UNIVERSIDAD NACIONAL AUTÓNOMA

DE MÉXICO

FACULTAD DE CIENCIAS

Análisis Genético de las secuencias de la subunidad pequeña de ribosomas de Eukarya, Archaea y Bacteria mediante Mapeos Autoorganizados.

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

MATEMÁTICO

P R E S E N T A :

CARLOS ALBERTO MONSALVO ORTIZ

FACULTAD DE CIENCIAS
UNAM

DIRECTOR DE TESIS:

Dr. PEDRO EDUARDO MIRAMONTES VIDAL

2015

CIUDAD UNIVERSITARIA, D. F.

1. Datos del alumno

Apellido paterno	Monsalvo
Apellido materno	Ortiz
Nombre(s)	Carlos Alberto
Teléfono	13 14 19 80
Universidad Nacional Autónoma de México	Universidad Nacional Autónoma de México
Facultad de Ciencias	Facultad de Ciencias
Carrera	Matemáticas
Número de cuenta	407024162

2. Datos del tutor

Grado	Dr.
Nombre(s)	Pedro Eduardo
Apellido paterno	Miramontes
Apellido materno	Vidal

3. Datos del sinodal 1

Grado	Dr.
Nombre(s)	Adonis Germinal
Apellido paterno	Cocho
Apellido materno	Gil

4. Datos del sinodal 2

Grado	Dr.
Nombre(s)	Manuel Jesús
Apellido paterno	Falconi
Apellido materno	Magaña

5. Datos del sinodal 3

Grado	Dra.
Nombre(s)	Natalia Bárbara
Apellido paterno	Mantilla
Apellido materno	Beniers

6. Datos del sinodal 4

Grado	Dra.
Nombre(s)	Mariana
Apellido paterno	Benítez
Apellido materno	Keinrad

7.Datos del trabajo escrito.

Título: Análisis Genético de las secuencias de la subunidad pequeña de ribosomas de Eukarya, Archaea y Bacteria mediante Mapeos Autoorganizados.

Número de páginas: 67 p.

Agradecimientos.

A mis padres.

Lorenzo y Graciela, quienes me han dado todo su amor y comprensión, sin ellos ninguno de mis logros habría sido posible.

A mis familiares.

A mi tita Alma y Marifer, quienes siempre me han abierto las puertas de su casa para recibirme, darme su cariño y apoyo incondicional, a mi hermano Andres, un ejemplo de lucha y perseverancia, a mis tíos José y Lolita, siempre solidarios, a mi tío Lalo y mi tierna tía Goyita, a mi Princesa Pilar la gran compañera de mi vida, a mi suegra Félix, con quien he compartido hermosos momentos y es un ejemplo a seguir y finalmente a toda la familia Torres Reyes, quienes me han hecho sentir parte de la misma.

A mis maestros.

Al Dr. Pedro Miramontes, por aceptar ser mi asesor y apoyarme inclusive estando lejos, a mis sinodales, Dr. Germinal, Dr. Manuel, Dra. Natalia, Dra. Mariana y al L. en C.C. Sergio Hernández, por dedicar su tiempo a la lectura y análisis de este trabajo.

A mis amigos.

Los de la facultad, Víctor, Pablo, Iván, los Ricardos (Haki y Pina), Oscar, Charly, Lucio y Diana, con quienes he compartido muchas risas, fiestas, deportes, clases e incluso poemas y finalmente a mi mejor amigo Arturo, con quien he compartido muchos momentos muy gratos desde la preparatoria, a todos Gracias.

Índice general

Introducción

El presente trabajo propone el uso de los *Mapeos Autoorganizados* para la clasificación de diferentes organismos, basándose sólo en propiedades moleculares que los caractericen. Esta herramienta es un tipo particular de *Red Neuronal Artificial* que tiene aplicaciones en una muy amplia variedad de campos que van desde la ingeniería, pasando por la medicina, biología hasta la economía por mencionar algunos [14],[15].

Al respecto de las redes neuronales artificiales, en el Capítulo 1 se hace una breve revisión histórica de aquellas que han sido relevantes en el desarrollo de esta disciplina. Además, se hace una descripción detallada de los mapeos autoorganizados y se exponen sus principales características como son, la *compresión de datos* y la *reducción dimensional*, de igual forma, se pone de manifiesto la relación existente con las redes neuronales biológicas.

En el capítulo 2, se hace una revisión de lo que se conoce como el Dogma Central de la Biología Molecular, el cual, pretende esclarecer como se usa la información contenida en el DNA, mediante procesos celulares conocidos como *transcripción* y *traducción*. En este mismo apartado se introducen los *Ribosomas*, macromolécula central para la traducción, que principalmente consideramos para nuestro estudio posterior en en el Capítulo 3.

Dentro de este último capítulo aprovechamos, las capacidades de los mapeos autoorganizados, para el análisis de una base de datos, construida tomando en cuanta los RNA ribosomales de una amplia variedad de especies, donde, mediante un procese iterativo, la red neuronal aprende a distinguir mas allá de los tres principales dominios conocidos Archaea, Bacteria y Eukarya. Finalmente los resultados son comparados con técnicas de Agrupamiento Jerárquico y el Mapeo es examinado con medidas de error topográfico

y error de cuantización.

Capítulo 1

Redes de Mapeos Autoorganizados

En general, entenderemos el termino *Red Neuronal Artificial*, como una estructura que posee un conjunto de unidades básicas que llamaremos *neuronas* las cuales interactúan entre sí a través de sus conexiones con el propósito de producir un estímulo o salida y que tiene como principal objetivo el de imitar, de la forma mas fiel posible, algunos de los comportamiento de los sistemas neuronales biológicos.

1.1. Un poco de Historia

La redes neuronales artificiales tienen su origen en el trabajo presentado en 1943 por Warren S. McCulloch y Walter Pitts [13] quienes propusieron un modelo matemático basado en la idea de que las neuronas sólo pueden tener dos estados o salidas posibles: apagado (0) o encendido (1), y este estado dependerá de que el valor de entrada o estímulo o la suma de varios estímulos proporcionados a la neurona supere o no cierto umbral θ que la caracteriza. Así el estado de una neurona al tiempo $t+1$ puede ser descrito de la siguiente manera:

$$\mathcal{S}(t+1) = \begin{cases} 1 & \text{si la suma de los estímulos supera el umbral } \theta \\ 0 & \text{en otro caso} \end{cases}$$

Posteriormente en 1949 Donal Hebb [8, pág. 63] propone su teoría sobre el aprendizaje y en su libro *Organization of Behavior* postula lo siguiente: "...Cuando ... una célula A está lo suficientemente cerca para excitar una célula B, y repetidamente o de forma persistente toma parte en dispararla, algún proceso de crecimiento o *cambio metabólico* toma lugar en una o ambas células, de modo que la eficiencia de A para disparar a B se incrementa.". A este cambio metabólico se le relaciona con el *aprendizaje*[1], el cual, puede considerarse como un tema central cuando se habla de redes neuronales artificiales.

Mas adelante, en 1957, Frank Rosenblatt plantea un modelo denominado *Perceptrón*, el cual se puede ser considerado como una generalización al propuesto por McCulloch-Pitts añadiendo a éste la capacidad de aprender con el objetivo de realizar tares de clasificación después de un periodo de *entrenamiento*. Así por ejemplo si tomamos en cuenta dos categorías A y B, y los estados o salidas posibles como {1,-1}, si la red neuronal aprendió bien y tiene éxito el estado de la neurona podrá interpretarse de la siguiente manera:

- Si la red produce salida 1, la entrada pertenece a la categoría A.
- Si la red produce salida -1, la entrada pertenece a la categoría B.

Aquí el proceso de entrenamiento consiste en ir introduciendo a la red patrones ya conocidos tanto de la clase A como de la clase B y el de aprendizaje en modificar las conexiones entre neuronas que participan de esta red para poder producir el resultado deseado: 1 en el caso en el que algún patrón $\vec{x}$ pertenezca a la clase A y -1 si pertenece a la clase B, esto independientemente si $\vec{x}$ pertenece o no a lo que llamaremos de ahora en adelante el *conjunto de entrenamiento*, denotado como X.

Ahora bien, si consideramos por ejemplo el caso de dos dimensiones, es decir, cuando $\vec{x} \in X \subset \mathbb{R}^2$, entonces, lo que se pretende es encontrar la recta $w_1x_1 + w_2x_2 + \theta = 0$ con pendiente $-\frac{w_1}{w_2}$ y ordenada al origen $-\frac{\theta}{w_2}$ para

[1]Cajal, hacia 1894 ya había advertido sobre la posibilidad de que estos cambios metabólicos entre la conexiones de neuronas estuvieran relacionadas con el aprendizaje.

algún $w_1, w_2, \theta \in \mathbb{R}$ que permita separar adecuadamente (si esto es posible) las categorías A y B de modo que se obtenga que:

$$\forall \vec{a} \in A : w_1 a_1 + w_2 a_2 + \theta > 0$$
$$\forall \vec{b} \in B : w_1 b_1 + w_2 b_2 + \theta < 0$$

Aquí el proceso de aprendizaje cambiará en el tiempo los valores inicialmente asignados a las conexiones entre neuronas w_1 y w_2, mediante un proceso iterativo que dependerá de los valores de salida $\{-1, 1\}$ ofrecidos por la red para cada uno de los datos del conjunto de entrenamiento, para conseguir el objetivo deseado de separar las categorías apropiadamente.

Tiempo mas tarde en 1960, Bernard Widrow diseñó una red neuronal muy parecida al perceptrón, a la que denomino Adaptive Linear Element o *Adaline*. Esta red se distingue de las anteriores en que permite como salida cualquier número real, por lo que ahora, tomando en cuenta un conjunto P que se obtiene del producto cartesiano entre el conjunto X y un conjunto D que contienen los valores $d_{\vec{x}} \in \mathbb{R}$ que se desean obtener de la red al introducir en ella un vector $\vec{x} \in X$, entonces, P queda definido como:

$$P = \{(\vec{x}, d_{\vec{x}}) | \vec{x} \in X \subset \mathbb{R}^n, d_{\vec{x}} \in \mathbb{R}\}$$

Por lo que el aprendizaje en esta ocasión depende de la diferencia $|d_{\vec{x}} - y_{\vec{x}}|^2$ donde $y_{\vec{x}}$ es la salida producida por la red y $d_{\vec{x}}$, como hemos dicho, es el valor esperado para un vector $\vec{x} \in X$ del conjunto de entrenamiento. Esta diferencia permite saber en cuánto se ha equivocado la red y no sólo determinar si se ha equivocado o no, aquí el proceso de aprendizaje es conocido como la *regla Delta*.

Este tipo de modelos abre la posibilidad de resolver problemas tanto de clasificación como de aproximación de funciones gracias al conjunto de ejemplos que se les proporciona. Sin embargo se ha observado que estos tienen grandes limitaciones al intentar resolver problemas sencillos, tal como sucede en el caso de la función OR exclusivo que se describe por el siguiente conjunto P:

[2]Medida como *voltaje* en el Adaline

$\vec{x}$		$d_{\vec{x}}$
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1

Cuadro 1.1: Conjunto P para la función OR exclusivo

Aquí al intentar generar una red neuronal que tenga como salida 1 cuando el vector $\vec{x}$ de entrada sea $(-1,-1)$ ó $(1,1)$, o bien se obtenga -1 cuando el vector de entrada $\vec{x}$ sea $(-1,1)$ o $(1,-1)$ no se puede conseguir. Y esto se debe en realidad a que el problema no es linealmente separable, es decir, a la no existencia de una línea recta $l \subset \mathbb{R}^2$ que separe eficientemente el conjunto de vectores $\{(-1,-1),(1,1)\}$ del conjunto $\{(-1,1),(1,-1)\}$.

Una manera de resolver este problema es añadir al perceptrón más perceptrones y organizarlos en capas para construir una red. La existencia de capas es una propiedad que también poseen las redes neuronales biológicas, tal es el caso de la corteza cerebral que en el caso de los humanos contiene de tres a seis capas cada una con rasgos funcionales y anatómicos característicos[10]. Esta idea de añadir capas para solucionar las limitaciones del Perceptrón fue propuesta por primera vez por Minsky y Papert en 1969 [16, pág. 231]. Sin embargo no fue sino hasta septiembre de 1985 que Rumelhart, Hinton y Williams [17] presentaron un algoritmo de aprendizaje capaz de resolver problemas no lineales como el aquí mencionado.

Para este *Perceptrón Multicapa* la regla de aprendizaje no es tan distinta de las anteriores, el mecanismo sigue siendo el de modificar los parámetros $w_i, \theta_j, i, j \in I$ de la red con el objetivo de obtener las salidas deseadas (o lo mas próximas posibles) después de ir introduciendo en ella vectores de entrenamiento. Naturalmente que los parámetros que le pertenecen a las capas ocultas deben ser tratados y adaptados de manera distinta que aquellos de la capa mas externa de la red.

De este modo si consideramos una función E que nos hable del error cometido por la red que tome en cuenta la diferencia que existe entre la salida de la red y la salida deseada y consideramos un vector $W(0) \in \mathbb{R}^m$ donde m es el número de parámetros de la red. Entonces el proceso de aprendizaje

consiste en ir desplazando el vector de parámetros $W(t-1)$ hasta una posición $W(t)$ en el espacio $\mathbb{R}^m$ de tal modo que este nuevo vector minimice la función de error E. Luego este proceso debe de continuar hasta encontrar un vector W^* que garantice un mínimo ($min_W E$) de la función E.

Las redes neuronales hasta aquí expuestas pueden ser consideradas como del tipo supervisadas, es decir, requieren de un conjunto de entrenamiento en el cual se sabe para cada dato a que clase pertenecen. Sin embargo, ¿Qué pasaría si nos encontramos con una base de datos de la cual se ignora por completo para cada dato, la clase a la que pertenecen?. Cuando este es el caso, se debe recurrir a redes neuronales no supervisadas, como lo son los *Mapeos Autoorganizados*.

1.2. Aprendizaje

Hasta el momento, sólo se ha mencionado que el *aprendizaje* es una característica esencial de las redes neuronales artificiales, pero en general no se ha esclarecido de que manera esto ocurre para cada una de ellas, por lo que el objetivo de la sección es exhibir a grandes rasgos dónde y cómo se lleva a cabo este proceso comenzando con las redes neuronales biológicas y trasladando el concepto a las redes neuronales artificiales.

Sinapsis

Primero, para establecer como es el mecanismo de comunicación entre neuronas hablaremos un poco de la estructura y función de las mismas. Empezaremos diciendo que la principales funciones de una neurona son *recibir, conducir y trasmitir señales*. Cada neurona consta de un cuerpo o soma celular el cual contiene al núcleo y para llevar a cabo sus funciones las neuronas tienen prolongaciones o ramificaciones que serán de dos tipos: Los *axones* que son los encargados de conducir las señales desde el soma hasta una célula vecina y las *dendritas* que son las encargadas de recibir las señales de los axones de otras neuronas.

El punto de contacto entre los axones de una neurona y las dendritas de

otra lo llamaremos *sinapsis*, y a la neurona que transmite la señal la llamaremos *presináptica* y a la que la recibe la llamaremos *postsináptica*, entonces, para que se lleve a cabo la comunicación debe ocurrir lo siguiente: el primer paso lo da la neurona presináptica, esto lo hace generando un estímulo eléctrico denominado *potencial de acción* o *impulso eléctrico*, el cual es transmitido a lo largo de de todo el axón hasta llegar al final de este, donde desencadena la liberación de pequeñas moléculas señal conocidas como *neurotransmisores* los cuales se encuentran almacenadas en compartimentos denominados *vesículas*.

Una vez liberados los neurotransmisores, éstos llegan a la membrana de la neurona postsináptica y son recibidos por un conjunto de proteínas que forman canales iónicos (es decir canales que sólo permiten el paso de iones) como pago para la apertura de éstos, generando así una membrana permeable que permite un flujo de diversos iones. A estos canales los denominaremos *canales iónicos regulados por transmisores*, y una característica importante de ellos es la posibilidad de transformar nuevamente estas señales químicas en señales eléctricas.

Los canales iónicos regulados por transmisores, son selectivos con respecto al tipo de iones que dejan pasar a través de la membrana y esta propiedad determinará la respuesta de la neurona postsináptica que puede ser ser de tipo excitadora o inhibidora. Cuando la respuesta es de tipo excitatoria es debido a la presencia de neurotransmisores que abren canales catiónicos (i.e que sólo permiten la entrada de iones con carga positiva tales como Na^+ y K^+) provocando así que la neurona postsináptica dispare su potencial de acción o impulso eléctrico, por el contrario cuando la respuesta es es de tipo inhibitorio es debido a la presencia de neurotransmisores que abren canales aniónicos (i.e que sólo permiten la entrada de iones con carga negativa, como Cl^-). lo que suprime la posibilidad de un potencial de acción o impulso eléctrico por parte de la neurona postsináptica.

Una característica muy importante, es el hecho de que las conexiones sinápticas se pueden fortalecer cuando la respuesta de las neurona postsináptica es de tipo excitatorio y por el contrario se pueden debilitar si la respuesta es de tipo inhibitoria. Este fortalecimiento o debilitamiento de la sinapsis se ve reflejado en un aumento (disminución) de la cantidad de canales iónicos de la neurona postsináptica, favoreciendo así una mejor comunica-

ción entre las neuronas, al mismo tiempo de que existe una modificación en la forma y tamaño de cierto tipo de prolongaciones dentro de la dendrita que son llamadas *espinas dendríticas*. A esta capacidad de modificar, fortalecer o debilitar las conexiones sinápticas se le conoce como *plasticidad sináptica*. De modo que, es precisamente esta cualidad de plasticidad, el cambio metabólico al que Hebb se refería al proponer su teoría de aprendizaje.

Flujo de señal

Consideraremos ahora una abstracción de las conexiones que se establecen entre neuronas y esto lo haremos mediante lo que llamaremos *gráficas de flujo de señal* (Figura 1.1) originalmente desarrollados por Mason (1953, 1963). Estas gráficas proveen un método para describir, como su nombre lo dice, el flujo de señales entre neuronas, además de que nos permitirán la construcción de diferentes arquitecturas de las redes neuronales artificiales, las cuales tienen las siguientes características:

a) Las gráficas de flujo de señal serán descritas por líneas (flechas) que conecten un par de puntos que llamaremos *nodos* o *neuronas*, las cuales, indicarán en que dirección fluyen las señales.

b) Una conexión típica entre nodos $i \longrightarrow j$ tendrá asociada una señal x_i y y_j respectivamente, donde la señal y_j dependerá de la señal x_i.

c) Toda conexión entre neuronas también tendrá asociada una *función f de transferencia*, que especifica la manera en la que la señal y_j depende de la señal x_i

Así, por ejemplo, en el caso de los modelos del Perceptrón y Perceptrón Multicapa, la forma en la que la señal salida y depende de los vectores $\vec{x} = (x_1, ..., x_m) \in X$ del conjunto de entrenamiento vienen dada por:

$$y = f(\sum_{i=1}^{m} w_i x_i + \theta)$$

$$y_i = f(\sum_{j=1}^{n_{C-1}} w_{ji}^{C-1} a_j^{C-1} + u_i^C))$$

Donde f, como hemos dicho, para el caso del Perceptrón, es una función de salida binaria $\{-1, 1\}$ y para el caso del Perceptrón Multicapa es una función de imagen continua, como por ejemplo, la función *sigmoidal* y la función *tangente hiperbólica*, que toman valores $f(\alpha)$ dentro de los intervalos $[0, 1]$ y $[-1, 1]$ respectivamente.

Además, cuando se trata de redes neuronales artificiales, suele ponerse para cada conexión entre neuronas (representada por dichas gráficas), un parámetro adicional w_{kj}, que se puede interpretar de alguna manera como la fuerza (o debilidad) del enlace que existe entre cada par de neuronas vinculadas. A este parámetro lo llamaremos *peso sináptico*. Entonces, como puede apreciarse en las dos funciones de transferencia anteriores, resulta, que su evaluación también depende del valor asignado a los pesos sinápticos w_i y w_{ji} respectivamente.

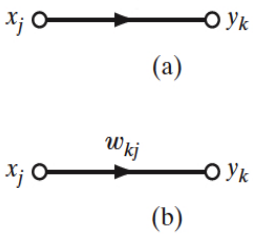

Figura 1.1: *Gráfica de flujo de señal.* (a) Típica gráfica con sus respectivas señales y nodos x_j ,y_k. (b) Gráfica a la que se le ha añadido su respectivo peso sináptico w_{kj}. Notemos que el subíndice es kj y no jk.

Y finalmente, como hemos dicho anteriormente, modificar los pesos sinápticos, será la forma de emular el aprendizaje. Entonces veamos como es que esto sucede.

Aprendizaje Hebbiano

Ahora que ya tenemos una forma preliminar de representar las conexiones entre neuronas tratemos de formular el concepto de aprendizaje, pero antes de esto debemos precisar mejor la idea y dar algunas características de este, en primer lugar daremos una nueva formulación del *Aprendijzaje hebbiano* en las dos siguientes reglas: (Stent, 1973, Changeus y Danchin, 1976).

1) Si dos neuronas en ambos lados de una conexión sináptica son activadas simultáneamente, es decir, de forma sincronizada, entonces, la fuerza de la sinapsis será incrementada.

2) Si dos neuronas en ambos lados de una sinapsis son activadas no simultáneamente, entonces esta sinapsis es debilitada o eliminada.

Así después de esta definición de sinapsis hebbiana, se pueden observar algunas características importantes tales como la *dependencia temporal*, que se refiere a que la modificación de la sinapsis depende del tiempo exacto de la ocurrencia de la señales, también se observa que es un *mecanismo local*, es decir que la modificación de la sinapsis sólo se lleva acabo entre neuronas cercanas y una tercera característica es su *mecanismo interactivo*, es decir que es dependiente de ambas señales tanto la presináptica como la postsináptica.

De modo que con esta definición y estas características estamos en posición de formular el aprendizaje hebbiano en términos matemáticos de la siguiente manera: Consideremos un par de neuronas presináptica y postsináptica con señales x_j y y_k respectivamente y consideremos el peso sináptico w_{kj} que relaciona a dichas neuronas. Entonces el ajuste aplicado a dicho peso sináptico será expresado en su forma general como:

$$\Delta w_{kj}(t) = f\left(y_k(t), x_j(t)\right)$$

Donde observamos que este ajuste es una función que depende del tiempo (t) y de las señales $x_j(t)$ y $y_k(t)$, además de que es un ajuste es local como lo requiere el aprendizaje hebbiano.Una forma particular de esta expresión puede ser como sigue:

$$\Delta w_{kj}(t) = \eta y_k(t)\, x_j(t)$$

Donde η es una constante positiva que denominaremos factor de aprendizaje, que nos habla de que tan rápido o lento puede llegar a ser este cambio en los pesos sinápticos, es decir que tan rápido o despacio aprendemos. Esta forma que hemos expuesto es la mas simple de aprendizaje hebbiano.

Mapas de la Corteza Cerebral

Ahora consideraremos otra propiedad importante que también posen las redes neuronales biológicas y es la de formar mapas en la *corteza cerebral*, la capa mas externa del cerebro. Sabemos que el cerebro es el responsable de procesar toda la información que proviene del exterior, información que es recibida por nuestros sentidos (vista, tacto, oído, olfato, gusto) la cual es finalmente procesada en lo que llamaremos *corteza sensorial*.

Cada uno de nuestros sentidos es procesado en áreas especificas de esta corteza sensorial y todas estas regiones tienen en común que son representaciones topológicas de sus respectivas superficies receptoras de estímulos (superficies epiteliales), formado así mapas de estas. Así por ejemplo podemos decir que el mapa de la superficie de la piel se caracteriza por el hecho de que regiones cercanas en la corteza responden a estímulos cercanos en la piel es decir el mapa es *somatotópico*.

De igual manera el mapa del sistema auditivo esta organizado de manera *tonotópica*, es decir tiene una representación espacial de las frecuencias de tal forma que neuronas cercanas entre sí responden a frecuencias cercanas (figura 1.2), lo mismo sucede con la vista donde células cercanas en la retina se mapean en células cercanas de la corteza es decir presentan una organización *retinotópica*.

Una característica importante que se ha demostrado con respecto a estas representaciones corticales es el hecho de que no son fijas, mas bien son dinámicas y se modifican continuamente por la experiencia. Además, se asume como mecanismo de formación de estos cambios, que están dados principalmente por la plasticidad sináptica, siguiendo las reglas del aprendizaje hebbianno descritas anteriormente.

1.3. Algoritmo de Mapeos Autoorganizados

Ahora consideraremos un algoritmo matemático que tiene como principal objetivo el de imitar los mapas topográficos producidos de manera natural en la corteza cerebral. Nos referimos a un caso especial de red neuronal arti-

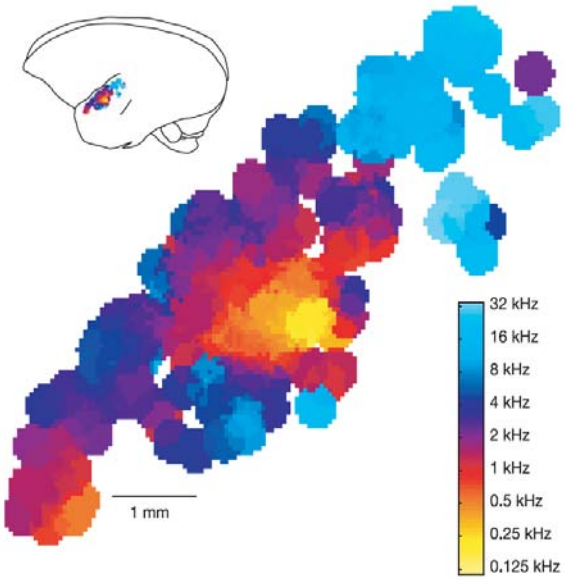

Figura 1.2: Mapa tonotópico de la corteza auditiva en primates.

ficial conocido como *Mapeo Autoorganizado* (SOM), tomando como modelo principal y descrito por Kohonen en 1982, en donde, de manera general y con el objetivo de generar el mapa, el algoritmo se lleva a cabo en tres fases distintas denominadas *competencia, cooperación y adaptación* que describiremos mas adelante.

Pero antes de describir con precisión el algoritmo hablemos de su arquitectura (figura 1.3), la cual consta de una capa bidimensional de nodos conectados entre sí formando una red en donde se construirá el mapa, el cual recibirá estímulos externos que serán vectores de cualquier dimensión los cuales denominaremos *datos de entrada*. El algoritmo tendrá la tarea de organizar estos nodos de la red en vecindarios locales que actuarán como clasificadores de las características de los datos de entrada.

La generación de este mapa topográfico se llevará a cabo mediante un proceso cíclico de comparación de los datos de entrada con vectores de la misma dimensión "almacenados" en cada nodo. El objetivo es encontrar dentro de este conjunto de vectores que representan cada nodo al vector que mejor se le

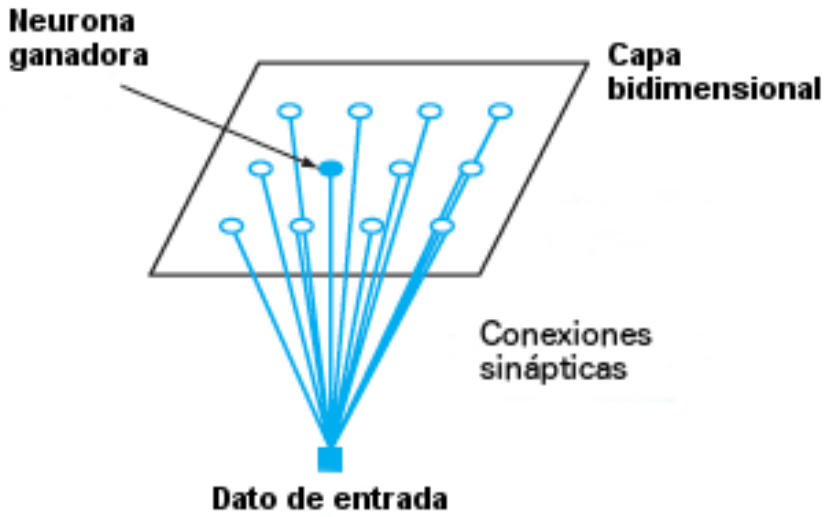

Figura 1.3: Arquitectura del modelo de Kohonen

parece al dato de entrada y al nodo cuyo vector cumple ser el mas parecido al dato de entrada lo llamaremos *neurona ganadora* o, por sus siglas del inglés *Best matching Unit*, BMU.

Ahora definiremos de una manera más precisa el algoritmo de los mapeos autoorganizados. Consideremos que estamos en presencia de un conjunto de estímulos o datos de entrada de dimensión m, entonces un dato de entrada es de la forma:

$$\vec{x} = [x_1, x_2, ..., x_m]^T$$

Para iniciar el algoritmo debemos "almacenar" en cada nodo un vector inicial de manera aleatoria, estos vectores serán llamados pesos sinápticos y serán denotados de la forma :

$$\vec{w}_j = [w_{j1}, w_{j2}, ..., w_{jm}]^T$$

donde j pertenece a un conjunto de índices (que denotaremos con la letra Λ) que representa el j-ésimo nodo de la red de neuronas. Una vez que se ha dado a cada nodo su peso sináptico consideramos que se ha inicializado el algoritmo.

1. Fase Competencia

Una vez que se ha inicializado el algoritmo se pasa a una segunda etapa donde a la red se le presentan los datos de entrada y las neuronas o nodos compiten entre ellas para determinar a la ganadora, es decir se determinará cual de ellas tiene asociado el peso sináptico mas parecido al dato de entrada esto se hace tomando todas las distancias que existen entre el vector estímulo $\vec{x}$ y los pesos $\vec{w}_j$ con $j\epsilon\Lambda$ tomando como ganadora la que tenga la menor distancia. Es decir se toma:

$$d\left(\vec{x},\vec{w}_j\right)=\sqrt{\sum_{i=1}^{m}\left(w_{ji}-x_i\right)^2}\qquad\forall\, j\,\epsilon\,\Lambda$$

Y debe tomarse el subíndice $j\,\epsilon\,\Lambda$ tal que la distancia entre $\vec{x}$ y $\vec{w}_j$ sea mínima, o equivalentemente, podemos decir que debe tomarse el subíndice $j\,\epsilon\,\Lambda$ de forma que el producto punto $\vec{w}_j^T\cdot\vec{x}$ sea máximo. Una forma de expresar lo anterior es de la siguiente manera:

$$i(\vec{x})=arg\min_j\|\vec{x}-\vec{w}_j\|$$

Donde el símbolo $arg\min_w f\left(w\right)$ representa al argumento w tal que hace mínima a la función $f\left(w\right)$, de modo que para la expresión $i\left(\vec{x}\right)$ estamos precisamente pidiendo lo deseado; es decir, el subíndice $j\epsilon\Lambda$ tal que hace que la distancia $\|\vec{x}-\vec{w}_j\|$ sea mínima. De modo que la neurona j con vector representante $\vec{w}_j$ es la neurona ganadora.

2. Fase Cooperativa

Tras la obtención de la neurona ganadora debemos pasar a una etapa más avanzada del algoritmo, donde debemos asumir que esta neurona ganadora se ha vuelto una neurona activa y como consecuencia ahora debe estimular a sus neuronas vecinas por lo que debe definirse una vecindad alrededor de dicha neurona, la cual define el conjunto de neuronas que ahora serán afectadas por dicho estímulo.

Dos propiedades importantes deben caracterizar a esta vecindad también llamada de *cooperación*, la primera es el hecho de que esta vecindad debe ser simétrica con respecto a la neurona ganadora, y la segunda es que entre más alejada se encuentre una neurona con respecto a la ganadora debe verse menos afectada por su estímulo. Entonces precisemos un poco mas estas ideas.

Denotando como $h_{j,i}$ la vecindad de cooperación centrada en la neurona i y como $d_{j,i}$ la distancia que existe entre la neurona i y una neurona j entonces se está en búsqueda de una función $h_{j,i}$ que dependa de $d_{j,i}$ que cumpla con las siguientes características:

1) La función vecindad $h_{j,i}$ debe ser simétrica con respecto a la neurona j, tal que $d_{j,i} = 0$ y además aquí dicha función debe obtener su valor máximo.

2) La función vecindad $h_{j,i}$ debe decrecer monótonamente hacia cero a medida de que $d_{j,i} \longrightarrow \infty$

Un par de funciones que cumple con estos dos requisitos son:

$$h_{j,i(x)} = exp\left(-\frac{d_{i,j}^2}{2\sigma^2}\right)$$

$$h_{j,i(x)} = max\{0, 1 - (\sigma - d_{j,i})^2\}$$

Donde $d_{j,i}$ esta dada por $d_{j,i}^2 = \|\vec{r}_j - \vec{r}_i\|^2$ en la cual los vectores $\vec{r}_i$,$\vec{r}_j$ de las neuronas i, j representan respectivamente la posición que ocupan dentro de la capa bidimensional y el parámetro σ representa que tan extensa es esta vecindad topológica. Debemos notar que se ha incluido en el subíndice la función $i\,(\vec{x})$, haciendo referencia a que la vecindad siempre tendrá como centro a la neurona ganadora para cada iteración del algoritmo.

Una característica muy importante del algoritmo es que dicha vecindad debe contraerse con respecto al tiempo, esto hace que el parámetro σ ahora deba de ser considerado como una función decreciente en el tiempo, esto con el objetivo de ir haciendo cada vez mas especifico el mapa topológico que se pretende desarrollar, de esta forma la función $\sigma\,(t)$ puede ser como sigue:

$$\sigma(t) = \sigma_0 exp\left(-\frac{t}{\tau_1}\right)$$

Aquí σ_0 es un valor para σ al inicio del algoritmo y donde τ_1 es un valor constante que entre mayor sea su valor mas tiempo tardara la función $\sigma(t)$ en tender al valor cero. Así entonces, para que la vecindad dependa y se reduzca con respecto al tiempo debemos reescribir $h_{j,i(x)}$ ahora en términos de la nueva función $\sigma(t)$ como sigue:

$$h_{j,i(x)}(t) = exp\left(-\frac{d_{i,j}^2}{2\sigma^2(t)}\right)$$

3. Fase Adaptativa

Ahora que se ya ha determinado una vecindad que define al conjunto de neuronas que serán modificadas por el estímulo de la neurona ganadora inclusive, debemos decir de que manera éstas se verán alteradas. El hecho es que la modificación debe llevarse a cabo en los pesos sinápticos, esto se hará siguiendo una regla que puede considerarse como una modificación de la regla de Hebb descrita anteriormente. El objetivo es que los vectores de los pesos sinápticos sean lo mas parecido posible al valor de entrada $\vec{x}$.

Para esto, debemos primero decir quienes son las señales x_j y y_k que son esenciales para la construcción de una forma que represente de alguna manera al aprendizaje hebbiano. Entonces definiremos como $x_j = \vec{x}$, el dato de entrada y a $y_k = h_{k,i(x)}$. Consideraremos también un termino de olvido que será de la forma $g(y_k) w_k$ donde w_k es el peso sináptico de la neurona k y $g(y_k)$ es una función escalar positiva con la propiedad de que $g(y_k) = 0$, cuando $y_k = 0$. Entonces, el cambio a los pesos sinápticos queda como sigue:

$$\Delta w_k = \eta y_k x - g(y_k) w_k$$

Notemos que la primera parte es la regla de Hebb (pág. 16) donde η es como antes el factor de aprendizaje, a esta regla le hemos restado el término

de olvido con el objetivo de que Δw_k no crezca indefinidamente. Ahora bien si consideramos como $g(y_k) = \eta y_k$ obtenemos.

$$\Delta w_k = \eta y_k x - \eta y_k w_k$$

O bien,

$$\Delta w_k = \eta h_{j,i(x)}(x - w_j) \quad dado,\ y_k = h_{k,i(x)}$$

Finalmente tendríamos entonces que el valor de los pesos sinápticos $\{w_k\}_{j\,\epsilon,\Lambda}$ puede ser definido recursivamente para el tiempo $t+1$ como sigue:

$$w_k(t+1) = w_k(t) + \Delta w_k(t)$$

Es decir,

$$w_k(t+1) = w_k(t) + \eta(t)\, h_{j,i(x)}(t)\,(x(t) - w_k(t))$$

Con esta adaptación de los pesos sinápticos concluye la descripción del mecanismo iterativo que se lleva a cabo para cada dato de entrada en el modelo de Kohonen de los mapeos autoorganizados, lo que sigue es ver algunas propiedades que los caracterizan.

1.4. Propiedades

Compresión de Datos

Una de las características mas importantes de los mapeos autoorganizados es el de almacenar mediante un proceso de compresión grandes cantidades de datos representados como vectores $\vec{x} \in X \subset \mathbb{R}^m$ en un pequeño conjunto de vectores $\vec{w_k} \in \mathcal{W}$ de la misma dimensión a los que ahora llamaremos prototipos, de tal manera que el conjunto $\mathcal{W} := \{\vec{w_k}\}$ con $k = \{1, 2, ..N\}$, proporcione una buena aproximación del conjunto original X.

Entonces, podemos ahora de manera mas general pensar a este algoritmo de mapeo autoorganizado que acabamos de describir como una función $\Phi : X \subset \mathbb{R}^m \to W$ que va del espacio X de datos de entrada al espacio discreto de los pesos sinápticos $\{\vec{w}_i\} = W$, tal que a cada dato $\vec{x} \in X$ se le asigna un peso sináptico $\vec{w}_i$ que mejor lo represente en un proceso de codificación y compresión del conjunto de datos de entrada.

Bajo estos términos podemos ahora pensar a la función Φ como un *cuantizador vectorial* de dimensión m y tamaño $|W|$, donde ahora, dentro de esta teoría de cuantización el conjunto W es llamado libro de códigos (*codebook*). Aquí a cada $\vec{w}_i \in W$ tiene asociado una región $\mathcal{R}_i = \{\vec{x} \in \mathbb{R}^m \mid \Phi(\vec{x}) = \vec{w}_i\}$ que cumple con las siguientes condiciones:

1. $\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$ para $i \neq j$
2. $\bigcup_{i=1}^{|W|} \mathcal{R}_i = X$

Ahora bien ya que estamos en un proceso de compresión, cuando $\vec{x}$ es cuantizado como $\vec{w}_i$ se origina un error o bien una distorsión que puede ser medida de diferentes formas las más comunes ya que suelen ser las más convenientes en términos matemáticos son las siguientes:

$$d(\vec{x}, \vec{w}_j) = \left\{ \sum_{i=1}^{k} |x_i - w_{ji}|^v \right\}^{\frac{1}{v}} := \|\vec{x} - \vec{w}_j\|_v$$

$$d(\vec{x}, \vec{w}_j) = \sum_{i=1}^{k} |x_i - w_{ji}|^v := \|\vec{x} - \vec{w}_j\|^v$$

Es claro que entre más pequeña sea la medida de distorsión, mejor es la representación que se tiene de los datos de entrada. Por lo que para hablar del desempeño que tiene un cuantificador Φ se recurre por lo general a la

medida $\mathrm{D}(\Phi) = \mathrm{E}\left[d(X, \Phi(X))\right]$ donde E representa el valor esperado.

Entonces decimos que un cuantizador Φ es óptimo si minimiza el valor esperado de la distorsión, esto es, Φ^* es óptimo si para todo cuantizador Φ que tiene N vectores de referencia en el libro de códigos se cumple $\mathrm{D}(\Phi^*) \leq \mathrm{D}(\Phi)$.Considerando ahora una función de distribución $\mathrm{P}(X)$ de los vectores de entrada X, entonces $\mathrm{D}(\Phi)$ se puede escribir como:

$$\mathrm{E}\left[d(X, \Phi(X))\right] = \sum_i d(X, \Phi(X))\mathrm{P}(X)$$

$$\mathrm{E}\left[d(X, \Phi(X))\right] = \int d(X, \Phi(X))\mathrm{P}(X)$$

Un manera para minimizar a la función $\mathrm{D}(\Phi)$ es mediante el método de *gradiente en descenso*, donde tomando valores iniciales $\vec{w}_r(0)$ con $r \in \{1, 2, ..., N\}$, todos los vectores de referencia son cambiados de acuerdo a la regla:

$$\vec{w}_r(t+1) = \vec{w}_r(t) - \frac{\eta}{2}\frac{\partial \mathrm{E}}{\partial \vec{w}_r}$$

Y si considerando como caso particular el valor esperado

$$\mathrm{E}\left[\|\vec{x} - \vec{w}_j\|^2\right] = \int \|\vec{x} - \vec{w}_j\|^2 \mathrm{P}(X)$$

Entonces el método de gradiente descendente queda como:

$$\vec{w}_r(t+1) = \vec{w}_r(t) + \eta \int_{i(\vec{x})=r} (\vec{x} - \vec{w}_r(t))\mathrm{P}(X)$$

Donde la aplicación repetida de este procedimiento dado un valor pequeño de el parámetro η conduce a la disminución de la función $\mathrm{D}(\Phi)$ hacia un mínimo local. Donde la condición $i(\vec{x}) = r$ restringe la región de integración a aquellos valores $\vec{x}$ tal que $\vec{w}_r$ es el mejor valor de referencia es decir nos estamos restringiendo a la región R_r

Sin embargo, no siempre es posible encontrar para todo conjunto $X \subset \mathbb{R}^m$ su función de distribución $\mathrm{P}(X)$. Esta dificultad puede superarse de alguna manera si remplazamos la expresión anterior por la aproximación:

$$\vec{w}_r(t+1) = \vec{w}_r(t) + \eta(\vec{x} - \vec{w}_r(t))$$

Al observar esta ecuación y compararla con la obtenida en la descripción de la fase adaptativa de nuestro algoritmo de Kohonen, podemos notar que son casi idénticas, salvo la función vecindad $h_{j,i}$. Así nuestro mapeo autoorganizado puede ser entendido como un procedimiento de cuantización o compresión basado en un método de gradiente descendente.

Formación de Mapas Somatotópicos

Ya hemos mencionado las propiedad somatotópica que tiene la corteza cerebral, entonces, vamos ahora a considerar un modelo algo distinto del algoritmo de Kohonen donde se tomara en cuenta una superficie sensorial en forma de mano, la cual contiene *receptores táctiles* y también una típica capa bidimensional de neuronas, de tal manera que cada neurona esta conectada con cada uno de los receptores, dejando así un conjunto de conexiones adaptativas, cuya fuerza inicial es seleccionada de manera aleatoria y que será modificada durante un proceso de simulación de estimulación táctil.

Esta estimulación será modelada por un contorno de excitación localizado de tipo gaussiano, que describirá la salida r_i del receptor i en la posición $\vec{x}_i$ en función de la distancia al centro del estimulo $\vec{x}_s$ de la siguiente manera:

$$r_i = Aexp[-(\vec{x}_i - \vec{x}_s)^2/\sigma_r^2]$$

Donde el ancho σ_r y la intensidad A del estímulo serán constantes a lo largo del proceso de estimulación y donde para cada paso en el proceso de adaptación el centro $\vec{x}_s$ tomará una posición diferente de manera aleatoria. Cada neurona (k,l) en la posición $\vec{y}_{kl}$ de la red de neuronas calculará una suma ponderada sobre todas las salidas de los receptores como sigue:

$$o_{kl} = \sum_i w_{kli} r_i$$

En donde w_{kli} denota la fuerza de la conexión o el peso sináptico que existe entre la neurona (k, l) y el receptor i. Además el dato de entrada para cada neurona queda establecido por el vector $\vec{r} = (r_1, r_2, ..., r_N)$, donde N representa el número total de receptores, de modo que, si seguimos el algoritmo de Kohonen debe seleccionarse una neurona ganadora, es decir, aquella neurona (r, s) tal que $o_{kl} \leq o_{rs}$ para toda neurona (k, l).

En este orden de ideas ahora debe de considerarse la modificación de las salidas o_{kl} de las neuronas (k, l) mediante una función de salida gaussiana $h_{rs;kl}$ centrada en la posición $\vec{y}_{rs}$ de tal manera que:

$$h_{rs;kl}(t) = exp[-(\vec{y}_{rs} - \vec{y}_{kl})^2 / \sigma_h^2(t)]$$

Como antes la vecindad $\sigma_h^2(t)$ debe decrecer desde un valor inicial σ_i hasta un valor final σ_f para permitir que las neuronas se especialicen. Finalmente los pesos sinápticos deben ser cambiados de acuerdo a una forma particular de aprendizaje hebbiano y una manera de hacerlo es como a continuación se describe:

$$w_{kli}(t+1) = (w_{kli}(t) + \epsilon(t) h_{rs;kl}(t) \cdot r_i) / \sum_i w_{kli}(t)$$

En donde el ancho de paso de aprendizaje $\epsilon(t)$ decrece exponencialmente desde un valor inicial ϵ_i hasta un valor ϵ_f. El resultado de una simulación de este proceso que consideró 16,384 neuronas, 800 receptores táctiles y un total

de 13,107,200 conexiones adaptativas pude verse en la Figura 1.4. Aquí se puede observar (parte inferior) como la capa bidimensional de neuronas fue cambiando hasta obtener el objetivo deseado de formar un mapa somatotópico.

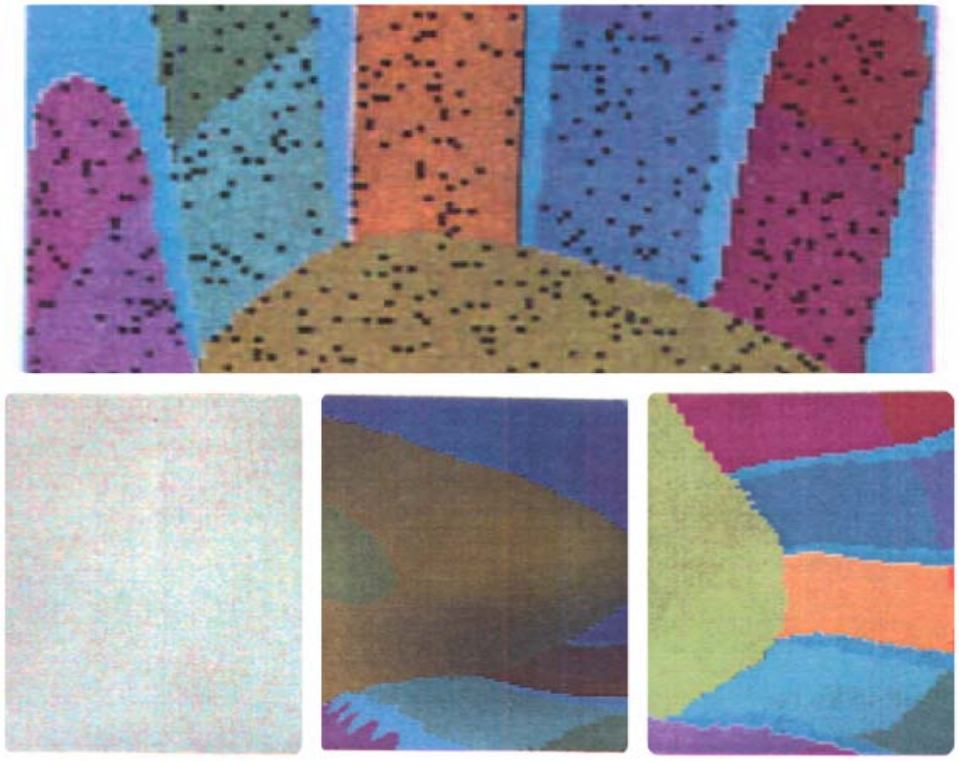

Figura 1.4: Kohonen

Reducción de la Dimensión

Ahora consideraremos una última propiedad de nuestros mapeos autoorganizados, y se trata de su capacidad para reducir la dimensión de los datos, y es de hecho esta característica la que resulta ser en nuestro caso la más importante para nuestro posterior análisis en el capítulo 3.

Cuando se pretende reducir la dimensión de una base de datos multidimensional, una buena estrategia sería identificar (de ser posible) las variables

que si son relevantes y desechar aquellas que no lo son. Por otro lado si se considera que todas las variables son relevantes pero la dimensión de los datos aún es grande, entonces, debe de considerarse, analizar las dependencias que existen entre las variables.

En este caso, si dos variables están fuertemente correlacionadas una mala estrategia sería conservar una de ellas y desechar la otra. En lugar de esto se debe plantear el problema de encontrar un nuevo conjunto de *variables transformadas* que mejor preserven la información contenida en el conjunto inicial. A los mapeos que pretenden transformar las variables observadas en un conjunto mas pequeño que mejor las represente serán llamados *proyecciones*.

Entonces vamos a considerar, con el propósito de exhibir y explicar algunos conceptos, una proyección muy ampliamente conocida denominada *Análisis de Componentes Principales* o (PCA). Esta técnica fue dada a conocer por primera vez por Karl Pearson en 1901 y una de las características de este método es que asume que las dependencias entre las variables observadas son lineales.

Es decir, que los vectores $\mathbf{y} = (y_1, ..., y_n)^T$ de nuestra base de datos son el resultado de aplicar una transformación lineal W a un conjunto de variables por ahora desconocidas, reunidas en el vector $\mathbf{x} = (x_1, ..., x_p)^T$, denominadas *variables latentes*.

$$\mathbf{y} = W\mathbf{x}$$

En general se asume pare este método lo siguiente:

a) Las columnas de W son ortogonales entre ellas y de norma unidad .

b) Las variables latentes tienen una distribución Gaussiana.

c) Tanto las variables observadas $\mathbf{y}$ como las latentes $\mathbf{x}$ deben estar centradas, i.e. $E_{\mathbf{y}}\{\mathbf{y}\} = 0 y E_{\mathbf{x}}\{\mathbf{x}\} = 0$ donde E, es el valor esperado.

En ocasiones es posible que no se cumpla el punto **c**), por lo que se debe entonces *preprocesar* la información, de modo que se debe tomar cada una de nuestras N observaciones de la variable $\mathbf{y}$, e ir restando a cada una de ellas $E_{\mathbf{y}}\{\mathbf{y}\} \neq 0$, es decir:

$$\mathbf{y}(n) - E_{\mathbf{y}}\{\mathbf{y}\} \rightarrow \mathbf{y}(n) \quad \forall n \in N$$

o equivalentemente

$$\mathbf{y}(n) - \frac{1}{N}\sum_{n=1}^{N} \mathbf{y}(n) \rightarrow \mathbf{y}(n) \quad \forall n \in N$$

Ahora, veamos como encontrar nuestra transformación lineal W, y la dimensión $p \leq n$, que garanticen nuestra hipótesis de que $\mathbf{y(n)} = W\mathbf{x} \quad \forall n \in N$, para algún conjunto $\mathbf{x} = (x_1, ..., x_p)^T$ de variables latentes. Entonces empecemos asumiendo que las variables latentes en $\mathbf{x}$ no están interrelacionadas y por el contrario, como hemos dicho antes, que las variables observadas en $\mathbf{y}$ si lo están, esto quiere decir que:

- La matriz de covarianzas $C_{\mathbf{xx}} = E\{\mathbf{x}\mathbf{x}^T\}$ es diagonal.
- La matriz de covarianzas $C_{\mathbf{yy}} = E\{\mathbf{y}\mathbf{y}^T\}$ no es diagonal.

Pese a esto las matrices anteriores se pueden relacionar de la siguiente manera:

$$\begin{aligned} C_{\mathbf{yy}} &= E\{\mathbf{y}\mathbf{y}^T\} \\ &= E\{W\mathbf{x}\mathbf{x}^T W^T\} \\ &= WE\{\mathbf{x}\mathbf{x}^T\}W^T \\ &= WC_{\mathbf{xx}}W^T \end{aligned}$$

Por consiguiente, como hemos supuesto que las columnas de W son ortogonales entre ellas y de norma unidad, entonces se cumple que $WW^T = I_p$, de modo que multiplicando por la izquierda W^T y pro la derecha W a nuestra identidad anterior, obtenemos que:

$$C_{\mathbf{xx}} = W^T C_{\mathbf{yy}} W$$

Entonces, la matriz de covarianzas $C_{\mathbf{yy}}$, por ser una matriz simétrica, puede ser factorizada por el Teorema de descomposición espectral en un producto de matrices $C_{\mathbf{yy}} = V\Lambda V^T$, donde $\Lambda = diag\{\lambda_1, ...\lambda_d\}$ es una matriz diagonal donde $\{\lambda_i\}_{i=1}^d$, es el conjunto de los valores propios de $C_{\mathbf{yy}}$ y $V = (v_1, ..., v_d)$ es una matriz ortogonal, donde $\{v_i\}_{i=1}^d$ es el conjunto de vectores propios normalizados asociados a sus respectivos valores propios $\{\lambda_i\}_{i=1}^d$. Entonces, sustituyendo esta descomposición en nuestra igualdad anterior, obtenemos:

$$C_{\mathbf{xx}} = W^T V \Lambda V^T W$$

Esta igualdad resulta ser cierta sólo cuando los las columnas de W son tomadas colineales con las columnas de V. Además, como la matriz de covarianzas es semidefinida positiva, entonces todos sus valores propios deben ser reales positivos y más aún, si el modelo se respeta íntegramente, es decir, ningún ruido daña las variables observadas, entonces, los primeros p valores propios en Λ son estrictamente positivos y el resto $(d - p)$ deben ser cero. Entonces, debemos mantener los vectores propios asociados a los p valores propios distintos de cero, de modo que si definimos:

$$W = V I_{dxp}$$

Esto, nos conduce a la igualdad:

$$C_{\mathbf{xx}} = I_{dxp} \Lambda I_{dxp}$$

Que nos muestra que los valores propios en Λ se corresponden con las varianzas de las variables latentes. Las cuales pueden ser aproximadas de la siguiente manera:

$$\hat{\mathbf{x}} = I_{pxd} V^T \mathbf{y}$$

Esta ecuación, nos muestra como obtener p coordenadas de las d originales, de modo que se reduzca la dimensión a través de proyectar las variables observadas en el subespacio generado por las variables latentes de forma lineal. Este recorte a la dimensión se logra gracias al factor I_{pxd} que desecha los vectores propios que aparecen en V que ahora llamaremos *componentes*,

que correspondan a los $(d-p)$ valores propios mas pequeños, manteniendo así los llamados *componentes principales*.

Finalmente, podemos afirmar que la solución propuesta bajo este método, garantiza preservar la máxima varianza de las variables observadas, o equivalentemente minimiza el error de reconstrucción (reproduciendo el método usando SVD), o también se puede afirmar que preserva la distancia, estudiando esta técnica, desde el punto de vista del escalamiento multidimensional (MDS). Todos estos criterios pueden ser considerados a la hora de elegir alguno de los métodos disponibles vinculados con la reducción dimensional. Sin embargo hemos de manifestar que nuestros mapeos autoorganizados, tienen en cuenta optimizar un criterio mas poderoso preservar la *topología*.

En general cuando dos variables dependen una de la otra, estas inducen algún tipo de estructura en su distribución, que independientemente de su forma detallada, es su conectividad intrínseca lo que realmente nos interesa de esta ,en este sentido, es precisamente la topología la que se encarga de abstraer dicha conectividad. Naturalmente, se tienen por tanto, que los los objetos de estudio en esta materia son precisamente los llamados *espacios topológicos*.

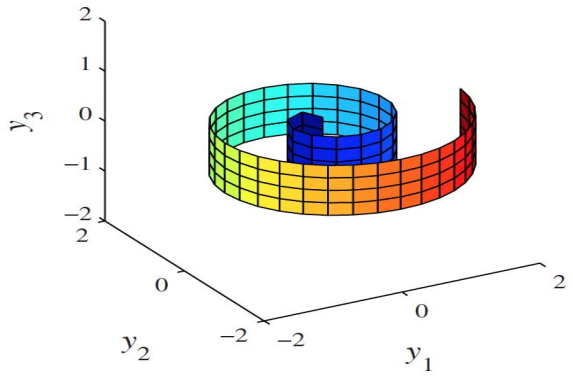

Figura 1.5: El *Swiss Roll* ejemplo de una 2-variedad encajada en $\mathbb{R}^3$

Un caso particular de estos espacios son las llamadas *variedades topológicas* $(\mathcal{M})$, las cuales cumplen con ser localmente euclidianas, es decir, satisfacen que para cada punto $x \in \mathcal{M}$ existe una vecindad $\mathcal{V}_x$ que es topológi-

camente idéntica a una bola unidad $\mathcal{B} \subset \mathbb{R}^d$. Desde un punto de vista mas intuitivo, esto quiere decir, que son localmente (en pequeña escala) planas, así por ejemplo la Tierra es una variedad ya que desde nuestro punto de vista parece ser plana, pero sabemos en realidad que es esférica.

A la vista de la topología, la reducción dimensional es equivalente a *encajar* una variedad $\mathcal{M}$ desde espacios de alta dimensión hacia espacios de baja dimensión, en donde por encaje entenderemos una *representación* de un objeto topológico en cierto espacio, usualmente $\mathbb{R}^d$ que preserva sus propiedades.

Sin embargo, en la práctica, una variedad $\mathcal{M}$ no es más que el *soporte* subyacente de la distribución de los datos que sólo se conoce a través de un conjunto finito, además, si se da el caso en que los datos vienen con ruido o perturbaciones, entonces sucede que estos ya no pertenecen a la variedad (pero es posible que se mantengan cerca). Por lo que podemos decir que la reducción dimensional , en realidad consta de una proyección no lineal de los datos en el encaje de la variedad de baja dimensionalidad.

Considerando el caso particular de la SOM, se ha visto en secciones anteriores, que este toma en cuenta un conjunto de nodos que están conectados entre sí formando una red, la cual pensaremos en esta ocasión como una representación discreta de una topología que en este nuevo contexto jugará el papel del espacio de encaje. Entonces la reducción dimensional sucede de la siguiente manera:

Hemos dicho, anteriormente, que todo nodo o neurona $i \in \Lambda$ ocupa una posición $\vec{r}_i$ dentro de nuestra red bidimensional y precisamos una distancia $d^2_{j,i} = \|\vec{r}_j - \vec{r}_i\|^2$ entre cada par de prototipos, que determinaba la relación de vecindad entre estos. Debemos hacer notar, que en casos mas generales nuestros vectores $\vec{r}_i$ pueden pertenecer un espacio $\mathcal{G} \subset \mathbb{R}^p$, $p > 2$. Por otro lado, también asignamos a cada nodo un vector prototipo $\vec{w}_i \in W \subset \mathbb{R}^d$.

Esto quiere decir que, simultáneamente, se han asignado a cada neurona i de nuestra red dos vectores, uno que pertenece al espacio inicial al cual también pertenecen datos de entrada $\mathbf{y}(n)$ y otro vector que pertenece al espacio final de baja dimensionalidad, los cuales son respectivamente $\vec{w}_i$ y $\vec{r}_i$.

Extrañamente, las coordenadas de $\vec{r}_i$ son conocidas antes de arrancar el algoritmo, mientras que las coordenadas de $\vec{w}_i$ son desconocidas por lo que tienen que ser asignadas al iniciar el algoritmo de manera aleatoria para posteriormente ser calculadas de manera satisfactoria, una vez que esto sucede se puede decir que el encaje del dato $\mathbf{y}(n)$, esta dado como las coordenadas p-dimensionales asociadas con el prototipo mas cercano en el espacio de datos, es decir:

$$\hat{\mathbf{x}} = \vec{r}_{i(\mathbf{y}(n))}$$

donde

$$i(\mathbf{y}(n)) = arg\,\min_j \|\mathbf{y}(n) - \vec{w}_j\|$$

Ahora vamos a considerar un ejemplo que ilustre como trabaja nuestro algoritmo en presencia de una variedad de nombre *Swiss Roll* por su parecido a un pastel del mismo nombre que se enrolla en forma de cilindro. Entonces, las ecuaciones paramétricas que generan dicha variedad son:

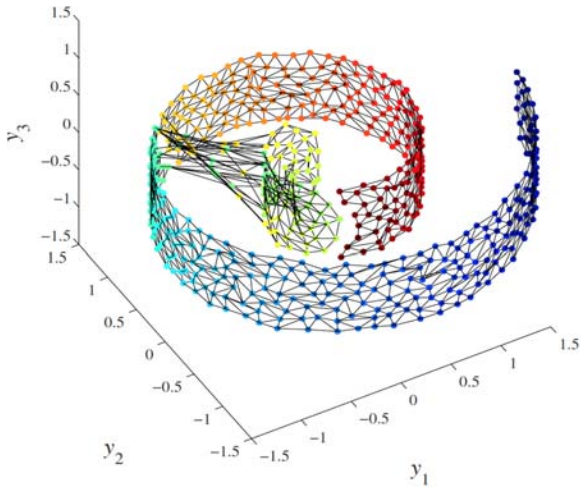

Figura 1.6: Patrón generado por un SOM 8 x 75 del *Swiss Roll*

$$\mathbf{y} = \begin{pmatrix} \sqrt{2+2x_1}cos(2\pi\sqrt{x+2x_1}) \\ \sqrt{2+2x_1}sen(2\pi\sqrt{x+2x_1}) \\ \frac{x_2}{2} \end{pmatrix} \tag{1.1}$$

Donde $\mathbf{x} = (x_1, x_2)^T \subset [-1, 1]^2$.

Como puede verse, las coordenadas sólo dependen de las variables x_1 y x_2, esto quiere decir, que la *dimensionalidad intrínseca* que definimos informalmente como el mínimo número de parámetros o variables latentes necesarias para describir un vector aleatorio $\mathbf{y}$, que en este caso resulta ser 2. Lo anterior puede expresarse diciendo que el *Swiss Roll* es una 2-variedad encajada en $\mathbb{R}^3$ (Figura 1.5).

Los resultados de entrenar una SOM con una red de 8x75, tomando una colección de valores generados por la ecuación paramétrica de la *Swiss Roll* puede apreciarse en la Figura 1.6 .Nuestro método preserva perfectamente la topología hacia el exterior, pero en el interior parece tener algunos problemas al exhibir relaciones que no se encontraban ahí originalmente.

Los errores generados en el ejemplo anterior pueden hacernos pensar que nuestro método no es muy fiable, sin embargo estas fallas no debe preocuparnos demasiado, ya que nuestro mapeo autoorganizado ha demostrado ser muy eficiente en el *análisis de datos* usando su capacidad para reducir la dimensión, como lo muestra el siguiente ejemplo.

Consideraremos entonces un conjunto artificial de datos (Figura 1.7), que consta de un par de agrupamientos ajenos, con dimensionalidad intrínseca 2 generados a partir de los siguientes pseudocódigos:

$ClaseA$

$$x = 0{,}5cos[\pi(-0{,}5 + rand\_unif())] + 0{,}025[rand\_gauss()]$$
$$y = 0{,}5sin[\pi(-0{,}5 + rand\_unif())] + 0{,}025[rand\_gauss()]$$
$$z = sin[2x]cos[2y] + 0{,}025[rand\_gauss()]$$

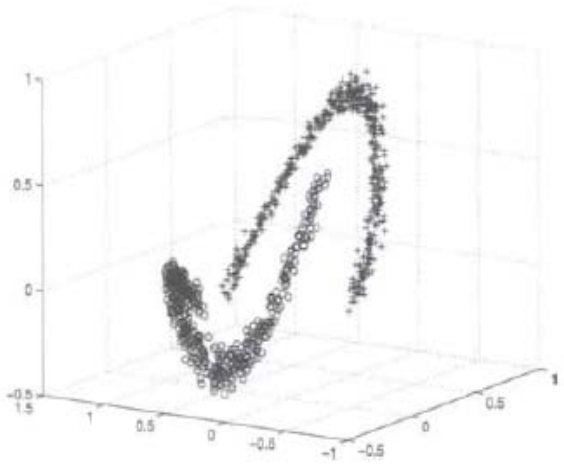

Figura 1.7: Conjunto artificial de datos que representa a dos clases ajenas.

$ClaseB$

$$x = 0{,}25 + 0{,}5cos[\pi(0{,}5 + rand\_unif())] + 0{,}025[rand\_gauss()]$$
$$y = 0{,}5 + 0{,}5sin[\pi(0{,}5 + rand\_unif())] + 0{,}025[rand\_gauss()]$$
$$z = sin[2x]cos[2y] + 0{,}025[rand\_gauss()]$$

Aquí, los resultados de entrenar a la SOM se pueden observar en la Figura 1.8, en donde para obtener una buena visualización de los vectores de pesos sinápticos, se elige mostrar las distancias entre estos mediante un código de escala de grises en lugar de sus respectivos valores. Se debe interpretar entonces que entre mayor sea la distancia, la tonalidad de grises es baja (mas próxima al blanco), mientras que tonalidades altas representan distancias cortas y debido a que cada neurona tiene 8 vecinas sólo se representa la distancia mas grande.

Lo que podemos concluir entonces de lo que se observa en la Figura 1.8 es que existe una separación ente ambos grupos manifestada por la notable frontera blanca que hace posible la división.

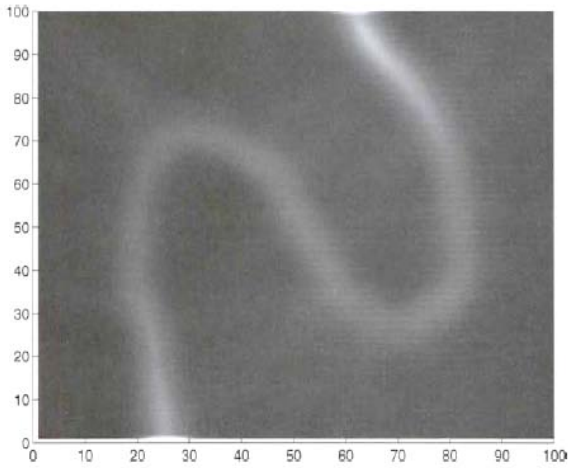

Figura 1.8: Representación 2 dimensional de la base de datos de la Figura 1.7, obtenida al entrenar una SOM y visualizar el resultado mediante un codigo de escala de grises.

Capítulo 2

Fenomenología de los Ácidos Nucleicos

2.1. Bases Nitrogenadas

Empezaremos por considerar en esta capítulo las unidades básicas que conforman tanto el DNA[1] [2] como el RNA, ambas macromoléculas serán grandes polímeros de moléculas orgánicas llamadas *nucleótidos*, es decir, largas cadenas o secuencias de dichas moléculas. Estos nucleótidos a su vez están formados por la unión de moléculas mas pequeñas que son: un *monosacárido* una *base nitrogenada* y un *grupo fosfato* (figuras 2.1 y 2.2) .

Tanto el monosacárido (ribosa en el caso de RNA y desoxirribosa en el caso de DNA) como el grupo fosfato pueden considerarse como unidades constantes en los nucleótidos y en consecuencia unidades constantes a lo largo de las cadenas tanto de DNA como de RNA (unidades estructurales) y las moléculas que puede variar son las bases nitrogenada. Es por esta razón que las cadenas tanto de DNA como de RNA pueden ser descritas sólo en términos de estas bases, las cuales se clasifican de la siguiente manera:

- **Bases purínicas** : Adenina (A), Guanina(G). Ambas presentes tanto en el DNA como en el RNA

[1] El DNA, fue descubierto en 1869 por el químico Friedrich Miescher.

[2] En 1914 Robert Fuelgen, descubre que todos los núcleos de las células tienen la misma cantidad de DNA, a excepción de los gametos que sólo tienen la mitad.

- **Bases pirimidínicas**: Timina (T), Citocina(C), y el uracilo (U). La Timina y la Citosina están presentes en el DNA, y la Citocina y el Uracilo en el RNA .

De esta manera podemos decir que es por medio de estas bases nitrogenadas y en particular por la forma en la que estas están secuencialmente ordenadas dentro de las moléculas de DNA y RNA donde se almacena toda la información hereditaria que le pertenece a cada ser vivo[3]. Así por ejemplo consideremos la siguiente secuencia de bases nitrogenadas escritas cada una de ellas en su forma abreviada en donde sólo aparecen sus iniciales:

GTGCACCTGA CTCCTGAGGA GAAGTCTGCC GTTACTGCCC TGTGGGGCAA GGTGAACGTG
GATGAAGTTG GTGGTGAGGC CCTGGGCAGG CTGCTGGTGG TCTACCCTTG GACCCAGAGG
TTCTTTGAGT CCTTTGGGGA TCTGTCCACT CCTGATGCTG TTATGGGCAA CCCTAAGGTG
AAGGCTCATG GCAAGAAAGT GCTCGGTGCC TTTAGTGATG GCCTGGCTCA CCTGGACAAC
CTCAAGGGCA CCTTTGCCAC ACTGAGTGAG CTGCACTGTG ACAAGCTGCA CGTGGATCCT
GAGAACTTCA GGCTCCTGGG CAACGTGCTG GTCTGTGTGC TGGCCCATCA CTTTGGCAAA
GAATTCACCC CACCAGTGCA GGCTGCCTAT CAGAAAGTGG TGGCTGGTGT GGCTAATGCC
CTGGCCCACA AGTATCAC

Cuadro 2.1: Secuencia de DNA. Fragmento de un gen que codifica para la cadena beta de la hemoglobina.

Este fragmento pertenece a una cadena en el DNA (un *gen*) contenida dentro del cromosoma 11 en humanos, y contiene la información necesaria para construir una pequeña parte conocida como la *cadena beta* de una proteína muy conocida llamada *hemoglobina*, la cual, tiene como función el transporte de oxigeno a todo el cuerpo dentro de la sangre. En realidad se necesita de mucha mas información para construir la proteína en su totalidad.

Una característica muy importante de las bases nitrogenadas es que presentan la propiedad de *complementariedad* [4] entre purinas y pirimidinas, es

[3]La prueba definitiva de que el DNA es el material genético o hereditario fue dada por Alfred Hershey y Margaret Chase en 1952 con su "experimento de la licuadora".

[4]En 1948 Erwin Chargaff y Hotchings, aplican la técnica de cromatografía para revelar que tanto pirimidinas como purinas se encuentran en igualdad de proporciones; es decir, A=T y G=C

decir, forman parejas entre sí. En el caso del DNA la adenina y la timina son complementarias (A=T), al igual que la guanina y la citosina (G=C), y en el caso de RNA dado que no existe timina son complementarias adenina y uracilo (A=U).

El hecho de que las bases nitrogenadas sean complementarias, provoca que dentro de la célula viva el DNA se presente no como una cadena sencilla, si no como una estructura de doble cadena que se enrolla una sobre la otra, generando así una estructura resistente que se conoce como la *doble hélice* [18] . Más aún esta propiedad de complementariedad juega un papel muy importante en el proceso de decodificación de la información contenida en el DNA. Veamos entonces como es que esto sucede.

2.2. Del DNA al RNA: Transcripción

Como se puede apreciar (figuras 2.1 y 2.2), las moléculas tano de DNA Y RNA son muy parecidas entre sí, por lo que el primer paso que da la célula para leer las instrucciones masivamente contenidas en el DNA es hacer una copia de una secuencia de nucleótidos perteneciente al DNA (un gen), en una secuencia de nucleótidos de RNA.

Esto se lleva a cabo gracias a la propiedad antes mencionada de la complementariedad de las bases nitrogenadas y a la ayuda de una enzima muy especial llamada *RNA polimerasa*. Esta enzima es capaz de leer la información contenida en la cadena complementaria de un gen y usarla como molde pare generar una cadena complementaria de RNA que de ahora en adelante llamaremos RNA mensajero o simplemente mRNA.

En este sentido podemos pensar a la RNA polimerasa como una función que va del conjunto de nucleótidos de DNA al conjunto de nucleótidos de RNA, Es decir si consideramos el conjunto de nucleótidos de DNA como $\mathcal{N}_{DNA} = \{A, G, C, T\}$, y al conjunto de nucleótidos de RNA como $\mathcal{N}_{RNA} = \{A, G, C, U\}$ entonces podemos definir la función polimerasa ρ :$\mathcal{N}_{DNA} \rightarrow \mathcal{N}_{RNA}$ como sigue:

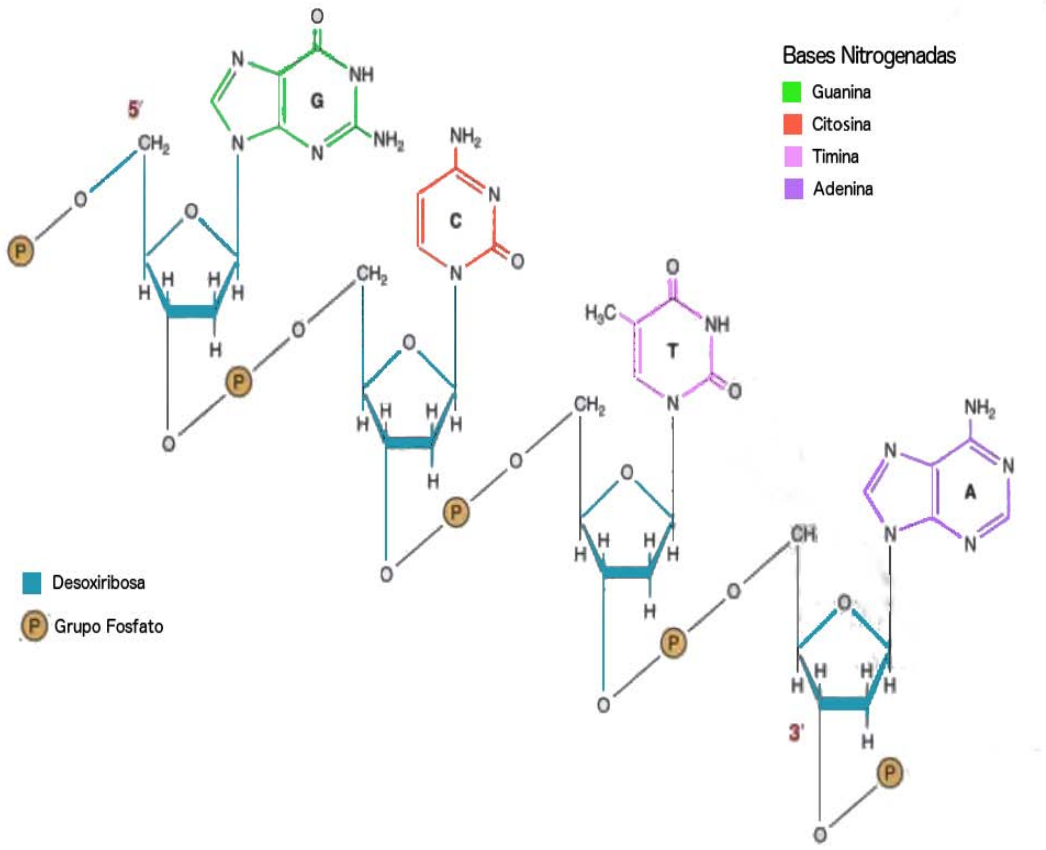

Figura 2.1: Diagrama de DNA, donde se presentan sus unidades estructurales (Desoxiribosa- Grop Fosfato) y sus unidades variables (Guanina, Citosina, Timina, Adenina.)

$$
\begin{aligned}
G &\overset{\rho}{\longmapsto} C \\
C &\overset{\rho}{\longmapsto} G \\
T &\overset{\rho}{\longmapsto} A \\
A &\overset{\rho}{\longmapsto} U
\end{aligned}
$$

Así por ejemplo si consideramos la cadena complementaria del gen de la sección anterior, la cual se encuentra adyacente al mismo gen debido a la naturaleza de doble hélice que presenta el DNA, descrita por:

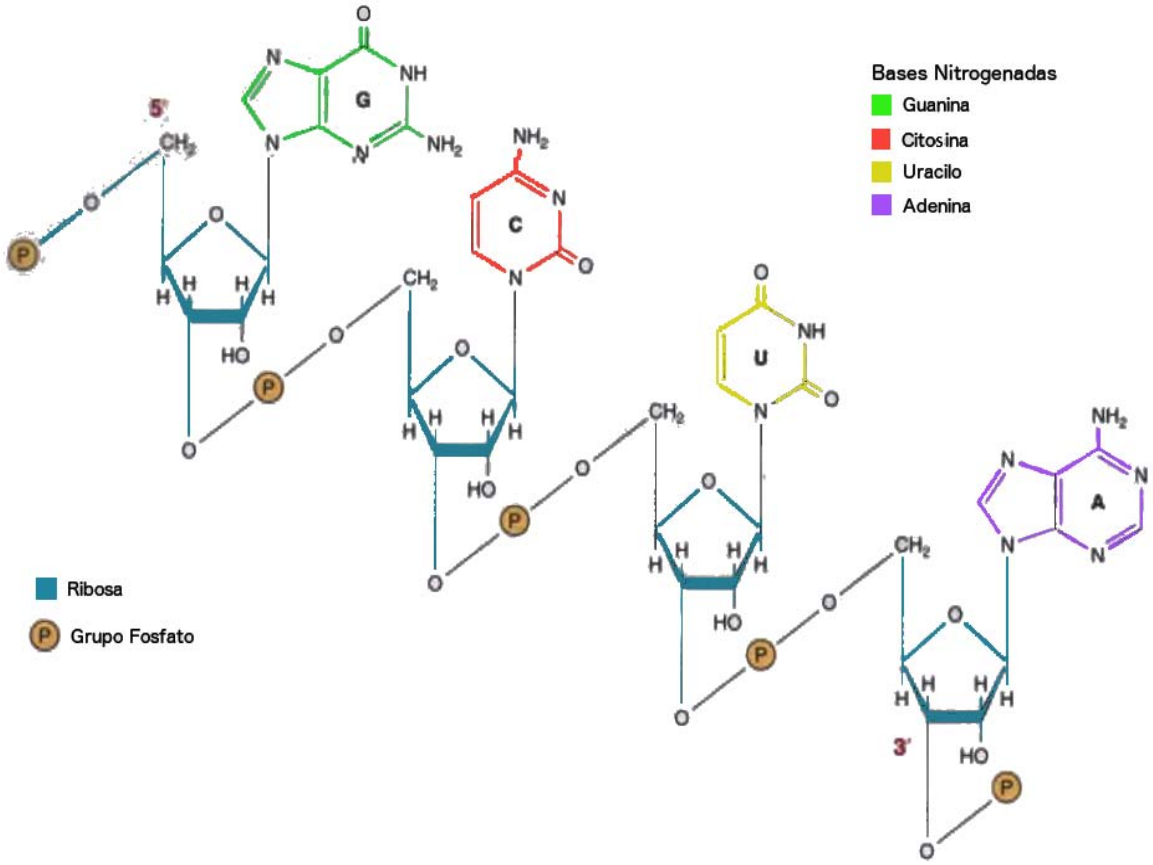

Figura 2.2: Diagrama de ARN, donde se presentan sus unidades estructurales (Ribosa- Grupo Fosfato) y sus unidades variables (Guanina, Citosina, Uracilo, Adenina.)

CACGTGGACT GAGGACTCCT CTTCAGACGG CAATGACGGG ACACCCCGTT CCACTTGCAC
CTACTTCAAC CACCACTCCG GGACCCGTCC GACGACCACC AGATGGGAAC CTGGGTCTCC
AAGAAACTCA GGAAACCCCT AGACAGGTGA GGACTACCAC AATACCCGTT GGGATTCCAC
TTCCGAGTAC CGTTCTTTCA CGAGCCACGG AAATCACTAC CGGACCGAGT GGACCTGTTG
GAGTTCCCGT GGAAACGGTG TGACTCACTC GACGTGACAC TGTTCGACGT GCACCTAGGA
CTCTTGAAGT CCGAGGACCC GTTGCACGAC CAGACACACG ACCGGGTAGT GAAACCGTTT
CTTAAGTGGG GTGGTCACGT CCGACGGATA GTCTTTCACC ACCGACCACA CCGATTACGG
GACCGGGTGT TCATAGTG

Cuadro 2.2: Cadena complementaria del fragmento de gen presentado en el cuadro 2.1, que sirve de molde para generar una copia de este con nucleótidos de RNA, con la ayuda de la polimerasa.

Y aplicamos a esta la función polimerasa, obtendríamos así una secuencia idéntica al gen original, con la excepción de que en lugar de timinas aparecerían uracilos,es decir, conseguiríamos una copia del gen, donde ahora sólo aparecen escritos nucleótidos de RNA. Este transcrito o RNA mensajero (mRNA), puede considerarse entonces, como un intermediario para la síntesis de proteínas, (en nuestro caso sólo un fragmento de esta).

2.3. Aminoácidos y proteínas

Las *proteínas* son otro tipo de macromolécula que también pueden ser consideradas como un polímero de moléculas mas pequeñas llamadas *aminoácidos*. Las proteínas, tienen una gran variedad de funciones y están presentes prácticamente en todos los procesos biológicos, tal es el caso de la RNA polimerasa o la hemoglobina antes mencionadas, o proteínas tales como los anticuerpos (inmunoglobulinas), o la insulina, por mencionar algunas. Veamos entonces algunas propiedades de estos aminoácidos que son la base de nuestras proteínas.

Los aminoácidos como su nombre lo indica tendrán una estructura constante compuesta siempre por un grupo amino ($-NH_2$) y un grupo carboxilo ($-COOH$) y una estructura variable que caracteriza a cada uno de los aminoácidos. En general podemos considerar que existe un total de 20 aminoácidos, que están presentes habitualmente en las proteínas (aminoácidos estándar). Los cuales son los siguientes:

Aminoácido	**Abreviatura 3 letras**	**Abreviatura 1 letra**
Ácido aspártico	Asp	D
Ácido glutámico	Glu	E
Arginina	Arg	R
Lisina	Lys	K
Histidina	His	H
Asparagina	Asn	N
Glutamina	Gln	Q
Serina	Ser	S
Treonina	Thr	T
Tirosina	Tyr	Y
Alanina	Ala	A
Glicina	Gly	G
Valina	Val	V
Leucina	Leu	L
Isoleucina	Ile	I
Prolina	Pro	P
Fenilalanina	Phe	F
Metionina	Met	M
Triptófano	Trp	W
Cisteína	Cys	C

Cuadro 2.3: Los 20 aminoácidos estándar, que son la base de la construcción de las cadenas que conforman a las proteínas.

Las proteínas entonces son largas cadenas de estos aminoácidos cada uno de los cuales está unido a sus vecinos mediante enlaces de tipo covalente peptídico, este enlace se forma cuando el átomo de carbono del grupo carboxilo de un aminoácido comparte electrones con el átomo de nitrógeno del grupo amino de un segundo aminoácido, es por ello que a las proteínas también se les conoce como *polipéptidos* y cada una de ellas posee una secuencia única que las caracteriza.

Entonces para la construcción de esta secuencia única de aminoácidos como hemos dicho anteriormente, debemos hacer uso del mRNA, que de alguna manera contiene esta información, el problema aquí, es que en el mRNA sólo hay cuatro nucleótidos y en una proteína puede haber hasta veinte tipos

distintos de aminoácidos, por lo que la traducción no puede darse mediante una relación uno a uno entre nucleótidos y aminoácidos. Este problema se resolvió en 1961[5] dando origen a lo que se conoce como *código genético*.

Este código nos dice que la secuencia de un mRNA se lee consecutivamente en grupos de tres nucleótidos,es decir, un triplete de nucleótidos dará origen a un aminoácido y dado que sólo son cuatro nucleótidos diferentes entonces se tiene en realidad $4x4x4 = 64$ combinaciones posibles de tres nucleótidos. Esto quiere decir que dado que sólo hay 20 aminoácidos, mas de un triplete nucleótidos dará como resultado el mismo aminoácido. Así el código genético queda como sigue:

Codón	**Aminoácido relacionado**
GCA, GCC, GCG, GCU	A
AGA,AGG,CGA,CGC,CGG,CGU	R
GAC,GAU	D
AAC,AAU	N
UGC,UGU	C
GAA,GAG	E
CAA,CAG	Q
GGA, GGC, GGG,GGU	G
CAC,CAU	H
AUA, AUC, AUU	I
UUA,UUG,CUA,CUC,CUG,CUU	L
AAA,AAG	K
AUG	M
UUC,UUU	F
CCA, CCC,CCG,CCU	P
AGC,AGU,UCA,UCC,UCG,UCU	S
ACA,ACC,ACG,ACU	T
UGG	W
UAC,UAU	Y
GUA,GUC,GUG,GUU	V

[5]El código genético es el resultado del trabajo realizado por Francis Crick y sus colaboradores, quienes en 1961 descubren, utilizando la región rII del bacteriófago T4, la forma en la que es traducido el alfabeto del DNA.

Cuadro 2.4: Código Genético

Ahora que ya hemos esclarecido como es que funciona el código genético podemos utilizarlo para interpretar la información que pose nuestro mRNA obtenido previamente y que es una copia de nuestro gen relativo a la cadena β de la hemoglobina (Cuadro 2.1) para obtener su correspondiente secuencia de aminoácidos, la cual escrita en términos de sus abreviaturas estaría dada por:

VHLTPEEKSA VTALWGKVNV DEVGGEALGR LLVVYPWTQR FFESFGDLST PDAVMGNPKV
KAHGKKVLGA FSDGLAHLDN LKGTFATLSE LHCDKLHVDP ENFRLLGNVL VCVLAHHFGK
EFTPPVQAAY QKVVAGVANA LAHKYH

Cuadro 2.5: Cadena de aminoácidos que resultan de la lectura del mRNA usando el código genético.

Ribosomas

Así como las cadenas de DNA y RNA son sintetizadas por las proteínas de nombre *polimerasa*, las secuencias de aminoácidos son sintetizadas por los *ribosomas*, los cuales son complejos macromoleculares de proteínas y RNA que estructuralmente están caracterizados por tener siempre una subunidad mayor y una menor. La subunidad pequeña controla la lectura de la información almacenada en el mRNA, mientras que la subunidad grande es la responsable de llevar a cabo las reacciones que conducen a la formación del enlace peptídico.

En general los ribosomas de células procariotas se distinguen por tener un *coeficiente de sedimentación* [6] de 70S , mientras que para células eucariontes este valor cambia a 80S. En la siguiente tabla se da un pequeño resumen de estas y otras características que distinguen a estos dos tipos diferentes de ribosomas.

[6] *El coeficiente de sedimentación* nos proporciona una medida del tamaño de una molécula y su unidad es el Svedberg (S), nombrada así en honor al físico y químico sueco Theodor Svedber.

Propiedad	Procariota	Eucariota
Tamaño global	70S	80S
Subunidad pequeña	30S	40S
Número de proteínas	≤ 20	≤ 30
Tamaño RNA	16S	18S
(número de bases)	≈ 1500	≈ 2300
Subunidad grande	50S	60S
Número de proteínas	≤ 34	≤ 50
Tamaño RNA	23S, 5S	28S, 5.8S, 5S
(número de bases)	$\approx 2900(23S)$	$\approx 4200(28S)$
	$\approx 120(5S)$	$\approx 160(5,8S)$
		$\approx 120(5S)$

Cuadro 2.6: Diferencias entre los ribosomas de procariontes y eucariontes.

Entonces, como hemos dicho, vamos a considerar para nuestro estudio, los RNAs ribosomales (rRNAs) y en particular, aquellos que pertenecen a la subunidad pequeña, es decir, aquellos que tienen coeficiente de sedimentación 16S y 18S para diferentes organismos.

Capítulo 3

Caso de estudio, subunidad 16s y 18s de ribosomas.

Consideraremos ahora la *filogenia* de los organismos, es decir, su historia evolutiva, deducida desde el estudio de las secuencias de los ácidos nucleicos. En la actualidad es claro que ciertas macromoléculas funcionan como *cronómetros evolutivos*, que nos ayudan a entender las distancias evolutivas entre los diferentes organismos, tal es el caso de la subunidad beta de las ATPasas[1], algunos genes funcionales y los RNAs ribosómicos.

En el presente capítulo tomaremos en cuenta el caso particular de los genes que codifican para los RNAs ribosomales de distintos seres vivos, de manera mas especifica aquellos genes que codifican para los rRNA de la subunidad 16S y 18S con el objetivo de demostrar que los mapeos autoorganizados son capaces de distinguir entre los tres principales dominios: Bacteria, Archaea y Eukarya, dadas algunas pocas secuencias llamadas *consenso*.

Para este estudio se eligieron las secuencias de un total de 28 organismos (Cuadro 3.2); de los cuales dentro del dominio Bacteria se encuentran representados algunos patógenos humanos, productores de antibióticos, probióticos e incluso bioluminiscentes como *Photobacterium phosphoreum*, entre otros. En el caso del dominio Archaea podemos encontrar especies de tipo hipertermófilos, anaerobios, thermoacidófilos, metanógenos, oxi-

[1]Las ATPasas son enzimas (moléculas de naturaleza proteica) que son capaces de producir la hidrólisis del ATP, principal fuente de energía para la mayoría de las funciones celulares.

dantes del azufre, y algunos otros como *Thermococcus gammatolerans* que pueden soportar los rayos gamma. Finalmente en el dominio Eukaryota tenemos algunos hongos, algas, protozoos, nemátodos, etc., así como también animales microscópicos como los *tardígrados* capaces de vivir en el vacío.

3.1. Alineamiento múltiple de secuencias

Con el propósito de hacer un primer análisis y encontrar las secuencias consenso, se realizó un *alineamiento múltiple*, el cual es un recurso utilizado para comparar y encontrar las similitudes de mas de tres secuencias. A continuación definiremos con precisión que se entiende por alineamiento múltiple y describiremos algunos de sus métodos. Comenzaremos con la siguiente definición:

1. Sea $\mathscr{A} = \{a_1, a_2, ..., a_l\}$ un *alfabeto* de l símbolos. Entonces:

$$\mathfrak{s} = b_1 b_2 ... b_n$$

es una *secuencia* de $\mathscr{A}$ si $b_i \in \mathscr{A}$ para toda i=1,2,...n. Donde n es la longitud de $\mathfrak{s}$

Es decir una secuencia es una cadena ordenada de elementos llamados símbolos o letras de un alfabeto los cuales se representan por una simple concatenación de estos elementos. Así por ejemplo las cadenas de DNA, RNA y proteína presentadas en el capitulo 2 (y cualquier otra que se nos pueda ocurrir), tienen como alfabeto $\mathscr{A}_D = \{A, G, C, T\}$ $\mathscr{A}_R = \{A, G, C, U\}$ $\mathscr{A}_P = \{D, E, R, K, H, N, Q, ..., M, W, C\}$(Abreviaturas 1 letra Cuadro 2.3) respectivamente y cumplen con la definición aquí expresada.

Ahora consideremos un conjunto de N (N ≥ 3) secuencias $\mathfrak{S} = \{\mathfrak{s}_1, \mathfrak{s}_2, ..., \mathfrak{s}_N\}$, de un alfabeto $\mathscr{A}$ tal que:

$$\mathfrak{s}_1 = b_{1_1} b_{1_2} ... b_{1_{n_1}}$$

$$\mathfrak{s}_2 = b_{2_1} b_{2_2} .... b_{2_{n_2}}$$

$$\vdots$$

$$\mathfrak{s}_N = b_{N_1} b_{N_2} .... b_{N_{n_N}}$$

Con $b_{i_j} \in \mathscr{A}$ para todo $j = 1, 2, ..., n_i$ y para todo $i = 1, 2, ..., N$. Entonces tenemos la siguiente definición:

2. Un *alineamiento múltiple* de N secuencias, es una matriz M_{NxL} donde $max(n_i) \leq L \leq \sum_{i=1}^{N} n_i$, de tal manera que el i-ésimo renglón se obtiene a partir de la secuencia $\mathfrak{s}_i$ agregando un número finito de espacios vacíos (-) de tal manera que $\mathbf{card}(\mathfrak{s}_i) + \mathbf{card}(-) = L$.

Ahora bien, el objetivo principal de los diferentes métodos computacionales para encontrar alineamientos múltiples es encontrar aquellos que sean óptimos, es decir, aquellos que maximicen el número de coincidencias entre los elementos contenidos en la j-ésima columna de la matriz M. En general se han desarrollado una gran cantidad de métodos para resolver este tipo de problemas, los cuales se clasificar como *exhaustivos* o *aproximados*.

Los procedimientos exhaustivos, se basan en programación dinámica y en técnicas de ramificación y acotamiento, mientras que los aproximados se dividen a su vez en *progresivos*(también conocidos como jerárquicos o de árbol) e *iterativos*. Los métodos progresivos se caracterizan por alinear las secuencias mas parecidas entre si, para después ir incorporando el resto de las cadenas por similitud. Por su parte los procedimientos iterativos utilizan un alineamiento inicial, el cual, se va perfeccionando a través de una serie de ciclos (iteraciones), hasta un punto en el cual ya no es posible realizar una mejora.

En el presente estudio se utilizo la técnica de alineamiento múltiple conocida como MUSCLE (de *Multiple Sequence Alignment by Log-Espectation*) que pertenece al grupo de los algoritmos iterativos. Gracias a este alineamiento fue posible encontrar las diferentes secuencias consenso (Cuadro 3.1), las cuales definen a los diferentes dominios. Se observó que no en todos los casos las secuencias de rRNA de cada organismo coincide con la secuencia consenso, por lo que se considero una *medida de semejanza* entre dichas secuencias tomando en cuenta:

Secuencia consenso	**Posición**
TGAAACTTAAAGG	1253-1266
YTYAATTG	1305-1312
CAACCYYCR	1488-1496
TCCCTG	1790-1795
TACACACCG	1803-1811

Cuadro 3.1: Secuencias Consenso altamente conservadas en los genes que codifican para los rRNA 16S y 18.

1. La longitud de la secuencia consenso (i.e cantidad bases) que denotaremos como $\ell(c_i)$ donde $c_i \in \mathcal{S}$ "El conjunto de todas las secuencias consenso", con $i \in \{1, 2, 3, 4, 5\}$.

2. Número de bases de la secuencia de rRNA que coinciden con la secuencia consenso que denotaremos como $\iota(r_{ij})$ donde $r_{ij} \in \mathcal{U}$ "El conjunto de secuencias del rRNA que son símil de la secuencia consenso c_i", con $i \in \{1, 2, ..., 5\}$,$j \in \{1, 2, ..., 28\}$.

Entonces la medida de semejanza que denotaremos como $\ominus$ queda caracterizada por todos los cocientes de la forma:

$$\ominus(c_i, r_{ij}) = \frac{\iota(r_{ij})}{\ell(c_i)}$$

Con esta medida hemos podido construir una base de datos con un total de 28 vectores en $\mathbb{R}^5$ donde cada uno de sus componentes es el valor generado con la expresión aquí dada de semejanza. (Cuadro 3.2). Lo que sigue es ver como el algoritmo de Kohonen, utilizando estos vectores como datos de entrada, fue capaz de distinguir los diferentes dominios. Pero antes de presentar los resultados veamos en detalle las técnicas de visualización y clasificación utilizadas para interpretar dicho algoritmo.

3.2. Visualización y Clasificación

Consideraremos en primer lugar la técnica de visualización conocida como *U-matrix*. Este método fue desarrollado para visualizar estructuras del espacio N-dimensional, el cual, es capaz de detectar la frontera entre diferentes

subconjuntos de los datos de entrada, permitiéndonos así descubrir los diferentes agrupamientos o *clusters* generados por la red neuronal.

La U-matrix se construye considerando para cada neurona η de la capa bidimensional en nuestro algoritmo, sus neuronas vecinas, entonces, denotemos al conjunto de neuronas vecinas como $NV(\eta)$ y consideremos $\vec{w}(\eta)$ el vector de peso sináptico asociado a la neurona η. Ahora definimos para cada neurona la altura como sigue:

$$U\text{altura}(\eta) = \sum_{m \in NV(\eta)} d(\vec{w}(\eta) - \vec{w}(m))$$

Aquí $d(x, y)$ representa la distancia usada por nuestro algoritmo de mapeos autoorganizados. De esta manera la U-matrix exhibe las U-alturas por encima de la red en las posiciones de cada neurona, en la cual las U-alturas son codificadas de acuerdo a un gradiente de colores. Así por ejemplo U-alturas pequeñas son codificadas en tonos claros y U-alturas grandes son codificadas en tonos obscuros.

Agrupamiento jerárquico

Ahora consideraremos otro método que pertenece a la clase de los agrupamientos jerárquicos, de nombre *Unweighted Pair Group Method with Arithmetic Mean* ó UPGMA, el cual, nos permitirá mediante la construcción de un dendrograma visualizar los diferentes cúmulos (*clusters*) generados por nuestra base de datos a diferentes niveles. El esquema general de estos procedimientos se describe a continuación:

Dado un conjunto de N objetos de estudio debemos considerar una distancia d con la propiedad de que $\forall x, y, z \in N \quad d(x, z) \leq max[d(x, y), d(y, z)]$ (desigualdad ultramétrica), con la cual, debe elaborarse inicialmente una *matriz de distancias* o *similitudes* de $N \times N$:

$$\begin{pmatrix} d(x_1, x_1) & d(x_1, x_2) & \cdots & d(x_1, x_N) \\ d(x_2, x_1) & d(x_2, x_2) & \cdots & d(x_2, x_N) \\ \vdots & & \ddots & \vdots \\ d(x_N, x_1) & \cdots & & d(x_N, x_N) \end{pmatrix}$$

Aquí debe notares que dado que $d(x, y)$ es una distancia esta matriz es simétrica y con diagonal idénticamente cero. Entonces deben considerarse los siguientes pasos:

Paso 1. Agrupamiento C_0. Se debe tomar como primer agrupamiento aquel cuyos únicos elementos sean aquellos que sólo tengan uno de los elementos del conjunto N y su matriz de similitudes como la descrita en el párrafo anterior. A este agrupamiento se le asigna el valor 0.

Paso 2. Dado el agrupamiento C_{i-1} y su respectiva matriz de similitudes, deben juntarse aquellos elementos en C_{i-1} que correspondan al valor más pequeño $\alpha_i \neq 0$ de las entradas de la matriz formado un nuevo elemento dentro del agrupamiento C_i. A este nuevo agrupamiento se le asignara el valor α_i

Paso 3. Construir una nueva matriz de similitudes para el agrupamiento C_i

Paso 4. Repetir los pasos 2 y 3 hasta obtener un agrupamiento C_j cuyo único elemento sea aquel que contenga todos los elementos del conjunto N. En ese caso hemos concluido.

En los pasos recién descritos, no se especifica para el tercero cómo es que debe construirse la nueva matriz de similitudes, lo cual, puede hacerse de diferentes maneras y esto es lo que distingue a cada uno de los distintos métodos que pertenecen a la clase de los agrupamientos jerárquicos. Entonces para cada caso se debe considerar lo siguiente:

Supongamos que existen dos elementos $x, y \in C_{i-1}$ que satisfacen la condición de que la distancia entre ellos cumple ser la mas pequeña dentro del conjunto de todas las distancias representadas en la matriz de similitudes, entonces, de acuerdo al segundo paso estos deben juntarse para formar un nuevo elemento [x,y] dentro del agrupamiento C_i, entonces, ahora debe decidirse como será $d([x, y], z) \quad \forall z \in C_i$, donde, en el caso del método UPGMA se considera que:

$$d([x, y], z) = \frac{1}{(|x| + |y|)|z|} \sum_{i=1}^{|x|+|y|} \sum_{j=1}^{|z|} d(w_i, w_j)$$

Donde $|\cdot|$ denota la cardinalidad y $w_i \in [x, y]$ y $w_j \in z$. Con esta nueva información y la que proviene de la matriz de similitudes del agrupamiento precedente, podemos ahora construir nuestra nueva matriz que contendrá todas las distancias entre los elementos del recién formado agrupamiento C_i.
En lo que sigue mostraremos y discutiremos los resultados obtenidos de los métodos U-matrix y de agrupamiento jerárquico generados dentro del entorno de Matlab. Los alineamientos múltiples de secuencias son presentados por separado al final de este capítulo.

3.3. Resultados

Se consideró el uso del paquete *SOM Toolbox* de Matlab, para poder ejecutar el algoritmo de Kohonen, considerando como datos de entrenamiento a los vectores de la base de datos descrita en párrafos anteriores. Los detalles de la base de datos se presenta en los anexos.

Una vez entrenada la red, se obtuvo su correspondiente U-matrix (Figura 3.1), con un gradiente de colores que van desde un azul oscuro para valores pequeños hasta el rojo para valores grandes de las U-alturas. Además con el propósito de distinguir para cada elemento de la base de datos a que agrupamiento pertenecen se hicieron visibles las etiquetas con el nombre de cada uno de los organismos.

De inmediato se puede notar que existen al menos 3 agrupamientos, el primero se encuentra en la esquina superior izquierda, el segundo en la esquina superior derecha y un tercero en la base. Aquí se pone de manifiesto, al observar las etiquetas, que se trata de los tres principales dominios Archaea, Eukarya y Bacteria, reafirmado lo ya dicho por Carl R. Woese hacia 1977 "cada sistema viviente representa a uno de tres lineas de descendencia" proponiendo años mas tarde (1990) los tres dominios aquí ya mencionados.

Un análisis mas detallado nos revela que las fronteras entre los agrupamientos reflejan sus distancias evolutivas, se puede notar por ejemplo que el dominio Eukarya presenta una relación mas próxima al dominio Archaea que al dominio Bacteria. Sin embargo también puede apreciarse una muy marcada distancia entre el dominio Archaea y el dominio Bacteria señalada por la notable franja rojiza presente a mitad de la imagen.

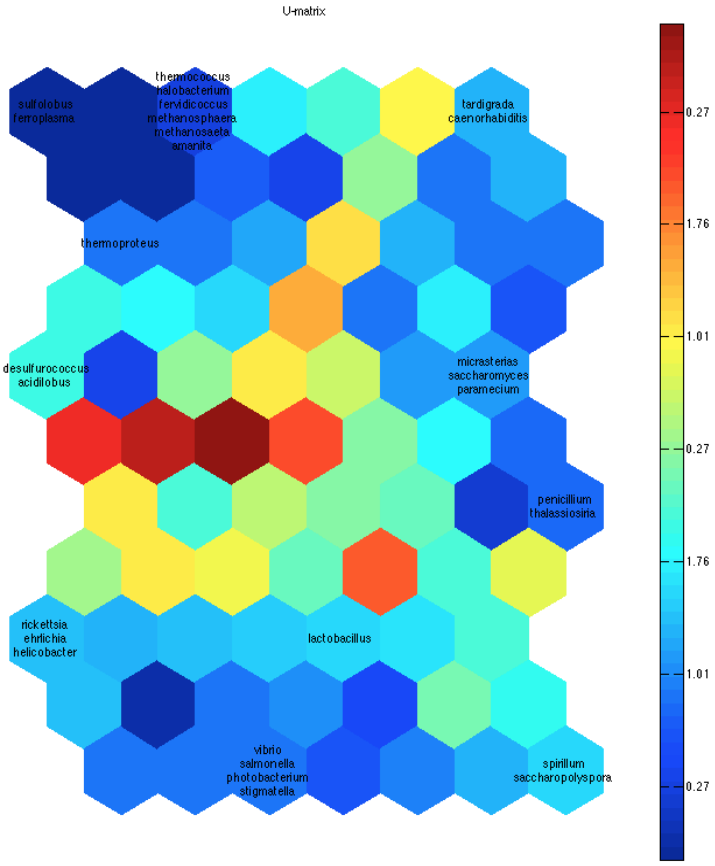

Figura 3.1: U matrix

Más sorprendente aún es el hecho de que algunos *Phylum* y *Clases* se agruparon dentro de una misma celda o muy cerca de ella. Tal es el caso de de los organismos *Vibrio*, *Salmonella* y *Photobacterium* que pertenecen a la Clase *Gammaproteobacteria*, así como también *Rickettsia* y *Ehrlichia* que son parte de la Clase *Alphaproteobacteria*, todos estos dentro del dominio Bacteria.

De la misma manera sucede con las especies del dominio Archaea, *Thermococcus*, *Halobacterium*, *Methanosphaera*, *Methanosaeta* y *Ferroplasma* que pertenecen al Phylum *Euryarchaeota*. Aquí, cabe aclarar que los tres primeros organismos de esta categoría si se encuentran dentro de la misma celda

mientras que *Ferroplasma* se encuentra en una celda vecina. Y finalmente en el dominio Eukaryota, al parecer, las especies que se encuentran juntas, no parece que tengan relación alguna desde el punto de vista taxonómico.

Finalmente y con el propósito de convalidar los resultados anteriores se consideró la construcción de un dendrograma (Figura 3.2), basado, en la información generada por el método UPGMA , descrito en párrafos anteriores, a partir de nuestra base de datos. En el puede observarse mediante el corte generado (línea roja) la presencia de nuestros tres principales dominios, incluidas además, sus respectivas distancias evolutivas.

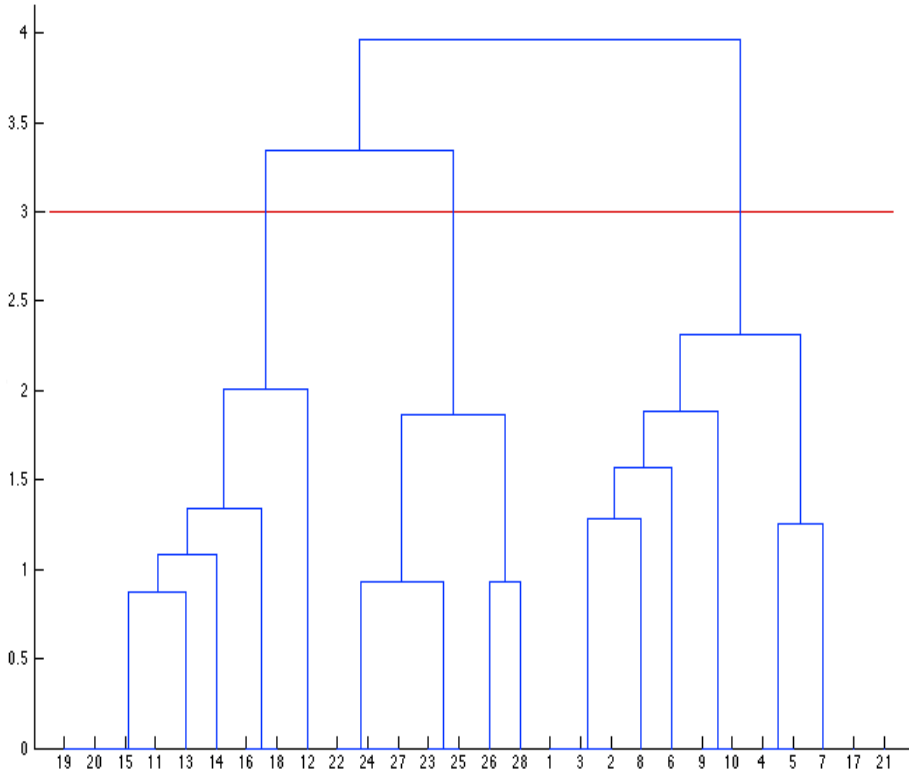

Figura 3.2: Dendrograma. Los números corresponden a los organismos según el orden establecido en la base de datos.

Sin embargo, a diferencia del análisis realizado con la U-matrix, en el dendograma no se pueden percibir las agrupaciones observadas en termino de

phylum y clase antes mencionadas y de ningún otro tipo. Por lo que, comparando ambos resultados, es evidente, que existe un mejor desempeño de nuestro algoritmo de mapeos autoorganizados.

Continuidad y resolución

Ahora, considerando un punto de vista mas teórico que comparativo, ¿Qué tan "bueno" resulta un mapa generado por nuestros mapeos autoorganizados?, Este es un aspecto que tal vez pueda ser evaluado usando los siguientes criterios :

- ¿En que grado el mapa es continuo?.
- ¿Cual es la resolución del mapa?.

En donde, *continuidad en el mapa* quiere decir que vectores de entrada cercanos son mapeados en vectores cercanos en el espacio de salida, mientras que por *buena resolución* entenderemos que no hay vectores que se encuentren distantes en el espacio de entrada que sean mapeados cerca en el espacio de salida. Se pretende entonces,conseguir un balance entre estas dos características, para tener en términos generales un mapeo "correcto", de modo que cuantificarlas resulta ser una tarea esencial.

Calcular la resolución se logra fácilmente, esto se hace usando la medida de *error de cuantización*, que esencialmente es la distancia promedio entre cada vector de datos y su BMU:

$$qe = \frac{1}{N} \sum ||\vec{x}_i - m_{\vec{x}_i}||$$

Por el contrario el calculo de la continuidad puede ser algo mas intrincado y en general, existen diversas formas de medirla (referencias.), por lo que en este texto sólo se considerarán dos métodos, basados cada uno en enfoques diferentes. El primero de ellos se apoya en el libro de códigos (*codebook*) y sus respectivos nodos, mientras que el segundo considera el espacio de entrada y la red del mapa. Estos son los métodos:

Producto topográfico. Primero definamos el vecino mas cercano k-ésimo del nodo j medido en el espacio de salida A como:

$$n_1^A(j) := \underset{j' \in A \setminus \{j\}}{argmin} \; d^A(j, j')$$

$$n_2^A(j) := \underset{j' \in A \setminus \{j, n_1^A(j)\}}{argmin} \; d^A(j, j')$$

$$\vdots$$

Y respectivamente, el vecino k-ésimo mas cercano del nodo j medido en el espacio de entrada V como:

$$n_1^V(j) := \underset{j' \in A \setminus \{j\}}{argmin} \; d^V(w_j, w'_j)$$

$$n_2^V(j) := \underset{j' \in A \setminus \{j, n_1^V(j)\}}{argmin} \; d^V(w_j, w'_j)$$

$$\vdots$$

Entonces, usando esta notación, definamos los siguientes cocientes:

$$Q_1(j,k) = \frac{d^V(w_j, w_{n_k^A(j)})}{d^V(w_j, w_{n_k^V(j)})}$$

$$Q_2(j,k) = \frac{d^A(j, n_k^A(j))}{d^A(j, n_k^V(j))}.$$

Para estas definiciones se tiene que,$Q_1(j,k) = Q_2(j,k) = 1$, sólo si los vecinos mas cercanos de orden k en el espacio de salida y en el espacio de entrada coinciden. Así cualquier desviación de Q_1 Q_2 de 1 apunta a un fallo en el ordenamiento de los vecinos mas cercanos debido al mapa. Estas ecuaciones puede considerarse como una medida de la continuidad, sin embargo resulta ser demasiado restrictivas, por lo que consideremos en su lugar:

$$P_1(j,k) = \left(\prod_{l=1}^{k} Q_1(j,l) \right)^{\frac{1}{k}}$$

$$P_2(j,k) = \left(\prod_{l=1}^{k} Q_2(j,l) \right)^{\frac{1}{k}}$$

Aquí, P_1 y P_2, pueden ser consideradas como una estimación de la distorsión de los primeros k nodos alrededor de j en el espacio de entrada y de salida respectivamente, además, si los primeros k-vecinos de j se mantienen cercanos en ambos espacios, entonces se cumple que $P_1 = P_2 = 1$, y en otro caso $P_1 \neq P_2$ con $(P_1, P_2) \neq 1$ .En general, siempre se cumplen las dos siguientes desigualdades:

$$P_1(j,k) \geq 1$$
$$P_2(j,k) \leq 1$$

Por lo tanto combinando ambas medidas en una sola obtenemos:

$$P_3(j,k) = \left(\prod_{l=1}^{k} Q_1(j,l) Q_2(j,l) \right)^{\frac{1}{2k}}$$

Este último paso, tiene el efecto, dada la naturaleza inversa de P_1 y P_2, de evitar que la medida se vea afectada debido las curvaturas, mientras que su capacidad de medir la continuidad se mantiene, apuntando a un error cuando $P_3 \neq 1$, apuntando a una dimensión del espacio de salida A muy grande cuando $P_3 > 1$ y muy pequeña cuando $P_3 < 1$. Finalmente considerando todos los nodos y todos los posibles k ordenes para la medida P_3 llegamos a la fórmula del producto topográfico dada por:

$$P = \frac{1}{N(N-1)} \sum_{j=1}^{N} \sum_{k=1}^{N-1} log(P_3(j,k)).$$

Aquí se toma el logaritmo ya que sólo nos interesan las desviaciones de la unidad.

Error topográfico. Esta medida es probablemente la más sencilla para cuantificar como se preserva la topología o la continuidad de un mapeo autooganizado y se calcula simplemente tomando cada uno de los vectores $\vec{x_k}$ $k \in \{1, ...N\}$ del espacio de entrada y se determina para cada uno de estos su primer y segundo **BMU**, si estos no son adyacentes, entonces, se considera como un error, finalmente se cuentan todos los errores y se normaliza el resultado en un rango de cero a uno, donde 0 quiere decir que la topología se ha preservado perfectamente.

$$te = \frac{1}{N}\sum_{k=1}^{N} u(\vec{x_k})$$

donde

$$u(\vec{x_k}) = \begin{cases} 1 & \text{si el 1º y 2º BMU no son adyacentes} \\ 0 & \text{en otro caso} \end{cases}$$

Entonces, en relación al mapeo presentado como resultado en la sección anterior el *SOM Toolbox* de Matlab proporcionó los siguientes valores:

error de cuantización= 0.581
error topográfico=0.0

Por lo que la conclusión evidente de este resultado es que se obtuvo una resolución regular, pero la topología se preserva perfectamente.

Discusión y Conclusiones

Este trabajo fue realizado con el propósito de exhibir a grandes rasgos dos enfoques que a juicio del autor se manifiestan siempre que se habla de redes neuronales artificiales. El primero de ellos es el hecho de que dichos algoritmos buscan imitar las redes neuronales biológicas y el segundo, es el uso práctico que se les pude dar en diferentes áreas del conocimiento.

Al respecto del primer planteamiento se hizo un análisis detallado del modelo de Mapeo Autoorganizado de Kohonen, red neuronal artificial de aprendizaje no supervisado capaz de emular la formación de mapas somatotópicos de la corteza cerebral, además de otras propiedades como la compresión de datos.

Además, poniendo en contexto el modelo de Kohonen, se hace un breve resumen de otros tipos de redes neuronales artificiales, poniendo principal atención en su capacidades de aprendizaje, comparándolos directamente con el tipo de aprendizaje de los sistemas biológicos.

Aquí, vale la pena mencionar, con el propósito de ampliar el panorama, los esfuerzos realizados por otros científicos en este tema. Tal es el caso de las redes neuronales de *memoria asociativa*, que tuvieron su origen en 1961 con la *Lernmatrix* creada por Karl W. Steinbuch. Estas redes neuronales tienen el propósito de recordar patrones completos a partir de patrones de entrada, tal como lo hacemos por ejemplo cuando nos acordamos de una canción con tan sólo oír un fragmento de esta.

Entonces podemos decir que una memoria asociativa puede formularse como un sistema entrada-salida, donde un patrón de entrada representado por un vector $\vec{x}$ se le debe relacionar con un patrón de salida $\vec{y}$, es decir, cada uno de

los patrones de entrada deben formar una *asociación* con su correspondiente patrón de salida $\vec{y}$, que denotaremos como una pareja ordenada $(\vec{x}, \vec{y})$

De esta manera, si consideramos un conjunto de asociaciones $F = \{(\vec{x_i}, \vec{y_i}) | i = 1, ..., p\}$ que llamaremos *conjunto fundamental* , el problema general de las memorias asociativas consiste en encontrar una manera de generar una matriz M y un grupo de operadores adecuados, de tal manera que dicha matriz sea capaz de almacenar las p asociaciones del conjunto fundamental, es decir, construir una memoria M, que cuando se opera con un patrón fundamental $\vec{x_i}$ de como salida el vector *vecy*.

Otra red neuronal que también consideramos importante mencionar, en virtud de nuestro primer planteamiento, son las *redes ART*, siglas en inglés de *Teoría de Resonancia Adaptativa*, desarrolladas por Stephen Grossberg y Gail Carpenter. El problema que llevó al desarrollo de este tipo de procedimientos se le conoce como *dilema de la estabilidad y plasticidad del aprendizaje*, el cual, surge de la necesidad de tener un sistema capaz de tener simultáneamente las siguientes propiedades:

1) Poder aprender nuevos patrones, es decir, tener *plasticidad de aprendizaje*.

2) Poder recordar los patrones ya aprendidos, es decir, ser *estable*.

Implementar juntas estas dos características puede ser algo difícil de conseguir, ya que al aprender nuevos patrones, se corre el riesgo de perder los ya memorizados.

En general las redes de este tipo buscan crear agrupamientos o clusters. La red debe de decidir, según el dato que se le introduzca, si pertenece a una categoría ya definida o bien se debe crear una nueva tomando a dicho dato como patrón modelo, esto, dependiendo si existe o no cierto grado de similitud con los grupos hasta ese momento vigentes.

Algunas características relevantes de las redes tipo ART, es que poseen una memoria de corto y largo plazo, además, de que son de aprendizaje competitivo y no supervisado.

En relación al enfoque práctico se realizó un estudio que consistió en el uso de estos mapeos autoorganizados para la clasificación de diferentes organismos usando las secuencias genéticas de sus RNAs ribosómicos, dando como resultado la separación de estos en sus diferentes dominios: Eukarya, Archaea y Bacteria.

En torno a este estudio, se dedica todo un apartado para describir procesos celulares como la transcripción y la traducción dejando de manifiesto la importancia de los genes y en particular aquellos que son relevantes para nuestro análisis, los que codifican para la subunidad pequeña del ribosoma.

En la actualidad el tema de las redes neuronales artificiales es ampliamente conocido y han evolucionado a tal grado que han comenzado a ser parte de nuestra vida cotidiana.

Cuadro 3.2: Base de datos

Especie	Oligonucleótido				
	TGAAACTTAAAGG	YTYAATTG	CAACCYYCR	TCCCTG	TACACACCG
BACTERIA					
1 Vibrio cholerae	0.769	0.875	0.889	0.667	1
2 Salmonella enterica	0.769	0.875	0.889	0.667	1
3 Photobacterium phosphoreum	0.769	0.875	0.889	0.667	1
4 Rickettsia rickettsii	0.846	0.875	1	0.5	0.889
5 Ehrlichia chaffeensis	0.846	0.875	1	0.5	0.889
6 Lactobacillus acidophilus	0.923	0.875	0.889	0.667	1
7 Helicobacter pylori	0.846	0.875	0.889	0.667	0.889
8 Stigmatella aurantiaca	0.846	0.875	1	0.667	1
9 Spirillum volutans	0.846	0.75	0.889	0.667	1
10 Saccharopolyspora erythraea	0.846	0.75	0.889	0.667	1
ARCHAEA					
11 Thermococcus gammatolerans	1	1	0.667	1	0.889
12 Sulfolobus metallicus	1	1	0.667	1	0.778
13 Thermoproteus neutrophilus	1	1	0.778	1	0.889
14 Ferroplasma acidiphilum	1	1	0.667	0.833	0.889
15 Halobacterium salinarum	1	1	0.667	1	0.889
16 Desulfurococcus fermentans	0.923	1	0.778	1	0.889
17 Fervidicoccus fontis	1	1	0.667	N/A	N/A
18 Acidilobus saccharovorans	0.923	1	0.778	1	0.889
19 Methanosphaera stadtmanae	1	1	0.667	1	0.889
20 Methanosaeta concilii	1	1	0.667	1	0.889
EUKARYOTA					
21 Amanita muscaria	1	N/A	N/A	N/A	N/A
22 Micrasterias radians	1	0.875	0.778	1	1
23 Penicillium chrysogenum	0.923	0.875	0.778	1	1
24 Saccharomyces cerevisiae	1	0.875	0.778	1	1
25 Thalassiosira pseudonnona	0.923	0.875	0.778	1	1
26 Tardigrada environmental	1	0.875	0.556	1	1
27 Paramecium bursaria	1	0.875	0.778	1	1
28 Caenorhabditis elegans	0.923	0.875	0.556	1	1

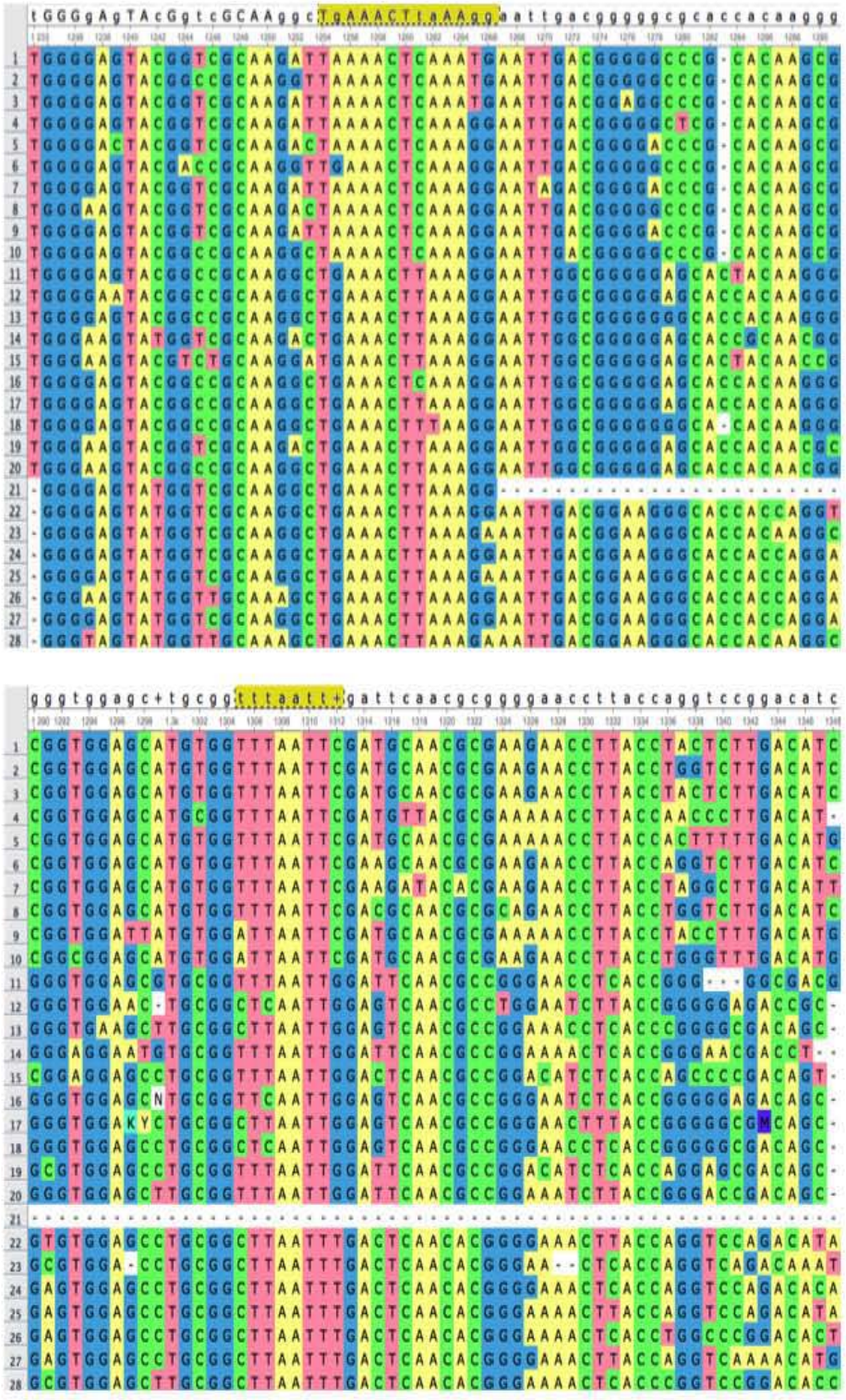

Figura 3.3: Secuencias Consenso. TGAAACTTAAAGG y YTYAATTG

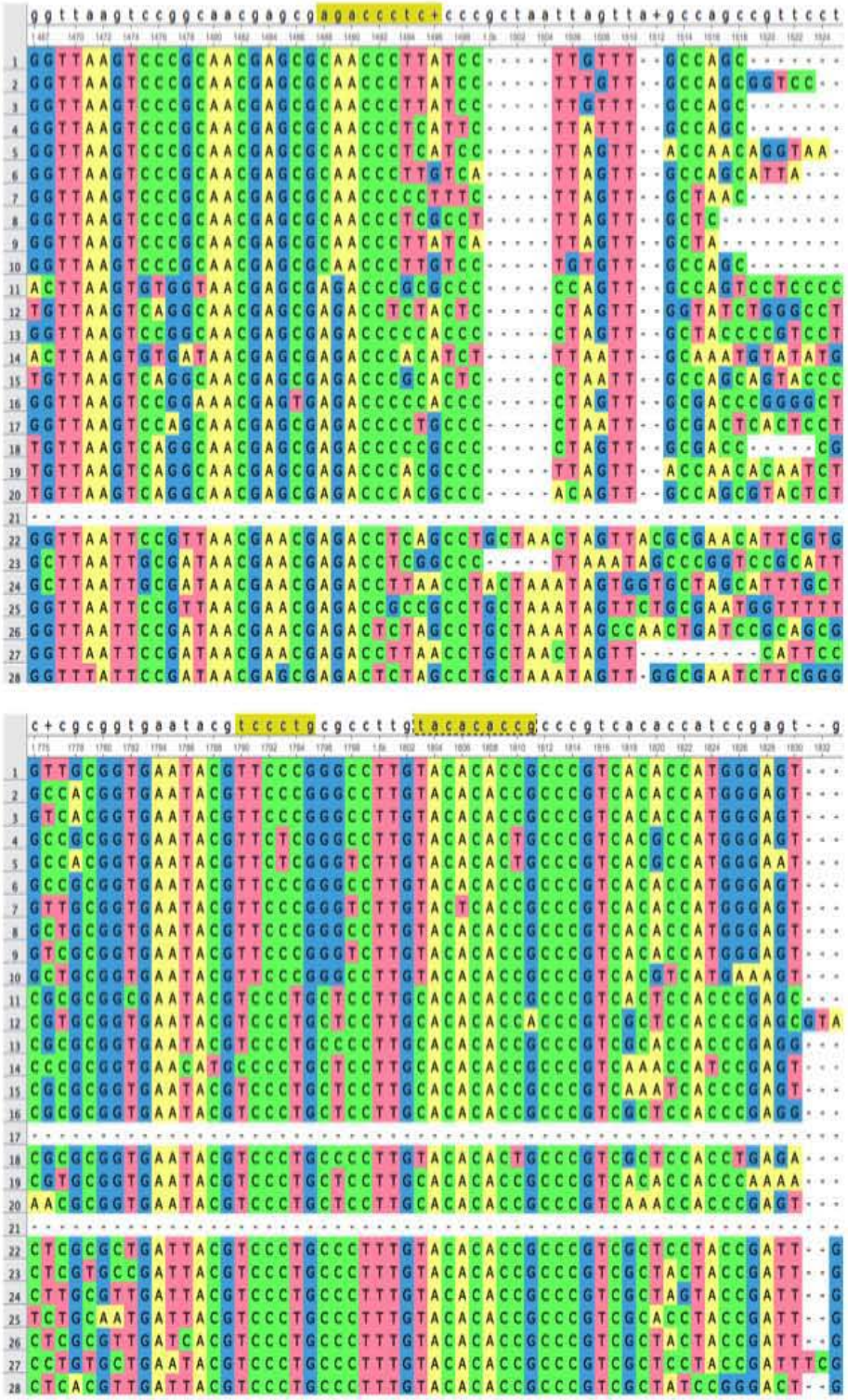

Figura 3.4: Secuencias Consenso CAACCYYCR, TCCCTG y TACACACCG

Bibliografía

[1] Kohonen, T.. (2001). *Self-Organizing Maps* . Berlin: Spinger-Verlag.

[2] Beale, R. y Jackson, T.. (1990). *Neural Computing: An Introduction* . Nueva York: Adam Hilger.

[3] Haykin, S.. (2009). *Neural Networks and Learning Machines* . Nueva Jersey: Prentice Hall.

[4] Fausett, L.. (1994). *Fundamentals of Neural Networks, Architectures, Algorithms and Applications* . Nueva Jersey: Prentice Hall.

[5] Ritter, H y Martinez, T. y Schulten, K.. (1992). *Neural Computation and Self-Organizing Maps* . Boston, MA: Addison-Wesley.

[6] Isasi, P. y Galván, I. . (2004). *Redes de Neuronas Artificiales, Un enfoque Práctico* . Madrid: Prentice Hall.

[7] Alexander, I..(1991). *An Introduction to Neural Computing* .Londres:Chapman & Hall.

[8] Hebb, D.. (1959). *The Organization of Behavior, A Neuropsychological Theory* . Nueva York: John Wiley & Sons.

[9] Murray, R., Bender, D., Botham, K., Kennelly, P., Rodwell, V. y Weil P.. (2010). *Harper. Bioquímica Ilustrada* . México, D.F: McGraw-Hill.

[10] Purves, D., Augustine, G., Fitzpatrick, D., Hall, W., Lamantia, A. y Mcnamar, J..(2007) *Neurociencia* . Buenos Aires: Editorial Medica Panamericana.

[11] Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K. y Walter, P..(2004). *Biología Molecular de la Célula* . Barcelona: Ediciones Omega.

[12] McKee, T. y McKee, J..(2003) *Bioquímica, La Base Molecular de la Vida* .Madrid: McGraw-Hill.

[13] McCulloch, W. y Pitts, W.. (1943). *A logical calculus of the ideas immanent in nervous activity* . Bulletin of Mathematical Biophysics, 5, pp.115-113.

[14] Kaski, S., Kangas, J. y Kohonen, T. (1998). *Bibliography of Self-Organizing Maps (SOM) Papers: 1981- 1997* . Neural Computing Surveys , 1, pp.102-350.

[15] Oja, M., Kaski, S. y Kohonen, T.. (2002). *Bibliography of Self-Organizing Maps (SOM) Papers: 1998- 2001 Addendum* . Neural Computing Surveys , 3, pp.1-156.

[16] Minsky, M. y Papert, S..(1969). *Perceptrons, An Introduction to Computational Geometry* .Massachusetts: MIT Press.

[17] Rumelhart, D., Hinton, G. y Williams, R..(1985). *Learning Internal Representations by Error Propagation*

[18] Watson, J. y Crick, F.(1953) *Molecular Structure of Nucleic Acids* . Nature, 171, pp.737-738.

[19] Madigan, M.T., Martinko, J.M. y Parker, J..(2006). *Brock Biología de los Microorganismos* .Madrid: Prentice-Hall.

[20] Woese, C.R., Kandler, O. y Wheelis, M.L..(1990). *Towards a natural system of organisms: Proposal for the domains Archaea, Bacteria, and Eucarya* .Proc. Natl. Acad. Sci. USA, 87, pp. 457-4579.

[21] Woese, C.R.y Fox G.E..(1977). *Phylogenetic structure of the prokaryotic domain: The primary kingdoms* .Proc. Natl. Acad. Sci. USA, 74, pp. 5088-5090.

[22] Obermayer, K., Ritter, H. y Schulten, K.(1990). *Large-Scale Simulation of a Self-organizing Neural Network: Formation of a Somatotopic Map* .Processing in Neural Systems and Computers, pp. 71-74.

[23] Ultsch, A..(1993) *Self-organizing neural networks for visualisation and classification* .Springer Berlin Heidelberg, pp. 307-313.

[24] Chan, S.C., Wong A.K.C. y Chiu, D.K.Y..(1992). *A survey of multiple sequence comparison methods* .Bulletin of mathematical biology,54(4), pp. 563-598.

[25] Johnson, S.C.(1967). *Hierarchical clustering schemes* .Psychometrika, 32(3), 241-254.

[26] Ultsch, A.(2003). *U*-matrix: a tool to visualize clusters in high dimensional data* .Berlin: Fachbereich Mathematik und Informatik.