquiera D en el plano. Entonces se tienen tres formas de estabilidad:

- **Equilibrio estable.** Un punto crítico (o de equilibrio) x_0 en D se dice que es **estable** si para cada $\epsilon > 0$ corresponde $\delta > 0$ tal que:
  - **a)** Dado $x(0)$ en D con $\|x(0) - x_0\| < \delta$, entonces la solución $x(t)$ existe en $0 \leq t < \infty$ y
  - **b)** $\|x(t) - x_0\| < \epsilon$ para $0 \leq t < \infty$

Desde el punto de vista gráfico, esto significa que cada trayectoria se mueve sobre el punto crítico dentro de un rango finito de distancia.

- **Equilibrio inestable.** El punto crítico x_0 es llamado **inestable** cuando es **no estable**, o sea, cuando a) o b) o ambos no se cumplen.
  Gráficamente esto significa que todas las trayectorias (en algunos casos unas pocas) comienzan en el punto crítico y posteriormente se mueven alejándose de dicho punto.

- **Equilibrio asintóticamente estable.** El punto crítico x_0 se dice que es **asintóticamente estable** si a) y b) se cumplen (es estable) y adicionalmente:

  c) $\lim_{t\to\infty} \|x(t) - x_0\| = 0$ para $\|x(0) - x_0\| < \delta$

  En términos gráficos esto significa que todas las trayectorias convergen hacía el punto crítico como $t \to \infty$.

A.1.2 Clasificación de los puntos críticos

La información más relevante sobre un retrato de fase nos la dará la forma de las trayectorias cerca de un punto crítico, por lo que diferentes estados iniciales, derivan en diferentes trayectorias. Por eso, solo se considerarán las trayectorias más representativas.

Los puntos críticos se clasifican de acuerdo a su estabilidad. Además, debido a la naturaleza verdaderamente bidimensional de las curvas paramétricas, los puntos críticos también se clasifican por su forma (o más bien, por la forma descrita por las trayectorias sobre cada punto crítico). La clasificación es la siguiente, tomando como base las soluciones del sistema autónomo:

- **Caso I. Valores propios reales diferentes para una solución general de la forma $\boldsymbol{x(t) = C_1 k_1 e^{r_1 t} + C_2 k_2 e^{r_2 t}}$.**
  1. *Cuando r_1 y r_2 son ambos positivos, o ambos negativos.* El retrato de fase muestra trayectorias que se alejan del punto crítico a una distancia infinita (cunado $r > 0$), o se mueven directamente hacía el punto crítico, convergiendo con él (cuando $r < 0$). A esta forma se le conoce como **nodo**. Muestra un comportamiento **asintóticamente estable** cuando r_1 y r_2 son ambos negativos, y muestra un comportamiento **inestable** cuando r_1 y r_2 son ambos positivos.
  2. *Cuando r_1 y r_2 tienen signos opuestos (dígase $r_1 > 0$ y $r_2 < 0$).* El retrato de fase muestra que las trayectorias dadas por los vectores propios de los valores propios negativos, inicialmente

empiezan infinitamente distantes, y comienzan a avanzar, convergiendo eventualmente con el punto crítico. Las trayectorias que representan los vectores propios de los valores propios positivos, se mueven exactamente en la dirección contraria: comienzan en el punto crítico y posteriormente divergen hacía el infinito. A este tipo de forma se le conoce como **punto silla** y siempre es **inestable**.

- **Caso II. Valores propios repetidos para una solución general de la forma $\boldsymbol{x(t) = C_1 k_1 e^{rt} + C_2 k_2 e^{rt} = e^{rt}(C_1 k_1 + C_2 k_2)}$ y la forma $\boldsymbol{x(t) = C_1 k e^{rt} + C_2 (kte^{rt} + \eta e^{rt})}$, respectivamente.**
    3. *Cuando hay dos valores impropios k_1 y k_2 linealmente independientes.* Cada solución diferente de cero traza una trayectoria de línea recta en la dirección dada por el vector $C_1 k_1 + C_2 k_2$. Las trayectorias se mueven lejos del punto crítico (cuando $r > 0$), o se mueven en dirección del punto crítico y convergen (cuando $r < 0$). A este tipo de forma se le conoce como **nodo propio**. Es asintóticamente estable cuando $r < 0$ e inestable cuando $r > 0$.
    4. *Cuando solo hay un valor impropio k linealmente independiente.* Aquí, el retrato de fase comparte características con el del nodo. Con un solo vector impropio, es un nodo de aspecto degenerado que resulta de la combinación entre un nodo y un punto espiral (ver punto 6). Todas las trayectorias divergen alejándose del punto crítico hacía el infinito (cuando $r > 0$), o todas convergen con el punto crítico (cuando $r < 0$). Esta forma recibe el nombre de **nodo impropio** y es asintóticamente estable si $r < 0$ e inestable si $r > 0$.
- **Caso III. Valores propios complejos conjugados para una solución $\boldsymbol{x(t) = C_1 e^{\lambda t}\big(a cos(\mu t) - b sen(\mu t)\big) + C_2 e^{\lambda t}\big(a sen(\mu t) + b cos(\mu t)\big)}$.**

5. *Cuando la parte real λ es cero.* En este caso las trayectorias no convergen hacía el punto crítico ni se alejan. Más bien, se mantienen constantes, en órbitas elípticas (raramente circulares). A esta forma se le conoce como **centro**. Tiene una clasificación de estabilidad única que no comparte con ninguna otra forma: es **estable.**
6. *Cuando la parte real λ no es cero.* Las trayectorias mantienen los trazos elípticos de la forma anterior. Sin embargo, con cada revolución, las distancias de cada elipse crecen o se reducen exponencialmente de acuerdo al término $e^{\lambda t}$. Por lo tanto, el retrato de fase muestra trayectorias de espiral que se acercan (cuando $\lambda > 0$) o se alejan (cuando $\lambda < 0$) del punto crítico. Esta forma recibe el nombre de **punto espiral**. Su comportamiento es **asintóticamente estable** si $\lambda < 0$ e **inestable** si $\lambda < 0$.

La siguiente figura muestra la representación gráfica de las formas descritas anteriormente.

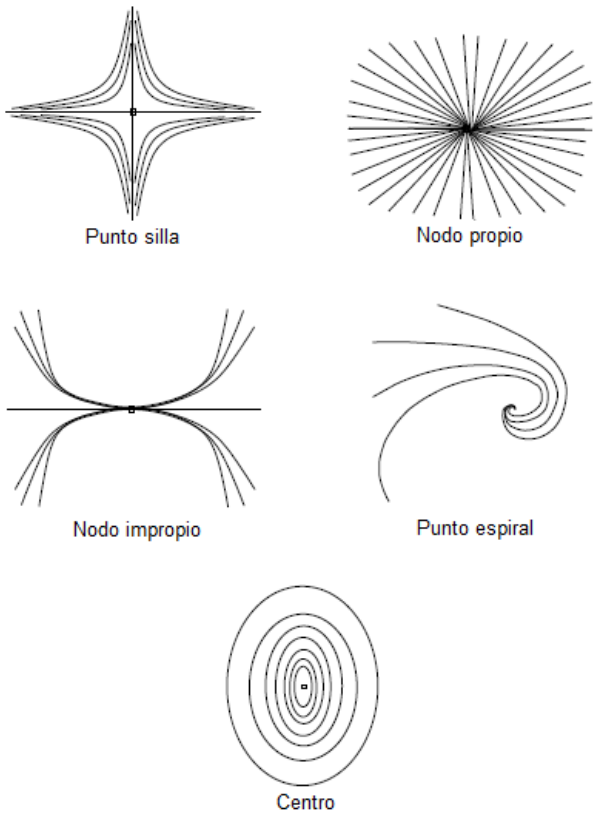


Fig. A.2. Clasificación de los puntos críticos de un sistema autónomo.

A.2 Introducción a la teoría de bifurcaciones

Una bifurcación se define como un cambio en los puntos de equilibrio u órbitas periódicas, o en sus propiedades de estabilidad, como consecuencia de la variación de un parámetro, o dicho de otra manera, una bifurcación ocurre cuando un pequeño cambio hecho a los valores de los parámetros de un sistema, causa un cambio repentino ya sea cualitativo o topológico en su comportamiento.

Cerca de una bifurcación, el sistema dinámico puede ser reducido a una forma matemática genérica mediante un cambio de variables y una reducción de su dimensión para mantener solo las direcciones implícitas en la bifurcación. Esas expresiones matemáticas reducidas son llamadas *formas normales* y cada forma se asocia a un tipo de bifurcación.

A.2.1 Atractores

Un atractor es un concepto muy importante en el campo de los sistemas dinámicos ya que permite conocer la estabilidad del sistema. El atractor o punto atractor se define como un conjunto de valores numéricos hacía los cuales un sistema tiende a evolucionar para una amplia variedad de condiciones iniciales del sistema. Los valores del sistema que se acercan lo suficiente a los valores atractores permanecen cerca, aunque sean levemente perturbados.

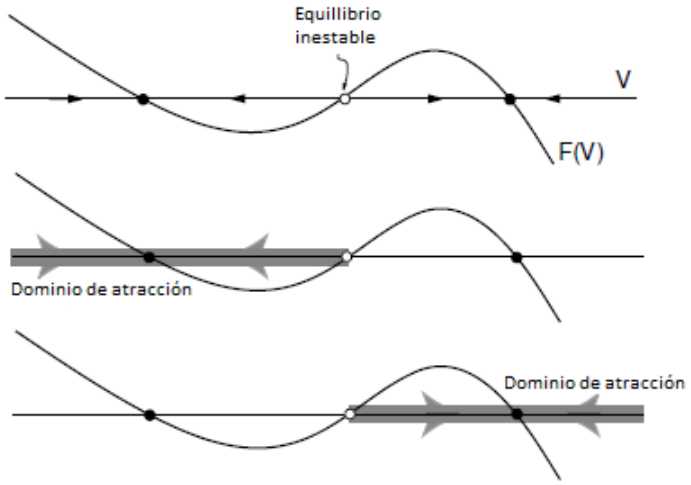


Fig. A.3. Dos dominios de atracción en un sistema unidimensional son separados por equilibrio inestable.

Los puntos atractores aparecen únicamente en sistema unidimensionales y presentan un conjunto de valores conocidos como *dominio de atracción,* los cuales agrupan todas las condiciones iniciales que conducen al punto atractor. La fig.A.3 muestra los dominios de atracción de dos equilibrios estables. El equilibrio inestable de enmedio siempre es una frontera de los dominios de atracción.

A.3 Ciclos límite

Una trayectoria que forma un lazo cerrado es llamada trayectoria periódica u órbita periódica. A veces las trayectorias periódicas son aisladas como el punto centro de la fig.A.2. y otras veces son parte de un lazo continuo como la fig.A.4 (izquierda). A una trayectoria periódica aislada se le conoce como *ciclo limite.*

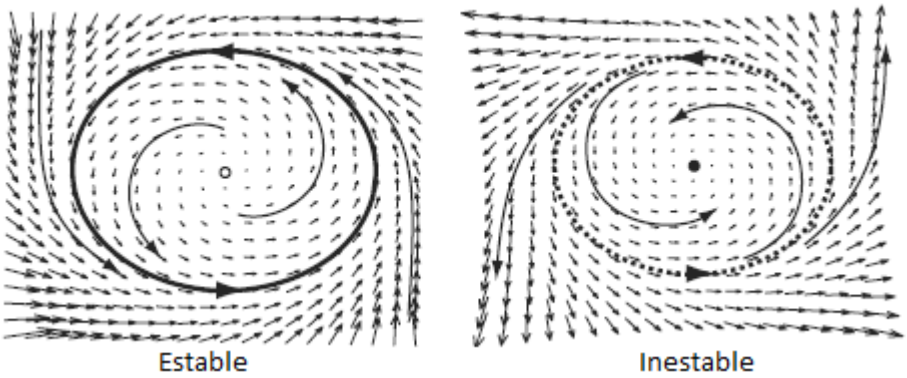


Fig. A.4. Ciclos límite (órbitas periódicas)

La existencia de un ciclo límite es una característica importante de los sistemas bidimensionales que no existe en $\mathbb{R}^1$. Si el punto inicial está en un ciclo límite, entonces la solución $\big(x(t), y(t)\big)$ se mantiene en el ciclo por siempre, y el sistema exhibe un comportamiento periódico, esto es $x(t) = x(t+T)$ y $y(t) = y(t+T)$ para algún valor $T > 0$. El valor de T mínimo para el cual esta igualdad se mantiene es llamado *periodo* del ciclo límite. Se dice que un ciclo límite es asintóticamente estable si cualquiera de sus trayectorias con el punto inicial lo suficientemente cerca del ciclo se mueve hacía el ciclo como $t \to \infty$. Tales ciclos son a veces llamados *atractores de ciclo límite,* ya que "atraen" todas las trayectorias cercanas.

A.3.1 Bifurcación Silla-Nodo

Una bifurcación silla-nodo es una colisión y desaparición de dos equilibrios en un sistema dinámico. En los sistemas autónomos generados por ecuaciones diferenciales ordinarias, esto ocurre cuando el equilibrio crítico tiene un valor

propio igual a cero. Éste fenómeno es también llamado bifurcación de punto de doblez o límite.

Para explicar lo anterior, considérese un sistema autónomo de ecuaciones diferenciales ordinarias

$$\dot{x} = f(x, \alpha), \quad x \in \mathbb{R}^n$$

Dependiente de un parámetro $\alpha \in \mathbb{R}$, donde f es infinitamente diferenciable (función suave).

- Supóngase que en $\alpha = 0$ el sistema tiene un equilibrio $x^0 = 0$.
- Supóngase además que su matriz jacobiana $A_0 = f_x(0,0)$ tiene un valor propio simple $\lambda_1 = 0$.

Entonces, de manera genérica, cuando α pasa por $\alpha = 0$, dos equilibrios colisionan y forman un equilibrio crítico silla-nodo (caso $\beta = 0$ en la fig.A.5), y desaparecen. Esta bifurcación es caracterizada por una sola condición de bifurcación $\lambda_1 = 0$ y aparece genéricamente en familias de ecuaciones diferenciales ordinarias suaves de un solo parámetro. El punto de equilibrio x^0 es una raíz doble de la ecuación $f(x, 0) = 0$.

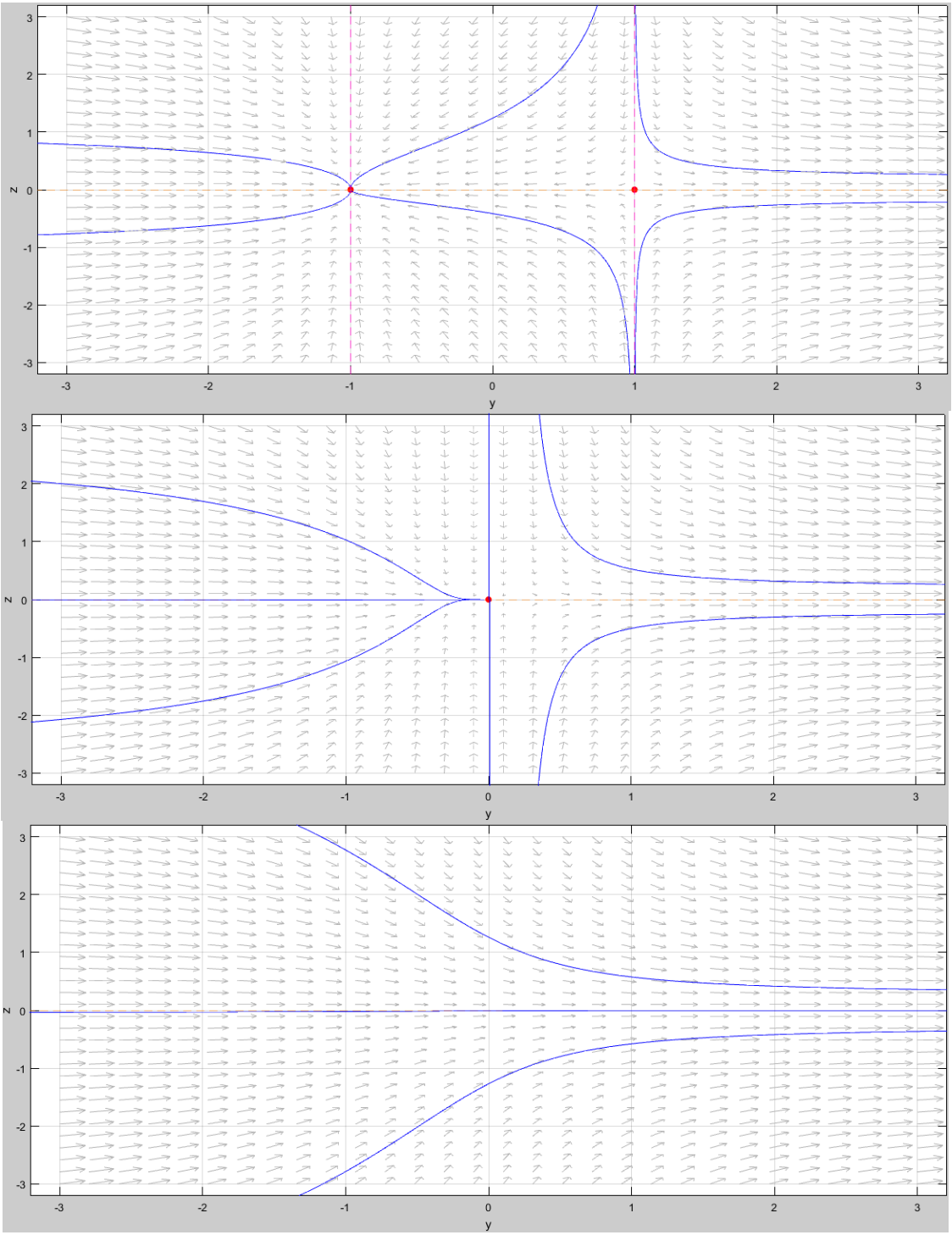


Fig. A.5. Bifurcación silla-nodo usando el sistema de ecuaciones diferenciales ordinarias $\dot{y} = \beta + y;\ \dot{z} = z$. En la figura superior se tiene $\beta < 0$ lo que genera la aparición de dos equilibrios, el izquierdo es un nodo mientras que el de la derecha es un punto silla. En la figura del centro se hace $\beta = 0$ por lo que ambos puntos equilibrios se acercan y finalmente en la figura inferior, $\beta > 0$ lo que da como resultado una colisión y desaparición de ambos equilibrios.

A.3.2 Bifurcación Andronov-Hopf

La bifurcación Andronov-Hopf es la generación de un ciclo límite desde un equilibrio en un sistema dinámico generado por ecuaciones diferenciales ordinarias, cuando dicho equilibrio cambia su estabilidad a través de un par de valores propios puramente imaginarios. Esta bifurcación puede ser *super-crítica* o *subcrítica.*

Considérese un sistema autónomo de ecuaciones diferenciales ordinarias:

$$\dot{x} = f(x, \alpha), x \in \mathbb{R}^n$$

dependiente de un parámetro $\alpha \in \mathbb{R}$, donde f es infinitamente diferenciable (función suave).

- Supóngase que para todo $|\alpha|$ lo suficientemente pequeño, el sistema tiene una familia de equilibrios $x^0(\alpha)$.
- Supóngase además que su matriz jacobiana $A(\alpha) = f_x(x^0(\alpha), \alpha)$, tiene un par de valores propios complejos $\lambda_{1,2}(\alpha) = \mu(\alpha) \pm i\omega(\alpha)$, que se convierten en puramente imaginarios cuando $\alpha = 0$, esto es $\mu(0) = 0$ y $\omega(0) = \omega_0 > 0$.

Entonces, de manera genérica, cuando α pasa por $\alpha = 0$, el equilibrio cambia de estabilidad y un único ciclo límite bifurca a partir de él. Esta bifurcación se caracteriza por una sola condición de bifurcación real $\lambda_{1,2} = 0$ y aparece genéricamente en familias de ecuaciones diferenciales ordinarias suaves de un solo parámetro.

Para describir la bifurcación de manera analítica, considérese el sistema con $n = 2$

$$\dot{x_1} = f_1(x_1, x_2, \alpha),$$
$$\dot{x_2} = f_2(x_1, x_2, \alpha)$$

Y si las siguientes condiciones de no degeneración se mantienen:

- $l_1(0) \neq 0$, donde $l_1(\alpha)$ es el *primer coeficiente de Lyapunov*
- $\mu'(0) \neq 0$,

Entonces el sistema es topológicamente equivalente localmente cerca del equilibrio a la forma normal

$$\dot{y_1} = \beta y_1 - y_2 + \sigma y_1(y_1^2 + y_2^2),$$
$$\dot{y_2} = y_1 + \beta y_2 + \sigma y_2(y_1^2 + y_2^2)$$

donde $y = (y_1, y_2)^T \in \mathbb{R}^2$, $\beta \in \mathbb{R}$, y $\sigma = \pm 1$.

- Si $\sigma = -1$, la forma normal tiene equilibrio en el origen, el cual es asintóticamente estable para valores de $\beta \leq 0$ e inestable para $\beta > 0$. Además, hay un único y estable ciclo límite circular que existe para $\beta > 0$ y tiene radio de $\sqrt{\beta}$. A esta forma se le conoce como bifurcación *super-crítica* Andronov-Hopf y se puede observar en la fig.A.6.
- Si $\sigma = +1$, el origen en la forma normal es asintotícamente estable para valores de $\beta < 0$ e inestable para $\beta \geq 0$, mientras que un único e inestable ciclo límite existe para $\beta < 0$. A esta forma se le conoce como bifurcación *subcrítica* Andronov-Hopf y se puede observar en la fig.A.7.

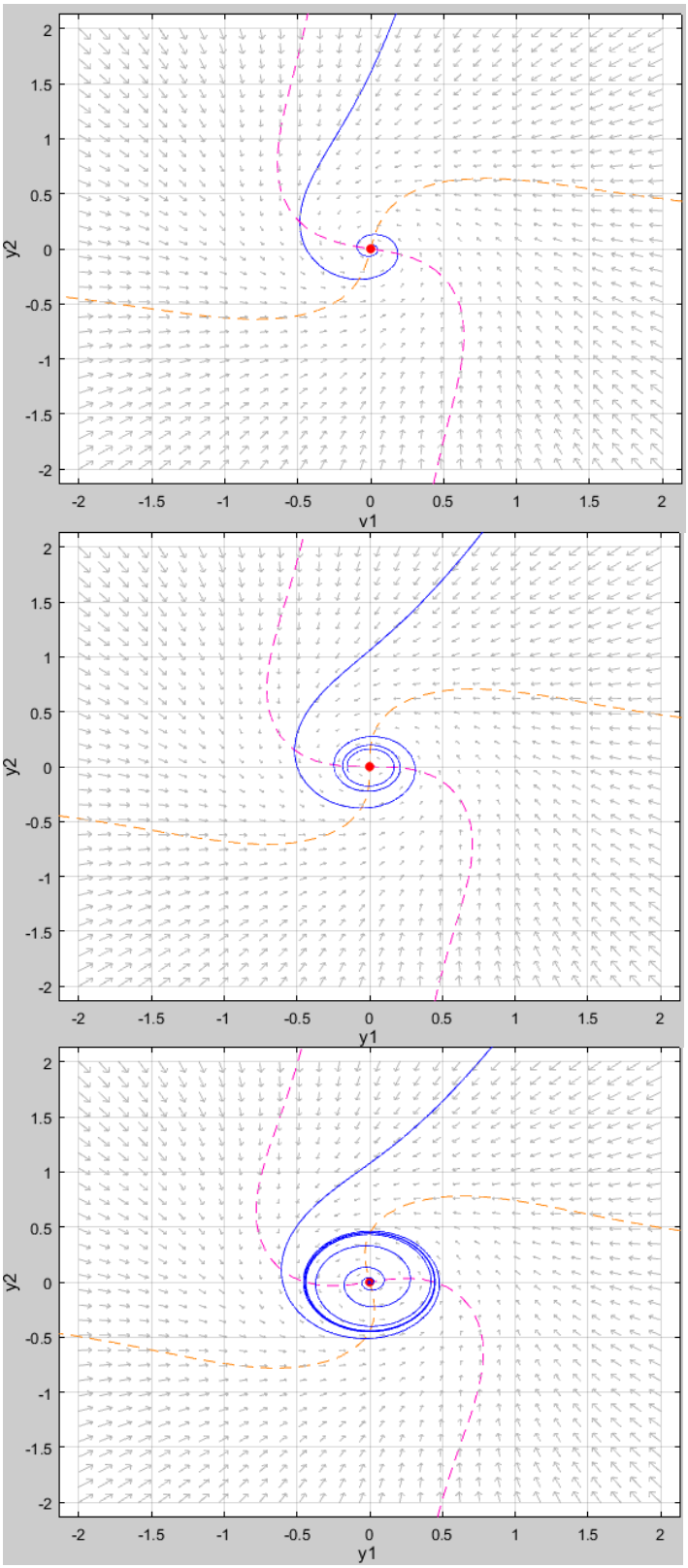


Fig. A.6. Bifurcación Andronov-Hopf super-crítica del sistema de ecuaciones ordinarias descrito anteriormente. La figura superior tiene un valor de $\beta < 0$, la del centro tiene $\beta = 0$ y la inferior tiene $\beta > 0$.

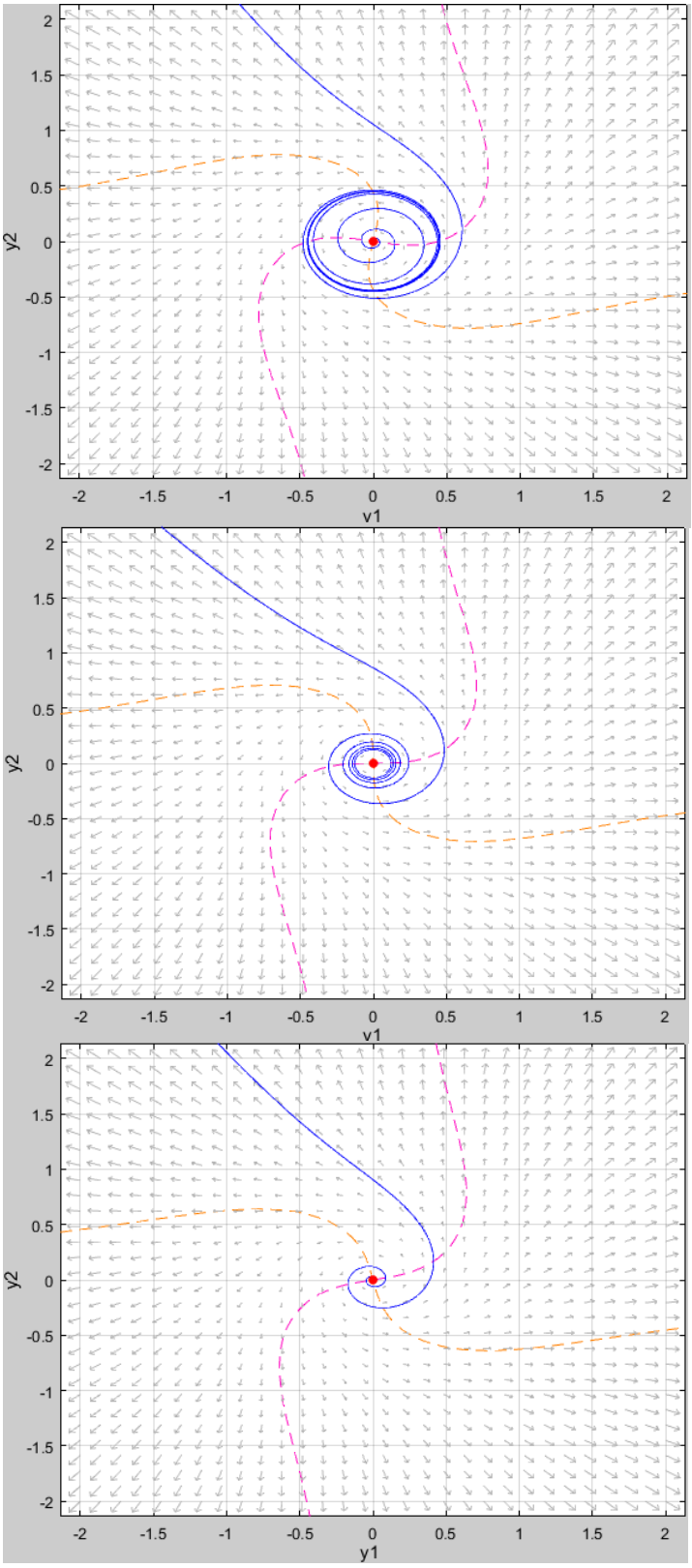


Fig. A.7. Bifurcación Andronov-Hopf subcrítica del sistema de ecuaciones ordinarias descrito anteriormente. La figura superior tiene un valor de $\beta < 0$, la del centro tiene $\beta = 0$ y la inferior tiene $\beta > 0$.

B APÉNDICE B

B.1 Códigos de Matlab de la red neuronal pulsada

B.1.1 Implementación STDP

```
clc, clear, close all;
A_p = 0.005; %0.005
A_n = 0.009; %0.009
tau_p = 15; %15
tau_n = 20; %20
dt = 0.1;
T = ceil(2100/dt);
M_n = zeros(T,1);
P_p = zeros(T,1);
pre = zeros(T,1);
pos = zeros(T,1);
wij = zeros(T,1);
k = 1;
l = 1;
g_aux = 0;
M_nn = 0;
P_pp = 0;
g_max = 0.035;
dly_v = [20 130 240 350 460 570 680 790 900 1010 1120 1230 1340 1450 1560
1670 1780 0];
v_t = [-80 -70 -60 -50 -40 -30 -20 -10 0 10 20 30 40 50 60 70 80 0];
[ss, aa] = size(dly_v);
dw = zeros(aa,1);
tt = ((1:T)*dt)';
for t=1:T

%Entrada Regular
    if t*dt == dly_v(l)
        pre_out = 1;
        l = l + 1;
    else
        pre_out = 0;
    end
    if t*dt == 100*k
        pos_out = 1;
        k = k + 1;
    else
        pos_out = 0;
    end
    pre(t) = pre_out*30;
    pos(t) = pos_out*30;

    if pre_out == 1 %Pulso presinaptico
        P_pp = P_pp + A_p; %Pa
        g_aux = g_aux + M_nn*g_max;
        if g_aux < 0
            g_aux = 0;
        end
    end

    if pos_out == 1 %Pulso postsinaptico
        M_nn = M_nn - A_n; %M
        g_aux = g_aux + P_pp*g_max;
        if g_aux > g_max
```

```
            g_aux = g_max;
        end
    end

    M_nn = (1-dt/tau_n)*M_nn;
    P_pp = (1-dt/tau_p)*P_pp;
    M_n(t) = M_nn;
    P_p(t) = P_pp;
    wij(t) = g_aux;
    dw(l) = g_aux;

end
wij(15000:end) = 0;
dw(14:18) = 0;
pot = dw(1:9);
dep = -dw(10:18);
STDP = cat(1,pot,dep)';
subplot(5,1,1)
plot((0:T-1)*dt, P_p,'g');
xlim([0 1700])
ylabel('[nS]');
title('P_{a} Plot (Conductance)');
subplot(5,1,2)
plot((0:T-1)*dt, pre,'r');
title('Membrane Voltage of Presynaptic Neuron');
xlim([0 1700])
ylim([0 30.5])
ylabel('[mV]');
subplot(5,1,3)
plot((0:T-1)*dt, pos,'b');
title('Membrane Voltage of Postsynaptic Neuron');
xlim([0 1700])
ylim([0 30.5])
ylabel('[mV]');
subplot(5,1,4)
plot((0:T-1)*dt, M_n,'m');
title('M_{a} Plot (Conductance)');
xlim([0 1700])
ylabel('[nS]');
subplot(5,1,5)
plot((0:T-1)*dt, wij/g_max*100,'k');
title('Synaptic Weight Change');
xlabel('Time [ms]');
ylabel('g/g_{max} [%]');
xlim([0 1700])
figure;
plot(v_t(1:17),STDP(1:17)/g_max*100,'.','MarkerSize',10)
title('STDP Implementation Plot');
xlabel('t_{pre}-t_{post} [ms]');
ylabel('g/g_{max} [%]');
table = [tt pre pos P_p M_n wij];
```

B.1.2 Entrenamiento de la red con STDP

```
clc, clear, close all;
dt = 0.1;
a_e = 0.02;
b_e = -0.1;
c_e = -55;
d_e = 6;
a_in = 0.02;
b_in = 0.25;
c_in = -65;
d_in = 2;
T = ceil(1000/dt);
v_e1 = zeros(T,1);
u_e1 = zeros(T,1);
v_e2 = zeros(T,1);
u_e2 = zeros(T,1);
v_in1 = zeros(T,1);
u_in1 = zeros(T,1);
v_in2 = zeros(T,1);
u_in2 = zeros(T,1);
v(1) = -70;
u(1)= -14;
n_in = 8;
rate = 50*1e-3; %50Hz
tau_AMPA_m_e1 = 5;
tau_NMDA_m_e1 = 150;
tau_GABAA_m_e1 = 6;
tau_GABAB_m_e1 = 15;
tau_AMPA_wta_e1 = 5;
tau_NMDA_wta_e1 = 150;
tau_GABAA_wta_e1 = 6;
tau_GABAB_wta_e1 = 15;
tau_AMPA_wta_in1 = 5;
tau_NMDA_wta_in1 = 150;
tau_GABAA_wta_in1 = 6;
tau_GABAB_wta_in1 = 15;
tau_AMPA_wta_e2 = 5;
tau_NMDA_wta_e2 = 150;
tau_GABAA_wta_e2 = 6;
tau_GABAB_wta_e2 = 15;
tau_AMPA_m_e2 = 5;
tau_NMDA_m_e2 = 150;
tau_GABAA_m_e2 = 6;
tau_GABAB_m_e2 =  15;
tau_AMPA_wta_in2 = 5;
tau_NMDA_wta_in2 = 150;
tau_GABAA_wta_in2 = 6;
tau_GABAB_wta_in2 = 15;
p_m_e1 = zeros(n_in,n_in);
p_m_e2 = zeros(n_in,n_in);
g_AMPA_m_ex1 = zeros(n_in, n_in);
g_NMDA_m_ex1 = zeros(n_in,n_in);
g_GABAA_m_ex1 = zeros(n_in,n_in);
g_GABAB_m_ex1 = zeros(n_in,n_in);
g_AMPA_m_ex2 = zeros(n_in, n_in);
g_NMDA_m_ex2 = zeros(n_in,n_in);
g_GABAA_m_ex2 = zeros(n_in,n_in);
g_GABAB_m_ex2 = zeros(n_in,n_in);
g_AMPA_wta_e1 = 0;
g_NMDA_wta_e1 = 0;
g_GABAA_wta_e1 = 0;
```

```
61 g_GABAB_wta_e1 = 0;
62 g_AMPA_wta_in1 = 0;
63 g_NMDA_wta_in1 = 0;
64 g_GABAA_wta_in1 = 0;
65 g_GABAB_wta_in1 = 0;
66 g_AMPA_wta_e2 = 0;
67 g_NMDA_wta_e2 = 0;
68 g_GABAA_wta_e2 = 0;
69 g_GABAB_wta_e2 = 0;
70 g_AMPA_wta_in2 = 0;
71 g_NMDA_wta_in2 = 0;
72 g_GABAA_wta_in2 = 0;
73 g_GABAB_wta_in2 = 0;
74 I_in = zeros(T,1);
75 e1_in=zeros(T,1);
76 e2_in=zeros(T,1);
77 T1 = 0;
78 T2 = 0;
79 DT1 = 0;
80 DT2 = 0;
81 dc1 = zeros(n_in, n_in);
82 dc2 = zeros(n_in, n_in);
83 Pot_Flag_1 = 0;
84 Pot_Flag_2 = 0;
85 Dep_Flag_1 = 0;
86 Dep_Flag_2 = 0;
87 A_p = 0.005;
88 A_n = 0.009;
89 tau_p = 15;
90 tau_n = 20;
91 g_max = 0.18;
92 k = 1;
93 l = 1;
94 n_patts = 2000; %500 %2000 %5000
95 V_P = ones(n_in, n_in);
96 H_P = ones(n_in, n_in);
97 c1_plot = zeros(T,1);
98 c2_plot = zeros(T,1);
99 aux_t1 = zeros(T,1);
100 aux_t2 = zeros(T,1);
101 D_ex1 = zeros(n_in, n_in);
102 P_ex1 = zeros(n_in, n_in);
103 D_ex2 = zeros(n_in, n_in);
104 P_ex2 = zeros(n_in, n_in);
105 g_aux_ex1 = zeros(n_in, n_in);
106 g_aux_ex2 = zeros(n_in, n_in);
107 PP_ex1 = zeros(T,1);
108 DD_ex1 = zeros(T,1);
109 PP_ex2 = zeros(T,1);
110 DD_ex2 = zeros(T,1);
111 plot_c1_1 = zeros(n_patts,1);
112 plot_c1_2 = zeros(n_patts,1);
113 plot_c2_1 = zeros(n_patts,1);
114 plot_c2_2 = zeros(n_patts,1);
115 plot_ev_PPex1 = zeros(n_patts*T,1);
116 plot_ev_DDex1 = zeros(n_patts*T,1);
117 plot_ev_e1in = zeros(n_patts*T,1);
118 plot_ev_e1out = zeros(n_patts*T,1);
119 plot_ev_v_pin1 = zeros(n_patts*T,1);
120 plot_ev_PPex2 = zeros(n_patts*T,1);
121 plot_ev_DDex2 = zeros(n_patts*T,1);
122 plot_ev_e2in = zeros(n_patts*T,1);
```

```
123 plot_ev_e2out = zeros(n_patts*T,1);
124 plot_ev_v_pin2 = zeros(n_patts*T,1);
125 plot_w_e1 = zeros(n_patts,n_in,n_in);
126 plot_w_e2 = zeros(n_patts,n_in,n_in);
127 rate_t2 = zeros(n_patts,1);
128 rate_t1 = zeros(n_patts,1);
129 v_p_in1 = zeros(T,1);
130 v_p_in2 = zeros(T,1);
131 aux_sr1 = 0;
132 ind = 1;
133 vin = 9;
134 rr = 1;
135 M = dlmread('Patts.txt');
136 tau_0 = 1;
137 V_0 = 100;
138 for i=1:n_patts
139     V_PS{i} = M(ind:ind+7,:);
140     H_PS{i} = M(vin:vin+7,:);
141     ind = ind + 16;
142     vin = vin + 16;
143 end
144 ZZ = zeros(n_in,n_in);
145 c_e1 = [0.0028    0.0002    0.0006    0.0014    0.0008    0.0024
0.0001    0.0023
146         0.0003    0.0003    0.0002    0.0000    0.0038    0.0018
0.0004    0.0031
147         0.0000    0.0003    0.0009    0.0000    0.0036    0.0015
0.0003    0.0002
148         0.0022    0.0003    0.0013    0.0002    0.0013    0.0012
0.0029    0.0034
149         0.0003    0.0002    0.0014    0.0025    0.0025    0.0006
0.0009    0.0036
150         0.0000    0.0004    0.0000    0.0007    0.0000    0.0018
0.0020    0.0001
151         0.0036    0.0015    0.0021    0.0000    0.0027    0.0000
0.0005    0.0008
152         0.0025    0.0003    0.0003    0.0028    0.0024    0.0013
0.0004    0.0013];
153
154 c_e2 = [0.0008    0.0001    0.0015    0.0002    0.0027    0.0034
0.0008    0.0014
155         0.0000    0.0015    0.0033    0.0023    0.0019    0.0009
0.0028    0.0034
156         0.0004    0.0037    0.0000    0.0000    0.0029    0.0031
0.0018    0.0038
157         0.0008    0.0010    0.0001    0.0007    0.0002    0.0001
0.0024    0.0007
158         0.0035    0.0011    0.0000    0.0004    0.0002    0.0002
0.0023    0.0000
159         0.0016    0.0011    0.0006    0.0037    0.0010    0.0002
0.0034    0.0015
160         0.0001    0.0039    0.0028    0.0006    0.0008    0.0015
0.0000    0.0004
161         0.0012    0.0023    0.0014    0.0023    0.0022    0.0012
0.0019    0.0028];
162
163 Col_Mat = {[102,0,51]   [51,102,0]    [102,52,0]    [0,102,102]
[0,51,102]    [0,0,102]     [51,0,102]    [102,0,102];
164            [153,0,76]   [76,153,0]    [153,76,0]    [0,153,153]
[0,76,253]    [0,0,153]     [76,0,153]    [153,0,153];
165            [204,0,102]   [102,204,0]   [204,102,0]   [0,204,204]
[0,102,204]   [0,0,204]     [102,0,204]   [204,0,204];
```

```
166             [255,0,127]   [128,255,0]   [255,128,0]   [0,255,255]
[0,128,255]   [0,0,255]     [127,0,255]   [255,0,255];
167             [255,51,153]  [153,255,0]   [255,153,51]  [51,255,255]
[51,153,255]  [51,51,255]   [153,51,255]  [255,51,255];
168             [255,102,178] [178,255,102] [255,178,102] [102,255,255]
[102,178,255] [102,102,255] [178,102,255] [255,102,255];
169             [255,153,204] [204,255,153] [255,204,153] [153,255,255]
[153,204,255] [153,153,255] [204,153,255] [255,153,255];
170             [255,204,229] [229,255,204] [255,229,204] [204,255,255]
[204,229,255] [204,204,255] [229,204,255] [255,204,255]};
171 HHHH=imagesc(c_e1);
172 axis equal off
173 colormap gray
174 colorbar
175 figure;
176 VVVV=imagesc(c_e2);
177 axis equal off
178 colormap gray
179 colorbar
180 figure;
181 tt = 1;
182 for it=1:n_patts
183     for t=1:T-1
184
185         %Entrada uniforme excitatoria 1
186         if mod(t,200) == 0 || t == 1 %20 ms - 50 Hz
187             p_m_e1 = V_PS{it};
188             p_m_e2 = H_PS{it};
189         else
190             p_m_e1 = ZZ;
191             p_m_e2 = ZZ;
192         end
193          e1_in(t) = p_m_e1(4,4);
194          e2_in(t) = p_m_e2(4,4);
195         if v_in2(t) >= 30
196             p_e1 = 1;
197         else
198             p_e1 = 0;
199         end
200         %Neurona excitatoria 1
201         %Se actualiza la conductancia de la matriz de entrada
202         g_AMPA_m_ex1 = g_AMPA_m_ex1 + c_e1.*p_m_e1;
203         g_NMDA_m_ex1 = g_NMDA_m_ex1 + c_e1.*p_m_e1;
204         g_GABAA_m_ex1 = g_GABAA_m_ex1 + c_e1.*p_m_e1;
205         g_GABAB_m_ex1 = g_GABAB_m_ex1 + c_e1.*p_m_e1;
206         g_AMPA_m_ex1 = (1-dt/tau_AMPA_m_e1)*g_AMPA_m_ex1; %Exponencial
207         g_NMDA_m_ex1 = (1-dt/tau_NMDA_m_e1)*g_NMDA_m_ex1; %Exponencial
208         g_GABAA_m_ex1 = (1-dt/tau_GABAA_m_e1)*g_GABAA_m_ex1;
%Exponencial
209         g_GABAB_m_ex1 = (1-dt/tau_GABAB_m_e1)*g_GABAB_m_ex1;
%Exponencial
210         g_AMPA_m_te1 = sum(sum(g_AMPA_m_ex1));
211         g_NMDA_m_te1 = sum(sum(g_NMDA_m_ex1));
212         g_GABAA_m_te1 = sum(sum(g_GABAA_m_ex1));
213         g_GABAB_m_te1 = sum(sum(g_GABAB_m_ex1));
214         %Se actualiza la conductancia de la capa WTA
215         g_AMPA_wta_e1 = g_AMPA_wta_e1 + p_e1;
216         g_NMDA_wta_e1 = g_NMDA_wta_e1 + p_e1;
217         g_GABAA_wta_e1 = g_GABAA_wta_e1 + p_e1;
218         g_GABAB_wta_e1 = g_GABAB_wta_e1 + p_e1;
219         g_AMPA_wta_e1 = (1-dt/tau_AMPA_wta_e1)*g_AMPA_wta_e1;
%Exponencial
```

```
220         g_NMDA_wta_e1 = (1-dt/tau_NMDA_wta_e1)*g_NMDA_wta_e1;
%Exponencial
221         g_GABAA_wta_e1 = (1-dt/tau_GABAA_wta_e1)*g_GABAA_wta_e1;
%Exponencial
222         g_GABAB_wta_e1= (1-dt/tau_GABAB_wta_e1)*g_GABAB_wta_e1;
%Exponencial
223         %Se calcula la corriente de sinapsis de cada entrada
224         I_syn_m_e1 = g_AMPA_m_te1.*(v_e1(t)-
0)+g_NMDA_m_te1.*(((v_e1(t)+80)/60)^2/(1+((v_e1(t)+80)/60).^2)).*(v_e1(t)-
0)+g_GABAA_m_te1.*(v_e1(t)+70)+g_GABAB_m_te1.*(v_e1(t)+90);
225         I_syn_wta_e1 = g_AMPA_wta_e1.*(v_e1(t)-
v_in2(t))+g_NMDA_wta_e1.*(((v_e1(t)+80)/60)^2/(1+((v_e1(t)+80)/60).^2)).*(v_
e1(t)-v_in2(t))+g_GABAA_wta_e1.*(v_e1(t)+70)+g_GABAB_wta_e1.*(v_e1(t)+90);
226         Iapp_e1 = -(I_syn_m_e1+I_syn_wta_e1);
227         I_in(t) = -Iapp_e1;
228         %Se actualiza la ecuación diferencial
229         if v_e1(t)<30
230             dv_e1=(0.04*v_e1(t)+4.1)*v_e1(t)+108-u_e1(t)+Iapp_e1;
231             v_e1(t+1) = v_e1(t) + dv_e1*dt;
232             du_e1 = a_e*(b_e*v_e1(t)-u_e1(t));
233             u_e1(t+1)=u_e1(t)+dt*du_e1;
234         else
235             v_e1(t) = 30;
236             v_e1(t+1) = c_e;
237             u_e1(t+1) = u_e1(t)+d_e;
238         end
239         if v_e1(t) >= 30
240             p_in1 = 1;
241         else
242             p_in1 = 0;
243         end
244         v_p_in1(t) = p_in1;
245         %Neurona inhibitoria 1 de la capa WTA
246         g_AMPA_wta_in1 = g_AMPA_wta_in1 + p_in1;
247         g_NMDA_wta_in1 = g_NMDA_wta_in1 + p_in1;
248         g_GABAA_wta_in1 = g_GABAA_wta_in1 + p_in1;
249         g_GABAB_wta_in1 = g_GABAB_wta_in1 + p_in1;
250         g_AMPA_wta_in1 = (1-dt/tau_AMPA_wta_in1)*g_AMPA_wta_in1;
%Exponencial
251         g_NMDA_wta_in1 = (1-dt/tau_NMDA_wta_in1)*g_NMDA_wta_in1;
%Exponencial
252         g_GABAA_wta_in1 = (1-dt/tau_GABAA_wta_in1)*g_GABAA_wta_in1;
%Exponencial
253         g_GABAB_wta_in1 = (1-dt/tau_GABAB_wta_in1)*g_GABAB_wta_in1;
%Exponencial
254         I_syn_m_in1 = g_AMPA_wta_in1.*(v_in1(t)-
v_e1(t))+g_NMDA_wta_in1.*(((v_in1(t)+80)/60)^2/(1+((v_in1(t)+80)/60).^2)).*(
v_in1(t)-
v_e1(t))+g_GABAA_wta_in1.*(v_in1(t)+70)+g_GABAB_wta_in1.*(v_in1(t)+90);
255         Iapp_in1 = -I_syn_m_in1;
256         if v_in1(t)<30
257             dv_in1=(0.04.*v_in1(t)+5).*v_in1(t)+140-u_in1(t)+Iapp_in1;
258             v_in1(t+1) = v_in1(t) + dv_in1.*dt;
259             du_in1 = a_in.*(b_in.*v_in1(t)-u_in1(t));
260             u_in1(t+1)=u_in1(t)+dt.*du_in1;
261         else
262             v_in1(t) = 30;
263             v_in1(t+1) = c_in;
264             u_in1(t+1) = u_in1(t)+d_in;
265         end
266
267         if v_e2(t) >= 30
```

```
268             p_e2 = 1;
269         else
270             p_e2 = 0;
271         end
272         %Neurona excitatoria 2
273         %Se actualiza la conductancia de la matriz de entrada
274         g_AMPA_m_ex2 = g_AMPA_m_ex2 + c_e2.*p_m_e2;
275         g_NMDA_m_ex2 = g_NMDA_m_ex2 + c_e2.*p_m_e2;
276         g_GABAA_m_ex2 = g_GABAA_m_ex2 + c_e2.*p_m_e2;
277         g_GABAB_m_ex2 = g_GABAB_m_ex2 + c_e2.*p_m_e2;
278         g_AMPA_m_ex2 = (1-dt/tau_AMPA_m_e2)*g_AMPA_m_ex2; %Exponencial
279         g_NMDA_m_ex2 = (1-dt/tau_NMDA_m_e2)*g_NMDA_m_ex2; %Exponencial
280         g_GABAA_m_ex2 = (1-dt/tau_GABAA_m_e2)*g_GABAA_m_ex2;
%Exponencial
281         g_GABAB_m_ex2 = (1-dt/tau_GABAB_m_e2)*g_GABAB_m_ex2;
%Exponencial
282         g_AMPA_m_te2 = sum(sum(g_AMPA_m_ex2));
283         g_NMDA_m_te2 = sum(sum(g_NMDA_m_ex2));
284         g_GABAA_m_te2 = sum(sum(g_GABAA_m_ex2));
285         g_GABAB_m_te2 = sum(sum(g_GABAB_m_ex2));
286         %Se actualiza la conductancia de la capa WTA
287         g_AMPA_wta_e2 = g_AMPA_wta_e2 + p_e2;
288         g_NMDA_wta_e2 = g_NMDA_wta_e2 + p_e2;
289         g_GABAA_wta_e2 = g_GABAA_wta_e2 + p_e2;
290         g_GABAB_wta_e2 = g_GABAB_wta_e2 + p_e2;
291         g_AMPA_wta_e2 = (1-dt/tau_AMPA_wta_e2)*g_AMPA_wta_e2;
%Exponencial
292         g_NMDA_wta_e2 = (1-dt/tau_NMDA_wta_e2)*g_NMDA_wta_e2;
%Exponencial
293         g_GABAA_wta_e2 = (1-dt/tau_GABAA_wta_e2)*g_GABAA_wta_e2;
%Exponencial
294         g_GABAB_wta_e2= (1-dt/tau_GABAB_wta_e2)*g_GABAB_wta_e2;
%Exponencial
295         %Se calcula la corriente de sinapsis de cada entrada
296         I_syn_m_e2 = g_AMPA_m_te2.*(v_e2(t)-
0)+g_NMDA_m_te2.*(((v_e2(t)+80)/60)^2/(1+((v_e2(t)+80)/60).^2)).*(v_e2(t)-
0)+g_GABAA_m_te2.*(v_e2(t)+70)+g_GABAB_m_te2.*(v_e2(t)+90);
297         I_syn_wta_e2 = g_AMPA_wta_e2.*(v_e2(t)-
v_in1(t))+g_NMDA_wta_e2.*(((v_e2(t)+80)/60)^2/(1+((v_e2(t)+80)/60).^2)).*(v_
e2(t)-v_in1(t))+g_GABAA_wta_e2.*(v_e2(t)+70)+g_GABAB_wta_e2.*(v_e2(t)+90);
298         Iapp_e2 = -(I_syn_m_e2+I_syn_wta_e2);
299         I_in(t) = -Iapp_e2;
300         %Se actualiza la ecuación diferencial
301         if v_e2(t)<30
302             dv_e2=(0.04*v_e2(t)+4.1)*v_e2(t)+108-
u_e2(t)+Iapp_e2;%I_syn_m_e2
303             v_e2(t+1) = v_e2(t) + dv_e2*dt;
304             du_e2 = a_e*(b_e*v_e2(t)-u_e2(t));
305             u_e2(t+1)=u_e2(t)+dt*du_e2;
306         else
307             v_e2(t) = 30;
308             v_e2(t+1) = c_e;
309             u_e2(t+1) = u_e2(t)+d_e;
310         end
311         if v_e2(t) >= 30
312             p_in2 = 1;
313         else
314             p_in2 = 0;
315         end
316         %Neurona inhibitoria 2 de la capa WTA
317         g_AMPA_wta_in2 = g_AMPA_wta_in2 + p_in2;
318         g_NMDA_wta_in2 = g_NMDA_wta_in2 + p_in2;
```

```
319         g_GABAA_wta_in2 = g_GABAA_wta_in2 + p_in2;
320         g_GABAB_wta_in2 = g_GABAB_wta_in2 + p_in2;
321         g_AMPA_wta_in2 = (1-dt/tau_AMPA_wta_in2)*g_AMPA_wta_in2;
%Exponencial
322         g_NMDA_wta_in2 = (1-dt/tau_NMDA_wta_in2)*g_NMDA_wta_in2;
%Exponencial
323         g_GABAA_wta_in2 = (1-dt/tau_GABAA_wta_in2)*g_GABAA_wta_in2;
%Exponencial
324         g_GABAB_wta_in2 = (1-dt/tau_GABAB_wta_in2)*g_GABAB_wta_in2;
%Exponencial
325         I_syn_m_in2 = g_AMPA_wta_in2.*(v_in2(t)-
v_e2(t))+g_NMDA_wta_in2.*(((v_in2(t)+80)/60)^2/(1+((v_in2(t)+80)/60).^2)).*(
v_in2(t)-
v_e2(t))+g_GABAA_wta_in2.*(v_e2(t)+70)+g_GABAB_wta_in2.*(v_in2(t)+90);
326         Iapp_in2 = -I_syn_m_in2;
327         if v_in2(t)<30
328             dv_in2=(0.04.*v_in2(t)+5).*v_in2(t)+140-u_in2(t)+Iapp_in2;
329             v_in2(t+1) = v_in2(t) + dv_in2.*dt;
330             du_in2 = a_in.*(b_in.*v_in2(t)-u_in2(t));
331             u_in2(t+1)=u_in2(t)+dt.*du_in2;
332         else
333             v_in2(t) = 30;
334             v_in2(t+1) = c_in;
335             u_in2(t+1) = u_in2(t)+d_in;
336         end
337         %Neurona E1
338         P_ex1 = P_ex1 + A_p*p_m_e1; %Pre
339         P_ex1 = (1-dt/tau_p)*P_ex1;
340         D_ex1 = D_ex1 - A_n*p_in1;  %Pos
341         D_ex1 = (1-dt/tau_n)*D_ex1;
342         %Neurona E2
343         P_ex2 = P_ex2 + A_p*p_m_e2; %Pre
344         P_ex2 = (1-dt/tau_p)*P_ex2;
345         D_ex2 = D_ex2 - A_n*p_in2;  %Pos
346         D_ex2 = (1-dt/tau_n)*D_ex2;
347         for i=1:8
348             for j=1:8
349                 %Neurona E1
350                 if p_m_e1(i,j) == 1  %pulso pre
351                    P_ex1 = P_ex1 + A_p*p_m_e1; %Pre
352                    g_aux_ex1(i,j) = g_aux_ex1(i,j) + g_max.*M_ex1(i,j);
353                    if g_aux_ex1(i,j) < 0
354                        g_aux_ex1(i,j) = 0;
355                    end
356                 end
357                 if p_in1 == 1   %pulso pos
358                    M_ex1 = M_ex1 - A_n*p_in1;  %Pos
359                    g_aux_ex1(i,j) = g_aux_ex1(i,j) + g_max.*P_ex1(i,j);
360                    if g_aux_ex1(i,j) > g_max
361                        g_aux_ex1(i,j) = g_max;
362                    end
363                 end
364                 P_ex1 = (1-dt/tau_p)*P_ex1;
365                 M_ex1 = (1-dt/tau_n)*M_ex1;
366
367                 %Neurona E2
368                 if p_m_e2(i,j) == 1   %pulso pre
369                    P_ex2 = P_ex2 + A_p*p_m_e2; %Pre
370                    g_aux_ex2(i,j) = g_aux_ex2(i,j) + g_max.*M_ex2(i,j);
371                    if g_aux_ex2(i,j) < 0
372                        g_aux_ex2(i,j) = 0;
373                    end
```

```
374                 end
375                 if p_in2 == 1  %pulso pos
376                    M_ex2 = M_ex2 - A_n*p_in2;  %Pos
377                    g_aux_ex2(i,j) = g_aux_ex2(i,j) + g_max.*P_ex2(i,j);
378                    if g_aux_ex2(i,j) > g_max
379                        g_aux_ex2(i,j) = g_max;
380                    end
381                 end
382
383                 P_ex2 = (1-dt/tau_p)*P_ex2;
384                 M_ex2 = (1-dt/tau_n)*M_ex2;
385
386             end
387         end
388         PP_ex1(t)=P_ex1(4,4);
389         DD_ex1(t)=D_ex1(4,4);
390         aux_t1(t)=g_aux_ex1(4,4);
391         PP_ex2(t)=P_ex2(4,1);
392         DD_ex2(t)=D_ex2(4,1);
393         aux_t2(t)=g_aux_ex2(4,1);
394         plot_ev_PPex1(rr) = P_ex1(4,4);
395         plot_ev_DDex1(rr) = D_ex1(4,4);
396         plot_ev_e1in(rr) = p_m_e1(4,4);
397         plot_ev_e2in(rr) = p_m_e2(4,4);
398         plot_ev_e1out(rr) = p_in1;
399         plot_ev_e2out(rr) = p_in2;
400         rr=rr+1;
401     end
402     t1_ind = find(v_e1>=30);
403     t2_ind = find(v_e2>=30);
404     [ax2,ay2] = size(t2_ind);
405     [ax1,ay1] = size(t1_ind);
406     size_t2(it) = ax2;
407     size_t1(it) = ax1;
408     s_rate1 = zeros(size_t1(it),1);
409     s_rate2 = zeros(size_t2(it),1);
410     if size_t1(it) > 3 %Verifica si hay al menos 3 spikes
411         for l=1:size_t1(it)-1
412             s_rate1(l) = (t1_ind(l+1)-t1_ind(l))/10000;%Mide el tiempo
entre spikes
413         end
414         ss_rate1 = s_rate1(2:ax1-1);
415         [s_t1,s_s1] = size(ss_rate1);
416         rate_t1(it) = s_s1/sum(ss_rate1);
417         aux_sr1 = rate_t1(it);
418         s_rate1(l) = aux_sr1;
419     end
420
421     if size_t2(it) > 3 %Verifica si hay al menos 3 spikes
422         for l=1:size_t2(it)-1
423             s_rate2(l) = (t2_ind(l+1)-t2_ind(l))/10000;%Mide el tiempo
entre spikes
424         end
425         ss_rate2 = s_rate2(2:ax2-1);
426         [s_t2,s_s2] = size(s_rate2);
427         rate_t2(it) = s_s2/sum(s_rate2);
428         aux_sr2 = rate_t2(it);
429     end
430     k=1;
431     c_e1 = c_e1 + g_aux_ex1;
432     c_e2 = c_e2 + g_aux_ex2;
433     plot_c1_1(it) = c_e1(4,4);
```

```
    plot_c1_2(it) = c_e1(2,2);
    plot_c2_1(it) = c_e2(4,4);
    plot_c2_2(it) = c_e2(2,2);
    plot_w_e1(it,:,:) = c_e1;
    plot_w_e2(it,:,:) = c_e2;

end
HHH=imagesc(c_e1);
axis equal off
colormap gray
colorbar
figure;
VVV=imagesc(c_e2);
axis equal off
colormap gray
colorbar
figure;
plot(0:n_patts-1,plot_c1_1,'r');
hold on
plot(0:n_patts-1,plot_c1_2,'k');
title('Synaptic Weight During Training of One Vertical Pattern Neuron');
xlabel ( 'Training Time[s]');
ylabel ('Total Synaptic Weight, Cij');
hold off
figure;
plot(0:n_patts-1,plot_c2_1,'r');
hold on
plot(0:n_patts-1,plot_c2_2,'k');
title('Synaptic Weight During Training of One Horizontal Pattern
Neuron');
xlabel ( 'Training Time[s]');
ylabel ('Total Synaptic Weight, Cij');
hold off
tvect = 0:n_patts*T-1;
figure;
hold on
for i = 1:8
    for j = 1:8
        plot(0:n_patts-1,plot_w_e1(:,i,j),'Color',Col_Mat{i,j}/255);
    end
end
title('Synaptic Weight During Training of All Vertical Pattern
Neurons');
xlabel ( 'Training Time[s]');
ylabel ('Total Synaptic Weight, Cij');
hold off
figure;
hold on
for i = 1:8
    for j = 1:8
        plot(0:n_patts-1,plot_w_e2(:,i,j),'Color',Col_Mat{i,j}/255);
    end
end
title('Synaptic Weight During Training of All Horizontal Pattern
Neurons');
xlabel ( 'Training Time[s]');
ylabel ('Total Synaptic Weight, Cij');
plot_e12(1,:) = plot_ev_e1in;
plot_e12(2,:) = plot_ev_e2in;
v_p_in1 = logical(v_p_in1)';
plot_e12 = logical(plot_e12);
plot_e_out(1,:) = plot_ev_e1out;
```

```
493 plot_e_out(2,:) = plot_ev_e2out;
494 plot_e_out = logical(plot_e_out);
495 figure;
496 plotRaster(plot_e_out, tvect/10000);
497 title('Membrane Potentials of Excitatory Neurons During Training');
498 xlabel ( 'Training Time[s]');
499 ylabel ('Trials (#Neurons)');
500 figure;
501 plot(0:n_patts-1,rate_t1,'.','Color','b');
502 figure;
503 plot(0:n_patts-1,rate_t2,'.','Color','r');
```

B.1.3 Prueba de funcionamiento y desempeño de la red neuronal

```
clc, clear, close all;
dt = 0.1;
a_e = 0.02;
b_e = -0.1;
c_e = -55;
d_e = 6;
a_in = 0.02;
b_in = 0.25;
c_in = -65;
d_in = 2;
T = ceil(1000/dt);
v_e1 = zeros(T,1);
u_e1 = zeros(T,1);
v_e2 = zeros(T,1);
u_e2 = zeros(T,1);
v_in1 = zeros(T,1);
u_in1 = zeros(T,1);
v_in2 = zeros(T,1);
u_in2 = zeros(T,1);
n_in = 8;
tau_AMPA_m_e1 = 5;
tau_NMDA_m_e1 = 150;
tau_GABAA_m_e1 = 6;
tau_GABAB_m_e1 = 5;
tau_AMPA_wta_e1 = 5;
tau_NMDA_wta_e1 = 150;
tau_GABAA_wta_e1 = 6;
tau_GABAB_wta_e1 = 5;
tau_AMPA_wta_in1 = 5;
tau_NMDA_wta_in1 = 150;
tau_GABAA_wta_in1 = 6;
tau_GABAB_wta_in1 = 15;
tau_AMPA_wta_e2 = 5;
tau_NMDA_wta_e2 = 150;
tau_GABAA_wta_e2 = 6;
tau_GABAB_wta_e2 = 15;
tau_AMPA_m_e2 = 5;
tau_NMDA_m_e2 = 150;
tau_GABAA_m_e2 = 6;
tau_GABAB_m_e2 =  15;
tau_AMPA_wta_in2 = 5;
tau_NMDA_wta_in2 = 150;
tau_GABAA_wta_in2 = 6;
tau_GABAB_wta_in2 = 15;
p_m_e1 = zeros(n_in,n_in);
p_m_e2 = zeros(n_in,n_in);
g_AMPA_m_ex1 = zeros(n_in, n_in);
g_NMDA_m_ex1 = zeros(n_in,n_in);
g_GABAA_m_ex1 = zeros(n_in,n_in);
g_GABAB_m_ex1 = zeros(n_in,n_in);
g_AMPA_m_ex2 = zeros(n_in, n_in);
g_NMDA_m_ex2 = zeros(n_in,n_in);
g_GABAA_m_ex2 = zeros(n_in,n_in);
g_GABAB_m_ex2 = zeros(n_in,n_in);
g_AMPA_wta_e1 = 0;
g_NMDA_wta_e1 = 0;
g_GABAA_wta_e1 = 0;
g_GABAB_wta_e1 = 0;
g_AMPA_wta_in1 = 0;
g_NMDA_wta_in1 = 0;
```

```
g_GABAA_wta_in1 = 0;
g_GABAB_wta_in1 = 0;
g_AMPA_wta_e2 = 0;
g_NMDA_wta_e2 = 0;
g_GABAA_wta_e2 = 0;
g_GABAB_wta_e2 = 0;
g_AMPA_wta_in2 = 0;
g_NMDA_wta_in2 = 0;
g_GABAA_wta_in2 = 0;
g_GABAB_wta_in2 = 0;
I_in = zeros(T,1);
e1_in=zeros(T,1);
e2_in=zeros(T,1);
V0 = 10;
k = 1;
l = 1;
Z = zeros(2,1);
bb = zeros(2,1);
sum1 = 0;
sum2 = 0;
sum3 = 0;
sum1_2 = 0;
sum2_2 = 0;
sum3_2 = 0;
sum_aux = 0;
V_P1 = [1 1 1 1 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 0 1 1 1 1];
V_P2 = [1 1 1 0 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1;
         1 1 1 1 0 1 1 1;
         1 1 1 0 1 1 1 1];
V_P3 = [1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1];
V_P4 = [1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 0 1 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 0 0 1 1 1;
        1 1 1 0 0 1 1 1];
V_P5 = [1 1 1 0 1 1 1 1;
        1 1 1 0 1 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 1 0 1 1 1;
        1 1 1 0 0 1 1 1;
```

```
123         1 1 1 0 0 1 1 1;
124         1 1 1 1 0 1 1 1;
125         1 1 1 1 0 1 1 1];
126 V_P6 = [1 1 1 0 0 1 1 1;
127         1 1 1 0 1 1 1 1;
128         1 1 1 0 1 1 1 1;
129         1 1 1 0 0 1 1 1;
130         1 1 1 1 0 1 1 1;
131         1 1 1 0 1 1 1 1;
132         1 1 1 0 1 1 1 1;
133         1 1 1 0 1 1 1 1];
134 V_P7 = [1 1 1 0 1 1 1 1;
135         1 1 1 0 0 1 1 1;
136         1 1 1 0 1 1 1 1;
137         1 1 1 1 0 1 1 1;
138         1 1 1 0 1 1 1 1;
139         1 1 1 1 0 1 1 1;
140         1 1 1 0 0 1 1 1;
141         1 1 1 1 0 1 1 1];
142 V_P8 = [1 1 1 0 0 1 1 1;
143         1 1 1 0 1 1 1 1;
144         1 1 1 0 0 1 1 1;
145         1 1 1 1 0 1 1 1;
146         1 1 1 0 1 1 1 1;
147         1 1 1 0 0 1 1 1;
148         1 1 1 1 0 1 1 1;
149         1 1 1 0 0 1 1 1];
150 V_P9 = [1 1 1 0 1 1 1 1;
151         1 1 1 0 1 1 1 1;
152         1 1 1 0 1 1 1 1;
153         1 1 1 0 0 1 1 1;
154         1 1 1 0 0 1 1 1;
155         1 1 1 1 0 1 1 1;
156         1 1 1 1 0 1 1 1;
157         1 1 1 0 1 1 1 1];
158 V_P10 = [1 1 1 0 1 1 1 1;
159           1 1 1 1 0 1 1 1;
160           1 1 1 0 1 1 1 1;
161           1 1 1 0 0 1 1 1;
162           1 1 1 0 1 1 1 1;
163           1 1 1 1 0 1 1 1;
164           1 1 1 1 0 1 1 1;
165           1 1 1 1 0 1 1 1];
166 V_P11 = [1 1 1 0 1 1 1 1;
167           1 1 1 0 1 1 1 1;
168           1 1 1 0 0 1 1 1;
169           1 1 1 1 0 1 1 1;
170           1 1 1 0 1 1 1 1;
171           1 1 1 0 0 1 1 1;
172           1 1 1 1 0 1 1 1;
173           1 1 1 1 0 1 1 1];
174 V_P12 = [1 1 1 1 0 1 1 1;
175           1 1 1 1 0 1 1 1;
176           1 1 1 0 1 1 1 1;
177           1 1 1 1 0 1 1 1;
178           1 1 1 0 1 1 1 1;
179           1 1 1 1 0 1 1 1;
180           1 1 1 0 1 1 1 1;
181           1 1 1 1 0 1 1 1];
182 V_P13 = [1 1 1 0 1 1 1 1;
183           1 1 1 0 1 1 1 1;
184           1 1 1 1 0 1 1 1;
```

```
185           1 1 1 1 0 1 1 1;
186           1 1 1 0 1 1 1 1;
187           1 1 1 0 1 1 1 1;
188           1 1 1 1 0 1 1 1;
189           1 1 1 1 0 1 1 1];
190 VH_P = [1 1 1 0 0 1 1 1;
191           1 1 1 0 0 1 1 1;
192           1 1 1 0 0 1 1 1;
193           0 0 0 0 0 0 0 0;
194           0 0 0 0 0 0 0 0;
195           1 1 1 0 0 1 1 1;
196           1 1 1 0 0 1 1 1;
197           1 1 1 0 0 1 1 1];
198 V_P15 = [0 0 1 1 1 1 1 1;
199           0 0 1 1 1 1 1 1;
200           1 0 0 1 1 1 1 1;
201           1 1 1 0 0 1 1 1;
202           1 1 1 0 0 1 1 1;
203           1 1 1 1 0 0 1 1;
204           1 1 1 1 1 1 0 0;
205           1 1 1 1 1 1 0 0];
206
207 H_P1 = [1 1 1 1 1 1 1 1;
208          1 1 1 1 1 1 1 1;
209          1 1 1 1 1 1 1 1;
210          1 0 0 0 0 0 1 0;
211          0 0 0 0 0 0 0 0;
212          1 1 1 1 1 1 1 1;
213          1 1 1 1 1 1 1 1;
214          1 1 1 1 1 1 1 1];
215 H_P2 = [1 1 1 1 1 1 1 1;
216          1 1 1 1 1 1 1 1;
217          1 1 1 1 1 1 1 1;
218          0 0 0 1 0 1 0 0;
219          1 1 0 0 1 0 1 0;
220          1 1 1 1 1 1 1 1;
221          1 1 1 1 1 1 1 1;
222          1 1 1 1 1 1 1 1];
223 H_P3 = [1 1 1 1 1 1 1 1;
224          1 1 1 1 1 1 1 1;
225          1 1 1 1 1 1 1 1;
226          0 0 0 0 1 0 0 0;
227          1 1 1 0 0 1 1 1;
228          1 1 1 1 1 1 1 1;
229          1 1 1 1 1 1 1 1;
230          1 1 1 1 1 1 1 1];
231 H_P4 = [1 1 1 1 1 1 1 1;
232          1 1 1 1 1 1 1 1;
233          1 1 1 1 1 1 1 1;
234          0 0 1 1 0 1 1 0;
235          1 1 0 0 1 0 0 1;
236          1 1 1 1 1 1 1 1;
237          1 1 1 1 1 1 1 1;
238          1 1 1 1 1 1 1 1];
239 H_P5 = [1 1 1 1 1 1 1 1;
240          1 1 1 1 1 1 1 1;
241          1 1 1 1 1 1 1 1;
242          0 0 0 0 0 0 0 0;
243          0 0 1 0 0 1 0 0;
244          1 1 1 1 1 1 1 1;
245          1 1 1 1 1 1 1 1;
246          1 1 1 1 1 1 1 1];
```

```
247 H_P6 = [1 1 1 1 1 1 1 1;
248          1 1 1 1 1 1 1 1;
249          1 1 1 1 1 1 1 1;
250          0 0 1 1 1 1 0 0;
251          0 0 0 0 0 0 0 0;
252          1 1 1 1 1 1 1 1;
253          1 1 1 1 1 1 1 1;
254          1 1 1 1 1 1 1 1];
255 H_P7 = [1 1 1 1 1 1 1 1;
256          1 1 1 1 1 1 1 1;
257          1 1 1 1 1 1 1 1;
258          0 0 1 0 1 1 0 0;
259          1 0 0 1 0 0 1 1;
260          1 1 1 1 1 1 1 1;
261          1 1 1 1 1 1 1 1;
262          1 1 1 1 1 1 1 1];
263 H_P8 = [1 1 1 1 1 1 1 1;
264          1 1 1 1 1 1 1 1;
265          1 1 1 1 1 1 1 1;
266          0 0 0 1 0 0 0 0;
267          0 1 1 0 1 1 0 1;
268          1 1 1 1 1 1 1 1;
269          1 1 1 1 1 1 1 1;
270          1 1 1 1 1 1 1 1];
271 H_P9 = [1 1 1 1 1 1 1 1;
272          1 1 1 1 1 1 1 1;
273          1 1 1 1 1 1 1 1;
274          0 0 0 0 1 1 1 1;
275          0 0 1 1 0 0 0 0;
276          1 1 1 1 1 1 1 1;
277          1 1 1 1 1 1 1 1;
278          1 1 1 1 1 1 1 1];
279 H_P10 = [1 1 1 1 1 1 1 1;
280           1 1 1 1 1 1 1 1;
281           1 1 1 1 1 1 1 1;
282           0 1 0 1 0 1 0 0;
283           0 0 1 0 1 0 1 0;
284           1 1 1 1 1 1 1 1;
285           1 1 1 1 1 1 1 1;
286           1 1 1 1 1 1 1 1];
287 H_P11 = [1 1 1 1 1 1 1 1;
288           1 1 1 1 1 1 1 1;
289           1 1 1 1 1 1 1 1;
290           1 1 0 0 1 1 0 0;
291           0 0 1 1 0 0 1 1;
292           1 1 1 1 1 1 1 1;
293           1 1 1 1 1 1 1 1;
294           1 1 1 1 1 1 1 1];
295 H_P12 = [1 1 1 1 1 1 1 1;
296           1 1 1 1 1 1 1 1;
297           1 1 1 1 1 1 1 1;
298           0 0 0 0 1 1 1 1;
299           1 1 1 1 0 0 0 0;
300           1 1 1 1 1 1 1 1;
301           1 1 1 1 1 1 1 1;
302           1 1 1 1 1 1 1 1];
303 H_P13 = [1 1 1 1 1 1 1 1;
304           1 1 1 1 1 1 1 1;
305           1 1 1 1 1 1 1 1;
306           1 1 1 0 0 1 1 1;
307           0 0 0 0 0 0 0 0;
308           1 1 1 1 1 1 1 1;
```

```
309          1 1 1 1 1 1 1 1;
310          1 1 1 1 1 1 1 1];
311 H_P14 = [1 1 1 0 0 1 1 1;
312          1 1 1 0 0 1 1 1;
313          1 1 1 0 0 1 1 1;
314          0 0 0 0 0 0 0 0;
315          0 0 0 0 0 0 0 0;
316          1 1 1 0 0 1 1 1;
317          1 1 1 0 0 1 1 1;
318          1 1 1 0 0 1 1 1];
319 T_P1 =  [1 1 1 1 1 1 0 0;
320          1 1 1 1 1 0 0 0;
321          1 1 1 1 0 0 0 1;
322          1 1 1 0 0 0 1 1;
323          1 1 0 0 0 1 1 1;
324          1 0 0 0 1 1 1 1;
325          0 0 0 1 1 1 1 1;
326          0 0 1 1 1 1 1 1];
327 T_P2 =  [0 0 1 1 1 1 1 1;
328          0 0 0 1 1 1 1 1;
329          1 0 0 0 1 1 1 1;
330          1 1 0 0 0 1 1 1;
331          1 1 1 0 0 0 1 1;
332          1 1 1 1 0 0 0 1;
333          1 1 1 1 1 0 0 0;
334          1 1 1 1 1 1 0 0];
335
336 T_P3 =  [1 1 0 0 1 1 1 1;
337          1 1 1 1 1 1 1 1;
338          1 1 0 0 1 1 1 1;
339          1 1 1 0 1 1 1 1;
340          1 1 0 1 1 1 1 1;
341          1 1 1 0 1 1 1 1;
342          1 1 0 1 1 1 1 1;
343          1 1 1 0 1 1 1 1];
344
345 T_P4 =  [1 1 1 1 1 1 1 1;
346          1 1 1 1 1 1 1 1;
347          0 0 1 1 0 0 1 0;
348          0 0 1 0 0 1 0 0;
349          1 1 1 1 1 1 1 1;
350          1 1 1 1 1 1 1 1;
351          1 1 1 1 1 1 1 1;
352          1 1 1 1 1 1 1 1];
353
354 T_P5 =  [1 1 1 0 0 1 1 1;
355          1 1 1 0 0 1 1 1;
356          1 1 1 0 0 1 1 1;
357          0 0 0 0 0 0 0 0;
358          0 0 0 0 0 0 0 0;
359          1 1 1 0 0 1 1 1;
360          1 1 1 0 0 1 1 1;
361          1 1 1 0 0 1 1 1];
362 H_PS = {H_P1 H_P2 H_P3 H_P4 H_P5 H_P6 H_P7 H_P8 H_P9 H_P10 H_P11 H_P12
H_P13 H_P14};
363 V_PS = {V_P1 V_P2 V_P3 V_P4 V_P5 V_P6 V_P7 V_P8 V_P9 V_P10 V_P11 V_P12
V_P13 VH_P};
364 HV_PS ={H_P1 VH_P V_P1 H_P2 V_P2 H_P3 V_P3 H_P4 V_P4 H_P5 V_P5 H_P6 VH_P
V_P6 H_P7 V_P7 H_P8 V_P8 H_P9 VH_P H_P10 V_P10 H_P11 V_P11 H_P12 V_P12 H_P13
V_P13 VH_P V_P9 T_P1 T_P2 T_P3 T_P4 T_P5};
365 tr = 35;
366 ZZ = zeros(n_in,n_in);
```

```
367 plot_ev_e1out = zeros(tr*T,1);
368 plot_ev_e1_1out = zeros(tr*T,1);
369 plot_ev_e2out = zeros(tr*T,1);
370 plot_ev_e2_2out = zeros(tr*T,1);
371 plot_v_e1 = zeros(tr*T,1);
372 plot_v_e2 = zeros(tr*T,1);
373 plot_v_in1 = zeros(tr*T,1);
374 plot_v_in2 = zeros(tr*T,1);
375 plot_prob_ex1 = zeros(tr*T,1);
376 plot_prob_ex2 = zeros(tr*T,1);
377 plot_prob_inoutZ = zeros(tr*T,1);
378 sse1 = zeros(tr,1);
379 sse2 = zeros(tr,1);
380 rr = 1;
381 c_e1 = [0.0008    0.0020    0.0006    0.0014    0.0008    0.0024
0.0001    0.0003
382         0.0013    0.0003    0.0002    0.0000    0.0008    0.0018
0.0004    0.0021
383         0.0000    0.0003    0.0009    0.0000    0.0036    0.0015
0.0003    0.0002
384         0.0221    0.0410    0.0301    0.0223    0.0320    0.0380
0.0371    0.0380
385         0.0313    0.0251    0.0401    0.0300    0.0333    0.0401
0.0409    0.0420
386         0.0000    0.0004    0.0000    0.0007    0.0000    0.0018
0.0020    0.0001
387         0.0036    0.0015    0.0021    0.0000    0.0027    0.0000
0.0005    0.0008
388         0.0025    0.0003    0.0003    0.0028    0.0014    0.0033
0.0002    0.0003];
389
390 c_e2 = [0.0008    0.0011    0.0005    0.0200    0.1027    0.0034
0.0008    0.0014
391         0.0000    0.0015    0.0033    0.0300    0.0190    0.0009
0.0028    0.0034
392         0.0004    0.0037    0.0000    0.0380    0.0529    0.0031
0.0008    0.0038
393         0.0008    0.0010    0.0001    0.1000    0.1500    0.0001
0.0024    0.0007
394         0.0015    0.0011    0.0000    0.0404    0.0522    0.0002
0.0013    0.0000
395         0.0016    0.0011    0.0006    0.0470    0.0410    0.0002
0.0034    0.0015
396         0.0011    0.0039    0.0028    0.0400    0.0701    0.0015
0.0000    0.0004
397         0.0012    0.0023    0.0014    0.0523    0.1220    0.0012
0.0019    0.0028];
398
399 c_ee(:,:,1) = c_e1;
400 c_ee(:,:,2) = c_e2;
401 for it=1:tr
402     for t=1:T-1
403         if t == 1
404             v_e1(t) = -65;
405             v_in1(t) = -80;
406             v_e2(t) = -65;
407             v_in2(t) = -80;
408         end
409         %Entrada uniforme excitatoria 1
410         if mod(t,200) == 0 || t == 1 %20 ms - 50 Hz
411             p_m_e1 = HV_PS{it};
412             k=k+1;
```

```
413         else
414             p_m_e1 = ZZ;
415         end
416          e1_in(t) = p_m_e1(1,5);
417         %Entrada uniforme excitatoria 2
418         if mod(t,200) == 0 || t == 1 %20 ms - 50 Hz
419             p_m_e2 = HV_PS{it};
420             l=l+1;
421         else
422             p_m_e2 = ZZ;
423         end
424
425
426         if v_in2(t) >= 30
427             p_e1 = 1;
428         else
429             p_e1 = 0;
430         end
431         %Neurona excitatoria 1
432         %Se actualiza la conductancia de la matriz de entrada
433         g_AMPA_m_ex1 = g_AMPA_m_ex1 + c_e1.*p_m_e1;
434         g_NMDA_m_ex1 = g_NMDA_m_ex1 + c_e1.*p_m_e1;
435         g_GABAA_m_ex1 = g_GABAA_m_ex1 + c_e1.*p_m_e1;
436         g_GABAB_m_ex1 = g_GABAB_m_ex1 + c_e1.*p_m_e1;
437         g_AMPA_m_ex1 = (1-dt/tau_AMPA_m_e1)*g_AMPA_m_ex1; %Exponencial
438         g_NMDA_m_ex1 = (1-dt/tau_NMDA_m_e1)*g_NMDA_m_ex1; %Exponencial
439         g_GABAA_m_ex1 = (1-dt/tau_GABAA_m_e1)*g_GABAA_m_ex1;
%Exponencial
440         g_GABAB_m_ex1 = (1-dt/tau_GABAB_m_e1)*g_GABAB_m_ex1;
%Exponencial
441         g_AMPA_m_te1 = sum(sum(g_AMPA_m_ex1));
442         g_NMDA_m_te1 = sum(sum(g_NMDA_m_ex1));
443         g_GABAA_m_te1 = sum(sum(g_GABAA_m_ex1));
444         g_GABAB_m_te1 = sum(sum(g_GABAB_m_ex1));
445         %Se actualiza la conductancia de la capa WTA
446         g_AMPA_wta_e1 = g_AMPA_wta_e1 + p_e1;
447         g_NMDA_wta_e1 = g_NMDA_wta_e1 + p_e1;
448         g_GABAA_wta_e1 = g_GABAA_wta_e1 + p_e1;
449         g_GABAB_wta_e1 = g_GABAB_wta_e1 + p_e1;
450         g_AMPA_wta_e1 = (1-dt/tau_AMPA_wta_e1)*g_AMPA_wta_e1;
%Exponencial
451         g_NMDA_wta_e1 = (1-dt/tau_NMDA_wta_e1)*g_NMDA_wta_e1;
%Exponencial
452         g_GABAA_wta_e1 = (1-dt/tau_GABAA_wta_e1)*g_GABAA_wta_e1;
%Exponencial
453         g_GABAB_wta_e1= (1-dt/tau_GABAB_wta_e1)*g_GABAB_wta_e1;
%Exponencial
454         %Se calcula la corriente de sinapsis de cada entrada
455         I_syn_m_e1 = g_AMPA_m_te1.*(v_e1(t)-
0)+g_NMDA_m_te1.*(((v_e1(t)+80)/60)^2/(1+((v_e1(t)+80)/60).^2)).*(v_e1(t)-
0)+g_GABAA_m_te1.*(v_e1(t)+70)+g_GABAB_m_te1.*(v_e1(t)+90);
456         I_syn_wta_e1 = g_AMPA_wta_e1.*(v_e1(t)-
v_in2(t))+g_NMDA_wta_e1.*(((v_e1(t)+80)/60)^2/(1+((v_e1(t)+80)/60).^2)).*(v_
e1(t)-v_in2(t))+g_GABAA_wta_e1.*(v_e1(t)+70)+g_GABAB_wta_e1.*(v_e1(t)+90);
457         Iapp_e1 = -(I_syn_m_e1+I_syn_wta_e1);
458         I_in(t) = -Iapp_e1;
459         %Se actualiza la ecuación diferencial
460         if v_e1(t)<30
461             dv_e1=(0.04*v_e1(t)+4.1)*v_e1(t)+108-u_e1(t)+Iapp_e1;
462             v_e1(t+1) = v_e1(t) + dv_e1*dt;
463             du_e1 = a_e*(b_e*v_e1(t)-u_e1(t));
464             u_e1(t+1)=u_e1(t)+dt*du_e1;
```

```
465         else
466             v_e1(t) = 30;
467             v_e1(t+1) = c_e;
468             u_e1(t+1) = u_e1(t)+d_e;
469         end
470         if v_e1(t) >= 30
471             p_in1 = 1;
472 
473         else
474             p_in1 = 0;
475 
476         end
477         Z1= p_in1;
478         %Neurona inhibitoria 1 de la capa WTA
479         g_AMPA_wta_in1 = g_AMPA_wta_in1 + p_in1;
480         g_NMDA_wta_in1 = g_NMDA_wta_in1 + p_in1;
481         g_GABAA_wta_in1 = g_GABAA_wta_in1 + p_in1;
482         g_GABAB_wta_in1 = g_GABAB_wta_in1 + p_in1;
483         g_AMPA_wta_in1 = (1-dt/tau_AMPA_wta_in1)*g_AMPA_wta_in1;
%Exponencial
484         g_NMDA_wta_in1 = (1-dt/tau_NMDA_wta_in1)*g_NMDA_wta_in1;
%Exponencial
485         g_GABAA_wta_in1 = (1-dt/tau_GABAA_wta_in1)*g_GABAA_wta_in1;
%Exponencial
486         g_GABAB_wta_in1 = (1-dt/tau_GABAB_wta_in1)*g_GABAB_wta_in1;
%Exponencial
487         I_syn_m_in1 = g_AMPA_wta_in1.*(v_in1(t)-
v_e1(t))+g_NMDA_wta_in1.*(((v_in1(t)+80)/60)^2/(1+((v_in1(t)+80)/60).^2)).*(
v_in1(t)-
v_e1(t))+g_GABAA_wta_in1.*(v_e1(t)+70)+g_GABAB_wta_in1.*(v_in1(t)+90);
488         Iapp_in1 = -I_syn_m_in1;
489         if v_in1(t)<30
490             dv_in1=(0.04.*v_in1(t)+5).*v_in1(t)+140-u_in1(t)+Iapp_in1;
491             v_in1(t+1) = v_in1(t) + dv_in1.*dt;
492             du_in1 = a_in.*(b_in.*v_in1(t)-u_in1(t));
493             u_in1(t+1)=u_in1(t)+dt.*du_in1;
494         else
495             v_in1(t) = 30;
496             v_in1(t+1) = c_in;
497             u_in1(t+1) = u_in1(t)+d_in;
498         end
499         if v_in1(t) >= 30
500             p_e2 = 1;
501         else
502             p_e2 = 0;
503         end
504         %Neurona excitatoria 2
505         %Se actualiza la conductancia de la matriz de entrada
506         g_AMPA_m_ex2 = g_AMPA_m_ex2 + c_e2.*p_m_e2;
507         g_NMDA_m_ex2 = g_NMDA_m_ex2 + c_e2.*p_m_e2;
508         g_GABAA_m_ex2 = g_GABAA_m_ex2 + c_e2.*p_m_e2;
509         g_GABAB_m_ex2 = g_GABAB_m_ex2 + c_e2.*p_m_e2;
510         g_AMPA_m_ex2 = (1-dt/tau_AMPA_m_e2)*g_AMPA_m_ex2; %Exponencial
511         g_NMDA_m_ex2 = (1-dt/tau_NMDA_m_e2)*g_NMDA_m_ex2; %Exponencial
512         g_GABAA_m_ex2 = (1-dt/tau_GABAA_m_e2)*g_GABAA_m_ex2;
%Exponencial
513         g_GABAB_m_ex2 = (1-dt/tau_GABAB_m_e2)*g_GABAB_m_ex2;
%Exponencial
514         g_AMPA_m_te2 = sum(sum(g_AMPA_m_ex2));
515         g_NMDA_m_te2 = sum(sum(g_NMDA_m_ex2));
516         g_GABAA_m_te2 = sum(sum(g_GABAA_m_ex2));
517         g_GABAB_m_te2 = sum(sum(g_GABAB_m_ex2));
```

```
518         %Se actualiza la conductancia de la capa WTA
519         g_AMPA_wta_e2 = g_AMPA_wta_e2 + p_e2;
520         g_NMDA_wta_e2 = g_NMDA_wta_e2 + p_e2;
521         g_GABAA_wta_e2 = g_GABAA_wta_e2 + p_e2;
522         g_GABAB_wta_e2 = g_GABAB_wta_e2 + p_e2;
523         g_AMPA_wta_e2 = (1-dt/tau_AMPA_wta_e2)*g_AMPA_wta_e2;
%Exponencial
524         g_NMDA_wta_e2 = (1-dt/tau_NMDA_wta_e2)*g_NMDA_wta_e2;
%Exponencial
525         g_GABAA_wta_e2 = (1-dt/tau_GABAA_wta_e2)*g_GABAA_wta_e2;
%Exponencial
526         g_GABAB_wta_e2= (1-dt/tau_GABAB_wta_e2)*g_GABAB_wta_e2;
%Exponencial
527         %Se calcula la corriente de sinapsis de cada entrada
528         I_syn_m_e2 = g_AMPA_m_te2.*(v_e2(t)-
0)+g_NMDA_m_te2.*(((v_e2(t)+80)/60)^2/(1+((v_e2(t)+80)/60).^2)).*(v_e2(t)-
0)+g_GABAA_m_te2.*(v_e2(t)+70)+g_GABAB_m_te2.*(v_e2(t)+90);
529         I_syn_wta_e2 = g_AMPA_wta_e2.*(v_e2(t)-
v_in1(t))+g_NMDA_wta_e2.*(((v_e2(t)+80)/60)^2/(1+((v_e2(t)+80)/60).^2)).*(v_
e2(t)-v_in1(t))+g_GABAA_wta_e2.*(v_e2(t)+70)+g_GABAB_wta_e2.*(v_e2(t)+90);
530         Iapp_e2 = -(I_syn_m_e2+I_syn_wta_e2);
531         I_in(t) = -Iapp_e2;
532         %Se actualiza la ecuación diferencial
533         if v_e2(t)<30
534             dv_e2=(0.04*v_e2(t)+4.1)*v_e2(t)+108-
u_e2(t)+Iapp_e2;%I_syn_m_e2
535             v_e2(t+1) = v_e2(t) + dv_e2*dt;
536             du_e2 = a_e*(b_e*v_e2(t)-u_e2(t));
537             u_e2(t+1)=u_e2(t)+dt*du_e2;
538         else
539             v_e2(t) = 30;
540             v_e2(t+1) = c_e;
541             u_e2(t+1) = u_e2(t)+d_e;
542         end
543         if v_e2(t) >= 30
544             p_in2 = 1;
545
546         else
547             p_in2 = 0;
548         end
549         Z2 = p_in2;
550         %Neurona inhibitoria 2 de la capa WTA
551         g_AMPA_wta_in2 = g_AMPA_wta_in2 + p_in2;
552         g_NMDA_wta_in2 = g_NMDA_wta_in2 + p_in2;
553         g_GABAA_wta_in2 = g_GABAA_wta_in2 + p_in2;
554         g_GABAB_wta_in2 = g_GABAB_wta_in2 + p_in2;
555         g_AMPA_wta_in2 = (1-dt/tau_AMPA_wta_in2)*g_AMPA_wta_in2;
%Exponencial
556         g_NMDA_wta_in2 = (1-dt/tau_NMDA_wta_in2)*g_NMDA_wta_in2;
%Exponencial
557         g_GABAA_wta_in2 = (1-dt/tau_GABAA_wta_in2)*g_GABAA_wta_in2;
%Exponencial
558         g_GABAB_wta_in2 = (1-dt/tau_GABAB_wta_in2)*g_GABAB_wta_in2;
%Exponencial
559         I_syn_m_in2 = g_AMPA_wta_in2.*(v_in2(t)-
v_e2(t))+g_NMDA_wta_in2.*(((v_in2(t)+80)/60)^2/(1+((v_in2(t)+80)/60).^2)).*(
v_in2(t)-
v_e2(t))+g_GABAA_wta_in2.*(v_e2(t)+70)+g_GABAB_wta_in2.*(v_in2(t)+90);
560         Iapp_in2 = -I_syn_m_in2;
561         if v_in2(t)<30
562             dv_in2=(0.04.*v_in2(t)+5).*v_in2(t)+140-u_in2(t)+Iapp_in2;
563             v_in2(t+1) = v_in2(t) + dv_in2.*dt;
```

```
564             du_in2 = a_in.*(b_in.*v_in2(t)-u_in2(t));
565             u_in2(t+1)=u_in2(t)+dt.*du_in2;
566         else
567             v_in2(t) = 30;
568             v_in2(t+1) = c_in;
569             u_in2(t+1) = u_in2(t)+d_in;
570         end
571        plot_ev_e1out(rr) = p_in2;
572        plot_ev_e1_1out(rr) = p_e1; %Horizontal
573        plot_ev_e2out(rr) = p_in1;
574        plot_ev_e2_2out(rr) = p_e2; %Vertical
575        plot_v_e1(rr) = v_e1(t);
576        plot_v_e2(rr) = v_e2(t);
577        plot_v_in1(rr) = v_in1(t);
578        plot_v_in2(rr) = v_in2(t);
579        plot_prob_ex1(rr) =
exp(v_e1(t)/V0)/(exp(v_e2(t)/V0)+exp(v_e1(t)/V0));
580        plot_prob_ex2(rr) =
exp(v_e2(t)/V0)/(exp(v_e1(t)/V0)+exp(v_e2(t)/V0));
581         for i = 1:8
582             for j = 1:8
583                 res1 = c_ee(i,j,1)*p_m_e1(i,j);
584                 res2 = p_in1*bb(1);
585                 res1_2 = c_ee(i,j,2)*p_m_e2(i,j);
586                 res2_2 = p_in2*bb(2);
587                 sum1 = sum1 + p_in1*res1;
588                 sum2 = sum2 + res2;
589                 sum3 = sum3 + res1;
590                 sum1_2 = sum1_2 + p_in2*res1_2;
591                 sum2_2 = sum2_2 + res2_2;
592                 sum3_2 = sum3_2 + res1_2;
593             end
594         end
595         s_aux = exp(sum1+sum1_2);
596         ss_aux = exp(sum2+sum2_2);
597         sss_aux = exp(bb(1)+bb(2)+sum3+sum3_2);
598         plot_prob_inoutZ(rr) = s_aux*ss_aux/sss_aux;
599        rr = rr + 1;
600     end
601     t_ind = find(v_e1>=30);
602 end
603 a=1;
604 for i=1:35
605 AA=find(plot_ev_e1out(a:(i*10000))>=1);
606 BB=find(plot_ev_e2out(a:(i*10000))>=1);
607 [se1,ce1] = size(AA);
608 [se2,ce2] = size(BB);
609 sse1(i) = se1;
610 sse2(i) = se2;
611 a=a+10000;
612 prdc1(i)=(sse1(i)*4750)/670;
613 prdc2(i)=(sse2(i)*5030)/670;
614 maxprdc(i)= max(prdc1(i),prdc2(i));
615 if maxprdc(i)==prdc1(i)
616     inx(i) = 2;
617 elseif maxprdc(i)==prdc2(i)
618     inx(i) = 1;
619 end
620 end
621 tvect = 0:tr*T-1;
622 plot_e_out(1,:) = plot_ev_e1out;
623 plot_e_out(2,:) = plot_ev_e1_1out;
```

```
plot_e_out(3,:) = plot_ev_e2out;
plot_e_out(4,:) = plot_ev_e2_2out;
plot_e_out = logical(plot_e_out);
[pksM,locsM] = findpeaks(plot_prob_ex1); %localiza los índices de los
máximos locales
[pksM_2,locsM_2] = findpeaks(plot_prob_ex2); %localiza los índices de
los máximos locales
for a=1:tr
      n_a(a) =  plot_prob_inoutZ(a*10000);
end
n_a(1:9) = n_a(1:9)*60;
n_a(10:18) = n_a(10:18)*4;
for a=1:tr
    n_a(a) = Probs(a);
end
HHH=imagesc(c_e1);
axis equal off
colormap gray
colorbar
figure;
HHH=imagesc(c_e2);
axis equal off
colormap gray
colorbar
figure;

for j=1:tr
subplot(7,5,j)
vvv=imagesc(HV_PS{j});
axis equal off
colormap gray
end
figure;
plotRaster(plot_e_out, tvect/10000);
title('Membrane Voltages of All Neurons During Testing');
xlabel ( 'Time[s]');
ylabel ('Trials (#Neurons)');
figure;
subplot(4,1,1)
plot((0:tr*T-1)/10000, plot_v_e1,'r');
title('Membrane Voltage of Excitatory Neuron 1');
xlabel ( 'Time[s]');
ylabel ('Membrane Voltage[mV]');
subplot(4,1,2)
plot((0:tr*T-1)/10000, plot_v_in1,'g');
title('Membrane Voltage of Inhibitory Neuron 1');
xlabel ( 'Time[s]');
ylabel ('Membrane Voltage[mV]');
subplot(4,1,3)
plot((0:tr*T-1)/10000, plot_v_e2,'b');
title('Membrane Voltage of Inhibitory Neuron 2');
xlabel ( 'Time[s]');
ylabel ('Membrane Voltage[mV]');
subplot(4,1,4)
plot((0:tr*T-1)/10000, plot_v_in2,'m');
title('Membrane Voltage of Excitatory Neuron 2');
xlabel ( 'Time[s]');
ylabel ('Membrane Voltage[mV]');
figure;
plot((0:tr*T-1)/10000, plot_prob_ex1,'k');
title('Firing Probability of Excitatory Neuron 1');
xlabel ( 'Time[s]');
```

```
684 ylabel ('Firing Probability');
685 xlim([0 3])
686 ylim([-.1 1.05])
687 figure;
688 plot((0:tr*T-1)/10000, plot_prob_ex2,'c');
689 title('Firing Probability of Excitatory Neuron 2');
690 xlabel ( 'Time[s]');
691 ylabel ('Firing Probability');
692 ylim([-.1 1.05])
693 xlim([0 3])
694 figure;
695 bar(n_a/1000,0.5,'r');
696 title('Firing Probability of WTA Network');
697 xlim([0 36])
698 xlabel ( 'Pattern Index');
699 ylabel ('Firing Probability');
700 figure;
701 hold on
702 for i = 1:length(maxprdc)
703     h=bar(i,maxprdc(i)/100, 0.5);
704     if inx(i) == 1
705         set(h,'FaceColor','b');
706     else
707         set(h,'FaceColor','m');
708     end
709 end
710 hold off
711 title('Prediction of the Test Set');
712 xlim([0 36])
713 xlabel ( 'Pattern Index');
714 ylabel ('Prediction');
```

B.2 Código en Verilog A de los modelos estocásticos de memristores

```
1 `include "disciplines.vams"
2 `include "constants.vams"
3  module memristor(tp,bn);
4  inout tp,bn;
5
6  electrical tp,bn;
7 parameter integer model = 2;  // (1: Yakopcic model - 2: Pickett model - 3:
Biolek model)
8  // parameters for Yakopcic memristor model
9  parameter real Ap = 0.005;
10  parameter real An = 0.08;
11  parameter real xp = 0.2;
12  parameter real xn = 0.5;
13  parameter real alpha_p = 1.2;
14  parameter real alpha_n = 3;
15  parameter real a1 = 3.7e-7;
16  parameter real a2 = 4.35e-7;
17  parameter real b = 0.7;
18  parameter real x0 = 0.1;
19  parameter real time_step_y = 0.001;
20 //parameters for picket memristor model
21  parameter real f_off = 3.5e-6;
22  parameter real a_off = 1.2e-9;
23  parameter real b_p = 500e-6;
24  parameter real wc = 107e-12;
25  parameter real D = 3e-9;
26  parameter real w_init = 0.5;
27  parameter real Ron_p = 100;
28  parameter real Roff_p = 2e5;
29  parameter real f_on = 40e-6;
30  parameter real a_on = 2e-9;
31  parameter real time_step_p = 1e-8;
32  // General threshold based model (biolek)
33  parameter real beta = 10e13;
34  parameter real Roff = 10e3;
35  parameter real Ron = 1e3;
36  parameter real Rinit = 5e3;
37  parameter real time_step_b=10e-10;       // time step for the transient
analysis
38
39  // local variables to use within the model
40  real I_t;
41  real Vm;
42  real vini;
43  real taw;
44  real lambda;
45  real prob_t;
46  real randm_p;
47  real x;
48  real x_i;
49  integer seed;
50
51
52  /// Yakopcic parameters /////
53  real Vp;
54  real Vn;
55  real f_x;
56  real g_v;
57  real wp;
58  real wn;
59  real dxy_dt;
60  real taw_p;
61  real lambda_p;
```

```
62  real taw_n;
63  real V_x;
64
65  /// Pickett parameters /////
66  real Im;
67  real R;
68  real dw_dt;
69  real w;
70  real V_t;
71  real w_1;
72  real i_on;
73  real i_off;
74  real i_t;
75  real i_max;
76 //// parameters for bipolar threshold-based memristor model
77   real X;
78   real dx_dt;
79   real Vt;
80   real x_1;
81  analog
82  begin
83 // seed = V(in_seed); // to initialize the random number generator
84 ////////////////////////////
85 //// Yakopcic Model //////
86 ////////////////////////
87 if (model == 1) begin
88
89      $bound_step(time_step_y);
90      @(initial_step)begin
91         x= x0;
92         V_x = 1.5;
93         Vp = 1.5;
94         Vn = 0.5;
95         seed = 584;
96      end
97      Vm = V(tp,bn);
98
99      // stochastic setting of the threshold voltage
100      taw = pow(10,(-2.67*abs(Vm*3)+5.43));   // calculation of the tau
parameter for the poisson distribution
101
102      lambda = 1/taw;
103
104      prob_t = lambda*time_step_y;
105
106      randm_p = abs($rdist_uniform(seed,0,1));
107
108     if(Vm > 0) begin
109      if (prob_t >= randm_p) begin            // stochastic induction on the
switching operation
110           if  (Vm<1.5) begin
111             V_x = Vm - 0.1;
112             end
113           else if (Vm > 1.5) begin
114                 V_x =1.5;
115           end
116       end
117         else if (prob_t < randm_p) begin
118             V_x = 1.5;
119         end
120     end
121     Vp = V_x;
122     //// regular operation ////
123      if (Vm>0) begin
124         if (x>=xp) begin
125             wp = ((xp-x)/(1-xp))+1;
```

```
126             f_x = (exp(-alpha_p*(x-xp)))*wp;
127         end
128         else if (x< xp) begin
129             f_x = 1;
130         end
131         // setting of g(V(t))
132         if (Vm> Vp) begin
133             g_v = Ap * (exp(Vm)-exp(Vp));
134         end
135         else if (Vm <= Vp) begin
136             g_v = 0;
137         end
138      end
139      else if (Vm <=0) begin
140         if (x <= 1-xn) begin
141             wn = x/(1-xn);
142             f_x = (exp(alpha_n*(x+xn-1)))*wn;
143         end
144         else if (x> 1-xn) begin
145             f_x = 1;
146         end
147
148         if (Vm < - Vn) begin
149             g_v = -An*(exp(-Vm) - exp(Vn));
150         end
151         else if (Vm > -Vn) begin
152             g_v = 0;
153         end
154      end
155
156      dxy_dt = g_v*f_x;
157      x = dxy_dt*time_step_p + x_i;
158      if (Vm >= 0) begin
159         I_t = a1*x*sinh(b*Vm);
160      end
161      else if (Vm < 0) begin
162         I_t = a2*x*sinh(b*Vm);
163      end
164     I(bn,tp) <+ I_t;
165      x_i = x;
166      Vp = 1.5;
167 end
168 ////////////////////////
169 /// Picket model /////
170 ////////////////////////
171 else if (model == 2) begin
172      $bound_step(time_step_p);
173      @(initial_step) begin
174         w_1 = w_init*D;
175         vini = 0;
176         seed = 24883;
177          i_off = 115e-6;
178         i_on = 8.9e-6;
179         i_t = 115e-6;
180         i_max = -inf;
181     end
182     Im = I(tp,bn);
183         if (Im > i_max) begin
184             i_max = Im;
185         end
186     ////////////////////////
187     /// Stochastic behavior of the threshold currents
188      taw = pow(10,(-2.67*R*abs(Im)/5 +5.43));   // calculation of the tau
parameter for the poisson distribution
189      lambda = 1/taw;
190      prob_t = lambda*time_step_p*1e4;
```

```
191      randm_p = abs($rdist_uniform(seed,0,1));
192
193      if (prob_t >= randm_p) begin            // stochastic induction on the
switching operation
194         if(Im <0) begin
195             if ((abs(Im) >=4.5e-6) && (abs(Im) <= 8.9e-6)) begin
196                 i_on  = abs(Im) - 1e-7;
197             end
198             else  begin
199                 i_on = 8.9e-6;
200             end
201         end
202         else if (Im >= 0) begin
203             if ((Im >= 50e-6) && (Im <= 115e-6)) begin
204                 i_t = Im - 1e-7;
205             end
206             else begin
207                 i_t = 115e-6;
208             end
209         end
210      end
211      else begin
212         i_on = 8.9e-6;
213         i_t = 115e-6;
214      end
215     //// threshold period application    ///
216      if (i_t != 115e-6) begin    // if the threshold was changed
217         i_off = i_t;
218      end
219
220       @(cross(Im-i_max,+1))begin
221         i_off = 115e-6;
222       end
223     //// Operation of the memristor
224      if (Im >= 0) begin // Off switching case
225         dw_dt = f_off*sinh(Im/i_off)*exp(-exp((w_1-a_off)/wc -abs(Im/b_p))-
w_1/wc);
226      end
227      else if (Im <0) begin
228         dw_dt = f_on*sinh(Im/i_on)*exp(-exp((a_on - w_1)/wc -abs(Im/b_p))-
w_1/wc);
229      end
230      w = dw_dt*time_step_p + w_1;
231      if (w>=D) begin
232         w = D;
233         dw_dt = 0;
234      end
235      if ( w <=0) begin
236         w = 0;
237         dw_dt = 0;
238      end
239      R = Ron_p*(1-w/D)+ Roff_p*w/D;
240
241      V_t = R*Im;
242      V(bn,tp) <+ V_t;
243     w_1 = w;
244 end
245 ////////////////////////////////////////////////
246 ///// Bipolar threshold-based memristor model /////
247 ////////////////////////////////////////////////
248 else if(model == 3) begin
249      $bound_step(time_step_b);
250
251       @(initial step) begin
252         vini = 0;
253         Vt = 4.6;
```

```
254         seed = 584;
255         x_1 = Rinit;
256       end
257
258       Vm= V(tp,bn);
259       // stochastic setting of the threshold voltage
260      taw = pow(10,(-2.67*abs(Vm)+5.43));   // calculation of the tau waiting
parameter for the poisson distribution t
261      lambda = 1/taw;
262      prob_t = lambda*time_step_b;
263      randm_p = abs($rdist_uniform(seed,0,1));
264
265      if (prob_t >= randm_p) begin            // stochastic induction on the
spiking operation
266         if((Vm >0) && (Vm <= 4.6)) begin
267             Vt = Vm - 0.01;
268         end
269         else if((Vm<0) && (abs(Vm)<= 4.6)) begin
270             Vt = Vm + 0.01;
271         end
272      end else if (prob_t < randm_p) begin
273         Vt = 4.6;
274      end
275
276
277      dx_dt= beta*(Vm-0.5*(abs(Vm+Vt)-abs(Vm-Vt)));
278      if (dx_dt > 0) begin
279         if ((X>Roff))begin
280             dx_dt = 0;
281         end
282     end
283      if (dx_dt < 0) begin
284         if ((X <Ron))begin
285             dx_dt = 0;
286         end
287     end
288      X = dx_dt*time_step_b  + x_1;
289      if (X<Ron)begin
290         X = Ron;
291      end
292      if(X>Roff) begin
293         X = Roff;
294      end
295      I(bn,tp) <+ Vm/X;
296      x_1 = X;
297      Vt = 4.6;
298   end // end bipolar model
299  end        // end analog
300 end module   // end model
```

8 REFERENCIAS

[1] M., Ambroise, T., Levi, Y., Bornat and S., Saighi, Biorealistic spiking neural network on FPGA, 2013 47th Annual Conference on Information Sciences and Systems (CISS), Baltimore, MD, 2013, (pp. 1-6)

[2] Song, S., Miller, K. D., & Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synapticplasticity. Nature Neuroscience, 3(9), 919-926.

[3] Kraft, M., Kasiński, A., & Ponulak, F. (2006). Design of the spiking neuron having learning capabilities based on fpga circuits. IFAC Proceedings Volumes, 39(17), 301-306.

[4] D., Sterratt, B., Graham, A., Gillies and D., Willshaw (2011). Principles of computational modelling in neuroscience (pp. 13-46). Cambridge New York: Cambridge University Press.

[5] Izhikevich, E. (2007). Dynamical systems in neuroscience: the geometry of excitability and bursting. Cambridge, Mass: MIT Press.

[6] Cruz-Albrecht, J. M., Yung, M. W., & Srinivasa, N. (2012). Energy-Efficient Neuron, Synapse and STDP Integrated Circuits. IEEE Transactions on Biomedical Circuits and Systems, 6(3), 246-256.

[7] Schutter, E. (2010). Computational modeling methods for neuroscientists. Cambridge (pp. 139-159), Mass: MIT Press.

[8] Gerstner, W., Kistler, W. (2002). Spiking neuron models: single neurons, populations, plasticity (pp. 39-99). Cambridge, U.K. New York: Cambridge University Press.

[9] Izhikevich, E. M. (2010). Hybrid spiking models. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 368(1930), 5061-5070.

[10] Paz, I. T., Hernández Gress, N., & González Mendoza, M. (2013). Pattern Recognition with Spiking Neural Networks. In Lecture Notes in Computer Science (pp. 279-288). Springer Berlin Heidelberg.

[11] Izhikevich, E. M. (2003). Simple model of spiking neurons. IEEE Transactions on Neural Networks, 14(6), 1569-1572.

[12] Izhikevich, E. M. (2004). Spike-timing Dynamics of Neuronal Groups. Cerebral Cortex, 14(8), 933-944.

[13] Guerrero-Rivera, R., Morrison, A., Diesmann, M., & Pearce, T. C. (2006). Programmable Logic Construction Kits for Hyper-Real-Time Neuronal Modeling. Neural Computation, 18(11), 2651-2679.

[14] Handrich, S., Herzog, A., Wolf, A., & Herrmann, C. S. (2009). A Biologically Plausible Winner-Takes-All Architecture. In Emerging Intelligent Computing Technology and Applications. With Aspects of Artificial Intelligence (pp. 315-326). Springer Berlin Heidelberg.

[15] Morrison, A., Diesmann, M., & Gerstner, W. (2008). Phenomenological models of synaptic plasticity based on spike timing. Biological Cybernetics, 98(6), 459-478.

[16] Paugam-Moisy, H., & Bohte, S. (2012). Computing with Spiking Neuron Networks. In Handbook of Natural Computing (pp. 335-376). Springer Berlin Heidelberg.

[17] Al-Shedivat, M. (2015). Brain-inspired Stochastic Models and Implementations (Master Degree Thesis). King Abdullah University of Science and Technology, Thuwal, Kingdom of Saudi Arabia.

[18] Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. The Journal of Physiology, 117(4), 500-544.

[19] Izhikevich, E. M. (2000). Neural excitability, spiking and bursting. International Journal of Bifurcation and Chaos, 10(6), 1171-1266.

[20] Kistler, W. M., Gerstner, W., & Hemmen, J. L. van. (1997). Reduction of the Hodgkin-Huxley Equations to a Single-Variable Threshold Model. Neural Computation, 9(5), 1015-1045.

[21] Kuznet͡sov, Y., A. (1998). Elements of applied bifurcation theory (pp.1-37, 79-111). New York. Springer.

[22] Handrich, S., Herzog, A., Wolf, A., & Herrmann, C. S. (2009). Prerequisites for integrating unsupervised and reinforcement learning in a single network of spiking neurons. In 2009 International Joint Conference on Neural Networks. IEEE.

[23] Purves, D., Augustine, G., J. (2004). Neuroscience (pp 31-165). Sunderland, Mass: Sinauer Associates, Publishers.

[24] Kohonen, T. (2001). Self-organizing maps (pp. 71-105). Berlin New York: Springer.

[25] Corradi, F., Eliasmith, C., & Indiveri, G. (2014). Mapping arbitrary mathematical functions and dynamical systems to neuromorphic VLSI circuits for spike-based neural computation. In 2014 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE.

[26] Wu, G. K., Arbuckle, R., Liu, B., Tao, H. W., & Zhang, L. I. (2008). Lateral Sharpening of Cortical Frequency Tuning by Approximately Balanced Inhibition. Neuron, 58(1), 132-143.

[27] Oster, M., & Shih-Chii Liu. (n.d.). A winner-take-all spiking network with spiking inputs. In Proceedings of the 2004 11th IEEE International Conference on Electronics, Circuits and Systems, 2004. ICECS 2004. IEEE.

[28] Alabau, V., Andrés, J.M., Casacuberta, F., Garćıa-Hernández, J.C., Giménez, A., Juan, A., Sanchís, A., Vidal, E. (2006). The naive Bayes model, generalisations and applications.

[29] Bi, G., & Poo, M. (1998). Synaptic Modifications in Cultured Hippocampal Neurons: Dependence on Spike Timing, Synaptic Strength, and Postsynaptic Cell Type. The Journal of Neuroscience, 18(24), 10464–10472.

[30] Hebb, D. O. The organization of behavior: a neuropsychological theory. Mahwah, N.J: L. Erlbaum Associates, (2002). Psychology Pr.

[31] Strukov, D. B., Snider, G. S., Stewart, D. R., & Williams, R. S. (2008). The missing memristor found. Nature, 453(7191), 80–83.

[32] Chua, L. O., & Sung Mo Kang. (1976). Memristive devices and systems. Proceedings of the IEEE, 64(2), 209–223.

[33] Radwan, A. G., & Fouda, M. E. (2015). On the Mathematical Modeling of Memristor, Memcapacitor, and Meminductor. Studies in Systems, Decision and Control. Springer International Publishing.

[34] Al-Shedivat, M., Naous, R., Cauwenberghs, G., & Salama, K. N. (2015). Memristors Empower Spiking Neurons With Stochasticity. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 5(2), 242–253.

[35] Naous, R., Al-Shedivat, M., & Salama, K. N. (2016). Stochasticity Modeling in Memristors. IEEE Transactions on Nanotechnology, 15(1), 15–28.

[36] Biolek, D., Di Ventra, M., Pershin, Y., V. (2013) Reliable spice simulations of memristors, memcapacitors and meminductors. Radioengineering, vol. 22.

[37] Yakopcic, C., Taha, T. M., Subramanyam, G., Pino, R. E., & Rogers, S. (2011). A Memristor Device Model. IEEE Electron Device Letters, 32(10), 1436–1438.

[38] Bill, J., & Legenstein, R. (2014). A compound memristive synapse model for statistical learning through STDP in spiking neural networks. Frontiers in Neuroscience, 8.

[39] Chang, T., Jo, S.-H., Kim, K.-H., Sheridan, P., Gaba, S., & Lu, W. (2011). Synaptic behaviors and modeling of a metal oxide memristive device. Applied Physics A, 102(4), 857–863.

[40] Pickett, M. D., Strukov, D. B., Borghetti, J. L., Yang, J. J., Snider, G. S., Stewart, D. R., & Williams, R. S. (2009). Switching dynamics in titanium dioxide memristive devices. Journal of Applied Physics, 106(7), 74508.

[41] Simmons, J. G. (1963). Generalized Formula for the Electric Tunnel Effect between Similar Electrodes Separated by a Thin Insulating Film. Journal of Applied Physics, 34(6), 1793–1803.

[42] Abdalla, H., & Pickett, M. D. (2011). SPICE modeling of memristors. In 2011 IEEE International Symposium of Circuits and Systems (ISCAS). IEEE.