UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

MAPAS TEMPORALES MEDIANTE REDES NEURONALES AUTO-ORGANIZATIVAS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

RODRIGO ENRICO HERNÁNDEZ CÁRCAMO

PROFESOR GUÍA:
PABLO ESTÉVEZ VALENCIA

MIEMBROS DE LA COMISIÓN:
RICHARD WEBER HASS
JUAN VELÁSQUEZ SILVA

SANTIAGO DE CHILE
AGOSTO 2008

UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

MAPAS TEMPORALES MEDIANTE REDES NEURONALES AUTO-ORGANIZATIVAS

RODRIGO ENRICO HERNÁNDEZ CÁRCAMO

COMISIÓN EXAMINADORA		**CALIFICACIONES** NOTA (n°)	NOTA (Letras)	Firma
PROFESOR GUÍA DR. PABLO ESTÉVEZ V.	:	.....	.....	.....
PROFESOR INTEGRANTE DR. RICHARD WEBER H.	:	.....	.....	.....
PROFESOR INTEGRANTE JUAN VELÁSQUEZ S.	:	.....	.....	.....
NOTA FINAL EXAMEN DE GRADO	:	.....	.....	.....

SANTIAGO DE CHILE
2008

RESUMEN DEL INFORME FINAL PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELECTRICISTA
POR: RODRIGO ENRICO HERNÁNDEZ CÁRCAMO
FECHA: 20 de agosto de 2008
PROFESOR GUÍA: DR. PABLO ESTÉVEZ V.

MAPAS TEMPORALES MEDIANTE REDES NEURONALES AUTO-ORGANIZATIVAS

En problemas del mundo real la información posee frecuentemente fuertes dependencias temporales y una sola muestra no es suficientemente explicativa para captar la dinámica subyacente. Las redes neuronales han demostrado una alta efectividad tanto en problemas lineales como no-lineales. El modelo Merge Neural Gas (MNG) es un poderosos algoritmo no supervisado para el procesamiento de secuencias temporales, su estabilidad y convergencia lo hacen una herramienta atractiva y simple. Las memorias Gamma constituyen un poderoso filtro que posee la eficiencia de los filtros de Respuesta Infinita al Impulso (IIR), la estabilidad y fácil entrenamiento como los filtros de Respuesta Finita al Impulso (FIR). Su principal característica reside en poder controlar la profundidad de la memoria y la resolución del filtrado.

El presente trabajo propone un nuevo modelo de contextos que puede ser combinado con distintos esquemas de cuantización estáticos como mapas auto-organizativos o Gas Neuronal, cuyas reglas de entrenamiento se deriva de la minimización de un funcional de cuantización temporal, permitiendo así el procesamiento de secuencias temporales. El modelo de contextos se basa en memorias Gamma, las cuales además de capturar la dinámica de la serie entregan al algoritmo propiedades fundamentales de los filtros IIR y FIR. Puesto que las memorias Gamma se construyen recursivamente, el modelo propuesto puede controlar la memoria temporal ajustando la cantidad de contextos utilizados. Para cuantificar la calidad de la cuantización temporal se utilizó el Error de cuantización Temporal (TQE) y mediante el uso de planos de recurrencia se evaluó la capacidad del algoritmo para reconstruir una aproximación del espacio de estado.

El nuevo modelo generaliza a MNG, haciendo de éste un caso particular del modelo propuesto cuando sólo se utiliza un contexto. Esto no sólo permite reutilizar las propiedades ya estudiadas para MNG, sino que le entrega un nuevo marco teórico. Diversas bases de datos benchmark y de la vida real han sido utilizadas a fin de estudiar experimentalmente las propiedades de Gamma NG. Distintos atractores caóticos permiten demostrar cómo el aumento del número de contextos mejora la reconstrucción de espacio de estado realizado por el modelo de contextos, justificándose así la una de las superioridades de Gamma NG por sobre MNG. En el atractor de Lorenz la reconstrucción de espacio de estado arrojó un error E=0,0138 para Gamma NG y E=0.0199 para MNG. En tareas de clasificación el porcentaje de acierto fue de 92,45 % para Gamma NG y 89,81 % para MNG.

El modelo de contextos Gamma resulta ser una herramienta que puede ser combinada con distintos esquemas convencionales de cuantización el cual mediante una simple regla de recurrencia basada en memorias Gamma permite evitar el uso de ventanas de tiempo mejorando el procesamiento de secuencias gracias a una mejor reconstrucción de espacio de estado.

A mis padres

Agradecimientos

Quisiera partir agradeciendo a mis padres, Enrico y Yanet, por la formación que me han dado, desde mi infancia me enseñaron el compromiso, la responsabilidad y la humildad. La mayoría de mis logros académicos se los debo a ellos: la dedicación de mi madre y al susto que le tenía a mi padre :). Papá, muchas gracias por todo el cariño y apoyo que me has dado y aún me sigues dando. Mamá, muchas gracias por tu comprensión y cercanía. Los quiero mucho.

Tío Gino y Gricy, ustedes han sido como mi segunda familia durante todos mis años en la universidad, no sólo me han permitido vivir en su casa, sino que me hicieron parte de su hogar y muchas gracias a mis hermosas primitas por iluminar mis días.

Kim, tu fortaleza y superación siempre las he admirado y han sido una motivación para llegar más lejos, siempre estás cuidando de los tuyos y conmigo has hecho lo mismo, sin duda has sido parte importante de mi vida y espero que lo sigas siendo. Gracias tía Mirtha y tío Gerardo por quererme como a un hijo más.

Gracias al profesor Pablo Estévez por confiar en mi capacidad desde que era un esclavo, por sus acertados consejos y recomendaciones. Ha sido un guía en el que he creído ciegamente en mis últimos años y una persona en la que realmente puedo confiar. También agradezco a los profesores Richard Weber, Juan Velásquez y Pablo Zegers por aceptar formar parte de mi comisión examinadora.

Muchas gracias a mis compañeros de eléctrica los cuales me acompañaron y ayudaron a superar las dificultades de la carrera alegrando cada día la jornada universitaria. Por supuesto también a mis amigos del laboratorio: Vera, Carloncho, Neven, Meme, Vanel, Ñulz y Vitoco, ahh y Causa, que no lo veía tan seguido.

Quisiera agradecer a CONICYT en especial al PROGRAMA NACIONAL DE BECAS DE POSTGRADO por la beca entregada, sin la cual no habría podido realizar mis estudios de magíster. Agradezco también al FONDECYT 1080643 por el apoyo prestado durante mi investigación.

Índice general

Capítulo 1

Introducción

Existe una gran variedad de métodos orientados al análisis de datos, tales como técnicas estadísticas, de clustering, etc. Las Redes Neuronales Artificiales (RNA) aparecen como una alternativa atractiva para extraer información desde los datos. Las RNA están inspiradas en los sistemas biológicos y se caracterizan por su capacidad para aprender gradualmente en el tiempo a partir de ejemplos presentados a la red. Algunas de las características más atractivas son la robustez y tolerancia al ruido, las cuales les han permitido desempeñarse exitosamente en tareas tales como reconocimiento de patrones, análisis de series de tiempo, aproximación de funciones, optimización, control automático, etc.

Existen dos grandes familias de redes neuronales, las de entrenamiento supervisado y las no-supervisadas. El entrenamiento supervisado requiere que las muestras estén etiquetadas en clases, lo cual no siempre se dispone. El algoritmo de retro-propagación del error [1, 2], usa la diferencia entre la salida deseada (etiqueta) y la entregada por la red, para ajustar los pesos sinápticos de las neuronas. Cuando no se dispone de datos etiquetados, el entrenamiento no-supervisado permite que la red cree su propia representación de los datos, y que logre descubrir patrones típicos, clusters u otro tipo de información interesante sobre el conjunto de entrenamiento, sin necesitar de información externa alguna.

Uno de los paradigmas más utilizados dentro de las redes neuronales no-supervisadas es el Mapa Auto-Organizativo (SOM: *Self-Organizing Map*) introducido por Teuvo Kohonen [3]. Este tipo de mapas usualmente utiliza una grilla bidimensional la cual permite visualizar la estructura

del espacio de entrada. Las neuronas de la grilla de salida tienen asociados vectores en el espacio de entrada, los cuales mediante aprendizaje competitivo son capaces de preservar la topología del espacio de entrada. Durante el entrenamiento se activa la neurona ganadora (la más cercana al dato presentado) y sus vecinas en la grilla, en consecuencia las neuronas vecinas reconocen patrones similares.

En problemas del mundo real la información posee fuertes dependencias temporales, por lo que una sola muestra no es capaz de capturar la dinámica temporal del problema. Se genera así la necesidad de muestrear una secuencia de eventos para capturar adecuadamente la historia de los datos, y crear una mejor representación del fenómeno. Un ejemplo es un negocio donde las ventas varían dependiendo si corresponden a días hábiles o a fines de semana. Luego, conocer las ventas del fin de semana pasado pueden ser de utilidad para estimar las ventas del próximo fin de semana.

Una forma simple de abordar el problema temporal es utilizando de ventanas de tiempo. Este enfoque consiste en seleccionar un conjunto de muestras consecutivas en el tiempo a fin de capturar parte de la dinámica temporal del problema, pero una de las principales dificultades es elegir el tamaño de la ventana. Una ventana muy pequeña no logrará capturar correctamente la dinámica de la serie, mientras que una ventana muy grande tiende a promediar los estadísticos no estacionarios de la serie y aumenta la complejidad de la red. Principe [4–7] propone una técnica basada en redes neuronales que utiliza memorias de corto plazo para atacar problemas de series de temporales, evitando así el uso de ventanas de tiempo.

Diversas herramientas estadísticas han sido aplicadas con éxito en el análisis de series de tiempo, e.g. modelos de la familia ARMA (Auto Regressive Moving Average) se han desempeñado de forma exitosa en la identificación de sistemas [8, 9], predicción de series de tiempo [10–12], entre otras aplicaciones. Sin embargo si el sistema es no-lineal o no estacionario la solución mediante este tipo de técnicas estadísticas no garantiza efectividad.

La mayoría de los problemas del mundo real presentan no-linealidades de distintos tipos, pudiendo tener distinta naturaleza dependiendo del número de variables de estado. Las variables de estado en un sistema dinámico corresponden a cualquier conjunto de variables que describen el comportamiento del sistema, siempre y cuando ese conjunto sea del menor tamaño posible. Por ejemplo en un péndulo estas variables pueden ser la posición angular θ y la velocidad angular $\dot{\theta}$. Cuando el número de variables de estado es infinito, por ejemplo en movimiento de fluidos, donde es

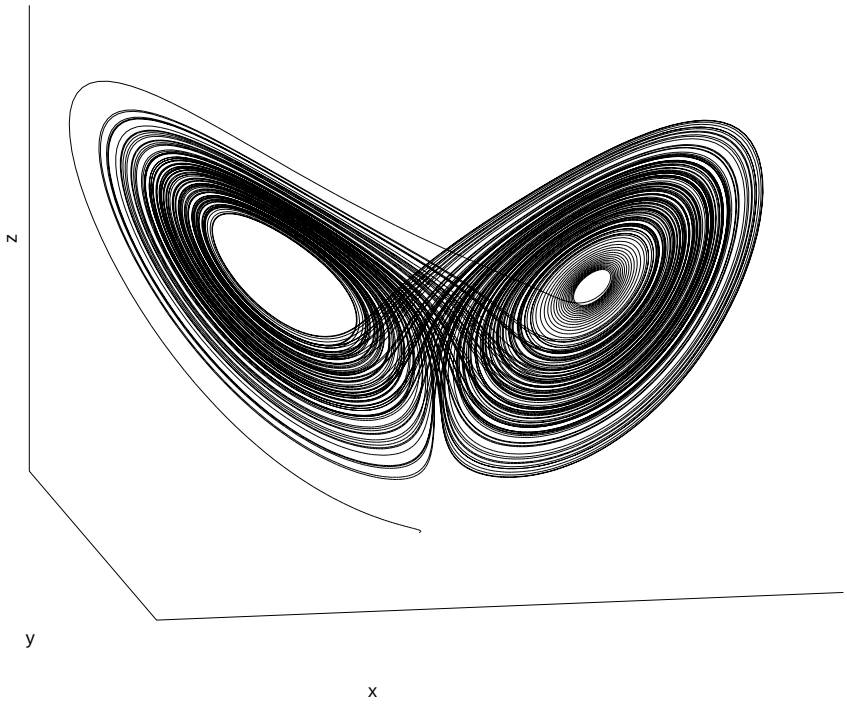


Figura 1.0.1: Espacio de estado 3D del atractor de Lorenz con σ=10, b=8/3 y R=28.

necesario conocer la temperatura, presión y velocidad en todos los puntos, el sistema se denomina espacio-temporal extendido. En este caso el sistema se modela mediante ecuaciones en derivadas parciales (EDP). Cuando el espacio de estado tiene dimensión finita el sistema dinámico admite un modelo en ecuaciones diferenciales ordinarias (EDO). El atractor de Lorenz (ver Figura 1.0.1) es un modelo físico de convección térmica, corresponde a un atractor caótico tri-dimensional, cuyas variables de estado están dadas por la solución del siguiente sistema

$$\begin{aligned} \dot{x} &= -\sigma x + \sigma y \\ \dot{y} &= Rx - y - xz \\ \dot{z} &= -Bz + xy. \end{aligned} \tag{1.0.1}$$

En general no se tiene acceso a todas las variables de estado del sistema, por lo que se utilizan técnicas de reconstrucción de espacio de estado para lograr estimar la solución de las EDOs. Takens [13] propuso un teorema de reconstrucción de espacio de estado en el cual define la estructura de

los vectores de espacio de estado a partir de una serie de tiempo.

1.1. Objetivo General

Desarrollar un nuevo método para cuantizar y visualizar secuencias de datos temporales basado en redes neuronales auto-organizativas.

1.2. Objetivos Específicos

1. Proponer una nueva red neuronal auto-organizativa que incluya memorias de corto plazo para el procesamiento de secuencias temporales. El esquema de cuantización utilizado como base del algoritmo será Gas Neuronal (NG: *Neural Gas*).

2. Desarrollar un modelo de contexto que permita capturar la dinámica de una serie basado en memorias Gamma. Estas son un tipo particular de filtros de tipo respuesta infinita al impulso (IIR: *Infinite Impulse Response*).

3. Verificar experimentalmente las capacidades del algoritmo para reconstruir una aproximación del espacio de estado de una secuencia.

4. Proponer un criterio de desempeño basado en planos de recurrencia.

5. Comparar la cuantización temporal del algoritmo propuesto con Merge NG [14,15] utilizando bases de datos *benchmark* y bases de datos reales. Una de las medidas de desempeño utilizada será el error de cuantización temporal (TQE).

El texto se organiza de la siguiente manera:

En el capítulo 2 se presenta una descripción de los algoritmos de cuantización no supervisados ya existentes, como el mapa auto-organizativo de Kohonen, Gas Neuronal y Gas Neuronal Constructivo. Se señalan los fundamentos teóricos y las características principales de cada algoritmo. Enseguida, se introducen las principales características y la evolución de los distintos algoritmos orientados al procesamiento de secuencias como: el mapa temporal de Kohonen, Mapas Recurrentes, Mapas Recursivos, SOMTAD y GASTAD y finalmente Merge Neural Gas. Se presenta el error de

cuantización temporal, que ha sido utilizado para medir el desempeño de los algoritmos de cuantización temporal propuestos en la literatura. A modo de introducción al procesamiento temporal se presentan dos propuestas de memorias temporales: las ventanas de tiempo y las memorias Gamma. Finalmente se presenta una descripción de los sistemas dinámicos no lineales abordando también las principales técnicas y dificultades en la resolución de estos problemas.

En el capítulo 3, se presenta un nuevo modelo de contexto. Mediante un esquema simple basado en ventanas de tiempo se deriva un funcional que puede ser fácilmente generalizado al uso de filtros Gamma. Utilizando memorias Gamma se deriva la regla de entrenamiento utilizada por Merge NG, caracterizándose la profundidad de la memoria y resolución de Merge NG. Posteriormente se propone un nuevo modelo de contextos, se deriva la regla de entrenamiento y se estudian propiedades fundamentales del modelo propuesto. Se realiza una interpretación basada en espacio de estado que permite visualizar como el modelo propuesto aborda el problema temporal desde esta perspectiva. Dada la capacidad del algoritmo para ajustarse a distintos esquemas de cuantización se proponen tres nuevos algoritmos Gamma SOM, Gamma NG y Gamma GNG. Finalmente se hace una descripción de las bases de datos a utilizar y se explican las principales medidas de desempeño a utilizar.

En el capítulo 4 se estudia experimentalmente el desempeño de los algoritmos propuestos y se presentan los resultados obtenidos con bases de datos benchmark y con las bases de datos del mundo real. Se mide el error de cuantización temporal TQE, se construyen planos de recurrencia y se aprovecha la reconstrucción de espacio de estado para resolver un problema de predicción.

En el capítulo 5 se discuten las medidas de desempeño utilizadas, se estudian características relevantes del algoritmo, como el costo computacional, y se da una orientación de cómo seleccionar los parámetros del algoritmo. Finalmente se comparan las ventajas de Gamma NG sobre MNG en las bases de datos utilizadas en función de los resultados obtenidos y las medidas de desempeño utilizadas.

En el capítulo 6 se presentan las conclusiones, exponiendo los principales aportes de este trabajo y las líneas de exploración para futuras investigaciones.

Capítulo 2

Marco Teórico

2.1. Redes Auto-Organizativas

2.1.1. Mapa de Kohonen

El mapa auto-organizativo de Kohonen (SOM: Self-Organizing Map) [3] es uno de los esquemas más utilizados dentro de las redes neuronales auto-organizativas. El SOM está basado en un modelo biológico de la actividad neuronal en el cerebro. A diferencia de los algoritmos supervisados, SOM no requiere los valores deseados de cada vector de entrada. Uno de los objetivos del SOM es poder generar un mapeo topológicamente ordenado desde un espacio de entrada d-dimensional a un espacio de salida a-dimensional, donde $a \leq d$. Las neuronas de la red generan cierta actividad ante el estímulo de los datos de entrada, esta actividad permite determinar que zonas del mapa, o más específicamente que neuronas han aprendido a representar ciertos patrones de la entrada. Se supone que neuronas de mayor actividad, son capaces de ajustarse o especializarse más fácilmente a los ejemplos que intentan representar. Por el contrario, si las neuronas no se han especializado en el presente vector de entrada, su representación estará ubicada en zona de baja actividad y las neuronas no necesitan ajustarse, pues perderían la capacidad de representación de los patrones aprendidos.

De este modo se logra generar un mapa cuyas zonas de actividad van cambiando a medida que se presentan distintos patrones. Los patrones que generan actividad en la misma zona poseen características similares y pueden ser agrupados dentro de una misma categoría o *cluster*.

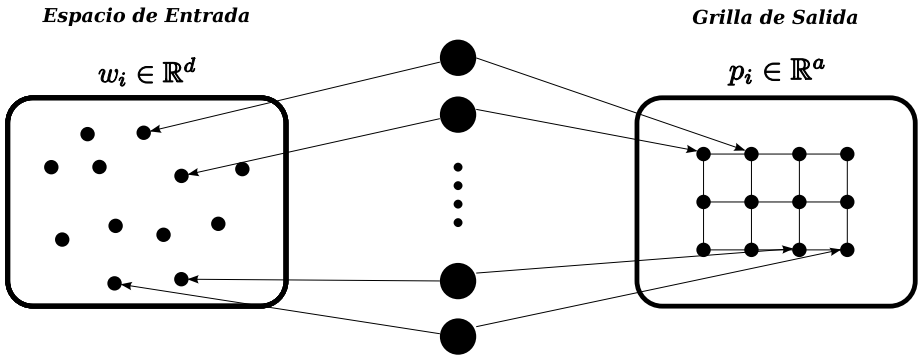


Figura 2.1.1: Arquitectura de la red SOM

El mapa auto-organizativo ordena las neuronas $\mathcal{N} = \{1, \ldots, M\}$ en una grilla fija a-dimensional sobre la cual se definen relaciones de vecindad. Las neuronas vecinas son aquellas conectadas directamente en la grilla. Si bien la dimensión a de la grilla no posee restricción alguna, por lo general ésta es escogida menor o igual a 3, permitiendo una visualización directa. Cada neurona posee vecinos de distintos niveles dependiendo de la distancia a la que se encuentran a ésta. Los vecinos de primer nivel son los más cercanos y están conectados directamente a la neurona. El número de vecinos en cada nivel depende del tipo de grilla. Los tipos de grilla más usados son la rectangular y la hexagonal, sin embargo éstas podrían tener cualquier topología.

Cada neurona tiene asociado un vector prototipo $w_i \in \mathbb{R}^d$ en el espacio de entrada y un vector posición $p_i \in \mathbb{R}^a$ en la grilla de salida. Los vectores prototipos en el espacio de entrada son ajustados a los datos durante el entrenamiento, mientras que la grilla se mantiene fija en el espacio de salida.

La Figura 2.1.1 muestra el arreglo de neuronas en la grilla de salida y el correspondiente vector prototipo en el espacio de entrada. El entrenamiento de los vectores es realizado en espacio de entrada sujeto a la topología dada por la grilla de salida.

2.1.1.1. Inicialización

- **Inicialización aleatoria**. El algoritmo SOM es lo suficientemente robusto como para aceptar una inicialización aleatoria de los vectores prototipo. Los vectores inicialmente desordenados en el espacio de entrada son ordenados a los largo del entrenamiento. Aunque La inicialización aleatoria no es la más rápida, de todas formas se obtienen buenos resultados.

- **Inicialización a partir de las muestras**. Consiste en seleccionar de forma aleatoria distintos puntos del conjunto de entrada para inicializar los vectores prototipos. Con esto se obtiene una aproximación de la distribución de los datos.
- **Inicialización lineal**. Los vectores prototipos son inicializados de un modo ordenado a partir del subespacio lineal generado por los dos vectores propios principales de los datos de entrada..

Ambas inicializaciones, aleatoria y a partir de las muestras, no son restrictivas del algoritmo SOM únicamente.

2.1.1.2. Aprendizaje

El aprendizaje esta basado en la idea de *soft competition*, Figura 2.1.2, i.e. la adaptación de los pesos se realiza no sólo a la unidad ganadora, sino que también a sus vecinos. La adaptación es más fuerte en las neuronas que presentan mayor actividad, es decir, las neuronas más cercanas al patrón de entrada la cual va luego decreciendo, mediante una distribución gausiana, para las neuronas que se encuentran más alejadas en la grilla con respecto a la neurona ganadora.

Dado un patrón de entrada $x(t) \in \mathbb{R}^d$, se calcula la distancia de todos los elementos de la red al ejemplo presentado. La unidad ganadora o *best matching unit* (BMU) es la que presenta mayor similitud al patrón de entrada $x(t)$, ésta se denota como i^*. La unidad ganadora se calcula determina según la ecuación siguiente

$$i^* = \underset{i=1\ldots M}{\operatorname{argmin}} \{\|x(t) - w_i\|\} \tag{2.1.1}$$

donde $\|\cdot\|$ es una medida de distancia (en la mayoría de los casos se utiliza la norma euclidiana), M es el número de neuronas en la red. Todas las neuronas son adaptadas al patrón de entrada a fin de lograr aprender el ejemplo presentado. Esta adaptación consiste en un acercamiento espacial hacia el vector presentado. Los pesos de toda la red son adaptados mediante la ecuación siguiente

$$w_i(t+1) = w_i(t) + \epsilon(t) h_{i,i^*}(t) \left(x(t) - w_i(t)\right) \tag{2.1.2}$$

donde $\epsilon(t)$ es la tasa de aprendizaje, h_{i,i^*} es una función de vecindad gausiana centrada en la unidad ganadora, que permite controlar la adaptación de toda la red según la distancia de cada neurona a

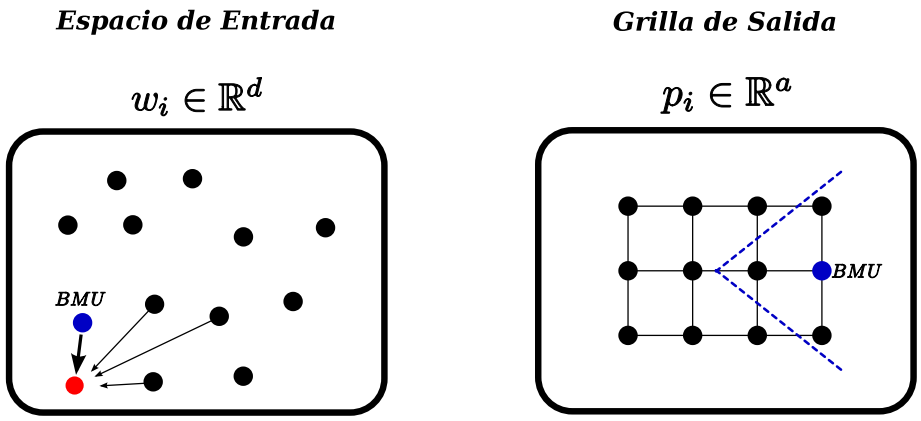


Figura 2.1.2: Entrenamiento de la red SOM. En el espacio de entrada la unidad ganadora BMU y sus vecinos topológicos son adaptados moviéndose hacia el patrón de entrada. El vector de mayor adaptación es la unidad ganadora BMU.

la BMU. Por lo general la tasa de aprendizaje y la vecindad decrecen en el tiempo. Típicamente la función de vecindad se define como:

$$h_{ii^*}(d_{\mathcal{N}}(i, i^*)) = \exp\left(\frac{-d_G(i, i^*)}{\sigma(t)}\right) \tag{2.1.3}$$

donde $d_G(i, i^*)$ es la distancia en la grilla de salida entre la neurona i y la neurona i^* ej. en una grilla bidimensional $d_G(i, i^*) = abs\,(x_i - x_{i^*}) + abs\,(y_i - y_{i^*})$.

El parámetro $\sigma(t)$ es el ancho de la vecindad el cual decae en el tiempo según

$$\sigma(t) = \sigma_i \left(\frac{\sigma_f}{\sigma_i}\right)^{\frac{t}{T}} \tag{2.1.4}$$

donde los valores σ_i y σ_f son escogidos al inicio del entrenamiento y corresponden al tamaño inicial y final de la vecindad, T es el número máximo de épocas de entrenamiento. Dado que $\sigma_i > \sigma_f$ el tamaño de la vecindad decae exponencialmente durante el entrenamiento.

La tasa de aprendizaje $\epsilon(t)$ corresponde a un función que decae en el tiempo , la cual está encargada de asegurar la convergencia del algoritmo. Esta es de la forma

$$\epsilon(t) = \epsilon_i \left(\frac{\epsilon_f}{\epsilon_i}\right)^{\frac{t}{T}} \tag{2.1.5}$$

donde los valores ϵ_i y ϵ_f son escogidos al inicio del entrenamiento, T es el número máximo de épocas de entrenamiento.

La configuración anterior permite que al inicio del entrenamiento el ajuste de los pesos sea mayor y que la red se auto-organice. Esta etapa es conocida como ajuste grueso y se caracteriza porque tanto los valores de la tasa de aprendizaje $\epsilon(t)$ como el ancho de la vecindad $\sigma(t)$ son altos. Hacia finales del entrenamiento, los valores de la tasa de aprendizaje y de la vecindad son pequeños, por lo que las variaciones Δw son pequeñas, permitiendo un ajuste fino.

El número de neuronas escogidas afecta la precisión del mapa, mientras más grande es el número de neuronas mejor es la resolución del mapa, sin embargo un número excesivo de neuronas puede hacer que el clustering sea inexistente, por ejemplo si cada patrón de entrada está representado por una neurona diferente.

El algoritmo SOM se resume a continuación.

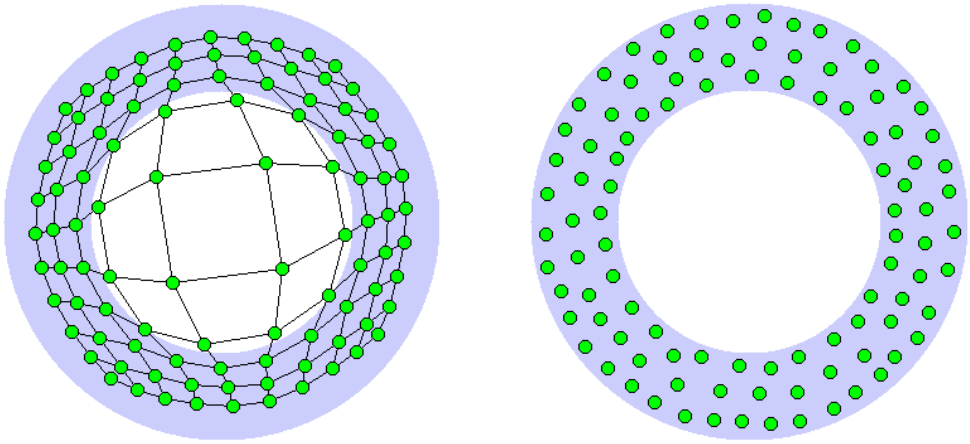
Algoritmo 1 Mapa de Kohonen SOM

1. Inicializar aleatoriamente los vectores prototipo, $w_i, i = 1 \dots M$.
2. Presentar un vector de entrada, $x(t)$, a la red.
3. Encontrar el BMU, j^*, usando (2.1.1)
4. Actualizar los vectores prototipo, w_i , mediante la regla (2.1.2).
5. Incrementar el número de la iteración ($t \rightarrow t + 1$).
6. Si $t < T$ volver al punto 2.

2.1.2. Gas Neuronal

El Gas Neuronal (NG: *Neural Gas*) [16] es un algoritmo de red neuronal auto-organizativos orientado a la cuantización vectorial de estructuras arbitrarias. A diferencia de SOM, no define una grilla que impone relaciones topológicas entre las unidades de la red. Los pesos sinápticos w_i son adaptados independientemente de cualquier estructura topológica, y la adaptación Δw_i se realiza en función de las distancias relativas (distorsiones) de la red al patrón, en el mismo espacio de entrada.

El algoritmo NG debe su nombre a la similitud entre la libertad de movimiento de las moléculas de los elementos en estado gaseoso y las unidades de la red. Esta misma libertad permite al algoritmo una mejor capacidad para aproximar la distribución de los datos en el espacio de entrada, ya que



(a) (b)

Figura 2.1.3: Cuantización de una distribución tipo anillo. a) muestra una cuantización mediante SOM, b) muestra una cuantización mediante NG. Se puede observar que debido a las restricciones topológicas, existen neuronas en el SOM que no están correctamente ubicadas (fuera de la distribución de los datos).

las neuronas no están obligadas a tener que mantener ciertas relaciones de vecindad, esto se puede observar en la Figura 2.1.3[1].

Dado un patrón de entrada $x(t)$, la información se ve reflejada en la red mediante distorsiones en cada neurona $D_{x(t)} = \{\|x(t) - w_i\| \, | i = 1 \dots M\}$, donde M es el número total de neuronas en la red. La regla de adaptación puede definirse como *winner-take-most*, es decir, la adaptación se realiza sobre toda la red, siendo la unidad ganadora (BMU) la de mayor adaptación. El resto de las neuronas va disminuyendo su adaptación de acuerdo a un ranking que ordena las distorsiones de menor a mayor, asignando el valor 0 a la menor distorsión. La unidad ganadora (BMU) se define como la de menor distorsión (2.1.1).

Cada patrón de entrada $x(t)$ genera una "excitación" $f_i(D_{x(t)})$ sobre cada unidad i de la red. Esta excitación depende de la distorsión en la neurona como $f_i(D_{x(t)}) = h_\lambda(k_i) = e^{-\frac{k_i}{\lambda}}$, donde λ es un parámetro que determina la vecindad o número de neuronas que se adaptarán de forma significativa durante la adaptación y $k_i \in \{0 \dots M-1\}$ corresponde al ordenamiento de menor a mayor de las distorsiones $D_{x(t)}$. La unidad escogida como ganadora se le asigna el valor $k = 0$. La adaptación de las neuronas finalmente depende de la tasa de aprendizaje $\epsilon(t)$, del ranking k y de

[1] Figura extraída de http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/DemoGNG/NG_2.html

la función de vecindad $h_\lambda(t)$.

El algoritmo NG se resume a continuación:

Algoritmo 2 Neural Gas

1. Inicializar aleatoriamente los vectores prototipo, $w_i, i = 1 \dots M$.
2. Presentar un vector de entrada, $x(t)$, a la red.
3. Encontrar el BMU, j^*, usando (2.1.1).
4. Actualizar los vectores prototipo, w_i , mediante la regla:

$$\Delta w_i = \epsilon(t) h_\lambda(t) \left(x(t) - w_i\right) \tag{2.1.6}$$

   donde $\epsilon(t)$ es la tasa de aprendizaje (2.1.5),T es el número de épocas de entrenamiento y

$$\lambda(t) = \lambda_i \left(\frac{\lambda_f}{\lambda_i}\right)^{\frac{t}{T}} . \tag{2.1.7}$$

5. Incrementar el número de la iteración $(t \to t + 1)$.
6. Si $t < T$ volver al punto 2.

2.1.3. Gas Neuronal Constructivo

En el algoritmo Gas Neuronal Constructivo (GNG: Growing Neural Gas) [17], a diferencia de los previamente descritos, el número de unidades crece durante el proceso auto-organizativo. Se comienza con dos unidades, para luego ir agregando neuronas a la red, sucesivamente. Para determinar dónde insertar nuevas unidades, se calculan algunas medidas de error locales durante el proceso de adaptación. Una nueva unidad se inserta cerca de aquella neurona que posee el mayor error acumulado. La topología inicial de la red es un conjunto de 2 neuronas. La neurona i tiene asociada un vector prototipo d-dimensional, w_i . El algoritmo GNG es el siguiente:

Algoritmo 3 Gas Neuronal Constructivo

1. Inicializar aleatoriamente los vectores prototipos, w_1 y w_2 . Inicializar el conjunto de conexiones C, $C = A \times A$, como el conjunto vacío e inicializar el número de iteraciones en $t = 0$
2. Presentar un vector de entrada, $x(t)$, a la red.
3. Encontrar el BMU, j^*, usando (2.1.1) y la segunda unidad más cercana k^* mediante

$$k^* = \underset{i \in A \setminus i^*}{\operatorname{argmin}} \|x(t) - w_i\| . \tag{2.1.8}$$

4. Adaptar los vectores prototipo de acuerdo a la regla del Gas Neuronal.
5. Si no existe una conexión entre j^* y k^* crearla $C = C \cup (j^*, k^*)$. Configurar la edad del borde entre j^* y k^* a cero (refrescar el borde), $age_{(j^*,k^*)} = 0$.
6. Incrementar la edad de todos los bordes que emanan de j^* :

$$age_{(j^*,i)} = age_{(j^*,i)} + 1, \quad \forall i \in N_{j^*} \tag{2.1.9}$$

   donde N_{j^*} es el conjunto de vecinos topológicos directos (existe un borde entre i y j^*) de j^*.
7. Remover aquellos bordes con una edad mayor que la edad máxima, $T(t)$ en la iteración t definida como:

$$T(t) = T_i \left(\frac{T_f}{T_i} \right)^{\frac{t}{t_{max}}} . \tag{2.1.10}$$

8. Incrementar el número de la iteración $(t \to t + 1)$.
9. Si $t < T$ volver al punto 2.

2.1.4. Celdas de Voronoi

Dado un conjunto de vectores prototipos, $\{w_1, w_2, \ldots, w_N\} \in R^d$, el conjunto o celdas de Voronoi, V_i de un vector prototipo particular, w_i , se define como el subconjunto de los vectores de entrada para los cuales, el vector prototipo w_i es el más cercano:

$$V_i = \{x | \, \|x - w_i\| < \|x - w_j\| \, , i \neq j\} \tag{2.1.11}$$

donde $i, j = 1, \ldots, N$ y $\|\cdot\|$ es la medida de distancia.

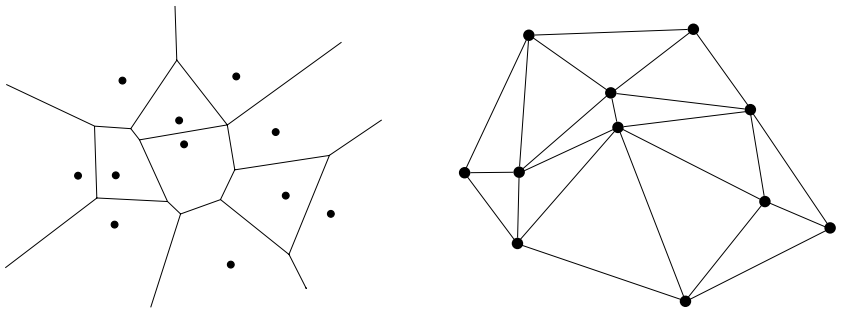


Figura 2.1.4: Dado un cierto conjunto de vectores prototipos. a) Celdas de Voronoi. b) triangulación de Delaunay

El conjunto de todas las celdas de Voronoi se conoce como Teselación de Voronoi. Al conectar todos los pares de vectores prototipo para los cuales las celdas de Voronoi respectivas comparten un borde, se obtiene la Triangulación de Delaunay [18]. La Figura 2.1.4 muestra las celdas de Voronoi y la triangulación de Delaunay para un mismo conjunto.

Normalización El efecto de la normalización permite que los valores de la tasa de aprendizaje $\epsilon(t)$ y el tamaño de la vecindad $\sigma(t)$ necesiten ajustes menores entre distintas bases de datos. La normalización de los datos ayuda de gran manera a que la red tenga un buen desempeño. Es común normalizar las componentes de los vectores de entrada de tal forma que tengan media cero y varianza unitaria. Esto puede lograrse sustrayendo la media μ y dividiendo la magnitud de las componentes por su desviación estándar σ, como se muestra en la siguiente ecuación

$$\hat{x} = \frac{x - \mu}{\sigma} \tag{2.1.12}$$

una alternativa es la normalización lineal, la que consiste en restarle a cada componente el valor mínimo x_{min} de ella dividido por la diferencia entre el valor máximo x_{max} y el valor mínimo de esa componente:

La normalización es realizada por columnas, ya que cada característica puede poseer distinta dimensión física, ya sea metros, centímetros, etc. De esta forma se busca lograr una equidad en el aporte de las características a la red. En la mayoría de los casos es incorrecto normalizar por filas, pues esto re-escala los vectores de entrada a la esfera unitaria perdiendo así la distribución original

de los datos. Sin embargo, este efecto puede ser deseable en aplicaciones de *text-mining*, donde lo que importa principalmente es el ángulo entre los vectores y no la distancia euclidiana entre ellos.

2.2. Redes Auto-Organizativas para el Procesamiento de Secuencias Temporales

2.2.1. Mapas Temporales de Kohonen

El Mapa Temporal de Kohonen (TKM: *Temporal Kohonen Maps*) [19] es uno de los primeros acercamientos basados en redes neuronales auto-organizativas para el procesamiento de secuencias temporales. Previo a TKM ya existían ciertos intentos por incluir el contexto en los Mapas de Kohonen [20], TKM propone una técnica alternativa con un claro realismo biológico, que permite evitar el uso de ventanas de tiempo.

El modelo se basa en que las neuronas son capaces de mantener su actividad durante un período de tiempo. Esto es modelado mediante un potencial de activación que se fundamenta en la constante de tiempo asociada al decaimiento de la actividad eléctrica en una neurona de la corteza cerebral.

La red Neuronal TKM está dada por un conjunto de neuronas $\mathcal{N} = \{1, \ldots, M\}$ las cuales poseen pesos $w_i \in \mathbb{R}^d$. Los datos $x^t \in \mathbb{R}^d, \forall t = 1 \ldots T$. El planteamiento original de TKM [19] consiste en maximizar la actividad $U_i(t)$ de las neuronas de la red

$$U_i(t) = \alpha U_i(t-1) - \frac{1}{2} \|x(t) - w_i\| \tag{2.2.1}$$

donde El término αU_i hace de umbral o barrera de activación.

El criterio de distancia (2.2.2), es un planteamiento alternativo[2] que básicamente consiste en mantener alejadas las neuronas que en instantes anteriores no han logrado calzar el patrón de entrada. La constante de tiempo α controla la capacidad de la red para recordar la actividad de las neuronas.

$$d_i(t) = \alpha d_i(t-1) + (1-\alpha) \left\| x^t - w^i \right\|^2 \tag{2.2.2}$$

[2]El motivo de esta representación, es mantener el criterio de minimización de la distancia para encontrar la unidad ganadora (BMU), el cual es usado a lo largo de la tesis.

Desenrollando el criterio de distancia (2.2.2) se obtiene

$$d_i(t) = \sum_{j=1}^{t} (1-\alpha)\,\alpha^{t-j} \left\|x^j - w^i\right\|^2 \tag{2.2.3}$$

y derivando el funcional con respecto a w se puede encontrar el vector de pesos que minimiza la distancia de la neurona al patrón de entrada,

$$\frac{\partial d_i(t)}{\partial w} = -\sum_{j=1}^{t} (1-\alpha)\,\alpha^{t-j} 2(x^j - w) = 0 \tag{2.2.4}$$

de donde se obtiene

$$w = \frac{\sum_{j=1}^{t} \alpha^{t-j} x^j}{\sum_{j=1}^{t} \alpha^{t-j}}. \tag{2.2.5}$$

El vector de pesos corresponde a la media ponderada de los patrones previamente presentados a la neurona.

Originalmente TKM fue formulado para los mapas de Kohonen, sin embargo como la elección del contexto no depende de arquitectura de la grilla, este puede ser utilizado con otros cuantizadores como Neural Gas.

El algoritmo TKM se resume a continuación:

Algoritmo 4 Temporal Kohonen Maps

---

1. Inicializar aleatoriamente los vectores prototipo en cada neurona, $w_i, d_i(0) = 0, \forall i = 1 \ldots m$
2. Presentar un vector de entrada, x^t, a la red.
3. Encontrar el BMU, I_t, usando (2.1.1), donde $d_i(t)$ se calcula como (2.2.2).
4. Actualizar los vectores prototipo, w_i , mediante la regla:

$$\Delta w_i = \epsilon(t) h_\sigma(d_{\mathcal{N}}(i, I)) \left(x^t - w_i\right) \tag{2.2.6}$$

   donde $\epsilon(t)$ es la tasa de aprendizaje definida en (2.1.5), T es el número de épocas de entrenamiento y $h_\sigma(d_{\mathcal{N}}(i, I))$ es la función de vecindad (2.1.3).
5. Incrementar el número de la iteración $(t \to t+1)$.
6. Si $t < T_s$ volver al punto 2.

2.2.2. Mapa Auto-organizativo Recurrente

El Mapa Auto-Organizativo Recurrente (RSOM:*Recurrent Self-Organizing Maps*) [21, 22], propone una mejora para el algoritmo TKM. RSOM define un vector para cada unidad en el mapa. Los pesos de la unidad ganadora son adaptados mediante ecuaciones de diferencias recursivas entre la nueva entrada presentada al mapa, el vector previo de diferencias y el peso de la unidad. El Mapa Temporal de Kohonen no usa directamente la información temporal del contexto en la actualización instantánea de los pesos y las entradas anteriores sólo son tomadas en cuenta en la selección de la unidad ganadora (BMU). RSOM logra incluir la naturaleza temporal de la serie tanto en la selección de los pesos como en la adaptación de ellos. Para ello se propone el vector de diferencias y_i^t para una entrada x^t, el cual se calcula según la siguiente expresión:

$$y_i^t = \alpha y_i^{t-1} + (1-\alpha)\left(x^t - w_i\right) \tag{2.2.7}$$

donde α controla la inclusión del vector de diferencias en el instante anterior, y el error entre el patrón de entrenamiento y el vector de cuantización w_i.

En resumen, RSOM no sólo considera la magnitud del error, sino que también se considera la dirección de este, de esta forma la adaptación de los pesos se realiza en mejores direcciones de descenso. El valor de $0 < \alpha < 1$ determina el efecto del vector de diferencias, un valor pequeño equivale a una corta memoria. El criterio de distancia es

$$d_i(t) = \left\| y_i^t \right\|. \tag{2.2.8}$$

Un análisis similar al realizado en TKM para determinar los pesos óptimos, se desarrolla en [23] para la distancia (2.2.8), siendo el resultado idéntico a (2.2.5). Una comparación detallada entre TKM y RSOM puede ser encontrada en [23].

RSOM se ha utilizado en la predicción de series de tiempo mediante modelos lineales locales [24] con resultados inferiores a redes de perceptrón multicapa (MLP:*Multilayer Perceptrón*).

El algoritmo RSOM se resume a continuación:

Algoritmo 5 Recurrent Self-Organizing Maps

1. Inicializar aleatoriamente los vectores prototipo y vectores de recurrencia en cada neurona, $w_i, y_i, \forall i = 1 \dots M$
2. Presentar una secuencia de entrada $\mathcal{S}_s$
   a) Presentar un vector de entrada, x^t, a la red.
   b) Encontrar el BMU, I_t, usando (2.1.1), donde $d_i(t)$se calcula como (2.2.8).
   c) Actualizar los vectores prototipo, w_i según (2.2.6) y vectores de diferencias y_i^tsegún (2.2.7) .
   d) Incrementar el número de la iteración $(t \to t+1)$.
   e) Si $t < T_s$ volver al punto $2a$.
3. Procesar una nueva secuencia $(s \to s+1)$
4. Si $s < S$ volver al punto 2.

2.2.3. SOMTAD y GASTAD

Principe [25] propone una arquitectura basada en dos elementos principales: auto-organización y difusión de la información. La difusión es utilizada para captar la dinámica de la secuencia. La arquitectura de la red establece conexiones entre las neuronas, por medio de las cuales se difunde la actividad, la cual decae en el tiempo. El ganador se elige mediante una combinación del vector de pesos y la actividad de la neurona. Este tipo de actividad implementa memorias de corto plazo.

SOMTAD (*Self-Organizing Maps with Temporal Activity Diffusion*) está definido en una red neuronal tipo SOM. En este tipo de redes las conexiones entre las neuronas están establecidas según la topología de la grilla. La actividad se difunde a través de las conexiones mediante un modelo de actividad (2.2.9) basado en Memorias Gamma [5], cuya ecuación

$$a(x,t) = (1-\mu)\, a(x,t-1) + \mu \, \|x - w_x\| \tag{2.2.9}$$

donde μ es un parámetro de retroalimentación de la actividad, $a(x,t)$ corresponde a la actividad de la neurona x con peso w_x en el instante t. La unidad ganadora BMU es la que minimiza (2.2.10).

$$d_i(t) = \|x - w_i\| - \beta a(x,t) \tag{2.2.10}$$

donde β es un parámetro de acoplamiento espacio-temporal, el cual representa la disminución del

umbral introducido por la difusión de la actividad. Un valor de β alto logra disminuir lo suficiente el umbral de activación, como para que el ganador en la siguiente iteración sea un vecino de la unidad ganadora en la iteración actual. Al hacer que $\beta \to 0$, se recupera el SOM original. El autor menciona que la actividad viaja a través de la red en forma similar al modo en que viajan las ondas al lanzar un piedra en una fuente.

El parámetro $0 < \mu < 1$, como se estudia con más detalle en la sección 2.3.2 de memorias Gamma, controla la profundidad de la memoria, en este caso la velocidad con que desaparecen las ondas o la capacidad de la red para recordar la actividad.

El modelo GASTAD (*Neural Gas with Temporal Activity Diffusion*), está montado sobre el cuantizador NG, el que a diferencia de SOM no posee una topología que conecte las neuronas de la red. Por lo tanto es necesario definir una estructura que permita determinar hacia que neuronas se difundirá la actividad de la neurona ganadora. Una conexión $p_{i,j}$ se refuerza o debilita de manera temporal según la regla (2.2.11). La idea principal es que la conexión entre neuronas que ganan consecutivamente se refuerza, mientras que el resto de las conexiones se debilita en cierta fracción. Se define $p_{i,j}$ como la fuerza de conexión, *connection strength*, entre las neuronas neuronas i, j, de manera que

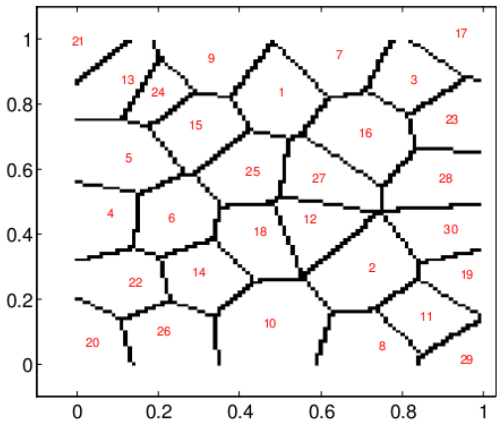
$$\Delta p_{I_{t-1},I_t} = b, \quad p_{i,j}(t) = p_{i,j}(t-1)\left(\frac{N}{N+b}\right) \tag{2.2.11}$$

Lo anterior permite determinar con que fuerza se propagará la actividad de la unidad ganadora al resto de las neuronas de la red. La actividad generada en la neurona i-ésima debido a la k-ésima es

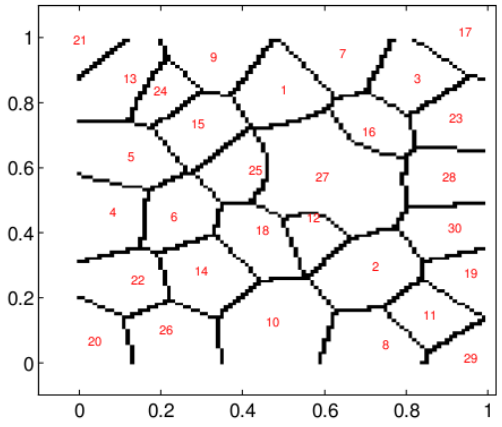
$$a_i(t+1) = \alpha a_i(t) + \frac{\sum_k \left[\{\mu f(\mathbf{d},k) + (1-\mu)\, a_k(t)\}\, p_{k,i}\right]}{max(p)} \tag{2.2.12}$$

donde $0 < \alpha < 1$ es una constante de olvido de la actividad, $\mathbf{d}$ es el vector de distancias entre las neuronas, y $f(\mathbf{d}, k)$ determina como la distancia espacial afecta la actividad en la neurona. Esta función podría ser un delta de dirac eliminando la sumatoria de la ecuación (2.2.12). La neurona ganadora BMU es seleccionada como la que minimiza (2.2.10) con la actividad definida en (2.2.12).

Un fenómeno interesante, debido al parámetro β, es el dinamismo en las celdas de voronoi, las neuronas ya no ganan sólo por la cercanía al patrón de entrada, pues la difusión al disminuir el



(a)



(b)

Figura 2.2.1: Dinamismo en las celdas de voronoi. La Figura b) muestra como las celdas de voronoi crecen con respecto a la Figura a) para el algoritmo SOMTAD. Figura extraída de [25]

umbral, podría permitir que se active una neurona que no sea necesariamente sea las más cercana al ejemplo presentado.

2.2.4. Mapa Auto-organizativos Recursivos

El Mapa Auto-Organizativo Recursivo (RecSOM: *Recursive Self Organizing Map*) [26] posee una representación de contexto más rica que los algoritmos presentados anteriormente, pero a un mayor costo computacional. RecSOM ocupa un vector de contexto c^t, el cual almacena la actividad del mapa completo, así cada neurona $i \in \mathcal{N}$ esta equipada con dos vectores, el vector de pesos $w_i \in \mathbb{R}^d$ y el vector de actividades $c_i \in \mathbb{R}^{|\mathcal{N}|}$. El algoritmo combina ambos vectores para seleccionar

la unidad ganadora (BMU), como la unidad que minimiza el siguiente criterio de distancia

$$d_i(t) = \alpha \left\| x^t - w_i \right\| + \beta \left\| c^t - c_i \right\| \tag{2.2.13}$$

donde los parámetros $\alpha, \beta > 0$ controlan la influencia del contexto. El contexto actual corresponde a la exponencial de las distancias o activaciones de las neuronas en el instante anterior, $c^t = \left(e^{-d_1(t-1)}, \ldots, e^{-d_{|\mathcal{N}|}(t-1)}\right)$. La elección de los parámetros α, β no es trivial, hay valores para los cuales el algoritmo es inestable. Para la elección de estos, hay que considerar que la dimensión del contexto puede ser mucho mayor que la dimensión de los datos, en cuyo caso el valor de β debe ser lo suficientemente pequeño para compensar las diferencia en el cálculo de las normas debido a las diferentes dimensionalidades. Es además recomendable que el peso del contexto en la selección de la BMU sea menor que el de la cuantización, es decir $\beta < \alpha$.

Una cualidad de este modelo es la independencia de la topología de la red, ya que puede ser usado con cualquier tipo de grilla, la adaptación a NG es directa, a diferencia de SOMTAD y GASTAD.

Voegtlin en [26] define una medida de desempeño a fin de poder determinar si la red logra capturar las regularidades temporales de la serie de entrada. Formalmente define una cuantización como una función desde un espacio de entrada a un alfabeto finito, o conjunto de prototipos $L_1, \ldots, L_M$. Cada vector prototipo w_i es asociado a cada letra L_i. Una medida clásica de desempeño de la cuantización es el medir la norma del vector prototipo con respecto al patrón de entrada $\|x^t - w_i\|$, pero esta fórmula es válida sólo cuando no existen dependencias temporales entre los datos.

Es posible generalizar el error de cuantización al ámbito temporal, distinguiendo la naturaleza continua o discreta de la serie de tiempo.

2.2.4.1. Cuantización de Series Discretas

Supongamos una serie de valores discretos, no independiente, idénticamente distribuidos. Para simplificar y sin pérdida de generalidad, supongamos que $x(t) \in \{0, 1\}$. El contexto temporal de la secuencia en el instante t está dado por $c^t = \left(x^t, x^{t-1}, x^{t-2}, \ldots\right) \in \{0, 1\}^{\mathbb{M}}$.

El contexto temporal es representado por un conjunto de alfabetos $L_1, \ldots, L_M$. Para cada instante t, una única letra $L_{k(t)}$ representa a c^t. Luego la reconstrucción de c^t es realizada por el

conjunto de alfabetos. Dado que c^t es infinito, la capacidad de representación de c^t está limitada a la capacidad de representación del alfabeto.

Formalmente, la cuantización del contexto se define como una función de $\{0,1\}^{\mathbb{M}}$ al alfabeto $\{L_1, \ldots, L_M\}$. El conjunto de contextos temporales representados por L_i es denominado campo de recepción *(receptive field)* y es denotado por R_i. El número de eventos pasados que pueden ser reconstruidos por L_i es denominado profundidad del campo receptivo R_i, y es denotado por n_i, lo cual corresponde al largo de la intersección de todos los contextos pertenecientes a R_i.

El conjunto de todos los contextos, $\{0,1\}^{\mathbb{M}}$, puede ser visto como un árbol binario infinito. El campo receptivo R_i corresponde a un conjunto de secuencias binarias que recorren a un camino infinito partiendo de la raíz del árbol. Un único nodo del árbol H_i puede ser asignado a L_i, que corresponde al nodo más profundo del árbol que pertenece a todos los contextos en R_i. La profundidad de R_i es igual a la profundidad del nodo H_i. Dado que cada contexto debe pertenecer a un campo receptivo R_i, cortar el árbol en los nodos $H_1, H_2, \ldots, H_M$ resulta en un árbol finito $\{H_1, H_2, \ldots, H_M\}$ que describe la cuantización completamente.

Si p_i es la probabilidad de que un contexto pertenezca a R_i, entonces el número medio de eventos pasados que pueden ser reconstruidos es:

$$\bar{n} = \sum_{1 \leq i \leq M} p_i n_i \tag{2.2.14}$$

El valor $\bar{n}$ es llamado profundidad del cuantizador, este valor es una medida de calidad del cuantizador. La Figura 2.2.2[3], muestra distintos arboles binarios y la respectiva profundidad de cuantización. En las figuras B y C se puede apreciar nuevamente la mejor capacidad de cuantización de NG con respecto a SOM.Más detalles y demostraciones pueden encontrarse en [27].

2.2.4.2. Cuantización de Series Continuas

El campo receptivo en el caso de series continuas es definido como la media de todas las secuencias que gatilla su selección. Esta definición está restringida a una ventana de tiempo. El error de cuantización temporal (TQE: *Temporal Quantization Error*), se define como la desviación estándar

[3] Figura extraída de [27]

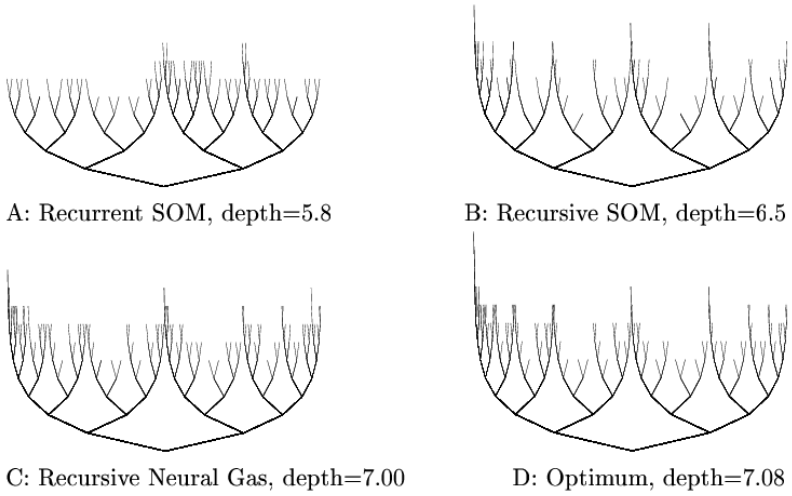


Figura 2.2.2: Arboles binarios correspondiente a los algoritmos A:Mapa Auto-organizativo Recurrente; B,C:Mapa Auto-organizativo Recursivo;D:muestra la profundidad óptima.

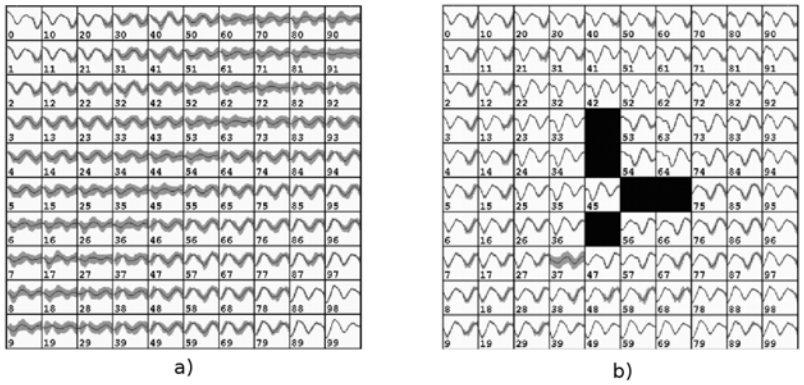


Figura 2.2.3: Comparación del campo receptivo temporal entre a) SOM y b) RecSOM.

de las secuencias asociadas a su campo receptivo. La Figura 2.2.3[4] muestra una comparación del campo receptivo entre SOM y RecSOM. Se puede observar que la desviación estándar de RecSOM es mucho más pequeña que la de SOM. Cinco unidades nunca fueron seleccionadas, aparentemente debido a la dificultad para encontrar un mapeo de baja dimensionalidad para una secuencia temporal que es representada por vectores de alta dimensionalidad.

Un análisis más detallado sobre el cálculo del error de cuantización temporal TQE, puede ser encontrado en la sección 2.2.6, referente a [28].

[4]Figura extraída de [27]

2.2.5. Merge Neural Gas

Merge Neural Gas (MNG) es un modelo no supervisado, rápido, intuitivo y poderoso, orientado al procesamiento de secuencias [14]. El modelo propuesto combina una arquitectura tolerante al ruido, basada en las unidades ganadoras (BMU) de los instantes anteriores, con el cuantizador Neural Gas (NG) [16]. El contexto de MNG se mezcla con la cuantización de la señal para ir construyéndose de manera recursiva. Una de las grandes cualidades de esta arquitectura es que puede ser combinada con grillas arbitrarias.

La red Neuronal MNG está dada por un conjunto de neuronas $\mathcal{N} = \{1, \ldots, M\}$ las cuales son equipadas con un peso $w^i \in \mathbb{R}^d$ y un contexto $c^i \in \mathbb{R}^d$. Dado un conjunto de secuencias temporales $\mathcal{S}_s = \left\{x^t \in \mathbb{R}^d | \forall t = 1 \ldots T_s\right\}$, para $s = 1 \ldots S$ se define I_t como la unidad ganadora en el instante que minimiza el siguiente criterio de distancia recursiva

$$d_i(t) = (1-\alpha)\left\|x^t - w^i\right\|^2 + \alpha\left\|c^t - c^i\right\|^2 \tag{2.2.15}$$

donde c^t corresponde al descriptor del contexto, que es una combinación lineal de las propiedades de la unidad ganadora I_{t-1} en el paso anterior

$$c^t = (1-\beta)w^{I_{t-1}} + \beta c^{I_{t-1}} \tag{2.2.16}$$

donde $0 \leq \beta \leq 1$ es un parámetro que controla la inclusión de la historia de la secuencia.

Durante entrenamiento de la red MNG se recomienda poner mucho cuidado al valor del parámetro $0 < \alpha < 1$ en la distancia recursiva (2.2.15). Se comienza con un pequeño valor de α a fin de que la contribución del contexto sea pequeña. Durante el entrenamiento se puede ir gradualmente aumentando el valor de α para permitir que aumente la contribución del contexto. Avanzado el entrenamiento se puede liberar el parámetro α y orientarlo a un valor que maximice la entropía de activación de la red.

La elección óptima del vector de pesos y del contexto depende de la minimización de la distancia recursiva (2.2.15). Se ha mostrado que disminuyendo el tamaño de la vecindad, al finalizar el entrenamiento, los vectores w_i y c_i convergen a una codificación fractal de los elementos presentados previamente en la secuencia [29].

Teorema 2.2.1 *Dado MSOM con computación recursiva del ganador*

$$d_i(t) = (1-\alpha)\left\|w^i - x^t\right\|^2 + \alpha\left\|c^i - \left((1-\beta)\, w^{I_{t-1}} + \beta c^{I_{t-1}}\right)\right\|^2 \tag{2.2.17}$$

contexto inicial $c^1 := 0$ *y una secuencia con entradas* x^j *para* $j \geq 1$. *Entonces los vectores de peso y contexto óptimos del ganador en el instante* t *son*

$$w^{opt(t)} = x^t, \quad c^{opt(t)} = \sum_{j=1}^{t-1} \beta(1-\beta)^{j-1} x^{t-j} \tag{2.2.18}$$

Estos vectores provienen de la dinámica de aprendizaje como puntos fijos estables, siempre que exista una cantidad suficiente de neuronas, que la cooperación de los vecinos se desvanezca y que los vectores $\sum_{j=1}^{t} \beta(1-\beta)^{j-1} x^{t-j}$ *sean diferentes para cada* t.

En [15] se encontró experimentalmente que la convergencia ocurre primero en los pesos y luego en el contexto, debido a lo anterior se recomienda que los pesos sean actualizados más rápido que el contexto, i.e. poner más atención al *matching* de los patrones que al contexto. Esto se logra escogiendo $\alpha < 0{,}5$. Lo anterior se debe principalmente a que la construcción del contexto se realiza en forma recursiva considerando la unidad ganadora del instante anterior, errores en la cuantización w^{t-1}, generan una mala codificación del contexto.

La red MNG ha sido exitosamente probada en problemas de secuencias de ADN, series caóticas, identificación de locutores, [29] y en detección de husos de sueño [30].

El algoritmo MNG se resume a continuación:

Algoritmo 6 Merge Neural Gas

1. Inicializar aleatoriamente los vectores prototipo y el contexto en cada neurona, $w_i, c_i, \forall i = 1 \dots m$.
2. Presentar una secuencia de entrada $\mathcal{S}_s$
   a) Fijar el contexto en un valor inicial c^1, ej. $\vec{0}$.
   b) Presentar un vector de entrada, x^t, a la red.
   c) Calcular el descriptor del contexto c^t como:
   $$c^t = (1-\beta)w^{I_{t-1}} + \beta c^{I_{t-1}} \tag{2.2.19}$$
   d) Encontrar el BMU, I_t, usando
   $$d_i(t) = (1-\alpha)\left\|x^t - w_i\right\|^2 + \alpha\left\|c^t - c_i\right\|^2 \tag{2.2.20}$$
   e) Actualizar los vectores prototipos, w_i y el contexto c_i, mediante la regla:
   $$\Delta w_i = \epsilon(t) h_\lambda(t)\left(x^t - w_i\right) \tag{2.2.21}$$
   $$\Delta c_i = \epsilon(t) h_\lambda(t)\left(c^t - c_i\right) \tag{2.2.22}$$
   donde $\epsilon(t)$ es la tasa de aprendizaje (2.1.5), T es el número de épocas de entrenamiento y λ se calcula como en (2.1.7).
   f) Incrementar el número de la iteración ($t \to t+1$).
   g) Si $t < T_s$ volver al punto $2b$.
3. Procesar una nueva secuencia ($s \to s+1$)
4. Si $s < S$ volver al punto 2.

2.2.5.1. *BackTracking*

El *BackTracking* está asociado a la idea de poder reconstruir las secuencias asociadas a las distintas unidades de la red. El contexto almacena la información temporal en cada unidad y este puede ser usado para la reconstrucción de tales secuencias.

La reconstrucción se realiza seleccionando la i-ésima unidad de la red e inicializando la lista $L_i = \{i\}$, que almacena las neuronas precedentes a la neurona i-ésima que conforman la reconstrucción de la secuencia mediante el *backtracking*. El objetivo ahora es determinar la neurona precedente a la neurona i-ésima. Esto se hace calculando el *merge* de todas las unidades de la red

$$m_j = \beta c_j + (1-\beta)w_j, \forall j \neq i \tag{2.2.23}$$

donde m_j o *merge* corresponde al descriptor de contexto en cada unidad (durante el entrenamiento el descriptor de contexto se calcula sólo para la unidad ganadora).

Sea $i^{(0)}$ el primer elemento de la lista L_i. Luego la unidad precedente es seleccionada como

$$k = \underset{j \neq i^{(0)}}{\operatorname{argmin}} \| m_j - c_{i^{(0)}} \|^2 \tag{2.2.24}$$

posteriormente la lista L_i es actualizada, colocando el elemento k como precedente a todos los elementos de la lista, es decir,

$$L_i = \{k\} \cup L_i. \tag{2.2.25}$$

Este procedimiento debe realizarse en forma iterativa hasta que el próximo elemento k a ser ingresado en la lista ya pertenezca a ella. El algoritmo se muestra a continuación:

Algoritmo 7 *BackTracking*

---

1. Seleccionar la i-ésima neurona.

   a) Inicializar $L_i = \{i\}$

   b) Para todas las neuronas calcular (2.2.23)

   c) Determinar k usando (2.2.24)

   d) **if** $k \in L_i$
   ir al punto 2,
   else
   ir al punto $1e$
   end if

   e) Actualizar lista (2.2.25)

   f) Volver a $1c$

2. Pasar a la siguiente neurona ($i \rightarrow i + 1$)

La razón de usar de las ecuaciones (2.2.23) y (2.2.24) se debe a la forma en que se construye el contexto en MNG. Supongamos que la neurona i-ésima es la ganadora en el instante presente. Calcular el *merge* de todas las neuronas es necesario para determinar cual de estas fue la ganadora en la etapa anterior, de esta forma el valor m_j equivale al descriptor del contexto c^t en la etapa presente. Dado que la neurona i-ésima minimiza (2.2.15), la ganadora en el instante anterior es la que posee el valor de m_j más cercano a c^i, lo que equivale a encontrar la neurona que minimiza $\|m_j - c_i\|^2$.

2.2.6. Error de Cuantización Temporal

El error de Cuantización Temporal (TQE: *Temporal Quantization Error*) [26, 28], mide la desviación de las secuencias asociadas a cada neurona con respecto a sus campos receptivos. Si el valor calculado es pequeño, significa que el espacio es óptimamente cubierto por la red. En el procesamiento de secuencias el error TQE se calcula para una ventana de tiempo, de esta forma se logra determinar cuan bueno es el desempeño de la red para una cierta cantidad de retardos dada por el ancho de la ventana w.

Dada una secuencia $\{\ldots, s_{t-1}, s_t, s_{t+1}, \ldots\}$, donde s_t corresponde al valor de la secuencia en el instante t, considere el conjunto Θ_i de todos los instantes de tiempo donde la neurona i es ganadora. La activación media $A_i(\tau)$ de la neurona i-ésima para el retardo τ está dada por:

$$A_i(\tau) = \frac{1}{|\Theta_i|} \sum_{j \in \Theta_i} s_{j-\tau} \tag{2.2.26}$$

donde $|\Theta_i|$ corresponde a la cardinalidad del conjunto Θ_i. Para la neurona i-ésima, la desviación de la señal con respecto a la actividad media en el retardo τ, se calcula como:

$$E_i(\tau) = \sqrt{\frac{1}{|\Theta_i|} \sum_{j \in \Theta_i} \left(A_i(\tau) - s_{j-\tau}\right)^2} \tag{2.2.27}$$

La formula anterior corresponde simplemente a la desviación estándar de las series asociadas a la neurona i-ésima en el retardo τ. El error del mapa $e(\tau)$ en el retardo τ está dado por la media de los errores de las unidades que componen la red.

$$e(\tau) = \frac{1}{M} \sum_{i \in \mathcal{N}} E_i(\tau) \tag{2.2.28}$$

Las señales de la Figura 2.2.4, para la neurona i-ésima se forman considerando las secuencias $\{s_k, s_{k-1}, s_{k-2}, \ldots, s_{k-w}\}$, con $k \in \Theta_i$.

Los algoritmos no orientados al procesamiento de secuencias generan una cuantización de los puntos en el instante presente sin considerar similitudes en la dinámica pasada de la serie. Algoritmos como SOM o NG obtienen un valor muy bajo de $e(0)$ y valores elevados a medida que se evalúan retardos anteriores.

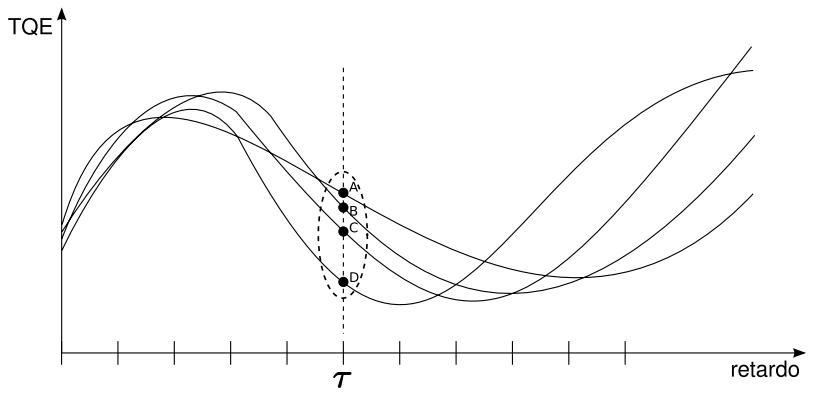


Figura 2.2.4: Señales asociadas a una unidad de la red. En el ejemplo, el error de cuantización temporal de la neurona i-ésima para el retardo τ $E_i(\tau)$, se calcula como la desviación estándar de los puntos A,B,C,D; los cuales pertenecen a las series temporales asociadas a la neurona i.

2.3. Memorias Temporales

La memoria de una red neuronal, está asociada a la capacidad de la red de retener la información presentada anteriormente. para poder utilizarla en el procesamiento del patrón actual. Dos tipos de memoria pueden asociarse a la redes, la de corto y largo plazo. Por lo general se asocia la memoria de largo plazo a los mapas estáticos, dado que estos interpretan la información como pesos los cuales son adaptados mediante la regla de aprendizaje. Los sistemas dinámicos tienen estructuras de corto plazo y a pesar de que las neuronas están equipadas con un vector de pesos, éstas deben además poseer estructuras que logren capturar la dinámica de la serie a fin de diferenciar patrones temporales.

Un ejemplo muy simple consiste en el cruce por cero de una sinusoide, el valor en el instante del cruce t_0 siempre será cero. Un esquema basado en memorias de largo plazo podría aprender la existencia de dicho valor y representarlo mediante cierto vector de pesos, sin embargo el cruce puede ser realizado en forma ascendente o bien descendente. Una memoria de corto plazo podría aprender que para $t < t_0$ existían valores negativos, en el caso ascendente, y positivos, en el caso descendente, logrando diferenciar ambas situaciones.

2.3.1. Memoria mediante ventanas de tiempo

Los datos estáticos suponen independencia temporal entre las muestras. Para el procesamiento de este tipo de datos es indiferente el orden en el cual se presenten los distintos patrones en la etapa

de entrenamiento. Sin embargo esto no es lo más adecuado si el objetivo es buscar o reconocer patrones temporales, donde la muestra en el instante actual depende de las muestras en instantes anteriores.

Una forma de adaptar los algoritmos orientados al procesamiento de datos estáticos para combatir problemas dinámicos es el uso de ventanas de tiempo. La idea básica es que para una serie $x_t \in \mathbb{R}$ se genera el vector de retardos $y_t = [x_t, x_{t-1}, \ldots, x_{t-K}]$, de esta forma la dinámica puede ser capturada dentro de la ventana de tiempo. Un problema que surge es la dificultad para seleccionar el tamaño de la ventana . Una ventana muy pequeña puede no lograr capturar la dinámica de la serie. Una ventana muy grande tiene una serie de desventajas: las redes neuronales escalan peor que linealmente con la dimensionalidad [31], una ventana muy grande de forma natural aumenta la dimensionalidad del problema. Si la serie no es invariante, una ventana de tiempo muy grande tiende a promediar los estadísticos variantes en el tiempo.

El vector y_t puede ser generado mediante la concatenación de la salida de distintas etapas de filtrado, donde (2.3.1) corresponde a la función de transferencia $G(z)$,

$$G(z) = \frac{1}{z}. \tag{2.3.1}$$

El filtrado definido en el dominio del tiempo toma la siguiente forma,

$$\hat{y}(t) = w \bullet y_t = \sum_{k=0}^{K} w_k c_k(t) \tag{2.3.2}$$

donde $c_k(t) = x(t-k)$.

2.3.2. Memorias Gamma

Introducidas por de Vries y Principe [4, 5, 32], las memorias Gamma pueden clasificarse como un filtro híbrido que combina la simpleza de los filtros FIR y la potencia de los filtros IIR. Distintas aplicaciones del filtro Gamma pueden ser encontradas en la literatura [7, 33–37], mostrando su gran potencial en tareas asociadas al procesamiento de datos temporales.

Dos características principales pueden describir este tipo de memoria de corto plazo (*short term memory*), profundidad y resolución (*depth and resolution*). La profundidad se refiere a la capacidad de las memorias para recordar valores anteriores. Una memoria de poca profundidad

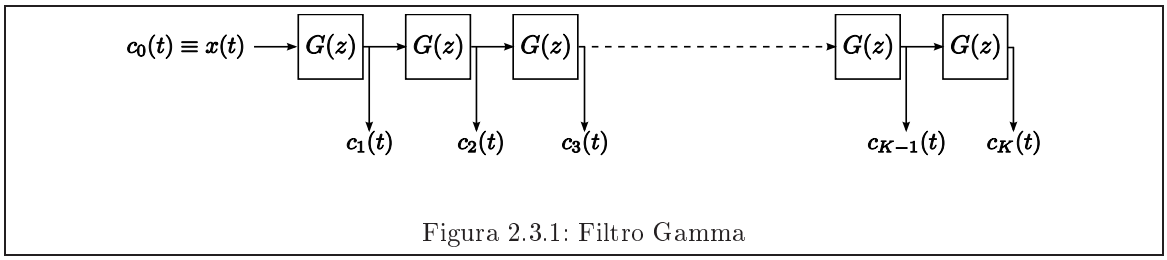


Figura 2.3.1: Filtro Gamma

mantiene sólo la información más reciente. La resolución por el contrario indica el detalle con el que se han almacenado los datos, i.e. con que frecuencia media ha muestreado la memoria los valores anteriores de la secuencia. Desafortunadamente existe un *trade-off* entre la selección de ambas variables, por lo que el aumento de la profundidad lleva inevitablemente a una disminución en la resolución.

El filtro Gamma (2.3.3) es definido en el dominio del tiempo de la siguiente forma.

$$y(t) = \sum_{k=0}^{K} w_k c_k(t) \tag{2.3.3}$$

$$c_k(t) = (1-\mu)\, c_k(t-1) + \mu c_{k-1}(t-1) \tag{2.3.4}$$

donde $c_0(t) \equiv x(t)$, para $k = 1 \ldots K$ y $0 < \mu < 1$ son los parámetros a adaptar. Este tipo de filtro corresponde a un filtro pasa bajos.

El operador de retardo Gamma puede representarse con transformada Z mediante función de transferencia (2.3.5), lo cual facilita el análisis de las distintas propiedades de las memorias Gamma.

$$G(z) = \frac{\mu}{z - (1-\mu)} \tag{2.3.5}$$

considerando que

$$\frac{Y(z)}{X(z)} = G(z) \tag{2.3.6}$$

se tiene

$$Y(z)\,(z - (1-\mu)) = \mu X(z) \tag{2.3.7}$$

despejando Y(z) y tomando antitransformada

$$\begin{aligned} zY(z) &= \mu X(z) + (1-\mu)\, Y(z) \\ \underbrace{y(t+1)}_{c_1(t+1)} &= (1-\mu)\,\underbrace{y(t)}_{c_1(t)} + \mu\,\underbrace{x(t)}_{c_0(t)} \end{aligned} \quad . \tag{2.3.8}$$

La representación mediante funciones de transferencia queda

$$Y(z) = \sum_{k=0}^{K} w_k \left[G(z)\right]^k X(z) \tag{2.3.9}$$

$$H(z) \equiv \frac{Y(z)}{X(z)} = \sum_{k=0}^{K} w_k \left[G(z)\right]^k \tag{2.3.10}$$

de lo anterior es directo que la salida del filtro puede representarse mediante una convolución entre la señal y ciertas funciones g_k (2.3.11) denominadas núcleos de retardo gamma, *gamma delay kernels*.

$$g_k(t) = \frac{\mu^k}{k!} t^k e^{-\mu t} \; \forall k = 0 \ldots K \tag{2.3.11}$$

La Figura 2.3.2 muestra dichos kernels para un valor de $\mu = 0{,}7$.

Es posible determinar la forma del k-ésimo filtrado de la señal evitando la recursión (2.3.4).

$$c_k(t) = \underbrace{G(z) \cdot G(z) \cdots G(z)}_{k-veces} x(t) = \left[G(z)\right]^k x(t) = \left[\frac{\mu}{z-(1-\mu)}\right]^k x(t) \tag{2.3.12}$$

Desarrollando ambos extremos de la ecuación anterior

$$c_k \left[z-(1-\mu)\right]^k = \mu^k x \tag{2.3.13}$$

$$c_k \sum_{j=0}^{k} \binom{k}{j} z^{k-j} (\mu-1)^j = \mu^k x \tag{2.3.14}$$

multiplicando por z^{-k} y despejando el termino asociado a $j = 0$.

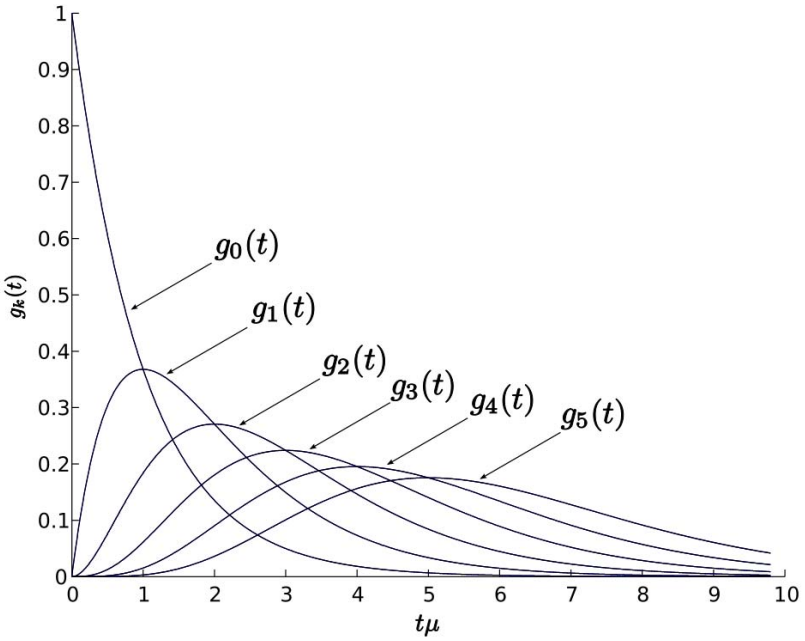


Figura 2.3.2: Kernels Gamma para $\mu = 0{,}7$. El eje x se ha escalado a $t \cdot \mu$ a fin de mostrar como para distintos valores de k la profundidad de la memoria se centra en $D = \frac{k}{\mu}$.

$$c_k = \mu^k z^{-k} x - c_k \sum_{j=1}^{k} \binom{k}{j} z^{-j} (\mu - 1)^j \tag{2.3.15}$$

Finalmente tomando antitransformada Z, se tiene

$$c_k(t) = \begin{cases} \mu^k x(t-k) - \sum_{j=1}^{k} \binom{k}{j} (\mu-1)^j c_k(t-j) & t \geq k \\ 0 & t < k \end{cases} \tag{2.3.16}$$

notar que cuando $k = 0$ la suma desaparece y $c_0(t) = \mu^0 x(t-0) = x(t)$.

Un análisis más detallado de las propiedades del filtro Gamma puede ser encontrado en [4, 38] donde se determina la profundidad D (2.3.17) y la resolución R (2.3.18).

$$D = \frac{K}{\mu} \tag{2.3.17}$$

$$R = \mu \tag{2.3.18}$$

La Figura 2.3.2 muestra como los kernels alcanzan la profundidad para distintos valores de k.

Una segunda versión de las memorias gamma (2.3.19) más versátil, puede actuar como filtro pasa bandas [38].

$$G_{II}(z) = \frac{\mu\left[z-(1-\mu)\right]}{\left[z-(1-\mu)\right]^2+\nu\mu^2}. \tag{2.3.19}$$

La estabilidad se obtiene bajo la condición $0 < \mu(1-\nu) < 2$ y $\mu > 0$.

Los polos que se obtienen encontrando las raíces del denominador de la función de transferencia son:

$$\rho_{1,2} = (1-\mu) \pm \mu\sqrt{-\nu}. \tag{2.3.20}$$

De lo anterior se puede observar que se obtienen polos complejos $\rho_{1,2} = re^{\pm iw_0}$ si $\nu > 0$, con $r = \sqrt{(1-\mu)^2+\nu\mu^2}$ y $w_0 = \tan^{-1}\frac{\mu\sqrt{\nu}}{1-\mu}$.

Desarrollando la expresión $c_k = G_{II}(z)c_{k-1}$ para $k = 1 \ldots K$ es posible encontrar la siguiente recursión para la construcción de los distintos contextos que caracterizan el filtro Gamma II,

$$z^2c_k - 2(1-\mu)zc_k + r^2c_k = \mu z c_{k-1} - \mu(1-\mu)c_{k-1} \tag{2.3.21}$$

multiplicando por z^{-2} y despejando c_k se tiene,

$$c_k = 2(1-\mu)z^{-1}c_k - r^2z^{-2}c_k + \mu z^{-1}c_{k-1} - \mu(1-\mu)z^{-2}c_{k-1} \tag{2.3.22}$$

finalmente tomando antitransformada Z se tiene en el dominio del tiempo,

$$c_k^t = 2(1-\mu)c_k^{t-1} - r^2c_k^{t-2} + \mu c_{k-1}^{t-1} - \mu(1-\mu)c_{k-1}^{t-2} \tag{2.3.23}$$

con $c_0^t \equiv x(t)$.

La Tabla 2.3.1 resume las propiedades de las memorias revisadas en ésta sección

	Retardos	Gamma	Gamma II
Profundidad	k	k/μ	no en forma cerrada
Resolución	1	μ	no en forma cerrada
Estabilidad	Siempre	$0 < \mu < 2$	$0 < \mu(1-\nu) < 2$
Tipo de Filtro	Pasa todo	Pasa bajos	Pasa bandas
Polos	0	1-μ	$\rho_{1,2}$

Tabla 2.3.1: Resumen de características. Retardos, Memoria Gamma y Gamma II.

2.4. Sistemas Dinámicos No Lineales

En un sistema dinámico el comportamiento evoluciona con el tiempo, por ejemplo: número de manchas en el sol, pulso cardiaco, ecuaciones diferenciales, posición de un péndulo, precio del dolar, etc. Los sistemas dinámicos se diferencian en: sistemas lineales y no lineales.

Existe una gran cantidad de herramientas de estudio que permiten estudiar sistemas lineales invariantes, sin embargo cuando el problema es no lineal este es mucho más difícil de resolver y las herramientas lineales no permiten describir con exactitud su comportamiento.

El esquema básico del análisis de señales considera una señal de entrada $x(t)$ la cual es aplicada a un sistema que entrega una salida $y(t)$. La salida puede ser descrita como una convolución entre la entrada y la respuesta al impulso $h(t)$ del sistema,

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau. \tag{2.4.1}$$

La transformada de Laplace permite hacer una transformación desde el espacio del tiempo al espacio de frecuencias, permitiendo facilitar el análisis del sistema. La ecuación (2.4.1) en el espacio de frecuencias queda representada como

$$Y(s) = X(s)H(s) \tag{2.4.2}$$

donde $H(s)$ es la función de transferencia. En un sistema lineal si las entradas x_1 y x_2 producen salidas y_1 e y_2 respectivamente, entonces $x_1 + x_2$ tiene como salida $y_1 + y_2$.

El caso de los sistemas no-lineales es mucho más interesante ya que su comportamiento es más rico. El análisis de los sistemas no-lineales puede realizarse mediante el uso de variables de

estado. Las variables de estado son las variables fundamentales que permiten describir un sistema por completo. Los sistemas espacio-temporales extendidos poseen infinitas variables de estado y sólo pueden ser modelados mediante ecuaciones diferenciales en derivadas parciales (EDPs). Sí el número de variables de estado es finito el sistema puede ser modelado mediante ecuaciones diferenciales ordinarias (EDOs). El modelo en EDOs de un sistema no lineal con un número finito n de variables de estado puede expresarse como

$$\dot{X}(t) = F(X(t)) \tag{2.4.3}$$

donde $X(t) \in \mathbb{R}^n$ es el vector de estado, cuyas coordenadas están dadas por las variables de estado del sistema y $F : \mathbb{R}^n \to \mathbb{R}^n$ es una función no lineal.

Los sistemas dinámicos disipativos poseen espacios de estado de estructura invariante, la cual permanece una vez que el transiente ha desaparecido, estas estructuras se llaman atractores. Hay distintos tipos distintos de atractor.

- **Puntos de equilibrio o punto fijo**. Son sistemas cuyo estado final permanece constantes en el tiempo. Por ejemplo un vaso de agua, el voltaje del condensador en un sistema RC, etc.
- **Órbitas periódicas (ciclos límites)**. Corresponde a una trayectoria cerrada en el espacio de estado, cumplen que al menos una trayectoria se acerca a la órbita periódica a medida que el tiempo se va a infinito. La Figura 2.4.1 muestra un ejemplo de este tipo de atractores.
- **Atractores caóticos**. La dimensión del espacio de estado es no entera (fractal) y son muy sensibles a las condiciones iniciales. El atractor de Lorenz es uno de los atractores caóticos más famosos.

La fibrilación ventricular, actividad eléctrica desorganizada de los ventrículos, corresponde a un atractor caótico, Figura 2.4.2[5]

[5]Imagen extraída de http://www.anestesia.com.mx/disritm.html

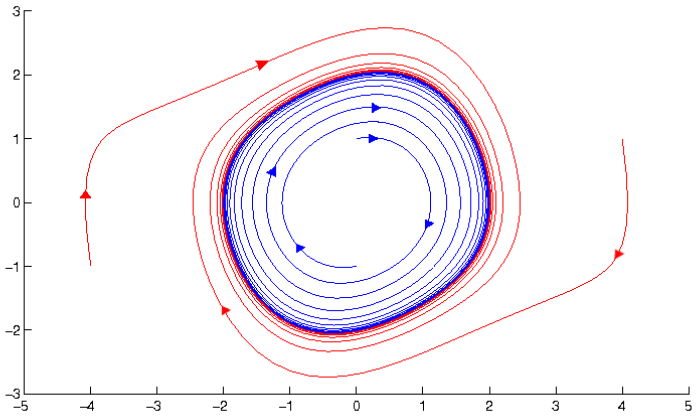


Figura 2.4.1: Ciclo límites, Oscilador de Van der Pol

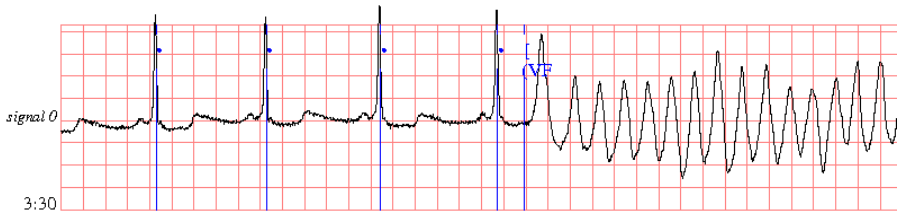


Figura 2.4.2: Taquicardia ventricular que se convierte en fibrilación ventricular.

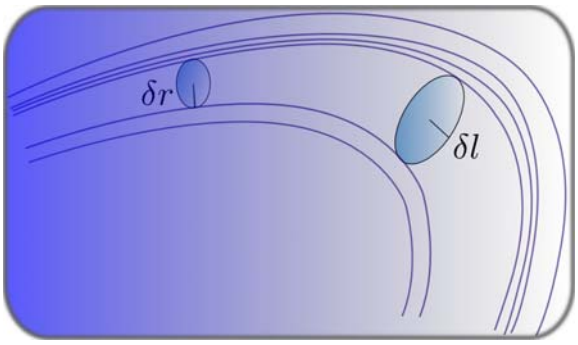


Figura 2.4.3: Exponente de Lyapunov

2.4.1. Exponente de Lyapunov

El exponente de lyapunov es un invariante que caracteriza la taza de separación entre trayectorias del espacio de estado infinitesimalmente cercanas. El número de exponentes de lyapunov esta dado por la dimensión del espacio de estado, de esta forma cada exponente mide la velocidad de divergencia en cada dirección.Cuando se habla del exponente de lyapunov se refiere a máximo valor del conjunto de exponentes y su importancia radica en que está directamente ligado al horizonte de predicción, este se define como:

$$\lambda = \lim_{t\to\infty} \frac{1}{t} \ln \frac{|\delta l_i|}{|\delta r|} \tag{2.4.4}$$

donde $|\delta r|$ es la separación inicial entre dos trayectorias.

El signo de los exponentes de lyapunov indican al convergencia de la serie.

- $\lambda < 0$. La órbita del atractor converge a una órbita periódica o punto fijo estable. Los exponentes negativos son característicos de sistemas disipativos o no conservativos. Mientras más negativo es el exponente, mayor es la estabilidad.
- $\lambda = 0$. La órbita es una órbita periódica. El sistema dinámico es conservativo y estable en el sentido de lyapunov.
- $\lambda > 0$. La órbita es inestable y caótica. No importa que tan cercanos sean dos puntos, en algún momento las trayectorias se separan.

Si la serie se ha discretizado o digitalizado, muchas de las propiedades de un atractor caótico se pierden, por ejemplo la aperiodicidad, sensibilidad a condiciones iniciales, etc. en particular el exponente de lyapunov (2.4.4) ya no puede ser calculado con la trayectoria infinitesimalmente más cercana, para esto se introduce la noción de exponente discreto de lyapunov [39,40]. Para su calculo se considera una biyección $F : Z_M \to Z_M$, $Z_M = \{1, 2, \ldots, M-1\}$, el exponente discreto de lyapunov se define como:

$$\lambda_F = \frac{1}{M} \sum_{i=0}^{M-1} \ln |F(c_i) - F(i)| \tag{2.4.5}$$

donde F es una discretización del atractor real f. c_i es $i+1$ si $i < M-1$, y $c_{M-1} = i-2$ (i.e. c_i es vecino de i).

2.4.2. Reconstrucción de espacio de estado

En problemas de la vida real no siempre es posible tener acceso a todas la variables de estado, pues estas pueden ser no medibles, lo cual hace imposible conocer la dinámica exacta del sistema. Es posible realizar un reconstrucción del espacio de estado a partir de tan sólo una variable de estado, si la reconstrucción es realizada adecuadamente, la dinámica del sistema reconstruido es topológicamente idéntica a la del sistema real, por lo tanto sus invariantes también lo serán.

Una de las técnicas más utilizadas en la reconstrucción es la introducida por Takens en 1980 [13]. Esta consiste en generar un espacio de estado a partir de retardos obtenidos de la misma serie. De esta forma si $x(t) \in \mathbb{R}^1, t = 1 \ldots T$ es una serie de tiempo, el vector $r(t)$ constituye una reconstrucción de dimensión embebida m y retardo τ del espacio de estado original. Es sabido que si $m > 2d+1$, donde d es la dimensión del atractor, se garantiza la preservación topológica de las estructuras del atractor original.

$$r(t) = [x(t), x(t-\tau), x(t-2\tau), \ldots, x(t-(m-1)\tau)] \tag{2.4.6}$$

La teoría no entrega información alguna sobre el valor de τ o de la dimensión embebida m. Distintas herramientas permiten calcular valores adecuados para dichas variables y poder así realizar la reconstrucción. Una de las técnicas convencionales consiste en usar información mutua y falsos vecinos cercanos (FNN: *false nearest neighbours*) para determinar τ y m respectivamente [41].

El primer paso consiste en determinar el retardo τ. Si τ es pequeño, las trayectorias de la reconstrucción del espacio de estado se ubicarán cerca de la diagonal principal, esto hace que las trayectorias sean muy sensibles al ruido, y la existencia de este podría causar cruces inexistentes entre la trayectorias. Por el contrario si τ es muy grande los puntos sucesivos $r(t)$ y $r(t+\Delta t)$ serán no correlacionados y el gran espacio de los puntos en $r(t)$ produce interferencias numéricas en las equivalencias topológicas [42].

La mejor elección para el valor de τ es donde la información mutua $I(\tau) = I(x(t), x(t-\tau))$ presente un mínimo local. Para una valor pequeño de τ la información mutua será alta, cercana a 1. Si τ es muy grande los puntos se volverán muy poco correlacionados, el primer mínimo en la información mutua corresponde a la menor separación posible que presenta también la menor correlación, lográndose así independencia entre las variables de estado.

La elección de la dimensión embebida se realiza en base a la preservación topológica existente que se logra al ir aumentando gradualmente la dimensión m. El algoritmo FNN [43] toma como entrada el valor de τ y de forma secuencial va aumentando la dimensión del vector de espacio de estado. El criterio a medir es la cantidad de falsos vecinos que existen al en la dimensión m. El algoritmo parte con $m = 1$, para cada punto i encuentra el vecino más cercano j, luego aumenta $m = 2$ y compara la separación entre los puntos en el nuevo espacio. Si la distancia entre i y su vecino más cercano j cambió abruptamente, quiere decir que j es un falso vecino de i, cercano en $m = 1$ debido a la proyección. Una versión mejorada del algoritmo de FNN es el introducido por Liangyue Cao [44] el cual no requiere del ajuste de parámetros en la comparación de las distancias, es computacionalmente eficiente y no depende fuertemente de la cantidad de puntos.

2.4.3. Planos de Recurrencia

No existe una receta general para evaluar el comportamiento de un sistema dinámico. Su gran diversidad dificulta de sobremanera la labor de encontrar patrones de similitud entre los sistemas, a pesar de los anterior dichas características existen principalmente en el espacio de estado, he ahí la importancia de poder reconstruir o aproximar el espacio de estado de un cierto proceso. La recurrencia de estados es una propiedad fundamental de los sistemas dinámicos, un estado j se dice recurrente a i, si estos se encuentran arbitrariamente cerca, bajo alguna norma. Si el sistema presenta cierto determinismo entonces es la evolución de las trayectorias deben ser similares durante

un cierto periodo de tiempo.

Por lo general el espacio de estado posee una dimensionalidad que no permite su visualización, Eckmann [45] ha introducido una herramienta que permite visualizar la recurrencia de los estados $\vec{x_i}$ en el espacio de estado, evitando el uso de proyecciones a espacios de menor dimensionalidad, donde el riesgo de no conservar la topología de los estados es alta. Para esto se usa una representación bidimensional de la recurrencia de los estados, si j es recurrente a i se marca con un punto negro en la posición (i, j). Formalmente el Plano de Recurrencia (RP: *Recurrence Plot*) se define como:

$$R_{i,j}^{m,\varepsilon_i} = \Theta(\varepsilon_i - \|\vec{x_i} - \vec{x_j}\|), \quad \vec{x_i} \in \mathbb{R}^m, \quad i, j = 1 \dots N \tag{2.4.7}$$

donde N es el numero de estados a considerar en la representación, ε_i es el umbral del estado $\vec{x_i}$ y $\Theta(\cdot)$ es una función escalón unitario.

Por definición del plano de recurrencia, este posee una recta diagonal dado que $R_{i,i} = 1$, ya que ε_i es distinto para cada i, se tiene que $R_{i,j} \neq R_{j,i}$, puesto que las vecindades de i y j no son necesariamente las mismas. En las pruebas realizadas se utiliza el RP simétrico, donde $\varepsilon_i = \varepsilon, \forall i$, por lo tanto $R_{i,j} = R_{j,i}$.

Existen distintos criterios que permiten fijar el valor de ε, algunos sugieren que el este valor no exceda el 10 % del máximo diámetro del espacio de estado [46], esto es $\varepsilon \leq 0{,}1 \cdot \text{máx}_{i,j} \|\vec{x_i} - \vec{x_j}\|$. Otra alternativa más reciente [47] ajusta el valor de ε a una cantidad de puntos recurrentes del orden del 1 %.

Dentro de las normas más usadas está la norma euclidiana $\|\cdot\|_2$,la norma uno $\|\cdot\|_1$ y la norma infinito $\|\cdot\|_\infty$, presentando esta última ventajas sobre las anteriores, ya que es independiente de la dimensión del espacio de estado.

Es común que se elimine la diagonal del RP a fin de obtener mejores estimaciones del exponente de lyapunov, en [48] sugieren considerar sólo los puntos que cumplen que $|i - j| \geq w$, el valor w es conocido como ventana de Theiler. Distintas variantes del RP [49–51] tratan de abordar distintas problemáticas del RP original.

Una de las variantes más interesantes consiste en el Plano de Recurrencia Cruzado (CRP: *Cross Recurrence Plot*), este permite comparar dos trayectorias del mismo sistemas en el mismo espacio de fase. Formalmente el CRP se construye como:

$$CR_{i,j}^{m,\varepsilon_i} = \Theta(\varepsilon_i - \|\vec{x_i} - \vec{y_i}\|), \quad x_i, y_i \in \mathbb{R}^m, \quad i = 1 \ldots N_x, j = 1 \ldots N_y \tag{2.4.8}$$

donde $\vec{x_i}, \vec{y_i}$ son vectores del mismo espacio de fase. El CRP no corresponde a una matriz cuadrada, dado que N_x no es necesariamente igual a N_y y puede ser útil en tareas donde se busque un patrón en particular, conocido, en este caso el plano de recurrencia marcará dicho patrón.

2.4.3.1. Análisis Cuantitativo del RP (RQA)

Una serie de medidas cuantitativas sobre el RP han sido propuestas en [52–55], conocidas como RQA: *Recurrence Quantification Analysis*. Dado un RP de tamaño NxN, con elementos $R_{i,j}, \forall i, j = 1 \ldots N$, se define:

- RR (*Recurrence Rate*). Corresponde a la densidad de puntos recurrentes en el RP.

$$RR = \frac{1}{N^2} \sum_{i,j=1}^{N} R_{i,j} \tag{2.4.9}$$

- DET (*Determinism*). Indica la predictibilidad del sistema. Esta es una medida basada en la distribución de los trazos diagonales de largo l , $P^{\varepsilon}(l) = \{l_i | i = 1 \ldots N_l\}$, donde N_l es el numero absoluto de lineas diagonales

$$DET = \frac{\sum_{l=l_{min}}^{N} l P^{\varepsilon}(l)}{\sum_{i,j=1}^{N} R_{i,j}} \tag{2.4.10}$$

  l_{min} es un umbral que permite considerar sólo las diagonales que cumplen con el un largo suficiente.

- L (*Average Diagonal Length*). Indica el largo promedio de las diagonales

$$L = \frac{\sum_{l=l_{min}}^{N} l P^{\varepsilon}(l)}{\sum_{l=l_{min}}^{N} P^{\varepsilon}(l)} \tag{2.4.11}$$

- DIV (*Divergence*). El largo de las diagonales está asociado al exponente de lyapunov, DIV indica el comportamiento de la divergencia de las trayectorias.

$$DIV = \frac{1}{L_{max}} \tag{2.4.12}$$

donde $L_{max} = \text{máx}\{l_i | i = 1 \dots N_l\}$.

- ENTR (*Entropy*). Corresponde a la entropía de Shannon de la frecuencia de distribución de las líneas diagonales. Refleja la complejidad del determinismo del sistema.

$$ENTR = -\sum_{l=l_{min}}^{N} p(l) \ln p(l), \quad p(l) = \frac{P^{\varepsilon}(l)}{\sum_{l=l_{min}}^{N} P^{\varepsilon}(l)} \tag{2.4.13}$$

- LAM (*Laminarity*). Es una medida de las estructuras verticales en el RP, la que representa la ocurrencia de estados laminares[6].

$$LAM = \frac{\sum_{v=v_{min}}^{N} vP^{\varepsilon}(v)}{\sum_{v=1}^{N} vP^{\varepsilon}(v)} \tag{2.4.14}$$

- TT (Trapping Time). Corresponde al largo promedio de las lineas verticales, es el tiempo medio en que el sistema se encuentra en un mismo estado.

$$TT = \frac{\sum_{v=v_{min}}^{N} vP^{\varepsilon}(v)}{\sum_{v=v_{min}}^{N} P^{\varepsilon}(v)} \tag{2.4.15}$$

Marwan define también $V_{max} = \text{máx}\{v_l | l = 1 \dots L\}$, un análogo a L_{max}.

[6]La laminaridad se puede definir como la transición entre caos y caos. Por ejemplo, la transición de un ritmo cardiaco normal a una fibrilación ventricular.

Capítulo 3

Modelo de Contextos Gamma

En este capítulo se propone un modelo de contextos orientado al procesamiento de secuencias. El modelo propuesto al igual que otros modelos de Cuantización Temporal (TKM, RSOM, RecSOM, MSOM etc), está montado sobre un esquema de cuantización estático como SOM o Gas Neuronal, los cuales realizan la cuantización espacial de las secuencias presentadas, siendo responsabilidad del modelo de contextos lograr capturar la dinámica de la secuencia. Una de las ventajas del modelo propuesto es que es independiente de la topología de la red. Al combinarse con arquitecturas SOM, Neural Gas, Growing Neural Gas, etc. da origen respectivamente a los algoritmos *Gamma Self-Organizing Map* (Gamma SOM), *Gamma Neural Gas* (Gamma NG), *Gamma Growing Neural Gas* (Gamma GNG).

El modelo de contexto basado en Redes Neuronales Auto-Organizativas combina una arquitectura de aprendizaje tolerante al ruido, construida sobre un modelo simple que utiliza referencia hacia el ganador de la etapa previa para construir un eficiente modelo de contextos. De esta forma toda la red aporta a la construcción de una representación interna de la dinámica.

El modelo de contextos Gamma debe su nombre a las memorias Gamma o filtros Gamma, que son la base en el diseño de esta estructura. Se identifican principalmente dos parámetros: el número de contextos K y el parámetro de fusión β. La elección de estos dos parámetros permiten ajustar la profundidad de la memoria y la resolución, como en el filtro Gamma.

Una de las características más interesantes del modelo de contextos propuesto es que generaliza el modelo MSOM [14], resultando este último un caso particular del modelo de Contextos Gamma

cuando $K = 1$.

3.1. Cuantización de Secuencias

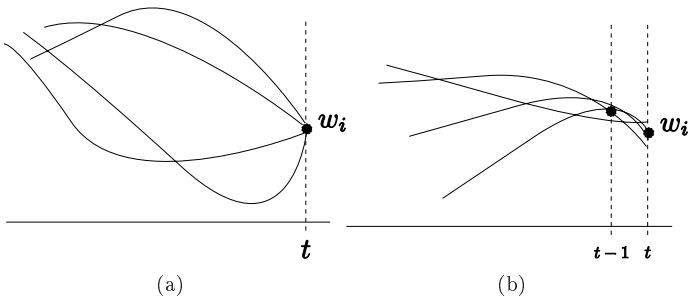
A continuación se muestra que el modelo de contextos Gamma, permite agrupar secuencias similares. La demostración se realiza usando el funcional de minimización del Gas Neuronal, el cual es capaz de agrupar vectores similares. Se realiza luego una adaptación al caso temporal mediante el uso de ventanas de tiempo, las cuales finalmente son adaptadas para derivar el funcional asociado al modelo de contextos.

Se comienza mostrando que Gas Neuronal agrupa vectores similares en el instante t, luego al considerar una ventana de tiempo de tamaño τ, los vectores agrupados por el Gas Neuronal también coinciden aproximadamente en los instantes $t-1, t-2, \ldots t-\tau$.

Sea $\vec{x}^t \in \mathbb{R}^d$. El funcional de minimización E utilizado por Gas Neuronal es el siguiente

$$E = \sum_{i=1}^{N} \sum_{t=1}^{T} h_\lambda \left(k_i(\vec{x}^t, \vec{w}) \right) \left\| \vec{x}_t - \vec{w}^{I_t} \right\|^2 \tag{3.1.1}$$

donde I_t corresponde a la unidad ganadora asociada al patrón x_t. El algoritmo Neural Gas genera para cada neurona i una partición del subespacio tal que los elementos asociados a la neurona i cumplen $V_i = \left\{ \vec{x}^t \mid \left\| \vec{x}^t - w^i \right\| \leq \left\| \vec{x}^t - w^j \right\| \forall j \right\}$. La Figura 3.1.1.a) muestra que para una cierta neurona $\vec{w}^i$, las secuencias aprendidas por la neurona calzan en el instante t, pero no en instantes anteriores puesto que dicha información no se encuentra en el funcional E.



(a) (b)

Figura 3.1.1: Cuantización Gas Neuronal. Para una cierta neurona w_i. a) las secuencias aprendidas por la neurona calzan perfectamente en el instante t, pero no en instantes previos. b) las secuencias aprendidas por la neurona calzan en el instante t y $t-1$ (ventana de tamaño 2), pero no en instantes anteriores.

Una adaptación que permite agregar temporalidad en el funcional E consiste en considerar ventanas de tiempo. Supongamos una ventana de tamaño 2, lo que es equivalente a considerar el vector de entrada $\vec{\mathbf{x}}_t = [\vec{x}_t, \vec{x}_{t-1}]$, con lo que el funcional toma la siguiente forma

$$E = \sum_{n=1}^{N} \sum_{t=1}^{T} h_\lambda \left(k_i(\vec{\mathbf{x}}_t, \vec{\mathbf{w}}) \right) \left\| \vec{\mathbf{x}}_t - \vec{\mathbf{w}}^{I_t} \right\|^2 \tag{3.1.2}$$

donde $dim(\vec{\mathbf{x}}_t) = dim(\vec{\mathbf{w}}^i) = 2d$. Esto permite que para un cierta neurona $\vec{\mathbf{w}}^i = [\vec{w}_1^i, \vec{w}_2^i]$, las secuencias aprendidas sean tales que

$$\left\| \vec{x}_t - \vec{w}_1^i \right\|^2 + \left\| \vec{x}_{t-1} - \vec{w}_2^i \right\|^2 \leq \left\| \vec{x}_t - \vec{w}_1^j \right\|^2 + \left\| \vec{x}_{t-1} - \vec{w}_2^j \right\|^2 \forall j \tag{3.1.3}$$

según se muestra en la Figura 3.1.1.b).

El desempeño de la cuantización en el sentido de poder considerar secuencias similares más largas se puede mejorar aumentando el tamaño de la ventana $\vec{\mathbf{x}}_t = [\vec{x}_t, \vec{x}_{t-1}, \ldots, \vec{x}_{t-\tau}]$, la estructura del funcional 3.1.2 se mantiene.

3.1.1. Funcional de Cuantización Vectorial E, en el modelo de Contextos

A partir del funcional (3.1.2), considerando una ventana de tamaño K, se muestra una representación alternativa que permite realizar la misma tarea de cuantización temporal. Sea un vector de entrada $\vec{\mathbf{x}}_t = [\vec{x}_t, \vec{x}_{t-1}, \ldots, \vec{x}_{t-K}]$, y un conjunto de neuronas $\mathcal{N}$ donde cada neurona $\vec{\mathbf{w}}^i = [\vec{w}_0^i, \vec{w}_1^i, \ldots, \vec{w}_K^i]$. Desarrollando la distancia, se tiene:

$$\left\| \vec{\mathbf{x}}_t - \vec{\mathbf{w}}^i \right\|^2 = \left\langle \vec{\mathbf{x}}_t - \vec{\mathbf{w}}^i, \vec{\mathbf{x}}_t - \vec{\mathbf{w}}^i \right\rangle = \sum_{d=0}^{K} \left(\vec{x}_{t-d} - \vec{w}_d^i \right)^2 = \left(\vec{x}_{t-d} - \vec{w}_0^i \right)^2 + \sum_{d=1}^{K} \left(\vec{x}_{t-d} - \vec{w}_d^i \right)^2 \tag{3.1.4}$$

cambiando la notación y haciendo $\vec{w}^i \equiv \vec{w}_0^i$ y $\vec{c}_k^i \equiv \vec{w}_k^i, \forall k = 1 \ldots K$, la ecuación anterior se puede interpretar de diferente manera: cada neurona posee un peso $\vec{w}^i$ encargado de realizar la cuantización en el instante t y un conjunto de K elementos $\vec{c}_k^i$ denominados contexto encargados

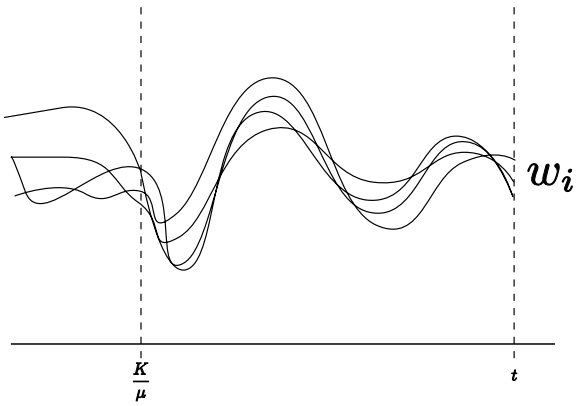


Figura 3.1.2: Secuencias agrupadas según el criterio del funcional (3.1.6)

de capturar la temporalidad. Así el nuevo funcional considerando el cambio de notación queda

$$E = \sum_{n=1}^{N} \sum_{t=1}^{T} h_\lambda \left(k_i(\vec{x}_t, \mathcal{N}) \right) \cdot \left\{ (\vec{x}_t - \vec{w}^i)^2 + \sum_{k=1}^{K} (\vec{x}_{t-d} - \vec{c}_k^i)^2 \right\} \tag{3.1.5}$$

si en lugar de utilizar retardos simples (sección 2.3.1) $\vec{x}_{t-d}$, se considera el Filtro Gamma (sección 2.3.2), la ecuación (3.1.5), queda

$$E = \sum_{n=1}^{N} \sum_{t=1}^{T} h_\lambda \left(k_i(\vec{x}_t, \mathcal{N}) \right) \cdot \left\{ (\vec{x}_t - \vec{w}^i)^2 + \sum_{k=1}^{K} (\vec{c}_k^t - \vec{c}_k^i)^2 \right\} \tag{3.1.6}$$

donde $\vec{c}_k^t$ corresponde al filtrado de orden k de la señal de entrada $\vec{x}_t$.

El uso de filtros Gamma desacopla la profundidad de la memoria K, logrando llegar a una profundidad K/μ sacrificando resolución. Además, el uso de este filtro pasa bajos permite lograr una mayor tolerancia al ruido.

El funcional 3.1.6 hace uso de las salidas del Filtro Gamma aplicadas sobre la señal x_t para realizar la cuantización de las señales, permitiendo agrupar secuencias similares.

3.2. Modelo de Contexto en Merge Neural Gas

En MNG el contexto óptimo converge a una codificación fractal de los elementos de la secuencia presentados previamente [29]. La elección óptima del vector de contextos y el vector de pesos se

obtiene mediante la minimización de la distancia al patrón presentado y al descriptor del contexto c^t respectivamente [15]. El Teorema 2.2.1 caracteriza el valor de c^{opt}. Desarrollando la recursión que define la construcción del contexto en MNG, ecuación (2.3.4), se tiene:

$$\begin{aligned}
c^t &= (1-\beta)c^{t-1} + \beta x^{t-1} = (1-\beta)((1-\beta)c^{t-2} + \beta x^{t-2}) + \beta x^{t-1} \\
c^t &= (1-\beta)^2 c^{t-2} + (1-\beta)\beta x^{t-2} + \beta x^{t-1} \\
c^t &= (1-\beta)^2((1-\beta)c^{t-3} + \beta x^{t-3}) + (1-\beta)\beta x^{t-2} + \beta x^{t-1} \\
c^t &= (1-\beta)^3 c^{t-3} + (1-\beta)^2\beta x^{t-3} + (1-\beta)\beta x^{t-2} + \beta x^{t-1} \\
c^t &= \textstyle\sum_{j=1}^{t} \beta(1-\beta)^{j-1} x^{t-j}
\end{aligned}$$

Se puede ver que la codificación fractal óptima c^{opt} de la ecuación (2.2.18) corresponde a una primera etapa de filtrado mediante el uso de Memorias Gamma (2.3.4). Es posible deducir la regla de entrenamiento de Merge mediante un desarrollo directo de la función de transferencia del Filtro Gamma

$$\begin{aligned}
C(z) &= G(z)X(z) = \tfrac{1-\beta}{z-\beta}X(z) = \tfrac{z^{-1}-\beta z^{-1}}{1-\beta z^{-1}}X(z) \\
&= (1-\beta)z^{-1}X(z) + \beta z^{-1}C(z)
\end{aligned} \tag{3.2.1}$$

finalmente tomando antitransformada Z, se obtiene

$$c^t = (1-\beta)x^{t-1} + \beta c^{t-1}. \tag{3.2.2}$$

Si se considera que el peso de la neurona ganadora es la mejor aproximación del patrón de entrada, es posible adaptar la ecuación del descriptor de contexto reemplazando x^{t-1} con $w^{I_{t-1}}$ y c^{t-1} con $c^{I_{t-1}}$para recuperar la ecuación 2.2.19.

De lo anterior se tiene que dada una red MNG y el parámetro de merge β, la profundidad de la memoria D_{MNG} y la resolución R_{MNG} quedan caracterizadas por

$$D_{MNG} = \frac{1}{1-\beta}, \quad R_{MNG} = 1-\beta \tag{3.2.3}$$

según lo calculado para las memorias gamma.

En consecuencia el modelo de contextos usado por MNG puede ser mejorado en el sentido de profundidad de la memoria y resolución mediante el uso de Memorias Gamma. Introduciendo más etapas de filtrado se puede lograr un aumento en la profundidad de la memoria sin necesidad de comprometer la resolución, mejorando la representación de la dinámica del sistema, o bien mejorar

la resolución sin comprometer la profundidad de la memoria.

3.3. Modelo Propuesto de Contextos Gamma Neural Gas

Dado que el contexto se construye referido al ganador de la etapa previa, el modelo necesita de un conjunto de neuronas $\mathcal{N} = \{1, \ldots, m\}$ equipadas con un vector de cuantización $w^i \in \mathbb{R}^d$ y un conjunto de contextos $\mathcal{C} = \left\{c_1^i, c_2^i, \ldots, c_K^i\right\}$, $c_k^i \in \mathbb{R}^d$, $k = 1, \ldots, K$, cuya cardinalidad K queda dada por la cantidad de etapas de filtrado a considerar en cada neurona. Para cada secuencia s los contextos del conjunto $\mathcal{C}$ deben ser inicializados en un valor arbitrario, por ejemplo 0.

Dado un patrón de entrada x^t la unidad ganadora I_t corresponde a la neurona que minimiza el criterio de distancia (3.3.1).

$$d_i(t) = \alpha_w \left\|x^t - w^i\right\|^2 + \sum_{k=1}^{K} \alpha_k \left\|c_k^t - c_k^i\right\|^2 \tag{3.3.1}$$

donde los valores α_k, $k \in \{1, 2, \ldots, K\} \cup \{w\}$ determinan la importancia de los distintos elementos en cada neurona. La unidad ganadora BMU se determina mediante (3.3.2).

$$I_t = \underset{i=1,\ldots,M}{\operatorname{argmin}}\, d_i(t) \tag{3.3.2}$$

La distancia (3.3.1) en el algoritmo Gamma NG requiere del cálculo de todos los descriptores de contexto asociados a las distintas etapas de filtrado. Estos son construidos mediante una adaptación de las Memorias Gamma [5–7]. El descriptor de contexto es construido con la colaboración de toda la red. La unidad ganadora en la etapa anterior I_{t-1} es seleccionada para calcular los K descriptores de contexto en la unidad actual.

$$c_k^t = \beta c_k^{I_{t-1}} + (1 - \beta)\, c_{k-1}^{I_{t-1}} \quad \forall k = 1, \ldots, K \tag{3.3.3}$$

donde $c_0^{I_{t-1}} \equiv w^{I_{t-1}}$ y en $t = 1$, las condiciones iniciales $c_k^{I_0}, \forall k = 1, \ldots, K$ son ajustadas de forma arbitraria.

El parámetro $0 \leq \beta \leq 1$ permite ajustar la profundidad de la memoria (2.3.17).

Debido a la naturaleza recursiva en la construcción del contexto es recomendable que $\alpha_w > \alpha_1 > \alpha_2 > \cdots > \alpha_K > 0$, de lo contrario es muy probable que una mala cuantización en las primeras

etapas del filtrado produzca errores en la construcción de los contextos de orden superior.

Para una red Gamma NG y el parámetro de merge β, la profundidad de la memoria $D_{GammaNG}$ y la resolución $R_{GammaNG}$ quedan caracterizadas por

$$D_{GammaNG} = \frac{K}{1-\beta}, \quad R_{GammaNG} = 1 - \beta \tag{3.3.4}$$

3.3.1. Regla de Entrenamiento de Vectores Prototipos y Vectores de Contexto

Mediante la definición de un Funcional de Energía para mapas auto-organizativos [56], es posible derivar una regla de entrenamiento para los Vectores Prototipos, encargados de realizar la cuantización estática y los Vectores de Contexto, encargados de realizar la representación temporal.

El Funcional de minimización se basa en la idea propuesta en la Sección 3.1.1, agregando los vectores de contexto al funcional de energía del algoritmo Neural Gas [57]. De esta forma se tiene

$$E = \sum_i h_\lambda \left(k_i(x^t, W)\right) d_i(x^t, W) \tag{3.3.5}$$

donde $d_i(x^t, W)$ corresponde a la distancia definida en la ecuación (3.3.1).Así el funcional buscado corresponde a

$$E = \sum_i h_\lambda \left(k_i(x^t, W)\right) \cdot \left\{ \alpha_w \left\| x^t - w^i \right\|^2 + \sum_{k=1}^{K} \alpha_k \left\| c_k^t - c_k^i \right\|^2 \right\} \tag{3.3.6}$$

de esta forma para un cierto instante t

$$-\frac{\partial E(t)}{\partial w^i} = \alpha_w h_\lambda \left(k_i(x^t, W)\right) (x^t - w^i) \tag{3.3.7}$$

$$-\frac{\partial E(t)}{\partial c_k^i} = \alpha_k h_\lambda \left(k_i(x^t, W)\right) (c_k^t - c_k^i) \tag{3.3.8}$$

donde $k_i(x^t, W)$ corresponde al ranking de la neurona i-ésima según las distancias $d(x^t, W)$.

Según lo anterior los pesos w^i y los contextos c_k^i son actualizados de acuerdo a

$$\Delta w^i = \epsilon(t) h_{i,I_t}(t) \left(x^t - w^i\right) \tag{3.3.9}$$

$$\Delta c^i = \epsilon(t) h_{i,I_t}(t) \left(c_k^t - c_k^i\right) \tag{3.3.10}$$

Notar que las reglas definidas en las ecuaciones anteriores son independientes de la topología de la red, dando origen a los algoritmos Gamma SOM y Gamma NG descritos en 3.5 y 3.6 respectivamente.

3.3.2. Parámetro α

En este segmento se presentan algunos detalles a considerar sobre el parámetro α. Este parámetro controla controla la contribución del peso w^i y de los distintos contextos $c_k^i, \forall k = 1 \dots K$, en la función de distancia (3.3.1). La contribución inicial del contexto en términos del cálculo de la distancia debe ser bajo, fijando un valor positivo pequeño para $\alpha_k, \forall k = 1 \dots K$. Esto se debe a que es de gran importancia lograr una buena cuantización de los valores de la señal en etapas iniciales. Si nos fijamos en la construcción del primer contexto, es posible observar que este se construye en función del peso de la neurona ganadora en el instante previo, esto es

$$c^t = (1 - \beta) w^{I_{t-1}} + \beta c^{I_{t-1}} \tag{3.3.11}$$

notemos que $w^{I_{t-1}}$ es el mejor representante de x^{t-1}, por lo tanto si la red no logra una buena representación en los valores de w^i el contexto no se construye en función de una buena representación de la dinámica de la señal.

Posteriormente, en una segunda etapa, cuando el peso w^i logra entregar una buena representación de la señal se pueden aumentar los valores de α_k para comenzar a construir las distintas etapas de contexto. En esta etapa de forma simultánea se disminuye la tasa de aprendizaje del peso w^i para no perder la cuantización debido al ruido que entrega en contexto en la función de distancia cuando este aún no logra estar correctamente entrenado.

3.4. Propiedades del Modelo de Contextos Gamma

Mediante el uso de Memorias Gamma, para la construcción del contexto, se generaliza el modelo utilizado por Merge. Al igual que el contexto de Merge, la regla de entrenamiento del modelo

propuesto se deriva directamente de la función de transferencia del Filtro Gamma

$$\begin{aligned} C_k(z) &= G(z)C_{k-1}(z) \\ &= \frac{1-\beta}{z-\beta}C_{k-1}(z) \\ &= \frac{z^{-1}-\beta z^{-1}}{1-\beta z^{-1}}C_{k-1}(z) \end{aligned} \tag{3.4.1}$$

$$C_k(z) - \beta z^{-1}C_k(z) = (1-\beta)z^{-1}C_{k-1}(z) \tag{3.4.2}$$

despejando el término $C_k(z)$

$$C_k(z) = (1-\beta)z^{-1}C_{k-1}(z) + \beta z^{-1}C_k(z) \tag{3.4.3}$$

finalmente tomando antitransformada Z

$$c_k^t = (1-\beta)c_{k-1}^{t-1} + \beta c_k^{t-1} \tag{3.4.4}$$

3.4.1. Convergencia de los Contextos

En la Sección 3.1.1 se mostró como una red orientada al procesamiento de datos estáticos es capaz de realizar el agrupamiento de secuencias, con la ayuda de información adicional sobre la historia de los datos.

Según la definición del modelo de Contextos Gamma, mediante un eficiente modelo recursivo que utilizada la unidad ganadora en el instante previo, los contextos son autogenerados por la red, esta representación interna basada en memorias de corto plazo permite a la red agrupar secuencias similares, prescindiendo de cualquier otra información que no sea la secuencia de entrada.

A continuación se muestra que los contextos generados por la red convergen a las distintas etapas de filtrado del Filtro Gamma, lo cual corresponde a un caso particular de memoria de corto plazo.

Dado un cierto patrón de entrada x^t la unidad ganadora I_t corresponde a la neurona que minimiza el criterio de distancia (3.3.1). A pesar de que el contexto es generado mediante una recursión que involucra las pasadas unidades ganadoras, la red no es recursiva, puesto que ésta básicamente actúa como un intermediario, generador del contexto, entre una etapa y la siguiente [58].

Un análisis de la ecuación (3.3.1) permite determinar los pesos y contextos óptimos en la cuantización vectorial, suponiendo que la cantidad de épocas de entrenamiento es lo suficientemente grande como para asumir que la actividad inicial, debido a la inicialización de la red, es insignificante. El análisis, cuando w es constante, es de la siguiente manera:

$$\frac{\partial d(t)}{\partial w} = -2\alpha_w \left\| x^t - w \right\| (x^t - w) \tag{3.4.5}$$

cuando w es óptimo en el sentido de la cuantización, la derivada (3.4.5) se anula, por lo tanto sustituyendo en el lado izquierdo de la ecuación (3.4.5)

$$0 = -2\alpha_w \left\| x^t - w \right\| (x^t - w) \tag{3.4.6}$$

se tiene:

$$w = x^t \tag{3.4.7}$$

Mediante un análisis similar al realizado con el peso, se puede encontrar que el contexto óptimo converge a $c_k = c_k^t$.

Una expresión en función de x^t puede ser encontrada considerando que el contexto es generado según la recurrencia $c_k^t = \beta c_k^{I_{t-1}} + (1-\beta)c_{k-1}^{I_{t-1}}$. Asumiendo $c_0 \equiv w = x^t$, se tiene que

$$c_1 = c_1^t = \beta c_1^{I_{t-1}} + (1-\beta)c_o^{I_{t-1}} \tag{3.4.8}$$

$$c_1^t = \beta c_1^{t-1} + (1-\beta)x^{t-1} \tag{3.4.9}$$

reemplazando $c_1^{t-1} = \beta c_1^{t-2} + (1-\beta)x^{t-2}$ en la ecuación anterior se tiene

$$c_1^t = \beta(\beta c_1^{t-2} + (1-\beta)x^{t-2}) + (1-\beta)x^{t-1} \tag{3.4.10}$$

$$c_1^t = \beta^2 c_1^{t-2} + \beta(1-\beta)x^{t-2} + (1-\beta)x^{t-1} \tag{3.4.11}$$

reemplazando $c_1^{t-2} = \beta c_1^{t-3} + (1-\beta)x^{t-3}$ en la ecuación anterior se tiene

$$c_1^t = \beta^2(\beta c_1^{t-3} + (1-\beta)x^{t-3}) + \beta(1-\beta)x^{t-2} + (1-\beta)x^{t-1} \tag{3.4.12}$$

$$c_1^t = \beta^3 c_1^{t-3} + (1-\beta)\beta^2 x^{t-3} + (1-\beta)\beta x^{t-2} + (1-\beta)x^{t-1} \tag{3.4.13}$$

el desarrollo anterior deriva en la siguiente fórmula para el contexto

$$c_1^t = \sum_{j=1}^{t-1} (1-\beta)\beta^{j-1} x^{t-j} \tag{3.4.14}$$

En general, siguiendo el mismo procedimiento para los contextos superiores se puede demostrar que

$$c_k^t = \sum_{j=1}^{t-1} (1-\beta)\beta^{j-1} c_{k-1}^{t-j}. \tag{3.4.15}$$

Las ecuaciones (3.4.14) y (3.4.15) corresponden exactamente a la definición del Filtro Gamma, por lo tanto los contextos óptimos en el Modelo de contextos Gamma convergen a las distintas etapas de filtrado del Filtro Gamma. De aquí también se deduce que el modelo de Contextos minimiza el funcional definido en (3.1.6), evitando la necesidad de entregar a la red el filtrado como entrada, permitiendo que ésta cree su propia representación contextual de la dinámica de la secuencia.

3.4.2. Interpretación basada en espacio de estado

Hasta el momento se ha mostrado la estrecha relación existente entre el modelo de contextos propuesto y las Memorias Gamma, primero en un sentido el modelo puede ser derivado de forma directa de la función de transferencia. En otro sentido el modelo, bajo ciertas restricciones sobre la red, converge a las distintas etapas de filtrado del Filtro Gamma.

Una interpretación basada en la reconstrucción de espacio de estado de la secuencia puede ser realizada, a fin de explicar el potencial del modelo de contextos Gamma sobre el modelo de contextos utilizado por Merge.

Como se ha visto en la sección 2.4.2 mediante el uso del Teorema de Takes es posible definir un vector

$$r(t) = [x(t), x(t-\tau), x(t-2\tau), \ldots, x(t-(m-1)\tau)] \tag{3.4.16}$$

que preserva la estructura topológica del atractor original. Mediante el algoritmo FNN [43] es posible determinar la dimensión del vector r. El algoritmo FNN busca una dimensión m tal que

la topología al pasar a la siguiente dimensión $m+1$ preserve las vecindades locales, finalmente se logra determinar una base ortogonal de dimensión m, donde vive la reconstrucción.

De manera similar a la reconstrucción de espacio de estado, el algoritmo MSOM o MNG crea una representación interna, cuantizada, del espacio de estado de dimensión 2

$$r(t) = [w^{I_t}, c^{I_t}] \tag{3.4.17}$$

que considera el peso de la neurona ganadora y el contexto.

Mediante el uso de una mayor cantidad de contextos, el Modelo de Contextos Gamma se crea una base de tamaño K

$$r(t) = [w^{I_t}, c_1^{I_t}, \ldots, c_{K-1}^{I_t}] \tag{3.4.18}$$

permitiendo a la red crear una representación interna de topológicamente más similar al espacio de estado original de la señal. Esto se comprueba experimentalmente mediante el uso de Planos de Recurrencia RPs.

3.5. Algoritmo Gamma SOM

En esta sección se describe el algoritmo Gamma SOM, el cual corresponde a la fusión entre el Mapa Auto-Organizativo (SOM) y el Modelo de Contextos Gamma. La topología inicial de la red Gamma SOM es una grilla de $N_x \times N_y$. La neurona i tiene asociada un vector prototipo d-dimensional, w^i , un conjunto de contextos d-dimensionales c_k^i y un vector posición bidimensional, p_i, que se mantienen fijos en la grilla de salida. Esta topología es análoga a la de la red SOM.

El algoritmo Gamma SOM se resume a continuación.

Algoritmo 8 Algoritmo Gamma SOM

1. Inicializar aleatoriamente los vectores prototipo, $w^i, c_k^i, \quad \forall i = 1 \dots M, \forall k = 1 \dots K$.
2. Presentar un vector de entrada, $x(t)$, a la red.
3. Calcular el descriptor de contexto c^t usando (3.3.3)
4. Encontrar el BMU, I_t, usando (3.3.2)
5. Adaptar las unidades de la red.
   a) Actualizar los vectores prototipo, w^i , mediante la regla (3.3.9).
   b) Actualizar los vectores de contexto, c^i , mediante la regla (3.3.10).
6. Incrementar el número de la iteración ($t \rightarrow t + 1$).
7. Si $t < T$ volver al punto 2.

3.6. Algoritmo Gamma Neural Gas

En esta sección se describe el algoritmo Gamma NG, el cual corresponde a la fusión entre el Gas Neuronal (NG) y el Modelo de Contextos Gamma. La topología inicial de la red es un conjunto de N neuronas. La neurona i tiene asociada un vector prototipo d-dimensional, w^i , un conjunto de contextos d-dimensionales c_k^i. Por naturaleza el algoritmo Gas Neuronal no posee espacio de visualización, sin embargo existen algoritmos on-line [59] que pueden ser montados sobre Gamma NG, generando un espacio donde observar la distribución de las neuronas.

El algoritmo Gamma NG se resume a continuación:

Algoritmo 9 Gamma Neural Gas

1. Inicializar aleatoriamente los vectores prototipo, $w^i, c_k^i, \quad \forall i = 1 \dots M, \forall k = 1 \dots K$.
2. Presentar un vector de entrada, $x(t)$, a la red.
3. Calcular el descriptor de contexto c^t usando (3.3.3).
4. Encontrar el BMU, I_t, usando (3.3.2).
5. Adaptar las unidades de la red.
   a) Actualizar los vectores prototipo, w^i , mediante la regla (3.3.9).
   b) Actualizar los vectores de contexto, c^i , mediante la regla (3.3.10).

donde la función de vecindad se ajusta mediante un ranking de la distancia 3.3.1.

1. Incrementar el número de la iteración $(t \rightarrow t + 1)$.
2. Si $t < T$ volver al punto 2.

Debido a que el algoritmo Gas Neuronal logra una mejora cuantización, la construcción del contexto es más cercana a c^{opt} que la que se puede lograr con SOM.

3.7. Algoritmo Gamma Growing Neural Gas

En esta sección se describe el algoritmo Gamma NG, el cual corresponde a la fusión entre el Gas Neuronal Constructivo (GNG) y el Modelo de Contextos Gamma. La topología inicial de la red es un conjunto de 2 neuronas. La neurona i tiene asociada un vector prototipo d-dimensional, w_i y un conjunto de contextos d-dimensionales c_k^i.

El algoritmo Gamma GNG es el siguiente:

Algoritmo 10 Gamma Growing Neural Gas

1. Inicializar aleatoriamente los vectores prototipos, w_1 y w_2 . Inicializar el conjunto de conexiones C, $C = A \times A$, como el conjunto vacío e inicializar el número de iteraciones en $t = 0$
2. Presentar un vector de entrada, $x(t)$, a la red.
3. Encontrar el BMU, I_t, usando (3.3.2) y la segunda unidad más cercana k^* mediante

$$k^* = \underset{i \in A \setminus i^*}{\operatorname{argmin}} d_i(t) \tag{3.7.1}$$

   donde $d_i(t)$ es la función de distancia definida en (3.3.1)
4. Adaptar los vectores prototipo de acuerdo a las reglas (3.3.9) y (3.3.10) .
5. Si no existe una conexión entre j^* y k^* crearla $C = C \cup (j^*, k^*)$. Configurar la edad del borde entre j^* y k^* a cero (refrescar el borde), $age_{(j^*,k^*)} = 0$.
6. Incrementar la edad de todos los bordes que emanan de j^* :

$$age_{(j^*,i)} = age_{(j^*,i)} + 1, \quad \forall i \in N_{j^*} \tag{3.7.2}$$

   donde N_{j^*} es el conjunto de vecinos topológicos directos (existe un borde entre i y j^*) de j^*.
7. Remover aquellos bordes con una edad mayor que la edad máxima, $T(t)$ en la iteración t definida como:

$$T(t) = T_i \left(\frac{T_f}{T_i} \right)^{\frac{t}{t_{max}}} \tag{3.7.3}$$

8. Incrementar el número de la iteración $(t \rightarrow t + 1)$.
9. Si $t < T$ volver al punto 2.

El algoritmo Gamma Growing Neural Gas presenta serias ventajas en lo que respecta al orden del algoritmo $O(n)$ en comparación con Gamma Neural Gas $O(n \log(n))$. No sólo esto sino que debido a que el número de neuronas en las iteraciones iniciales es pequeño, el número de cálculos por iteración es mucho menor, aumentando aún más la velocidad del algoritmo. Debido a la pequeña cantidad de neuronas y a que las nuevas neuronas son inicializadas en la vecindad de neuronas que ya representan la dinámica de la señal (peso y contexto entrenados), no es necesario diferenciar en dos etapas el entrenamiento, permitiendo mantener el valor de α constante de inicio a fin del entrenamiento. Debido a las ventajas en termino de tiempo, del algoritmo Gamma GNG, es altamente recomendable su implementación para abordar bases de datos de un tamaño mayor.

3.8. *ForwardTracking*

El *ForwardTracking*, basado en el *BackTracking* (2.2.5.1), está asociado a la idea de poder reconstruir las secuencias asociadas a las distintas unidades de la red utilizando el contexto de las neuronas. Esta vez la reconstrucción es hacia adelante, esto permite observar la capacidad de la red para generar señales que aún no han sido presentadas, basándose sólo en la información anterior.

La reconstrucción se realiza seleccionando la i-ésima unidad de la red e inicializando las lista $L_i = \{i\}$ que almacena las secuencias reconstruidas. El objetivo ahora es determinar la neurona posterior a la neurona i-ésima. A diferencia del *BackTraking* ahora calcula el *merge* de la i-ésima unidad.

$$m_i = \beta c_i + (1 - \beta) w_i \tag{3.8.1}$$

Sea $i^{(fin)}$ el último elemento de la lista L_i. Luego la unidad siguiente es seleccionada como

$$k = \underset{j \neq i^{(fin)}}{\operatorname{argmin}} \| c_j - m_{i^{(fin)}} \|^2 \tag{3.8.2}$$

luego la lista L_i es actualizada, colocando el elemento k al final de la lista.

$$L_i = L_i \cup \{k\} \tag{3.8.3}$$

Este procedimiento debe realizarse en forma iterativa hasta que el el próximo elemento k a ser ingresado en la lista. El algoritmo se muestra a continuación:

La razón del uso de las ecuaciones (3.8.1) y (3.8.2) se debe a la forma en que se construye el contexto en MNG (2.2.16). Supongamos que la neurona i-ésima es la ganadora en el instante presente. El calcular el *merge* de las neuronas i-ésima es necesario para determinar cual es la ganadora en la etapa siguiente, de esta forma el valor m_i equivale al descriptor del contexto c^t en la etapa presente. El *ForwardTracking* es intuitivamente más natural que el *BackTracking*, pues preserva la estructura del algoritmo original.

El Largo Promedio se calcula para la red como

$$\overline{L} = \frac{1}{N} \sum_{i=1}^{N} L_i \tag{3.8.4}$$

Algoritmo 11 *BackTracking*

1. Seleccionar la i-ésima neurona.

   a) Inicializar $L_i = \{i\}$

   b) Para todas las neuronas calcular (3.8.1)

   c) Determinar k usando (3.8.2)

   d) **if** $k \in L_i$
   ir al punto 2,
   else
   ir al punto $1e$
   end if

   e) Actualizar lista, (3.8.3)

   f) Volver a $1c$

2. Pasar a la siguiente neurona ($i \rightarrow i + 1$)

3.9. Metodología

En esta sección se describe la metodología utilizada para evaluar el comportamiento del modelo de contextos propuesto y las bases de datos utilizadas.

3.9.1. Medidas de Comparación

3.9.1.1. Comparación mediante Planos de Recurrencia (RP)

Los Planos de Recurrencia, son capaces de capturar la dinámica de la secuencia mediante topologías muy locales, estas son representadas en un plano bidimensional como se describe en la sección 2.4.3. La ventaja de realizar una comparación mediante el uso de esta herramienta es la posibilidad de determinar la capacidad de la red para construir una representación interna de la dinámica de la secuencia. De ésta forma es posible realizar una comparación entre dos espacios de estado que viven en espacios de distinta dimensión, pero representan la misma dinámica.

Cuando se tiene acceso al espacio de estado de la secuencia es posible utilizarla para comparar los RPs originales con los RPs generados por la red neuronal. La medida utilizada corresponde a una comparación directa de los RP's. Sea (RP) el RP original y (NRP) el generado por la red, se define

$$(D)_{ij} \begin{cases} 0, & si\,(RP)_{ij} = (NRP)_{ij} \quad \forall i = 1 \ldots I, \forall j = 1 \ldots J \\ 1, & si\,(RP)_{ij} \neq (NRP)_{ij} \quad \forall i = 1 \ldots I, \forall j = 1 \ldots J \end{cases} \tag{3.9.1}$$

donde $(M)_{ij}$ corresponde al elemento i, j de la matriz M.

El error corresponde a la suma de los elementos de la matriz de diferencia D, normalizado por el tamaño de la matriz.

$$E_{RP} = \frac{1}{I \cdot J} \sum_{i,j} (D)_{ij} \tag{3.9.2}$$

Es importante considerar en la construcción de los distintos RPs, que se debe fijar la cantidad de puntos recurrentes (*recurrence rate*) y no el ancho de la vecindad.

3.9.1.2. Comparación Mediante Error de Cuantización Temporal

El error de Cuantización temporal, es usado como mediada de desempeño, sin embargo éste definido sólo para series de tiempo. Los algoritmos de cuantización que no están orientados al procesamiento de secuencias obtienen un pequeño error en el instante $t = 0$, pues el funcional de minimización ajusta los vectores prototipos al ejemplo entrenado en el instante presente, el error aumenta a medida que aumenta el orden del retardo.

Los algoritmos orientados al procesamiento de secuencias sacrifican parte de la cuantización, en $t = 0$, agregando al funcional de minimización información temporal, por lo tanto, si bien el error puede no ser tan pequeño en el retardo de orden cero, este sí mejora a medida que se van considerando secuencias mas largas ($t = 0 \ldots T$), es decir se tienen secuencias similares.

3.9.1.3. Medias sobre el Desempeño de la Red

Cuando se tiene información sobre la señal o la red se entrena para una tarea específica, se puede evaluar el desempeño de la red en dicha tarea. En este trabajo de tesis, se evaluaron dos tareas específicas: Clasificación y Predicción.

La predicción se realiza considerando que la red genera información del espacio de fase, de esta forma ante la presencia de determinismo se puede obtener el punto siguiente como función del punto en el instante presente, esto es

$$x(t+1) = F(x(t)) \tag{3.9.3}$$

donde $F(\cdot)$ es alguna función. La función $F(\cdot)$ puede ser aproximada mediante el uso de redes neuronales de perceptrón multicapa.

En las tareas de clasificación las neuronas son etiquetadas según las etiquetas de los datos reales; para cada neurona, se realiza un histograma en las categorías, para determinar finalmente la etiqueta de la neurona. La comparación final se realiza mediante el uso de matriz de confusiones.

3.9.2. Conjuntos de Entrada

Los conjuntos de entrada son procesados por el algoritmo mediante archivos en codificación ASCII. Una secuencia temporal es un conjunto de vectores $x_t\, t = 1 \dots T$ que poseen una relación de orden en la variable temporal t, cuya dimensión espacial es $\mathbb{R}^d$. Los datos que pueden ser procesados corresponden a conjuntos $s\, s = 1 \dots S$ de secuencias temporales, de esta forma se tienen vectores $x_t^s\, t = 1 \dots T_s, s = 1 \dots S$. En el archivo de entrada, los escalares escritos horizontalmente corresponden a las distintas componentes del vector de d-dimensional; verticalmente, hacia abajo, se escribe la secuencia, las distintas secuencias son separadas mediante una linea en blanco.

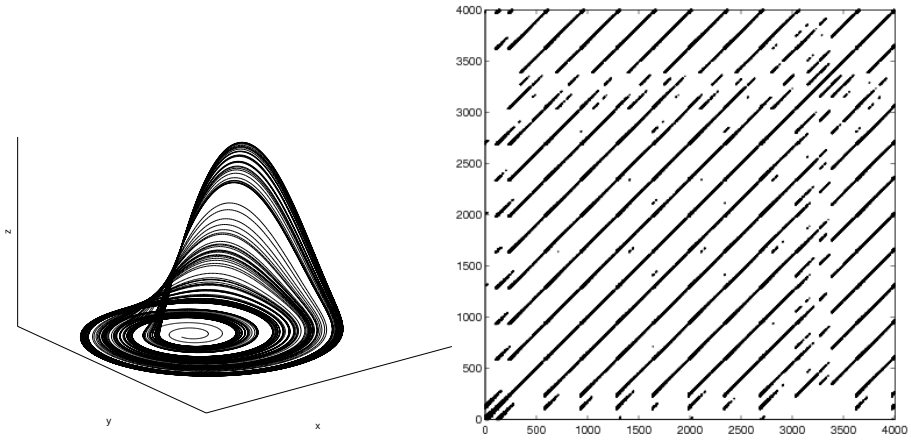
3.9.3. Descripción de las bases de Datos

3.9.3.1. Atractor de Rössler

En 1976 Otto E. Rössler encontró un sistema que es probablemente la construcción geométrica más elemental de caos en sistemas continuos. Este se obtiene como solución del siguiente sistema de ecuaciones diferenciales

$$\begin{aligned} \dot{x} &= -y - z \\ \dot{y} &= x + ay \\ \dot{z} &= b + xz - cz \end{aligned} \tag{3.9.4}$$

donde $(x, y, z) \in \mathbb{R}^3$ corresponden a las variables del espacio de estado y a, b, c son parámetros que determinan la topología del atractor. Para un cierto punto dado (x_0, y_0, z_0) el sistema define una única trayectoria parametrizada en t que satisface el sistema de ecuaciones (3.9.4) para todo



(a) (b)

Figura 3.9.2: La Figura de la Izquierda muestra el Espacio de estado del Atractor de Rössler, la Figura de la Derecha muestra su correspondiente plano de recurrencia.

instante de tiempo.

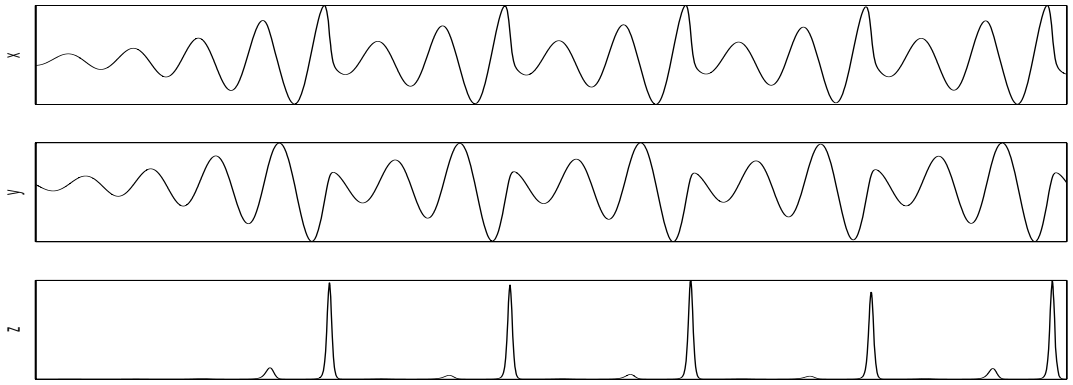
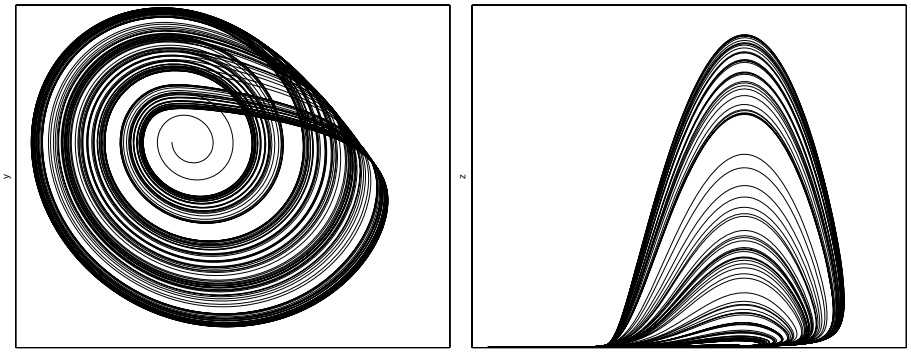


Figura 3.9.1: Solución del sistema (3.9.4). Condición inicial $(x_0, y_0, z_0) = (-1, 0, 0)$, parámetros $(a; b; c) = (0{,}2; 0{,}2; 5{,}7)$.

La Figura 3.9.1, muestra la solución del sistema (3.9.4) en función del tiempo, sin embargo es mucho más interesante observar las trayectorias del atractor en su espacio de estado, Figura 3.9.2.

Las trayectorias del atractor pasan la mayor parte del tiempo en el plano XY, Figura 3.9.3. Cuando las órbitas han alcanzado una distancia crítica al origen, entonces estas pasan a marcar



(a) (b)

Figura 3.9.3: Atractor de Rössler, a) vista del plano XY, b) vista del plano XZ

presencia en la coordenada Z, alejándose del plano XY. Una vez las trayectorias han alcanzado cierta altura, estas vuelven hacia el centro del atractor. Mientras mayor es la altura en el eje Z, más cercana al origen del plano XY es la reinserción de las trayectorias. Dado que el atractor de Rössler es un atractor caótico las trayectorias nunca se sobreponen, y si se deja que el tiempo transcurra libremente, cada trayectoria puede llegar a tener una trayectoria vecina infinitesimalmente cercana.

3.9.3.2. Atractor de Lorenz

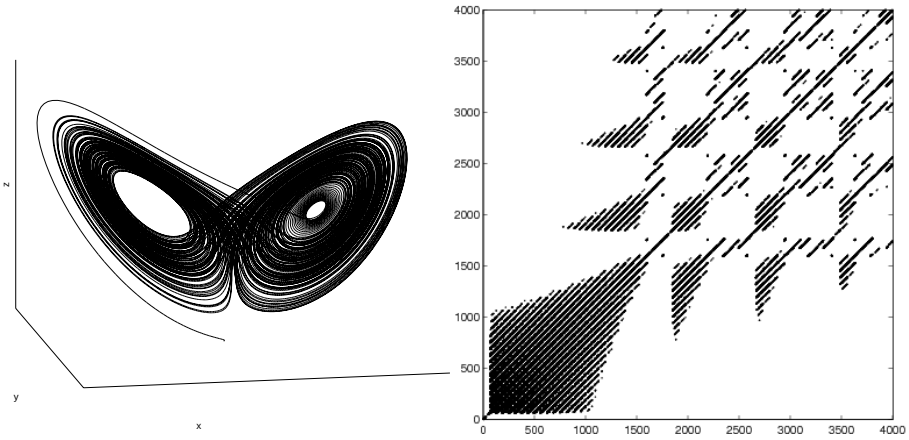
El atractor de Rössler es un sistema artificial diseñado con el propósito de crear un atractor caótico mediante un mecanismo simple. De hecho el atractor de Rössler es un modelo del atractor de Lorenz. El atractor de Lorenz es un modelo físico de convección térmica.

$$\begin{aligned} \dot{x} &= -\sigma x + \sigma z \\ \dot{y} &= Rx - y - xy \\ \dot{z} &= -Bz + xy \end{aligned} \tag{3.9.5}$$

donde $(x, y, z) \in \mathbb{R}^3$ corresponden a las variables del espacio de estado y σ, B, R son parámetros físicos del sistema que Lorenz fijó en

$$\sigma = 10, \quad B = \frac{8}{3}, \quad R = 28 \tag{3.9.6}$$

Para un cierto punto dado (x_0, y_0, z_0) el sistema define una única trayectoria parametrizada en



(a) (b)

Figura 3.9.5: La Figura de la Izquierda muestra el Espacio de estado del Atractor de Lorenz, la Figura de la Derecha muestra su correspondiente plano de recurrencia.

t que satisface el sistema de ecuaciones (3.9.5) para todo instante de tiempo.

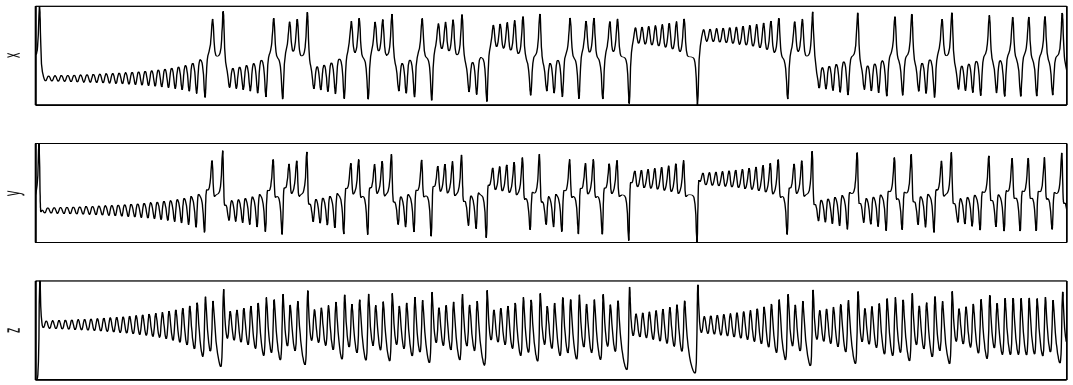
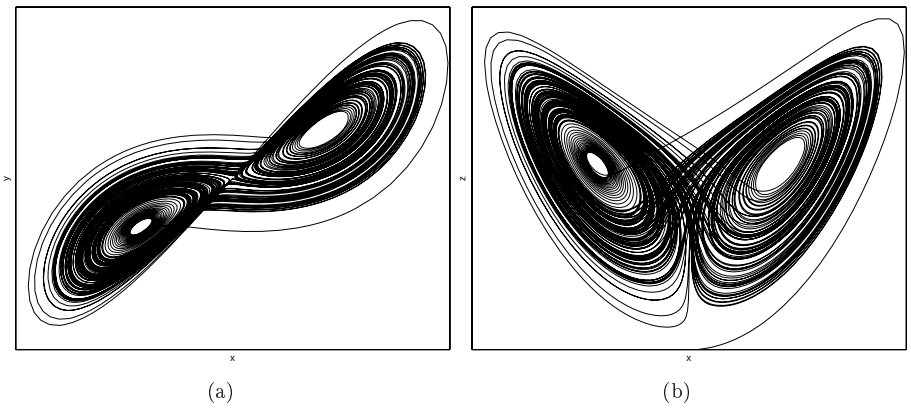


Figura 3.9.4: Solución del sistema (3.9.5). Condición inicial $(x_0, y_0, z_0) = (-1, 0, 0)$.

La Figura 3.9.4, muestra la solución del sistema (3.9.5) en función del tiempo, las trayectorias en el espacio de estado se pueden observar en la Figura 3.9.5.

A diferencia del atractor de Rössler, las trayectorias del atractor pasan se mueven tanto en el plano XY como en el plano XZ, Figura 3.9.6. Se pueden observar claramente dos espirales.



(a) (b)

Figura 3.9.6: Atractor de Lorenz, a) vista del plano XY, b) vista del plano XZ

Cuando las órbitas han alcanzado una distancia crítica al origen de una de las espirales, entonces las trayectorias son atraídas al origen de la otra espiral y comienzan nuevamente a alejarse del centro de esta. Una vez que las trayectorias han alcanzado cierta distancia crítica, estas nuevamente pasan de una a otra espiral. Mientras mayor sea la distancia crítica desde el centro de una de las espirales, la nueva órbita será más cercana al origen de la otra espiral. Al igual que el atractor de Rössler, las trayectorias nunca se sobreponen, y si se deja que el tiempo transcurra libremente, cada trayectoria puede llegar a tener una trayectoria vecina infinitesimalmente cercana.

3.9.3.3. Serie de Tiempo Mackey-Glass

La serie de Mackey-Glass corresponde a un sistema dinámico que ha utilizado como base de datos benchmark para una gran cantidad de algoritmos orientados al procesamiento de secuencias. Esta serie fue usada en [60] y está definida por la siguiente ecuación diferencial:

$$\frac{dx}{d\tau} = bx(\tau) + \frac{ax(\tau - d)}{1 + x(\tau - d)^{10}} \tag{3.9.7}$$

Para d>16.8, la serie se vuelve caótica y por lo tanto es sensible a las condiciones iniciales. La serie utilizada en esta tesis se muestra en la Figura 3.9.7, donde $d = 17$,$a = 0{,}2$, $b = -0{,}1$.

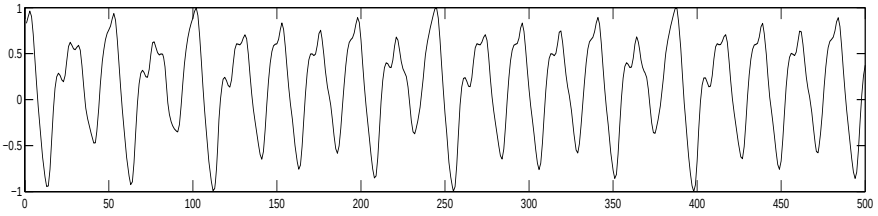


Figura 3.9.7: Serie de tiempo Mackey-Glass, $d = 17$,$a = 0{,}2$, $b = -0{,}1$.

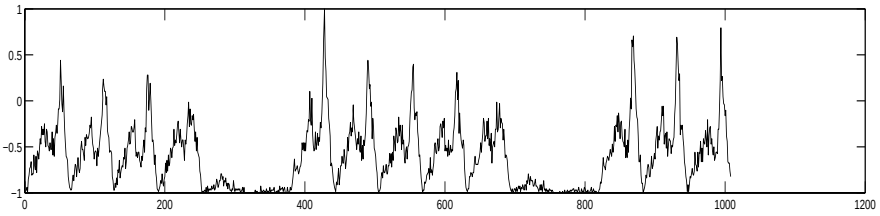


Figura 3.9.8: Serie de Tiempo Bicup 2006

3.9.3.4. Bicup 2006

La base de datos Bicup 2006[1] corresponde a una serie de tiempo que mide la cantidad de pasajeros que ingresan a un servicio de transporte urbano, en la ciudad de Santiago de Chile. Cada pasajero que cruza el torniquete en la entrada de la estación del metro marca un evento, cada instante de tiempo de la serie x_t corresponde a la cantidad de pasajeros que ingresan al servicio sobre un intervalo de 15 minutos, cada día está compuesto por 63 muestras. Esta base de datos fue utilizada para predicción en la competencia Bicup de series de tiempo realizada el año 2006.

En la Figura 3.9.8 se muestra la serie Bicup 2006, los 4 primeros periodos corresponden a los días martes, miércoles, jueves y viernes. Luego se puede observar un claro cambio en la distribución de los datos, que corresponden al fin de semana, sábado y domingo; las muestras posteriores corresponden a los días siguientes. En total se tiene $x_t, t = 1 \ldots 1008$, lo que equivale a 16 días.

3.9.3.5. Bicup 12d

De la base original Bicup 2006 se han extraído los fin de semana, ya que estos presentan una distribución totalmente distinta a los días hábiles (Lunes a Viernes). Al dejar solamente los días

[1] Base disponible en http://www-personal.buseco.monash.edu.au/~hyndman/TSDL/data/bicup2006.dat

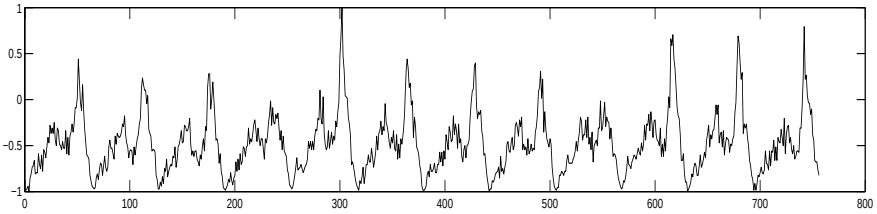


Figura 3.9.9: Serie de Tiempo Bicup 12d.

hábiles la serie es más periódica, facilitando el problema de cuantización temporal. La serie se muestra en la Figura 3.9.9. En total se tiene $x_t, t = 1 \ldots 756$, lo que equivale a 12 días.

3.9.3.6. Fibrilación Ventricular , PhysioBank

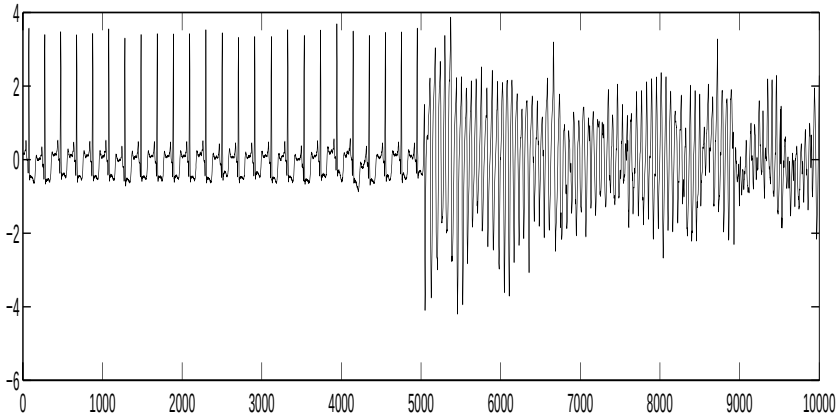
PhysioBank [61] posee cerca de 7000 registros anotados y digitalizados de señales fisiológicas y series de tiempo, organizadas en cerca de 40 bases de datos. Todas están disponibles de sin restricción alguna desde PhysioNet[2]. Dentro de las 40 bases de datos existe las base de datos de *Ventricular Tachyarrhythmia* de la Universidad de Creighton. La base de datos incluye 35 registros, cada uno de una duración de 8 minutos, de ECG de pacientes que experimentaron episodios de taquicardia ventricular, y fibrilación ventricular.

El registro cu01 fue obtenido de un registro de preexistente ECG, reproducido en tiempo real para su digitalización; los otros registros fueron digitalizados en tiempo real desde los pacientes mediante un sistema de monitoreo análogo de alto nivel (1 V/mV nominal). Todas las señales fueron pasadas por un filtro pasa bajos activo Bessel de segundo orden, con una frecuencia de corte de 70 Hz, y fueron digitalizadas a 250 Hz con 12-bits de resolución sobre un rango de 10 V (10 mV nominales relativo a las señales no amplificadas). Cada registro contiene 127.232 muestras (poco menos de 8,5 minutos)

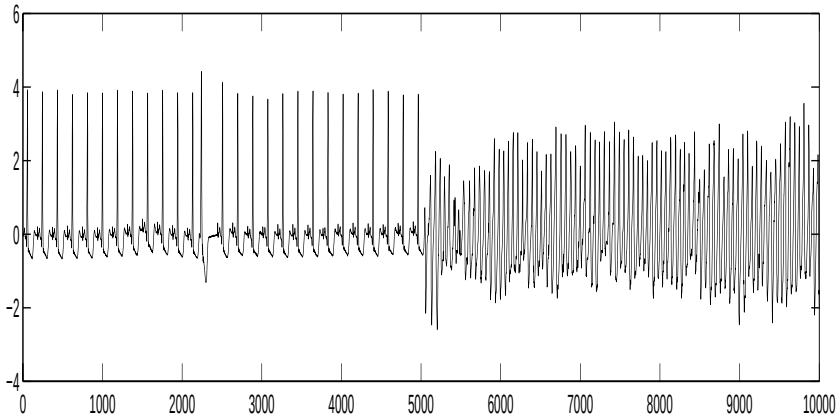
Los episodios de fallo cardiaco, fibrilación, son en la mayoría de los casos precedidos por una taquicardia ventricular, la cual eventualmente termina en la fibrilación ventricular misma.

De los 35 registros que presenta la base, se seleccionaron los registros cu23 y cu24 debido a la similitud que presentan entre ellos, lo que permitiría probar generalización. La Figura 3.9.10, muestra 10.000 puntos tomados de los registros cu23 y cu24. Del registro cu23 se seleccionaron

[2]http://www.physionet.org/



(a)



(b)

Figura 3.9.10: Registros de Fibrilación Ventricular. a) cu23, b) cu24.

10.000 muestras comprendidas entre [78.589 - 88.588], dando origen a cu23-train; del registro cu24 se seleccionaron 10.000 muestras comprendidas entre [84.146 - 94.145], dando origen a cu24-val. En ambos segmentos las primeras 5.000 muestras corresponden a pulso cardiaco normal, las otras 5.000 registran fibrilación ventricular.

Capítulo 4

Resultados

Las dos primeras bases de datos, Rössler y Lorenz, muestran como al aumentar el número de contextos mejora la representación de espacio de estado de la cuantización. Para medir esta mejora se utiliza la medida propuesta basada en la comparación de los planos de recurrencia RP, la cual se utiliza también en el resto de las bases de datos, no sólo debido a la poderosa herramienta que el análisis de espacio de estado ha demostrado ser en la resolución de problemas dinámicos no lineales, sino que también debido al marco teórico basado en espacio de estado que se le ha dado al algoritmo propuesto durante la tesis.

La base de datos de Mackey-Glass, la cual ya ha sido utilizada por otros algoritmos de cuantización temporal [26, 28], se utiliza para mostrar como el algoritmo propuesto mejora el desempeño de otros algoritmos.

Las dos últimas bases de datos, bicup y fibrilación ventricular, se han utilizado para verificar el desempeño del modelo de contextos propuesto en bases de datos de la vida real.

4.1. Atractor de Rössler

4.1.1. Cuantización

El algoritmo Gamma NG fue utilizado para cuantizar el atractor de Rössler. Dado que se quiere evaluar la capacidad de la red neuronal para generar una representación de espacio de estado del atractor, la cuantización se realizó sobre una proyección del espacio de estado en $\mathbb{R}^3$ al eje real $\mathbb{R}$,

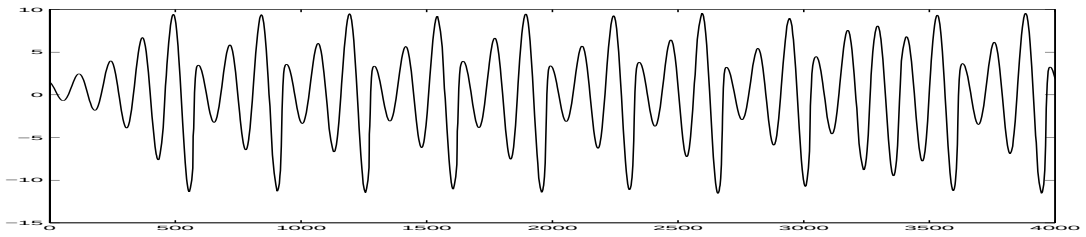


Figura 4.1.1: Proyección unidimensional del espacio de estado del atractor de Rössler.

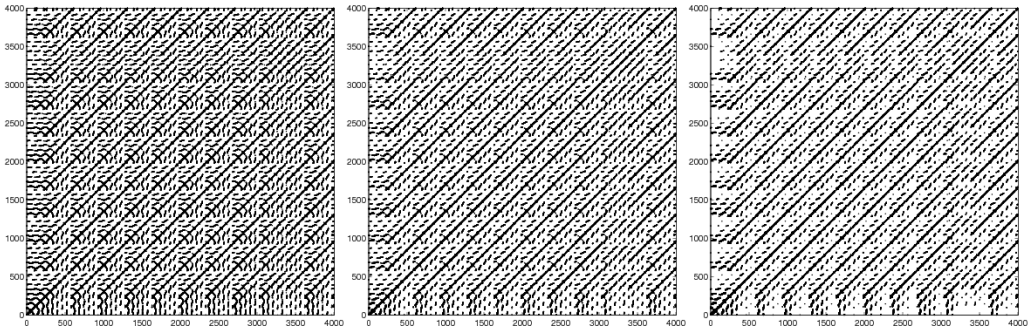
	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9
error	0.0189	0.0160	0.0143	0.0135	0.0131	0.0130	**0.0122**	0.0125	0.0128

Tabla 4.1.1: Error entre la representación generada por la red y el RP real.

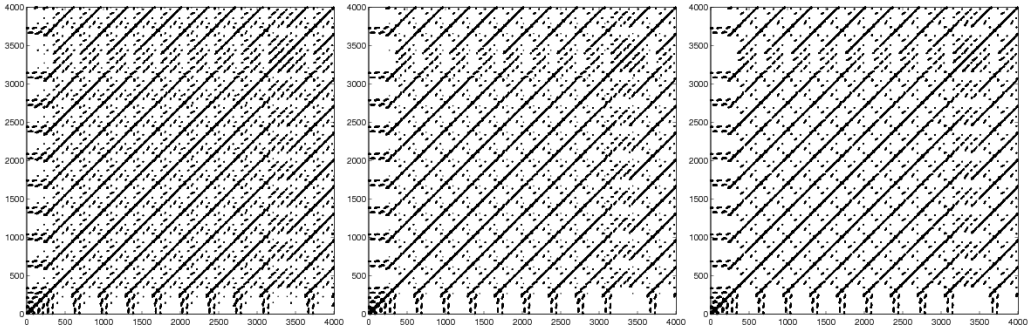
mediante el uso de PCA. La señal utilizada se muestra en la Figura 4.1.1. Se utilizó una cantidad de neuronas equivalente al 15 % del largo de la serie, M=750. El valor de β se fijó en 0,5, el número de contextos K utilizados se varió entre 1 y 9. Se seleccionó $\alpha_w = \frac{K+1}{(K+1)(K+2)/2}, \alpha_k = \frac{K-k+1}{(K+1)(K+2)/2} \forall k = 1 \ldots K$, lo que corresponde a una recta descendiente que cumple $\sum \alpha_w = 1$; los valores de α_w son fijados al inicio del entrenamiento y se mantienen constantes a lo largo de este. La tasa de aprendizaje se varió entre los valores iniciales y finales $(\epsilon_i; \epsilon_f) = (0{,}3; 0{,}001)$ y el ancho de la vecindad se ajustó en $(\lambda_i; \lambda_f) = (\frac{M}{10}; 0{,}01)$.

Para evaluar el desempeño de Gamma NG se compararon las distintas representaciones de espacio de estado, obtenidas variando el número de contextos K, mediante Planos de Recurrencia (RP), tal como se ilustra en la Figura 4.1.2. Para generar cada plano de recurrencia se fijó en la ecuación (2.4.7) $\varepsilon_j = \varepsilon$, de manera tal que la densidad de puntos recurrentes en el RP sea $0,025$, según los criterios mencionados en la sección 2.4.3.

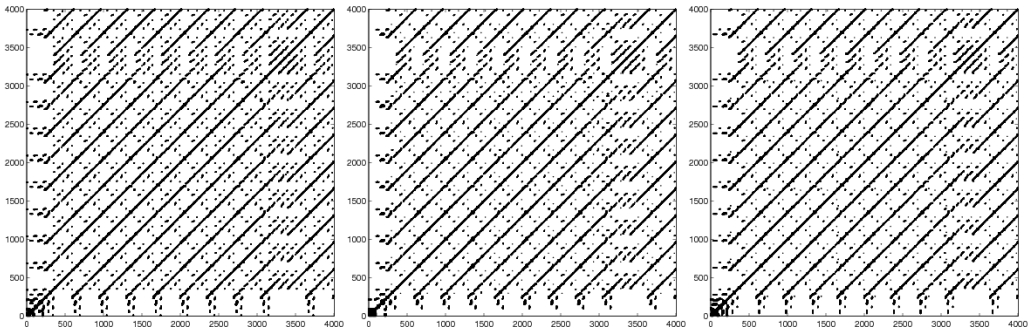
Para comparar las distintas representaciones se calculó el número de puntos que difieren entre la representación generada por la red y el RP real que se muestran en la Figura 4.1.3, y se normalizó por T^2, donde T es el número total de vectores de estado ($T = 4000$ en este caso particular). Esta medida de error se presenta en la Tabla 4.1.1 para distintos valores de K.



(a) K=1 (b) K=2 (c) K=3



(d) K=4 (e) K=5 (f) K=6



(g) K=7 (h) K=8 (i) K=9

Figura 4.1.2: Plano de Recurrencia de la cuantización del atractor de Rössler 4.1.1.

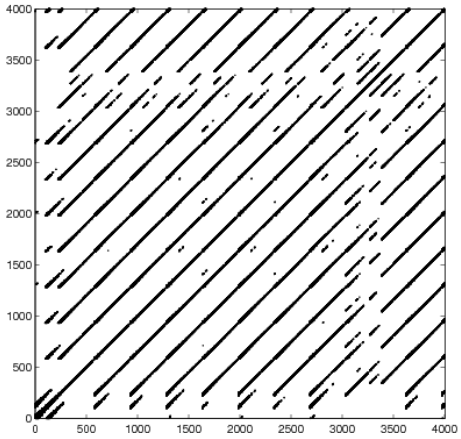


Figura 4.1.3: Plano de recurrencia real del atractor de Rössler.

4.2. Atractor de Lorenz

4.2.1. Cuantización

El algoritmo Gamma NG fue utilizado para cuantizar el atractor de Lorenz. Dado que se desea evaluar la capacidad de la red neuronal para generar una representación de espacio de estado del atractor, la cuantización se realizó sobre una proyección del espacio de estado en $\mathbb{R}^3$ al eje real $\mathbb{R}$, mediante el uso de PCA. La señal utilizada se muestra en la Figura 4.2.1. Se utilizó una cantidad de neuronas equivalente al 15 % del largo de la serie, esto es M=750; una menor cantidad de neuronas no logra capturar la dinámica de la serie por falta de detalle debido a un exceso de cuantización, una cantidad de vectores mayor, si bien logra una pequeña mejora en la representación, aumenta en exceso el tiempo de la simulación. El valor de β se fijo en 0.5, que representa un buen compromiso entre resolución y profundidad; el número de contextos K utilizados se varió entre 1 y 9. Se seleccionó $\alpha_w = \frac{K+1}{(K+1)(K+2)/2}, \alpha_k = \frac{K-k+1}{(K+1)(K+2)/2} \forall k = 1 \dots K$ los cuales son fijados al inicio del entrenamiento y se mantienen constantes a lo largo de este. La tasa de aprendizaje se varió entre dos valores iniciales y finales $(\epsilon_i; \epsilon_f) = (0{,}3; 0{,}001)$ y el ancho de la vecindad se ajustó en $(\lambda_i; \lambda_f) = (\frac{M}{10}; 0{,}01)$.

Para evaluar el desempeño de Gamma NG se comparan las distintas representaciones de espacio de estado, obtenidas variando el número de contextos K, mediante Planos de Recurrencia (RP), tal como se muestra en la Figura 4.2.2. Para generar cada plano de recurrencia se fijo en la ecuación (2.4.7) $\varepsilon_j = \varepsilon$, de manera tal que la densidad de puntos recurrentes en el RP sea 0,025.

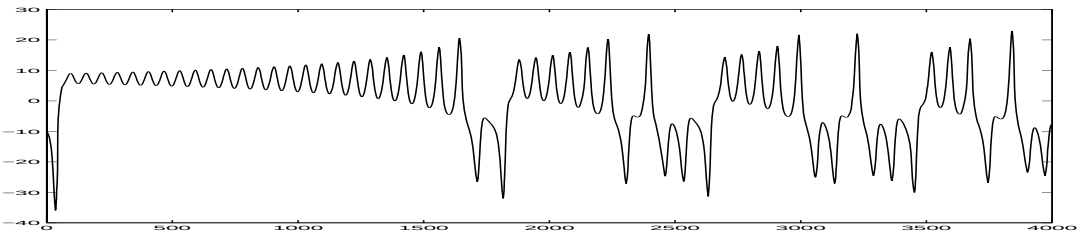


Figura 4.2.1: Proyección unidimensional del espacio de estado del atractor de Lorenz.

	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9
error	0.0199	0.0170	0.0161	0.0142	0.0149	0.0144	0.0145	0.0154	**0.0138**

Tabla 4.2.1: Error entre la representación generada por la red y el RP real.

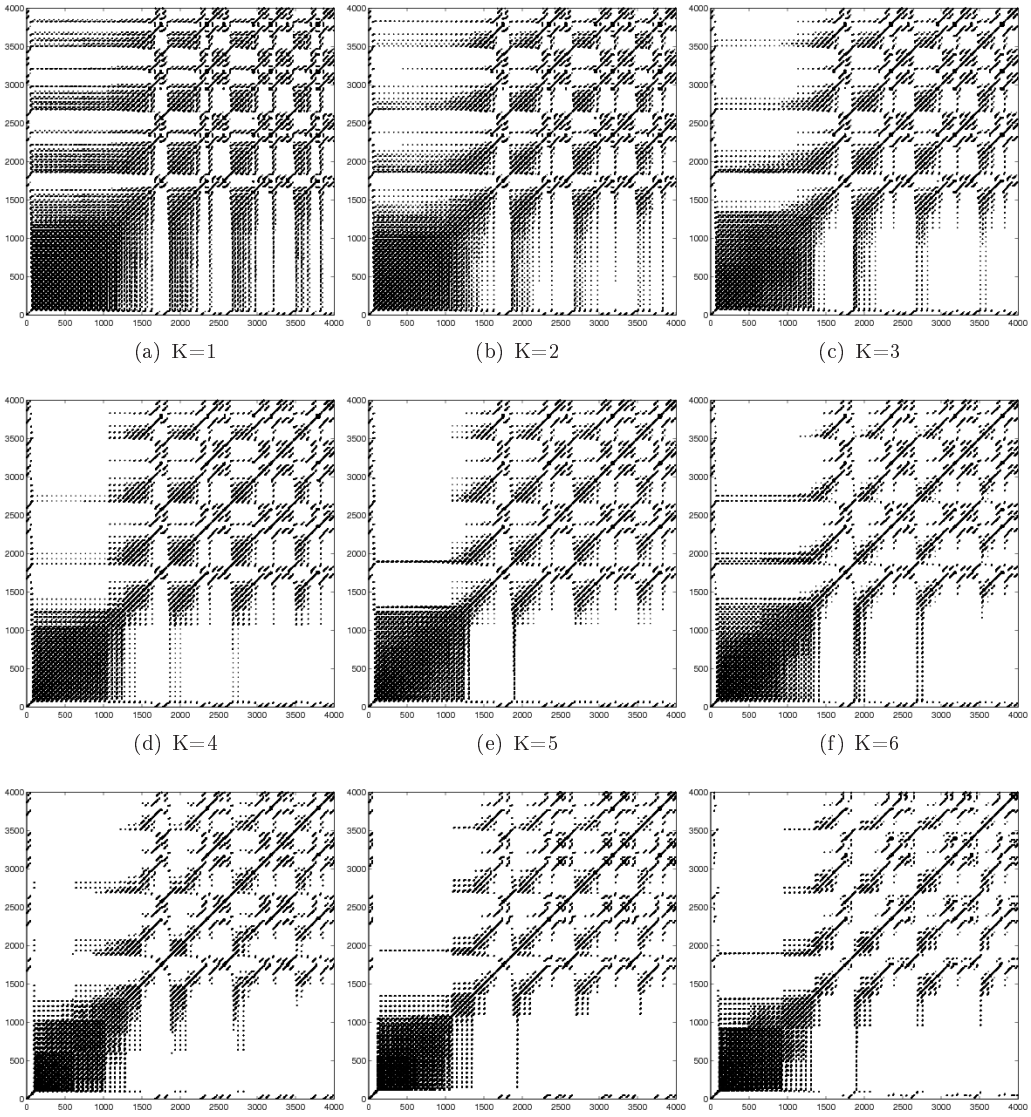
Para comparar las distintas representaciones se calculó el número de puntos que difieren entre la representación generada por la red y el RP real y se normalizó por T^2, donde T es el número total de vectores de estado. Estas medidas de error se presentan en la Tabla 4.2.1.

4.3. Serie de Tiempo Mackey-Glass

4.3.1. Cuantización

El algoritmo Gamma NG fue utilizado para cuantizar la serie de tiempo Mackey-Glass. Se utilizó una cantidad de neuronas equivalente al 15 % del largo de la serie, esto es M=75. El valor de β se fijó en 0.5, el número de contextos K utilizados se varió entre 1 y 9. Se seleccionó $\alpha_w = \frac{K+1}{(K+1)(K+2)/2}, \alpha_k = \frac{K-k+1}{(K+1)(K+2)/2} \forall k = 1 \ldots K$ los cuales son fijados al inicio del entrenamiento y se mantienen constantes a lo largo de este. La tasa de aprendizaje se varió entre los valores iniciales y finales $(\epsilon_i; \epsilon_f) = (0{,}3; 0{,}001)$ y el ancho de la vecindad se ajustó en $(\lambda_i; \lambda_f) = (\frac{M}{10}; 0{,}01)$.

Como medida de desempeño se utilizó el Error de Cuantización Temporal (TQE). La Figura 4.3.1 muestra el error de cuantización temporal variando el número de contextos (taps) y para

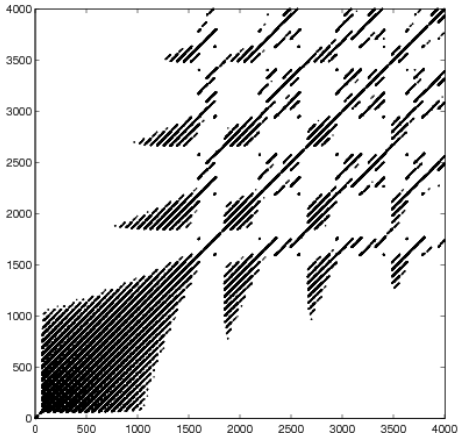


(a) K=1 (b) K=2 (c) K=3

(d) K=4 (e) K=5 (f) K=6

(g) K=7 (h) K=8 (i) K=9

Figura 4.2.2: Plano de Recurrencia de la cuantización del atractor de Lorenz 4.2.1.



(a)

Figura 4.2.3: Plano de recurrencia real del atractor de Lorenz.

	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9
error	8.1154	7.4537	6.4868	6.4184	**6.3788**	7.0424	6.7878	6.9150	7.1083

Tabla 4.3.1: Integración temporal del error de cuantización.

una ventana de ancho 50. Utilizar sólo un contexto equivale al algoritmo MNG. Se puede observar que el mayor TQE se obtiene al utilizar sólo un contexto, se observa una disminución del TQE al aumentar K, la curva de menor error, medida como la suma de los errores temporales se tiene para $K = 5$, según muestra la Tabla 4.3.1.

4.3.2. Predicción

La predicción es realizada a un paso, mediante el uso de una red MLP de 5 unidades ocultas y una unidad en la capa de salida, el entrenamiento se realizó durante 500 épocas y el conjunto de datos se dividió en un 60 % para entrenamiento, 20 % validación y 20 % test. La entrada consiste en el peso y el contexto de la unidad ganadora en el instante t, $(w^{I_t}, c_1^{I_t}, \ldots, c_K^{I_t})$, la salida a entrenar corresponde al valor x_{t+1}. Para cada cuantización, se corrieron 10 simulaciones en la red MLP, por lo que la varianza en la predicción se debe a la red Perceptrón Multicapa y no al algoritmo Gamma NG. El motivo de realizar más de una simulación es evitar que debido a un mal entrenamiento de

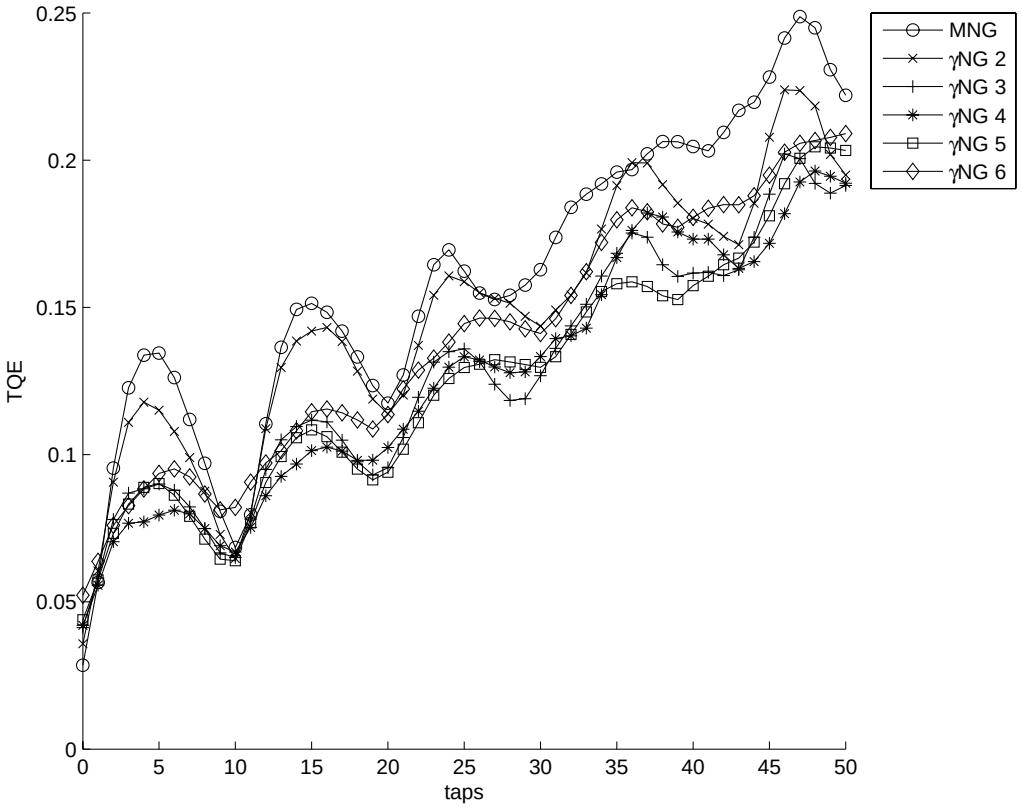


Figura 4.3.1: Error de Cuantización Temporal, para la serie de tiempo Mackey-Glass. Ventana de tamaño 50.

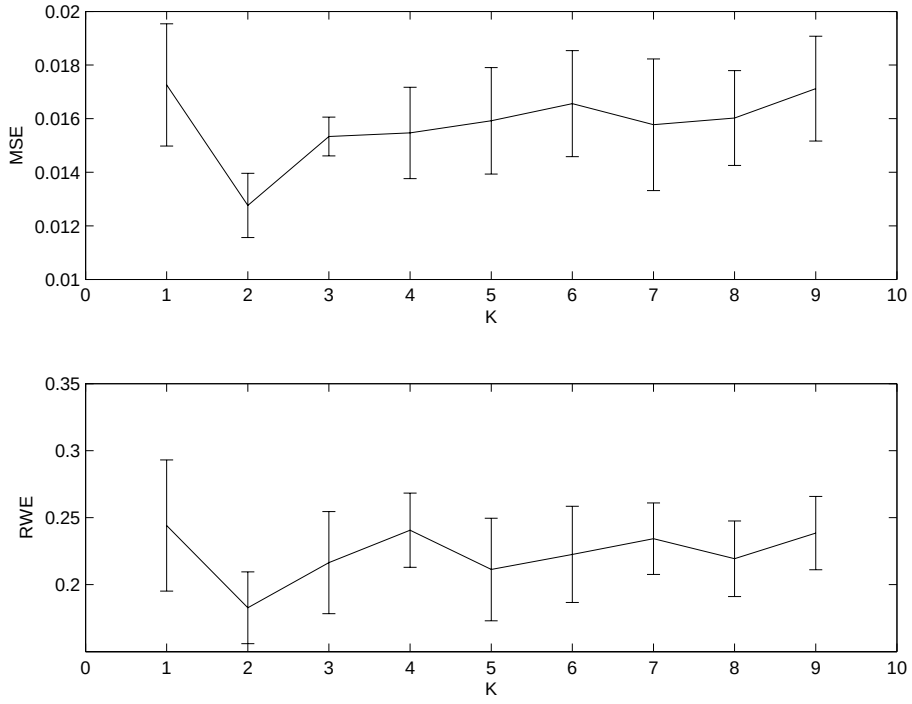


Figura 4.3.2: Media y Varianza del error de Predicción para distintos valores de K. La figura superior muestra el MSE, la figura inferior muestra el "Random Walk Error" (RWE).

la red MLP se sesgue la capacidad real de la red Gamma NG. La Figura 4.3.2 muestra la media y la varianza del error obtenida de las 10 simulaciones, para distintos valores de K.

4.3.3. Cuantización de Espacio de Estado

Al proyectar mediante Análisis de Componentes Principales (PCA) la cuantización obtenida por la red Gamma NG se puede visualizar una aproximación (transformación) del espacio de estado en el plano bi-dimensional. En las Figuras 4.3.3 y 4.3.4, se muestran las neuronas que se activan y las conexiones entre ellas cuando se presenta la señal. Se puede observar que a medida que aumenta el número de contextos las trayectorias se van haciendo cada vez más suaves, facilitando así la detección de patrones similares en la serie dentro de los periodos de la red. La red MNG equivalente a Gamma NG con $K = 1$, en la Figura 4.3.3 no se observan vecindades tan marcadas como en la Figura 4.3.4.

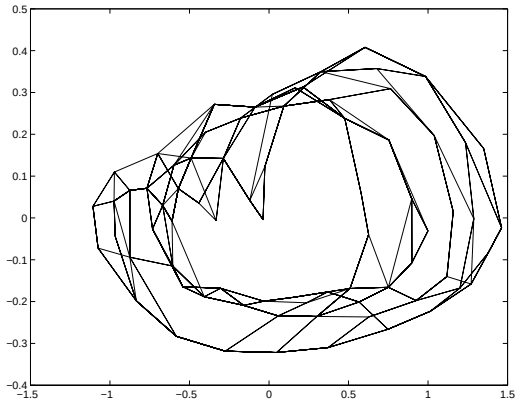
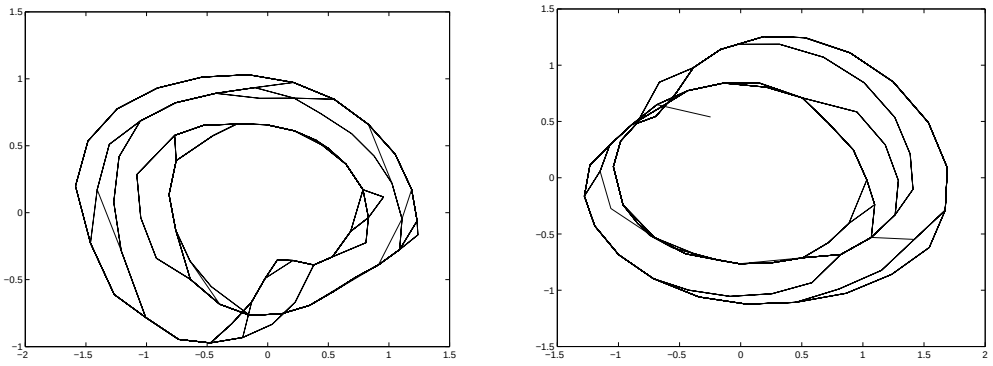


Figura 4.3.3: Proyección bidimensional de las trayectorias en la red MNG al presentar la señal Mackey-Glass



(a) (b)

Figura 4.3.4: Proyección bidimensional de las trayectorias en la red Gamma NG al presentar la señal Mackey-Glass. La Figura a) muestra una cuantización realizada con K=4, mientras que la Figura b) muestra una cuantización con K=8.

4.4. Bicup 2006

4.4.1. Cuantización

El algoritmo Gamma NG fue utilizado para cuantizar la serie de tiempo Bicup 2006. Se utilizó una cantidad de neuronas equivalente al 15 % del largo de la serie, esto equivale a M=152. El valor de β se fijó en 0,5, el número de contextos K utilizados se varió entre 1 y 9. Se seleccionó $\alpha_w =$

	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9
error	9.6336	7.7492	7.3855	6.7017	6.3881	5.8169	5.8567	5.6011	**5.1525**

Tabla 4.4.1: Integración temporal del error de cuantización.

$\frac{K+1}{(K+1)(K+2)/2}, \alpha_k = \frac{K-k+1}{(K+1)(K+2)/2} \forall k = 1 \ldots K$ los cuales son fijados al inicio del entrenamiento y se mantienen constantes a lo largo de este. La tasa de aprendizaje se varió entre $(\epsilon_i; \epsilon_f) = (0{,}3; 0{,}001)$ y el ancho de la vecindad se ajustó en $(\lambda_i; \lambda_f) = (\frac{M}{10}; 0{,}01)$.

Como medida de desempeño se utilizó el Error de Cuantización Temporal (TQE). La Figura 4.4.1 muestra el error de cuantización temporal al variar el número de contextos, para una ventana de ancho 50. Utilizar sólo un contexto equivale al algoritmo MNG. Se puede observar que el mayor TQE se obtiene al utilizar sólo un contexto, al aumentar el valor de K se observa una disminución del TQE, obteniéndose el menor error con $K = 9$, como se puede observar en la Tabla 4.4.1.

4.4.2. Predicción

La información temporal de la serie es almacenada en el contexto de las distintas neuronas de la red. Aumentar el número de contextos permite mejorar la capacidad de almacenamiento de la red, aumentando la profundidad de la memoria sin perder la resolución de ésta. Para estudiar la mejora en la capacidad de almacenamiento temporal de las neuronas, se ha utilizado la red en tareas de predicción. La predicción es realizada a un paso, mediante el uso de una red MLP de 5 unidades ocultas y una unidad en la capa de salida. El entrenamiento se realizó durante 500 épocas y el conjunto de datos se dividió en un 60 % para entrenamiento, 20 % validación y 20 % test. La entrada consiste en el peso y el contexto de la unidad ganadora en el instante t, $(w^{I_t}, c_1^{I_t}, \ldots, c_K^{I_t})$, la salida a entrenar corresponde al valor x_{t+1}. Para cada cuantización, se corrieron 10 simulaciones en la red MLP, por lo que la varianza en la predicción se debe a la red Perceptrón Multicapa y no al algoritmo Gamma NG. La Figura 4.4.2 muestra la media y la varianza del error obtenida de las 10 simulaciones, para distintos valores de K.

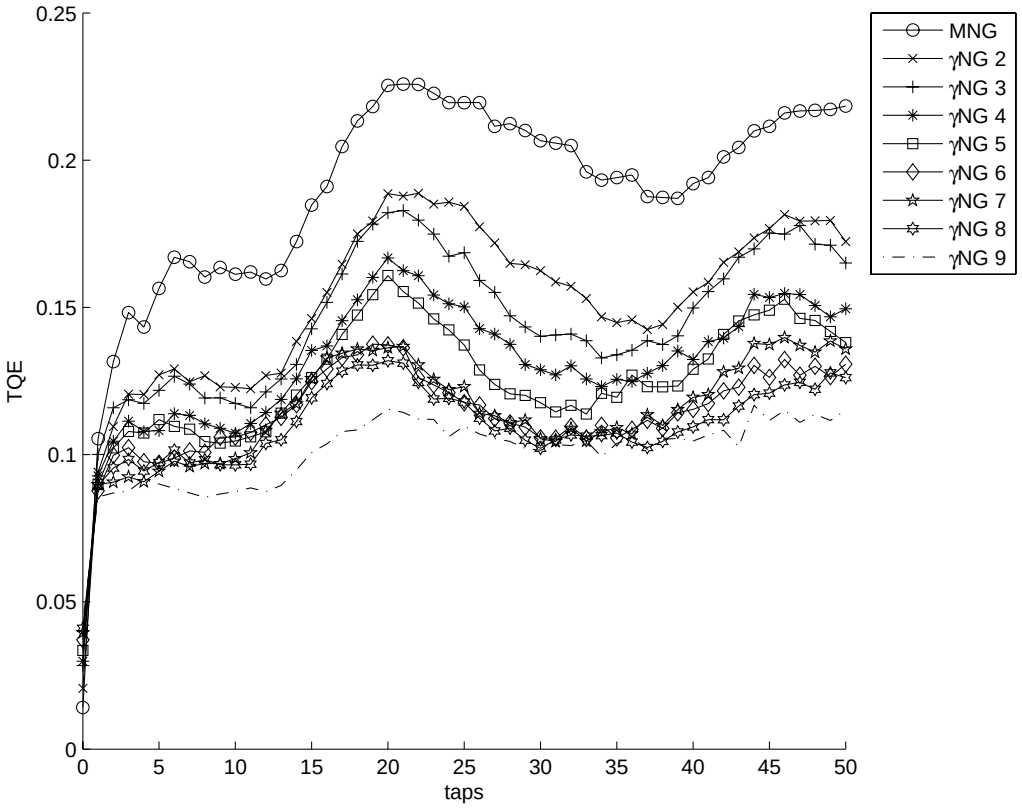


Figura 4.4.1: Error de Cuantización Temporal, para la serie de tiempo Bicup 2006. Ventana de tamaño 50

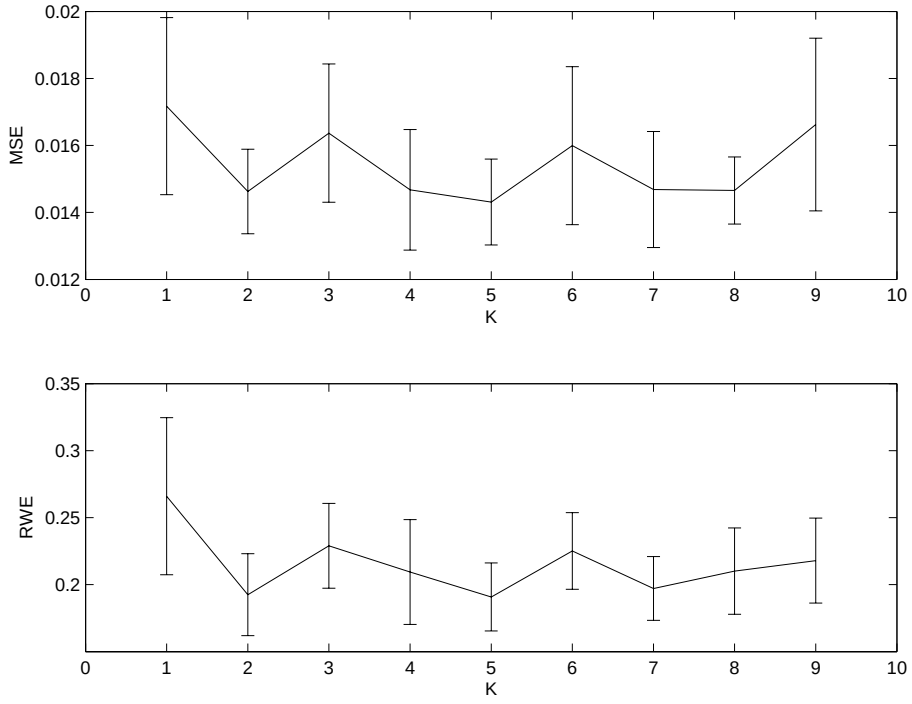


Figura 4.4.2: Error de Predicción para distintos valores de K. La figura superior muestra el MSE, la figura inferior muestra el "Random Walk Error"

4.4.3. Cuantización de Espacio de Estado

Al proyectar mediante Análisis de Componentes Principales la cuantización obtenida por la red Gamma NG se puede visualizar una aproximación (transformación) del espacio de estado en el plano bi-dimensional. En la Figura 4.4.3 se pueden observar las neuronas que se activan y las conexiones entre ellas cuando se presenta la señal. Usando Gamma NG en la Figura 4.4.4 se puede observar que a medida que aumenta el número de contextos las trayectorias se van haciendo cada vez más suaves, facilitando la detección de patrones similares en la serie dentro de los periodos de la red. La red MNG correspondiente a la Figura 4.4.3 muestra una agrupación más desordenada de las neuronas, no se observan patrones tan claros como en las de la Figura 4.4.4, que muestra un aspecto mucho más similar al de un atractor caótico.

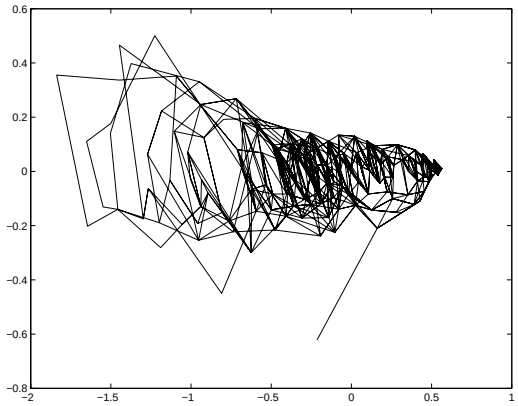
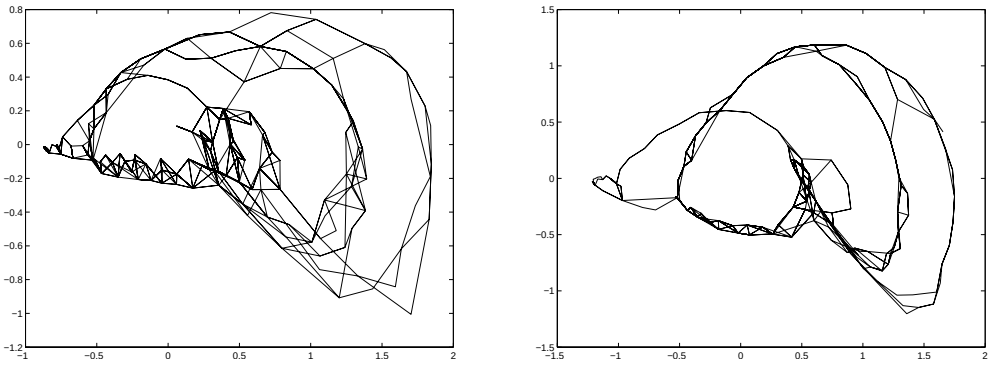


Figura 4.4.3: Proyección bidimensional de las trayectorias en la red MNG al presentar la señal Bicup 2006.



(a) (b)

Figura 4.4.4: Proyección bidimensional de las trayectorias en la red Gamma NG al presentar la señal Bicup 2006. La Figura a) muestra una cuantización realizada con K=4, mientras que la Figura b) muestra una cuantización para K=9.

4.5. Bicup 12d

4.5.1. Cuantización

El algoritmo Gamma NG fue utilizado para cuantizar la serie de tiempo Bicup 12d. Se utilizó una cantidad de neuronas equivalente al 15 % del largo de la serie, esto equivale a M=114. El valor de β se fijo en 0,5, el número de contextos K utilizados se varió entre 1 y 9. Se seleccionó $\alpha_w =$

	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9
error	8.4553	7.5142	6.9791	6.3447	6.3359	5.9941	5.8940	5.7163	**5.6285**

Tabla 4.5.1: Integración temporal del error de cuantización.

$\frac{K+1}{(K+1)(K+2)/2}, \alpha_k = \frac{K-k+1}{(K+1)(K+2)/2} \forall k = 1 \ldots K$ los cuales son fijados al inicio del entrenamiento y se mantienen constantes a lo largo de este. La tasa de aprendizaje se varió entre $(\epsilon_i; \epsilon_f) = (0{,}3; 0{,}001)$ y el ancho de la vecindad se ajustó en $(\lambda_i; \lambda_f) = (\frac{M}{10}; 0{,}01)$.

Como medida de desempeño se utilizó el Error de Cuantización Temporal (TQE), la Figura 4.5.1 muestra el error de cuantización temporal variando el número de contextos y una ventana de ancho 50. El utilizar sólo un contexto equivale al algoritmo MNG. Al igual que en la serie Bicup 2006 se observa que el mayor TQE se obtiene al utilizar sólo un contexto, este error disminuye al aumentar el valor de K, como se puede observar en la Tabla 4.5.1.

4.5.2. Predicción

La predicción es realizada a un paso bajo las mismas condiciones de Bicup 2006, mediante el uso de una red MLP de 5 unidades ocultas y una unidad en la capa de salida. El entrenamiento se realizó durante 500 épocas y el conjunto de datos se dividió en un 60 % para entrenamiento, 20 % validación y 20 % test. La entrada consiste en el peso y el contexto de la unidad ganadora en el instante t, $(w^{I_t}, c_1^{I_t}, \ldots, c_K^{I_t})$, la salida a entrenar corresponde al valor x_{t+1}. Para cada cuantización, se corrieron 10 simulaciones en la red MLP, por lo que la varianza en la predicción se debe a la red Perceptrón Multicapa y no al algoritmo Gamma NG. La Figura 4.5.2 muestra la media y la varianza del error obtenida de las 10 simulaciones, para distintos valores de K.

4.5.3. Cuantización de Espacio de Estado

Al proyectar mediante Análisis de Componentes Principales (PCA) la cuantización obtenida por la red Gamma NG se puede visualizar una aproximación (transformación) del espacio de estado en el plano bi-dimensional. En las Figuras 4.5.3 y 4.5.4, se pueden observar las neuronas que se

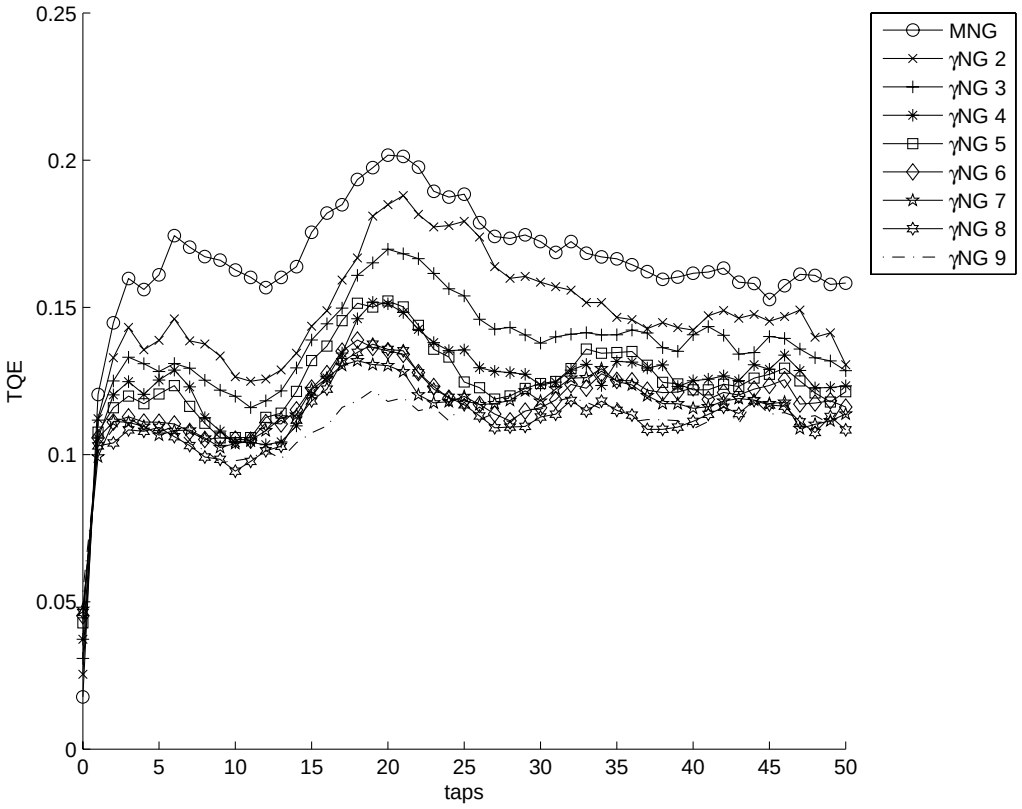


Figura 4.5.1: Error de Cuantización Temporal, para la serie de tiempo Bicup 12d. Ventana de tamaño 50

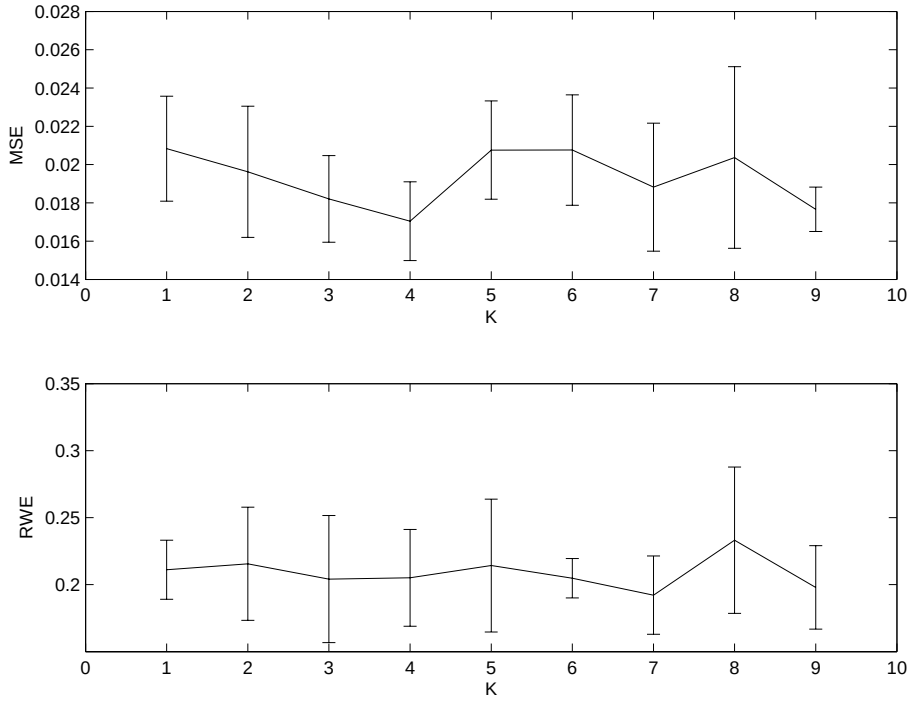


Figura 4.5.2: Error de Predicción para distintos valores de K. La figura superior muestra el MSE, la figura inferior muestra el "Random Walk Error"

activan y las conexiones entre ellas cuando se presenta la señal. Se puede observar que a medida que aumenta el número de contextos las trayectorias se van haciendo cada vez más suaves, facilitando así a la red la detección de patrones similares en la serie dentro de los periodos de la red. Es interesante realizar una comparación entre la Figura 4.5.3 y la Figura 4.4.3, a pesar de que en la primera se han eliminado los fines de semana, la distribución de las neuronas y las conexiones entre ellas han variado muy poco. En cambio las Figuras 4.4.4 y 4.5.4, se observa una diferencia en la proyección, asociada a la capacidad de la red para diferenciar espacialmente los fines de semana.

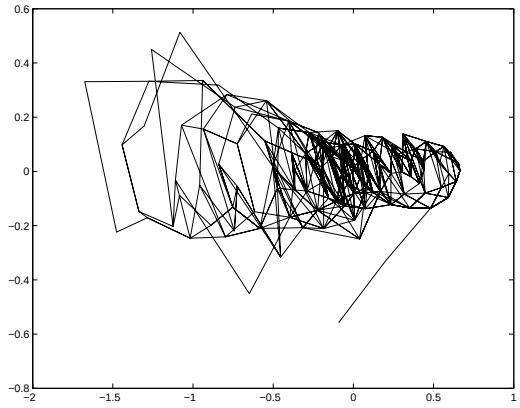
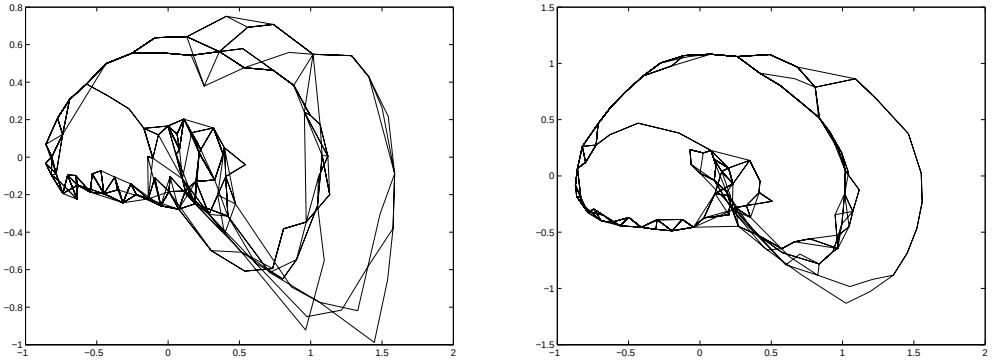


Figura 4.5.3: Proyección bidimensional de las trayectorias en la red MNG al presentar la señal Bicup 12d



(a) (b)

Figura 4.5.4: Proyección bidimensional de las trayectorias en la red Gamma NG al presentar la señal Bicup 12d. La Figura a) muestra una cuantización realizada con K=4, mientras que la Figura b) muestra una cuantización para K=8.

4.6. Fibrilación Ventricular, PhysioBank

4.6.1. Resultados de Clasificación

La red Gamma NG fue usada para tareas de clasificación en la detección de fibrilación ventricular. El registro cu23-train fue utilizado para entrenamiento y cu24-val para validación.

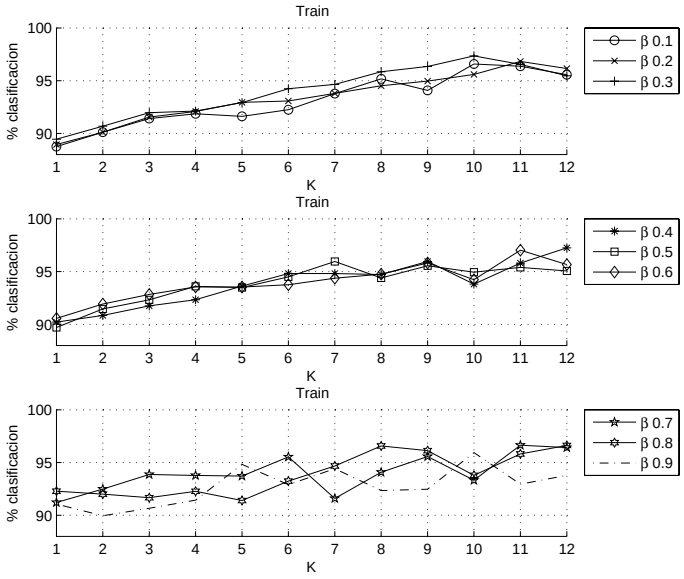


Figura 4.6.1: Clasificación, conjunto de entrenamiento. El número de retardos K se varió entre 1 y 12. Dado que Gamma NG generaliza el modelo de Contexto de MNG, el considerar sólo 1 retardo es equivalente a usar MNG.

En cada simulación se utilizaron 350 neuronas, el valor de β se varió entre 0,1 y 0,9 con un paso de 0,1; el número de contextos utilizados se varió entre 1 y 12. Se seleccionó $\alpha_w = \frac{K+1}{(K+1)(K+2)/2}, \alpha_k = \frac{K-k+1}{(K+1)(K+2)/2} \forall k = 1 \ldots K$ los cuales son fijados al inicio del entrenamiento y se mantienen constantes a lo largo de este. Dado que Gamma NG realiza una generalización del contexto, el considerar sólo un retardo es equivalente a MNG. Las Figuras 4.6.1 y 4.6.2 muestran el porcentaje de acierto logrado en los conjuntos de entrenamiento y validación, respectivamente. Este valor es calculado como

$$Acierto = 100 * \frac{vp + vn}{vp + vn + fp + fn} \tag{4.6.1}$$

donde vp corresponde al número de verdaderos positivos, vn verdaderos negativos, fp falsos positivos y fn falsos negativos.

El mejor valor obtenido de la red Gamma NG en el conjunto de entrenamiento, el cual se obtuvo para $\beta = 0{,}3$ y $K = 10$, para $K = 1$ (MNG), el mejor resultado se obtuvo para $\beta = 0{,}8$. Utilizando

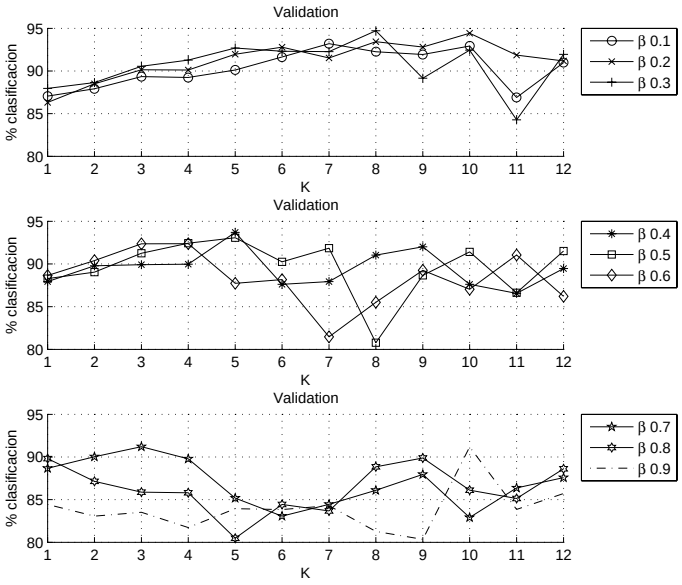


Figura 4.6.2: Desempeño en tareas de clasificación. Las figuras muestran los resultados en el conjunto de validación para distintos valores del parámetro β. El número de retardos se varió entre 1 y 12. Dado que Gamma NG generaliza el modelo de Contexto de MNG, el considerar sólo 1 retardo es equivalente a usar MNG.

% de Acierto	MNG	Gamma NG
Entrenamiento	92,28 %	97,35 %
Validación	89,81 %	92,45 %

Tabla 4.6.1: Porcentaje de Acierto. El mejor resultado obtenido con MNG ($K = 1$) se logró con $\beta = 0{,}8$. Para Gamma NG el mejor resultado se obtuvo con $K = 10, \beta = 0{,}3$.

las redes mencionadas anteriormente se presentó el conjunto de validación, los resultados obtenidos se muestran en la Tabla 4.6.1

Para el filtro Gamma, la resolución de la memoria R esta dada por (2.3.18) y la profundidad D por (2.3.17). La Tabla 4.6.2 muestra la resolución y la profundidad de la memoria en la red para la base de datos PhysioBank.

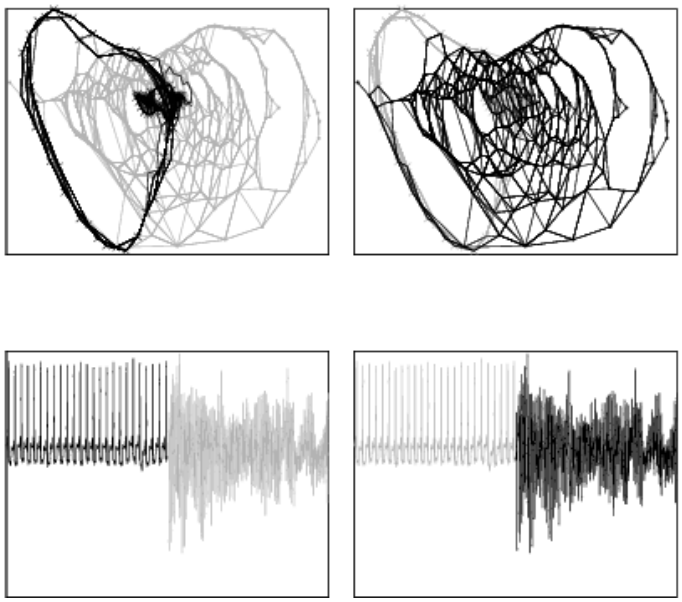
	MNG	Gamma NG
$R = 1 - \beta$	0,2	0,7
$D = \frac{K}{1-\beta}$	5	14,29

Tabla 4.6.2: Profundidad D y Resolución R de la Cuantización para PhysioBank.

4.6.2. Cuantización de Espacio de Estado para PhysioBank

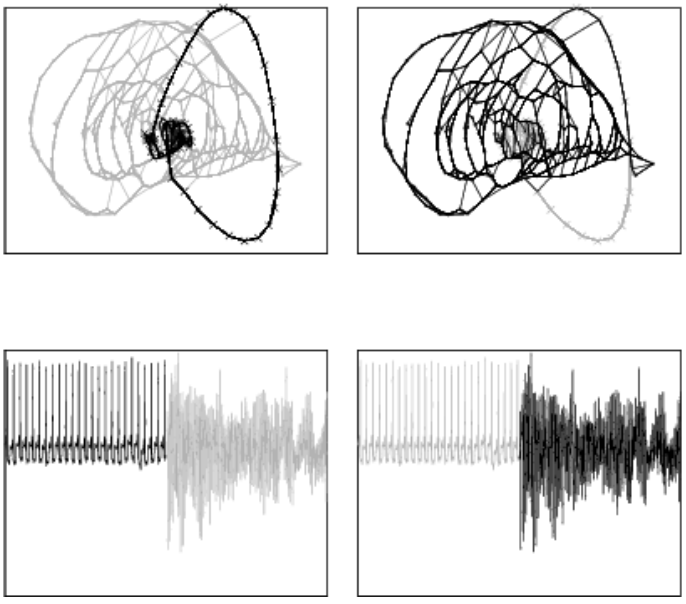
Al proyectar la cuantización obtenida por la red Gamma NG se puede visualizar una aproximación (transformación) del espacio de estado. En las Figuras 4.6.3 y 4.6.4, se pueden observar las neuronas que se activan y las conexiones entre ellas cuando se presentan distintos segmentos de la señal.

Comparando las Figuras 4.6.3 a) y 4.6.4 a) se puede ver que para cuantizar el pulso normal la red Gamma NG necesita de una menor cantidad de neuronas y las trayectorias que sigue la señal presentan una menor varianza. En MNG en cambio (Figura 4.6.3 a)) las trayectorias siguen caminos distintos y por lo mismo la cantidad de neuronas que se necesitan para representar estos caminos debe ser mayor. Esto muestra que las pequeñas alteraciones que hay en el ritmo cardiaco hacen que la señal de pulso normal cambie lo suficiente como para que la red MNG deba ocupar nuevas neuronas para aprenderlas como secuencias distintas. Gamma NG en cambio presenta una mayor robustez, logrando ocupar las mismas neuronas para las secuencias ya aprendidas.



(a) (b)

Figura 4.6.3: Activación de la red MNG ante estímulos de la señal. La Figura a) muestra ritmo cardiaco normal, mientras que la Figura b) muestra la condición anormal



(a) (b)

Figura 4.6.4: Activación de la red Gamma NG ante estímulos de la señal.La Figura a) muestra ritmo cardiaco normal, mientras que la Figura b) muestra la condición anormal

Capítulo 5

Discusión

5.1. Análisis de las Medidas de Desempeño

El Error de Cuantización Temporal (TQE), trata de generalizar el error de cuantización normal, el cual considera solamente la similitud entre el peso de la neurona y el patrón de entrada en el instante presente. Esta modificación no involucra el peso de la neurona ni los posibles contextos que esta posea, por lo tanto no se trata de la generalización natural del Error de Cuantización, sino que de una adaptación temporal que trata de incluir para las distintas señales asociadas a la neurona, la similitud entre las señales en instantes previos. De esta forma se mide cuan parecidas son las señales que ha aprendido la neurona. Esta medida resulta ser independiente de la cantidad de contextos, ya que sólo involucra a la señal. En la presente tesis se ha utilizado TQE para medir el desempeño sobre series de tiempo, sin embargo esta medida se podría generalizar tomando la norma de las varianzas medidas por componentes, en el caso multidimensional. El TQE busca medir la capacidad de la red para aprender secuencias similares.

Los Planos de Recurrencia sirven para medir ciertas características de los sistemas dinámicos de una forma simple mediante el uso de topologías locales. Dado que el modelo desarrollado trata de generar una representación interna de la dinámica de la secuencia, es deseable poder medir qué tanto se parece esta representación a la dinámica real de la secuencia. Cuando se tiene información sobre el espacio de estado de la secuencia se puede comparar esta representación generada por la red con la dinámica original. Es aquí donde los planos de recurrencia, que corresponden a la firma

de la secuencia, permiten que esta comparación sea realizada. La ventaja de utilizar los planos de recurrencia es que nos posibilitan comparar representaciones de distinta dimensión, permitiéndonos comparar el modelo Merge que sólo utiliza un contexto con el modelo Gamma que utiliza una mayor cantidad de contextos. La comparación mediante planos de recurrencia, sección 3.9.1.1, permite medir que la capacidad de la red para representar la dinámica de la secuencia entrenada.

5.2. Análisis del Modelo Propuesto

El modelo de contextos Gamma puede ajustarse a distintos tipos de algoritmos de cuantización, lo anterior se debe a que la construcción del contexto se basa solamente en las características de la unidad ganadora del instante previo, logrando así ser independiente de la topología de la red.

La relación ente el algoritmo propuesto y los filtros IIR es muy estrecha, siendo éstos últimos utilizados como una herramienta indispensable para lograr capturar la dinámica de la señal. En la sección 3.4 se ha mostrado como a partir de la función de transferencia es posible determinar una regla de entrenamiento para el contexto. Esto abre la opción de utilizar un sin fin de diferentes filtros, que poseen distintas propiedades, los cuales pueden ser utilizados para definir una regla de entrenamiento y aprovechar así dichas propiedades.

La utilización de filtros tipo IIR permite un desacoplamiento entre la cantidad de retardos utilizados y la profundidad de la memoria del algoritmo. La importancia principal de este punto radica en que si la dinámica de la serie requiere de una alta profundidad, o una gran cantidad de retardos, para poder lograr capturar dicha dinámica, la dimensión de la red no aumenta de forma indiscriminada, sino que mediante el control de distintos parámetros es posible ajustar un configuración de menor dimensión que logre capturar las propiedades temporales de la señal.

A diferencia de MNG, las reglas de ajuste de Gamma NG se obtienen mediante derivación directa de un funcional que involucra directamente los pesos y los contextos con la dinámica de la secuencia. El funcional es propuesto inicialmente como una cuantización estática de ventanas de tiempo, representada como un vector de retardos. Pero esto no asegura que la cuantización de secuencias se obtiene hasta K instantes anteriores, donde K corresponde al tamaño de la ventana de tiempo o dimensión del vector de retardos. El reemplazar la ventana por un vector cuyas componentes corresponden a filtrados de la señal original nos permite disponer de una herramienta mucho más poderosa, en la que al seleccionar filtros IIR fácilmente desacoplamos la profundidad de la memoria

del algoritmo. Se ha seleccionado el Filtro Gamma, debido a sus propiedades conocidas y a la simplicidad de su recurrencia.

La regla de ajuste del contexto se dedujo directamente desde la función de transferencia del filtro en la sección 3.2. Esta deducción permite fácilmente adaptar otros tipos de filtros y desarrollar nuevos modelos entrenamiento.

Debido a que los contextos se construyen recursivamente a partir de la cuantización, es necesario que inicialmente la contribución del contexto sea baja, hasta que el vector de cuantización w_i de la unidad adquiera una mayor fiabilidad. Si no se sigue esta recomendación se producen contextos que no son fieles a la dinámica de la secuencia y se obtiene una mala cuantización debido al error introducido por los contextos en el aprendizaje competitivo. Lo anterior puede evitarse mediante una selección adecuada del parámetro α, como se mencionó en la sección 3.3.2.

El parámetro β controla la profundidad de la memoria y la resolución de la misma, un buen compromiso entre ambas propiedades se obtiene al seleccionar un valor $\beta = 0,5$.

5.3. Costo Computacional

El modelo de contextos está montado sobre un algoritmo de cuantización, sobre el cual se comienza a generar la construcción de contextos. Una de las ventajas de este modelo es que es independiente de la topología de la red, por lo que no requiere modificaciones extras que aumenten el costo computacional del modelo. Existen distintas etapas claramente marcadas en la construcción del descriptor de contexto y el entrenamiento del contexto de las neuronas.

La construcción del descriptor de contexto requiere de la unidad ganadora en la etapa anterior por lo que no es necesario realizar cálculos para determinar que unidad fue la ganadora, sino que solamente se requiere almacenar la unidad ganadora de una época a otra.

El entrenamiento del contexto se realiza según aprendizaje Hebbiano, el cual puede ser competitivo o no competitivo, dependiendo del algoritmo sobre el cual se monte el modelo. La carga computacional adicional se debe solamente a la necesidad de cálculos adicionales del modelo de contextos, pero no aumenta el orden del algoritmo.

En el caso del algoritmo NG el mayor costo computacional se concentra en la realización de un ranking de las neuronas, mediante el uso de un buen algoritmo de ordenamiento, como QuickSort, el orden se puede reducir $O(N \log N)$, por lo que el algoritmo Gamma NG posee el mismo orden.

En el caso de SOM o de GNG no existe un ordenamiento y sólo se requiere encontrar la BMU, logrando un algoritmo de orden $O(N)$. Cabe mencionar que tanto NG como GNG han mostrado experimentalmente mejores resultados de cuantización por sobre SOM.

5.4. Selección de los Parámetros

A continuación se explica como se seleccionaron algunos de los parámetros más importantes del algoritmo

El número de neuronas M: es ajustado alrededor del 15 % o 20 % del número total de ejemplos del conjunto de entrenamiento, una menor cantidad de neuronas no logra capturar la dinámica de la serie por falta de detalle debido a un exceso de cuantización, una cantidad de vectores mayor, si bien logra una pequeña mejora en la representación, aumenta en exceso el tiempo de la simulación.

Los parámetros del modelo de contextos β y K están estrechamente relacionados con la dinámica de la serie, durante la tesis no fue posible desarrollar un criterio de ajuste adaptivo para el parámetro β que minimice el error de cuantización TQE, sin embargo se estudiaron otras variantes basadas en estadísticos de varianza 6.1.

El parámetro β: Una de las ventajas del modelo de contextos Gamma es que éste converge para cualquier valor de de β dentro del intervalo $(0, 1)$. Un valor de $\beta = 0,5$ es un buen punto de partida cuando no se tiene información alguna de la serie. En la sección 2.4.2 se describe un criterio de información mutua para determinar un retardo adecuado τ para la reconstrucción de espacio de estado, un retardo τ pequeño requiere un contexto más orientado al los valores presentes de la serie, i.e. un valor de β pequeño, de manera contraria un retardo grande requerirá un valor de β mayor. Es recomendable tener en mente que la profundidad de la memoria $D = \frac{1}{1-\beta}$ puede asociarse al retardo τ, de manera en que $\tau \sim \frac{1}{1-\beta}$.

El número de contextos K: El criterio de FNN explicado en la sección 2.4.2 permite determinar cuantos retardos son necesarios para explicar el comportamiento de la serie, una vez seleccionado el valor de β es muy posible que la profundidad de la memoria $\frac{1}{1-\beta}$ no sea lo suficientemente alta, el parámetro K permite aumentar la profundidad de la memoria $D = \frac{K}{1-\beta}$ logrando capturar una historia mayor de la secuencia.

5.5. Resultados

Se realizaron comparaciones entre distintas series de tiempo y atractores caóticos a fin de estudiar las propiedades del Algoritmo Gamma NG. A continuación se presentan los casos estudiados

5.5.1. Atractores de Rössler y Lorenz

Tanto el atractor de Rössler como el atractor de Lorenz fueron utilizados como entrada a la red Gamma NG. Al resolver las ecuaciones diferenciales no lineales que modelan el comportamiento en ambos atractores, se obtiene la solución completa de cada variable de estado, en éste caso particular el espacio de estado reside en $\mathbb{R}^3$. En la mayoría de los problemas ocurre que no todas las variables de estado son accesibles, esto se simula proyectando o sumando las coordenadas al eje real. Dado que se tiene la solución original del sistema, ésta es usada para validar la reconstrucción de espacio de estado realizada por la red. A fin de medir cuantitativamente la similitud en la reconstrucción se utilizó una medida que compara la similitud entre distintos Planos de Recurrencia. En ambos casos se obtuvo que al aumentar el número de contextos se mejoró la similitud entre el plano de recurrencia obtenido por reconstrucción mediante la red Gamma NG y el plano de recurrencia real. Debemos recordar que MNG es un caso particular de Gamma NG, cuando el número de contextos utilizados es uno. Por lo tanto al obtener mejores resultado aumentando el número de contextos estamos mejorando los posibles resultados que se puedan obtener con MNG. Para el atractor de Lorenz el plano de recurrencia de MNG ($K = 1$) Figura 4.2.2 muestra ruido y se observan puntos recurrentes en zonas donde estos no existen. A medida que aumenta el número de contextos estos puntos ruidosos van disminuyendo.

5.5.2. Series de Tiempo Mackey-Glass, Bicup 2006 y Bicup 12d

La serie Mackey-Glass ha sido ampliamente utilizada por algoritmos orientados al procesamiento de secuencias. En [15] se mostró que MNG resulta superior en términos de Error de Cuantización Temporal, por sobre otros algoritmos existentes como: TKM, RSOM y RecSOM. Para esta base de datos, la Figura 4.3.1 muestra como Gamma NG obtiene curvas de error inferiores a las obtenidas por MNG. Los mismos resultados se observan en dos series del mundo real, Bicup 2006 y Bicup 12d.

Adicionalmente se agregó un problema de predicción, en el cual se entrenó un MLP cuya entrada

corresponde a la salida de la red Gamma NG. En la base de datos de Mackey-Glass no se observa una mejoría de Gamma NG por sobre MNG debido a que la dinámica de la serie resulta ser relativamente simple y MNG ya logra resolver bastante bien el problema de reconstrucción de espacio de estado.

Al comparar las proyecciones obtenidas de los espacios de estados, mostrados en las Figuras 4.5.3 y 4.4.3, donde a pesar de haber eliminado los fines de semana ambas proyecciones son prácticamente iguales, cosa que no ocurre al comparar las Figuras 4.5.4 y 4.4.4.

5.5.3. Base de datos Fibrilación Ventricular

En esta base, para un ser humano es muy fácil identificar el momento en el que se produce el cambio en el ritmo cardiaco. La red Gamma NG mostró un mejor resultado que MNG en el porcentaje de acierto. Ambos espacios de estado resultan ser bastante similares mostrando Gamma NG una cuantización de éste menos ruidosa en las trayectorias referentes al pulso normal.

Capítulo 6

Conclusiones

El modelo de contextos basado en Memorias Gamma logra evitar el uso de ventanas de tiempo en el procesamiento de secuencias, así como obtener mejores resultados que los otros algoritmos de cuantización orientados al procesamiento de secuencias.

La proposición del modelo de contextos Gamma generaliza el modelo de contextos propuesto en Merge SOM, al punto en que este último se puede recuperar como un caso particular del modelo Gamma cuando el número de contextos utilizados por cada neurona se reduce a uno. Esto permitió aprovechar el marco teórico desarrollado para MSOM sobre el cual ya se han estudiado diferentes propiedades que avalan el desempeño del algoritmo. En el caso particular de MSOM, se ha demostrado que el modelo recursivo converge a una eficiente codificación fractal que se obtiene como punto fijo de la dinámica del entrenamiento de la red, al igual que otros algoritmos recursivos como TKM y RSOM.

En la sección 3.1 se logró proponer un funcional que considera los instantes anteriores de la secuencia. Inicialmente se utiliza un funcional simple basado en ventanas de tiempo, para luego ser adaptado a elementos más complejos basados en filtros Gamma. La minimización del funcional propuesto permite al algoritmo agrupar características temporales similares, cumpliendo así con el propósito de la cuantización temporal.

El modelo propuesto se basa en la utilización de filtros IIR como herramienta fundamental para la construcción de los contextos. La utilización de filtros Gamma da origen a la construcción del algoritmo Gamma NG. Al ser Gamma NG una generalización de MNG, se tiene un nuevo marco

teórico que explica el comportamiento de este último algoritmo. La utilización del Filtro Gamma entrega al algoritmo una gran robustez y tolerancia al ruido.

Se han utilizado distintas bases de datos benchmark y del mundo real que muestran como el algoritmo Gamma NG supera los resultados de distintos algoritmos ya existentes. Los atractores caóticos permitieron mostrar como el aumento del número de contextos mejora la reconstrucción de espacio de estado realizado por el modelo de contextos, justificándose así la una de las superioridades de Gamma NG por sobre MNG.

Para el atractor de Rössler el mínimo de la función de error se obtuvo para $K = 7$, sobre MNG ($K = 1$); cabe mencionar que el error es máximo cuando $K = 1$. Validando los resultados previos con el atractor de Lorenz, el algoritmo Gamma NG muestra sus mejores resultados para $K = 9$ y al igual que en el atractor de Rössler, la disimilitud entre la reconstrucción y el espacio de estado original es máxima con $K = 1$. Al aumentar el número de contextos el algoritmo Gamma NG presenta mejores capacidades para reconstruir el espacio de estado que el algoritmo MNG. Esto es de gran importancia, puesto que el espacio de estado corresponde a la solución de las ecuaciones diferenciales que modelan el problema, el poder aproximarlo puede entregarnos una manera fácil de resolver un problema no lineal complejo.

El algoritmo Gamma NG fue también comparado con MNG en tareas de cuantización, para las cuales se utilizó la base de datos Mackey-Glass, la cual es recurrentemente utilizada en problemas de procesamiento temporal. En Mackey-Glass, las curvas del Error de Cuantización Temporal (TQE) obtenidas para Gamma NG ($2 \leq K \leq 9$) son todas menores que las obtenidas por MNG ($K = 1$).

Para la serie BICUP, la cual corresponde a una base de datos del mundo real, el algoritmo Gamma NG tiene un mejor desempeño que MNG, lo que puede comprobarse en el error de cuantización temporal donde claramente las curvas para $K > 1$ muestran un error menor que $K = 1$. En lo que respecta a tareas de predicción realizadas sobre la misma serie, los errores resultan comparables y es difícil concluir si el algoritmo Gamma NG obtiene mejores resultados.

En tareas de clasificación, Gamma NG obtuvo mejores resultados sobre MNG. Para comprobar este punto se utilizaron bases de datos del mundo real, relacionadas con un fenómeno cardíaco específico llamado fibrilación ventricular. El algoritmo Gamma NG logró un porcentaje de acierto del $94,72\,\%$ mientras que MNG llegó a un $89,81\,\%$.

6.1. Recomendaciones para trabajo Futuro

Una de las principales desventajas del algoritmo es que al aumentar el número de contextos aumenta también el numero de parámetros del algoritmo. Muchas veces se escoge el valor $\beta = 0,5$ considerándose un buen compromiso entre la profundidad de la memoria y la resolución. Para las series variantes en el tiempo los estadísticos cambian a medida que se recorre la serie. De esta forma pueden encontrarse series correlacionadas temporalmente con periodos muy pequeños, lo que sugiere un valor de β pequeño que aumente la resolución del algoritmo a fin de no perder información relevante. Al contrario, series de tiempo correlacionadas temporalmente con periodos largos requerirían valores de β mayores a fin de no incluir información redundante en la reconstrucción del espacio de estado. En lo anterior reside la razón principal del porqué un modelo adaptivo que permita ajustar el valor de β automáticamente de acuerdo a la dinámica de la serie sería muy atractivo.

Capítulo 7

Modelo Adaptivo

Uno de los parámetros de gran importancia dentro del algoritmo es el parámetro β, el cual controla la profundidad y la resolución de la memoria de corto plazo. Por el momento este parámetros es elegido dentro del rango $[0-1]$ con un valor típico de 0.5, el cual corresponde a un buen compromiso entre la profundidad de la memoria y la resolución.

A continuación se propone un modelo basado en la minimización de la varianza entre la salida de la red, definida como una combinación lineal de los contextos, y la señal de entrada, esto es

$$\text{mín}\, V = Var(x - y) \tag{7.0.1}$$

donde y^t es un estimador insesgado de x^t

$$y^t = \sum_{k=1}^{K} \lambda_k c_k^{I_t} + \lambda_0 \tag{7.0.2}$$

Es de interés que junto con la minimización de la varianza se pueda eliminar el sesgo de la salida con respecto a la señal original. Esto se puede hacer imponiendo condiciones sobre λ_0, lo cual se desarrolla a continuación. Primero se calcula la esperanza de los contextos para luego imponer condiciones sobre la esperanza de la salida.

- Esperanza de los contextos

$$E(c_k^t) = E(c_{k-1}^t) = \ldots = E(x^t) \tag{7.0.3}$$

Tomando esperanza a la definición del contexto

$$\begin{aligned} E(c_k^t) &= E(\beta c_k^{t-1} + (1-\beta)c_{k-1}^{t-1}) \\ &= \beta E(c_k^{t-1}) + (1-\beta)E(x_{k-1}^{t-1}) \end{aligned}$$

suponiendo que el tiempo tiende a infinito, para cualquier señal s^t se cumple que $E(s^t) = E(s^{t-1})$, por lo tanto

$$(1-\beta)E(c_k^t) = (1-\beta)E(c_{k-1}^t)$$

evaluando en $k = 1$ se tiene que $E(c_1^t) = E(x^t)$, luego se procede por inducción.

- Esperanza de la salida

$$E(y^t) = E(x^t), \quad \lambda_0 = E(x^t)\left(1 - \sum_{k=1}^{K} \lambda_k\right) \tag{7.0.4}$$

La salida es insesgada si se impone que $E(y^t) = E(x^t)$, usando además la propiedad anterior, se tiene

$$\sum_{k=1}^{K} \lambda_k E(c_k^{I_t}) + \lambda_0 = \sum_{k=1}^{K} \lambda_k E(x^t) + \lambda_0 = E(x^t)$$

despejando λ_0 en la última igualdad

$$\lambda_0 = E(x^t)\left(1 - \sum_{k=1}^{K} \lambda_k\right)$$

7.1. Merge Adaptivo

En el caso más simple, cuando se tiene un contexto, la salida se puede definir simplemente como $y^t = c^t$, en este caso dado que $E(c^t) = E(x^t)$, $\lambda_0 = 0$ hace que la salida sea insesgada. Supondremos que la entrada x^t está dada por una señal s^t y un ruido gausiano aditivo n^t

$$x^t = s^t + n^t \tag{7.1.1}$$

donde n^t tiene $E(n^t) = 0$ y $Var(n^t) = \sigma^2$, el cual no esta correlacionado con la señal. La varianza entre la salida y^t y la entrada x^t

$$\begin{aligned} Var(x^t - c^t) &= Var(x^t) + Var(c^t) - 2cov(x^t, c^t) \\ &= Var(s^t) + Var(n^t) - 2cov(s^t, n^t) + Var(c^t) - 2cov(x^t, c^t) \\ &= Var(s^t) + Var(n^t) + Var(c^t) - 2cov(x^t, c^t) \end{aligned}$$

dado que $cov(s^t, n^t) = 0$. Por otro lado, utilizando la bilinealidad de la covarianza agrupando $Var(s^t) + Var(c^t) - 2cov(s^t, c^t) = Var(s^t - c^t)$, se tiene que

$$Var(x^t - c^t) = Var(s^t - c^t) + \sigma^2 - 2cov(n^t, c^t). \tag{7.1.2}$$

La ecuación anterior indica que al minimizar la varianza entre la entrada y la salida se minimiza la varianza entre la señal s y la salida, pero también se maximiza la covarianza entre el ruido y el contexto. Estudiando el término $cov(n^t, c^t)$ probaremos que éste tiende a cero cuando $t \to \infty$.

Desarrollando $c^t = \beta c^{t-1} + (1 - \beta)x^{t-1}$ en la covarianza se tiene

$$\begin{aligned} cov(n^t, c^t) &= \beta cov(n^t, c^{t-1}) + (1 - \beta)cov(n^t, x^{t-1}) \\ &= \beta cov(n^t, c^{t-1}) + (1 - \beta)cov(n^t, s^{t-1}) + (1 - \beta)cov(n^t, n^{t-1}) \end{aligned}$$

dado que $x^t = s^t + n^t$. Puesto que no hay correlación entre la señal y el ruido $cov(n^t, s^{t-1}) = 0$ y que la para un proceso de ruido blanco con media cero, tiene como correlación $E(n^{t_1} n^{t_2}) \sim \delta(t_1 - t_2)$, se tiene que $cov(n^t, n^{t-1}) = 0$. Luego

$$cov(n^t, c^t) = \beta cov(n^t, c^{t-1}) = \ldots = \beta^N cov(n^t, c^{t-N}) \tag{7.1.3}$$

en el límite

$$\lim_N cov(n^t, c^t) = \lim_N \beta^N cov(n^t, c^{t-N}) = 0 \tag{7.1.4}$$

Según lo anterior se tiene que

$$Var(x^t - c^t) = Var(s^t - c^t) + \sigma^2 \tag{7.1.5}$$

así la minimización entre la señal de entrada y la salida, minimiza la varianza entre la salida y la señal limpia.

$$\text{mín}\, \hat{V} = Var(s^t - c^t) \tag{7.1.6}$$

7.1.1. Regla de Ajuste β para el Modelo de Contextos de Merge

Es posible derivar una regla de aprendizaje para el parámetro β, de manera que este se ajuste en forma adaptiva. Esto se obtiene derivando V con respecto a la variable β. Si se define $\Delta^t = x^t - c^t$, entonces según la definición de varianza

$$V = \frac{1}{T}\sum_{t=1}^{T}\left(\Delta^t - \frac{1}{T}\sum_{t=1}^{T}\Delta^t\right)^2 \tag{7.1.7}$$

luego

$$\frac{\partial V}{\partial \beta} = \frac{1}{T}\sum_{t=1}^{T}\left(\Delta^t - \frac{1}{T}\sum_{t=1}^{T}\Delta^t\right)\left(\frac{\partial \Delta^t}{\partial \beta} + \frac{1}{T}\sum_{t=1}^{T}\frac{\partial \Delta^t}{\partial \beta}\right) \tag{7.1.8}$$

donde $\frac{\partial \Delta^t}{\partial \beta} = -\frac{\partial c^t}{\partial \beta}$, si se considera que en el modelo de contexto de MNG, $c^t = \beta c^{I_{t-1}} + (1-\beta)w^{I_{t-1}}$, entonces

$$\frac{\partial c^t}{\partial \beta} = c^{I_{t-1}} - w^{I_{t-1}} \tag{7.1.9}$$

finalmente, se tiene que

$$\beta(t+1) = \beta(t) - \eta\frac{\partial V}{\partial \beta} \tag{7.1.10}$$

7.1.2. Resultados

Se utilizó una sinusoide a la cual se le agregó ruido aditivo, que fue generado mediante una distribución normal de media $\mu = 0$ y desviación σ la que se varió entre los siguientes valores $\sigma = [0,1; 0,2; 0,3; 0,4; 0,5; 0,6]$. Primero se corrió el algoritmo variando el valor de β con paso $0,1$ entre los valores 0 y 1, a fin de explorar como varía la varianza entre la señal y el ruido para distintos valores de β, luego se corrió el algoritmo adaptivo. Los resultados se muestran a continuación.

En las Figuras se puede observar que el modelo adaptivo adapta el valor de β a un valor cercano al óptimo de mínima varianza entre las señales.

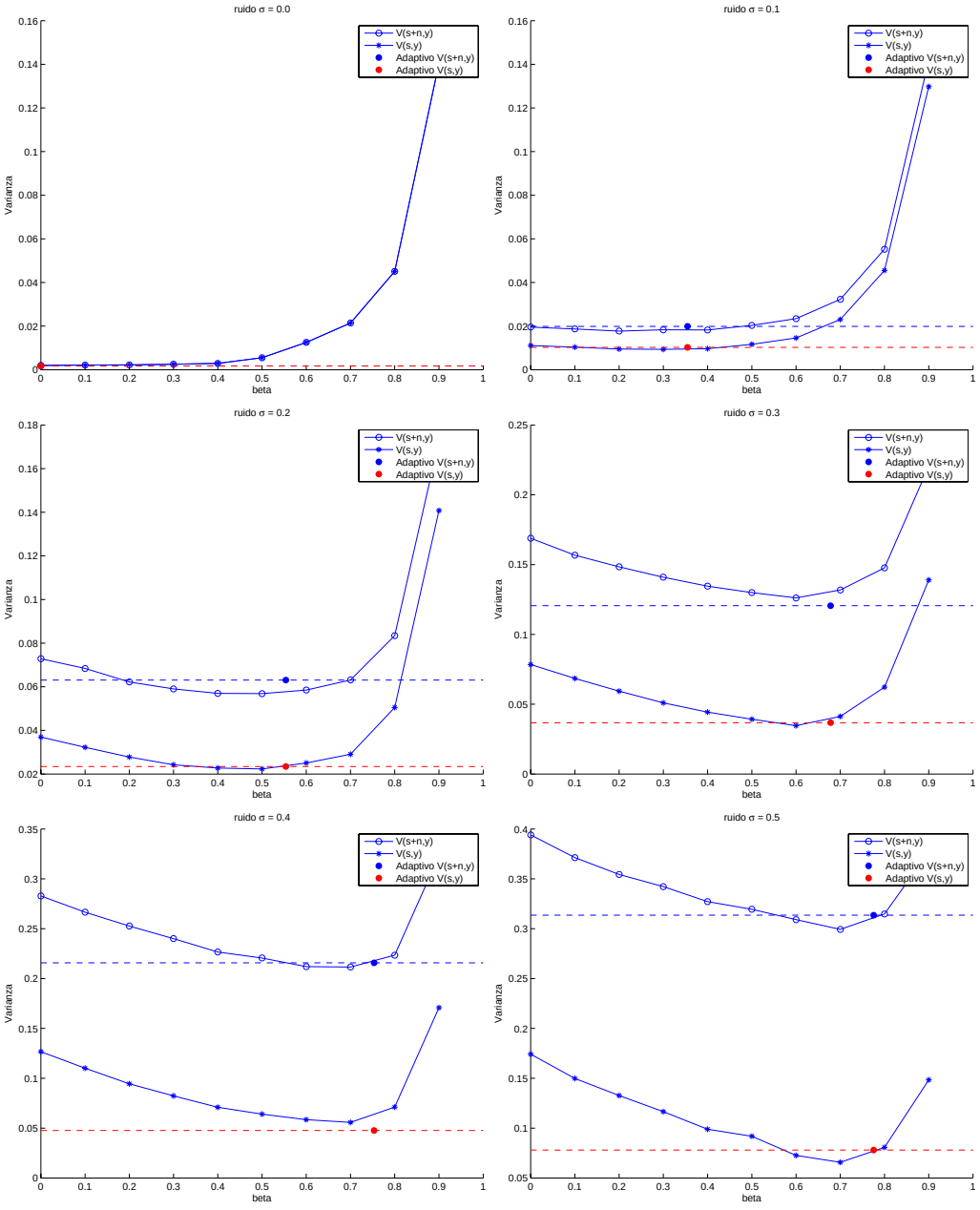


Figura 7.1.1: Resultados para MNG Adaptivo. Las Figuras muestran la varianza $V(s+n,y)$ que corresponde a la varianza entre la señal s incluyendo el ruido aditivo n y la señal de salida y y para $V(s,y)$ que corresponde a la señal s sin ruido y la salida y. Ambas varianzas se muestran para distintos valores de β.

7.2. Modelo de Contextos Gamma Adaptivo

Es interesante poder adaptar el modelo adaptivo desarrollado para MNG al modelo de Contextos Gamma, puesto que en este último caso la cantidad de parámetros β es mayor, ya que se tiene uno por cada etapa de filtrado, complicando aún más la elección de estos parámetros en forma experimental. De manera análoga se tiene que

$$Var(x^t - y^t) = Var(s^t - y^t) + \sigma^2 - 2cov(n^t, y^t) \tag{7.2.1}$$

desarrollando el último término de la ecuación anterior, utilizando la bilinealidad de la covarianza, se tiene

$$cov(n^t, y^t) = y^t = \sum_{k=1}^{K} \lambda_k cov(n^t, c_k^t) \tag{7.2.2}$$

ya se demostró en una etapa anterior que $cov(n^t, c_1^t) = 0$, luego

$$cov(n^t, c_2^t) = \beta cov(n^t, c_2^{t-1}) + (1-\beta)cov(n^t, c_1^{t-1}) = \beta cov(n^t, c_2^{t-1}) = \beta^N cov(n^t, c_2^{t-N}) \tag{7.2.3}$$

bajo el mismo supuesto con el que se procedió para $cov(n^t, c_1^t)$, se tiene que $cov(n^t, c_2^t) = 0$, de forma inductiva se tiene que

$$cov(n^t, c_k^t) = 0, \quad \forall k = 1 \dots K \tag{7.2.4}$$

luego

$$Var(x^t - y^t) = Var(s^t - y^t) + \sigma^2 \tag{7.2.5}$$

Regla de Ajuste de β para el Modelo de Contextos Gamma La principal diferencia con el modelo adaptivo para Merge es que en este caso también se debe encontrar una regla para adaptar los ponderadores λ_k de cada contexto c_k^t. La regla se obtiene derivando V con respecto a la variable β_k y λ_k. Si se define $\Delta^t = x^t - c^t$, entonces se tiene

$$\frac{\partial V}{\partial \beta_k} = \frac{1}{T}\sum_{t=1}^{T}\left(\Delta^t - \frac{1}{T}\sum_{t=1}^{T}\Delta^t\right)\left(\frac{\partial \Delta^t}{\partial \beta_k} + \frac{1}{T}\sum_{t=1}^{T}\frac{\partial \Delta^t}{\partial \beta_k}\right) \tag{7.2.6}$$

donde

$$\frac{\partial \Delta^t}{\partial \beta_j} = -\frac{\partial y^t}{\partial \beta_j} = -\lambda_j \frac{\partial c_j^t}{\partial \beta_j} \tag{7.2.7}$$

si se considera que en el modelo de contexto Gamma, $c_j^t = \beta_j c_j^{I_{t-1}} + (1-\beta_j) c_{j-1}^{I_{t-1}}$, entonces

$$\frac{\partial \Delta^t}{\partial \beta_j} = -\lambda_j \left(c_j^{I_{t-1}} - c_{j-1}^{I_{t-1}} \right) \tag{7.2.8}$$

finalmente, se tiene que

$$\beta_k(t+1) = \beta_k(t) - \eta(t) \frac{\partial V}{\partial \beta_k} \tag{7.2.9}$$

donde $\eta(t) = \eta_i \left(\frac{\eta_f}{\eta_i} \right)^{t/t_{max}}$.

Al derivar con respecto a λ_j se tiene

$$\frac{\partial V}{\partial \lambda_k} = \frac{1}{T} \sum_{t=1}^{T} \left(\Delta^t - \frac{1}{T} \sum_{t=1}^{T} \Delta^t \right) \left(\frac{\partial \Delta^t}{\partial \lambda_k} + \frac{1}{T} \sum_{t=1}^{T} \frac{\partial \Delta^t}{\partial \lambda_k} \right) \tag{7.2.10}$$

donde

$$\frac{\partial \Delta^t}{\partial \lambda_j} = -\frac{\partial y^t}{\partial \lambda_j} = -c_j^{I_t} \tag{7.2.11}$$

con lo que finalmente se tiene

$$\lambda_k(t+1) = \lambda_k(t) - \eta \frac{\partial V}{\partial \lambda_k} \tag{7.2.12}$$

donde $\eta(t) = \eta_i \left(\frac{\eta_f}{\eta_i} \right)^{t/t_{max}}$.

El hecho de no incluir el peso w^i en la salida se debe estrictamente a que $y^t = 1 \cdot w^{I_t} + \sum_{k=1}^{K} 0 \cdot c_k^{I_t}$ es solución al problema de minimización buscado.

Eventualmente en tareas de predicción, es posible definir

$$\text{mín}\, V = Var(x^{t+1} - y^t) \tag{7.2.13}$$

donde $y^t = y^t = \sum_{k=1}^{K} \lambda_k c_k^{I_t} + \lambda_0 w^{I_t} + \lambda_p$.

Bibliografía

[1] P. Werbos, *Beyond regression: new tools for prediction and analysis in the behavioral science.* PhD thesis, Harvard University, 1974.

[2] D. B. Parker, "Learning-logic," Invention Report S81-64, File 1, Office of Technology Licensing, Stanford University, Palo Alto, CA, 1982.

[3] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[4] B. De Vries, J. C. Principe, and Guedes, "Adaline with adaptive recursive memory," in *Neural Networks for Signal Processing [1991]., Proceedings of the 1991 IEEE Workshop*, pp. 101–110, 1991.

[5] J. C. Principe, B. de Vries, and P. G. de Olivieira, "The gamma filter – A new class of adaptive IIR filters with restricted feedback," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 649–656, 1993.

[6] J. C. Principe, J.-M. Kuo, and S. Celebi, "An analysis of the gamma memory in dynamic neural networks," *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 331–337, 1994.

[7] M. Motter and J. Principe, "A gamma memory neural network for system identification," *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, vol. 5, pp. 3232–3237 vol.5, Jun-2 Jul 1994.

[8] Y. Li and Z. Ding, "Arma system identification based on second-order cyclostationarity," *Signal Processing, IEEE Transactions on*, vol. 42, pp. 3483–3494, Dec 1994.

[9] K. H. Chon and R. J. Cohen, "Linear and nonlinear arma model parameter estimation using an artificial neural network," *Biomedical Engineering, IEEE Transactions on*, vol. 44, no. 3, pp. 168–174, 1997.

[10] A. Kowalski and D. Szynal, "An optimal prediction in general arma models," *Journal of Multivariate Analysis*, vol. 34, pp. 14–36, July 1990.

[11] K. S. Man, "Long memory time series and short term forecasts," *International Journal of Forecasting*, vol. 19, no. 3, pp. 477–491, 2003.

[12] K. Kumar, "Pade approximation and its application in time series analysis," *Applied Mathematics and Computation*, vol. 48, pp. 139–151, April 1992.

[13] F. Takens, "Detecting strange attractors in turbulence," in *Dynamical systems and turbulence* (D. A. Rand and L. S. Young, eds.), pp. 366–381, New York: Spinger-Verlag, 1980.

[14] M. Strickert and B. Hammer, "Neural gas for sequences," in *WSOM'03*, pp. 53–57, 2003.

[15] M. Strickert and B. Hammer, "Merge som for temporal data," *Neurocomputing*, vol. 64, pp. 39–71, 2005.

[16] T. Martinetz and K. Schulten, "A "neural-gas"network learns topologies," *Artificial Neural Networks*, vol. I, pp. 397–402, 1991.

[17] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems 7* (G. Tesauro, D. S. Touretzky, and T. K. Leen, eds.), pp. 625–632, Cambridge MA: MIT Press, 1995.

[18] S. M. Omohundro, "The delaunay triangulation and function learning;," Tech. Rep. TR-90-001, Berkeley, CA, 1990.

[19] G. J. Chappell and J. G. Taylor, "The temporal kohonen map," *Neural Netw.*, vol. 6, no. 3, pp. 441–445, 1993.

[20] T. Kohonen, K. Mkisara, O. Simula, and J. Kangas, "The hypermap architecture," *Artificial Neural Networks*, pp. 1357–1360, 1991.

[21] M. Varsta, J. Heikkonen, and J. del R. Millan, "Context learning with the self organizing map," in *Proceedings of WSOM'97, Workshop on Self-Organizing Maps, Espoo, Finland, June 4–6*, pp. 197–202, Espoo, Finland: Helsinki University of Technology, Neural Networks Research Centre, 1997.

[22] T. Koskela, M. Varsta, J. Heikkonen, and K. Kaski, "Temporal sequence processing using recurrent som," *Knowledge-Based Intelligent Electronic Systems, 1998. Proceedings KES '98. 1998 Second International Conference on*, vol. 1, pp. 290–297 vol.1, Apr 1998.

[23] M. Varsta, J. Heikkonen, J. Lampinen, and J. D. R. Millán, "Temporal kohonen map and the recurrent self-organizing map: Analytical and experimental comparison," *Neural Process. Lett.*, vol. 13, no. 3, pp. 237–251, 2001.

[24] T. Koskela, M. Varsta, J. Heikkonen, and K. Kaski, "Time series prediction using RSOM with local linear modesl," Tech. Rep. B15, Espoo, Finland, 1997.

[25] J. Principe, N. Euliano, and S. Garani, "Principles and networks for self-organization in space-time," *Neural Networks*, vol. 15, no. 8-9, pp. 1069–1083, 2002.

[26] T. Voegtlin, "Recursive self-organizing maps," *Neural Networks*, vol. 15, no. 8-9, pp. 979–991, 2002.

[27] T. Voegtlin, *Neural Networks and Self-Reference*. PhD thesis, L'Université Lumière Lyon 2, 2002.

[28] M. Strickert and B. Hammer, "Unsupervised recursive sequence processing," in *Neurocomputing*, pp. 433–439, D-side Publications, 2003.

[29] B. Hammer, A. Micheli, A. Sperduti, and M. Strickert, "Recursive self-organizing network models," *Neural Networks*, vol. 17, no. 8-9, pp. 1061–1085, 2004.

[30] P. EstÃ©vez, R. Zilleruelo-Ramos, R. HernÃ¡ndez, L. Causa, and C. Held, "Sleep spindle detection by using merge neural gas," in *WSOM'07 Bielefeld/Germany*, 2007.

[31] N. K. Perugini and W. E. Engeler, "Principles and networks for self-organization in space-time," *Proceedings of the International Joint conference on neural networks*, vol. 2, pp. 395–401, 1989.

[32] B. de Vries and J. C. Principe, "The gamma model – a new neural network for temporal processing," *Neural Networks*, vol. 5, no. 4, pp. 565–576, 1992.

[33] M. Motter and J. Principe, "Classification and prediction of wind tunnel mach number responses using both competitive and gamma neural networks," *Proc. World Conference on Neural Networks*, vol. 2, pp. 25–29, 1995.

[34] J.-M. Kuo and J. C. Principe, "Speech classification using a modified focused gamma network," in *Neural Networks, 1996., IEEE International Conference on*, vol. 4, pp. 1877–1882 vol.4, 1996.

[35] C. Wang, D. Xu, and J. C. Principe, "Speaker verification and identification using gamma neural networks," in *Neural Networks,1997., International Conference on*, vol. 4, pp. 2085–2088 vol.4, 1997.

[36] M. Palkar and J. C. Principe, "Echo cancellation with the gamma filter," in *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. iii, pp. III/369–III/372 vol.3, 1994.

[37] S. Celebi and J. Principe, "Analysis of spectral feature extraction using the gamma filter," *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, vol. 7, pp. 4497–4501 vol.7, Jun-2 Jul 1994.

[38] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems: Fundamentals through Simulations with CD-ROM.* New York, NY, USA: John Wiley & Sons, Inc., 1999.

[39] L. Kocarev, J. Szczepanski, J. Amigo, and I. Tomovski, "Discrete chaos-i: Theory," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 53, pp. 1300–1309, June 2006.

[40] G. Jakimoski and K. Subbalakshmi, "Discrete lyapunov exponent and differential cryptanalysis," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 54, pp. 499–501, June 2007.

[41] H. Abarbanel, *Analysis of observed chaotic data.* Springer, 1996.

[42] M. Berthold and D. J. Hand, eds., *Intelligent data analysis.* New York, NY, USA: Springer-Verlag New York, Inc., 2003.

[43] M. Kennel, R. Brown, and H. D. I. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," *Phys. Rev. A.*, vol. 45, pp. 3403–3411, 1992.

[44] L. Cao, "Practical method for determining the minimum embedding dimension of a scalar time series," *Phys. D*, vol. 110, no. 1-2, pp. 43–50, 1997.

[45] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle, "Recurrence plots of dynamical systems," *Europhysics Letters*, vol. 4, pp. 973–977, 1987.

[46] G. B. Mindlin and R. Gilmore, "Topological analysis and synthesis of chaotic time series," *Physica D*, vol. 58, pp. 229–242, 1992.

[47] J. P. Zbilut, J. M. Zaldivar-Comenges, and F. Strozzi, "Recurrence quantification based liapunov exponents for monitoring divergence in experimental data," *Phys Lett Sect A Gen At Solid State Phys*, vol. 297, no. 3-4, pp. 173–181, 2002.

[48] J. Gao and Z. Zheng, "Direct dynamical test for deterministic chaos and optimal embedding of a chaotic time series," *Phys. Rev. E*, vol. 49, pp. 3807–3814, May 1994.

[49] R. Manuca and R. Savit, "Stationarity and nonstationarity in time series analysis," *Phys. D*, vol. 99, no. 2-3, pp. 134–161, 1996.

[50] J. S. Iwanski and E. Bradley, "Recurrence plots of experimental data: To embed or not to embed?," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 8, no. 4, pp. 861–871, 1998.

[51] J. M. Choi, B. H. Bae, and S. Y. Kim, "Divergence in perpendicular recurrence plot; quantification of dynamical divergence from short chaotic time series," *Physics Letters A*, vol. 263, pp. 299–306, Dec. 1999.

[52] J. P. Zbilut and C. L. Webber, "Embeddings and delays as derived from quantification of recurrence plots," *Physics Letters A*, vol. 171, pp. 199–203, Dec. 1992.

[53] C. L. W. Jr. and J. Zbilut, "Dynamical assessment of physiological systems and states using recurrence plot strategies," *Journal of Applied Physiology*, vol. 76, no. 2, pp. 965–973, 1994.

[54] N. Marwan and J. Kurths, "Nonlinear analysis of bivariate data with cross recurrence plots," *Physics Letters A*, vol. 302, p. 5, 2002.

[55] N. Marwan, N. Wessel, U. Meyerfeldt, A. Schirdewan, and J. Kurths, "Recurrence-plot-based measures of complexity and their application to heart-rate-variability data," *Phys. Rev. E*, vol. 66, p. 026702, Aug 2002.

[56] T. Heskes, "Energy functions for self-organizing maps," in *Kohonen Maps* (S. K. E. Oja, ed.), Amsterdam: Elsevier.

[57] S. S. K. Martinetz, T.M.; Berkovich, "'neural-gas' network for vector quantization and its application to time-series prediction," *Neural Networks, IEEE Transactions on*, vol. 4, no. 4, pp. 558–569, Jul 1993.

[58] G. de A. Barreto and A. F. R. Araújo, "Time in self-organizing maps: An overview of models," *International Journal of Computer Research, Special Issue: Past, Present and Future of Neural Networks*, vol. 10, no. 2, pp. 139–79, 2001. Guest Editors: P.G. Anderson and G. Antoniou and V. Mladenov and E. Oja and M. Paprzycki and N. C. Steele.

[59] P. A. Estévez and C. J. Figueroa, "Online data visualization using the neural gas network," *Neural Networks*, vol. 19, no. 6, pp. 923–934, 2006.

[60] M. C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," vol. 197, pp. 287–289, 1977.

[61] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000 (June 13). Circulation Electronic Pages: http://circ.ahajournals.org/cgi/content/full/101/23/e215.