72
291

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

UTILIZACION DE REDES NEURONALES EN LA IDENTIFICACION DE FASES SISMICAS

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A :

NORMA G. Guadalupe ORTEGA HERNANDEZ

DIRECTOR: ING. EMILIO NAVA ALATORRE

COODIRECTOR: M.C. ROBERTO ORTEGA RUIZ

CD. MEXICO, D. F. ABRIL 1997

INDICE

INTRODUCCION 1

I.INTRODUCCION A LAS REDES NEURONALES 3

Generalidades 3
I.1 Historia de las Redes Neuronales 5
I.2 Caracteristicas Funcionales de la Neurona Biológica 6
I.3 Conceptos sobre Redes Neuronales 9
I.4 Algoritmos de Aprendizaje 11
I.5 Aprendizaje(adaptación o entrenamiento) 12
I.6 Redes Neuronales Artificiales con Aprendizaje Supervisado 13

II. MODELOS DE REDES NEURONALES 14

II.1 Perceptrón 14
II.1.1 Entrenamiento 17
II.1.2 Limitaciones del Perceptrón 18
II.2 Perceptrón Multicapa 20
II.3 Adaline y Madaline 20
II.3.1 Entrenamiento 21

III. METODOLOGIA DE DESARROLLO 23

III.1 Análisis del Sistema 25
III.1.1 Definición del Problema 26
III.1.2 Objetivo General 27
III.1.3 Determinación de Requerimientos 27
III.1.3.1 Requerimientos del Sistema de Preprocesado de Eventos Sismicos 27
III.1.3.2 Requerimientos del Sistema de la Red Neuronal 28
III.1.4 Diseño Conceptual 29
III.1.4.1 Diagrama del Sistema de Preprocesado 31

III.1.4.2 Diagrama del Sistema de la Red Neuronal 32
III.2 Elección del modelo a implantar para la Red Neuronal 35
III.3 Modelo de Retropropagación 36
III.3.1 Propagación hacia adelante 39
III.3.2 Propapación hacia atrás 40
III.3.3 Algoritmo de Aprendizaje 41
III.4 Otras Aplicaciones del Modelo de Retropropagación 42
III.5 Fases Sísmicas 42

IV.DESARROLLO DEL SISTEMA DE PREPROCESADO DE EVENTOS SISMICOS 45

IV.1 Descripción del Sistema de Adquisición de Datos 45
IV.1.1 Obtención de Registros Sísmicos 45
IV.1.2 Análisis de Registros Sísmicos 46
IV.1.3 Base de Datos 48
IV.2 Entorno de Programación 50
IV.2.1 Lenguaje de Programación 50
IV.2.2 Elementos de Programación 51
IV.2.3 Compilación 51
IV.3 Descripción del ambiente gráfico 52
IV.3.1 Elementos que constituyen el ambiente gráfico 52
IV.4 Implantación del Sistema de Preprocesado de Eventos Sísmicos 54
IV.4.1 Diseño y Desarrollo del Sistema de Preprocesado de Eventos Sísmicos 54
IV.4.1.1 Petición de parámetros 54
IV.4.1.2 Extracción de datos importantes del archivo de entrada 54
IV.4.1.3 Diezmado 55
IV.4.1.4 Resta del offset 56
IV.4.1.5 Cálculo de valores absolutos 56
IV.4.1.6 Algoritmo para procesar el registro sísmico 56
IV.4.1.6.1 STA de 5 ó 20 muestras 57
IV.4.1.6.2 Integración 58
IV.4.1.7 Normalización 59

IV.4.1.8 Obtención del tiempo del arribo de la fase "p" 60
IV.4.1.9 Deslizamiento de patrones 61
IV.4.1.10 Copia de datos preprocesados al archivo destinado para la salida 63
IV.4.2 Descripción de la interface gráfica presentada en el Sistema de Preprocesamiento 65
IV.4.3 Formato de los archivos de salida 72

V. IMPLANTACION DE LA RED NEURONAL 74
V.1 Diseño y desarrollo de la Red Neuronal 74
V.1.1 Parámetros proporcionados por el usuario 74
V.1.2 Lectura del archivo de entrada a la fase de Aprendizaje y extracción de los datos grabados en el encabezado 75
V.1.3 Aprendizaje 75
V.1.3.1 Dimensionamiento de las estructuras utilizadas en el Aprendizaje 76
V.1.3.2 Inicialización de la matriz de pesos 77
V.1.3.3 Regla Delta. Propagación hacia adelante 77
V.1.3.4 Propagación hacia atrás de los errores 78
V.1.3.5 Cambio en los pesos 80
V.1.3.6 Condición de finalización del Aprendizaje 81
V.1.3.7 Finalización del Aprendizaje 82
V.1.4 Escritura de los parámetros del Aprendizaje a un archivo 83
V.1.5 Escritura de pesos y umbrales a un archivo 83
V.1.6 Lectura de los parámetros del Aprendizaje 84
V.1.7 Lectura de pesos y umbrales 84
V.1.8 Reconocimiento 84
V.2 Descripción de la interface gráfica del Sistema de la Red Neuronal 85

VI. PRUEBAS,DISCUSION DE RESULTADOS Y EVALUACION DEL SISTEMA 93

VI.1 Pruebas de Integración 93
VI.2 Pruebas de Operatividad 95
VI.2.1 Número de muestras por patrón 95

VI.2.2 Razón de diezmado 97
VI.2.3 Tipo de Proceso 97
VI.3 Calibración de la Red 98
VI.4 Resultados 99
VI.5 Mejoras propuestas 101

VII. CONCLUSIONES 108

APENDICES

APENDICE A.
Declaración de los objetos de OPENWINDOWS utilizados en la creación del ambiente de ventanas del sistema.

APENDICE B.
Presentación de los 14 eventos sismicos que constituyen tanto al grupo de Entrenamiento como al de Reconocimiento.

DEDICATORIAS Y AGRADECIMIENTOS

Dedico este presente trabajo a todas las personas que en menor o mayor grado se involucraron en su realización. Culminar un sueño no es algo fácil, pero la satisfacción de lograrlo es una de las mejores cosas que existen en la vida. Puedo decir ahora que, el apoyo y fé depositados en mí a cada paso, contribuyeron en gran medida a realizarlo. Gracias a todos.

A Dios:
Quien ha guiado siempre mis pasos en los momentos más difíciles así como en los más gratos. Por darme la capacidad de aprender de mis errores y gozar con mis aciertos. Simplemente, gracias por darme la oportunidad de valorar lo que significa este momento.

A mi madre, Señora Juana Hernández de Ortega:
Por ser siempre el apoyo más grande en mi vida, por darme una palabra de aliento cuando sentía que los obstáculos eran mayores creyendo incondicionalmente en mí. Por sacrificarte a lo largo de tu vida para darme la oportunidad de tener una formación profesional que atesoro y atesoraré. Gracias por ser amiga y comprenderme . Pero sobre todo gracias por el inmenso amor que me profesas.

A mi padre, Señor Jose Luis Ortega López:
Por el apoyo moral recibido, por cada una de las enseñanzas que me has transmitido y que has acumulado mediante el camino andado. Por esa necesidad de superación emanada de ti. Gracias por la confianza y el amor que muy a tu manera me has demostrado a lo largo de mi vida

A ambos:
Por darme la vida y la ayuda necesaria para formarme tanto moral como económica, inculcándome el amor a todo lo que hago principalmente al estudio. Por enseñarme que lo que se desea se consigue con esfuerzo y dedicación. Todo lo que soy y lo que seré se los debo a ustedes Nunca los defraudaré. Este trabajo es en gran medida por y para ustedes. Los quiero.

A la memoria del Señor Guadalupe Suarez Cervantes:
Me hubiera gustado que en este momento estuvieras aquí , compartiendo conmigo la satisfacción de ver culminado este trabajo, pero sé que donde quiera que estés me apoyas y guiarás mis pasos. Gracias por haber sido un Tio ejemplar y por enseñarme la inmensa capacidad de demostrar mi afecto. Que mas puedo decir sino que te quiero, querré y recordaré siempre.

Al Señor Antonio Hernández:
A veces nos comportamos de una manera contradictoria y le hacemos daño a lo que más queremos, lamento que la vida y nosotros mismos no nos demos la oportunidad de estar juntos, disculpa si en algún momento no te hice sentir mi apoyo Tio, pese a esas circunstancias agradezco a ti y a mi tía los momentos vividos donde me hicieron sentir el gran cariño y confianza que sentían por mí Gracias..

A mi Universidad:
Gracias por las enseñanzas adquiridas dentro de tus aulas, porque debido a ellas me brindas hoy la felicidad de este momento. Porque en los corredores de la Facultad de Ingeniería aprendí el significado de tratar y ser mejor . Ten por seguro que siempre llevaré en alto lo que tu nombre significa . Gracias también a mis profesores, quienes con su dedicación a la cátedra dan a miles de jóvenes la oportunidad de lograr sus sueños, dando como principal arma para enfrentar al futuro la capacidad de buscar y construir nuestro propio destino.

A mi direcor de Tesis, Ingeniero Emilio Nava:
Por la guía que me brindaste en el desarrollo de este trabajo, porque en los momentos de incertidumbre me diste aliento y sobre todo por ser la gran persona que eres y por tener la gran familia que tienes. Gracias.

A Gretel :
Por los momentos vividos, por la gran afinidad que experimentamos, pero sobre todo por saber comprenderme y estar ahí cuando lo necesito.Admiro tu fortaleza para enfrentar los problemas,en fin, gracias amiga.

A Martha:
Por enseñarme que aunque vivas momentos difíciles tenemos la capacidad de asimilarlos y luchar por lo que deseamos. Gracias por tus palabras y amistad.

A mis amigos:
Quienes supieron consolarme, pero también impulsarme para conseguir todo lo que un dia soñé conquistar, porque un verdadero amigo nos hace ver nuestras fallas pero también tiene la capacidad de percibir nuestras cualidades. Gracias a cada uno de ustedes. A mis amigos burbujos por las sonrisas , las lágrimas, pero sobre todo gracias por ser amigos verdaderos. Siempre habrá un lugar en mi corazón para ustedes.

Al Instituto de Ingenieria:
Un sincero agradecimiento al Instituto de Ingenieria así como a todas aquellas personas que colaboraron de una u otra manera en el desarrollo de este trabajo: Miguel, Tatiana, Arturo, Nicolas, Salvador, Verónica, Rosa, Horacio, Adela, Martin, Jorge, Alejandro, Marco, Leobardo, Francisco Sánchez, Sra. Josefina, en fin, si llegara a omitir a alguien , mil perdones, pero ellos saben bien el apoyo que me brindaron.

A Humberto :
Por la ayuda que me brindaste para finalizar este trabajo, porque sin ella hubiera sido más difícil culminarlo, pero sobre todo gracias por el apoyo incondicional que me has demostrado.

A mis compañeros y amigos de trabajo:
Por apoyarme dentro y fuera del trabajo, por alentarme y contribuir con su granito de arena a lograr que esta tesis llegue a su culminación. Gracias mil por su colaboración desinteresada: Claudia, Arturo, Isaías, Noé, Jorge, Julio .

Gracias infinitas
Norma.

INTRODUCCION

Las redes neuronales pueden definirse como el intento del hombre por crear un cerebro artificial. Sin embargo, dada su étapa actual de desarrollo y su habilidad para inferir e intuir a partir de información incompleta ó confusa, sería más correcto hablar de ellas como un intento de la humanidad por simular la forma en que el cerebro funciona

De igual manera el hombre ha intentado entender los fenómenos naturales, siendo éstos una constante fuente de investigación, dada su interaccion con él mismo y su entorno Entre los fenómenos que le preocupan se pueden citar los sismos

Esa inquietud por conocer y entender lo que le rodea ha provocado que la ciencia y la tecnología hayan tenido una evolución importante, pudiendo constatarse en el campo de acción bastante amplio con el que cuentan actualmente las Redes Neuronales,debido a su utilización en diversas áreas como: Análisis médico, Análisis de imágenes (desde complejas a simples dependiendo del nivel de abstracción), Reconocimiento de caracteres escritos a mano, pero sobre todo en el procesamiento de señales como interpretación y clasificación de datos sísmicos

Este presente trabajo conjuga la teoría de las Redes Neuronales con la sismología, ya que mediante las técnicas que han sido encontradas se tiene una mayor gama de posibilidades para automatizar y optimizar procesos con el fin de estar a la vanguardia tecnológica o simplemente comprobar que dichas teorías pueden ser utilizadas en problemas muy específicos

Por su distancia un sismo se puede clasificar en 3 diferentes categorías: sismos locales, regionales, y lejanos (telesismos) La localización de un sismo depende de la identificación de varios parámetros contenidos en él , dentro de los cuales se encuentran las fases llamadas "P" y "S" Este trabajo se enfoca a la identificación de la fase "P" utilizando para esto, un modelo de Redes Neuronales llamado Retropropagación. Se pretende mediante esta técnica identificar el arribo de una manera automática

Ahora bien, se presenta un grupo de 14 eventos sísmicos (tanto locales como regionales) homogéneo en sus características generales. Los sismos fueron seleccionados por su claro arribo en las fases "P" y "S" , con el objetivo de lograr un fácil reconocimiento mediante la Red Neuronal . Se pueden describir tres étapas básicas en el desarrollo de este trabajo :

- **Preprocesado.**
- **Aprendizaje de la Red Neuronal (Presentación de los patrones de entrada a la Red y la clasificación de éstos).**
- **Reconocimiento por la Red Neuronal (Se presentan patrones de entrada diferentes a los utilizados en el aprendizaje, pretendiendo que el algoritmo los reconozca en base a la experiencia adquirida) .**

Mediante los pasos realizados en el Preprocesado se pretende tener a la salida una señal lo mas depurada posible que sea la entrada a la Red Neuronal , esta es la razón por la cual este paso es sumamente importante , ya que dará la pauta para un buen desempeño.

Se seleccionó el modelo de Retropropagación con aprendizaje supervisado, para utilizarse en el reconocimiento de la fase en los registros sísmicos, esto significa que, durante la presentación de un patrón de entrada, se especifica también el patrón de salida, dicho patrón constituye la clasificación correcta del patrón de entrada .

Se va a reconocer el arribo de la fase P siendo necesario categorizar la señal con el siguiente criterio:

- Antes del arribo de la fase , la salida será 0.1.
- Después de éste será de 0.9 .

El archivo de salida con estos parámetros será de tipo texto, y la información estará distribuida de la siguiente manera :

Patron 1 (datos de salida del preprocesado) valor deseado de salida (0.1 antes del arribo de P, y 0.9 en el arribo de P y después de el) . Patrón 2 valor de salida deseado , ..., Patrón n valor de salida deseado .

Ahora, teniendo la señal preprocesada y categorizada, se procede a entrenar la Red con ciertos eventos sísmicos . Finalizando la étapa de Aprendizaje se presentan eventos diferentes a los utilizados en el entrenamiento, con el fin de ser reconocidos por la Red(Fase de Reconocimiento), logrando con esto comprobar la teoría de los algoritmos inteligentes que aprenden de la experiencia como lo son las **REDES NEURONALES** en un área como lo es *El Preprocesamiento de Señales* y especificamente la *Clasificación de Eventos Sísmicos*

Cuando se corre la parte del Reconocimiento , el algoritmo de la red indica en que patrón de entrada de ese evento sísmico se encontró el arribo de la fase P, así como la fecha y hora de ocurrencia del mismo

Para saber el grado de eficiencia del algoritmo se procedió a tomar el tiempo de ejecución del aprendizaje y del reconocimiento, notando que este último es bastante rápido, debido a que se toman los valores de la última matriz de pesos a la que se llego en el aprendizaje, realizándose los cálculos correspondientes al pasar sólo una vez por el procedimiento "forward" (propagación hacia adelante), tomando el valor de salida calculado por la red como valor final , es decir, no se realiza fase de retropropagación de errores .

Resumiendo , un buen reconocimiento de un evento sísmico significa que la red asocia el patrón de entrada que está procesando con los anteriores patrones aprendidos y así lo clasifica, por eso se dice que utiliza los principios de funcionamiento del cerebro , asociando y clasificando la información según la experiencia.

I . INTRODUCCION A LAS REDES NEURONALES.

Generalidades

Entender cómo funciona el cerebro ha sido un problema complejo para el conocimiento humano, esto se debe a que las funciones que éste realiza incluyen precisamente aquellas que hacen al hombre el más evolucionado de los seres vivos: ***las funciones mentales.***

El cerebro es una computadora muy notable. Es capaz de interpretar información imprecisa suministrada por los sentidos a un ritmo increiblemente rápido, así como reconocer objetos tanto concretos como abstractos, es decir, logra discernir un susurro en una sala ruidosa, un rostro en una calle mal iluminada, etc. Y lo más impresionante de todo: " El cerebro aprende (sin instrucciones explícitas de ninguna clase) a crear las representaciones internas que hacen posibles tales habilidades".

Pero ¿cómo el cerebro logra identificar un objeto aún siendo confuso?, pues bien, cuando una persona percibe un objeto, infiere y asocia su percepción con información derivada de su experiencia , en este caso, la percepción representa un proceso de formación de un modelo complejo del mundo, el cerebro reacciona no a lo que observa sino a la diferencia entre lo que espera y lo que sucede, haciendo hipótesis, y cambiándolas cuando algo inesperado ocurre y contradice estos modelos de trabajo , a este fenómeno se le denomina adaptación . En otras palabras , el cerebro compara el modelo esperado con los datos recibidos.

En base a lo anterior, es importante notar que el cerebro trabaja de la siguiente manera :

- *Organización asociativa*
- *Adaptabilidad*
- *Aprendizaje*
- *Percepción*

Se puede decir que el reconocimiento de cualquier patron es realmente la discriminación de la información que recibimos del mundo externo, mediante la extracción de caracteristicas ó atributos .

Por todo lo mencionado anteriormente, los investigadores han perseguido la posibilidad de construir dispositivos de procesamiento de información, que imitaran las estructuras y principios de operación encontrados en los seres humanos. Esa necesidad provocó la creación de una nueva clase de computadoras llamadas neurocomputadoras.

Las neurocomputadoras fueron diseñadas para procesar la información que provenia del mundo externo, sin pasar por la representación simbólica, siendo ésta, la diferencia entre las computadoras digitales y las neurocomputadoras.

En muchas de las aplicaciones del mundo real, se quisiera que las computadoras ejecutaran problemas complejos de reconocimiento de patrones como:

- *Distinguir entre diferentes clases de objetos similares*
- *Comprender el significado de formas en imágenes visuales. Procesamiento de imágenes.*
- *Procesamiento de Señales.*
- *Neurocontrol,etc.*

Por esta razón, se pidieron prestadas algunas características de la fisiologia del cerebro, como base de nuevos modelos de procesamiento; De aquí surgió la tecnologia ANS (SISTEMAS NEURONALES ARTIFICIALES) ó simplemente REDES NEURONALES.

La elección del algoritmo de aprendizaje utilizado por la Red Neuronal depende de:

- La función de activación
- La forma de conexión entre las neuronas designada con el nombre de conectividad, la cual puede ser de dos tipos: feedback y feedforward.
- El tipo de información que va a ser procesada por el algoritmo .

Además , existen 2 maneras de clasificar a los algortimos de aprendizaje.

- Grado de supervisión , el diseñador decide que tipo de supervision utilizará según los requerimientos del problema al que se enfrenta, para discernir entre la mejor opción se tienen 3 grados de supervisión dentro de los algoritmos de aprendizaje :

- Aprendizaje Supervisado (utilizado en este trabajo)
- Aprendizaje Reforzado
- Aprendizaje No Supervisado.

- Determinismo, en el cual los algortimos son agrupados por la naturaleza de sus reglas de aprendizaje , pudiendo ser :

- Aprendizaje Deterministico
- Aprendizaje Estocástico.

Dentro de la teoria de Redes Neuronales Artificiales (ANNs) existen modelos tanto simples como complejos, los cuales demuestran la evolución que dicha teoria ha experimentado desde sus inicios (figura 1.1) . El tener una gran diversidad de modelos dá como consecuencia un panorama amplio de acción.

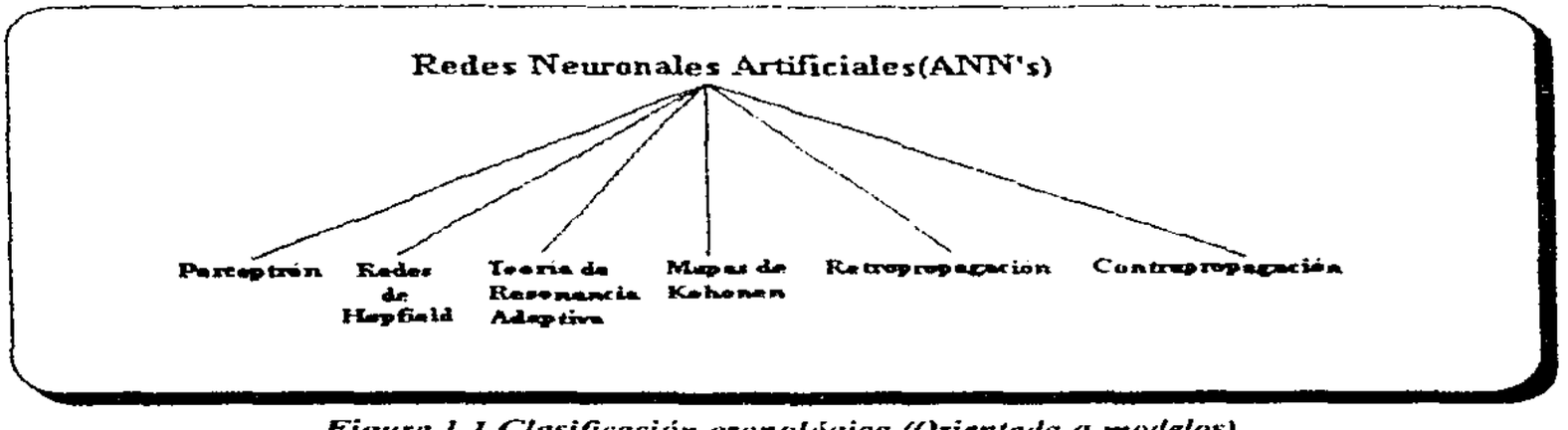


Figura 1.1 Clasificación cronológica (Orientada a modelos).

En los últimos años, el renacimiento de las Redes Neuronales ha sido de gran importancia. Desafortunadamente la cantidad de tiempo y de recursos requeridos para capturar el conocimiento

suficiente y para programar la computadora es demasiado grande, aún cuando la finalidad sea resolver un problema poco complicado. Ante esta situación resurge la idea del aprendizaje adaptivo, que permite que la máquina adquiera el conocimiento por medio de interacciones de prueba y error con su medio ambiente, tal y como ocurre en el caso de los seres humanos.

I .1 Historia de las Redes Neuronales .

La teoría de las Redes Neuronales nace en los 40's con el trabajo de **McCulloch y Pitts,** en el cual se concibe al cerebro como una unidad integrada por elementos computacionales bien definidos, ***las neuronas.*** Se supuso a cada neurona como un elemento generador, el cual recibe señales de entrada estandarizadas y responde de acuerdo a las leyes que definen la operación de los circuitos lógicos o booleanos.

Muchos creen que los escritos del psicólogo William James en el siglo XIX originaron las ideas fundamentales del conexionismo, en ellos él propone una regla para el aprendizaje que dice: ***La Co-ocurrencia causa reforzamiento en los procesos del cerebro, el grado de reforzamiento depende de la fuerza de co-ocurrenica de los eventos.*** También se sabe que alrededor de 1900, se dieron dos de los más grandes avances acerca del cerebro En Esparta, **Ramón y Cajal** muestran estudios anatómicos de muchas regiones del cerebro, revelando su estructura como una red de neuronas. En Inglaterra, los estudios fisiológicos de **Charles Sherrington**, sobre el primer entendimiento de la sinápsis, muestran los puntos de unión entre neuronas.

A los modelos creados por **McCulloch y Pitts** se les llamó Neuronas Formales, en estos modelos, cada unidad es activada si y sólo si, su excitación total alcanza o excede el 0 (umbral), mostraron que la neurona puede ser modelada como un dispositivo de umbral para ejecutar una función lógica.

Estos modelos llamados "neuronas formales" nos describen lo que hacen las neuronas reales Desafortunadamente, no se contempló el caso en el que las redes neuronales tuvieran un mal funcionamiento, debido a esto, **Von Neumann** (uno de los grandes matemáticos de éste siglo) introdujo el concepto de ***redundancia***(usar muchas neuronas para hacer el trabajo de una), logrando que la función de las redes fuera más confiable.

Después de las Redes de **McCulloch-Pitts,** surgió el trabajo de **Winograd-Cowan,** ellos construyeron una red en la que utilizaban una representación distribuida de la información, un bit de información fue representado redundantemente por muchas neuronas, como en la Red de **Von Neumann**(antes mencionada).

A finales de los 50's aparecieron 3 importantes modelos de ANN's : El perceptrón de **Rosenblatt,** Pandemonium de **Selfridge** y la Adaline de **Widrow** Entre éstos años y principios de los 60's, los modelos neuronales fueron redefinidos dentro del trabajo de **Rosenblatt, Widrow y Hoff** llamado ADALINE (Elemento Lineal Adaptivo) y la Matriz de Aprendizaje de **Steinbuch.** La convergencia del aprendizaje del perceptrón **(Block 1962),** se basó en experimentos relacionados con tareas de reconocimiento visual. El modelo *Pandemonium* de Selfridge compuesto de capas múltiples como el perceptrón, fué diseñado para ejecutarse en clasificación de patrones, en este modelo se introdujeron algunos principios fundamentales del aprendizaje estocástico: **Selfridge** estableció que las unidades de las capas ocultas deberían aprender distintas características de los patrones de entrada

Hopfield no fué el primero en reconocer las propiedades de integración temporal y espacial de las neuronas. A principios de los 70's **Paul Werbos** descubrió los principios matemáticos del algoritmo de propagación. Durante los 70's, **Steven Grossberg** en la Universidad de Boston y **Teuvo Kohonen** en la Universidad Helsinki, hicieron contribuciones importantes. **Grossberg** junto con **Gail Carpenter**, desarrollaron una arquitectura para la Red Neuronal llamada *Teoria de Resonancia Adaptiva(ART)*, basada en la idea de que el cerebro se organiza espontáneamente dentro de códigos de reconocimiento; la dinámica de la red también fué modelada a través de ecuaciones diferenciales de primer orden.

A mitad de los 80's,**David Rumelhart** y sus colegas publicaron su libro *"LANDMARK"* sobre procesamiento distribuido paralelo, el cual establece el algoritmo de retropropagación y la Red de Capas Feedforward, como los mejores descubrimientos en el campo.

1985 marca el advenimiento de la investigación de las ANN's modernas, pero también en este año el algoritmo de *Retropropagación (Backpropagation)* es redescubierto por **Parker, Le Cun, Rumelhart, Hinton y William.** La tabla siguiente resume las diferentes etapas por las que ha pasado la historia de las Redes Neuronales.

1890-1950	**Estudios preliminares de las bases de la Inteligencia: James, Hebb, Lashley y otros.** **Modelos Abstractos computacionales de las Redes Neuronales : McCulloch y Pitts.**
1950-1970	**Primeras Redes Neuronales Computacionales: Minsky,Rosenblatt,Selfridge, Widrow y Samuel.** **Aparece el libro de Minsky y Papert en 1969, marcando el fin de esta era.**
1970-1985	**Continuación del trabajo sobre memorias asociativas: Kohonen, Willshaw, Kanerva,Anderson y Hopfield.** **Máquina de Boltzmann:Kirkpatrick,Ackley,Hinton,Sejnowski.** **Aprendizaje en Redes Multicapas:Werbos,Parker,Le Cun y Rumelhart.**

1.2 Características funcionales de la Neurona Biológica .

Los conceptos sobre redes neuronales se basan en los principios de funcionamiento de las neuronas biológicas, de aqui la importancia de conocer dichos principios para un mejor entendimiento de cómo trabaja una red neuronal. Es importante mencionar que las neuronas son células vivientes capaces de recibir y transmitir señales electroquímicas

Fundamentos Biológicos :

La Anatomia de una neurona real (biológica) comprende :

- *Ramificaciones del citoplasma llamadas dendritas,donde la neurona capta señales que provienen de otras neuronas.*
- *El cuerpo de la célula, llamado pericarión o soma.*
- *Una larga línea de transmisión llamada axón.*
- *Estructuras que se encuentran al final del axón,llamadas botones sinápticos*

Una neurona típica del cerebro humano recoge señales procedentes de otras, a través de un gran número de delicadas estructuras llamadas ***dendritas***. La neurona emite impulsos de actividad eléctrica a lo largo de una fibra delgada y larga denominada ***axón***. La sinápsis convierte la actividad procedente del axón en efectos eléctricos que inhiben o provocan actividad en las neuronas a las que está conectado. Cuando las señales excitadoras que una neurona recibe alcanzan suficiente intensidad frente a las señales inhibidoras, la neurona envía a lo largo de su axón un breve impulso de actividad eléctrica.

Las neuronas que constituyen el cerebro con sus variadas formas y tamaños, no están distribuidas al azar una junto a la otra, sino que están organizadas de una manera extraordinariamente precisa, de tal modo que forman complicadas vías ó circuitos, en los que cada una de ellas se puede comunicar sólo con aquellas que forman parte de la vía ó del circuito.

Cajal menciona que no hay continuidad entre las neuronas, pero si contiguidad, la cual se manifiesta en forma especial, precisamente en los sitios específicos de comunicación entre las neuronas, a estos sitios **Sherrington** les llamó : *Sinápsis* , definiéndola como la unión de dos neuronas que se lleva acabo al ponerse en contacto las prolongaciones del axón de una neurona con las dendritas de otra.

En Resumén, una neurona recibe señales(tipicamente en la forma de un tren de pulsos) de sus vecinos, vía la sinápsis, entonces ejecuta una suma de pesos de las entradas y calcula una función de umbral, y cuando éste valor excede un cierto umbral produce una salida.

La figura 1.2(a) muestra una neurona real (biológica), en cambio la figura 1.2(b) muestra la misma neurona pero de una forma matemática

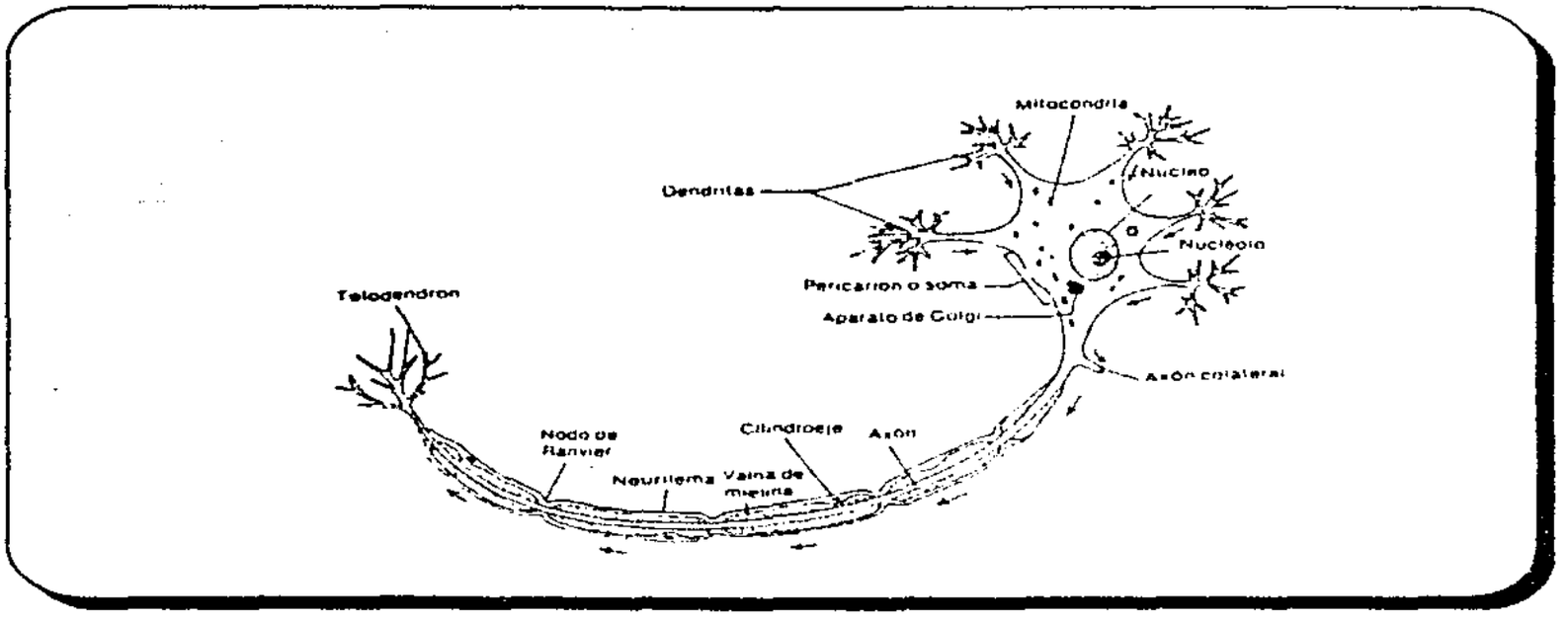


Figura 1.2(a). Neurona real (biológica).

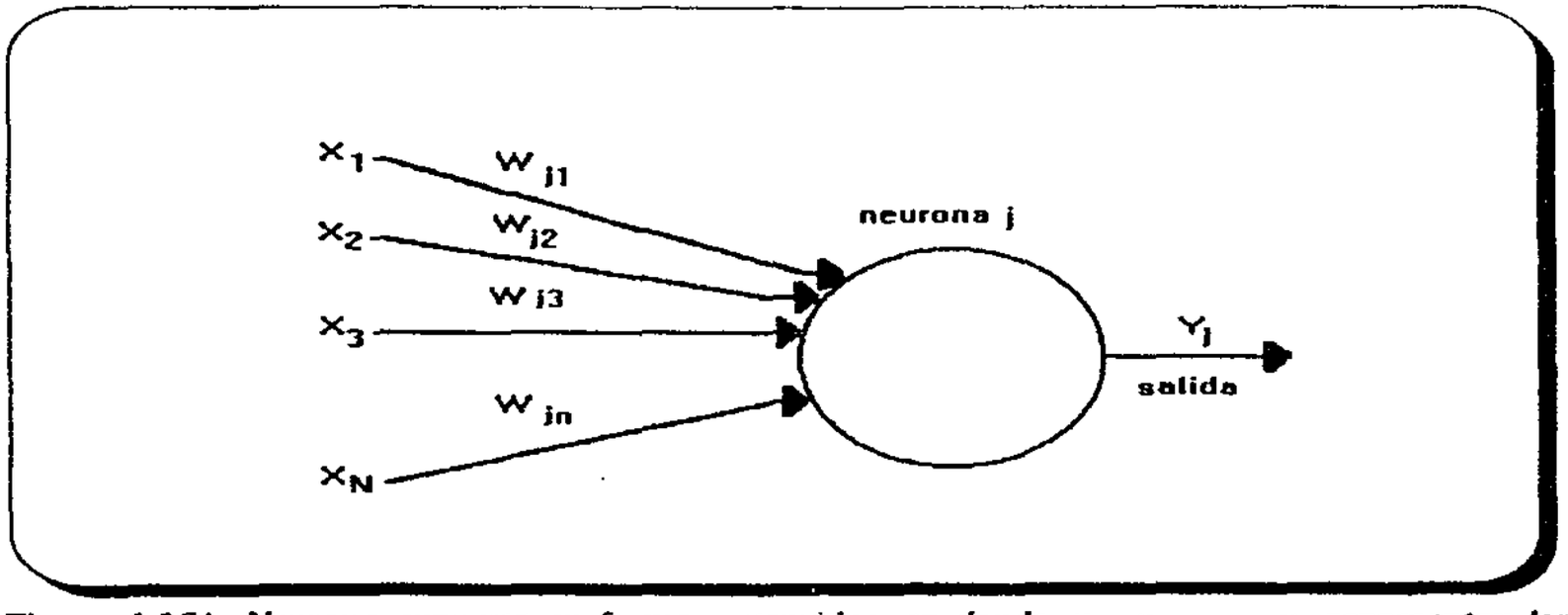


Figura 1.2(b). Neurona expuesta en forma matemática, en donde, x_1, x_2 ....x_n representan las entradas que recibe la neurona j, las W_{j1} representan las sinapsis y Y_j es la salida de la neurona j.

1 . 3. Conceptos sobre Redes Neuronales .

Una Red Neuronal es una colección de procesadores en paralelo,conectados entre sí, organizados de tal manera que la estructura de la Red resuelva el problema que se le presenta.

Una Red Neuronal debido a su naturaleza masivamente paralela, puede ejecutar cálculos en un rango muy alto. Debido a su naturaleza adaptiva, puede adaptarse a cambios en los datos y aprender las características de las señales de entrada. En cuanto a su naturaleza no lineal puede ejecutar aproximaciones y operaciones de filtrado

Está constituida por: *-Elementos de procesamiento (PES) llamados también nodos ó unidades(Representando a las neuronas) ,y lineas de interconexión entre ellos (Representando a las Sinápsis).* La dirección del flujo de información dentro de la Red, se designará a través de flechas.

Las entradas son de varios tipos, esto es, una entrada puede producir diferentes efectos. Una conexión de entrada puede ser *excitatoria ó inhibitoria*, por ejemplo Las conexiones excitatorias tienen pesos positivos, mientras que las conexiones inhibitorias tienen pesos negativos. Cada PE determina un valor de entrada de la red, basada en todas sus conexiones de entrada. La entrada de la red se puede expresar de la siguiente forma:

$$\mathbf{net}_i = \sum_{j=1}^{n} x_j w_{ij} \quad \text{..... (1.1)}$$

Donde, n es el número de todas las conexiones del PE.

Una vez que se ha calculado el valor de la entrada, se procede a encontrar el valor de activación del PE, que en general es una función del siguiente tipo

$$a_i(t) = F_i\,(a_i(t-1), \mathbf{net}_i(t)) \quad \text{..... (1.2)}$$

El valor de activación puede depender de un valor de activación previo. El valor de salida corresponde a una función del tipo

$$x_i = f_i(a_i) \quad \text{..... (1.3)}$$

En muchos modelos de redes neuronales, es más útil representar sus cantidades en términos vectoriales , la ecuación (1.1) puede expresarse de la siguiente manera

$$\mathbf{net}_i = \mathbf{x} \cdot \mathbf{w}_i \quad \text{..... (1.4)}$$

Donde: x es un vector n-dimensional $\mathbf{x} = (x_1, x_2, x_3, \ldots, x_n)^t$, t significa transpuesta.

Como una Neurona Real, el PE tiene muchas entradas, pero sólo una salida, que a su vez puede alimentar a otros PE's de la Red. Cada conexión tiene asociado un valor w denominado Peso ó Fuerza de Conexión. El peso de conexión entre el nodo j al nodo i es expresado como Wji. La alteración del peso asociado a cada conexión es lo que permite la adaptación a nuevas situaciones (Figura 1.3) .

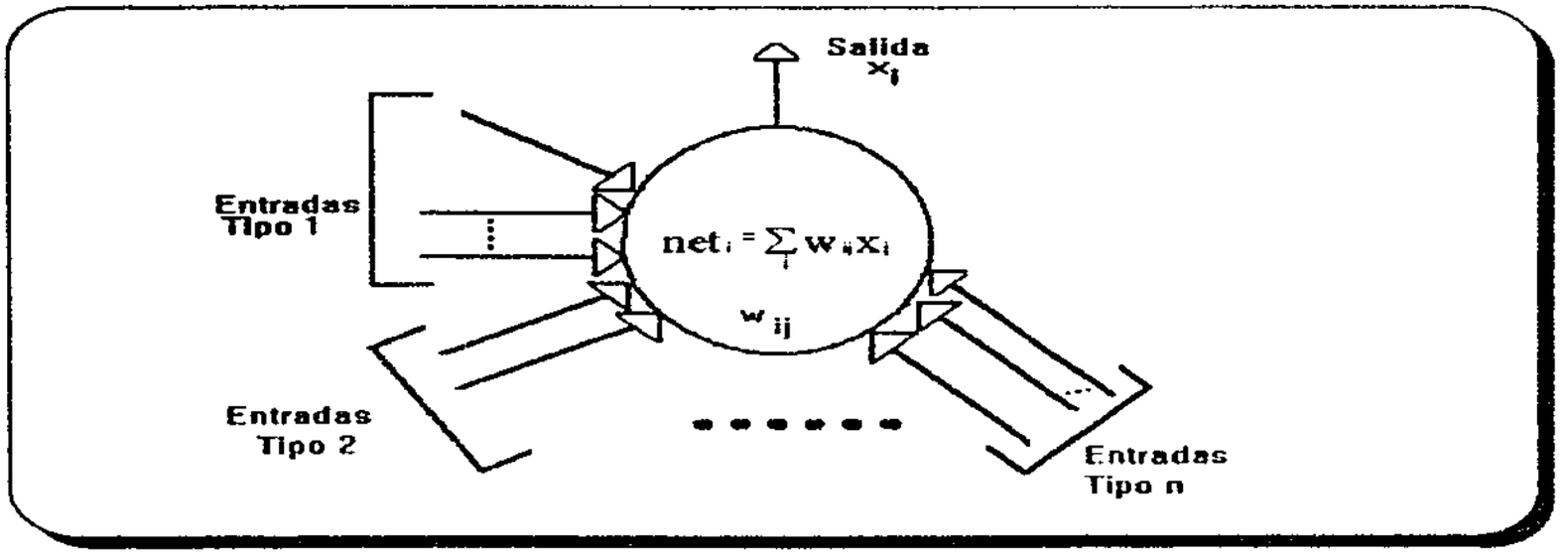


Figura 1.3. Modelo general de un Elemento de procesamiento (PE) en una Red.

Es importante hacer notar que el comportamiento de una red neuronal artificial depende tanto de los coeficientes de ponderación como de la función de transferencia que se especifica para las unidades. Tal función puede ser: **a) lineal, b) de umbral** ó **c) sigmoide.**

a)En el caso de unidades lineales,la actividad de salida es proporcional a la entrada ponderada total.

b)Para unidades de umbral, la salida queda fija a uno de dos niveles, dependiendo de que la entrada o ingreso total sea mayor o menor que cierto valor crítico, llamado umbral.

c)En las unidades sigmoides, la salida varia continuamente con la entrada, pero la dependencia no es lineal. Las unidades sigmoides guardan mayor parecido con las neuronas reales que las unidades lineales o de umbral (mencionada en la ecuación 1.2) ; Sin embargo, las tres son una mera aproximación.

Las funciones sigmoidales no lineales usadas en las Redes con capas ocultas, tienen una desventaja, causan múltiples mínimos, que aparecen durante el aprendizaje, dando como resultado que nunca se esté seguro si el sistema alcanzó el máximo global. Se puede decir entonces que los 3 ingredientes esenciales en una Red Neuronal Artificial son : la función de transferencia, la arquitectura y la regla de aprendizaje.

Para construir una red neuronal capaz de realizar cierta tarea especifica debemos establecer cómo estarán conectadas unas unidades con otras y determinar adecuadamente los pesos atribuidos a las conexiones. Las conexiones determinan si es posible que una unidad influya sobre otra : es decir, los pesos definen la intensidad de la influencia.

Los elementos de procesamiento (PE) con entradas externas son llamadas ***unidades de entrada***, y con salidas externas son llamadas ***unidades de salida***, el resto son llamadas ***unidades ocultas.***

I . 4. Algoritmos de Aprendizaje.

Una red neuronal aprende al incorporar experiencia pasada a sus patrones de interconexión. Generalmente, las redes realizan cálculos u operaciones en paralelo y utilizan los resultados para ajustar sus patrones de interconexión.La razón por la que las redes utilizan el proceso en paralelo es que, los modelos se enfocan en la manera en la cual cada estado que constituye un problema contribuye en forma simultánea en su solución.

Es conveniente mencionar que existen 8 aspectos básicos enumerados por **Rumelhart (1986)** que caracterizan a un modelo de red neuronal de procesamiento en paralelo:

1.Un conjunto de unidades de procesamiento.
2.Un estado de activación
3.Una función de salida para cada unidad.
4.Un patrón de conectividad.
5.Una regla de propagación de actividades para patrones de propagación
6.Una regla de activación para combinar las entradas que afecten una unidad, con un estado definido para producir la salida
7 Regla de aprendizaje, através de la cual puedan modificarse los pesos de las interconexiones en base a la experiencia
8.Medio ambiente bajo el cual el sistema de aprendizaje debe aprender

En la figura 1.4 se muestra una unidad de procesamiento simple, en la cual, la función mostrada en la caja indicará si la suma (tomando en cuenta los pesos), de las dos entradas es mayor o menor que cero El patrón de entrada (i1,i2), contiene valores de +1 ó -1. Los valores de los pesos w1 y w2 son los que deciden la importancia relativa de las entradas i1 e i2 para determinar la salida final. Entonces, si por ejemplo, se establece w1 a 0, esto implica que la salida reflejará el signo de la entrada i2.

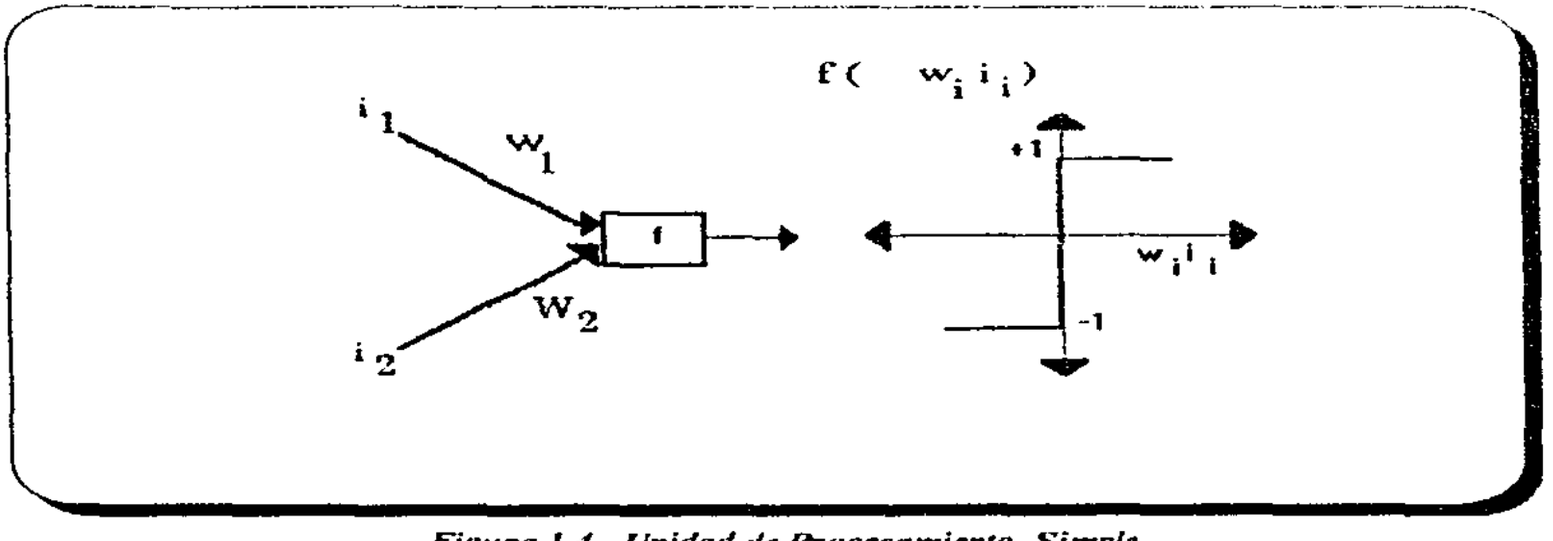


Figura 1.4. Unidad de Procesamiento Simple.

Los pesos tienen el efecto de determinar una linea en un plano de dos dimensiones tal que, todos los puntos de un lado la de la linea se distinguen por medio de la función de salida de aquellos puntos del otro lado.

De esta forma, un valor positivo para w1 implicaría que la entrada i1 es excitadora para este nodo en particular, si el peso fuera negativo, eso significaría que la entrada es inhibidora.

Una interconexión entre unidades (análoga a la sinápsis en biología), tiene dos características básicamente:

1) su naturaleza (puede ser excitadora o inhibidora)
2) el grado de influencia que la unidad de la cual proviene la interconexión tiene en la unidad incidente, lo cual es representado por el peso asociado a la interconexión

La modificación de las interconexiones típicamente implica una modificación de los pesos asociados y ocurre cuando las redes neuronales aprenden algo en respuesta a nuevas entradas o cambios en el medio ambiente. Una de las reglas de aprendizaje más comunes creada por **Hebb (1949)** y llamada por lo tanto regla hebbiana, tiene el siguiente modelo matemático propuesto por **Shutton (1981)**.

$$W_i\ (t+1) = W_i\ (t) + c\ x_i(t)\ y\ (t) \ldots\ (1.5)$$

donde: $x_i(t)$ es una de las entradas al nodo cuya salida es y (t) y c es una constante positiva que determina el índice de aprendizaje.

Entonces, se puede decir que las redes neuronales no se programan, ellas aprenden por medio de ejemplos. Se le muestra a la red neuronal un conjunto de patrones de entrenamiento con el fin de que ésta aprenda. Los ejemplos o patrones de entrenamiento están representados como vectores, y se obtienen a partir de transformaciones de imágenes, señales de voz, movimientos de brazos robóticos, etc

1.5. Aprendizaje (adaptación ó entrenamiento).

El aprendizaje es una de las características más importantes de las Redes Neuronales Artificiales. Es un proceso dinámico en el cual la organización de una red (la estructura de su conectividad) es modificada de acuerdo con la interacción de la red con estímulos externos.

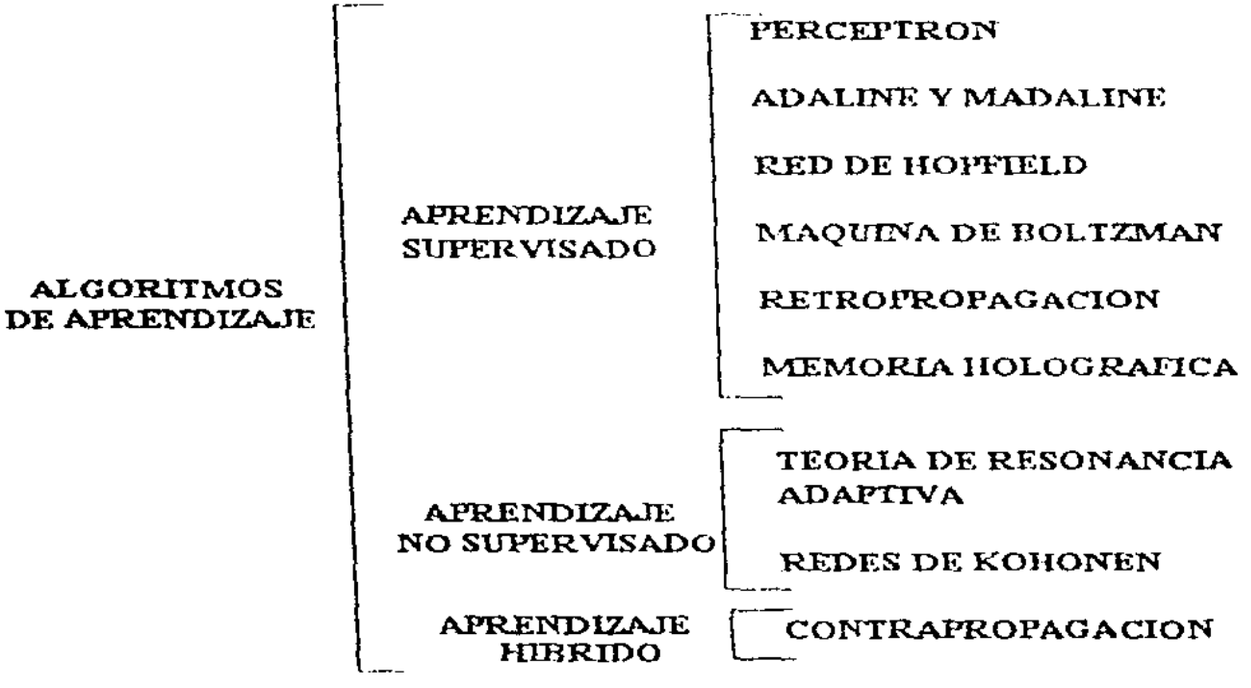
Se presenta a la red un determinado estímulo durante un periodo relativamente largo, suficiente para que dicha red pueda modificar su estructura sináptica asimilando el estímulo que se le está aplicando. De esta forma, el comportamiento futuro de la red ante un estímulo similar que se presente en un tiempo corto, provocará en la red, un proceso dinámico como resultado del aprendizaje ya que la red ha aprendido a responder ante ese estímulo ó alguno similar

En las redes neuronales, el conocimiento es representado por las flechas de conexión y los pesos en la red. Los pesos son actualizados o modificados durante el proceso de aprendizaje. La modificación dinámica de estos pesos permite a la red aprender o más bien adaptarse.

Un peso representa la fuerza de asociación, que es la co-ocurrencia de diferentes eventos durante el periodo de entrenamiento. A nivel de la red, representa qué tan frecuentemente una unidad receptora ha sido activada simultáneamente con la unidad emisora ó mensajera. De aquí se concluye que el cambio de pesos entre 2 unidades depende de la frecuencia de que ambas unidades tengan una salida positiva. Durante la fase de aprendizaje, la tarea es organizar el espacio de estado del sistema hasta llegar a estados " ejemplares ".

APRENDIZAJE SUPERVISADO:
Durante la presentación de un patrón de entrada, se especifica también el patrón de salida correcto. Es decir, la red se entrena con patrones de entrada asociados a patrones de salida, donde los patrones de salida constituyen la respuesta correcta de la red o clasificación correcta del patrón de entrada.

APRENDIZAJE NO SUPERVISADO:
La red ajusta sus pesos en respuesta a distintos patrones de entrada que estén asociados a patrones de salida. En este tipo de aprendizaje, la red clasifica los patrones de entrada en categorías similares.Existen muchos algoritmos de aprendizaje en redes neuronales, en el siguiente cuadro se muestran algunos de los más comunes.



I. 6. Redes Neuronales Artificiales con Aprendizaje Supervisado.

En el aprendizaje supervisado, la red asocia una serie de patrones de entrada a un conjunto de patrones de salida. Los escenarios de entrenamiento para redes neuronales artificiales más comúnmente utilizados, emplean este tipo de aprendizaje. Cuando las redes neuronales se entrenan de esta manera, se utilizan como clasificadores ó memorias asociativas El patrón de salida representa normalmente la respuesta correcta ó la clasificación correcta para el patrón de entrada.

El aprendizaje supervisado se puede definir de la siguiente manera: Un patrón de entrada es mostrado a la red para que ésta genere una salida determinada, esta es comparada entonces con el patrón de salida deseado y en base a la diferencia se adaptan los pesos de la red conforme a las reglas de entrenamiento.

Este procedimiento generalmente se repite un gran numero de veces durante el entrenamiento para que la red ajuste de forma gradual los pesos de las conexiones y llegue al punto en que puede producir la respuesta correcta para cada patrón de entrada.

II. MODELOS DE REDES NEURONALES

II.1. Perceptrón

El Perceptrón se introdujo como un nuevo enfoque al problema de Reconocimiento de Patrones, en él se mostraba cómo con pequeñas modificaciones, las Redes de **McCulloch-Pitts** pueden ser entrenadas para clasificar un grupo de patrones tanto similares como distintos.

Un Perceptrón es una red que se entrena bajo supervisión y que está constituida por un grupo de unidades "sensitivas" conectadas através de una capa simple de neuronas **McCulloch-Pitts**, asi como un grupo de unidades "motor", además puede ser usada tanto con valores continuos como binarios.

El perceptrón fué inventado en 1957 por **Frank Rosenblatt** (aproximadamente 10 años después de la publicación del trabajo de **Pitts** y **McCulloch**), quien posteriormente demostró un teorema llamado ***teorema de convergencia*** que define las condiciones bajo las cuales, se garantiza que es posible llegar al patrón de salida deseado. Este teorema establece que dado un conjunto de clases linealmente separables, un perceptrón puede, en un conjunto finito de intentos, desarrollar un vector de peso que separará las clases. Esto es independiente del valor inicial de los pesos.

En la figura 2 se presenta una breve descripción de un esquema muy simple de un perceptrón, en el esquema se pueden observar las siguientes partes:

- Hay una unidad de procesamiento designada por el circulo mostrado en la figura ,y que representa a la neurona i, éste circulo representa el cuerpo de la célula, al que se le puede llamar soma.

- Un conjunto de lineas de entrada conectadas lógicamente al soma. En la figura se observan como un conjunto de flechas entrando al cuerpo de la célula.

- Cada canal de entrada es la combinación de una dendrita y una sinápsis, por lo que , habrá tantos canales de entrada a una neurona como sinápsis conectadas a sus dendritas.

- Los canales de entrada son activados mediante las señales que reciben de las cajas a las cuales están
conectados estos canales (neuronas presinápticas) . Están representados por pequeñas cajas en la figura.

A cada linea de entrada se le asocia un parámetro Wij ,el subindice i representa la neurona que estamos considerando y el subindice j se refiere a una de las muchas entradas a la neurona. El valor numérico de Wij es el efecto sináptico que determina la cantidad que será sumada al soma i si el canal j está activo.

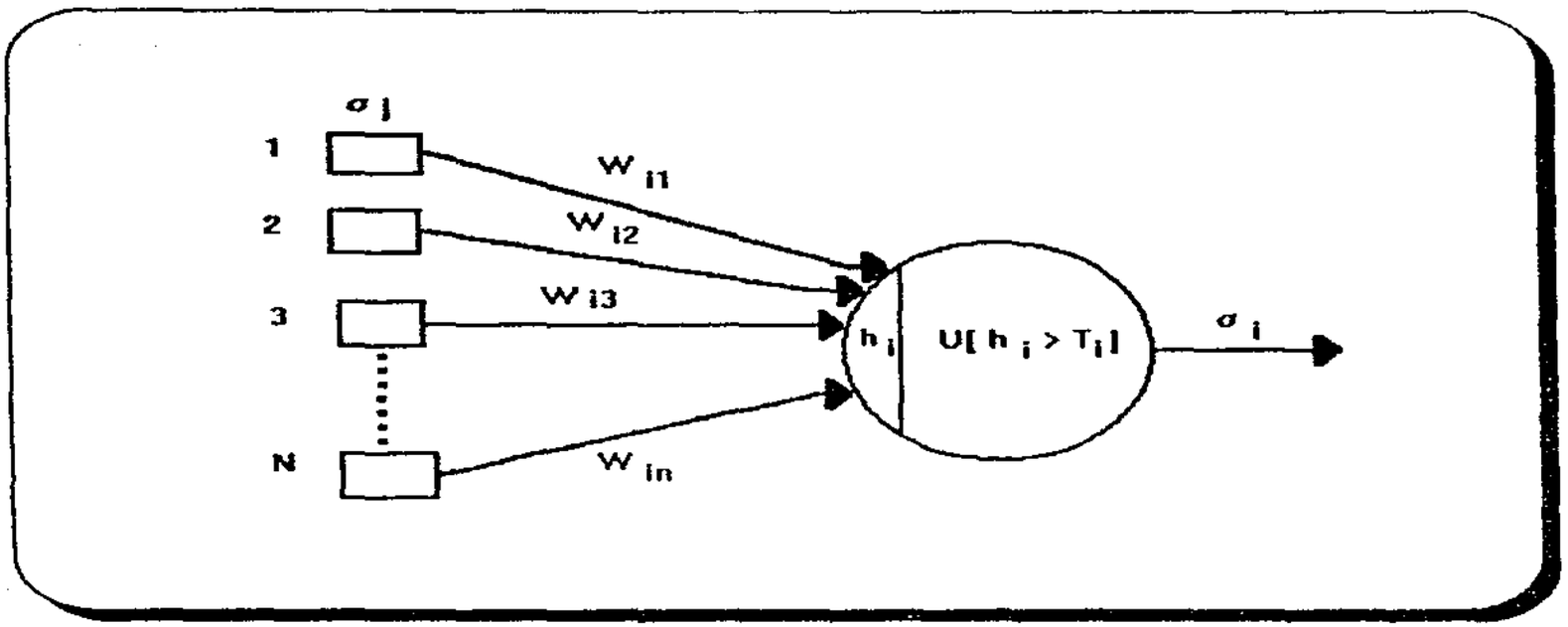


Figura 2. Estructura lógica de la neurona como un perceptrón . Wij son los pesos de cada una de las entradas que llegan a la neurona i, representada por el círculo. Las salidas de las neuronas presinápticas toman valores de 1 ó 0 y están representadas por pequeñas cajas en la figura, hi representa la suma de los pesos de las entradas que están activas y U [hi] es la función de decisión de la neurona.

En la figura σ_j representa la salida de las neuronas presinápticas y σ_i la salida de la neurona.

Hay una línea de salida única, representada por una flecha saliendo del cuerpo de la neurona en la figura. Esta línea de salida única expresa el valor de salida de la neurona que puede tomar dos valores 0 ó 1.

- La operación de la unidad es como sigue:
- En un momento determinado, algunas de las entradas lógicas están activadas.
- El soma recibe una entrada que es la suma de los pesos Wij de aquellos canales que fueron activados.
- La suma resultante es comparada con un valor de umbral de la neurona i, de esta forma, el canal de salida se activa si y sólo si, la suma es mayor que el valor de umbral. Si no lo es,no se activa.

Por lo tanto, se puede decir que la variable σ puede tomar los valores 0 ó 1, indicando si la caja a la cual está conectada,está activa ($\sigma = 1$) ó inactiva ($\sigma = 0$) .Entonces, la función hi en la figura, se expresa como:

$$h_i = \sum_{j=1}^{N} W_{ij}\, \sigma_j \quad \ldots\ldots (2.1)$$

donde: N es el número de cajas pequeñas ó neuronas presinápticas.

De esta forma, la operación de la neurona que se muestra en la figura 2, se puede expresar de la siguiente forma:

$$\sigma_i = \psi \ [\ h_i > T_i\ ] \quad \ldots\ldots (2.2)$$

Donde:

$$\psi = \begin{cases} 1 \text{ si } h_i \geq T_i \\ 0 \text{ si } h_i < T_i \end{cases}$$

La meta de un perceptrón se ilustra en la figura 2.1 , en ésta se pueden observar dos clases de puntos en el espacio de n dimensiones, a los que llamaremos de clase 0 y de clase 1 , por eso se dice que un patrón es simplemente un punto en el espacio n dimensional Las coordenadas del punto representan los atributos y caracteristicas del objeto a ser clasificado, tales como: peso, altura, densidad, frecuencia, etc. Se observa también que las dos clases pueden separarse una de la otra por un simple hiperplano lineal (en un espacio de 2 dimensiones el hiperplano es una linea,en uno de 3 dimensiones el hiperplano es un plano ordinario y en general,en un espacio de n dimensiones, el hiperplano es una superficie de n-1 dimensiones).

Las clases que tienen la propiedad de ser clasificadas se denominan linealmente separables . Entonces , el objetivo, es encontrar un conjunto de pesos o coeficientes adaptivos denominados w1 ,w2 ,w3 ,...., wn , de forma que la salida del perceptrón sea un 1 si el punto (ó patrón de entrada) , representado por un vector (x1 ,x2 ,x3 ..... ,xn) pertenece a la clase 1, y que sea 0 si el vector pertenece a la clase 0.

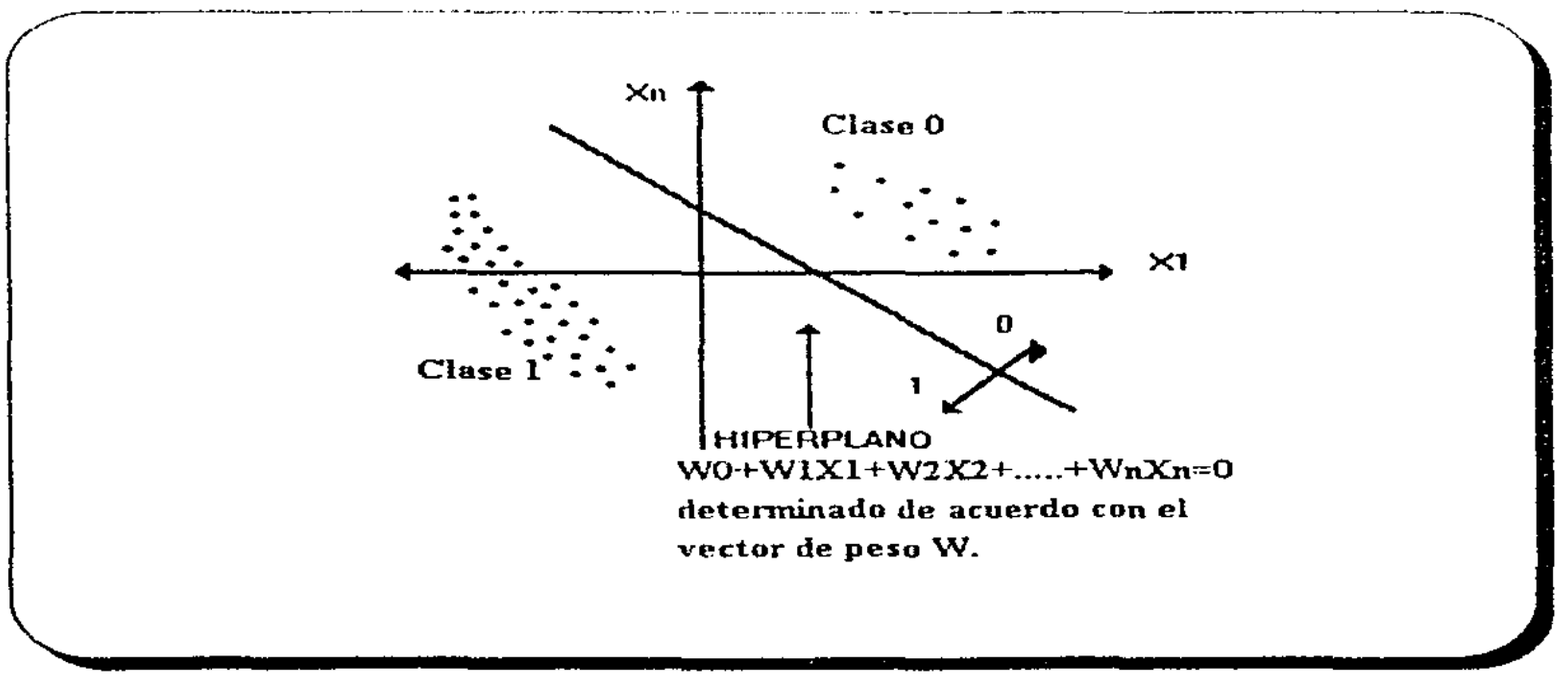


Figura 2.1. El perceptrón en este caso, funciona correctamente puesto que el hiperplano está orientado adecuadamente de forma que es posible separar los puntos en dos clases linealmente separables.

II. 1. 1. ENTRENAMIENTO:

Una vez comprendido el funcionamiento del perceptrón, y para simplificar la notación se puede considerar la figura 2.2, en base a la cual se presenta la descripción del entrenamiento de esta red neuronal.

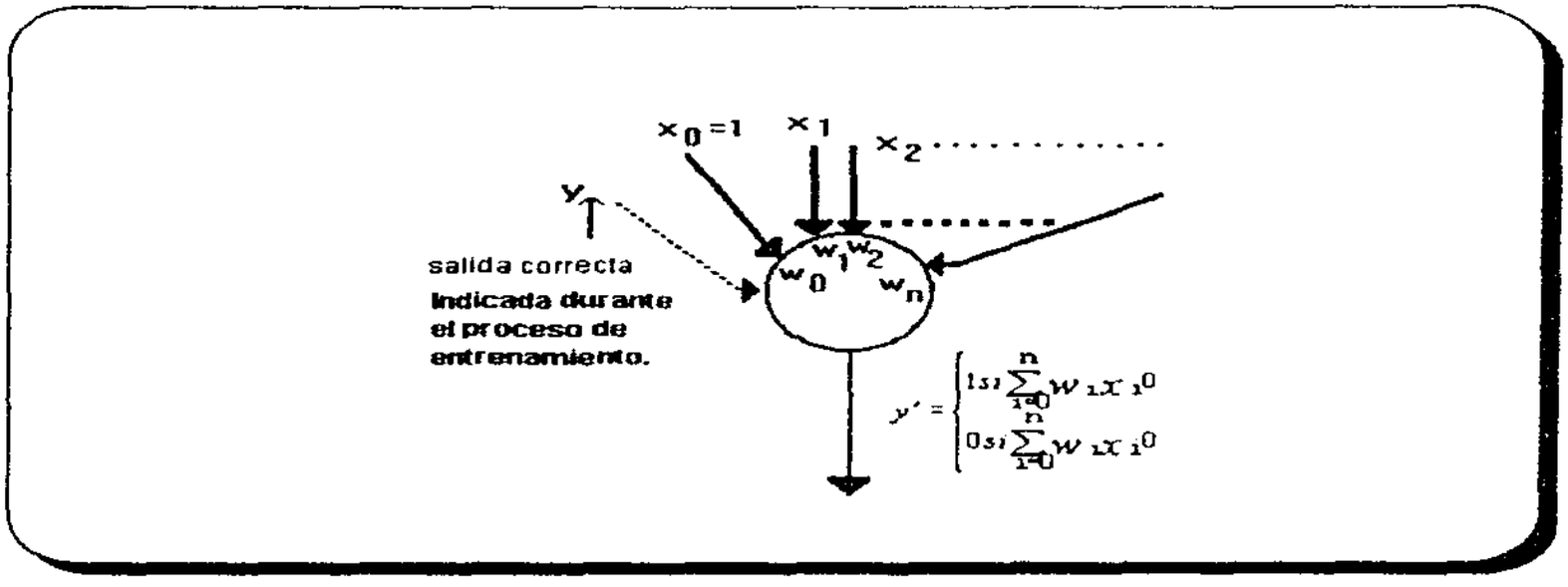


Figura 2.2. Perceptrón. El perceptrón tiene una entrada que consiste en un vector x= (x0, x1, x2, ... , x n) ,donde x está permanentemente en 1 (de forma que en la función de salida se compara con 0 en vez de comparar con un umbral). La salida del perceptrón está dada por y '.

La regla de aprendizaje de un perceptrón opera durante todo el proceso de entrenamiento, através del cual se muestran al perceptrón un conjunto de ejemplos en forma de vectores de entrada seleccionados de una forma aleatoria a los que se denomina x (se presenta uno a la vez). Cada vez que se le presenta un ejemplo, se indica también a qué clase pertenece dicho ejemplo (0 ó 1). Y en cada paso del proceso de aprendizaje, la regla de aprendizaje modifica los pesos de acuerdo con la siguiente ecuación:

$$W^{new} = W^{old} + (y - y')\,x \quad \ldots\ldots (2.3)$$

donde: y es el número correcto de la clase del patrón de entrada x y y ' es la salida del perceptrón ver (figura 2.2).

La idea de esta regla de aprendizaje es que, si el perceptrón tiene un error (y-y') en su salida, este error indica que es necesario reorientar el hiperplano w (de pesos) de forma que el perceptrón tienda a disminuir el error para el ejemplo x en particular. Se puede observar que el error (y - y ') será cero si la salida del perceptrón es correcta. En este caso, el peso no cambiará. Si la salida es errónea, entonces, (y - y') será 1 ó -1, y w será modificado en forma adecuada.

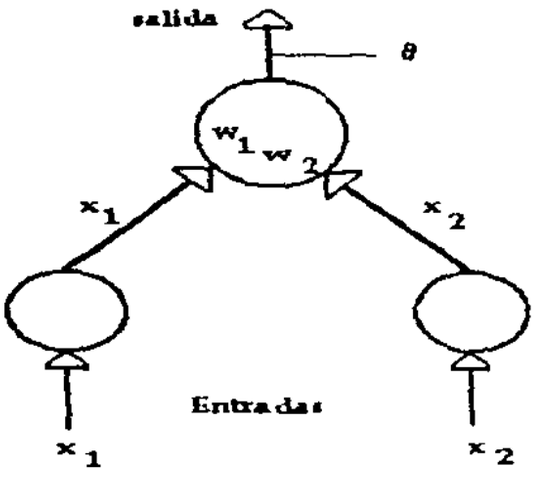
El perceptrón recibió un éxito considerable cuando se dió a conocer, debido a su simplicidad; Sin embargo, ésto no duró mucho, ya que cuando **Minsky y Papert** provaron matemáticamente que no podía ser usado para funciones lógicas complejas, su éxito terminó.

II. 1. 2. LIMITACIONES DEL PERCEPTRON.

Con el fin de observar y comprender algunas de las limitaciones del perceptrón, se presenta uno de los ejemplos más simples que no puede ser resuelto por un perceptrón como el mostrado en la figura 2.3: el problema de la XOR.

En la figura 2.3 se presenta un perceptrón que tiene dos nodos en la capa de entrada con valores x1 y x2 que pueden tomar los valores 0 ó 1. Se desea que la red responda a las entradas de forma que la salida sea

la función XOR de las entradas como se indica en la tabla.



x_1	x_2	salida
0	0	0
0	1	1
1	0	1
1	1	0

Figura 2.3. Perceptrón. Problema XOR.

En la figura 2.4 se pueden ver el plano x1, x2 los cuatro puntos (0,0) , (0,1) , (1,0) y (1,1) que forman parte de la salida de la XOR, según se observa en la tabla. La línea.

$$\theta = w_1 x_1 + w_2 x_2 \quad \ldots\ldots (2.4)$$

divide al plano en dos regiones pero , sin embargo , no puede separar en forma correcta los puntos (0,0) y (1,1) de los puntos (0,1) y (1,0)

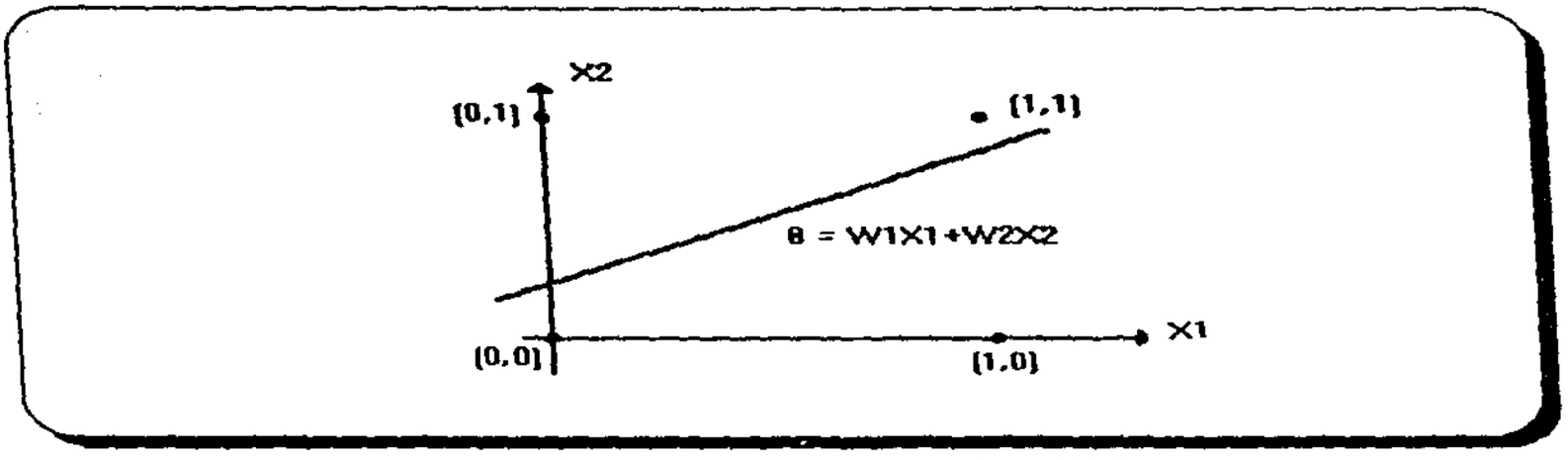


Figura 2.4. Perceptrón. Problema XOR

divide al plano en dos regiones pero, sin embargo, no puede separar en forma correcta los puntos (0,0) y (1,1) de los puntos (0,1) y (1,0). En este caso, el problema puede ser resuelto añadiendo una capa oculta ó capa intermedia en la red de la figura 2.3 resultando la red de la figura 2.5.

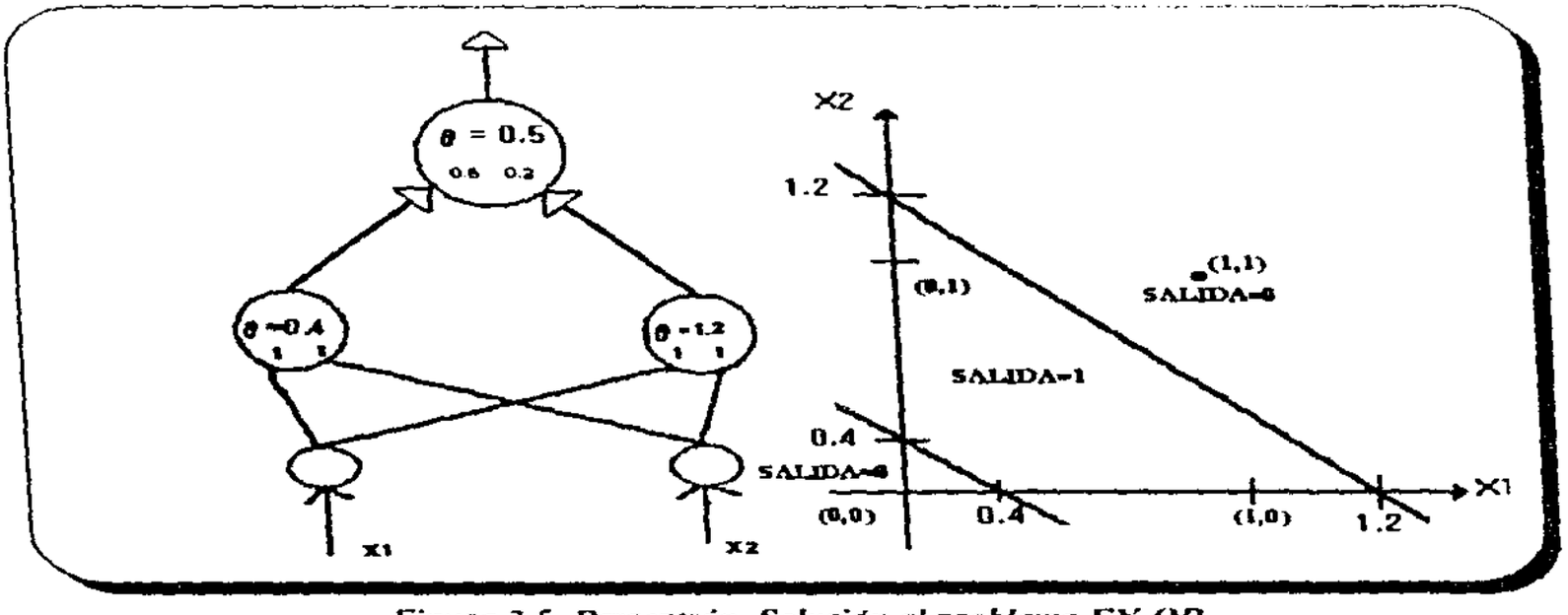


Figura 2.5. Perceptrón. Solución al problema EX-OR.

En este caso , la capa intermedia provee dos lineas que pueden ser usadas para separar el plano en 3 regiones, de forma que las 2 regiones que contienen a los puntos (0,0) y (1,1) se asocien con la salida 0 de la red. Y la región central esté asociada con la salida 1 de la red. Entonces, el añadir la capa intermedia en esta red, permitió a la red resolver el problema de la XOR. Sin embargo, es conveniente mencionar que no siempre el añadir nuevas capas intermedias nos lleva a la solución del problema

II. 2. PERCEPTRON MULTI-CAPA .

Son redes " **feedforward** " con una ó más capas entre los nodos de entrada y de salida. Estas capas adicionales contienen unidades ocultas ó nodos.

Los perceptrones multi-capa superan muchas de las limitaciones de los perceptrones de capa simple. Las capacidades de los perceptrones multi-capa provienen de las no linealidades usadas dentro de los nodos. Si los nodos son elementos lineales , entonces una red de una sola capa con pesos apropiados , podría duplicar lo cálculos ejecutados por una red multi-capa Un perceptrón de una capa forma regiones de decisión semi-planas, y uno de 2 capas puede formar regiones convexas en el espacio extendido sobre las entradas.

II. 3. ADALINE Y MADALINE

ADALINE Y MADALINE ó Múltipl e-Adaline fueron desarrolladas por **Bernard Widrow** en la Universidad de Stanford. En el aprendizaje de Widrow el objetivo principal es, encontrar el mejor vector de pesos desde un tipo simple de elementos de proceso. La ley de aprendizaje de Widrow es una de las más poderosas en Neurocomputación , ya que converge a los valores óptimos de los vectores de pesos desde cualquier valor de entrada El término de ADALINE ha sufrido cambios desde su creación , inicialmente fue ADAptive LInear NEuron, que cambió a ADAptive LINear Element cuando las redes neuronales surgieron en 1960.

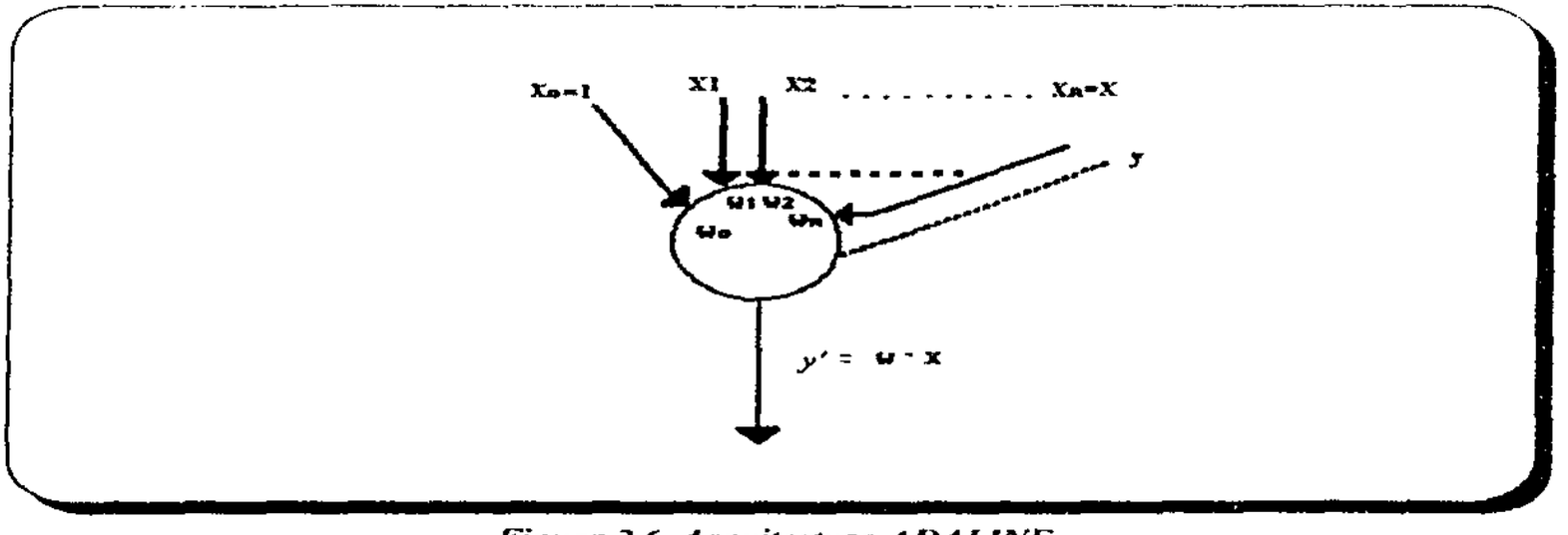


Figura 2.6. Arquitectura ADALINE.

La figura 2 6 muestra la arquitectura ADALINE. El vector **x= (x0 , x1 , x2 , ... , xn)** es introducido a los elementos de proceso, y el número **y ' = w*x =w0x0+ w1x1+.....+wnxn** es emitido, donde **w = (w0 , w1 , w2 ,.....,wn)** es el vector de pesos de los elementos de proceso de ADALINE. Cabe hacer notar que el componente cero del vector x (x0) es siempre igual a 1.

ADALINE es una estructura simple de elementos de proceso que tiene un vector real x como entrada y un número real y ' como salida y que utiliza la ley de Widrow.

La entrada de ADALINE es **x = (x0 , x1 , x2 ,....., xn) t** , donde la entrada **x0** , llamada término de bias ó entrada de bias , es siempre igual a 1. ADALINE tiene un vector de pesos w, y la salida del elemento de proceso de ADALINE es y ' .

La suma de los pesos de entrada **w1 x 1 + w2 x 2 + w3 x 3 + wn x n** es conocida como una combinación lineal. Si nosotros sumamos al término de bias **w0** como una combinación lineal, obtendremos lo que es llamado combinación afin. Es por esto que ADALINE es llamada algunas veces, combinador afin (particularmente cuando la conexión con la Ley de Widrow no ha sido enfatizada).

II. 3. 1. ENTRENAMIENTO

La ley de aprendizaje de Widrow (comúnmente conocida como la ley de aprendizaje de Widrow-Hoff, o la ley LMS, ó bien la regla delta) es la siguiente

$$w_{k+1} = w_k + \alpha \delta_k x_k \quad \ldots\ldots (2.5)$$

donde: α es tipicamente seleccionada por prueba y error.

En general, si δ es muy grande el vector de pesos no converge,en cambio si es muy pequeña, la convergencia es más tardada. Tipicamente tiene valores entre 0.01 y 10.00; con un valor de 0.1 frecuentemente usado como valor inicial. Escoger y ajustar este valor es un arte que el interesado en neurocomputación debe aprender.

ADALINE tiene algunas semejanzas con el perceptrón, por lo caul tiene limitaciones parecidas a las de éste, por ejemplo: una estructura ADALINE de dos entradas no puede realizar la función XOR. Combinando varias Adalines en capas estructuradas, podemos solucionar este problema, creando así la estructura MADALINE.

MADALINE puede ser presentada con un vector de entrada de dimensión larga. Con el entrenamiento, la red puede dar la respuesta con un número binario +1 sobre algunos nodos de salida, cada uno de los cuales corresponde a diferentes categorias de la entrada.

Para entrenar a la red, se necesita aplicar el algoritmo de LMS a la capa de salida cuando se está entrenando a la red, teniendo identificados previamente los patrones de entrada, el vector de salida deseado es conocido. Lo que nosotros no conocemos es la salida deseada para dar a los nodos en una de las capas intermedias

El algoritmo LMS puede operar en salidas análogas (ALC), no en los valores de salida bipolar de ADALINE. Por estas razones se han desarrollado diferentes estrategias de entrenamiento para ADALINE

Por estas razones se han desarrollado diferentes estrategias de entrenamiento para MADALINE. Varias estructuras de MADALINE han sido usadas recientemente, para demostrar la aplicación de esta arquitectura para adaptar patrones de reconocimiento, teniendo propiedades de invarianza en traslación, en rotación y escala.

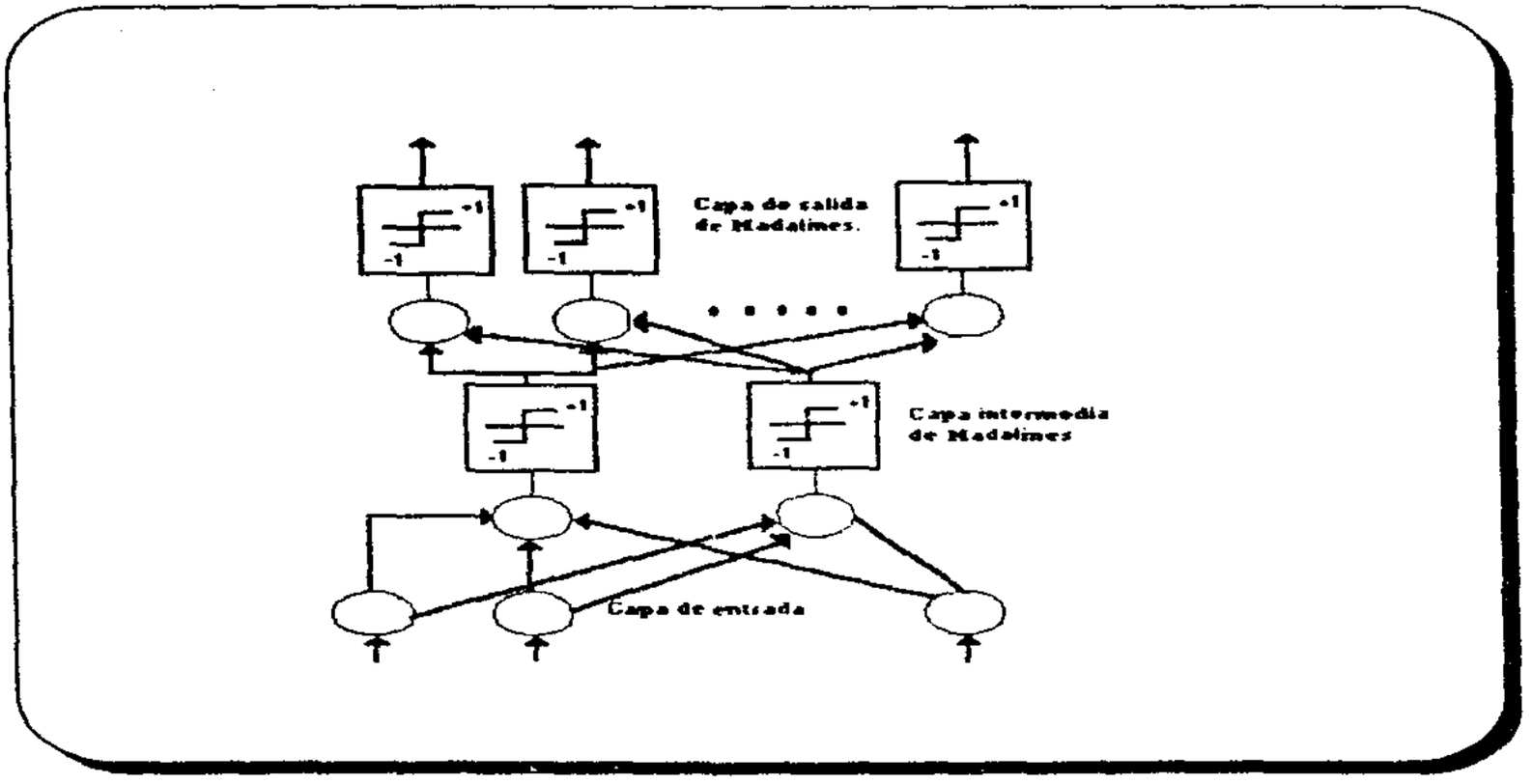


Figura 2.7 Estructura MADALINE.

Estos sólo son algunos modelos con los que cuenta la teoría de Redes Neuronales para la resolución de problemas , la elección de alguno de ellos dependerá de la complejidad del problema en cuestión, es decir, para realizar una correcta selección se debe analizar los requerimientos del problema: datos de entrada, salida a la que se debe llegar, etc.

III. METODOLOGIA DE DESARROLLO.

El avance en cuanto a computación se refiere ha dado la pauta para automatizar procesos complicados que antes eran muy difíciles de manejar, elaborando algoritmos que la computadora pueda ejecutar utilizando sus recursos, a estos algoritmos se les ha denominado de varias maneras una de ellas es ***sistemas*** .

El término de ***sistema*** se refiere a un grupo organizado de componentes relacionados funcionalmente entre sí. Un sistema existe debido a que es diseñado para alcanzar un objetivo.

Para su elaboración así como para su mejora (en el caso de que el sistema ya exista), se requiere de una serie de pasos a seguir que lo que buscan es construir un sistema que cumpla con las características definidas al momento de su concepción.

Todo sistema al igual que los seres humanos consta de un ciclo de vida que empieza desde el momento en que surge la idea de su creación hasta el momento de su puesta en marcha .

Existe una amplia gama de alternativas para la creación de un sistema por lo que es sumamente importante plancarlo con cuidado tomando en cuenta todos los aspectos involucrados, requiriéndose de una secuencia detallada de los pasos a considerar

El enfoque de sistemas (ES) es posiblemente la técnica mas utilizada para el estudio de los sistemas. El **ES** es un proceso de desarrollo ordenado y analítico que se puede utlizar continuamente para analizar, evaluar y diagnosticar la naturaleza de un sistema, así como los resultados de su desempeño para captar todo lo necesario a esos fines y proveer la continua autocorrección del funcionamiento del sistema con el propósito de alcanzar los objetivos propuestos.

El proceso de desarrollo puede resumirse en las siguientes etapas o pasos :

- *Análisis*
- *Diseño*
- *Desarrollo*
- *Instrumentación*
- *Evaluación*

El **ES** se basa en los 5 puntos mencionados anteriormente; las actividades que se desarrollan en cada paso se pueden resumir en forma general de la siguiente manera:

1. **Análisis**
   - Representación o caracterización del sistema
   - Se describen detalles concretos y útiles acerca del sistema.
   - Se definen y analizan las entradas.
   - Se describen los procesos y sus características.
   - Las salidas se definen , identifican y cuantifican.
   - La estructura del sistema se describe para aclarar las relaciones entre los componentes.
   - El ambiente se define para aclarar su interacción con el sistema.
   - El flujo de información se define y en especial los mecanismos de realimentación.
   - Finalmente, se estudia la relación entre las salidas del sistema y los objetivos y metas.

2. Diseño

- Con el análisis se han descubierto fallas en el sistema y el paso siguiente es diseñar nuevos métodos para el sistema. Según sea la naturaleza del problema, el diseño puede comprender
  - Un nuevo sistema
  - Cambio de componentes
  - Cambio en las entradas y/o salidas
  - Cambios en los procesos
- Para llevar acabo el diseño,se prepara un plan detallado con todas las especificaciones requeridas para modificar al sistema.

3. Desarrollo

- Sobre la base de los diseños se procede a la construcción, edificación, explicación o lo que sea necesario para el desarrollo del sistema.

4. Instrumentación

- Una vez que se ha desarrollado el nuevo componente, elemento, proceso ó sistema debe incorporarse a lo ya existente

5. Evaluación

- Después de haber instrumentado el nuevo componente este debe evaluarse.
- Siendo necesario contestar a las siguientes preguntas : ¿Logró el nuevo componente resolver los problemas planteados?, ¿Provocó nuevos problemas? . La evaluación debe hacerse en términos de los objetivos que se establecieron en las etapas de análisis y diseño.

En cualquier plan para el desarrollo de sistemas de información , encontramos que los pasos del ES gobiernan de forma general el proceso. Este plan también es llamado "ciclo de vida del desarrollo de un sistema"

Las fases en el ciclo de vida para el desarrollo de sistemas de información son explicados por varios autores, pero las diferencias están principalmente en la cantidad de detalle con la que son descritos. Dependiendo de las diversas corrientes que existen para el desarrollo de sistemas, se mencionarán a continuación los siguientes pasos:

Según Elias M. Awad .-

1. Anális del sistema

- Definición inicial del problema
- Recolección de datos
- Organización de los datos
- Análisis de costo beneficio

2. Diseño

- Diseño de salidas
- Diseño de entradas
- Diseño de archivos
- Diseño de procesos
- Documentación

3. **Pruebas e implantación**
   - Codificación de los programas
   - Preparación de datos de prueba
   - Corrida en paralelo.

En cambio, para **Kenniston W. Lord y James B. Steiner** la apreciación de las etapas a seguir por el ciclo de vida del sistema es la siguiente :

1. **Análisis del sistema**
2. **Diseño del sistema**
3. **Selección y adquisición del equipo**
4. **Programación**
5. **Prueba y Conversión**
6. **Instalación**
7. **Operación**
8. **Mantenimiento**
9. **Evaluación**

III . 1 Análisis del Sistema.

En realidad, es difícil que sólo uno de los métodos existentes de diseño se ajusten perfectamente tanto al tipo de proyecto, características, así como los recursos de que se dispone, por lo tanto en este trabajo se utilizó una mezcla de las metodologías citadas anteriormente entre el análisis, construcción de prototipos y el ciclo de vida del sistema

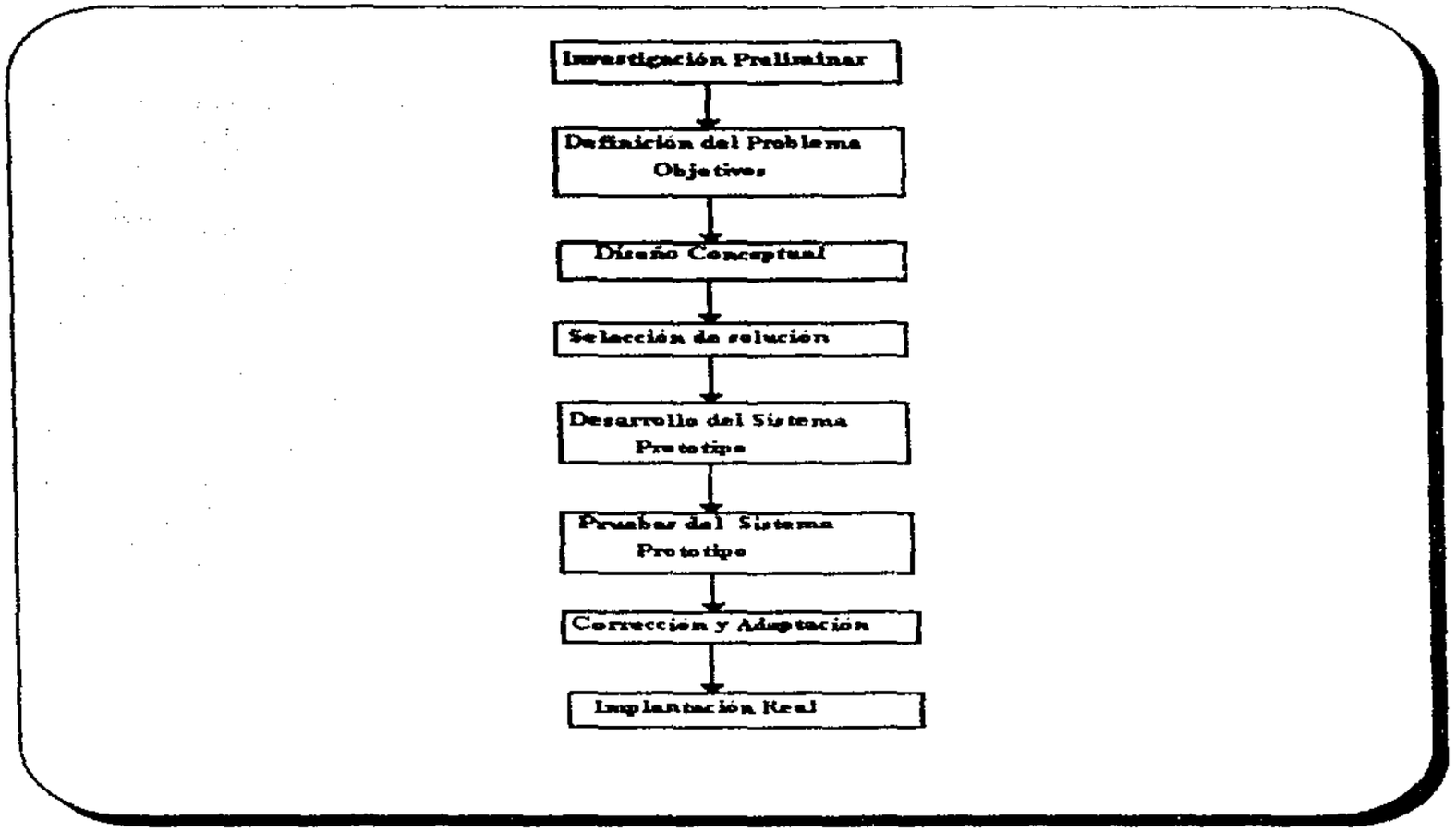


Figura 3.1. Metodología a seguir en el desarrollo de los Sistemas de este trabajo.

III . 1. 1. Definición del problema.

El estudio de la actividad sísmica es de gran importancia debido a que la República Mexicana está situada en una de las regiones sísmicamente más activas del mundo, es por esto que al ocurrir un evento sísmico es necesario conocer la zona en la que se originó para su posterior estudio.

Para la localización de un evento sísmico es necesario extraer algunos parámetros de la señal sísmica como lo son : *Fase " P ", Fase " S ", amplitud , duración, etc*

La extracción de estos parámetros es llevada acabo por un analista experto quien debe identificarlos. La lectura de estos datos en algunos casos no es tan sencilla debido a que las señales sísmicas son tan variadas y pueden contener interferencia como ruido, haciendo que la señal no sea lo suficientemente clara para ser identificada confundiéndose el arribo de la fase sísmica y causando por ende una localización errónea del evento.

Por lo anterior es importante establecer que la identificación del arribo sísmico es un punto sumamente importante puesto que está sujeto a la apreciación del analista.

Razón por la cual se necesita de un método que pueda ayudarlo a discernir entre el primer arribo de un evento sísmico para su posterior localización y estudio. Para resolver el problema expuesto anteriormente será necesario en principio, elaborar el sistema que estará encargado de la depuración de la señal sísmica, posteriormente se decidirá el modelo de Red Neuronal a ser utilizado, en base a varios criterios de selección que se describirán en el capitulo IV sección 7, esto servirá como pauta para la elaboración del algoritmo que se encargará del aprendizaje y reconocimiento de las señales sísmicas , lo que constituirá el sistema de la Red Neuronal.

III . 1. 2. Objetivo General.

Desarrollar una Red Neuronal que identifique el primer arribo de un evento sísmico con la finalidad de ayudar al analista experto a determinarlo en caso de que la señal no sea muy clara, por lo tanto la Red Neuronal será entonces una herramienta de apoyo para la realización de su labor.

Para esto es necesario elaborar un sistema que preprocese las señales sísmicas de tal manera que se tenga a la salida una señal lo suficientemente depurada para ser la entrada a la Red, esta es la razón por la que el sistema tiene que cumplir con una serie de Requerimientos establecidos al momento de su concepción.

Es pertinente mencionar que el Análisis de los eventos sísmicos digitales es llevado acabo en la estación de trabajo, por lo tanto se decidió que este trabajo debería correr bajo la misma plataforma pretendiendo con ésto globalizar el sistema.

III . 1. 3. Determinación de Requerimientos.

Para definir como debe funcionar un sistema es necesario determinar los requerimientos, para lo cual es primordial tener una amplia comunicación con los usuarios del sistema, es decir, se debe trabajar conjuntamente con ellos para identificar lo que se debe satisfacer, pretendiendo con ésto que el sistema sea lo más eficiente posible.

III . 1. 3. 1. Requerimientos del Sistema de Preprocesamiento de eventos sísmicos.

a) El sistema debe implementarse en la estación de trabajo SUN SOLARIS (equipo con el que cuenta la Coordinación de Ingeniería Sismológica en el Instituto de Ingeniería U.N.A.M quien proporcionó los recursos para la elaboración de este trabajo) . Este equipo corre bajo la plataforma de UNIX y cuenta con lenguaje de progamación C . Por lo tanto , el sistema debe ser programado bajo esta herramienta.

b) El sistema debe tener la capacidad de leer la información correspondiente a la señal sísmica y segmentarla en un número determinado de patrones (el cual dependerá del tamaño de la señal), a su vez cada uno de estos patrones contendrá un numero de muestras (el cual será determinado con base en pruebas realizadas posteriormente). Es pertinente mencionar que los archivos que se tomarán como entrada al sistema de preprocesamiento contienen valores de amplitud de la señal sísmica.

c) Se debe tener la opción de seleccionar entre diferentes razones de diezmado pretendiendo con ésto disminuir el número de datos de la señal para que el tiempo de ejecución sea lo más óptimo posible, es pertinente mencionar que al diezmar la señal sigue conservando sus caracteristicas .

d) Opción de seleccionar la traza a ser procesada así como la opción para deslizar los patrones tantas muestras como se indique, con la finalidad de localizar el arribo de la fase P lo más cercano posible.

e) Se debe contemplar una fase donde se le reste a cada dato de la señal original un valor correspondiente al ofsett(definido en otro procedimiento). Asegurando con esto que la señal se encuentre al mismo nivel .

f) Como la señal sísmica consta de valores tanto positivos como negativos, es necesario contar con una fase que dé la posibilidad de tener únicamente valores positivos, puesto que al realizar cualquier proceso los valores negativos constituirían un problema.

g) Determinar algunos algoritmos para preprocesar las señales de tal manera que se tenga una señal de salida lo más depurada posible para un fácil entrenamiento de la Red Neuronal, el algoritmo seleccionado se utilizará tanto en la fase de aprendizaje como en la de reconocimiento.

h) Normalización de la señal .

i) Categorizar la señal en la fase de aprendizaje para tener la facilidad (dentro del algoritmo de la red) de supervisar si la salida obtenida es igual a la señal esperada, el valor de categorización se determinará con base en pruebas realizadas posteriormente

j) Opción para seleccionar la fase para la cual se preprocesará la señal, es decir, fase de aprendizaje ó de reconocimiento.

k) La salida del preprocesamiento debe ser grabada en un archivo de texto con un cierto formato para alimentar al algoritmo de la Red El formato será el siguiente.

i) Patrón 1 (muestra1, muestra2, . . , muestra n) ***Valor de categorización para la salida del patrón 1.***
Patrón 2 (muestra1, muestra2, . . , muestra n) ***Valor de categorización para la salida del patrón 2.***
..... , **Patrón N** (muestra1, muestra2, . . . , muestra n) ***Valor de categorización para la salida del patrón N.***

l) Al finalizar el preproceso se deben mostrar ciertos datos como: Nombre del archivo procesado, traza(s) procesada(s), nombre del archivo de salida, así como todos los parámetros que se seleccionaron para realizar el proceso.

Habiendo establecido los requerimientos que cumplirá el sistema de preprocesamiento, es importante igualmente definir los correspondientes al algoritmo de la Red Neuronal. Cabe hacer notar que cada uno de estos sistemas constituye una parte importante en la globalización de este trabajo

III . 1. 3. 2. Requerimientos del Sistema de la Red Neuronal.

a) El modelo de Red Neuronal seleccionado debe ser programado mediante Lenguaje C, es decir, tiene que correr bajo la misma plataforma mencionada en pasos anteriores.

b)El sistema debe estar en posibilidad de leer el formato de salida del sistema de preprocesamiento de eventos sísmicos, leyendo del encabezado datos como: el número de patrones a ser leídos, número de muestras por patrón, número de trazas procesadas, e Indicación de la fase para la cual los datos fueron procesados pudiendo ser Aprendizaje ó Reconocimiento.

c) Se debe tener la opción mediante la interface gráfica para seleccionar factores importantes para la ejecución del proceso de aprendizaje de la Red Neuronal como lo son: Razón de aprendizaje, Razón de momentum, error máximo permitido para cada patrón así como error máximo global para todo el proceso de los patrones. Dichos factores facilitarán u obstaculizarán el encontrar el estado óptimo necesario para lograr un buen reconocimiento del patrón presentado.

d) Al igual que el paso anterior se debe tener la opción mediante la interface gráfica para seleccionar los parámetros adecuados para determinar la Arquitectura de la Red Neuronal como lo son: Número de capas ocultas, número de unidades en la capa de salida, así como el número máximo de iteraciones a realizar. Cabe mencionar que de la buena elección de estos parámetros depende también obtener un buen desempeño del algoritmo.

Por lo tanto se determinó que la petición de cualquier parámetro al usuario se debía hacer por medio de la creación de ventanas hecha a través de la interface gráfica facilitada por OPENWINDOWS.

e) Se debe contemplar una etapa en la que se especifique si se quiere aprender ó reconocer cierto evento sísmico

f) Así mismo una etapa de presentación de los datos más importantes del archivo de entrada (que a su vez es el archivo de salida del sistema de preprocesamiento de los eventos sísmicos), pretendiendo con esto mantener informado al usuario final para que corrobore los datos proporcionados.

g) Se debe contar con la facilidad de elegir el evento sísmico de aprendizaje que sea similar al que se pretende reconocer, utilizando para esto la matriz de pesos a la que se llegó con el sismo seleccionado Con esta opción se pretende ahorrar tiempo logrando que el sistema sea más eficiente y a la vez que se obtenga un mejor reconocimiento del evento .

h) Si se seleccionó la fase de reconocimiento se debe estar en la posibilidad de proporcionar el nombre del archivo procesado para dicha fase.

i) Para la etapa de Reconocimiento se debe realizar sólo una vez el ajuste de la matriz de pesos, es decir, con los valores a los que se llegó se debe calcular el valor de salida del patrón presentado para deducir si el evento pudo o no ser reconocido.

j) Al finalizar el proceso se deben mostrar datos como: Fecha y hora en la que ocurrió el sismo, el segundo en el que ocurrió el arribo de la fase P, los factores que determinaron el buen desempeño del algoritmo así como si se tuvo éxito con el reconocimiento.

III . 1. 4. Diseño Conceptual.

El diseño conceptual es un bosquejo que señala las características del sistema, así como la función e interacción que guarda con otros elementos y su entorno.

El uso del diagrama de flujo hace fácil visualizar las relaciones entre los diferentes elementos que constituyen el sistema para explorar formas que mejoren tanto la eficiencia como el entendimiento del mismo.

A continuación se presenta el diagrama general del sistema figura 3.2, considerando la interface gráfica

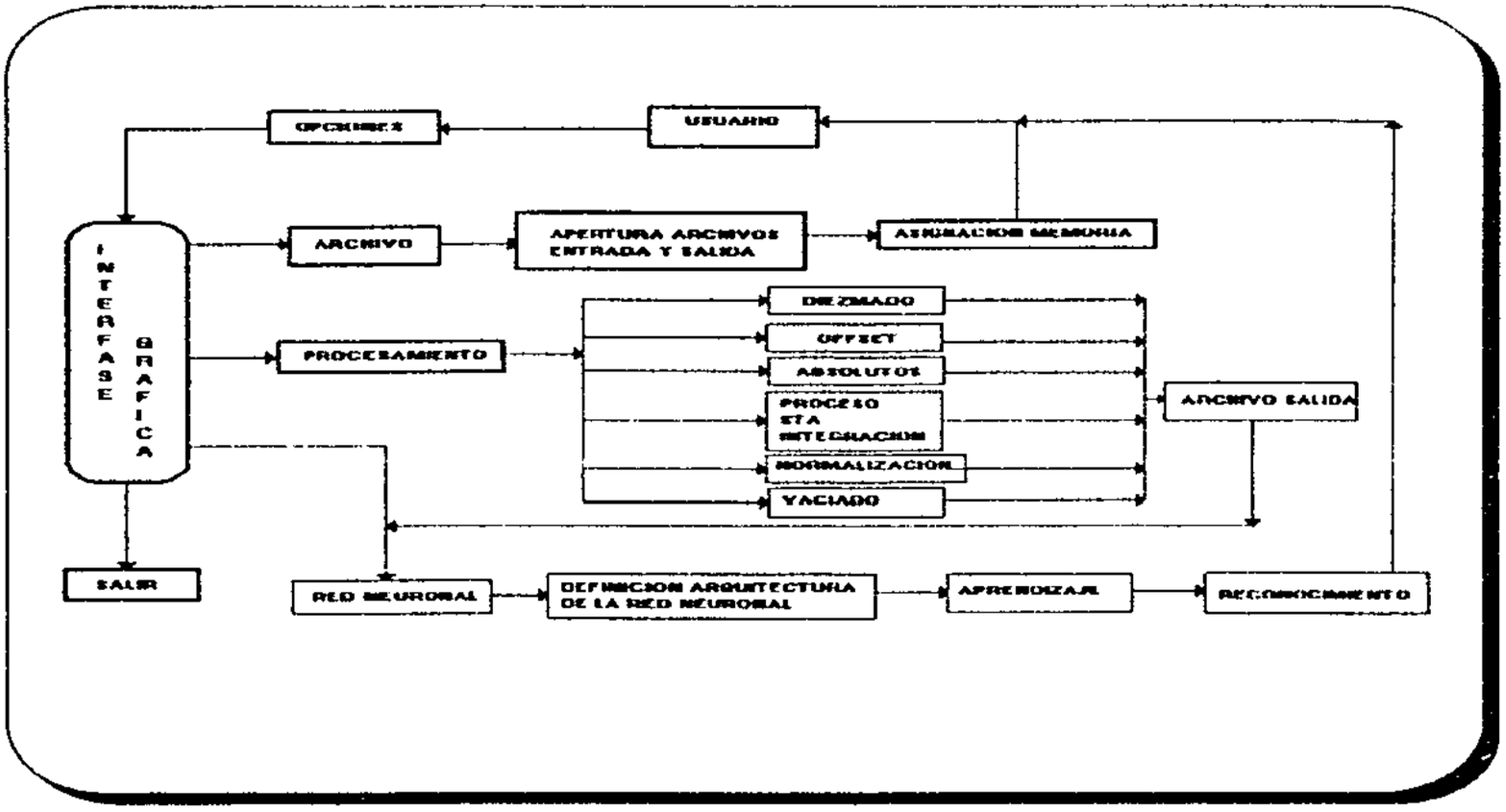


Figura 3.2. Diagrama general del sistema.

En este diagrama se conjuntan los dos sistemas , tanto del preprocesamiento como el de la Red Neuronal. Es pertinente mostrar esquemáticamente las funciones que cada uno de los sistemas cubre por separado para tener una visión más general de él.

III . 1. 4. 1. Diagrama del sistema de preprocesamiento de eventos sísmicos.

Primeramente se va a mostrar el diagrama general del sistema de preprocesamiento considerando la interface gráfica (figura 3.3).

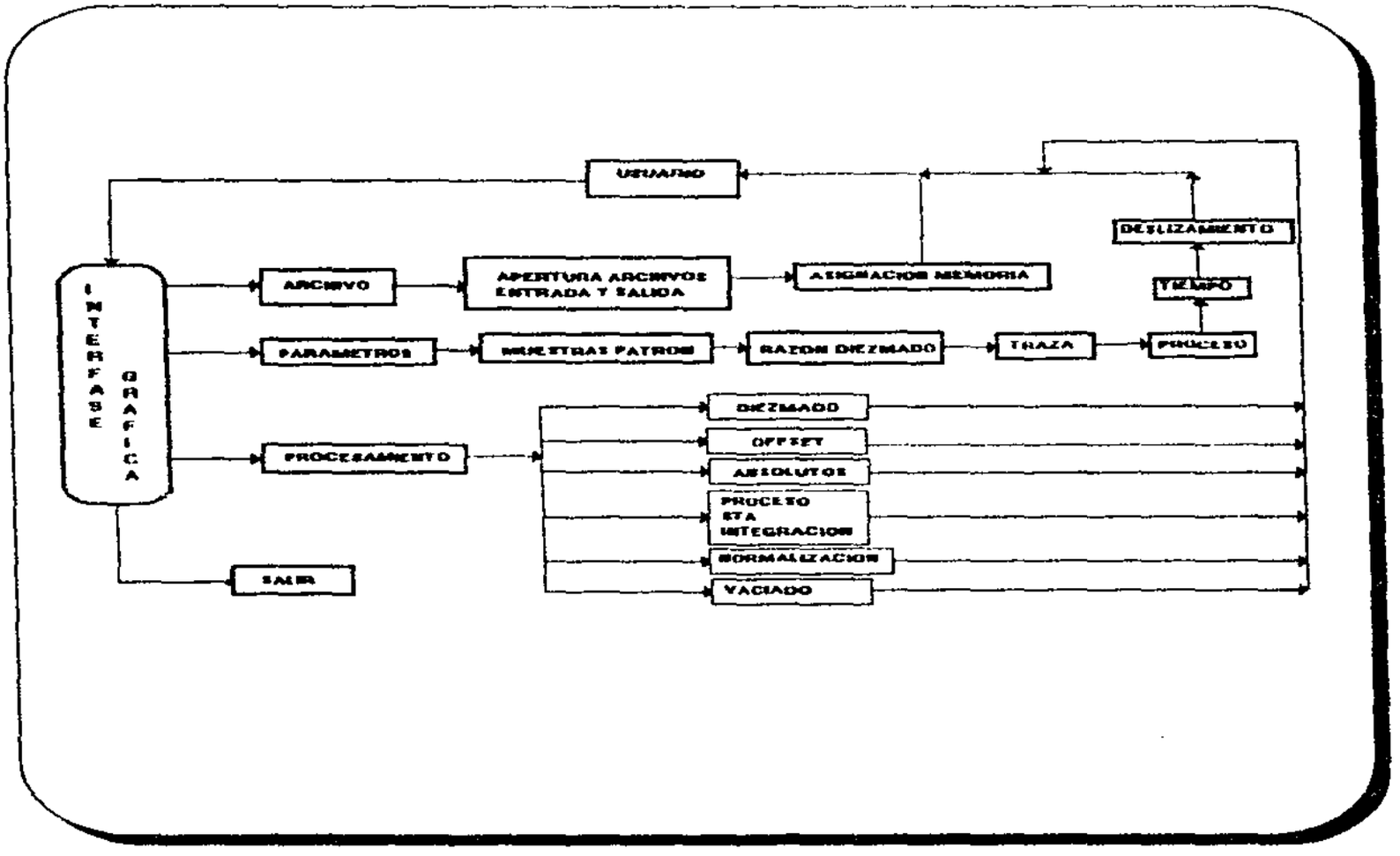


Figura 3.3. Diagrama de flujo del sistema de preprocesamiento considerando la interface gráfica.

Ahora bien, en el diagrama de la figura 3.4 se mencionan a grandes rasgos las étapas que constituyen al sistema de preprocesamiento.

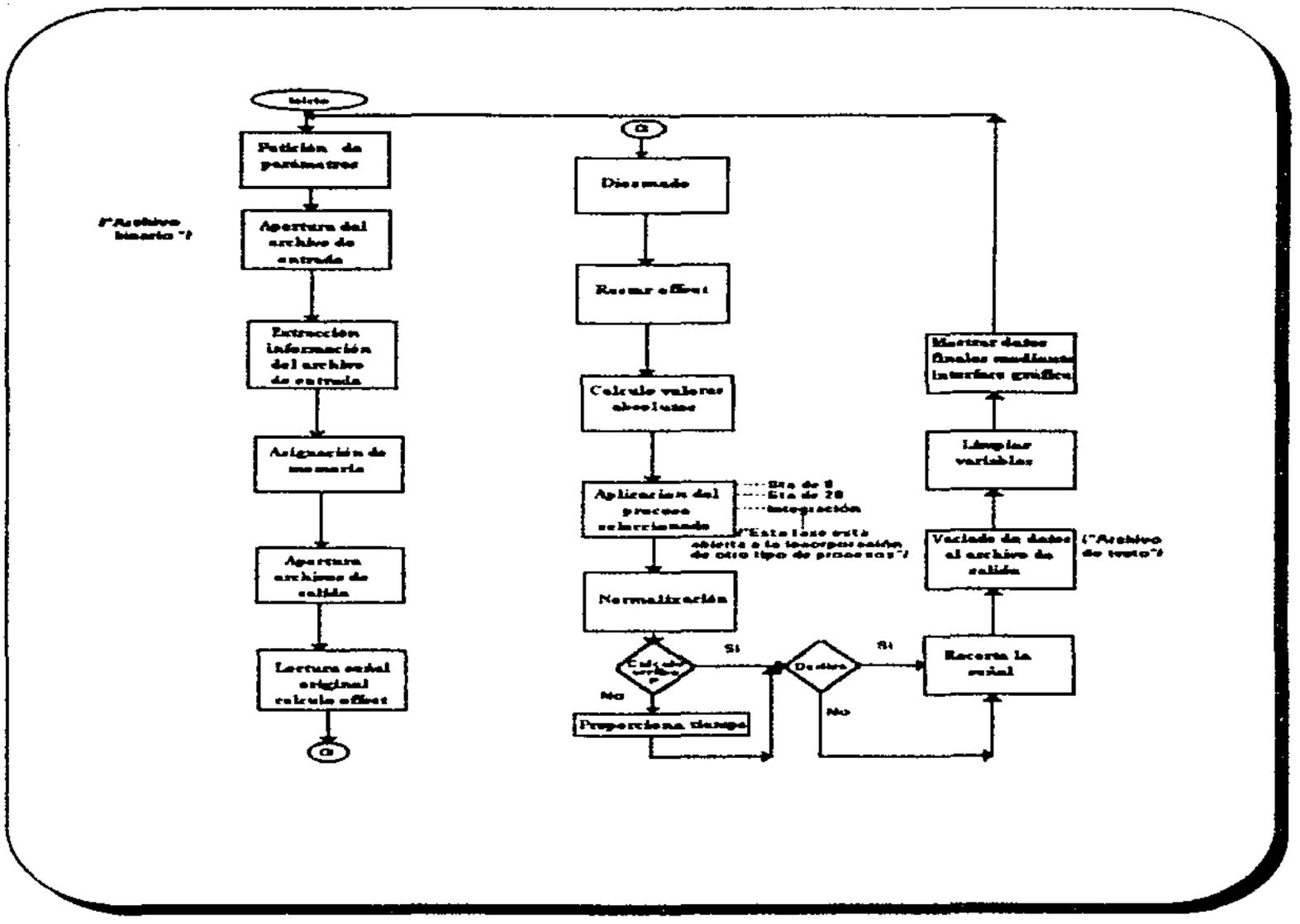


Figura 3.4. Diagrama de flujo del sistema de preprocesamiento mostrando las étapas que lo constituyen.

III . 1. 4. 2. Diagrama de la Red Neuronal.

En el diagrama 3.5 se muestra el sistema de la Red Neuronal tomando en cuenta la interface gráfica, la cual crea el ambiente de ventanas para interactuar de una manera más amigable con el usuario final.

En esta figura se ejemplifica la interacción que guarda el sistema con la facilidad gráfica con que se cuenta.

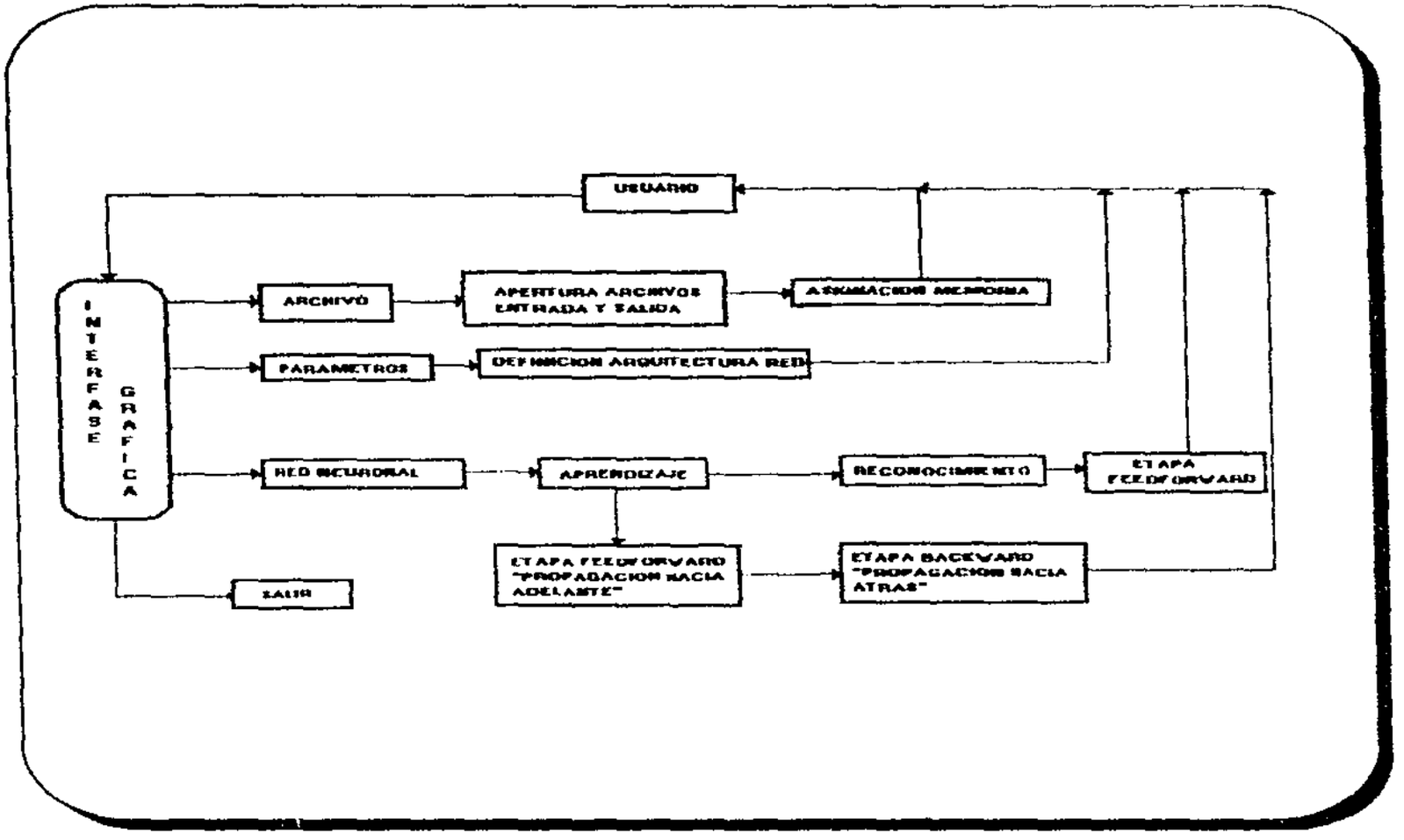


Figura 3.5. Diagrama de flujo del sistema de la Red Neuronal tomando en cuenta la interface gráfica.

Por otra parte, el diagrama de la figura 3.6 se mencionan las étapas de que consta la Red Neuronal

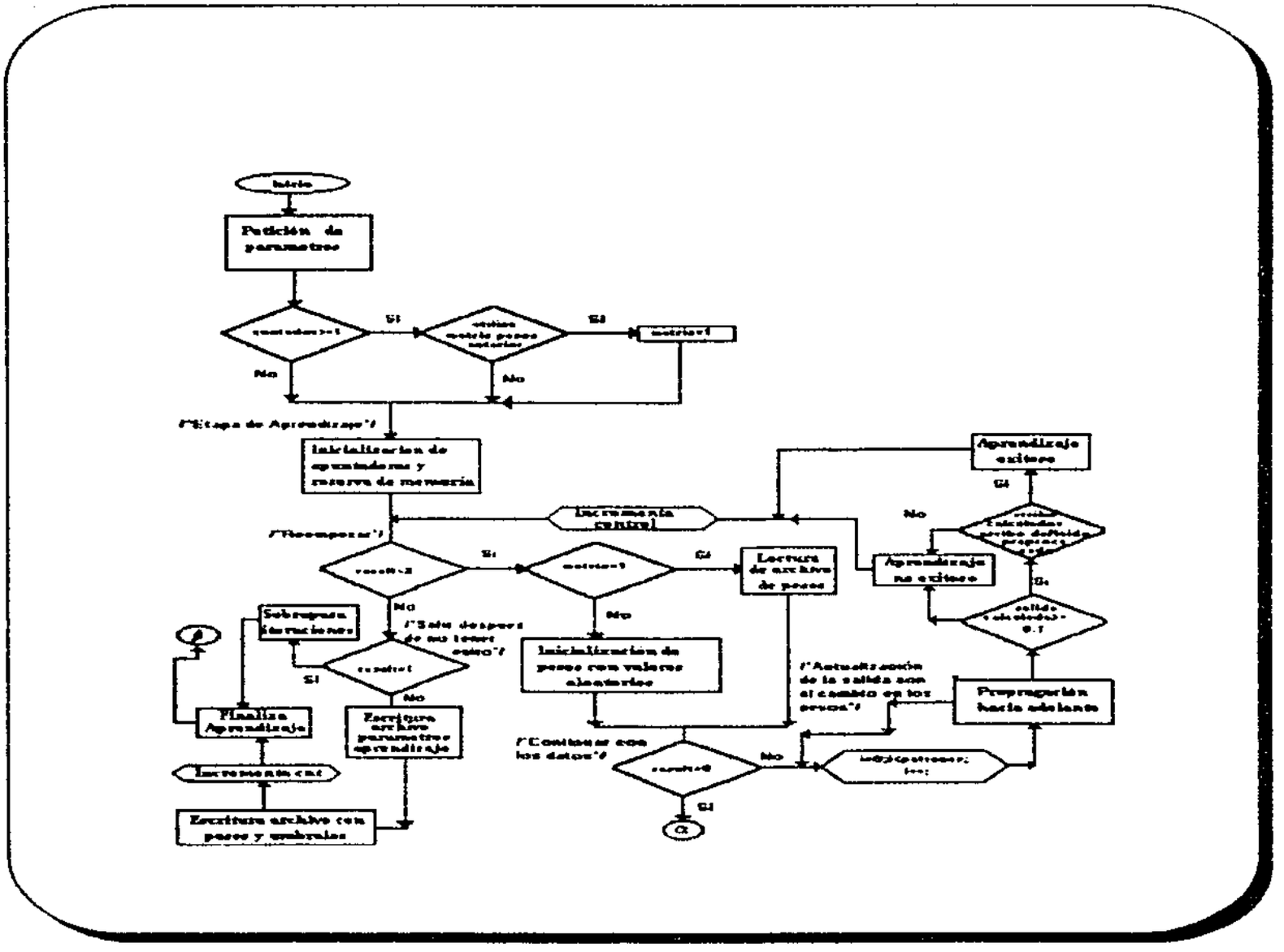


Figura 3.6 Diagrama de la Red Neuronal.

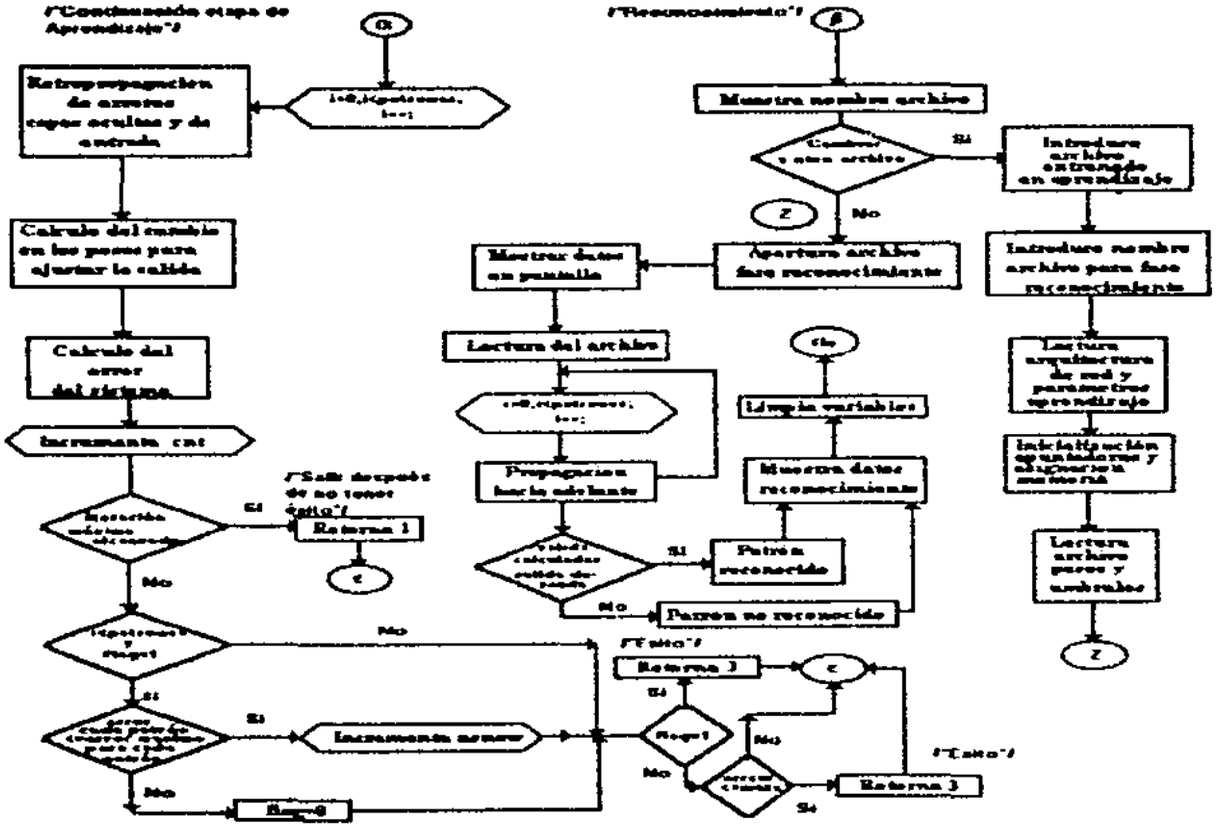


Figura 3.7 Continuación del Diagrama de la Red Neuronal.

III . 2. Elección del modelo a implantar para la Red Neuronal.

Habiendo planteado el problema lo importante es determinar la manera de enfrentarlo, siendo necesario buscar qué tipo de modelo de red neuronal se ajusta a los requerimientos de éste. Es pertinente mencionar que en algunos eventos sísmicos el arribo de la fase P no es tan claro, siendo la identificación de ésta una tarea difícil para el propio analista, esta es una de las razones por las cuales se pretende tener una señal bien depurada como entrada al algoritmo.

Como ya se mencionó existe una gran variedad en modelos de Redes Neuronales, dejando a la consideración del diseñador la elección del algoritmo según los requerimientos del problema a solucionar. En este caso el algoritmo a utilizar para el reconocimiento de fases sísmicas es el de **Retroprogapación.**

Se pensó utilizar este modelo ya que la retropropagación en cuanto modelo de aprendizaje requiere de un instructor que proporcione la salida deseada para el ejemplo de entrenamiento , y por el tipo de problema se necesita categorizar las señales que serán utilizadas para entrenar a la Red , es decir, se necesita que se dé el valor de salida para cada patrón utilizando un criterio para saber en qué momento de la señal se tiene el arribo, en este caso se utiliza un 0.1 en los patrones que se encuentran antes del arribo de la fase P, y 0.9 para los patrones después de dicho arribo, esto significa que la red aprenderá que el arribo se detecta cuando se encuentra la variación en los valores de salida dados de antemano.

En el aprendizaje supervisado, la red asocia una serie de patrones de entrada a un conjunto de patrones de salida. Los escenarios de entrenamiento para redes neuronales artificiales más comúnmente utilizados, emplean este tipo de aprendizaje.

Cuando las redes neuronales se entrenan de esta manera, se utilizan como clasificadores ó memorias asociativas. El patrón de salida representa normalmente la respuesta correcta ó la clasificación correcta para el patrón de entrada

En este tipo de aprendizaje , un patrón de entrada es mostrado a la red para que ésta genere una salida determinada, la cual es comparada entonces, con el patrón de salida deseado y con base en la diferencia se adaptan los pesos de la red conforme a las reglas de entrenamiento.

Este procedimiento generalmente se repite un gran número de veces durante el entrenamiento, para que la red ajuste de forma gradual los pesos de las conexiones y llegue al punto en que puede producir la respuesta correcta para cada patrón de entrada

III . 3. Modelo de Retropropagación.

Es uno de los métodos de aprendizaje en el que sus posibles aplicaciones abarcan un gran número de áreas. Además, la Retropropagación es uno de los métodos más fáciles de entender, sus procedimientos de aprendizaje y actualización son muy simples.

Si la red entrega una respuesta equivocada, entonces los pesos se corrigen de tal manera que el error disminuya y las respuestas posteriores de la red sean más acertadas. La Retropropagación puede resolver cualquier problema que requiera del mapeo de patrones

La base del concepto de retropropagación fue presentada en un principio por **Paul Warbos (1974)**, después fué reinventada independientemente por **David Parker (1982)** y publicada por **Rumelhart y McClelland (1986)**. La Retropropagación encuentra su ancestro en el " perceptrón " , sin embargo, la

diferencia entre los modelos es enorme, el perceptrón por ejemplo, sólo puede aprender patrones que son linealmente separables.

El algoritmo de Retropropagación llegó a ser muy popular debido a que se había demostrado que lograba entrenar a las capas intermedias, consiguiendo que éstas produjeran representaciones interesantes de patrones complejos ofrecidos como entrada

Además, el algoritmo de retropropagación ha demostrado poseer una sorprendente eficacia para entrenar redes neuronales de muchas capas y conseguir que éstas desempeñen un amplio conjunto de tareas. El algoritmo es de máxima utilidad en situaciones donde la entrada y la salida guardan relación o lineal y hay bastante datos de entrenamiento

En el campo de las neurociencias, **Richard Andersen** del Instituto de Tecnología de Massachusetts (MIT) y **David Zipser**, de la Universidad de California en San Diego, han demostrado que el algoritmo de retropropagación constituye un instrumento útil para explicar la función de ciertas neuronas de la corteza cerebral. Estos investigadores aplicaron la retropropagación en el entrenamiento de una red neuronal que habría de responder a estímulos visuales

Hallaron despues que, las respuestas de las capas intermedias se asemejaban sorprendentemente a las de neuronas auténticas, responsables de la conversión de información visual procedente de la retina a formas más adecuadas para áreas visuales ubicadas más profundamente en el cerebro. Este fue un descubrimiento importante, ya que significaba que se había logrado una total analogía entre la Teoría de las Redes Neuronales y el real comportamiento de las neuronas del cerebro.

Sin embargo, en cuanto a la teoría del aprendizaje real de las neuronas biológicas, la retropropagación ha sido recibida con división de opiniones Por una parte, éste algoritmo ha supuesto una valiosa aproximación a nivel abstracto, ademas, es francamente eficiente en la creación de representaciones verosímiles en las capas intermedias.

En consecuencia, los investigadores adquirieron confianza y han ido acertando con procedimientos en los cuales, los pesos son gradualmente ajustados para reducir errores. Hasta entonces, eran muchos los investigadores que suponían que ésta clase de métodos no ofrecían ninguna posibilidad, por conducir inevitablemente a soluciones localmente óptimas, pero globalmente muy malas. Por ejemplo, una red de reconocimiento de dígitos, podría converger sistemáticamente hacia una distribución de pesos que confundiera unos y sietes, no obstante existir un sistema de pesos que permitiera a la red discriminar claramente entre ambos dígitos. Este temor vino a respaldar la extendida convicción de que, un proceso de aprendizaje sólo sería interesante cuando estuviera garantizada su convergencia hacia la solución globalmente óptima.

La Retropropagación demostró que, en muchas tareas no es necesaria la convergencia global para conseguir funcionamientos adecuados. Por otra parte, este algoritmo no parecía ser biológicamente pausible, y la diferencia más obvia que existía para afirmar lo anterior, es que, la información debía viajar por las mismas conexiones en sentido retrógrado, es decir, desde cada estrato al precedente, lo que se suponía no ocurría con las neuronas reales Pero esta objeción es en realidad bastante superficial, ya que ahora se sabe que el cerebro es rico en redes que retrogradan de unas capas a otras anteriores, y puede utilizar estas sendas de múltiples formas para aportar la información que el aprendizaje requiere

De mayor importancia es el problema que plantea la velocidad del algoritmo de retropropagación. La cuestión relevante es ahora el ritmo de crecimiento del tiempo de aprendizaje necesario al crecer el tamaño

de la red,el tiempo invertido en calcular las derivadas del error respecto a los pesos es proporcional al tamaño de la red porque el volumen de cómputo es proprocional al número de pesos.

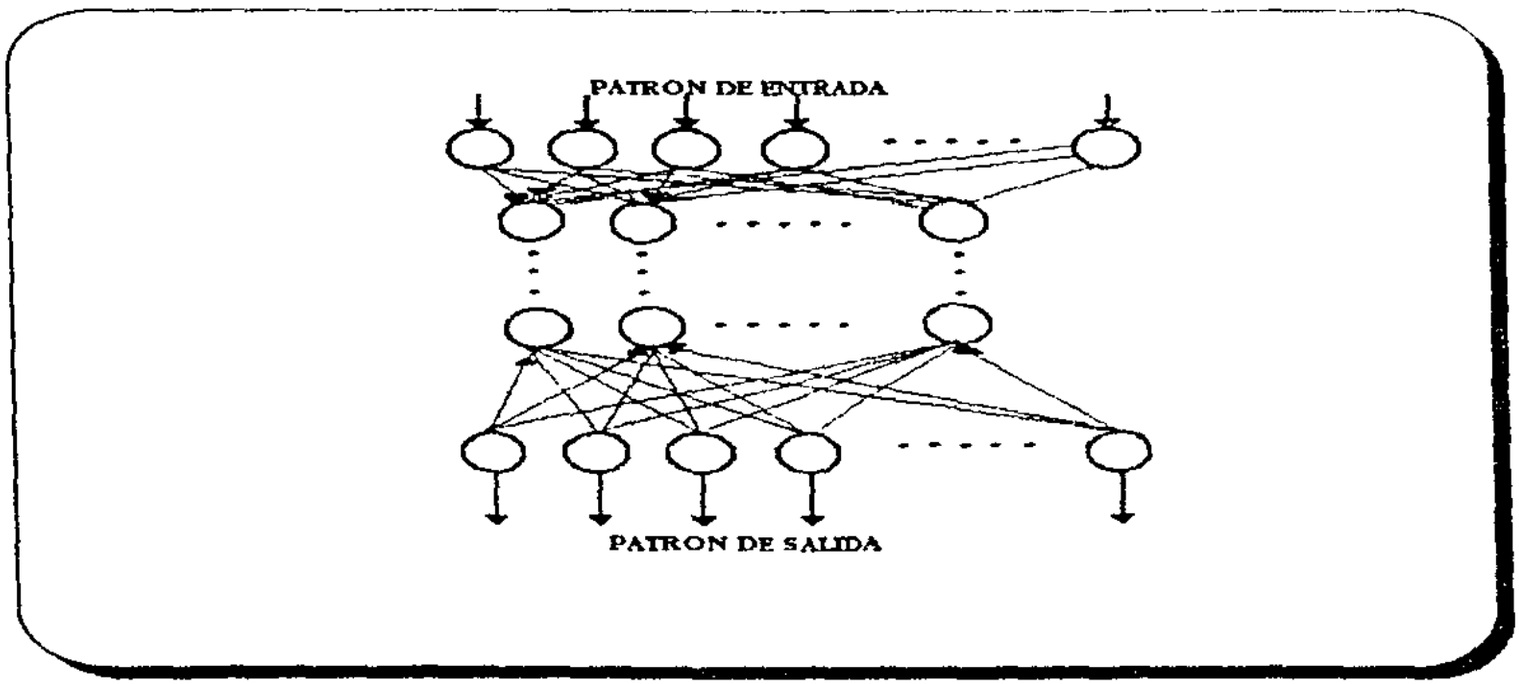


Figura 3.8 Arquitectura de una Red de Retropropagación.

Además, cuanto mayor sea la red, más ejemplos y mayor será el número de veces que se han de corregir los pesos; por consiguiente, el tiempo de aprendizaje crece mucho más rápidamente que el tamaño de la red. Para un mejor entendimiento en la figura 3.8 se puede apreciar la arquitectura de una Red Neuronal

Normalmente, la retropropagación utiliza redes neuronales de 3 ó más capas de unidades. La arquitectura de estas redes es muy común, sólo la capa de entrada recibe entradas externas, no hay limitantes en el número de capas intermedias y sólo hay una capa de salida. Cada unidad de las capas intermedias está conectada con todas las unidades de las capas inferior y superior a ella.

En la arquitectura clásica de este tipo de redes, se puede observar que las unidades no se pueden conectar con otras en la misma capa, pero las conexiones si pueden brincar capas intermedias figura 3.9.

Las redes de retropropagación no necesariamente deben estar interconectadas en su totalidad, sin embargo, muchas aplicaciones han tenido buenos resultados con las capas totalmente interconectadas.

Los pesos de la unidad i a la unidad j serán denotados como Wij. Durante el entrenamiento de la red, los pesos asociados a cada interconexión son ajustados; al concluir el entrenamiento, el valor final de los pesos hace que el error total disminuya, por lo que la red recuerda mejor las asociaciones *entrada-salida*.

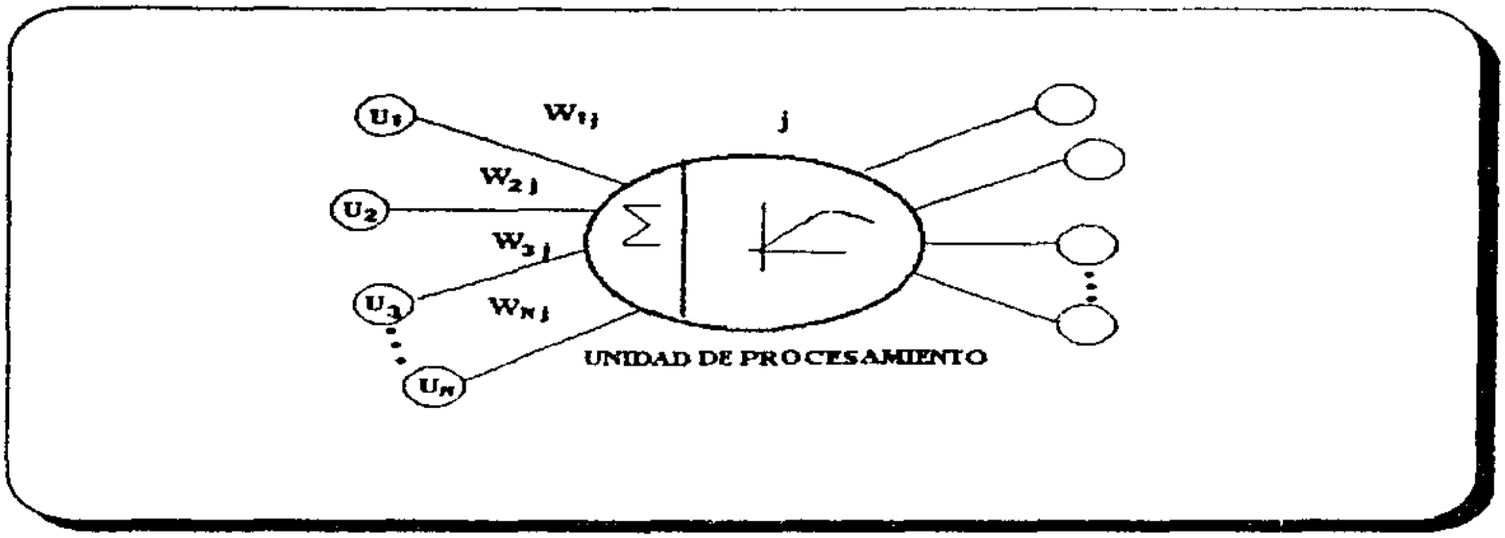


Figura 3.9 . Neurona utilizada por Retropropagación.

El algoritmo de entrenamiento por retropropagación tiene básicamente dos pasos:

III . 3. 1. Propapación hacia adelante.

Comienza con la presentación de un patrón de entrada a la red. En las redes entrenadas por retropropagación las unidades de entrada no son ciertamente procesadores, éstas toman el valor del vector de entrada y lo hacen pasar a la siguiente capa. Una vez presentado el patrón, los niveles de activación en la capa de entrada son programados hacia las capas intermedias hasta la capa de salida, definiendo así los niveles de activación de las unidades. Cada unidad realiza la suma de todas sus entradas y después la hace pasar por una función no-lineal para generar el nivel de activación. La entrada a la unidad queda definida entonces como:

$$S_j = \sum U_i W_{ij} \quad \text{..... (3.1)}$$

Donde :

U_i : Es el nivel de activación de la unidad i

W_{ij} : Es el peso de la unidad i a la unidad j.

La función empleada para definir el nivel de activación utilizada con mayor frecuencia es la función sigmoidal, expresada como:

$$u_j = \frac{1}{1 + e^{-S_j}} \quad \text{..... (3.2)}$$

En realidad, se puede emplear cualquier tipo de función no-lineal, que cumpla con ser derivable, continua y acotada, para generar el nivel de activación. Posteriormente, este valor se pasa a través de las interconexiones de salida de la unidad.

Para proporcionar una convergencia más rápida a la red se incluye un umbral, generalmente conocido como bías,se implementa con una neurona más en cada etapa,excepto en la capa de salida,que está conectada a todas las neuronas de la capa intermedia posterior y cuyo nivel de activación es por definición 1.

Una red con 4 unidades en la capa de entrada, 3 unidades en la capa intermedia y 4 unidades en la capa de salida con término de bías figura 3.10 ejemplificará la función de dicho término .

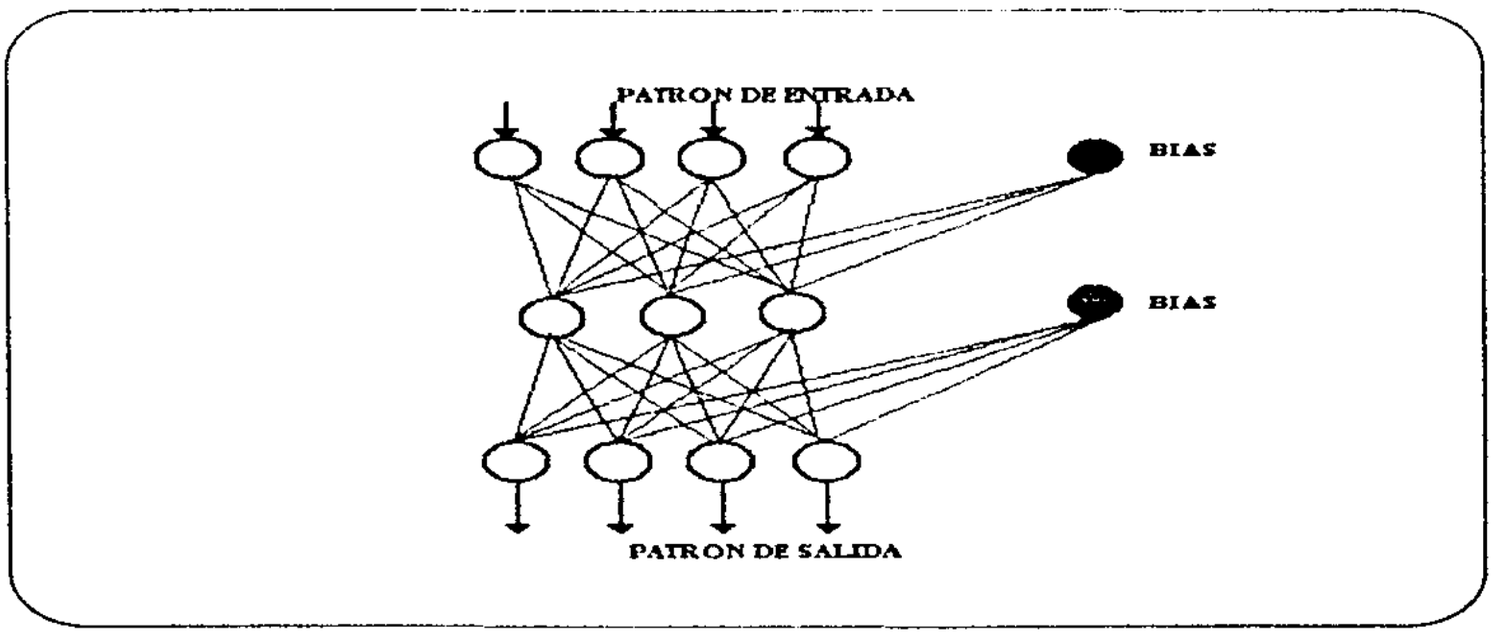


Figura 3.10 . Red 4-3-4 con unidades de Bías.

III . 3. 2. Propapación hacia atrás.

La propagación hacia atrás, comienza con la comparación de la salida de la red con el patrón deseado, generando asi un error para cada una de las unidades de salida. Este error se propaga sobre las capas intermedias hasta la capa de entrada y es calculado para todas las unidades. Los pesos son modificados en cantidades proporcionales al error generado con la finalidad de disminuirlo. Cada unidad tiene asociado un error generalmente denotado como s .La variable s para las unidades de salida se calcula de la siguiente manera:

$$\sigma_j = (Y_j - U_j) F'(S_j) \quad \text{..... (3.3)}$$

Y_j : Es el valor del patrón deseado para la unidad j
U_j : Es el nivel de activación de la unidad
$F'(x)$: Es la derivada de la función no-lineal

La derivada de la función sigmoidal escala el error para obligar a una corrección más fuerte cuando la suma Sj está cerca del nivel umbral El ajuste de los pesos se realiza mediante los valores de las s de cada unidad, cada peso se ajusta considerando el valor de la s de la uniad que recibe entrada de esa interconexión. El ajuste se lleva a cabo de la siguiente manera:

$$W_{ij}\ (t+1) = W_{ij}\ (t) + \alpha\ \sigma_j U_i \quad \ldots\ldots (3.4)$$

Para verificar el grado de aprendizaje de la red se calcula el mínimo error cuadrático, que resulta ser la función de costo a minimizar por el algoritmo. Cuando el mínimo error cuadrático es próximo al cero se dice que la red ha convergido para el conjunto de patrones presentados.

La convergencia no es una garantía en retropropagación, el proceso de aprendizaje toma por lo general bastante tiempo y la red puede caer en mínimos locales haciendo que el aprendizaje se detenga. Existen una serie de técnicas para evitar que la red quede atrapada en un mínimo local , como pueden ser :

- cambiar la escala de aprendizaje, ó
- cambiar el número de unidades en las capas intermedias .

Sin embargo, la técnica más efectiva es hacer que la red haga un salto de una configuración con mínimo local a otra, agregando valores aleatorios a los pesos.

III . 3. 3. Algoritmo de Aprendizaje.

1.Inicializar los pesos de la red con valores aleatorios.
2.Mostrar un patrón de entrada.
3.Obtener la salida de la red. La actividad de las neuronas de la capa de entrada se define como el patrón de entrada. La actividad de las neuronas de las capas intermedias y de salida se define mediante la ecuación (3.2).
Donde:

$$S_j = \sum_i u_k\ w_{kj} \quad \ldots\ldots (3.5)$$

4.Actualizar los pesos de la red de acuerdo a:

$$W_{ij}\ (t+1) = W_{ij}\ (t) + \alpha \delta_j U_i + \beta\ (\ W_{ij}(t) - W_{ij}(t-1)\ ) \quad \ldots\ldots (3.6)$$

Si la j-ésima neurona pertenece a la capa de salida entonces:

$$\delta_j = u_j(1 - u_j)\ (y_j - u_j) \quad \ldots\ldots (3.7)$$

Donde :

y_j : Representa al patrón de salida deseado.

Si la j-ésima neurona pertenece a una capa intermedia entonces:

$$\delta_j = \left[\sum_k \delta_k w_{kj}\right] f'(S_j) \quad \ldots\ldots (3.8)$$

El índice k se refiere a las neuronas inmediatas posteriores a la capa que refiere el índice j.

5.Repetir desde el paso 2 para cada uno de los patrones.

6.Calcular el error cuadrático:

$$E = \frac{1}{2} \sum_j \sum_i (y_{ji} - u_{ji})^2 \quad \ldots\ldots (3.9)$$

El índice k se refiere a las neuronas inmediatas posteriores a la capa que refiere el índice j.

El índice j representa a cada uno de los patrones de salida. Si $E << 0.5$ el entrenamiento termina, en caso contrario se debe repetir desde el paso 2.

III. 4. Otras aplicaciones del modelo de Retropropagación.

Específicamente este modelo tiene una amplia gama de aplicaciones, ha sido utilizado en el diagnóstico médico para asociar un cuadro clínico o una serie de síntomas a enfermedades particulares,así como en la detección de arritmias, taquicardias y otras anomalías cardiovasculares.

En la robótica se ha empleado, sobre todo en área de control, para manejar los movimientos de brazos mecánicos o dar seguimiento a trayectorias espacio-temporales , incluso para entrenar redes que identifican células precancerosas y son capaces de ajustar el espejo de un telescopio para compensar y eliminar las distorsiones de naturaleza atmosférica.

III. 5 Fases Sísmicas.

Primeramente es importante establecer la base teórica de lo que es una fase sísmica, para de esta manera tener una visión más completa de qué es lo que se pretende identificar mediante el algoritmo de la Red Neuronal seleccionado.

Los sismos se originan por una liberación repentina de energía acumulada durante intervalos largos de tiempo en la corteza terrestre , de tal manera que la tierra experimenta una vibración debida a la propagación de ondas. Cuando los esfuerzos acumulados en una falla exceden la capacidad de almacenamiento de esfuerzos en la roca, los bloques que están sometidos a ellos se desplazan y originan 2 tipos de ondas sísmicas : **Ondas de cuerpo** (las cuales se propagan a través del interior de la tierra) y **Ondas de superficie,** siendo las primeras las que interesan en este trabajo. Las **ondas de cuerpo** se dividen en:

a) Ondas P . - Llamadas también ondas primarias o compresionales, son las primeras en ser registradas por una estación sísmica. La trayectoria de las partículas es una zona alternativa de compresión y dilatación Figura 3.11. Las partículas del medio se mueven en una dirección que coincide con la de la propagación de la energía en esa posición (**Al-Sadi, 1980)** , es decir, las partículas del medio se mueven en el mismo sentido en que se propaga la onda .

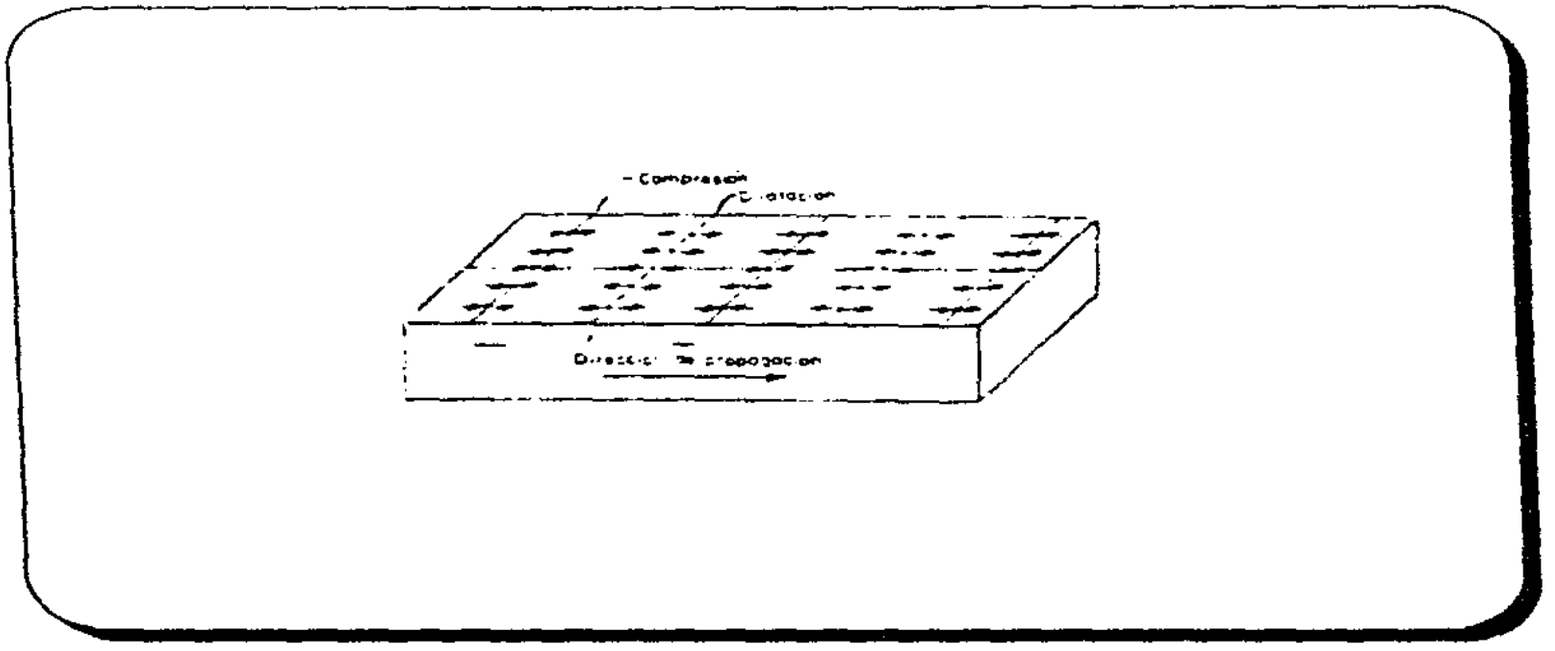


Figura 3.11. Movimiento de una partícula al paso de una Onda P.

b) Ondas S . - También denominadas ondas transversales o secundarias Figura 3 12.
En este tipo de ondas las particulas se desplazan en un plano perpendicular a la dirección de propagación **(Al -Sadi, 1980).**

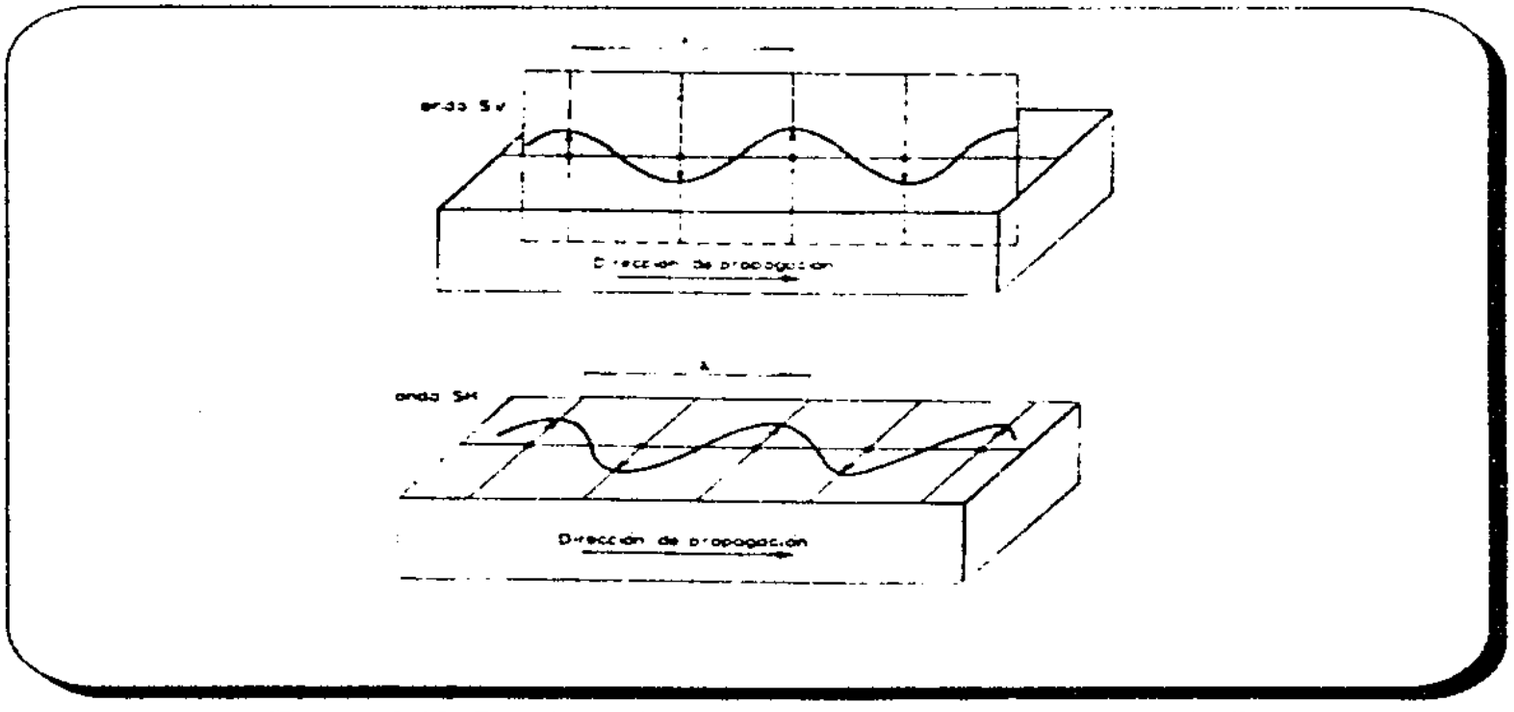


Figura 3.12. Desplazamiento de una particula ante una Onda S.

Cabe mencionar que los eventos son registrados tanto analogica como digitalmente, en la fase analógica se reciben por medio de tambores analógicos instalados en el Instituto de Ingenieria en la Coordinación de Instrumentación Sismica, por otra parte, la fase digital para la recepción de eventos sismicos será descrita en el capitulo siguiente (IV) asi como el detalle de la realización de los dos sistemas mencionados a lo largo de este trabajo.

IV. DESARROLLO DEL SISTEMA DE PREPROCESADO DE EVENTOS SISMICOS.

IV.1. Descripción del Sistema de Adquisición de Datos.

El sistema de Adquisición de datos se implementó utilizando el modelo propuesto por **Lee y Stewart (1981)** con las siguientes caracteristicas:

- Discretizacion de 100 ó más muestras por segundo. Con resolución de 12 bits.
- 16 canales de datos
- Despliegue de las señales sismicas en tiempo real.
- Datos registrados de forma continua o por algoritmos de disparo.

La Red sismotelemétrica (SISMEX) comprende nueve estaciones distribuidas dentro y en el entorno de la Cuenca de México, dichas estaciones registran los eventos sísmicos que ocurren dentro del área que cada uno comprende, además se cuenta con un Puesto Central de Registro (PCR) ubicado en el campus universitario de la Universidad Nacional Autónoma de México,el cual registra de manera continua la señal enviada por cada una de las estaciones antes mencionadas y registradas en los tambores que se encuentran en el Instituto de Ingeniería

Con el fin de digitalizar las señales que llegan a los tambores se implantó un sistema de adquisición digital de registros sísmicos, dicho sistema está formado por una PC (la cual está adquiriendo datos en tiempo real) ,una tarjeta conversora y software que los controla.

Dicha tarjeta contiene un panel de tornillos que conectan las señales provenientes de las estaciones con la computadora, la cual debe tener como requerimientos mínimos :

Requerimientos de hardware mínimos

Procesador 80286
Coprocesador matemático
Disco duro de 30 MB
1.5 MB de memoria RAM (640 KB de RAM principal y 768 KB de memoria extendida)
Monitor EGA.

El procedimiento para analizar los registros sísmicos se divide en dos etapas:

IV. 1. 1. Obtención de registros sísmicos.

Los registros sísmicos se obtienen en forma digital después del proceso de conversión analógico/digital, dichos registros son guardados en archivos con extensión **WVM** siguiendo una numeración consecutiva.

Diariamente se realiza la transferencia de estos archivos a una estación de trabajo SUN mediante una red local utilizando FTP (File Transfer Protocol), todo este proceso se realiza de manera automática. Bajo este ambiente el algoritmo de la Red está implementado.

IV. 1. 2. Análisis de registros sísmicos.

Una vez que los datos son recibidos se transforman de un formato binario multiplexado en 16 canales, a un formato estándar de registros sísmicos (llamado SEISAN, Formato Noruego de Registros Sísmicos) . La tarjeta conversora A/D tiene en el primer canal una señal de tiempo que sirve para corregir el reloj interno de la computadora de detección y referenciar los registros a un tiempo absoluto.

Los registros son analizados por un experto, quien debe primero que todo verificar si el evento realmente es un sismo ó sólo ruido, si se ha confirmado que se está frente a un evento sísmico entonces se procede a registrarlo en la base de datos.

Una vez almacenados en la base de datos se procede a identificar las fases importantes en él, a fin de localizarlo y clasificarlo dentro de las siguientes categorías. sismo lejano conocido también como telesismo, regional, local, ó explosión.

Todo el proceso mencionado anteriormente es ejemplificado mediante la figura 4.1 en donde se representa esquemáticamente la Adquisición y análisis de datos del sistema digital , el cual con sus registros digitales alimentará al sistema de preprocesamiento .

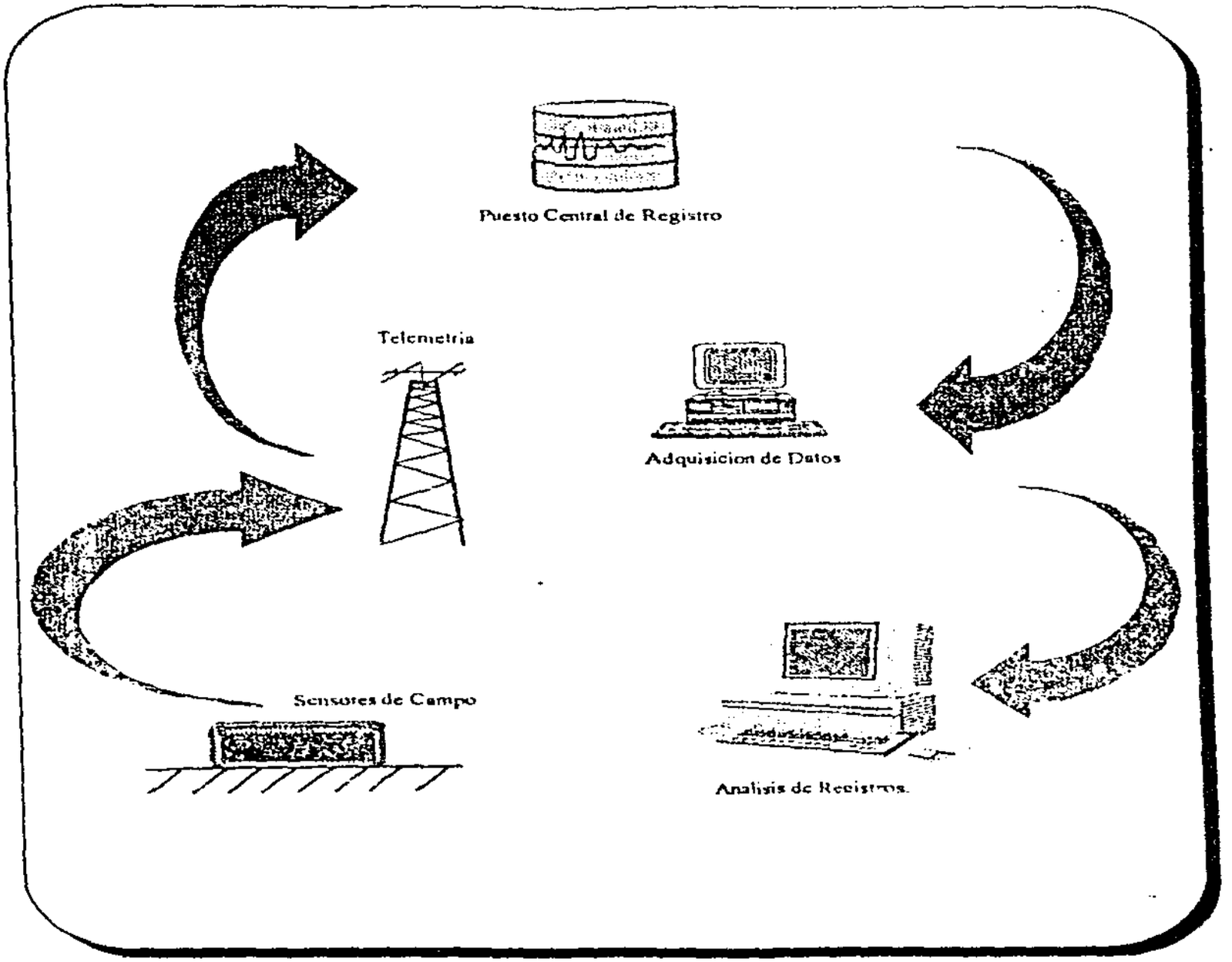


Figura 4.1 . Representación esquemática de la Adquisición y Análisis de Datos del sistema propuesto por Lee y Stewart (1981).

IV. 1. 3. Base de Datos.

Se tiene una base de datos jerárquica con el objeto de accesar a la información eficazmente, los eventos sismicos son registrados en base a la fecha de ocurrencia de éstos y a la estación de registro, por ejemplo: Un sismo que ocurrió el día 5 de Enero del año 1996 a las 20:20 con 34 segundos se grabará en un archivo con nombre: *9601-05-2020-34S.SMX_16* (año mes - día - hora minuto- segundo y la designación S.SMX_16, en donde el 16 expresa que se tienen registrados 16 canales en el archivo delevento sismico) y las siglas SMX que definen a la Red de SISMEX

Una vez registrados los eventos sismicos se procede a analizarlos mediante software instalado en la estación de trabajo (**Haskov, 1994**) , por medio de éste se leen los arribos de las fases sismicas de cada estación del evento para poder ser localizado más tarde con los parámetros obtenidos de la lectura realizada, como lo son : Arribo de fase P, Arribo de fase S, Amplitud de la señal , y duración de la misma .

Los archivos binarios de entrada están formados por datos enteros constituidos por 2 bytes, cada uno representa el valor de amplitud de la señal sismica, el encabezado contiene factores importantes como: el tiempo de grabación (el cual determina el tamaño del archivo), la razón de muestreo (el número de muestras por segundo), número total de muestras por canal, fecha del evento sismico, hora a la que ocurrió, etc.

El archivo está conformado por 16 canales , cada uno representa una traza en particular. El primer canal contiene la señal del tiempo, el cual se toma como referencia para la graficación del sismo digitalmente.

En la figura 4.2 se muestran las etapas de que consta el sistema de Análisis de Datos Sismicos con el fin de ejemplificar de una manera más clara como interactúan los procesos entre sí y con su entorno.

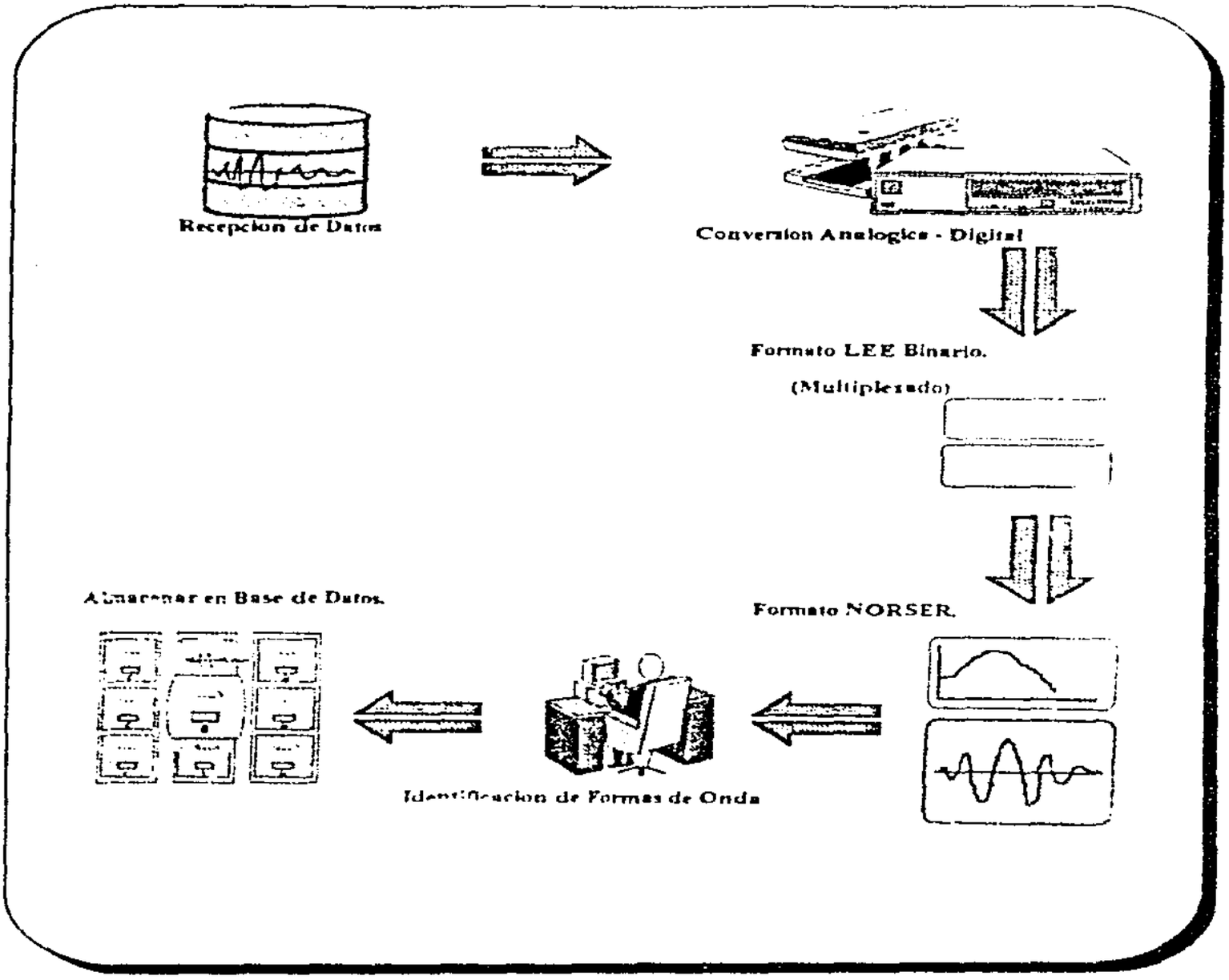


Figura 4.2 . Representación del Sistema de Análisis de Datos Sísmicos SEISAN propuesta por Haskov y Lindholm (1994).

IV. 2. Entorno de Programación.

IV. 2. 1. Plataforma de diseño.

Tanto el algoritmo de preprocesado como el de la Red Neuronal corren bajo el sistema operativo UNIX y están escritos en lenguaje C

El sistema operativo UNIX cuenta con un software llamado **OPENWINDOWS Versión 2.** El cual permite una interface gráfica para obtener una mejor visualización mediante el manejo de ventanas Esto hace más fácil la utilización del sistema al usuario final .

Una de las ventajas de utilizar esta interface gráfica es la manipulación de ventanas , así como la utilización del multiprocesamiento dando al usuario la facilidad de ejecutar varias aplicaciones a la vez sin ningún problema.

IV. 2. 1. Lenguaje de programación.

El lenguaje C es un lenguaje de nivel medio porque combina elementos de lenguajes de alto nivel con la funcionalidad del lenguaje ensamblador. Como lenguaje de nivel medio C permite la manipulación de bits, bytes y direcciones (elementos básicos con los que funciona la computadora), esto le hace particularmente adecuado para la programación de sistemas, en la que estas operaciones son muy comunes.

C es un lenguaje estructurado, cuenta con el concepto de compartimentalización de código y datos, es decir tiene la capacidad de seccionar y esconder del resto del programa toda la información y las instrucciones necesarias para llevar a cabo una determinada tarea Se logra la compartimentalización mediante el uso de subrutinas ó módulos que emplean variables locales con el objeto de hacer más fácil el diseño y programación de cualquier programa

El código de C es muy portable (significa que es posible adaptar el software escrito para un tipo de computadora en otra), esto es sumamente práctico ya que redunda en un ahorro de tiempo y costo al implantar cualquier sistema en otro equipo. Por otra parte, C permite casi todas las conversiones de tipos de datos, a su vez no lleva a cabo la comprobación de errores en el tiempo de ejecución como pueden ser: el sobrepasar los límites de los arreglos ó la incompatibilidad entre los tipos de los argumentos.

Las subrutinas (procedimientos) que conforman cualquier programa pueden ser declaradas antes del módulo principal ó después de él (técnica utilizada en este trabajo) . En la primera parte del programa se declaran las bibliotecas a utilizar así como los valores constantes necesarios para realizar los cálculos pertinentes. Posteriormente se declaran los procedimientos que conforman al programa al igual que las variables globales, declarando finalmente el módulo principal.

En conclusión , C proporciona al programador lo que requiere: pocas restricciones, estructuras de bloques, funciones independientes y un conjunto compacto de palabras clave. De acuerdo a lo anterior, se decidió desarrollar el sistema bajo este lenguaje.

Como este sistema involucra objetos declarados en **OPENWINDOWS Versión 2**, la forma de compilar los programas es diferente a la de compilar un programa desarrollado bajo ANSI C , lo que se explica posteriormente .

IV. 2. 2. Elementos de programación.

En muchas situaciones de programación es imposible saber qué tan grande se requiere un arreglo, siendo imposible utilizar uno predefinido. Al utilizar este tipo de arreglos las dimensiones quedan establecidas en tiempo de compilación y no se pueden cambiar durante la ejecución , ésta es una desventaja cuando se confirma que el tamaño definido no satisface los requerimientos.

En el caso del preprocesamiento de los datos, no se conoce el número de arreglos a ser utilizados ya que éste dependerá del tamaño de los archivos binarios de entrada de la señal sísmica, dicho tamaño es determinado a través de la razón de muestreo (número de muestras por segundo a las que se grabó el evento), así como del tiempo de grabación (duración del evento en segundos), así que la solución está en crear un **arreglo dinámico,** éste usa memoria de la región libre denominada el montón, y es accesado indexando un puntero a esa región de memoria.

Por lo mencionado anteriormente, los datos procesados serán asignados a un arreglo dinámico de apuntadores, definido como *(arreg_dinamic[j] + i) ; el asterisco indica que se trata de un apuntador, y lo que se encuentra dentro del paréntesis indica un arreglo en donde la j se refiere al número de arreglo que se está accesando. Se pretende que el valor de j indique el número de patrón accesado, a su vez el índice i indica el número de muestra dentro de dicho patrón

Además se utilizan varios punteros (variables que contienen una dirección de memoria), sobre todo para el acceso de lectura y escritura a los archivos.

Para la utilización de dicha estructura se necesita utilizar dos ciclos, en donde el externo indique el número de patrón y el interno el número de muestra a ser leída, el ciclo externo variará de 0 hasta el número de patrones - 1, y el ciclo interno de 0 hasta el número de muestras por patrón -1, ahora bien cuando se necesita incrementar la posición del arreglo para ir accesando cada uno de los elementos que lo conforman se utilizará la siguiente instrucción (arreg_dinamic[j]++).

Al llegar hasta la última muestra de cada patrón se debe regresar el arreglo de apuntadores (arreglo dinámico) al inicio para poder accesar al siguiente patrón y así sucesivamente, para lograr lo anterior es necesario declarar nuevamente dos ciclos en donde se indique que el arreglo debe retroceder (arreg_dinamic[j]--).

En el sistema de preprocesamiento de los eventos sísmicos al finalizar cada procedimiento los datos son salvados a un archivo binario, para contar con dichos valores en la siguiente etapa, éstos se guardan en formato float (el sistema maneja 4 bytes para las variables de este tipo) .

En cambio en el algoritmo de la Red Neuronal se utilizan arreglos bidimensionales tanto para almacenar los valores de entrada correspondientes a los patrones presentados, los valores de las capas ocultas así como el valor del nodo de salida a ser clasificado.

IV. 2. 3. Compilación.

Se emplea el compilador de C denominado cc , el cual es propio del Sistema Operativo UNIX.
La sintaxis de compilación es la siguiente.

```
cc -I/usr/openwin/include procesa.c -o PROCESA -lm -lxview -lX11 -L${OPENWINHOME}/lib
```

donde:

-I/usr/openwin/include	Declaración de la ruta de acceso de librerías a incluir.
procesa.c	Nombre del programa a compilar
-o PROCESA	Nombre del archivo de salida (ejecutable)
-lm -lxview -lX11	Indicación de las librerías utilizadas
-LS(OPENWINHOME)/lib	Direccionamiento de las librerías.

Una vez compilado el sistema se procede a ejecutarlo desde la línea de comandos. Lo mismo sucede con el programa de la Red Neuronal llamado neurona.c .

IV. 3. Descripción del ambiente gráfico

La descripción de los sistemas desarrollados en este trabajo se dividirá en dos puntos principalmente, **a) Interface gráfica y b) Lógica de diseño de los procedimientos que conforman los dos sistemas programados.**

El primero explicará el ambiente bajo el cual dichos sistemas están corriendo, y el segundo la parte conceptual y lógica de los módulos que los constituyen, pretendiendo con ésto dar una visión general de los elementos involucrados en el desarrollo .

Se pretende describir el ambiente bajo el cual corren los dos sistemas (como se mencionó anteriormente), sobre todo la interface gráfica que hace uso de los objetos de OPENWINDOWS, ésta es una diferencia importante en cuanto a la codificación de un programa escrito solamente bajo el lenguaje C , que cuenta también con una facilidad gráfica pero no tiene la posibilidad de crear todo un ambiente de ventanas.

Los dos sistemas se desarrollaron tomando como base una ventana segmentada en varias secciones. Cada una de ellas definida de distinta manera puesto que realizan tareas diferentes.

La manipulación de las ventanas se realiza mediante una serie de "pushbuttons" , así como menus "pop-up" ó "pull-down"(menus en cascada) entre otros objetos definidos bajo OPENWINDOWS. Todo esto con el fin de facilitar la interacción del usuario con el sistema

IV. 3. 1. Elementos que constituyen el ambiente gráfico.

El sistema consta de una pantalla principal, la cual proporciona todo el ambiente gráfico que es requerido para realizar la manipulación de ventanas, permitiendo de esta manera un mejor desenvolvimiento del usuario con el sistema La pantalla está constituida por diferentes secciones, las cuales serán mencionadas a continuación en el orden que conservan dentro de ella.

- **Barra de título**
- **Barra de menus**
- **Sección de graficación**
- **Sección de mensajes**

Barra de título.- Muestra datos de interés como el nombre de la aplicación, fecha y hora de inicio de la sesión.

Barra de menus.- Se muestran los diferentes botones que despliegan información acerca de los procesos que conforman a la aplicación, cada botón muestra un menu con una serie de comandos ó acciones a realizar. Se puede accesar un menu seleccionándolo mediante el mouse, éste puede exhibir en forma de cascada(descendentemente) las opciones que se pueden llevar a cabo en forma de una lista de comandos ó sólo una opción. Se puede contar con menus que contengan a su vez submenus, es decir, un comando dentro de un menu puede contener a su vez otro menu

Sección de graficación.- En esta sección se presentan los resultados importantes que surgen de los procesos que se efectuaron.

Sección de mensajes.- En esta sección se pueden enviar mensajes correspondientes al proceso que se está llevando a cabo, asi como su nombre, todo esto con el objeto de que el usuario esté enterado de lo que está haciendo el sistema

En la figura 4.3 se pueden apreciar las secciones que conforman la pantalla principal del entorno gráfico para el sistema de preprocesamiento de eventos sísmicos tal como lo indica la barra de titulo.

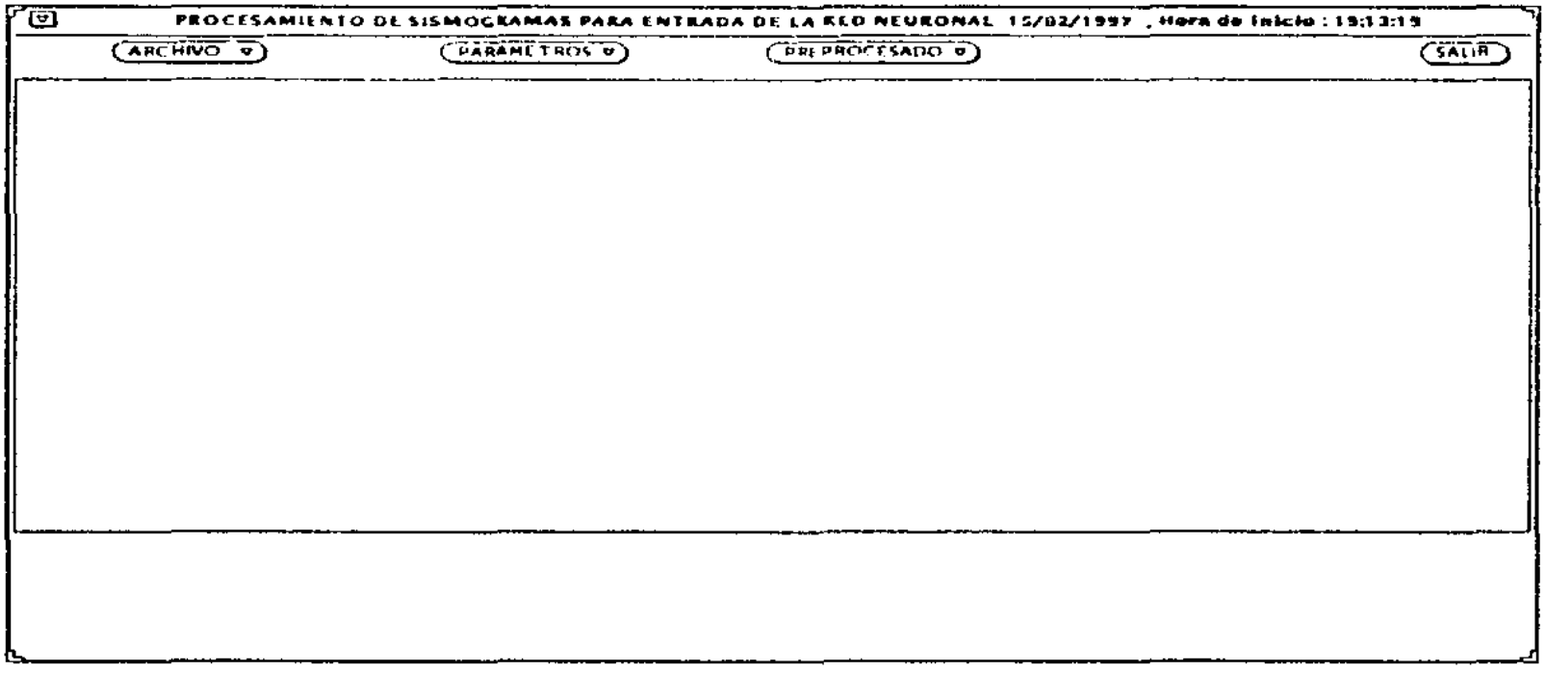


Figura 4.3 . Pantalla principal .

El apéndice A (anexado al final) contiene la declaración de los objetos de OPENWINDOWS utilizados en la creación del ambiente de ventanas del sistema, pretendiendo con ésto dejar bien establecido las bases del entorno gráfico presentado en los sistemas realizados en este presente trabajo .

IV. 4. Implementación del Sistema de Preprocesado de Eventos Sísmicos.

A continuación se mostrará todo lo relacionado al diseño y desarrollo del sistema correspondiente al preprocesamiento de los eventos sísmicos.

IV. 4.1. Diseño y Desarrollo del sistema de preprocesado.

IV. 4. 1. 1. Petición de parámetros.

En esta etapa se piden al usuario parámetros importantes para el preprocesado como lo son: Número de muestras por patrón, Razón de diezmado (pudiendo ser de 50 ó 20 muestras por segundo, es decir se toma 1 de cada 2 valores ó 1 de cada 5) , Número de la traza a ser procesada, Selección de la fase ya sea aprendizaje ó reconocimiento (de esta selección depende la grabación del encabezado de los archivos de salida tipo texto), Tipo de proceso, Deslizamiento de ventanas, y finalmente se tiene la opción de proporcionar el Tiempo de arribo de la Fase "P" ó de permitir que el sistema lo calcule por medio de un algoritmo implantado dentro del sistema, el cual entre otras cosas será descrito posteriormente.

IV. 4. 1. 2. Extracción de datos importantes del archivo de entrada.

Del encabezado se extrae la información correspondiente a: Fecha del evento, así como hora ,minuto y segundo de inicio de grabación del mismo, Tiempo de grabación (segs), Razón de muestreo pudiendo ser 100 ó más muestras por segundo, así como el número de muestras grabadas, después de esta parte existe un encabezado para cada canal precediéndole los valores de amplitud correspondientes a éste, los cuales también serán extraídos para su posterior utilización .

Es pertinente mencionar que el manejo de bytes entre la PC y la estación de trabajo es un poco diferente, es decir, mientras que un byte en la PC está constituido por 2 bytes en la estación de trabajo es de 4 bytes, pero como el archivo de entrada procede de la PC los datos están grabados como enteros de 2 bytes, por lo tanto para calcular el número de bytes utilizados por cada canal, es necesario multiplicar el número de muestras (valor obtenido anteriormente) por 2.

Para solucionar el problema de la disparidad de bytes de que consta un entero en el archivo que sirve como entrada con respecto a la estación de trabajo, se utilizó una variable tipo short integer para leer los datos, de esta manera se asegura que la variable leída sólo contendrá 2 bytes .

Ahora bien, para conocer el número de bytes que existen entre un canal y otro se debe sumar el número de bytes que constituyen cada canal más el número de espacios en blanco que existen (1056 bytes), este número siempre es constante no importa de qué archivo se trate.

Al conocer el número de muestras por patrón(parámetro proporcionado por el usuario) , es posible definir cuántos patrones de entrada se tendrán, esta cantidad es calculada de la siguiente manera, El número de muestras por patrón es multiplicado por la razón de diezmado (la señal necesita ser diezmada para trabajar con menor número de muestras, obteniendo un mejor tiempo de ejecución así como garantiza que la señal no perderá sus características esenciales, ya que disminuye en puntos pero no pierde su forma original solo suavizándose), este número indicará la cantidad de datos que deben contener los bloques que serán extraídos del archivo para ser leídos, al finalizar el proceso de diezmado los patrones contendrán el número de muestras que fué proporcionado por el usuario.

bloque = muesvent * tamdiez . . . . . (4.1) Bloque de datos a ser leídos para diezmar la señal.

Como la función que se encarga de posicionar el cursor dentro del archivo para extraer los datos espera como dato la posición en bytes, es necesario multiplicar las cantidades calculadas por 2 (número de bytes que almacenan un dato o muestra).

Si se selecciona deslizamiento de ventanas , el número de muestras por ventana se divide entre 4, ésto se determinó en base a pruebas realizadas al implantar el sistema, lo que será tratado en el capitulo V. El número de patrones totales deslizados con el que se contará se calcula multiplicando el número de patrones por el factor de desplazamiento.

desp = muesvent / 4 . . . . . (4.2) Factor de deslizamiento.
ventotal = numvent * desp . . . . . (4.3) Número de patrones totales.

Después de realizar los cálculos pertinentes para llevar acabo el procesado de los registros, se toma la posición del canal a procesar para empezar a leer los datos que serán guardados en una variable temporal, la cual es asignada a su vez a un apuntador cercano llamado dinamic_arreg (éste tiene la dimensión del número de muestras por patrón), esto con el fin de guardar los datos en un archivo binario, para la lectura de datos se utiliza la función de fread, utilizándose fwrite para la escritura.

De la fecha extraida se separa la hora, minuto y segundo en el que ocurrió el evento. El archivo de entrada describe en que momento se inició la grabación del evento, porporciona información como el año, mes, dia, hora, minuto, segundo , asi como las iniciales de la agencia y el _16 indica el número de trazas, por ejemplo: 9401-02-1855-58S.SMX_16

IV. 4. 1. 3. Diezmado.

Primeramente, se pensó en realizar un proceso que permitiera que la señal no perdiera sus caracteristicas pero que disminuyera el número de datos a ser leidos para mejorar el tiempo de ejecución de los procesos subsecuentes, concibiendo entonces el proceso de diezmado.

Se accesan los datos de la señal de entrada dependiendo de la razón de diezmado seleccionada, es decir, se multiplica el número de muestras por patrón y la razón de diezmado para determinar el tamaño del bloque de datos a ser leidos del archivo, al mismo tiempo que la señal es diezmada se determinan los patrones con los que se seguirá trabajando en las etapas posteriores.

Cabe mencionar que para determinar las razones de diezmado establecidas, se hicieron pruebas con algunas, seleccionando las que mejor influian en el proceso para suavizar la señal de entrada.

Dependiendo de la traza seleccionada se determina en qué posición se debe colocar el apuntador de lectura del archivo para extraer los datos.

Cada vez que se va a utilizar algún apuntador se reserva una cierta cantidad de memoria para él, la que no debe ser excedida ya que el apuntador se puede perder generando una serie de problemas tanto simples como complejos, contemplando lo anterior el arreglo de apuntadores se recorrerá mientras la muestra que se está accesando sea diferente al número de muestras por patrón - 1

Cabe mencionar que en este procedimiento se calcula el valor promedio del primer patrón para realizar la resta del offset encontrado, es decir, el dato calculado indicará el nivel sobre el que se encuentran los valores de amplitud correspondientes a la señal sísmica

suma += *(arreg_dinamic[j]) . . . . . (4.4) Suma de los datos correspondientes al primer patrón.
donde: j representa el número del patrón que se está procesando en este caso sólo se calcula la suma para el primer patrón..

cota0 = suma / muesvent . . . . . (4.5) Valor promedio del primer patrón.

IV. 4. 1. 4. Resta del valor de offset.

El valor promedio del primer patrón de la señal original (mencionado anteriormente), es restado a cada valor de la señal diezmada. Para asegurar que la señal se encuentra sobre el mismo nivel .

Después de leer todas las muestras correspondientes a un patrón se recorre el arreglo de apuntadores al principio de este, para evitar perder los valores correctos , esta operación debe ser realizada cada vez que se accesa al arreglo.

(arreg_dinamic[j]) = ((arreg_dinamic[j]) - cota0) . . . . . (4.6) Resta del offset a cada valor.

IV. 4. 1. 5. Cálculo de valores absolutos.

Se calcula el valor absoluto de cada dato de la señal diezmada, garantizando con ésto que la señal quedará solamente en el eje positivo para evitar que algunos de los valores sean anulados ó disminuidos a causa de los negativos, desvirtuandose con esto la señal que se quiere obtener a la salida. De esta manera se puede asegurar que el proceso posterior de los datos será más fidedigno.

Al término de cada procedimiento los datos son guardados en un archivo binario como tipo float, para poder ser accesados por el procedimiento subsecuente.

(arreg_dinamic[j]) = abs((arreg_dinamic[j])) . . . . . (4.7) Cálculo del valor absoluto de cada dato de la señal.

IV. 4. 1. 6. Algoritmo para procesar el registro sísmico

Para preprocesar la señal sísmica se desarrollaron 3 procesos distintos con el fin de tener una gama más amplia de posibilidades de trabajar con los datos y registrar cual de ellos entrega una señal mejor depurada a la salida .

Dentro de los procesos se encuentran: STA de 5 muestras,STA de 20 muestras y un proceso de integración (no numérica). STA significa **short term average** es decir, se utilizan pocas muestras para calcular un promedio entre ellas y obtener un nuevo valor que será asignado al arreglo. STA se referencia a unidades de tiempo, siendo entonces para un muestreo estándar de 100 muestras/seg, correspondiendo 0.05 para el STA de 5 muestras y 0.2 segundos para el de 20.

Pero como el diezmado ya ha sido aplicado a la señal entonces el muestreo varia, y tomando como razón de diezmado 1 muestra de cada 5 tendremos que para el STA de 5 le corresponde a cada muestra una fracción de tiempo de 0.05 segundos.

IV. 4. 1. 6. 1. STA de 5 ó de 20 muestras.

El objetivo de este proceso es leer tantos datos como lo indica el proceso seleccionado, siendo 20 ó 5 muestras, dichos datos serán sumados y divididos entre el mismo número de datos leidos , con la finalidad de tener el valor promedio de esas lecturas.

Para obtener el siguiente valor promedio se toman los siguientes n datos del archivo desplazados por 1, es decir, se tienen n-1 datos iguales que en el cálculo anterior con la diferencia que para tener el mismo número de datos a ser sumados desplazando un valor es necesario tomar el valor contigúo a la posición donde se quedó en el cálculo anterior.

Si el archivo tuviera los siguientes valores 1234567890 y se seleccionara el proceso STA de 5 muestras, entonces se tomarian los primeros cinco valores teniendo los datos 12345 si se suman y se dividen como se mencionó anteriormente , se tendria el valor de 3 el cual será asignado al primer elemento del apuntador, subsecuentemente se desplaza un dato tomando 23456 , el 6 es el nuevo valor accesado, asi sucesivamente se realizará el proceso.

Para lograr lo que se pretende dentro del procedimiento se calcula el número de datos diezmados de que consta la traza a ser procesada, esto es, se multiplica el número de muestras por patrón por el número de patrones, este valor multiplicado por la razón de diezmado debe dar el número de muestras grabadas (dato que fué extraído del encabezado general del archivo) Si se seleccionó cualquiera de los procesos concernientes al STA, entonces el valor de muestras de que constará (parámetro proporcionado por el usuario que puede ser 5 muestras ó 20) es almacenado en una variable.

Se procede entonces a leer del archivo de salida del procedimiento anterior tantos datos como el número seleccionado del STA, éstos datos son sumados y posteriormente divididos entre el mismo número, de esta manera se obtiene un valor que será asignado al primer elemento del arreglo de salida, este proceso será realizado dentro de un ciclo hasta que el número total de datos sea alcanzado.

El valor promedio de los datos leídos mencionado anteriormente es asignado a un apuntador cercano designado como ***(dinamic_arreg+l)*** donde ***"l"*** se incrementará cada vez que sea leido el siguiente dato, el valor calculado también es asignado a una variable llamada ***max*** que guardará el valor máximo leido durante este proceso para ser usado en la etapa subsecuente El valor será asignado al apuntador ***dinamic_arreg*** siempre y cuando ***"l"*** sea menor que el número de muestras por patrón disminuidas en una unidad (muesvent -1),de esta manera se proceden a leer del archivo los siguientes datos desplazados por 1, incrementándose la variable que controla el número de datos a ser leidos.

Cuando ***"l"*** es igual al número de muestras por patrón-1 entonces se asignan al arreglo dinámico los valores del apuntador cercano, al terminar de asignar los datos se procede a regresar al arreglo de apuntadores a la posición inicial, al mismo tiempo que se incrementa la variable que contiene el número del arrego dinámico a ser procesado (indicando el número del patrón). Hecho esto entonces se inicializa tanto el valor de ***"l"*** como el de los indices de los apuntadores utilizados con el fin de empezar a procesar los datos del siguiente patrón

En la figura 4.4. se puede apreciar el diagrama de flujo que ejemplifica la lógica seguida en el desarrollo de este procedimiento.

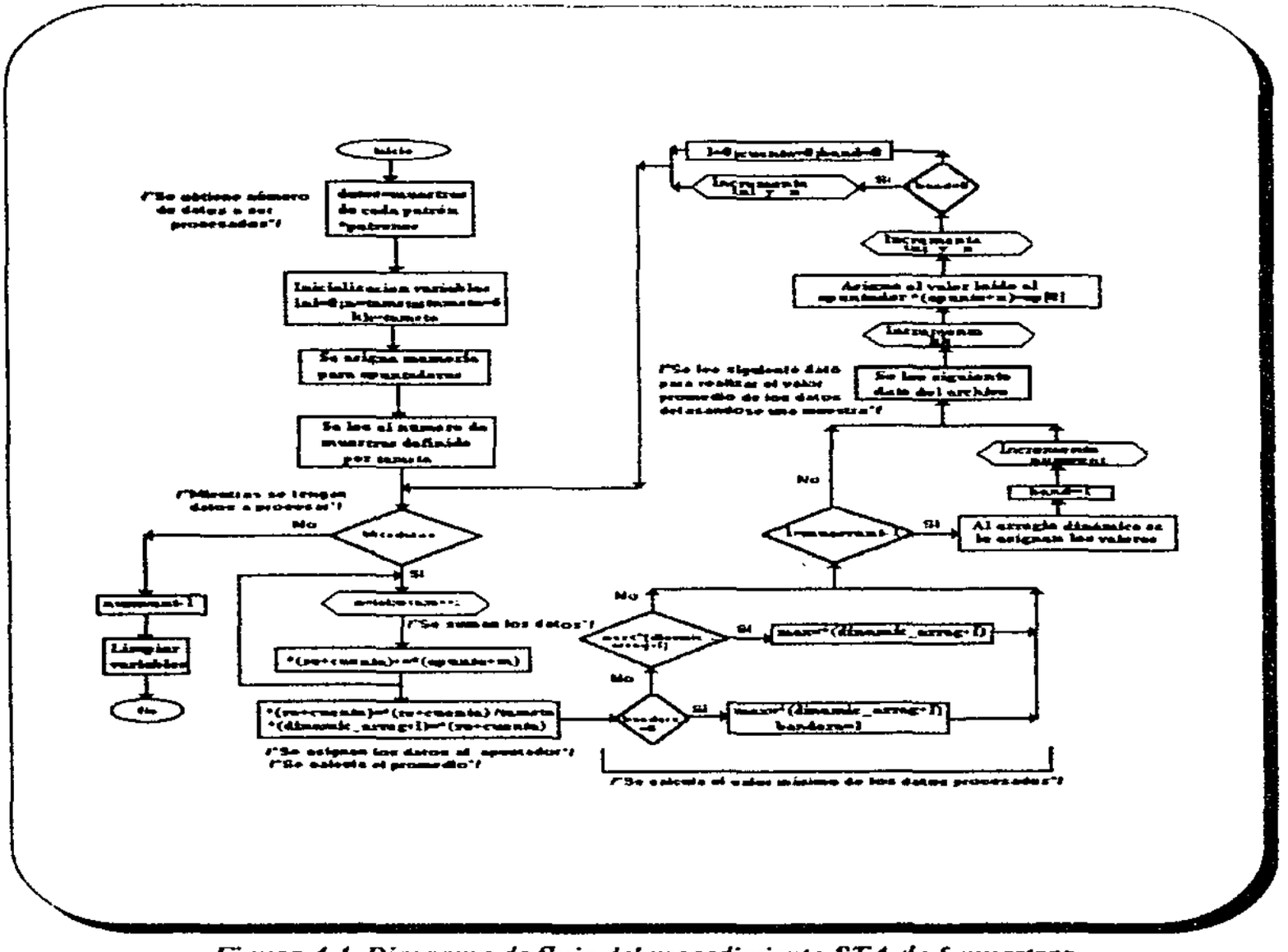


Figura 4.4 Diagrama de flujo del procedimiento STA de 5 muestras

IV. 4. 1. 6. 2. Integración

Este proceso pretende obtener una curva ascendente en donde un cambio brusco de la pendiente indicaría el arribo de la fase *"P"* . Para conseguir lo anterior es necesario obtener en cada paso un valor cada vez mayor en magnitud.

Desde 0 hasta el número de patrones y mientras la variable que controla el número de muestras por patrón sea menor que el número proporcionado por el usuario se abre un ciclo dentro del cual se tomarán 2 datos para ser sumados y divididos entre el mismo número de datos leídos, al valor calculado mediante este proceso, es decir, al valor promedio se le suma el promedio anterior, con la finalidad de obtener un dato cada vez mayor.

Para obtener el siguiente valor se desplaza la toma de los 2 datos una posición hacia adelante realizando el promedio y sumándolo al dato anterior y así sucesivamente.

En un apuntador denominado *app* se vacian los datos del arreglo dinámico, mientras el número de muestra leída sea diferente al número de muestras por patrón -1 (muesvent-1) se posiciona el arreglo dinámico en el siguiente dato, en caso contrario se iguala el último dato al anterior, esto se determinó de esta manera puesto que al leer 2 datos y obtener uno a la salida el número de muestras por patrón es reducido en 1.

Por lo anterior, el penúltimo y último valor serán los mismos estabilizando los valores finales de cada patrón pretendiendo con esto que la llegada del arribo sea más clara para la Red Neuronal siendo asociado correctamente.

Al calcular el valor promedio se asigna al apuntador denominado ***dinamic_arreg*** en donde la variable ***"i"*** indicará el número de la muestra en dicho patrón. Cuando se llega a la última muestra de cada patrón se lee el primer dato del siguiente calculándose el promedio y sumándosele por supuesto el valor considerado como último para el patrón recien utilizado,siendo asignado como primer elemento del patrón en cuestión.

Al terminar de procesar cada patrón las variables involucradas son inicializadas, para empezar de nuevo con el proceso. Se cuenta con una rutina que calcula el valor máximo leído para ser usado en la etapa de normalización de los datos.

IV. 4. 1. 7. Normalización.

Tomando como base el valor máximo obtenido en la etapa anterior, se procede a dividir todos los valores que constituyen a los patrones entre el máximo dato de la señal sísmica. El proceso de normalización sirve para tener a los datos dentro de un mismo rango y de esta manera facilitar los cálculos posteriores, el rango variará de 0 a 1, puesto que si todos los valores son divididos entre el máximo se obtendrá la unidad.

Al dividir cada dato entre el máximo será asignado al apuntador cercano nombrado en otros procesos ***(dinamic_arreg)*** , solamente para corroborar que el proceso arroja los valores deseados en el rango establecido se procedió a realizar una subrutina para extraer tanto el dato mínimo como el máximo.

Cada vez que se termina con los datos pertenecientes a un cierto patrón se mandan escribir a un archivo binario para ser utilizados en el proceso subsecuente

***(arreg_dinamic[j]) = (*(arreg_dinamic[j]) / max) . . . . . (4 8) Normalizaciónde la señal.**

A continuación se muestra el diagrama de flujo del proceso de normalización figura 4.5.

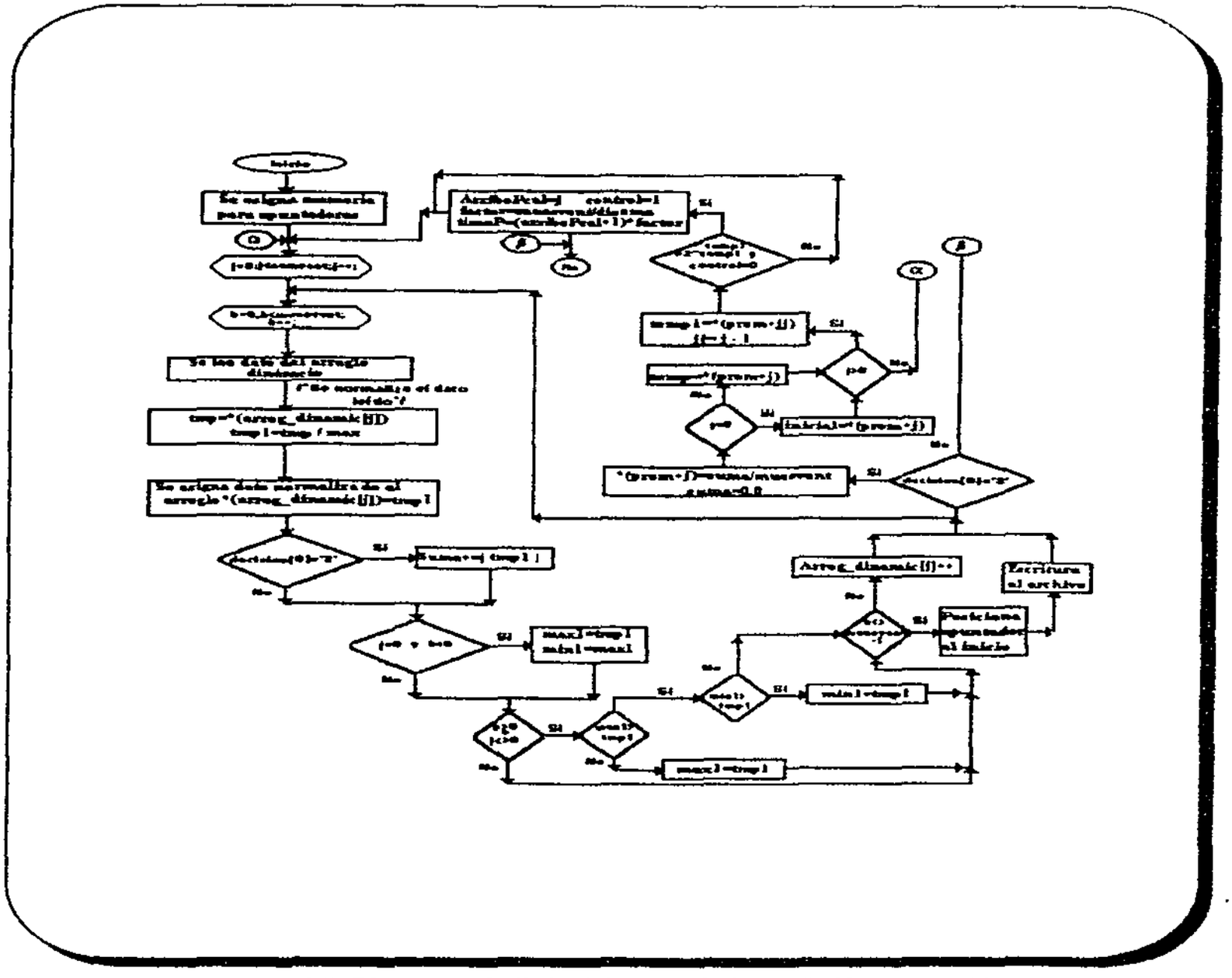


Figura 4.5 Diagrama de flujo del procedimiento de Normalización

IV. 4. 1. 8. Obtención del tiempo del Arribo de la Fase P.

Se tienen dos algotimos para la obtención del tiempo del arribo de la fase "P" y son:

1) El dato es proporcionado por el usuario cuando la fase de aprendizaje es seleccionada.

2) El dato es calculado mediante un procedimiento.

Si el usuario proporcionó el tiempo del arribo, éste es asignado a una variable llamada **timeP** utilizándose en el procedimiento del vaciado de los datos, para saber cómo debe ser categorizada la señal con los valores de salida destinados para ello.

Por el contrario, si el usuario determinó que el propio algoritmo fuera el que calculara este dato, entonces en el procedimiento de Normalización se calcula el promedio de los datos que constituyen a cada patrón, procediendo a comparar el promedio del primero con el del subsecuente, desplazándose un patrón para continuar con la comparación y así sucesivamente.

Si se cumple la siguiente condición se da por hecho que se ha encontrado el arribo:

```
if (( temp >= 2 * temp1)  &&  ( control == 0)) . . . (4 9)  Condición para detectar el arribo de la fase P.
```

donde: temp1 indica el patrón anterior.

La variable **control** garantiza que la primera vez que se cumpla la condición se asignará el arribo de la fase "P" a la variable denominada **arriboPcal,** después de que se encuentra el arribo de la fase la frecuencia de la onda varía dando como consecuencia que se pueda cumplir nuevamente la condición antes especificada.

Con esta condición se establece que cuando el patrón actual sea mayor ó igual que el doble del patrón anterior se estará frente a un cambio significativo en los valores de amplitud. Este criterio puede ser aplicado a señales que no sean tan ruidosas, buscando con esto encontrar un cambio en el nivel de la señal antes del arribo comparado con el nivel de la señal al momento de detectarlo.

IV. 4. 1. 9. Deslizamiento de patrones.

Se posiciona el cursor al principio del archivo que se obtuvo como salida en el procedimiento de Normalización, se procede entonces a determinar el factor de deslizamiento, es decir, el número de muestras que serán deslizadas en cada patrón, este número es calculado dividiendo el número de muestras entre un dato determinado en base a las pruebas realizadas.

Se toma el primer patrón que contendrá tantas muestras como el valor determinado por el usuario, posteriormente se desplazan tantas muestras como se ha determinado, tomando el número de muestras por patrón y así sucesivamente, ejemplificando lo anterior se tiene:

Suponiendo que se tienen 12345678910111213141516171819202122232425262728293O como los primeros 30 datos y que el número de muestras por patrón es de 16 y el factor de deslizamiento de 4 el proceso es el siguiente: 1234567891011121314151 6 serán los datos que contendrá el primer patrón, posteriormente se desplazarán 4 muestras y se tomarán los siguientes 16 datos obteniendo el segundo patrón que contendrá 5678910111213141516171819 20 y así sucesivamente.

Haciéndose notar que las segundas 4 muestras para el patrón 0 serán las primeras para el patrón 1, las terceras muestras del patrón 0 serán las segundas del patrón 1 y las primeras del patrón 3 y así sucesivamente. De este modo las muestras leídas serán asignadas a diferentes patrones pero en distintas posiciones dentro de cada uno .

Una manera simple de realizar lo anterior seria tomar las muestras por patrón asignándolas mediante un ciclo que fuera de 0 al número de muestras - 1 como el primer patrón y después mediante otro ciclo sólo desplazar las muestras indicadas con el factor de desplazamiento volviendo a utilizar el primer ciclo para asignar nuevamente las siguientes muestras y así formar los patrones sucesivos.

Este proceso, de implantarlo seria poco eficiente ya que al tomar muestras que ya han sido asignadas en patrones anteriores el proceso de lectura se vuelve repetitivo redundando en un incremento en el tiempo de procesamiento .

Analizando a fondo este proceso se llegó a una manera óptima de realizarlo, evitando con esto escribir nuevamente los valores leídos Durante el análisis se observó lo siguiente:

El primer patrón no guarda ninguna relación con los siguientes puesto que sólo se toman las primeras m muestras para formar el patrón 0 (m indica el número de muestras por patrón), pero los subsecuentes si guardan una cierta relación, la cual es presentada hasta el cuarto patrón designado como 3 dentro del proceso ya que se inicia desde el patrón 0, en este patrón el número de veces que las siguientes muestras estarán contenidas en los subsecuentes patrones será igual al número de muestras deslizadas, esta relación también determina desde qué patrón se presentará esta regla.

Antes del cuarto patrón el número de veces que las siguientes muestras estarán contenidas en los subsecuentes patrones variará guardando la única relación de incrementarse hasta llegar al patrón en el que se convertirá en un valor constante como ya se mencionó.

Como las pruebas realizadas indicaron que el mejor factor de deslizamiento era de 4 tomando como base 16 muestras por patrón, los patrones se incrementaron el cuadruple. A continuación se explicará cómo se implementó este algoritmo dentro del sistema .

Se leen del archivo tantas muestras como lo indique el factor de deslizamiento asignando dicho valor a la variable denominada *val* entrando a un ciclo que se efectuará mientras el valor de esta variable sea diferente de NULL. Esto indicará en que momento se llegará al final del archivo puesto que val regresará un valor nulo.

Dentro del ciclo mencionado anteriormente 2 ciclos son abiertos, el ciclo externo determinará el número de la muestra que se está procesando y variará de 0 hasta que el índice sea menor que el factor de deslizamiento, en cambio el ciclo interno determinará a qué patrón se le va a asignar la muestra procesada, este ciclo está definido mediante variables desde que el índice es igual a *vent* hasta que es menor ó igual a *cont*. Siendo *vent* el número de patrón de inicio y *cont* el número de patrón con el que se finalizará en cada repetición del algoritmo.

Dentro de estos ciclos los valores leídos del archivo son asignados al arreglo dinámico determinándose algunos criterios para no exceder el número de muestras por patrón lo que causaría perder el apuntador a los valores, dificultando con esto el buen funcionamiento del algoritmo. En ese momento se cierran los ciclos entrando a una fase de verificación para saber en qué momento el patrón ha sido constituido siendo necesario incrementar las variables *vent* y *cont* para asignar los valores correspondientes al siguiente patrón, de la misma manera se determina hasta qué momento la variable *cont* debe incrementarse puesto que ésta es la variable que determina el patrón al que se debe llegar.

Además, como se ha finalizado la asignación de un patrón el apuntador del arreglo dinámico debe ser regresado al principio de éste para utilizarlo con el siguiente patrón.

Es necesario controlar la variable *cont* puesto que llegará el momento en el que ya no se tengan más muestras para constituir un nuevo patrón llegando a la finalización del proceso .Se leen las siguientes 4 muestras del archivo y si el valor de *val* es diferente de nulo el proceso continúa, en caso contrario el proceso es finalizado.

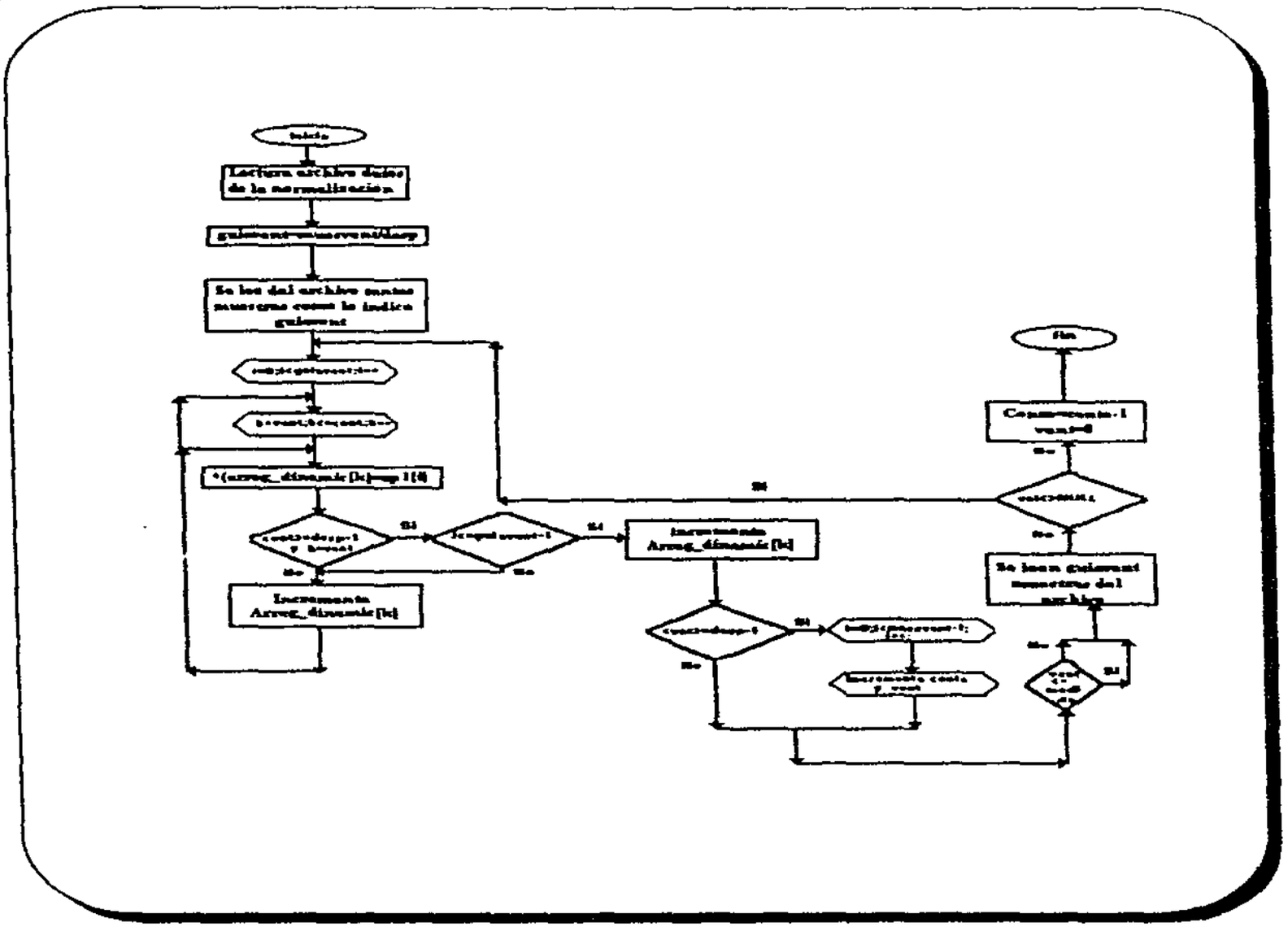


Figura 4.6 Diagrama de flujo del procedimiento de Deslizamiento de muestras.

IV. 4. 1. 10. Copia de datos preprocesados al archivo destinado para la salida.

Se asigna a la variable denominada *val0p* el valor de 0.1 y a *val1p* el valor de 0.9 con el fin de categorizar la señal cuando sea necesario. Posteriormente se abren 2 ciclos , el ciclo externo variará de 0 hasta el número de patrones y el ciclo interno de 0 hasta el número de muestras por patrón.

Dentro de este ciclo se escribe el dato del arreglo dinámico al archivo destinado para la salida. Si el tiempo donde el arribo de P se encuentra fue proporcionado, entonces es necesaario contar con una variable que indique qué avance lleva la señal en cuanto a tiempo (la cual fué denominada ***time***) para comparar posteriormente este valor con el proporcionado por el usuario (variable denominada ***timeP***).

En caso contrario, si el tiempo fué calculado por el algoritmo se toma el patrón en el que se encontró el arribo y en base a él se categoriza a la señal, asignando 0.1 a la salida de cada patrón si el patrón procesado es menor que el patrón en donde se encontró el arribo, en caso contrario se asigna 0.9.
Debido a la forma de la función sigmoidal las salidas pueden nunca alcanzar el valor 0 ó el valor 1, razón por la cual se categorizó la señal con valores de 0.1 para representar al 0 y de 0.9 representando al 1.

j<arriboPcal .... (4.10) Criterio para categorizar la señal con 0.1 como salida de cada patrón, mientras se cumpla con la condición.

j>=arriboPcal .... (4.11) Criterio para categorizar la señal con 0.9 como salida de cada patrón, mientras se cumpla con la condición.

donde, j representa el valor del patrón actual y arriboPcal el patrón donde se encontró el arribo.

Pero como se calcula la variable time así como su correspondiente incremento?. Por ejemplo, si se hubiera seleccionado la fase de deslizamiento para el primer patrón time se calcularía así:

time = Muestras por patrón / (tiempo de grabación / razón de diezmado) . (4.12)

El incremento de time se determinaria de la siguiente manera:

time = Muestras deslizadas por patrón / (tiempo de grabación / razón de diezmado) ..... (4.13)

El tiempo de grabación dividido entre la razón de diezmado dá como resultado el número de muestras por segundo que se tendrán en cada patrón después de la fase de diezmado. Entonces, si se divide el número de muestras por patrón entre el número de muestras por segundo que contiene la señal, se tendrá a la salida el tiempo que el primer patrón representa para la señal. En cambio, el incremento es calculado dividiendo el número de muestras deslizadas entre el número de muestras por segundo .

Por el contrario, si la fase de deslizamiento de ventanas no hubiera sido seleccionada, el cálculo de ***time*** se haría como lo muestra la expresión 4.12, siendo los incrementos del mismo valor determinado por la expresión antes citada.

Se procede entonces a comparar el valor de ***time*** con el valor de ***timeP***, si time es menor entonces al terminar de escribir el primer patrón al archivo de salida se procede a escribir el valor de 0.1 al archivo indicando que todavía no se llega al patrón en el que se encontró el arribo.

Cuando el valor de time es mayor entonces después de escribir los valores correspondientes a ese patrón al archivo se escribirá el valor de 0.9 indicando que el arribo ha sido encontrado en dicho patrón, ese valor es guardado en una variable para ser escrito en el encabezado del archivo de salida para la fase de reconocimiento.

time<timeP ... (4.13) Criterio para categorizar la señal con 0.1 como salida de cada patrón, mientras se cumpla con la condición.

time>=timeP .... (4.14) Criterio para categorizar la señal con 0.9 como salida de cada patrón, mientras se cumpla con la condición.

Si se seleccionó la fase de Reconocimiento lo único que varia en cuanto a todo lo descrito anteriormente es el hecho de que la señal no es categorizada, es decir, no se escriben los valores de salida deseados antes del arribo de P (0.1) y después del arribo (0.9). A continuación se describirán el formato de los archivos de salida que generará el sistema de preprocesamiento de registros sismicos

IV. 4. 2. Descripción de la interface gráfica presentada en el sistema de preprocesamiento.

Es pertinente mencionar que los menus que constituyen el sistema de la Red Neuronal así como el del preprocesamiento de los datos serán descritos de una manera no tan profunda, ya que con esto se pretende solamente dar una idea de la organización de la información mediante la facilidad gráfica que ofrece **OPENWINDOWS,** posteriormente en la etapa de diseño y desarrollo se procederá a explicar más a detalle cada una de las etapas de que consta cada sistema.

El sistema de preprocesamiento de registros sísmicos cuenta con 6 menus disponibles en la Barra de Menus, en esta área se definen las caracteristicas que los conforman, es decir, se realiza la declaración de cada uno de los comandos que forman parte de cada menu así como del proceso asociado a ellos. A continuación se mencionarán los menus declarados en este trabajo:

Archivo: En esta parte se muestra un menu descendente con 3 comandos los cuales son :

Abrir - Se presenta una sub-ventana en donde se pide el nombre del archivo a ser procesado.
En la parte inferior se encuentra un botón denominado Cerrar, el cual una vez tecleado el nombre lo asignará a la variable designada para ello, finalizando la apertura del archivo . El procedimiento al que se hace referencia se denomina *pide archivo*.

La ventana que aparecerá cuando se seleccione este comando se muestra en la figura 4.7. Entonces el nombre del archivo a ser procesado debe ser introducido.

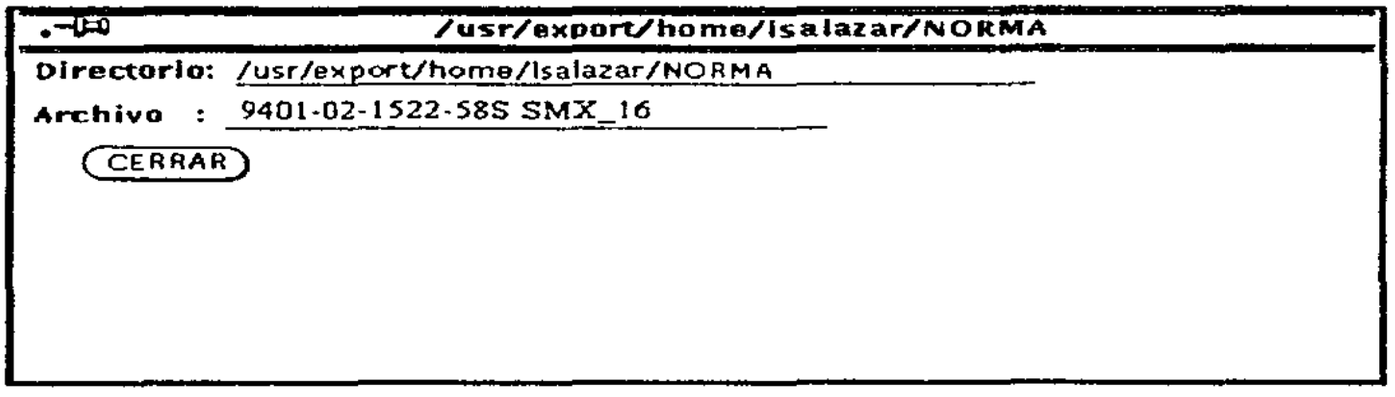


Figura 4.7 Subframe perteneciente al comando Abrir dentro del menu Archivo.

Borrar - Accesa un proceso que borra la pantalla.

Salir .- Indica el fin de sesión. Se manda un mensaje que demanda al usuario si quiere realmente salir de la aplicación, presentándose dos botones con las posibles respuestas, **Cancela** ó **Confirma**,si la respuesta es negativa el comando es cancelado y se retorna a la pantalla principal, en el caso contrario , se sale completamente de la aplicación. El procedimiento al que se hace referencia se denomina *salir* .

Parámetros.- Se muestra también un menu descendente con los siguientes 2 comandos:

Lista Etapas.-Se presenta un rectángulo que contiene precisamente la lista de las etapas que constituyen la aplicación, esta lista tiene el objeto de identificar más al usuario con el sistema, proporcionándole un poco de información sobre el proceso que vendrá a continuación

Este rectángulo está hecho no con los objetos de OPENWINDOWS que definen el ambiente de ventanas, sino con una función que dibuja una línea (proporcionando las coordenadas del inicio y fin de la misma) en la sección de graficación definida anteriormente. Para dibujar el rectángulo e imprimir la lista de las etapas se utilizan las siguientes 3 instrucciones

XDrawRectangle(dpy, xwin, gc, 300, 20, 500, 190) .- Dibuja el rectángulo
donde:

dpy, xwin y gc son parámetros propios del sistema que lo preparan para dibujar la línea deseada habilitando el display.

Las coordenadas siguientes definen el área, las primeras dos representan el primer punto donde comenzará a ser dibujado el rectángulo, y las 2 últimas el punto de finalización del área. Esta instrucción se utiliza en el procedimiento denominado *encabezado*, al igual que la siguiente instrucción sprintf :

sprintf(buffer,"mensaje a ser impreso en el área delimitada por el rectángulo"). Guarda en la variable buffer el mensaje que se desea aparezca dentro del área del rectángulo.

En el mismo procedimiento se define una función denominada **texto** con ciertos parámetros, por ejemplo: **texto(dpy, xwin, gc, 365, 65, buffer)**.- en donde 365 y 65 indican las posiciones tanto en X como en Y. La función texto a su vez contiene la siguiente instrucción:

XDrawString(dpy, xwin, gc, X1, Y1, cadena, longitud de cadena).- Imprime el mensaje asignado en la variable cadena, dicha variable es pasada por parámetro de un procedimiento a otro. El procedimiento al que se hace referencia es denominado *prepara canvas*.

Selección parámetros.- Se inicia la petición al usuario de los parámetros requeridos por el sistema para procesar los eventos sísmicos. Cabe mencionar que en esta parte se utilizan procedimientos en cascada (utilización de un procedimiento dentro de otro).

La petición de parámetros empieza en el procedimiento *parámetros*, en el cual aparece una sub-ventana(subframe) que pide el **número de muestras** , mostrando los valores posibles los cuales están entre 16, 32, 64 ó 128 muestras , en la parte inferior de todos estos procedimientos en cascada se encuentra un botón que al ser presionado dá por finalizado el procedimiento haciendo referencia al siguiente, en este caso cerrar_mues el cual asigna el valor seleccionado a la variable *muesvent* , tal como lo muestra la figura 4.8.

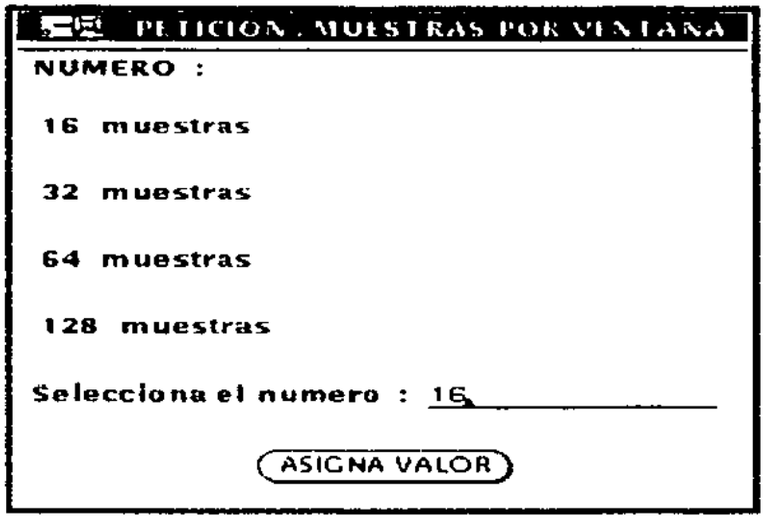


Figura 4.8 Subframe perteneciente al comando selección parámetros dentro del menu Parámetros. Referente al número de muestras con que serán procesados los patrones .

A su vez se abre otra ventana en donde se pedirá otro parámetro tratándose en esta ocasión de la **razón de diezmado**, siendo los posibles valores 2 (50 muestras por segundo) ó 5 (20 muestras por segundo), dicho valor es asignado a la variable *tamdiez* a través del procedimiento cerrar_razon, la figura 4.9 muestra la apariencia del subframe destinado en la petición de la razón de diezmado.

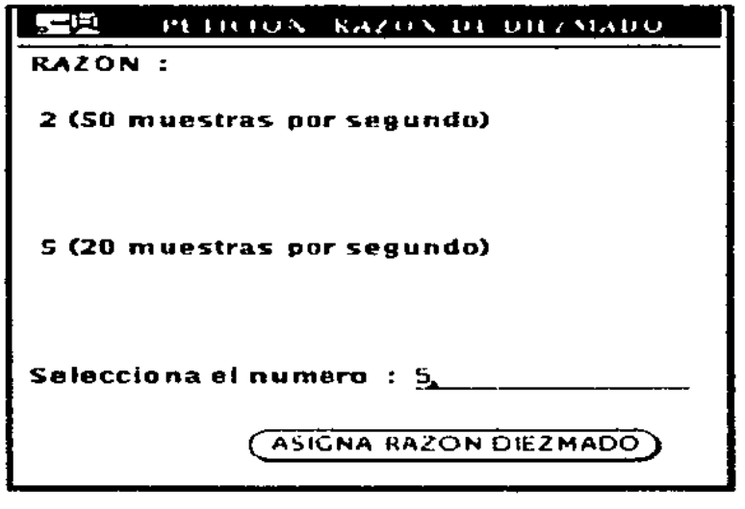


Figura 4.9 Subframe en donde se proporciona la razón de diezmado que se le aplicará a la traza sísmica.

En la figura 4.10 se muestra el siguiente parámetro (designado dentro de los procedimientos en cascada de los cuales se ha estado haciendo referencia),la traza a ser procesada. Entre los posibles valores se encuentran las 16 estaciones sismológicas que componen la Red Sismotelemétrica ***"SISMEX"*** .el valor proporcionado es asignado a la variable *canal* mediante cerrar_traza .

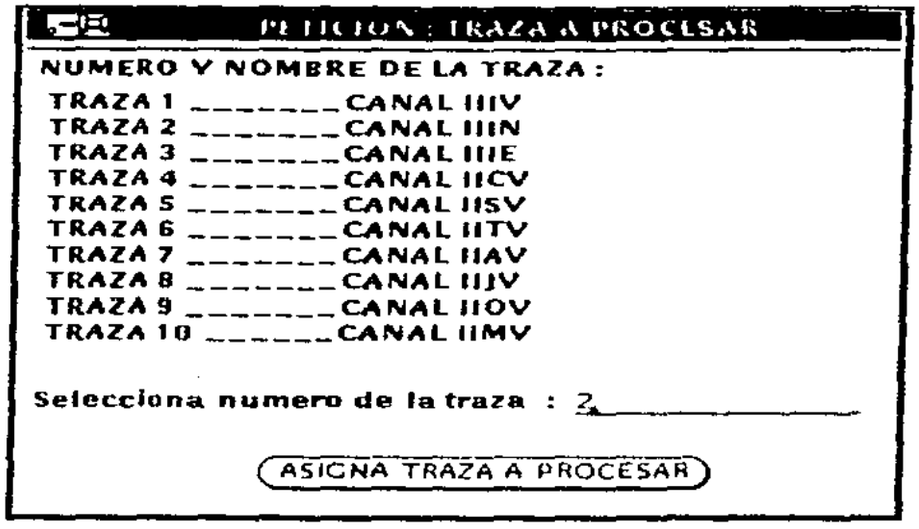


Figura 4.10 Subframe que demanda la traza a ser procesada.

La figura 4.11 muestra la fase para la cual serán procesados los datos sismicos, fase de aprendizaje, reconocimiento ó ambas, valor asignado a *fase* por medio de cerrar_decide.

Ahora bien, el último parámetro es el **tipo de proceso** que le será aplicado a la señal teniendo como posibles opciones: STA de 5 muestras, STA de 20 muestras ó Integración, siendo asignado el proceso seleccionado a la variable llamada *nuproceso* mediante cerrar_proc.

Posteriormente, aparece una sub-ventana en donde se le pregunta al usuario si desea utilizar deslizamiento de ventanas figura 4.12.

Figura 4.11 Subframe en donde se proprociona la fase seleccionada para procesar la señal.

Figura 4.12 Subframe que demanda el deslizamiento de ventanas.

Antes de describir las sub-ventanas que son utilizadas para la etapa central del preprocesado, es necesario hacer referencia a la parte del proceso donde se pide, ya sea el tiempo de arribo de la fase P ó si se desea que el algoritmo lo calcule. La figura 4.13 muestra el subframe de petición de estos factores.

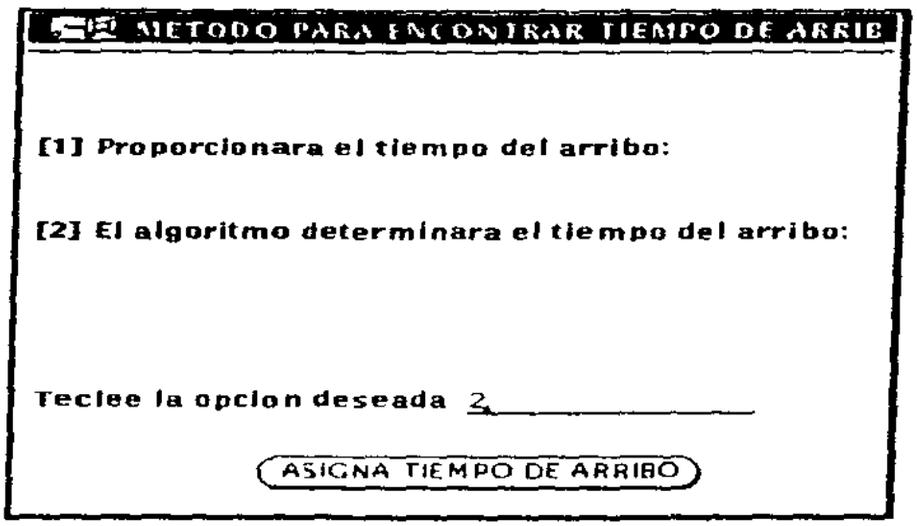


Figura 4.13 Petición del método para obtener el tiempo de arribo de la fase P.

Al finalizar la petición de parámetros aparece una ventana en donde se le indica al usuario el siguiente paso a realizar como se muestra en la figura 4.14

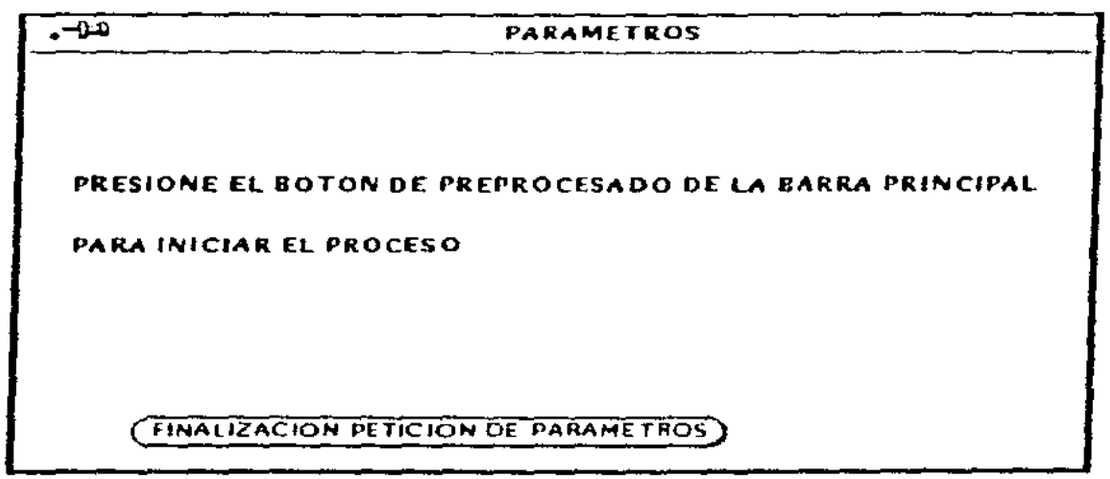


Figura 4.14 Subframe que aparece al momento de finalizar el preprocesamiento.

Preprocesamiento.- En esta etapa el menu solo contiene un comando denominado **INICIA.** El cual, inicia la etapa del preprocesado de los eventos sísmicos. Dentro de él están contenidas todas las etapas contempladas en el proceso. Cuando la fase de preprocesamiento de la traza seleccionada ha finalizado, aparece la siguiente ventana con todas las etapas realizadas como lo indica la figura 4.15.

Figura 4.15 Subframe que aparece al momento de finalizar el preprocesamiento.

Entonces los datos más significativos son expuestos dentro de dos subframes para que el usuario final conozca el resultado del preprocesado, las figuras 4.16 y 4.17 lo ejemplifican

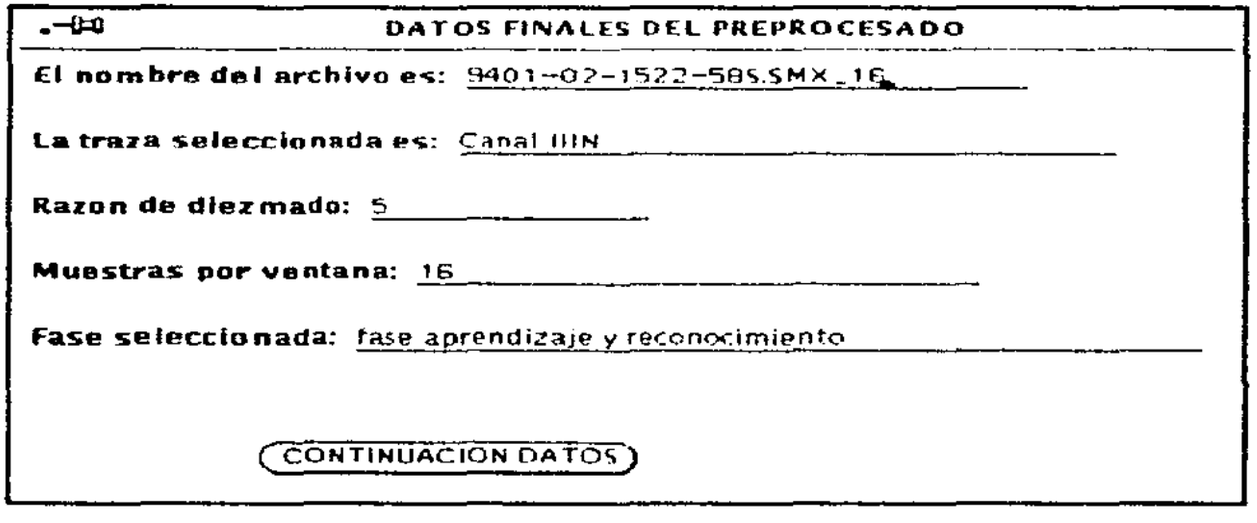


Figura 4.16 Subframe que muestra la primera parte de los resultados del preprocesamiento.

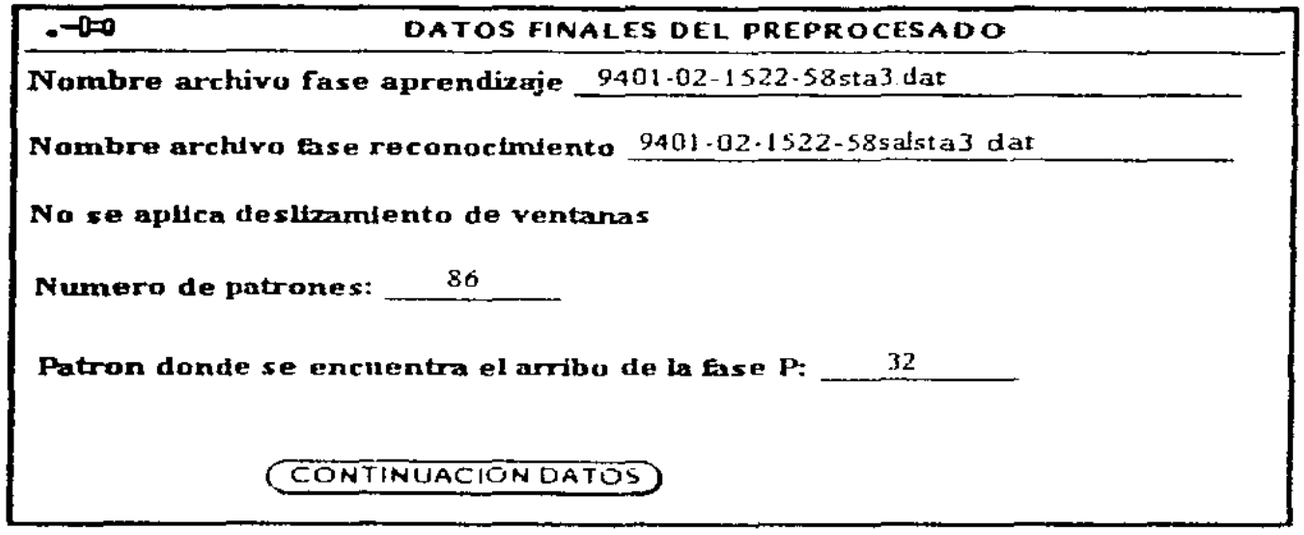


Figura 4.17 Subframe que muestra la segunda parte de los resultados del preprocesamiento.

Salir.- Este es solo un comando no propiamente un menu, igual al mencionado en ***Archivo***.

IV. 5. 3. Formato de los Archivos de Salida .

El archivo de salida cuenta con 2 formatos: Formato para la fase de Aprendizaje asi como el Formato para la fase de Reconocimiento

Formato para el Aprendizaje . - En el encabezado se graba la siguiente información :

Indicador del número de patrones de entrada
Número de patrones de entrada
Número de muestras por patrón
Nombre de la traza seleccionada
Indicador de deslizamiento de las ventanas
Indicador del patrón donde fué encontrado el arribo
Patrón donde fué encontrado el arribo

Grabándose después los datos procesados tal como lo muestra la tabla 4.18.

Datos	*Salidas*
Datos pertenecientes al patrón 1	**Valor de salida deseado para el patrón 1**
Datos pertenecientes al patrón 2	**Valor de salida deseado para el patrón 2**
Datos pertenecientes al patrón 3	**Valor de salida deseado para el patrón 3**
. . . . .	. . . . .
. . . . .	. . . . .
Datos pertenecientes al patrón N	**Valor de salida deseado para el patrón N**

Tabla 4.18 Descripción del formato de Salida para la fase de Aprendizaje.

Cada traza procesada se graba en un archivo, es decir, se tendrán tantos archivos como trazas procesadas del mismo registro sísmico.

Formato para el Reconocimiento .- En el encabezado se graban algunos parámetros más que en el formato para el Aprendizaje, la información es grabada de la siguiente manera:

Indicador del número de patrones de entrada
Número de patrones de entrada
Número de muestras por patrón
Razón de diezmado
Fecha del registro
Razón de muestreo
Nombre de la(s) traza(s) seleccionada(s)
Indicador de deslizamiento de las ventanas
Indicador del patrón donde fué encontrado el arribo
Arribo donde fué encontrado el arribo.

La manera de grabación de los datos también se realiza de una manera diferente. Los datos son procesados de la siguiente manera:

Datos
Datos pertenecientes al Patron 1
Datos pertenecientes al Patrón 2
Datos pertenecientes al Patrón 3
. . . . .
. . . . .
Datos pertenecientes al Patrón N

Tabla 4.19 Descripción del formato del Archivo para la fase de Reconocimiento.

V. IMPLANTACION DE LA RED NEURONAL.

V. 1. Diseño y Desarrollo de la Red Neuronal.

Primeramente se declaran todas las variables globales utilizadas durante la ejecución del programa , declarando primero todo lo correspondiente al manejo del ambiente gráfico, posteriormente se declaran mediante la declaración define, el número máximo de los valores que constituirán más tarde la Arquitectura de la Red, es decir, número máximo de capas ocultas, número máximo de características de salida , número máximo de patrones de entrada, y el número máximo de nodos de entrada .

defines	Variables	Valores	Descripción
# define	NMXHLR	5.00	Número máximo de capas ocultas
# define	NMXOATTR	3.00	Número máximo de nodos de salida
# define	NMXINP	1000.00	Número máximo de patrones de entrada
# define	NMXIATTR	300.00	Número máximo de nodos de entrada por patrón
defines	**Variables del control del aprendizaje**	**Valores**	**Descripción**
# define	SEXIT	3.00	Exito en aprendizaje
# define	RESTRT	2.00	Reempezar
# define	FEXIT	1.00	Salir después de no tener éxito
# define	CONTNE	0.00	Continuar con los cálculos

Tabla 5.1 Variables definidas en el Sistema mediante la instrucción define.

Después se procede a la apertura del archivo y la selección de los parámetos que constituirán la Arquitectura de la Red mediante la interface gráfica proporcionada por los objetos manipulables en ***OPENWINDOWS*** (antes mencionada).

V. 1. 1. Parámetros proporcionados por el Usuario.

Seleccionando el botón correspondiente a ***Parámetros*** el sistema mostrará unos cuadros de diálogo para que el usuario proporcione los datos necesarios para conformar la Arquitectura de la Red, con el fin de adecuarla a las características de cada evento sísmico.

Estos parámetros son importantes para obtener un buen Aprendizaje tanto como un buen Reconocimiento, por lo mismo se tiene que jugar con ellos para determinar qué valores son los óptimos para cada evento y con esta información deducir qué rango de valores es el óptimo para el conjunto de entrenamiento seleccionado .

Los parámetros mencionados son:

Razón de aprendizaje(valor por default=0.9) designada como eta.
Momentum(valor por default=0.7) designado como alpha.
Error máximo total(default=0.01) designado como maxe.
Error máximo individual(default=0.01) designado como maxep.
Número máximo de iteraciones(default=1000) designado como cnt_num.
Número de Capas ocultas designado como nhlayer.

Posteriormente se declara un ciclo que va desde el 0 hasta el número de capas ocultas con el fin de asignar el número de unidades por cada una de ellas De igual manera el número de unidades de salida es asignado a la última capa de la Red

V. 1. 2. Lectura del archivo de entrada a la fase de Aprendizaje y extracción de los datos grabados en el encabezado .

En el procedimiento **pide_archivo** se lee el archivo de entrada especificado por el usuario durante la sesión, haciendo referencia a éste al momento de presionar el botón dentro del área de Menus llamado *Archivo.*

Habiendo abierto el archivo con el que se va a trabajar, se accesa al procedimiento denominado **cerrar_par,** el cual extrae los datos grabados en el encabezado como lo son: Número de patrones de entrada, número de trazas a entrenar, número de muestras por patrón, el nombre de la traza ó trazas procesadas, así como el número de unidades en la capa de salida, estos valores serán presentados a fin de informar al usuario final del sistema sobre los datos con que el sistema efectuará su función.

Se procede entonces a asignar dentro de un ciclo que variará de 0 hasta el número de patrones -1 cada muestra de éstos al arreglo bidimensional denominado *input [número de patrón] [número de muestra].*

Así mismo se asignan a otro arreglo bidimensional denominado *target [número de patrón] [número de unidades de salida]* los valores de salida asociados a cada patrón .

Ya constituida la Red se procede a seleccionar mediante el botón ***Red Neuronal,*** la fase para la cual los patrones serán procesados siendo éstas Aprendizaje ó Reconocimiento

V. 1. 3. Aprendizaje.

Si la fase de aprendizaje es seleccionada se accesa el procedimiento **aprende,** cada vez que se accese a él se incrementará un contador con la ayuda del cual se sabrá cuantas veces se ha seleccionado la fase de aprendizaje. Después de que finaliza esta fase para una determinada señal sísmica se obtiene una matriz de pesos, la cual será almacenada en un archivo con el mismo nombre que el de entrada pero con la terminación "_w.dat", a fin de reconocer de qué evento se trata.

Esto con la finalidad de utilizarlo en la fase de aprendizaje de otro evento sísmico, es decir, si el evento que se quiere presentar para ser aprendido por la Red es similar en forma a otro evento ya presentado, entonces se le indica a la Red que se quiere utilizar la matriz de pesos obtenida durante el aprendizaje de dicho evento, de esta manera se optimiza dicha fase para el nuevo evento, puesto que es más fácil para la Red

asociar patrones en base a una matriz de pesos a la cual se llegó anteriormente que empezar asignando nuevamente valores aleatorios a dicha matriz como paso inicial para los posteriores cálculos.

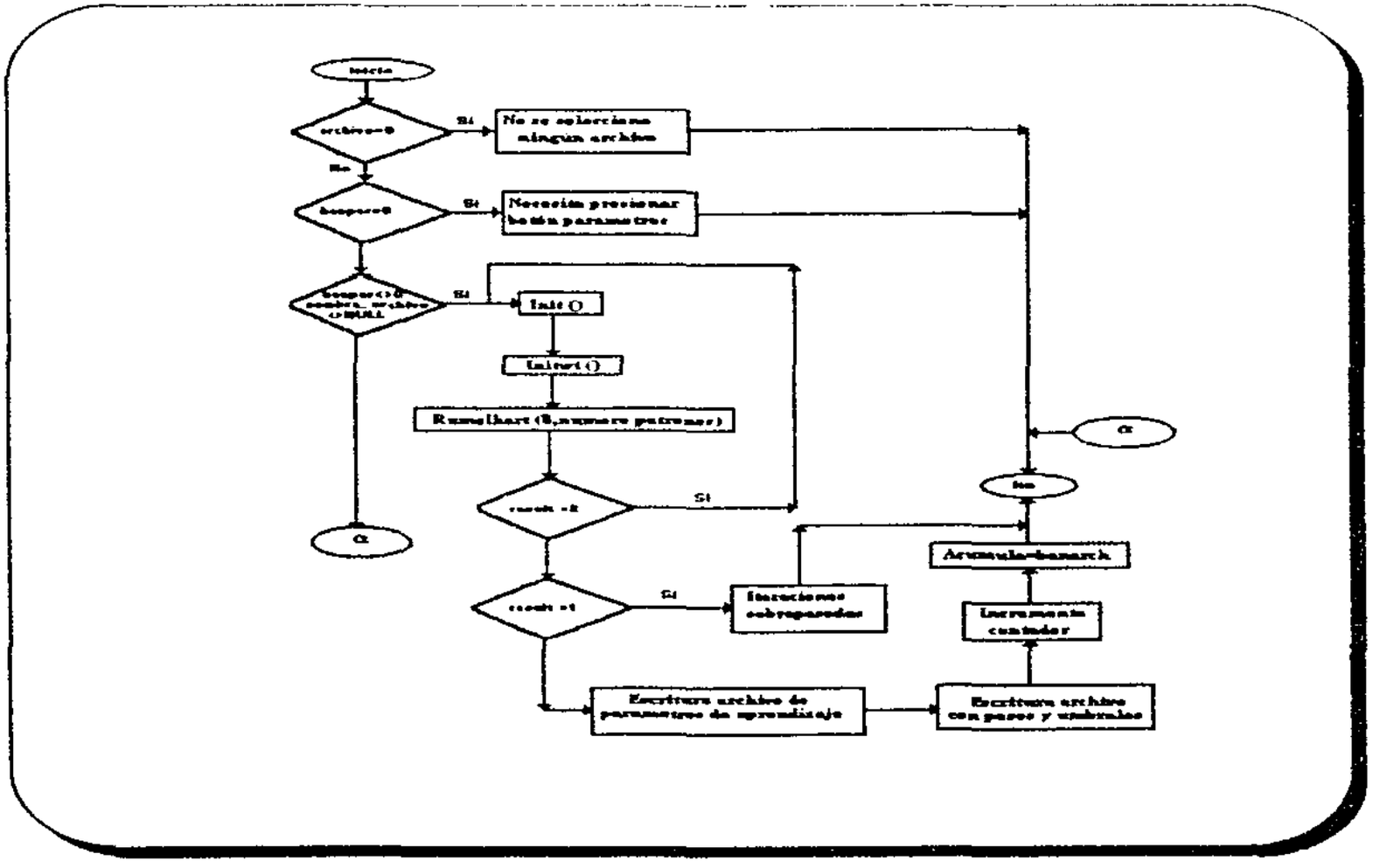


Figura 5.2 Diagrama de flujo del procedimiento Aprende.

V. 1. 3. 1. Dimensionamiento de las estructuras utilizadas en el Aprendizaje.

Después de teclear la respuesta a la pregunta de si se desea utilizar la matriz de pesos a que se llegó en un anterior aprendizaje se llama al procedimiento init en donde se dimensionan los apuntadores a utilizar dependiendo este proceso del número de capas totales de que consta la Red.

La Red está constituida por la capa de entrada, la(s) capa(s) oculta(s) y la capa de salida, por lo tanto el número total de capas será el número de capas ocultas + 2. Ahora bien, el dimensionamiento se calcula de la siguiente manera:

Se multiplica el número de muestras que constituyen la primer capa + 1 (unidad de bias = umbral) por el número de muestras de la siguiente capa, esto con el fin de que todas las unidades esten conectadas con las unidades de la capa siguiente.

Es pertinente mencionar que el umbral es tratado como un peso de unión de un nodo virtual cuyo valor de salida es la unidad.

Como ya se mencionó se asigna el valor de 1 a todos los umbrales ó unidades de bias incrementándose a su vez el contador que acumula el número de veces que el aprendizaje ha sido realizado sobre un evento sísmico.

V. 1. 3. 2. Inicialización de la Matriz de Pesos.

Dentro de un ciclo mientras el valor de la variable ***result*** sea igual a 3(valor correspondiente a ***RESTR*** que indica que el proceso no ha finalizado todavia) , se llama al procedimiento **initwt** donde se inicializan los pesos por primera vez con números aleatorios entre - 0.5 y + 0.5 utilizando para esto la función generadora de números aleatorios denominada ***randseed***.

Si se decidió utilizar la matriz de pesos a la que se llegó en un aprendizaje anterior se procede a abrir el archivo correspondiente para asignar los valores a la estructura definida para almacenar los datos de la matriz de pesos ***wtptr*** , a la vez se asigna el valor de 0 a la matriz de deltas (matriz en donde las variaciones hechas a los pesos serán almacenadas), si la respuesta hubiera sido negativa se le asignan a la matriz inicial de pesos valores aleatorios y la matriz de deltas será inicializada con 0's

V. 1. 3. 3. Regla Delta.
Propagación hacia adelante.

Después de esto se entra a la fase de procesamiento de los datos mediante la Regla Delta, denominada con el nombre de la función **rumelhart** pasándole como parámetros (0, número de patrones), en ésta se abre un ciclo que utiliza los parámetros proporcionados por el procedimiento anterior, el cual varia de 0 hasta el número de patrones, es decir, los cálculos que serán realizados se efectuarán para cada patrón , se llama entonces al procedimiento **forward** pasándole a su vez como parámetro el número del patrón que se está procesando, en este procedimiento se realizan los cálculos desde la primera capa de la red hasta la última, para lo cual se abre un ciclo que variará desde el valor 0 hasta el número de muestras de ese patrón, asignando los valores correspondientes a la capa de entrada a la estructura ***outptr*** .

Posteriormente se abre un doble ciclo , el primero va desde 1 hasta el número de capas ocultas + 2 a este se le accesará mediante la variable ***m***, este cubrirá el cálculo de la salida para la capa oculta y la capa de salida, el segundo ciclo va de 0 hasta el número de muestras tanto de la capa oculta como de la de salida y este será accesado mediante la variable ***n***.

Se inicializa entonces la variable llamada ***net*** con el valor de 0, habiendo hecho esto se abre un tercer ciclo que va de 0 hasta el valor de muestra de la capa tratada -1 al cual se le suma la unidad, dicho ciclo será denotado por la variable ***p***, la creación de este tercer ciclo tiene la finalidad de conectar la capa de entrada con la capa subsecuente generando las conexiones para asignar los valores de la matriz de pesos,por medio de la suma de la unidad se contempla la unidad de bias ó umbral que será adicionadaen cada capa. En este

último ciclo se le asignará a una variable llamada *offset* un valor ascendente incrementado en 1 que irá desde el 0 hasta tantas interconexiones se hagan entre 2 capas consecutivas.

Teniendo este dato se procede a calcular la sumatoria de los valores de entrada que llegan a cada uno de los nodos de la capa subsecuente, en esta parte se simula lo que ocurre con una neurona biológica al momento de recibir ciertos impulsos, pudiendo ser tanto inhibitorios como excitatorios, dependiendo de lo que predomine más se tendrá un valor a la salida cuyo efecto logrará excitar ó inhibir a los nodos de la capa siguiente

Si la neurona es excitada mandará el impulso de salida hacia otra neurona , en caso contrario no estará en posibilidad de conectarse con otra neurona(nodo).Esta sumatoria de valores se logra mediante la ecuación (1 1) a la cual se hizo referencia en el capitulo I.

Ahora bien, la fórmula que expresa dicha sumatoria dentro de la lógica del programa se expresa como:

net += * (wtptr[m-1]+offset) * (*(outptr[m-1]+p)) . . . . (4.16)

Al terminar de sumar todas las entradas a los nodos se procede a aplicar la función de activación que dará a la salida un cierto valor, el cual será a su vez la entrada a los nodos de la capa subsecuente, los valores calculados aplicando la función de activación son asignados a la estructura denominada ***outptr [número de la capa] [número del nodo de dicha capa]*** .

En este tipo de algoritmo (Retropropagación) se utiliza la función sigmoidal, la cual asumirá un valor máximo de +1 y mínimo de 0 Dicha función es expresada mediante la ecuacion (3.5) asi como la (3.6).

Siendo expresada dentro del programa como:

***(outptr[m]+n) = 1 / (1+exp(-net)) . . . . (4.17)** Función Sigmoidal.

Estos cálculos se realizan dentro de los ciclos mencionados anteriormente calculando la sumatoria de las entradas para todos los nodos entre 2 capas, y asi hasta llegar a la capa de salida. Donde el valor de salida para cada uno de los nodos que constituyen dicha capa son asignados a una estructura denominada ***outpt [número de patrón] [número de nodo de la capa de salida].***

A los cálculos realizados desde la capa de entrada hasta la de salida para obtener los valores que serán comparados con los deseados, se le llama aprendizaje hacia adelante ó *"feedforward"*

V. 1. 3. 4. Propagación hacia atrás de los Errores.

Posteriormente teniendo los valores calculados hasta la capa de salida se procede a propagar el error (diferencia entre el valor de salida calculado - valor de salida deseado) hacia atrás, empezando la fase de retropropagación desde la capa de salida hasta la capa de entrada ó *"backward"*.

El error es propagado sobre las capas intermedias hasta la capa de entrada y es calculado para todas las unidades Ademas , es usado para ajustar cada conexión de pesos Para calcularlo se declara un ciclo denotado por ***m*** , el cual variará de 0 hasta el número de nodos de la capa de salida. La estructura que guardará los errores se denomina ***errptr***, la fórmula utilizada para calcular el error en la

capa de salida es expresada mediante la ecuación (3.8). Siendo expresada dentro de la lógica del programa como:

*** (errptr[nhlayer+1]+m) = (target[i][m] - out) * (1-out) * out** . . . (4.18) Error en la capa de salida.

Ahora bien , la forma de calcular el error para las capas ocultas así como para la de entrada varía siendo más difícil el cálculo debido a que las salidas deseadas no son explícitamente definidas para ellas, aquí el término de error es calculado de los errores propagados através de la red.

Para calcular el error en las capas ocultas y en la de entrada se procede a abrir 3 ciclos, el primero será denotado por la variable ***m*** la cual variará del número de capas ocultas +1 hasta que el valor sea mayor ó igual a 1, es decir , el ciclo será declarado de la siguiente manera.

for(m=nhlayer+1; m>=1; m++) . . . . . (4.19) Ciclo que retrocede de la capa de salida a la primer capa oculta para calcular el error en estas capas.

El error de la capa de salida ya ha sido calculado, por lo tanto se necesita propagar hacia atrás para saber qué proporción de cambio será sumada a la matriz de pesos para cada nodo, es por esto que el ciclo toma en cuenta desde la capa de salida hasta la capa de entrada.

El segundo ciclo se denotará por la variable ***n*** e irá desde 0 hasta el número de nodos de la capa que se está tratando -1 sumándole el término de bias ó umbral, es decir, se declarará como.

for(n=0; n<nunit[m-1]+1; n++) . . . . . (4.20) Ciclo que varía de 0 al número de nodos de la capa denota por el ciclo anterior + el término de bias ó umbral.

Inicializando al error con el valor de 0, ***(errptr[m-1]+n)=0.0;** El tercer ciclo denotado por ***p*** el cual varía de 0 hasta el número de nodos de la capa considerada, declarándose como:

for(p=0; p<nunit[m]; p++) . . . . . (4.21) Ciclo que varía de 0 hasta el número de nodos en la capa estimada

Se utiliza nuevamente la variable llamada ***offset*** la cual tiene la misma función que ya ha sido explicada anteriormente, procediendo a calcular la variación que será asignada al peso determinado, para asignar las variaciones de pesos(ó deltas) se declaró una estructura identificada como ***delw [capa tratada] [número de interconexión denotado por la variable offset].***

El valor delta representa la modificación realizada en los pesos con la finalidad de disminuir el error, dicha modificación se realiza en cantidades proporcionales al error generado. El cálculo de este valor es expresado mediante la ecuación (3.7). Expresado dentro de la lógica del programa como:

***(delw[m-1]+offset)=eta*(*(errptr[m]+p))*(*(outptr[m-1]+n))+alpha*(*(delw[m-1]+offset))**. . . . (4.22)

Se procede a calcular el error en las capas intermedias y capa de entrada. En la ecuación (4.23) se calcula el error para cada nodo involucrado en las capas intermedias, en donde ahora el error calculado en la capa de salida fungirá como si fuera el valor de entrada al nodo, multiplicándose a su vez por el valor de peso, haciendo la analogía del proceso "feedforward"

Lo único que cambia para calcular el error de la capa de entrada es el valor tomado como entrada a cada uno de los nodos, el cual en este caso será el error calculado en las capas intermedias.

```
*(errptr[m-1]+n)+=*(errptr[m]+p))*(*(wtptr[m-1]+offset)) .
```

(4.23) Suma de los errores de cada neurona en las capas intemedias y de entrada.

El valor delta es entonces calculado basándose en el error obtenido en la capa de entrada, siendo sumado al valor de la matriz de pesos para tener una proporción de cambio en los pesos, dependiendo de qué tan cercano fué el valor de salida calculado con respecto al valor de salida deseado.

Al ajustar los pesos en la fase backward para minimizar el error se está realizando en sí el aprendizaje. La propagación hacia atrás aplica lo que se conoce como aprendizaje supervisado.

V. 1. 3. 5. Cambio en los Pesos.

Existen 2 maneras para ajustar los pesos durante la fase de entrenamiento, el llamado remplazo simultáneo ó el remplazo sucesivo. En el remplazo simultáneo utilizado en este trabajo los ajustes de pesos ó deltas son calculados y almacenados para cada patrón de entrada. No utilizándose inmediatamente para la actualización de los pesos sino hasta que se han terminado de presentar todos los patrones de entrada a la Red, a esta acción se le denomina ciclo computacional

En cambio en el remplazo sucesivo los deltas son calculados para cada patrón de entrrenamiento, los pesos de la red son inmediatamente actualizados una vez que todos ellos son conocidos y antes de que el siguente patrón de entrenamiento sea presentado

Al final de la presentación y realización de los cálculos descritos para cada patrón la matriz de pesos es remplazada por la que surgio en el proceso, esta matriz no será utilizada hasta que todos los patrones sean procesados por la Red, en ese momento el proceso empieza desde presentar el primer patrón ahora con los nuevos valores de la matriz de pesos para ajustarlos y verificar si el valor calculado para la capa de salida se asemeja ó es igual al valor deseado , si esto no sucediera todavía el proceso se repite hasta que se cumplan las restricciones proporcionadas por el usuario

Teniendo esto se procede a calcular el error cuadrático para cada patrón asignado al arreglo ***ep[número de patrón]***, obteniendo el valor absoluto de la resta entre el valor de salida deseado y el valor de salida calculado.

Para lo cual es necesario abrir un ciclo desde 0 hasta el número de capas definidas mediante la arquitectura de la Red, denotado por m.

```
for(m=0; m<nunit[nhlayer+1]; m++) . . . .
```

(4.24) Ciclo que avanza por cada capa para calcular el error de cada patrón.

```
ep[i]+=fabs((target[i][m] - *(outptr[nhlayer+1]+m)))  . . .
```

(4.25) Error de cada patrón.

```
err_curr += ep[i] * ep[i]  ...
```

(4.26) Error cuadrático

Posteriormente se calcula el error normalizado descrito en la ecuación (4.0). Expresado dentro de la lógica del programa como:

err_curr = 0.5 * err_curr / ninput ..... (4.27) Error normalizado.

El error normalizado del sistema es utilizado como una de varias condiciones para verificar si el aprendizaje debe terminar.

V. 1. 3. 6. Condiciones de Finalización del Aprendizaje.

Posteriormente se verifica la condición para terminar el aprendizaje, esto se realiza mediante el procedimiento **introspective** con los siguientes parámetros (0, número de patrones) el resultado que se genere en dicho procedimiento se asignará a la variable ***result*** , para que el sistema dé por terminado el procedimiento, esta variable debe tener un valor diferente a 0.

Dentro de este procedimiento se verifica primero el número máximo de iteraciones, si este ha sido alcanzado ó sobrepasado se retorna la variable ***FEXIT*** (el cual se mencionó en la explicación de los defines), este tiene el valor de 1 e indica salir después de no tener éxito, por lo que al regresar y verificar si el valor de result es igual a 0 se dará por terminado el proceso de aprendizaje.

En caso contrario se verifica que el error normalizado sea lo suficientemente pequeño, para lo cual se enciende una bandera llamada ***flag***, se abre entonces un ciclo que varia de 0 al número de patrones, si ***ep[número de patrón]*** es menor ó igual al error máximo permitido para cada patrón entonces se incrementa el valor de la variable ***nsnew***, en caso contrario la bandera es apagada.

Si la bandera está encendida entonces se retorna la variable ***SEXIT*** la cual tiene el valor de 3 e indica que se logró tener éxito en el aprendizaje por lo tanto el proceso de aprendizaje finalizará.

Ahora bien, si no se han cumplido estas condiciones se procede a verificar si el error total del sistema es lo suficientemente pequeño, es decir, si el error es menor ó igual al error máximo proporcionado por el usuario se retorna el valor de ***SEXIT*** sucediendo lo mismo que en el caso explicado anteriormente.

En el caso de que ninguna de las restricciones se hayan cumplido se retorna el valor de 0 provocando que el proceso se inicie desde el principio empezando con el primer patrón y realizando todos los pasos ya explicados.

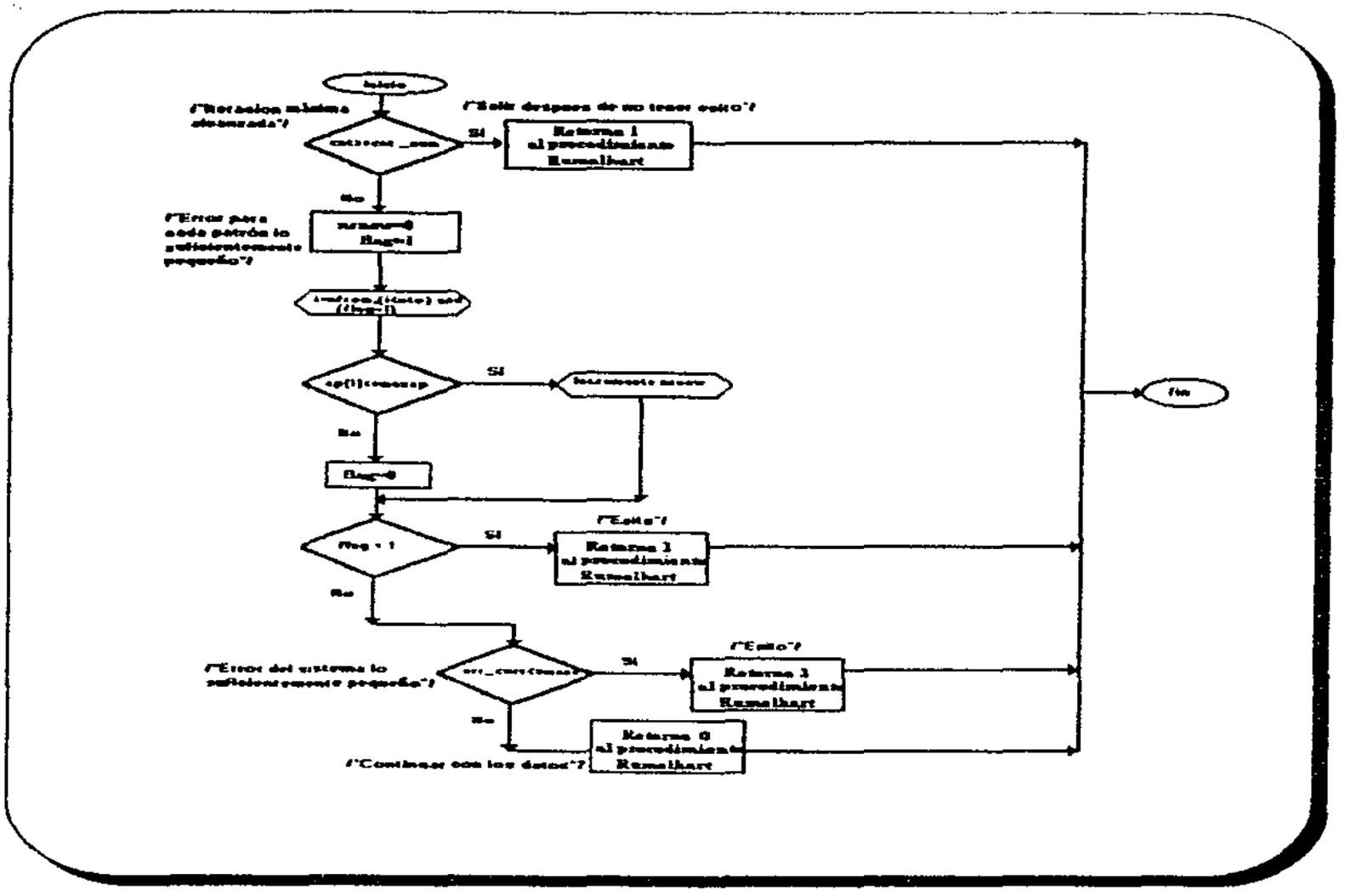


Figura 5.3 Diagrama de flujo del procedimiento introspective(condiciones de finalización del Aprendizaje).

V. 1. 3. 7. Finalización del Aprendizaje.

Cuando la fase de aprendizaje ha finalizado entonces se procede a abrir un ciclo que va desde el 0 hasta el número de patrones, llamando por última vez al procedimiento denominado **forward** mediante el cual se calculará el valor de salida para cada patrón, si ya se llegó al valor óptimo que determinó que el proceso de aprendizaje finalizara significa que se llegó a una matriz de pesos óptima para esa señal sísmica por lo que realizando una vez más el proceso se asegura que el valor de salida será lo más cercano al valor que se espera .

Se imprimen en la pantalla los pesos a los que se llegó en este último proceso, así como el número del patrón y su correspondiente valor de salida calculado por la Red y el valor deseado.

V. 1. 4. Escritura de los parámetros del Aprendizaje a un Archivo.

Entonces se llama al procedimiento **dwrite** para la escritura del archivo tipo texto con extensión _v.dat que contendrá los parámetros del aprendizaje pasándole como parámetro el nombre del archivo, el objetivo de la creación de este archivo es tener los datos significativos del proceso almacenados para su posterior utilización.

El encabezado tendrá el siguiente formato:

nombre variable	significado de la variable
ninput	Número de patrones de entrada
noutattr	Número de nodos de salida
ninattr	Número de nodos de entrada en c/patrón
eta	Razón de aprendizaje
alpha	Momentum
nhlayer	Número de capas ocultas
cnt_num	Número de iteraciones dado por el usuario.
nunit[i] donde i designa el número de capa	Número de nodos en cada capa
cnt	Número de iteraciones efectuadas durante el proceso
err_curr	Error normalizado del sistema
outpt[i][j] donde: i designa el número de patrón j designa los nodos en la capa de salida	Salida a la que se llegó en el proceso de aprendizaje en cada patrón.

Tabla 5.4 Descripción de las variables involucradas en el archivo con sufijo _v.dat.

V. 1. 5. Escritura de pesos y umbrales a un Archivo.

Se cuenta también con un procedimiento llamado **wtwrite** que crea un archivo tipo texto con extensión _w.dat para salvar pesos y umbrales aprendidos durante el entrenamiento, pasándole como parámetro el nombre del archivo igual que en el procedimiento descrito anteriormente.

Este archivo será abierto para lectura cuando el usuario decida utilizar la matriz de pesos a la que se llegó en un determinado aprendizaje anterior. El formato del archivo se describirá a continuación:

nombre variable	significado de la variable
wtptr[i]+j donde: i indica el número del patrón j indica el número de la interconexión del peso con la capa subsecuente	Valor de cada peso de interconexión para cada nodo de cada patrón.

Tabla 5.5 Descripción de las variables involucradas en la escritura del archivo con sufijo _w.dat.

Al igual que existen 2 procedimientos de escritura de archivos con los parámetros alcanzados en la etapa de aprendizaje, se cuenta con el mismo número de procedimientos para leer dichos archivos en caso de ser necesario.

V. 1. 6. Lectura de los parámetros de Aprendizaje.

El procedimiento **dread** recibe el nombre del archivo a ser leído, el cual contiene los datos que establecieron la Arquitectura de la Red así como los parámetros del aprendizaje, el archivo tiene extensión _v.dat. Los datos leídos por el procedimiento son los establecidos en la Tabla 5.4

V. 1. 7. Lectura de pesos y umbrales.

Ahora bien , el procedimiento **wtread** el cual también recibe el nombre del archivo a ser leído , leerá los pesos a los que se llegó al final del aprendizaje.Es pertinente mencionar que este archivo de lectura tiene extensión _w.dat. Los datos leídos por el procedimiento son los establecidos en la Tabla 5.5 .

V. 1. 8. Reconocimiento.

Si la fase de Reconocimiento es seleccionada se proporciona el nombre del archivo con el cual se trabajó en el aprendizaje, preguntándose si se quiere cambiar a otro archivo que se haya utilizado en otro aprendizaje con el objetivo de poder elegir la forma de onda que se ajuste más al evento a ser reconocido, facilitándole a la Red el llegar a un óptimo reconocimiento

Si se quiere cambiar de archivo se procede a pedir el nombre de éste, llamando al procedimiento de lectura **dread** antes descrito, entonces se hace referencia al procedimiento **init** donde se inicializan y dimensionan los apuntadores y estructuras a ser utilizadas.

Subsecuentemente se llama al procedimiento **wtread** para asignar los valores de pesos correspondientes al archivo que fué seleccionado dentro de la estructura asignada para tal labor.

Entonces se introduce el nombre del archivo procesado para la etapa de Reconocimiento(el cual a diferencia del archivo prepocesado para la etapa de aprendizaje, no tiene valores de salida que asocien a cada patrón, es decir, la señal no está categorizada).

Como este archivo tiene un formato predefinido (el cual ya fué mencionado) entonces se extraen las características importantes grabadas en el encabezado, como lo son el número de patrones a ser procesados, el número de muestras por patrón, el patrón en el que se detectó el arribo al momento de ser procesado, el nombre de la traza, y los datos acerca de la hora, minuto y segundo en el que se grabó el evento sísmico

Con estos datos extraídos se procede a leer para cada patrón los valores grabados en el archivo, entrando a un ciclo que irá desde 0 hasta el número de patrones en donde se llamará al procedimiento **forward** imprimiendo el valor de salida calculado por la red para cada patrón, pudiéndose del cambio que van experimentando las salidas calculadas con respecto a las esperadas

El valor que se estableció para los patrones anteriores al arribo de la fase "P" fué de 0.1 y el valor establecido para los patrones donde se encuentra el arribo así como los posteriores a él es de 0.9 , por lo tanto se consideró que un valor mayor a 0.7 significaría un buen reconocimiento del arribo de la fase.

Debido a lo anterior, al momento de terminar la presentación en pantalla de los datos de salida tanto calculados por el algoritmo como los esperados se estableció la siguiente condición:

Si la salida calculada es mayor ó igual a 0.7 se guarda en una variable el número del patrón en el que se cumplió tal condición Si los patrones fueron desplazados en el preprocesamiento se divide el número de muestras desplazadas entre el número de muestras por patrón para tener la proporción de muestras que variará de un patrón a otro, teniendo este dato y conociendo la fracción de tiempo que dura cada patrón se procede a calcular el tiempo en el que ocurrió el arribo del evento sísmico tratado.

Este tiempo es sumado al segundo de inicio de grabación del evento para tener el segundo absoluto en el que el arribo de la fase P fué presentado. De esta manera el sistema de la Red muestra al finalizar la etapa de Reconocimiento la hora, minuto y segundo en el que ocurrió el arribo, así como el tiempo en el que se empezó a grabar el evento, y el patrón el que se asoció el arribo.

Para verificar si el arribo fué reconocido realmente se procedió a diseñar un algoritmo dentro del sistema de preprocesamiento que nos indica en que patrón se presentó, este algoritmo va comparando el patrón que se va procesando con el anterior y cuando encuentra que el promedio de los datos contenidos en el patrón es mayor ó igual al doble del patrón anterior se determina que la causa de esa variación tan marcada lo es la presentación del arribo de la fase "P"

Dicho valor que indica en que patron se asoció el arribo se graba en el archivo preprocesado para la fase de reconocimiento , entonces este valor es comparado con el patrón que asocia la Red y en base a esto se asegura la eficiencia del algoritmo para reconocer ese evento sísmico en específico

Ahora bien, si los patrones no han sido desplazados entonces el tiempo del arribo de la fase P que proporciona la Red puede no ser tan exacto.

Por esta razón se decidió tomar el desplazamiento de los patrones dentro del preprocesamiento normal de los datos En la etapa de discusión de resultados se explicará más profundamente los conceptos antes citados.

Finalmente, gracias a la facilidad otorgada por la interface gráfica se está en posibilidad de repetir el proceso cuantas veces se desee

V. 2. Descripción de la Interface gráfica del Sistema de la Red Neuronal.

El sistema de la Red Neuronal presenta en la Barra de Titulo el siguiente mensaje:
"Identificación de la Fase Sísmica P (Versión 1.0)". Cuenta con 4 menus disponibles en la Barra de Menus.

Es pertinente mencionar que cada vez que se selecciona alguno de los botones de la barra de menus, aparece una indicación del proceso que se está llevando acabo, en la parte inferior izquierda dentro de la sección de mensajes.

Ahora bien, la pantalla principal del Sistema de la Red Neuronal es mostrada en la figura 5.6

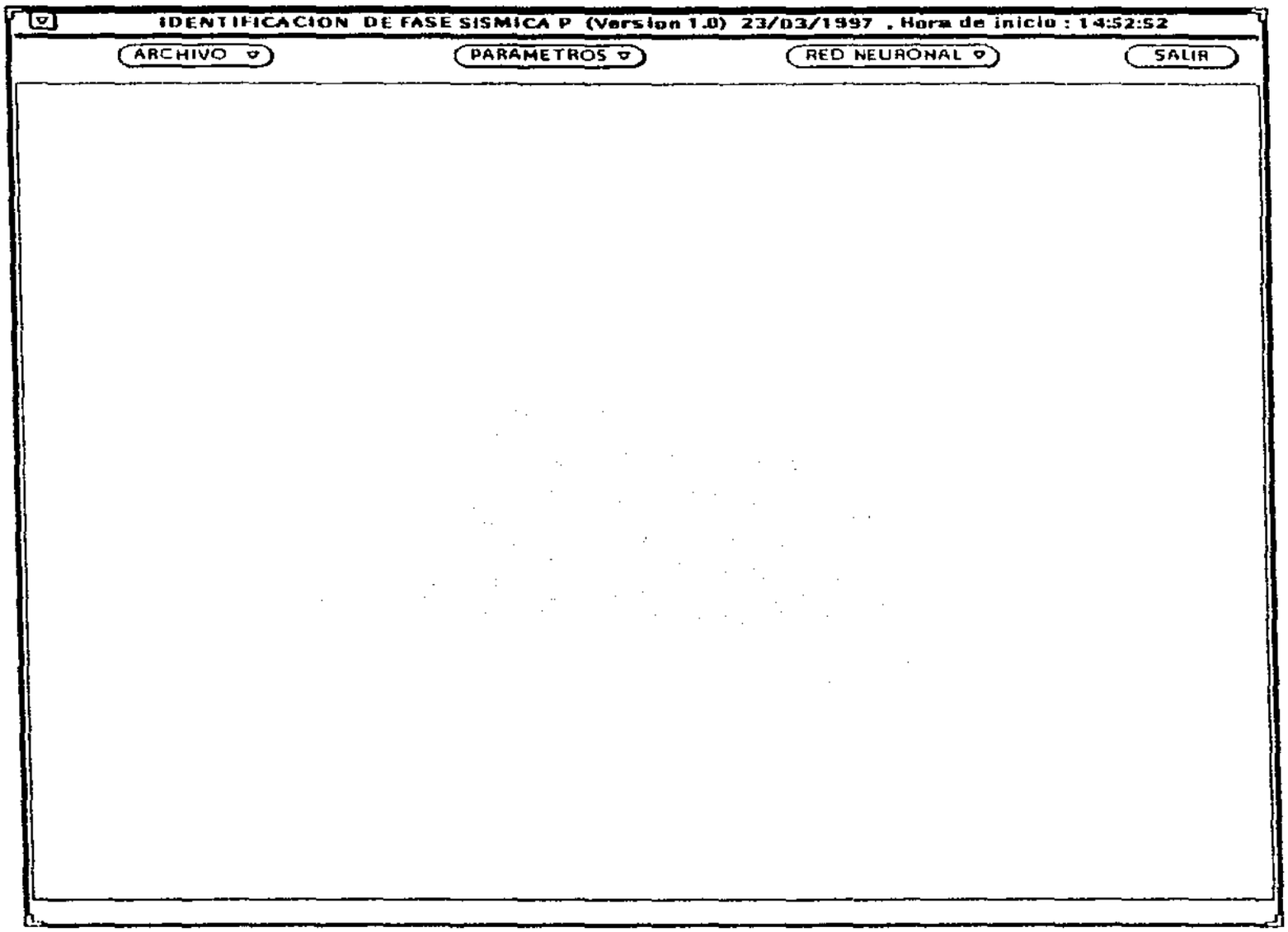


Figura 5.6 Pantalla principal del Sistema de la Red Neuronal.

A continuación se mencionarán los menus declarados en el sistema:

Archivo: En esta parte se muestra un menu descendente con 3 comandos los cuales son:

Abrir - Se presenta una subventana en donde se pide el nombre del archivo a ser procesado . En la parte inferior se encuentra un botón denominado Cerrar el cual una vez tecleado el nombre , finalizará con la apertura del archivo .El procedimiento al que se hace referencia se denomina *pide archivo*.

En la figura 5.7 se muestra el subframe para la petición del nombre de archivo a ser abierto . Cabe hacer notar que el nombre del archivo en esta fase de apertura debe introducirse sin la extensión ".dat".

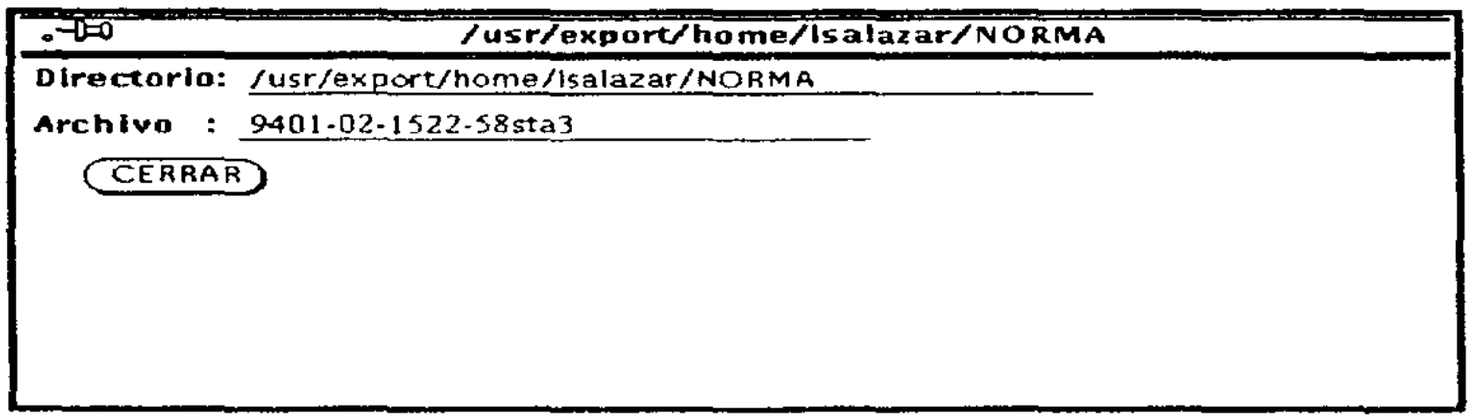


Figura 5.7 Subframe donde se asigna el nombre del archivo a ser abierto.

Borrar -Accesa un proceso que borra la pantalla.

Salir - Indica el fin de sesión. Se manda un mensaje que demanda al usuario si quiere realmente salir de la aplicación, presentándose dos botones con las posibles respuestas,**Cancela** ó **Confirma**,si la respuesta es negativa el comando es cancelado y se retorna a la pantalla principal, en caso contrario, se sale completamente de la aplicación. El procedimiento al que se hace referencia se denomina *salir*.

Parámetros.- Se muestra también un menu descendente con 3 comandos los cuales serán descritos a continuación:

Arquitectura de la Red Neuronal -Se inicia la petición al usuario de los parámetros requeridos por el sistema con el objetivo de definir la Arquitectura de la Red. Esta petición se realiza mediante cuadros de diálogo

La petición inicia con el procedimiento denominado *parámetros*,en donde después de declarar los objetos de OPENWINDOWS correspondientes se procede a seleccionar la **Razón de aprendizaje** (designada como eta con valor por default=0.9), este parámetro determinará el nivel de aprendizaje de la Red Neuronal,es decir,mediante dicho valor la Red estará en posibilidad de aprender rápidamente , lentamente ó simplemente no asimilar ni asociar los patrones de entrada presentados con su correspondiente valor de salida. Al finalizar cada petición como en el sistema de preprocesamiento se mostrará en la parte inferior de la subventana un botón , el cual al ser presionado asignará los valores proporcionados a sus correspondientes variables para ser utilizadas dentro del proceso posteriormente.

Se procede entonces a seleccionar la **Razón de momentum** (alpha con valor default= 0.7,este parámetro permite estabilizar el criterio de aceptación en cada una de las etapas Después aparece la petición del **Error máximo total** (default=0.01) , este parámetro indica el umbral ó límite al cual debe de llegar el algoritmo al realizar las iteraciones necesarias tanto con los datos del archivo de entrada como con los proporcionados por el usuario.

Asi mismo se pide el **Error máximo individual**(default=0.001), este parámetro fungirá también como límite para indicarle al sistema si debe o no continuar con los cálculos correspondientes al patrón presentado, es decir, si después de las iteraciones realizadas con un cierto patrón para obtener a la salida el valor deseado el error es más pequeño ó igual que el dato proporcionado por el usuario, el algoritmo asumirá que se ha llegado a un valor óptimo y asociará dicho patrón de entrada con la matriz de pesos que se obtuvo, pasando al aprendizaje del siguiente patrón y así sucesivamente.

El parámetro siguiente es el **Número máximo de iteraciones(default=1000)**, este valor también servirá como límite, indicándole al sistema cuando debe dejar de realizar los cálculos pertinentes,pues bien puede suceder que el número de iteraciones alcanze el número máximo proporcionado por el usuario terminando de esta manera la etapa de aprendizaje y que el sistema no haya asociado correctamente los patrones presentados.

Esto significa que la red no tuvo la capacidad de aprender, por lo que no asoció al patrón de una manera adecuada, pudiendo deberse a una mala selección de los parámetros que determinan la Arquitectura de la Red aunado a los datos de la señal a ser aprendida. Los valores seleccionados no fueron los óptimos para los datos que constituyen a la señal. Los valores óptimos serán descritos en el capítulo VI

El último parámetro importante, es el **Número de capas ocultas** con que contará la Red. Este valor ayuda a definir el número de relaciones entre una capa u otra, este valor se determinó como 1 en base a las pruebas realizadas.

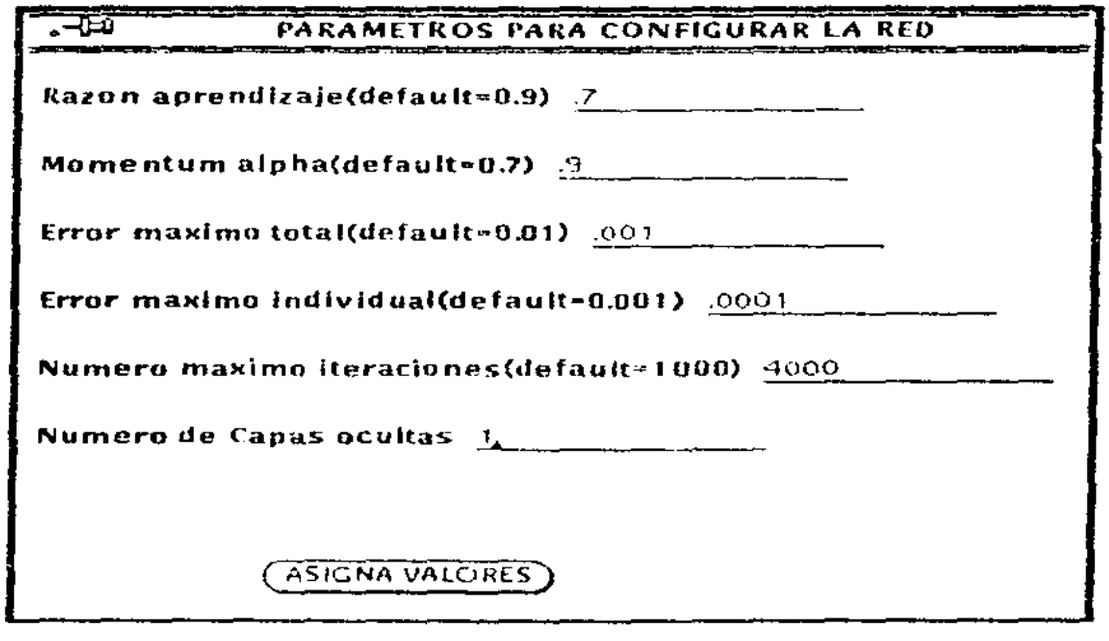


Figura 5.8 Subframe donde se asignan los valores para definir la arquitectura de la Red Neuronal.

Posteriormente aparece una ventana donde se muestran los datos del encabezado del archivo de entrada a la Red tal como se observa en la figura 5.9

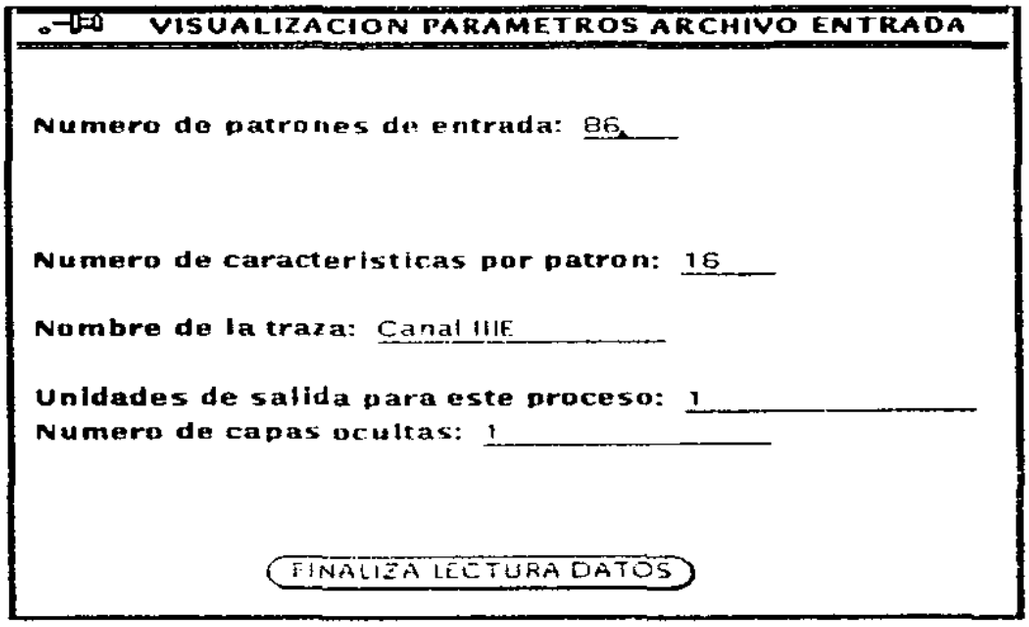


Figura 5.9 Se muestran los datos del encabezado del archivo de entrada a la Red.

Desde la segunda vez que se lleva acabo la fase de aprendizaje, una ventana demandando al usuario si desea utilizar la matriz de pesos a la que se llegó en el anterior aprendizaje se presenta, si la respuesta es afirmativa la matriz de pesos es inicializada con esos valores en lugar de iniciarla con valores aleatorios.

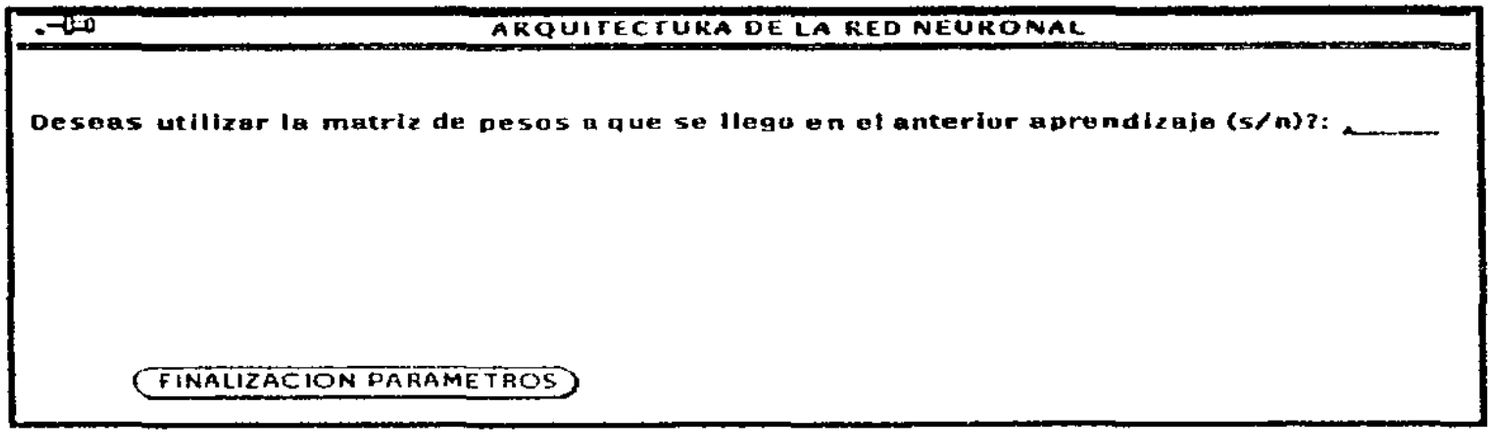


Figura 5.10 Se demanda al usuario si desea utilizar la matriz de pesos del aprendizaje anterior.

A continuación se muestra la figura 5.11 con las instrucciones precisas a realizar para continuar con el proceso.

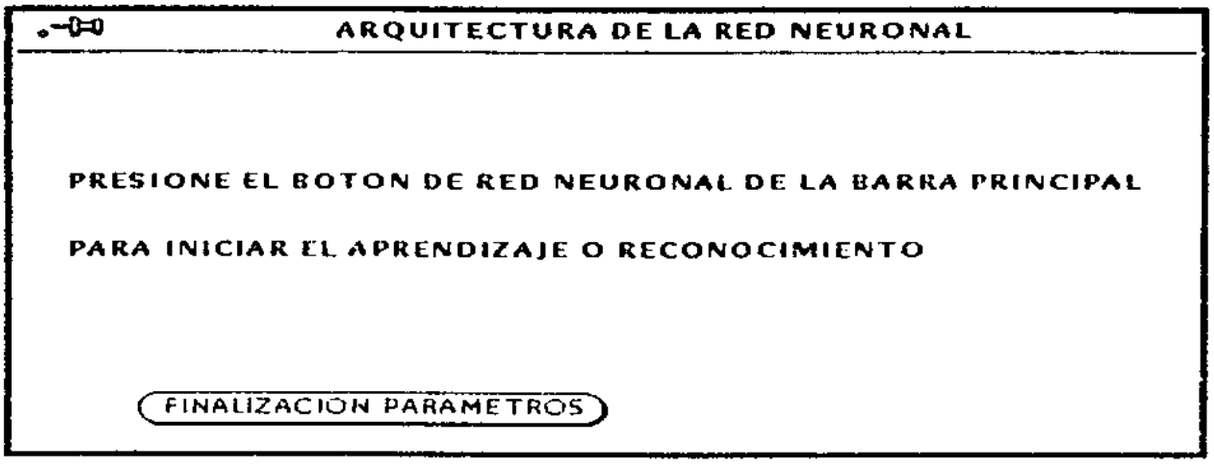


Figura 5.11 Se le indica al usuario cual es el siguiente paso a realizar dentro del proceso.

Restricciones sobre el sistema -Se hace referencia al procedimiento llamado *prepara canvas* , el cual como se ha mencionado habilita a la sección de graficación (ó de presentación de la información) , para imprimir la información necesaria que mantendrá informado al usuario sobre el proceso que está realizando..

Red Neuronal.- En esta etapa el menu contiene 2 comandos mencionados a continuación:

Aprendizaje.- El procedimiento al que se hace referencia es llamado aprende
En este procedimiento el usuario decide utilizar la matriz de pesos obtenida en el aprendizaje anterior, para a partir de esos valores ejecutar nuevamente la etapa de aprendizaje aplicada a un nuevo evento sísmico . Esto se lleva acabo con la finalidad de hacer más eficiente el sistema ahorrando tiempo de ejecución así como recursos, por otra parte se pretende lograr un mejor reconocimiento basándose a su vez en un buen aprendizaje.

Cuando se presiona el botón correspondiente al Aprendizaje ubicado en el comando designado como Red Neuronal dentro de la barra de menus, el algortimo de Retropropagación se empieza a llevar acabo, realizándose los cálculos pertinentes Cuando el aprendizaje ha finalizado aparece el mensaje que le indicará al usuario el resultado del aprendizaje. El contenido de dicho mensaje puede ser: "Aprendizaje éxitoso" ó "Aprendizaje no éxitoso". Si el aprendizaje no fué éxitoso entonces, dentro del mensaje aparece la sugerencia de cambiar algunos de los parámetros que configuran la Arquitectura de la Red, pretendiendo con ésto obtener un buen aprendizaje. En base a pruebas se puede conocer qué parámetros pueden ayudar para lograrlo. Ahora bien, si se sobrepasaron las iteraciones estipuladas dentro de los parámetros proporcionados antes de llegar a un aprendizaje optimo, se le indicará al usuario.

Para iniciar la fase de Reconocimiento es necesario haber logrado un aprendizaje éxitoso, de esta manera la Red es más capaz de realizar una buena asociación y clasificación de los patrones que constituyen la traza a ser reconocida que si no hubiera tenido éxito en la fase de entrenamiento. Es importante mencionar que un aprendizaje éxitoso tampoco implica que cualquier traza que se presente va a ser reconocida por la traza entrenada, existe un rango de riesgo

Reconocimiento.-Se muestra el nombre del archivo con el que se estaba trabajando en la etapa de aprendizaje preguntando si se desea cambiar de archivo, si la respuesta es negativa como se muestra en la figura 5.12.

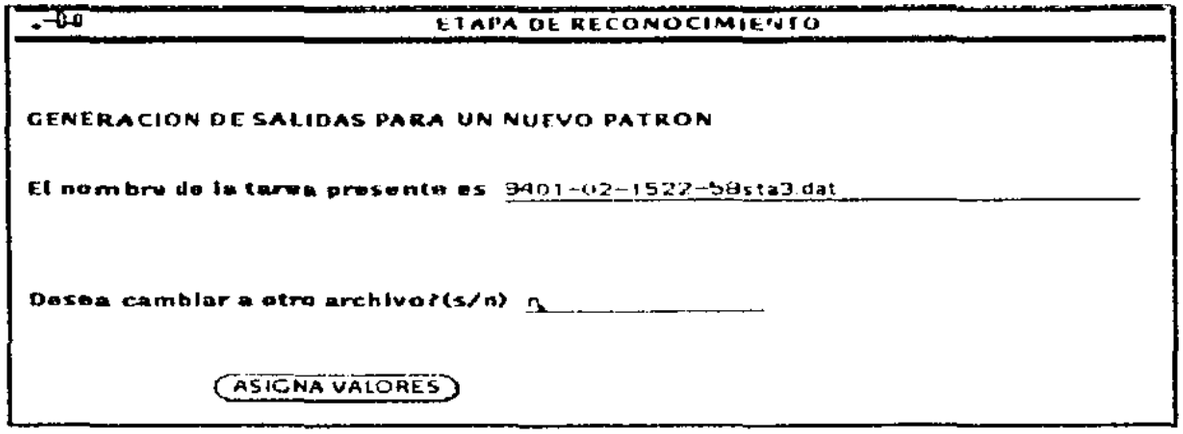


Figura 5.12 Subframe donde se establece con qué traza se procederá a reconocer .

Aparecerá entonces otra ventana donde se pedirá el nombre del archivo para la fase de reconocimiento tal y como lo muestra la figura 5.13. Esto indica que la traza será reconocida mediante el la traza utilizada en la etapa de entrenamiento.

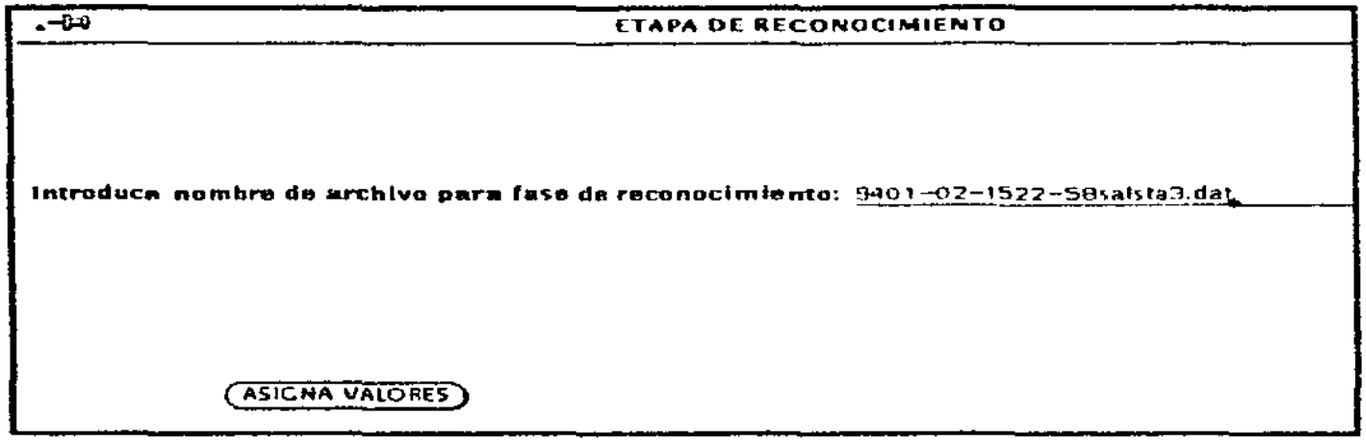


Figura 5.13 Se proprociona el nombre del archivo para la fase de reconocimiento.

En cambio si la respuesta es positiva, entonces se introduce el nombre tanto del archivo utilizado en la fase de entrenamiento(siendo importante mencionar que el archivo tuvo que haber sido presentado a la Red en la etapa de aprendizaje puesto que se utilizará la matriz de pesos que se obtuvo) como el que se quiere sea reconocido tal y como se ejemplifica en la figura 5.14.

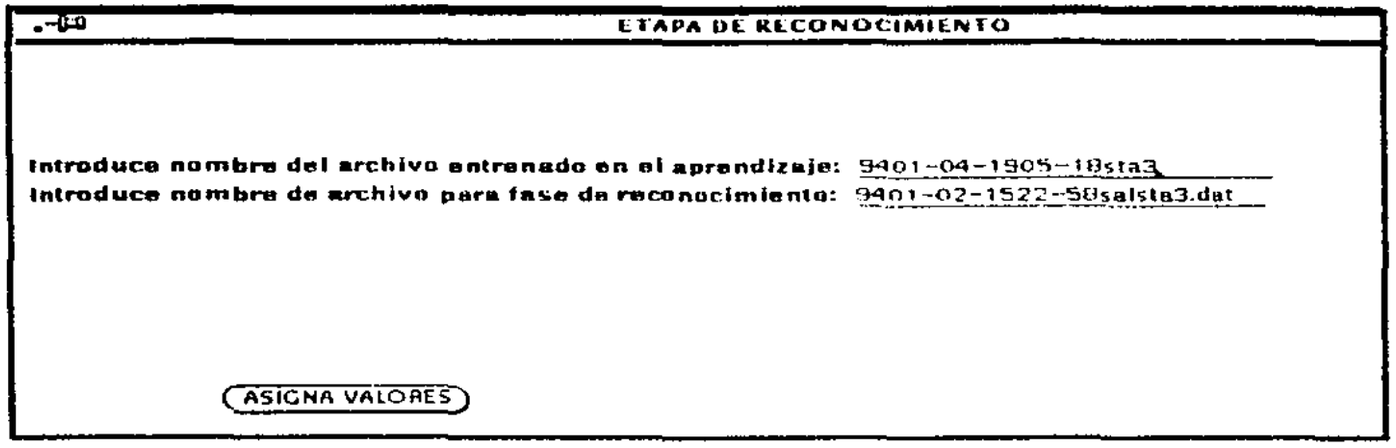


Figura 5.14 Se proprociona el nombre del archivo para la fase de reconocimiento, así como el nombre del archivo que se utilizó en la fase de entrenamiento con el que se procederá a reconocerlo.

Teniendo el nombre del archivo a reconocer, el algoritmo lee entonces el encabezado de dicho archivo para extraer el número de patrones que serán procesados, así como el número de muestras por patrón.

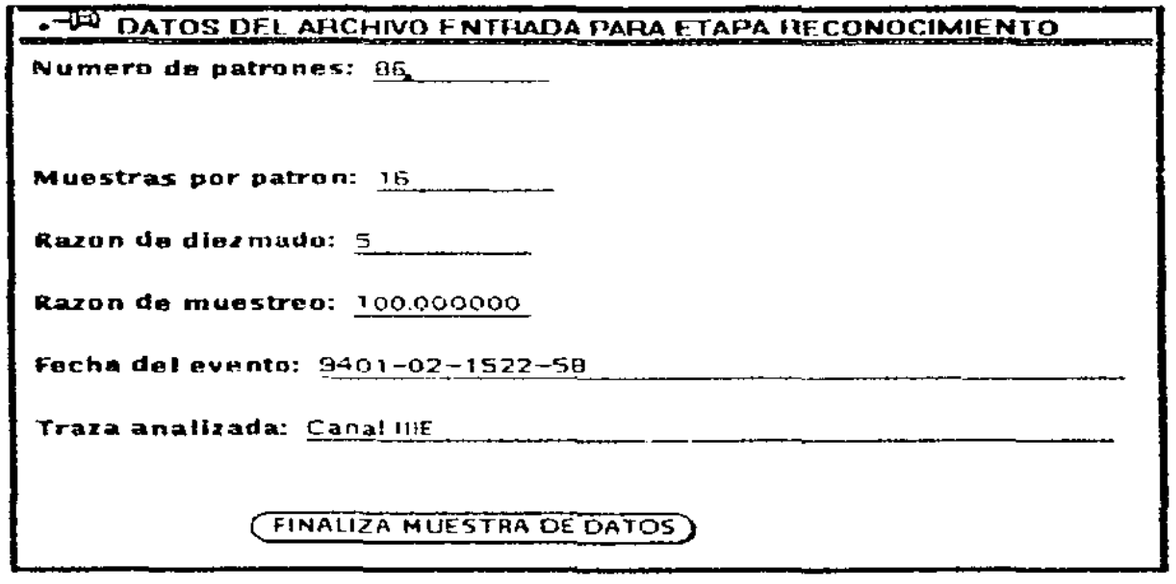


Figura 5.15 Datos del encabezado del archivo utilizado en el reconocimiento.

Salir.- Este es solo un comando no propiamente un menu, siendo igual al mencionado en el menu ***Archivo***.

VI . PRUEBAS, DISCUSION DE RESULTADOS Y EVALUACION DEL SISTEMA

El nivel de la calidad de un sistema está en función del diseño, desarrollo, pruebas e implantación del mismo. Un aspecto de la calidad lo determina la confiabilidad. Ahora bien, las pruebas constituyen una parte fundamental dentro de la evaluación del sistema para determinar el nivel de calidad alcanzado.

El objetivo de las pruebas es garantizar que el funcionamiento del sistema se realize de forma adecuada, para lo cual se pretende encontrar errores al momento de su ejecución, evitando que al liberarse se produzcan fallas peligrosas y costosas.

De esta manera, la evaluación permite determinar si el sistema cumple con las expectativas impuestas al momento de diseñarlo, conociendo sus limitantes Analizando también donde serán necesarias las mejoras y de qué manera se pueden anexar en un futuro.

VI. 1. Pruebas de Integración

En primer lugar se realizaron pruebas modulares aplicando la filosofía de integración ascendente, en donde cada módulo era probado por separado(individualmente), verificando su adecuado funcionamiento con la finalidad de irse integrando al sistema, llevando acabo también pruebas en el momento de la integración para asegurar a su vez que ese módulo no interferiría con los ya instalados. De esta manera se iba avanzando en el desarrollo del diseño, logrando tener al final un sistema constituido por procesos independientes pero a la vez complementarios

Dentro del preprocesado de los eventos sísmicos se tienen 3 algoritmos que realizan el preprocesado de las señales. En base a los resultados generados mediante las pruebas realizadas se determinaron los factores que influirán para utilizar cada uno de los algoritmos citados anteriormente, los cuales serán citados en la sección V.1, pretendiendo con esto obtener a la salida del sistema una señal donde el arribo de la fase "P" sea fácilmente reconocido por el analista experto, dando como consecuencia que la Red Neuronal pueda aprenderlo y reconocerlo asociando los patrones procesados.

Para tener una idea de la transformación que iba experimentando la señal (debida a los procesos aplicados), al final de cada módulo los datos fueron grabados dentro de un archivo tipo texto para ser graficados, con el objeto de verificar la salida y determinar si el proceso iba contribuyendo a la buena depuración del arribo de la fase "P"

En base a la secuencia establecida en cuanto al procesamiento aplicado a las señales, las gráficas fueron una base importante para determinar tanto los procesos que contribuían a que el arribo de la fase continuara siendo claro, así como los procesos que eran necesario suprimir. Desarrollando en su lugar otro tipo de proceso que lograra la identificación cuantas veces fuera necesario.

Para establecer los valores óptimos dentro de cada uno de los procedimientos del preprocesado al igual que con los procedimientos que consituyen el algoritmo de la red se realizaron varias pruebas, hasta establecer cuales eran los valores óptimos. De esta manera se le proporciona al usuario final más de una opción para procesar, dependiendo por supuesto del tipo de señal

En la primera fase del desarrollo del algoritmo de la Red se implantó la parte de la propagación hacia adelante así como la propagación hacia atrás, es decir , la base funcional en que se sustenta el

algoritmo de Retropropagación , posteriormente se pensó en las condiciones que debía de cumplir el sistema para lograr los requerimientos establecidos en la etapa de análisis, ajustándolo.

Como ya se mencionó el proceso se realizó modularmente, por lo tanto si surgía alguna otra idea para lograr una buena ejecución del sistema, se diseñaba en paralelo al desarrollo del mismo , robusteciéndolo. Esto se lleva acabo de esta manera puesto que al momento de su concepción se tiene una idea base que será enriquecida durante la fase de desarrollo.

Al finalizar la etapa de desarrollo como la de diseño de los procedimientos que constituyen al Sistema del Preprocesado tanto como al de la Red Neuronal, se contaba ya con los tipos de proceso que se necesitaban para enfrentar el problema propuesto.

Ahora bien, una vez construida la Red y el algoritmo de Preprocesado, lo que se hace es entrenarla, siendo necesario contar con una serie de registros que tengan una ó varias trazas en donde el arribo de la fase "P" sea fácil de identificar para el propio analista, ya que esto dará la pauta para asegurar que el Aprendizaje así como el Reconocimiento trabajaran más eficientemente que con un registro donde el arribo sea confuso hasta para el mismo experto

Para familiarizarse con las formas de onda de cada uno de los 14 sismos utilizados en la fase de entrenamiento y reconocimiento es importante mencionarlos:

Sismos utilizados en la fase de entrenamiento y reconocimiento
9401-02-1522-58S.SMX_16
9401-04-1905-18S.SMX_16
9401-05-1853-31S.SMX_16
9401-12-0246-41S.SMX_16
9402-19-2000-01S.SMX_16
9402-27-0407-33S.SMX_16
9403-03-1420-20S.SMX_16
9403-19-0421-44S.SMX_16
9403-20-0506-25S.SMX_16
9403-25-1633-04S.SMX_16
9403-25-1637-18S.SMX_16
9409-17-1916-58S.SMX_16
9410-04-2002-56S.SMX_16
9412-25-0409-28S.SMX_16

Tabla 6.1. Sismos utilizados en la fase de entrenamiento y reconocimiento.

Pero de aquí en adelante, el buen desempeño de ellos dependerá exclusivamente de la selección de los parámetros que determinan tanto la arquitectura de la Red como la depuración del evento sísmico. Estos parámetros se fueron determinando, a modo de conocer el comportamiento que cada uno tendría en la fase de entrenamiento como en la de reconocimiento. Los parámetros que determinarán la buena depuración del evento sísmico son.

a) Número de muestras por patrón.

b) Razón de diezmado.
c) Tipo de proceso.

Por lo tanto es muy importante seleccionar los parámetros adecuados para el tipo de señal que se esté analizando. Cabe mencionar que para realizar las pruebas pertinentes y asegurar el buen funcionamiento de los sistemas involucrados, se pensaba que el valor ideal para generar los patrones debia ser de 256 muestras, el cual fué variandose hasta determinar que el valor óptimo debía ser de 16 muestras.

VI. 2. Pruebas de Operatividad.

V. 2. 1. Número de muestras por patrón.

Las pruebas realizadas con patrones de 256 muestras no dieron buenos resultados, la fase de aprendizaje tardaba varios minutos y no era exitosa , es decir, la Red no asociaba los valores de cada patrón con el valor de salida deseado. Pero, ¿qué significaban estas primeras pruebas dentro de la evaluación del sistema?, **bastante**, puesto que dieron la pauta para analizar qué tipo de parámetros serian los convenientes para lograr que la Red otorgará buenos resultados.

Por ejemplo, analizando que: La razón de muestreo con que se cuenta es de 100 muestras/segundo y seleccionando como razón de diezmado 1 muestra de cada 5, se tendrán 20 muestras por segundo, por lo tanto si se generan patrones de 256 muestras se está tomando 12.8 segundos de la señal por cada patrón. Esta es una razón importante por la cual la fase de aprendizaje así como la de reconocimiento no tuvieron éxito, 12.8 segundos de defasamiento en tiempo entre un patrón y el siguiente representan variaciones muy grandes en una señal sísmica.

Observando que si el arribo de la fase "P" se encontrara entre dos patrones, al momento de su reconocimiento existiría una diferencia de por lo menos 12.8 segundos del tiempo real en el que ocurre el arribo. Partiendo de este hecho, se prosiguió seleccionando el número de muestras en orden descendente hasta encontrar el valor óptimo. Cabe mencionar que desde 64 muestras descendiendo hasta 16, se obtiene un buen aprendizaje en los sismos involucrados en la fase de entrenamiento, el problema a enfrentar es que el Reconocimiento no se alcanzó por parte de 6 de los sismos obteniéndose una eficiencia de la Red del 57.14%. Los sismos no reconocidos con ventanas de 64 muestras fueron:

Sismos no reconocidos en la fase de reconocimiento con 64 muestras
9401-12-0246-41S.SMX_16
9402-27-0407-33S.SMX_16
9403-03-1420-20S.SMX_16
9409-17-1916-58S.SMX_16
9410-04-2002-56S.SMX_16
9412-25-0409-28S.SMX_16

Tabla 6.2. Tabla que muestra los sismos no reconocidos en la fase de reconocimiento utilizando 64 muestras.

Se procedió a investigar el comportamiento de la Red ante patrones constituidos con 32 muestras, encontrándose algo muy interesante, los eventos que son procesados con patrones de 16 que no tienen éxito en la fase de aprendizaje son rescatables procesándolos con 32 muestras, siempre y cuando no se haya tenido éxito variando los parámetros que definen la arquitectura de la Red : Razón de aprendizaje y Razón de momentum (explicados en la sección V.3).

La única restricción que se encontró al respecto fué que, cuando se trata de reconocer un sismo aprendido con patrones de diferente número de muestras que el utilizado en la fase de reconocimiento, la Red no está capacitada para asociar dichos patrones y obtener un reconocimiento óptimo.

Se encontró que 2 de los 14 sismos no tuvieron éxtio en la fase de aprendizaje al ser procesados con 16 muestras.

Sismos no reconocidos en la fase de aprendizaje con 16 muestras.
9409-17-1916-58S.SMX_16
9412-25-0409-28S.SMX_16

Tabla 6.3 Sismos no reconocidos en la fase de aprendizaje con 16 muestras.

De acuerdo a las observaciones realizadas, la causa que originó esto fué que, en promedio los valores de amplitud de la señal no varian rápidamente después de encontrar el arribo de la fase "P" , por lo cual se necesita tomar un patrón con más número de muestras para asociar a ese patrón con el arribo, de otra manera si se toman patrones con un número menor de muestras no se estará en posibilidad de discernir entre ese patrón y el subsecuente para asociar la llegada del arribo, lo mismo sucede con los sismos que en la fase de reconocimiento dan el arribo calculado en el patrón siguiente a donde ocurre realmente.

Otra causa que puede provocar que la Red reconozca el arribo en el patrón siguiente al real es que, los valores de amplitud de la señal después del arribo propasan en mucho los valores del patrón anterior donde se encontró el arribo.

Es importante tener una señal constante antes del arribo de la fase "P" para que al momento de ser detectado, el algoritmo esté en posibilidad de clasificarlo correctamente, debido a la diferencia en valores de amplitud antes y después de éste, por lo tanto, otro factor que influye directamente en el Reconocimiento es: la variación entre los valores de amplitud que se encuentren después del arribo, ya sea que el promedio de éstos se incremente ó decremente, pretendiendo con lo anterior que los valores asociados por la Red no sean tan constantes en valor para facilitar el aprendizaje .

Si una señal no cuenta con las caracteristicas citadas anteriormente, puede provocar que la Red se confunda al asociar los patrones al momento del Reconocimiento y lo que sería peor, que reconozca el arribo en un patrón erróneo.

Ahora bien, con los resultados obtenidos hasta el momento, se procedió a entrenar al grupo de registros con patrones de 16 muestras, encontrando también que la razón de diezmado, así como el tipo de proceso son fundamentales dentro del procesamiento de los registros para lograr un buen desempeño de la Red.

VI. 2. 2. Razón de diezmado.

Se determinaron dos razones de diezmado: 1 de cada 2 muestras y 1 de cada 5 muestras, encontrando que esta última era la óptima tanto en tiempo de ejecución como en suavizado de la señal. Por otra parte, la razón de diezmado 1 de cada 2 muestras provocaba tener más datos a ser procesados, redundando invariablemente en el tiempo de ejecución del sistema.

VI. 2.3. Tipo de Proceso.

Al haber seleccionado correctamente los dos parámetros antes mencionados se está en el 50% de avance en el camino de conseguir un buen procesado para el registro, la otra parte es seleccionar adecuadamente el tipo de proceso a aplicar.

El tipo de proceso adecuado para cada señal depende precisamente de la forma de onda. Con las primeras pruebas realizadas se determinó que el proceso óptimo era el STA de 20 muestras, obteniéndose las bases para conseguir un reconocimiento satisfactorio.

Después de analizar los sismos procesados con STA de 20 que no eran reconocidos por la Red, se observó entonces la forma de onda de las señales expuestas, deduciendo que la manera en la que llega el arribo es importante para determinar el proceso a ser aplicado Con respecto a este punto cabe mencionar que el arribo de la fase puede ser Emergente ó Impulsivo.

Es decir, Emergente se refiere a la forma de onda que llega muy suavemente, por lo cual no contiene tanta energía, la frecuencia en este tipo de ondas es mucho menor que la que tiene un arribo Impulsivo, considerándose Impulsivo la forma de onda que llega con más fuerza

Cuando el arribo es Emergente el tipo de proceso aplicado generalmente debe ser STA de 20 muestras, esto es porque la variación entre el nivel de la señal antes del arribo y después de él no varía tan notablemente, es decir, los valores permanecen en promedio constantes. Por lo anterior, es necesario tomar patrones de 20 muestras que tomarán más valores de la señal y así encontrar de una manera más fidedigna el arribo.

Pero, siendo Emergente también se puede utilizar el proceso STA de 5 siempre y cuando la variación entre los valores de amplitud después del arribo no sean tan constantes, en caso contrario se aplicará el STA de 20 muestras como se había ya establecido anteriormente.

En cambio cuando el arribo es Impulsivo, como los valores son contrastantes en diferencia al nivel de la señal antes del arribo es necesario tomar ventanas de 5 muestras de la señal.

Ahora bien, si el nivel de la señal antes del arribo no fuera muy constante, es decir, que se haya mezclado con la señal una pequeña interferencia, y el arribo es muy Emergente se puede confundir el momento del arribo , esto puede suceder cuando se selecciona que el algoritmo detecte el arribo de la fase.

Para ejemplificar parte de lo expuesto anteriormente, se anexarán al final de este capítulo las gráficas correspondientes a 3 sismos pertenecientes al grupo de estudio seleccionado, esto con el objeto de mostrar la diferencia entre la señal original y la señal depurada que entrará a la Red para ser procesada. De esta manera, se tendrá una idea más clara de la importancia que juega el tipo de proceso dentro del desempeño de la Red Neuronal.

Ahora bien, es importante especificar que un factor que se cumple para todos los sismos analizados es que pueden ser reconocidos por ellos mismos.

VI. 3. Calibración de la Red.

La etapa más sensible del entrenamiento es la calibración de la Red, entendiéndose como calibración al proceso de encontrar las constantes que mejor se ajusten para la identificación del patrón. Este proceso se realiza mediante una gran cantidad de ensayos con los patrones a identificar y variando poco a poco cada uno de los parámetros de la red.

Los parámetros que definen la Arquitectura de la Red son:

Razón de aprendizaje(valor por default=0.9, valor óptimo=0.7) designada como eta.
Momentum(valor por default=0.7,valor óptimo=0.9) designado como alpha.
Error máximo total(default=0.01,valor óptimo=0.001) designado como maxe.
Error máximo individual(default=0.001,valor óptimo=0.0001) designado como maxep.
Número máximo de iteraciones(default=1000,valor óptimo=4000) designado como cnt_num.
Número de Capas ocultas (valor óptimo=1) designado como nhlayer.

Los errores al igual que el número maximo de iteraciones tienen la función de controlar en qué momento se debe parar el proceso de aprendizaje, como se mencionó en la sección IV.6.

El parámetro de Razón de aprendizaje determina la variación con la que los patrones serán procesados por la Red, este valor debe ser positivo y menor que 1. Usualmente se determina este valor entre 0.05 y 0.25, pero en este caso , en base a las pruebas realizadas el valor óptimo es de 0.7.

Un valor pequeño asignado a la razón de aprendizaje significa que la Red tendrá que realizar un mayor número de iteraciones. Si este se incrementa,el error decrecerá ayudando a que la velocidad de convergencia se incremente, haciéndose por supuesto más rápido el proceso de aprendizaje. Pero si la razón de aprendizaje fuera demasiado grande el sistema no convergería, dando como resultado que la Red no tenga éxito en el proceso de aprendizaje.

Así como un valor bien seleccionado para la razón de aprendizaje incrementa la velocidad de convergencia, también lo logra el término llamado momentum (mencionado en secciones anteriores).
Cuando se calcula el cambio en los pesos, se está adicionando una proporción del cambio anterior, como se muestra en la ecuación (3.7) , este término hace que el cambio en los pesos vaya siempre en la misma dirección.

En base a las pruebas realizadas se decidió que el valor óptimo para el número de capas ocultas debía ser de 1, con 8 nodos en la capa, es decir, la mitad de nodos que constituyen la capa de entrada. La arquitectura de la Red no es complicada para este caso especificamente, ya que se tienen 16 nodos en la capa de entrada, 8 en la capa oculta y 1 nodo en la capa de salida; se establecieron 16 nodos en la capa de entrada porque se apreció que el valor óptimo para este tipo de proceso era de 16 muestras por patrón lo que se discutió en la sección anterior. Ahora se tiene un sólo nodo en la capa de salida porque sólo se quiere reconocer un valor , en este caso la fase "P".

El proceso de calibración de la red es muy delicado, y requiere de mucho cuidado porque cualquier variación sin control puede ser causante de una identificación errónea en la red.

Debido a lo anterior, se realizaron pruebas exhaustivas en donde se variaron los parámetros de Razón de aprendizaje y momentum, para encontrar los valores en donde el grupo de entrenamiento consiguiera los mejores resultados.

VI. 4. Resultados.

Enriqueciendo lo mencionado anteriormente, es importante establecer que se aplicó el procesamiento que se creía era el adecuado para cada tipo de señal a tratar, según lo establecido en base a las primeras pruebas obtenidas(comentadas en la sección anterior). La mayoría de los eventos fueron estudiados aplicándoles el respectivo procesamiento a las trazas correspondientes a la estación localizada en las cercanías de Iguala, Guerrero. A continuación se mencionan las trazas que la conforman:

- Traza 1 Canal HHV
- Traza 2 Canal HHN
- Traza 3 Canal HHE

Esta estación de la Red de Sismex es la más cercana a la costa , donde se tiene la mayor ocurrencia de sismos, razón por la cual es de las estaciones que registran más claramente los eventos.

Se presentaron dos casos en los que el aprendizaje no tuvo éxito, siendo necesario aplicar los dos tipos de procesamiento para asegurarse que ese no fuera el factor que estaba provocando la no clasificación satisfactoria de los patrones, siendo el caso de los sismos denotados con el número 12 y 14 respectivamente dentro de la tabla 6.4. En estos sismos fue necesario estudiar más a fondo qué otro factor estaba interfiriendo en el procesamiento ó al momento de definir la Arquitectura de la Red, para lo cual fué necesario cambiar uno a uno los parámetros.

El factor que generaba problemas en este caso específico fué el número de muestras por patrón, por lo que se decidió cambiarlo a 32. Estudiando el evento 9409-17-1916-58S SMX_16 designado con el número 12 dentro de la tabla 6.4, se encontró que el nivel de la señal antes del arribo de la traza procesada era muy estable, y que el momento del arribo se encontraba en el segundo 15.108. Ahora bien, puesto que al generar patrones de 16 muestras se está tomando realmente 0.8 segundos de la señal, y tomando en cuenta que el patrón 18 termina en el segundo 15.2, entonces el arribo es identificado dentro de las últimas muestras pertenecientes al patrón 18, el problema se presenta entonces dentro de los valores de amplitud que le preceden puesto que no varían en demasía, provocando que al tomar las muestras para constituir el siguiente patrón (el cual tendrá un promedio similar al anterior) sea difícil para la Red asociar dichos valores y sobre todo, determinar claramente en qué momento se identificó el arribo, es por esto que con ventanas de 16 muestras es imposible detectarlo siendo necesario tomar más número de muestras buscando tener la capacidad de discernimiento para identificarlo correctamente. Sucediendo algo similar con el evento identificado con el número 14.

Por otra parte, se presentaron situaciones donde a pesar de que el aprendizaje era éxitoso, la fase de reconocimiento para algunos de los sismos que constituían el grupo de estudio no tenía el éxito esperado, entonces la solución óptima que se encontró para estos casos fué utilizar otra traza perteneciente al evento que presentaba el problema, con el objeto de entrenar nuevamente a la Red.

Para ejemplificar lo anterior se cuenta con el caso del sismo 9410-04-2002-56S.SMX_16, primeramente se utilizó la traza 2 obteniendo un aprendizaje éxitoso pero el reconocimiento de algunos de los sismos que constituyen al grupo de estudio no tuvo el éxito esperado, es decir, la traza entrenada no logró una

clasificación óptima de los patrones. Ante esta situación se decidió utilizar la traza correspondiente al Canal IIIS logrando un éxito rotundo, los 13 sismos restantes se reconocieron sin ningún problema con la traza entrenada.

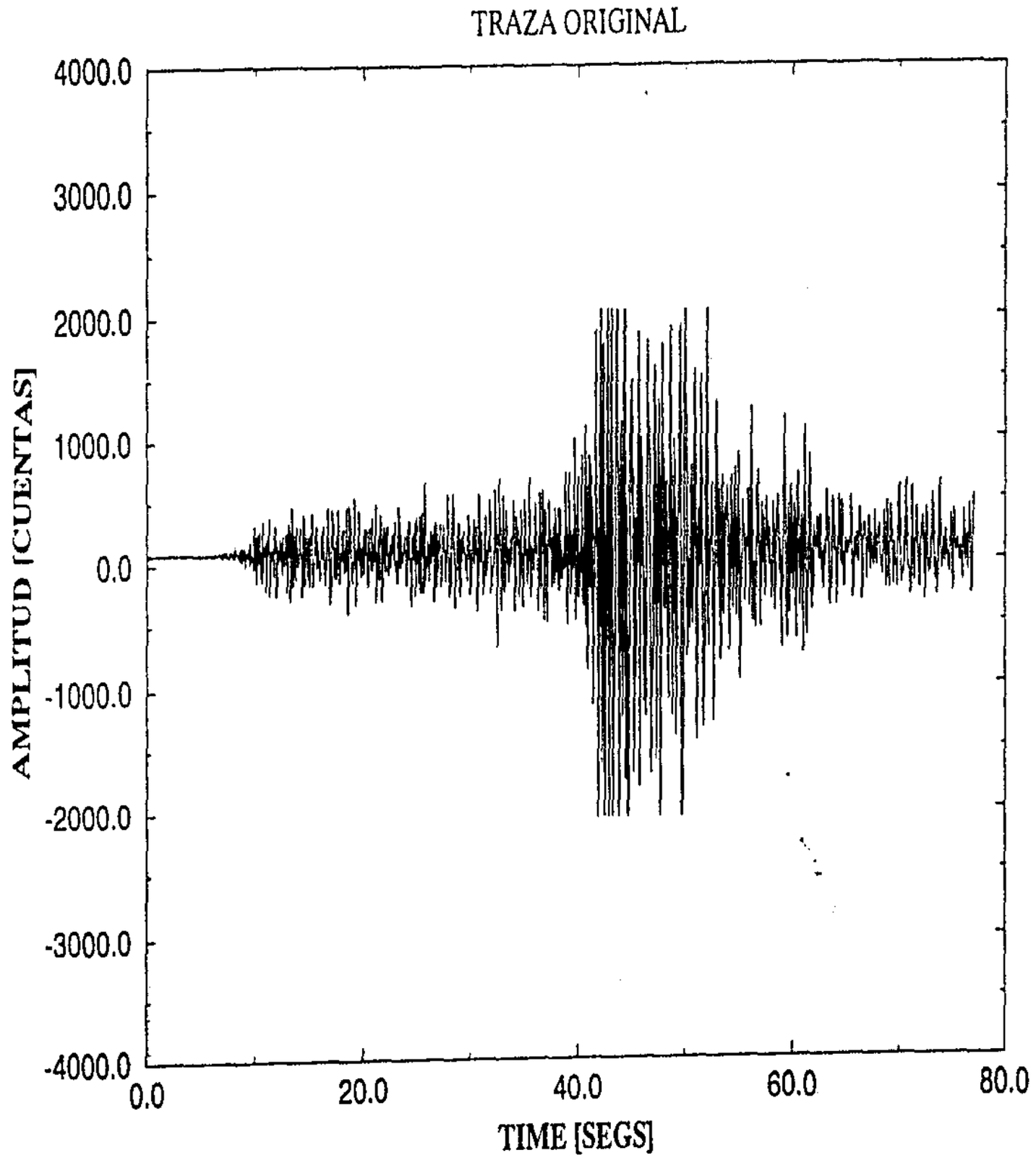
A continuación se mostrará una tabla con los datos referentes a los 14 sismos utilizados tanto en la étapa de entrenamiento como en la de reconocimiento. El campo referente al Aprendizaje indica el resultado de esta étapa, y los 2 últimos indican qué tipo de procesamiento se aplicó.

Número	Año	Mes	Día	Hora	Minuto	Segundo	Aprendizaje	STA 5	STA 20
1	94	1	2	15	22	58	éxitoso		Si
2	94	1	4	19	5	18	éxitoso		Si
3	94	1	5	18	53	31	éxitoso		Si
4	94	1	12	2	46	41	éxitoso	Si	
5	94	2	19	20	0	1	éxitoso	Si	
6	94	2	27	4	7	33	éxitoso	Si	
7	94	3	3	14	20	20	éxitoso	Si	
8	94	3	19	4	21	44	éxitoso		Si
9	94	3	20	5	6	25	éxitoso	Si	
10	94	3	25	16	33	4	éxitoso		Si
11	94	3	25	16	37	18	éxitoso		Si
12	94	9	17	19	16	58	no éxitoso	Si	Si
13	94	10	4	20	2	56	éxitoso	Si	
14	94	12	25	4	9	28	no éxitoso	Si	Si

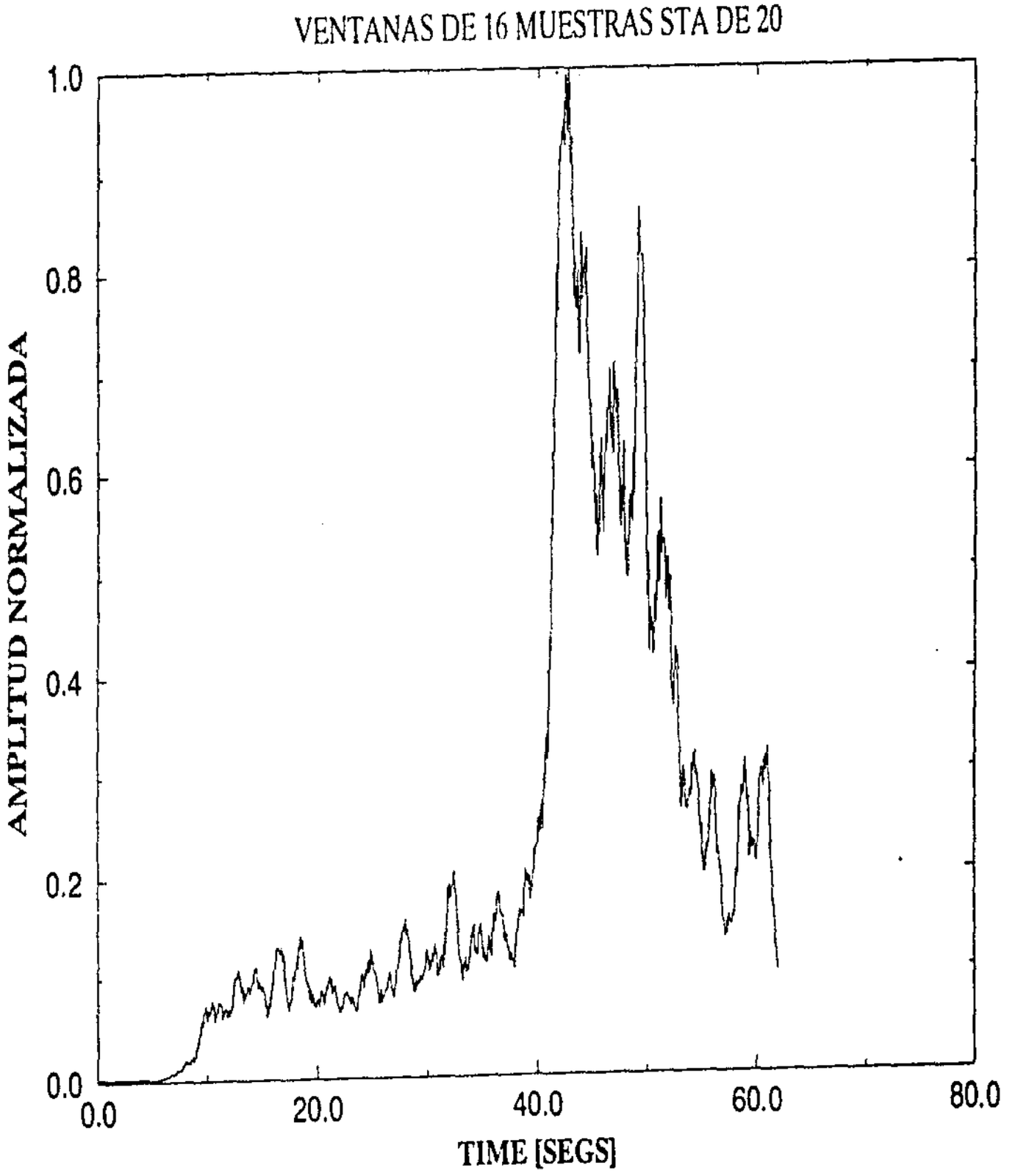
Tabla 6.4 Algunas características de los sismos utilizados en el entrenamiento.

Para tener una idea más clara de la eficiencia de la Red se muestran los resultados a los que se llegó dentro de la fase de Reconocimiento con el grupo de estudio en cuestión. Se identificarán a los eventos con el número designado en la tabla anterior.

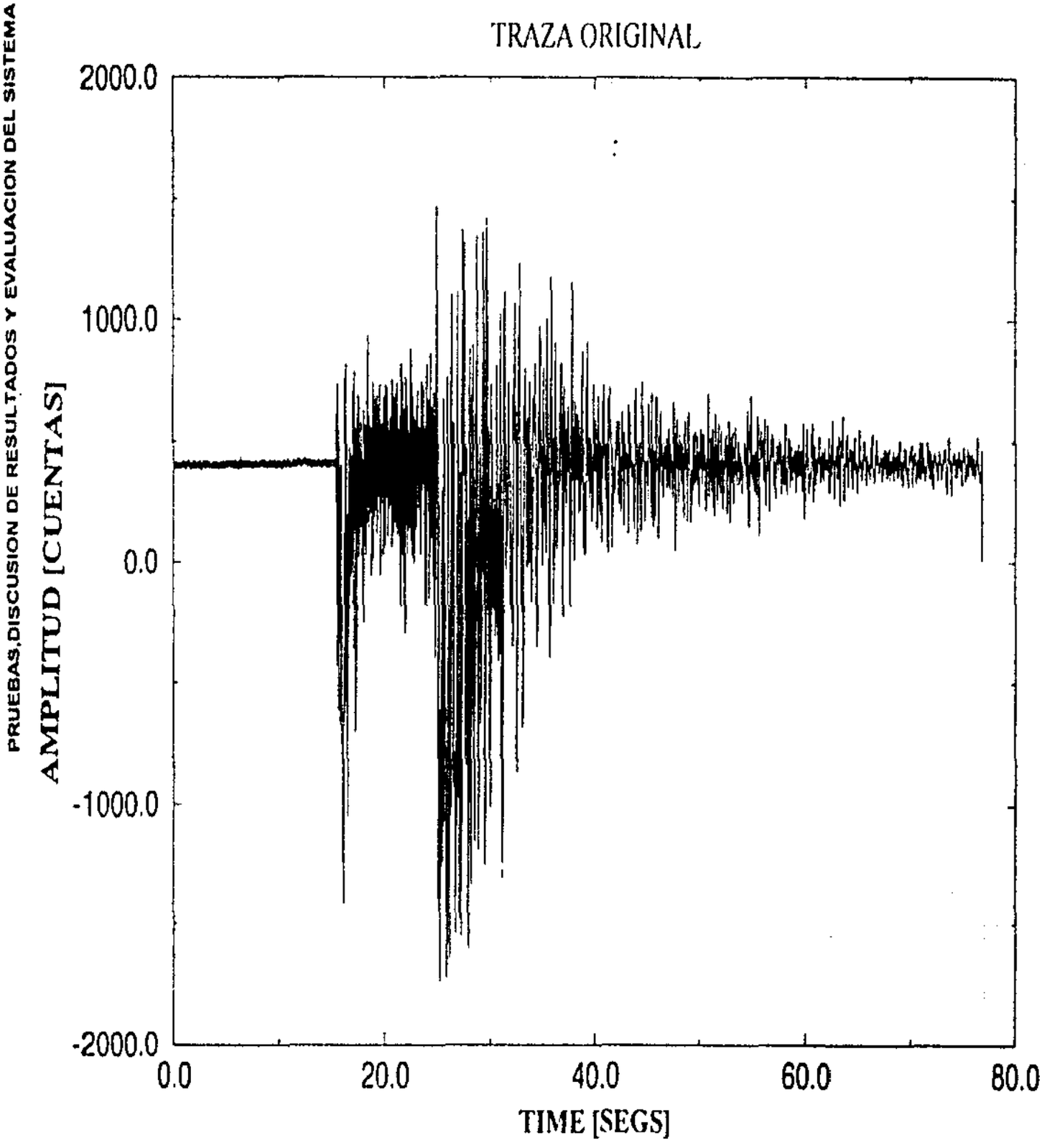
REGISTRO:9402-27-0407-33



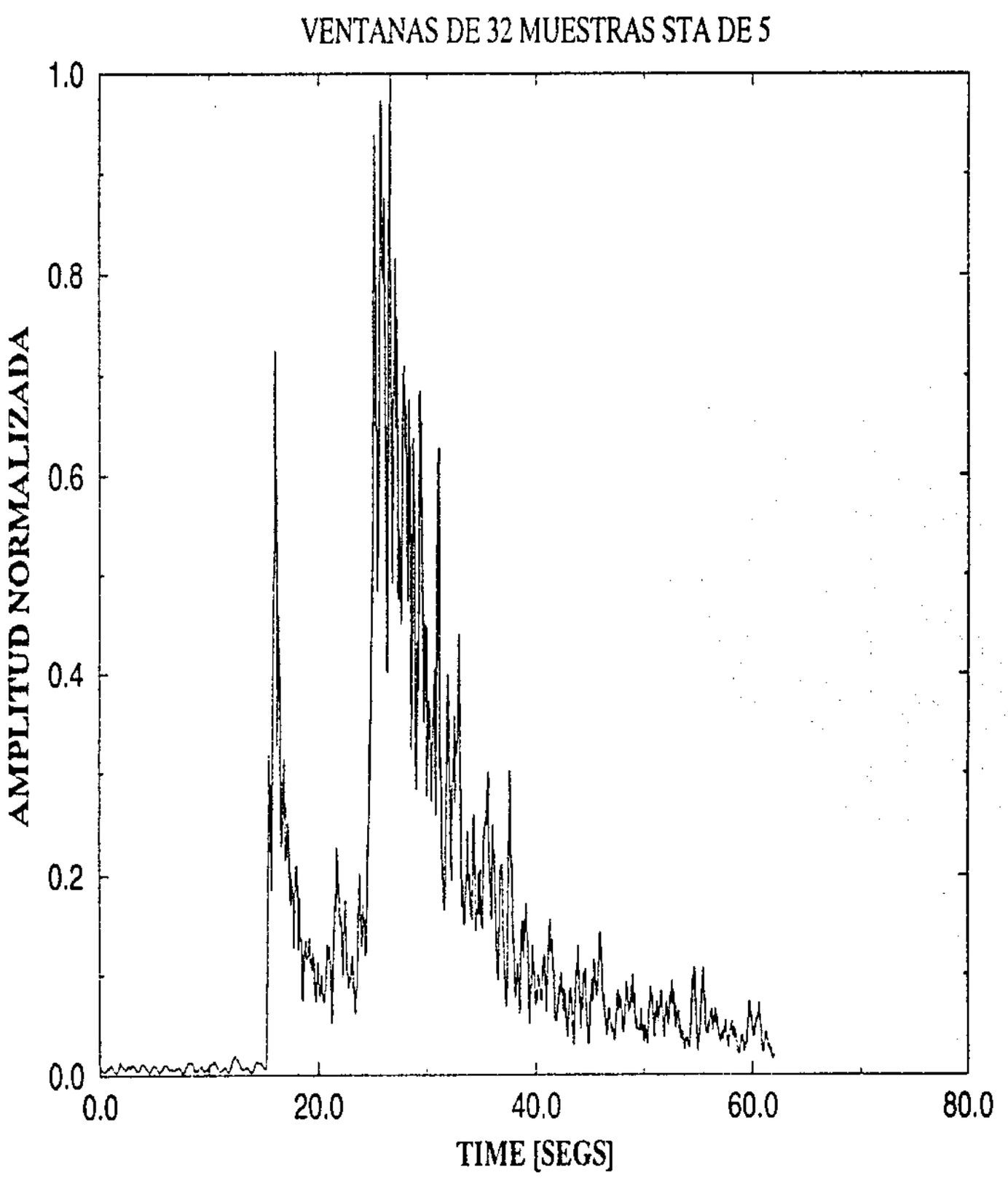
REGISTRO:9402-27-0407-33



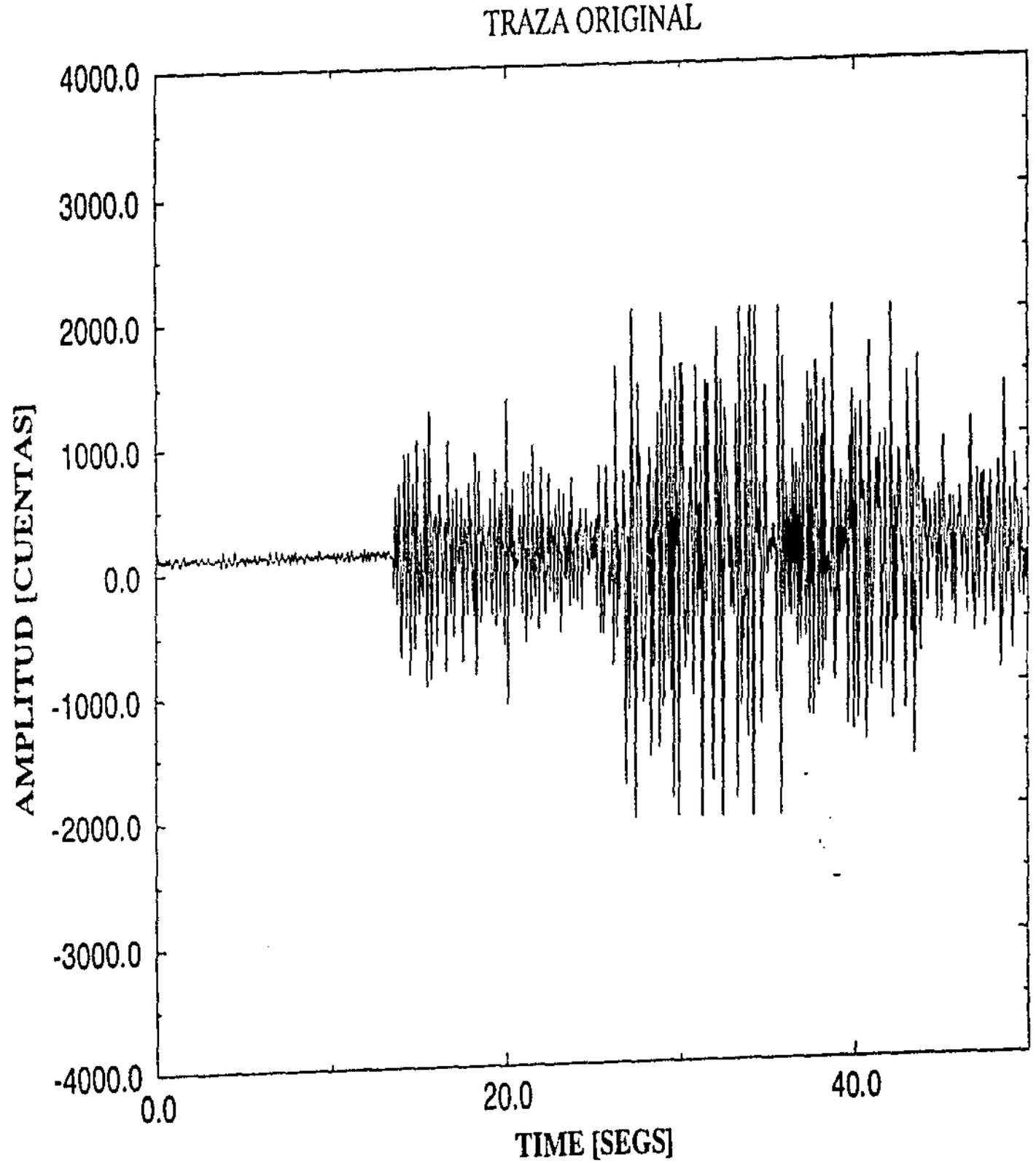
REGISTRO:9409-17-1916-58



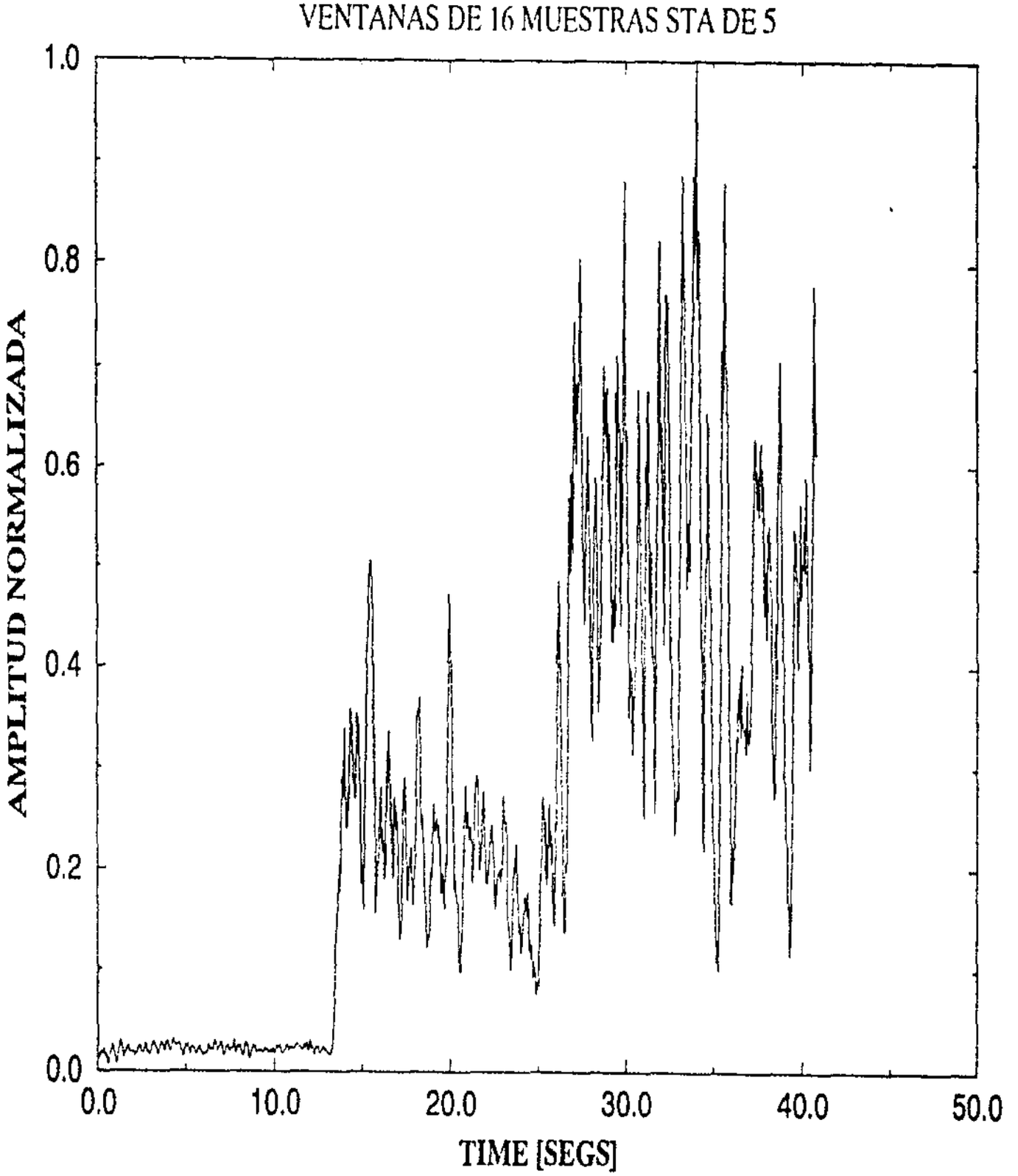
REGISTRO:9409-17-1916-58



REGISTRO:9410-04-2002-56



REGISTRO:9410-04-2002-56



VII. CONCLUSIONES.

Se cumplió el objetivo principal establecido al elaborar esta tesis, demostrando que las Redes Neuronales podían ser aplicadas en el reconocimiento de señales sísmicas , logrando a su vez ser un instrumento de ayuda para los sismólogos y lecturistas.

Este tipo de Redes Neuronales ya han sido probados con éxito en otros ámbitos y así mismo en el de la Sismología, pero no con señales sísmicas originadas por temblores.

Los programas diseñados están organizados de tal manera que permiten agregar nuevas funciones que pudieran mejorar al sistema, sin tener que realizar grandes cambios a lo ya implantado. Adaptándose de esta manera a las nuevas necesidades que surgan en el usuario, así como las innovaciones que genere el avance de la tecnología

Los primeros módulos fueron desarrollados en una computadora personal, pero se tuvieron algunos problemas debidos a la memoria con la que se contaba, esto sucede porque hay registros que contienen demasiados valores a ser procesados. Se decidió entonces que si el análisis de los registros digitales se llevaba acabo en la Estación de Trabajo, era mejor desarrollar el sistema bajo la misma plataforma, pretendiendo con esto tener disponibles las herramientas utilizadas para el análisis en un mismo lugar, globalizándolas.

Por lo tanto, se utilizaron las herramientas disponibles con que se contaba en la Coordinación para implementar los sistemas, reduciendo un posible gasto al implantar otro tipo de equipo para desarrollarlos

Se decidió implementar una interface gráfica aprovechando la facilidad que proporciona OPENWINDOWS, para facilitarle al usuario final la utilización y comprensión de los sistemas. Ahora bien, con la idea de globalizar el sistema se cuenta con una versión de los sistemas sin interface gráfica, pretendiendo con esto que sean compatibles con cualquier sistema, es decir, si se quiere utilizar en un sistema que no disponga de OPENWINDOWS se está en posibilidad de implantarlo.

Para ejecutarlo en cualquier estación de trabajo es necesario habilitar las siguientes variables de ambiente:

- *setenv LD_LIBRARY_PATH $OPENWINHOME*
- *setenv DISPLAY dirección de la máquina en donde se está trabajando físicamente.*

Analizando el comportamiento de las señales ante los sistemas se tiene que:

- Se observa una confiabilidad aceptable en la identificación del arribo.
- La identificación es muy sensible a la relación señal a ruido del registro y depende de una buena calibración de los parámetros de la Red.
- Es muy importante un buen preprocesado aplicado al registro, para obtenerlo, se debe observar la forma de onda de la señal sísmica y así determinar qué tipo de proceso se aplicará, puesto que si alguno de los parámetros mencionados en el capítulo anterior no es el adecuado, la Red Neuronal no garantiza el buen funcionamiento de la fase de aprendizaje así como de la fase de reconocimiento.
- La disminución en décimas de los errores provocan que los valores de salida calculados por la Red se ajusten al valor deseado dando por ende un mejor reconocimiento de la señal.

•Si no se tiene éxito en la fase de aprendizaje, primero se deben variar los parámetros que definen la Arquitectura de la Red, si habiéndolo hecho se continúa sin tener éxito, entonces se recomienda variar las muestras que constituyen a cada patrón, duplicándolas.

•Cuando se trata de reconocer un sismo aprendido con patrones de diferente número de muestras que el utilizado en la fase de reconocimiento, la Red no está capacitada para asociar dichos patrones y obtener un reconocimiento óptimo.

•Si en promedio los valores de amplitud de la señal no varían rápidamente después de encontrar el arribo de la fase "P" , se necesita tomar un patrón con más número de muestras para asociar a ese patrón con el arribo, de otra manera si se toman patrones con un número menor de muestras no se estará en posibilidad de discernir entre ese patrón y el subsecuente para asociar la llegada del arribo.

•La Red puede reconocer el arribo en el patrón siguiente al real, si los valores de amplitud de la señal después del arribo propasan en mucho los valores del patrón anterior donde se encontró el arribo.

•El tipo de proceso adecuado para cada señal depende precisamente de la forma de onda. Cuando el arribo sea Emergente el tipo de proceso aplicado generalmente debe ser STA de 20 muestras , esto es porque la variación entre el nivel de la señal antes del arribo y después de él no varía tan notablemente, es decir, los valores permanecen en promedio constantes.

•Se puede utilizar el proceso STA de 5 en señales emergentes, siempre y cuando la variación entre los valores de amplitud después del arribo no varíen tan constantemente.

•Cuando el arribo sea Impulsivo,como los valores son contrastantes en diferencia al nivel de la señal antes del arribo, es necesario tomar ventanas de 5 muestras de la señal.

•Si el nivel de la señal antes del arribo no fuera muy constante, es decir, que se haya mezclado con la señal una pequeña interferencia, y el arribo es muy emergente se puede confundir el momento del arribo , esto puede suceder cuando se selecciona que el algoritmo detecte el arribo de la fase.

Es importante mencionar que cuando se trabaja con Redes Neuronales sobre todo con el modelo de Retropropagación , se tienen que contemplar varios factores que influyen directamente en el desempeño del algoritmo (como se pudo constatar a lo largo del desarrollo de este trabajo) , es por esto que las pruebas a realizar para dar seguimiento a los problemas presentados son numerosas y variadas. Aunado esto con el hecho de que las señales sísmicas son tan diferentes se enfrentan otros problemas que no se habían contemplado.

Investigar y aplicar maneras diferentes de optimar los procesos utilizando las bases que la tecnología impone es una tarea del Ingeniero , con todo lo que implica la implantación de nuevos métodos después de la fase de prueba correspondiente. Por lo tanto se puede decir que este trabajo dá la pauta para la implantación del algoritmo de Retropropagación en el reconocimiento de señales sísmicas, dándole un nuevo enfoque a la identificación del arribo de la fase "P".

BIBLIOGRAFIA.

- Herbert Schildt
TURBO C/C ++
Manual de Referencia
Osborne McGraw - Hill, 1992.

- G. Suárez R. - Z. Jiménez J.
CUADERNOS DEL INSTITUTO DE GEOFISICA 1
Universidad Nacional Autónoma de México.

- Freeman/Skapura
Neural Networks. Algoritms, Applications and Programming Techniques.
Addison Wesley Publishing Company, Inc. USA, 1992.

- Hecht-Nielsen
Neurocomputing
Addison Wesley Publishing Company, Inc. California, USA, 1990.

- Pankaj Mehra - Benjamin W. Wah
Artificial Neural Networks. Concepts and Theory.
The Institute of Electrical and Electronics Engineers Inc USA, 1992.

- Clifford Lau
Neural Networks. Theorical Foundations and Analysis
The Institute of Electrical and Electronic Engineers, Inc. New York, 1991.

- Yoh-Han Pao
Adaptive Pattern Recognition and Neural Networks
Addison Wesley Publishing Company, Inc. USA, 1989.

- Haskov J. - Lindholm C. 1994.
The SEISAN earthquake analysis system.
Institute of Solid Earth Physics. pp.139.

- Jin Wang - Ta-Liang Teng
Artificial Neural Network-Based Seismic Detector
Bulletin of the Seismological Society of America, Vol. 85, No. 1
pp. 308-319, February 1995.

- Youngjik Lee - Sang-Hoon Oh - Myung Won Kim
An Analysis of Premature Saturation in Backpropagation Learning
Neural Networks, Vol. 6, pp 719-728.
Pergamon Press. USA, 1993

- Stevan V. Ordri- Dusan P. Petrovacki- Gordana A. Krstonosic
Evolutional Development of a Multilevel Neural Network
Neural Networks, Vol. 6, pp. 583-595
Pergamon Press. USA, 1993.

- Ibrahim Palaz - Ronald C. Weger
Waveform recognition using neural networks
Geophysics: The Leading Edge of Exploration. Special Section:Computers
The Society of Exploration Geophysicists
Vol. 9, No. 3. pp. 28-31. March, 1990.

- Farid U. Dowla - Steven R. Taylor- Russell W. Anderson
Seismic Discrimination with Artificial Neural Networks
Preliminary Results with Regional Spectra Data
Bulletin of the Seismological Society of America
Vol, 80. No. 5, pp.1346-1373, October 1990.

- Stelios C. A. Thomopoulos - Dimitrios K. Bougoulias
A Self-Organizing Neural Network for Automatic Pattern Recognition
and Classification
Proceedings of the 30th Conference on Decision and Control
Brighton, England. December 1991.

R	E	C	O	N	O	C	I	M	I	E	N	T	O
		1	2	3	4	5	6	7	8	9	10	11	13
E	Sismo												
N	1	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si
T	2	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si
R	3	Si	Si	Si	Si	Si	Si	Si	Si	Si	No	Si	Si
E	4	Si	Si	Si	Si	Si	Si	No	Si	Si	Si	Si	Si
N	5	No	Si	Si	No	Si	No	Si	No	No	Si	No	Si
A	6	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	No	Si
M	7	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si
I	8	Si	No	Si	Si	Si	Si	Si	Si	Si	Si	No	Si
E	9	Si	Si	Si	Si	Si	Si	Si	Si	Si	No	Si	Si
N	10	Si	Si	No	No	Si	Si	Si	Si	Si	Si	No	Si
T	11	Si	Si	Si	Si	Si	Si	Si	Si	Si	No	Si	Si
O	13	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si

Tabla 6.5 Resultados de la Fase de Reconocimiento .

Como se observa la eficiencia del algoritmo es de un 80-90%. Un muy buen resultado para la fase de desarrollo de este sistema, sobre todo teniendo en cuenta que se partió de un procesamiento básico para estudiar a detalle lo que esto implicaba, pensando en un procesamiento más complejo al que se está en posibilidad de llegar tomando en consideración los resultados aportados através de este trabajo

VI. 5. Mejoras propuestas.

La parte del proceso aplicado a las señales sísmicas dentro del sistema de preprocesamiento esta abierto para incluir nuevos procesos, que pudieran ayudar a depurar más y mejor la señal de entrada a la Red Neuronal. Esta inclusión de nuevos procesos dependería de un estudio más exhaustivo en base a los resultados generados , para saber en donde se podría robustecer a la Red.

Mejorar el procedimiento que calcula el momento en que se produce el arribo, para asegurarnos que el tiempo de éste sea más éxacto.

Utilizar el algoritmo de deslizamiento de los patrones para tener una aproximación más eficiente del tiempo en el que ocurrió el arribo, claro que para lograrlo se tendría que estudiar la manera de incrementar la velocidad de convergencia de la red con patrones deslizados.

Se podría también estudiar los valores de las matrices de pesos a las que se llegaron en la fase de entrenamiento para discernir en qué momento éstos no varían considerablemente, y desactivar algunos nodos de la capa oculta, inhibiéndolos. Probar otros rangos posibles para los parámetros de definición de la arquitectura de la Red pretendiendo ajustarlos

APENDICE A.

Declaración de los objetos de OPENWINDOWS utilizados en la creación del ambiente de ventanas del sistema.

Primeramente, para crear el ambiente en el que se desarrollará la aplicación es necesario diseñar el frame ó ventana principal, este prepara la pantalla para ser utilizada por los otros objetos como son subframe, canvas y paneles, estos últimos para crear botones, imprimir y recibir variables, por lo mencionado anteriormente se puede decir que el frame permite presentar la información al usuario de una manera más clara.

Ambiente Principal.

Para declarar el frame ó ambiente principal, se deben declarar ciertos parámetros que indiquen las caracteristicas que contendrá el área de trabajo, la función xv_init inicializa el ambiente, ahora la descripción del frame es la siguiente:

```
frame= (Frame) xv_create ((Window)0, FRAME,
        WIN_X,                  0,
        WIN_Y,                  0,
        XV_HEIGHT,              500,
          XV_WIDTH,               200,
        FRAME_LABEL,            "titulo",
        FRAME_NO_CONFIRM,       FALSE,
        FRAME_SHOW_FOOTER,      TRUE,
        NULL);
```

donde:xv_create permite crear el frame, las siguientes coordenadas son de inicio tanto en X como en Y e indican los valores de la posición de la ventana en pantalla , en este caso la altura del frame será de 500 pixeles y el ancho de 200 indicando el tamaño de la ventana , por cierto el pixel es la celda minima de graficación.

FRAME_LABEL indica el titulo que tendrá la aplicación. Las demás condiciones mostradas son etiquetas del mismo frame, en donde se le indica si se quiere que se muestren comentarios al final del frame, para un mejor entendimiento es pertinente hacer la analogia de ésto como si se tratara de notas al pie de página.

Una vez declarado el frame, se procede a declarar cada una de las áreas que conforman el ambiente de trabajo, como lo es la Barra de Menus, Sección de graficación y finalmente sección de mensajes (mencionados anteriormente).

Barra de Menus.

Para la construcción del menu se deben declarar las variables de ambiente que almacenarán los parámetros de éste, así como de sus respectivos submenus ó comandos, por ejemplo:

```
static Menu Menu1;
 static  Menu  Menu2;              /* Declaración de variables de ambiente*/
.....
static Menu Submenu1;
```

Subsecuentemente se presenta la declaración de los submenus que contienen la lista de comandos a seguir pertenecientes al Menu principal .

```
Submenu1=(Menu)xv_create(NULL,MENU,                    Crea lista de comandos.
            MENU_NOTIFY_PROC,        proceso1,         Llamado a procedimiento.
            MENU_ITEM,
                MENU_STRING,         "comando",        Título del comando ó función.
                MENU_NOTIFY_PROC,    proceso2,         Procedimiento asociado al comando.
            NULL,
            MENU_ITEM,
                MENU_STRING,         "comando",
                MENU_NOTIFY_PROC,    proceso3,
            NULL,
            MENU_ITEM,
                MENU_STRING,         "comando",
                MENU_NOTIFY_PROC,    proceso4,
            NULL,
NULL);
```

Este es un tipo de submenu que consta de una serie de comandos a realizar según se seleccione. Submenu1 debe ser el nombre del procedimiento asignado a la variable **PANEL_ITEM_MENU** en la creación del botón que accesará dicho menu.

A continuación se presentará otro tipo de submenu en donde se tiene una lista de comandos asociados a un sólo procedimiento. Es usado cuando se desea seleccionar un parámetro ó valor requerido por el procedimiento.

```
Submenu2=(Menu)xv_create(NULL,MENU,
            MENU_STRINGS,              "comando1",
                                     "comando2",
                                     "comando3",....., NULL,
            MENU_NOTIFY_PROC,          proceso,
 NULL);
```

MENU_ITEM, anuncia que se trata de la declaración de un menu. **MENU_STRING** menciona el nombre que se desea aparezca en el comando dentro del menu. **MENU_NOTIFY_PROC** contiene el nombre del procedimiento al cual se hará referencia al presionar ese comando, **NULL** es como una etiqueta que indica al sistema la finalización de la declaración .Se puede utilizar la declaración **MENU_ITEM** tanto como comandos se necesiten

Como último paso, las instrucciones correspondientes para la creación de los botones que accesarán los menus que forman parte de la barra de menus son declaradas. En éstos se realiza el llamado de las variables que contienen la lista de comandos que se desean mostrar. La declaración **PANEL_BUTTON** accesa a los menus ó lista de comandos creando un botón gráficamente.

```
(void) xv_create(panel,  PANEL_BUTTON,                  Declaración de menu PARAMETROS
       PANEL_ITEM, X,          60,                      Posición en la Barra de Menus.
       PANEL_ITEM, Y,          6,
       PANEL_LABEL_STRING,  "PARAMETROS",               Titulo aparece en la Barra de Menus.
       PANEL_NOTIFY_PROC,   seleccion,
       PANEL_ITEM_MENU,     submenu1,                   Llamado a procedimiento que contiene
       NULL);                                           lista de comandos a ser accesados.
                                                        Fin de declaración menu PARAMETROS.
```

PANEL_ITEM_X como **PANEL_ITEM_Y** tienen las coordenadas en donde el botón va a ser creado, **PANEL_LABEL_STRING** contiene el nombre del botón , **PANEL_NOTIFY_PROC** contiene el nombre de un procedimeinto que muestra en la parte inferior del frame el botón seleccionado para mantener al usuario informado sobre la etapa en la que se encuentra el sistema, finalmente **PANEL_ITEM_MENU** contiene el nombre del procedimiento en donde se declarará a su vez la lista de comandos(o menu) que podrán ser accesados.

Cuadros de diálogo.

Un cuadro de diálogo es una ventana en donde se pide al usuario que proporcione información requerida por algún proceso para ser ejecutado Un cuadro de diálogo puede estar formado por botones de selección, botones de comando, mensajes, etc.

```
Panel    panel_x;
```

Es necesario declarar este parámetro referente al panel para la habilitación de él.

```
panel_x=xv_get(subframe, FRAME_CMD, PANEL,
          PANEL_LAYOUT,  PANEL VERTICAL,
          NULL);
```

A continuación se presenta la declaración de un frame en donde se contendrán los controles del cuadro de diálogo.

```
subframe=xv_create(NULL,FRAME_CMD,              Declaración de variable de ambiente.
          WIN_X,  600,
          WIN_Y,  100,
          FRAME_LABEL,  nombre,
          WIN_SHOW,  TRUE,
          NULL);
```

Se procede a presentar la declaración que habilita el área que contendrá los controles. Esta variable debe ser declarada al inicio del procedimiento que la utilizará.

```
xv_create(panel_x,PANEL_MESSAGE,
          PANEL_ITEM_X,   5,
          PANEL_ITEM_Y,   5,
          PANEL_LABEL_BOLD,  TRUE,
          PANEL_LABEL_STRING,  "Razon de diezmado",
          NULL);
```

PANEL_MESSAGE habilita al panel para imprimir mensajes.

PANEL_ITEM_X, 5 asi como **PANEL_ITEM_Y, 5,** indican las coordenadas dentro del panel en las cuales se localizará el mensaje asignado a la variable **PANEL_LABEL_STRING** colocando dicho mensaje entre " ".

```
valor=xv_create(panel_x,PANEL_TEXT,
        PANEL_ITEM_X,   5,
        PANEL_ITEM_Y, 200,
        PANEL_LABEL_BOLD,   TRUE,
        PANEL_LABEL_STRING, "       ",
        PANEL_VALUE_STORED_LENGTH, 15,
        PANEL_VALUE_DISPLAY_LENGTH, 15,
        PANEL_NOTIFY_PROC, pide_dato,
        PANEL_NOTIFY_LEVEL, PANEL_ALL,
        PANEL_VALUE, razon,
        0);
```

panel_x debe ser declarada previamente(como se explicó anteriormente) para habilitar al subframe y poder utilizarlo al imprimir los mensajes **PANEL_TEXT** identifica el tipo de panel utilizado, TEXT habilita al panel ya sea para imprimir el valor dado ó proporcionarlo, la estructura es la misma para cualquiera de estas 2 funciones, la variacion se presenta en la variable a ser impresa ó proporcionada, es decir, si dicha variable ya tiene asignado un valor será impresa pero si no tiene ninguno entonces se procede a aceptar el valor.

PANEL_ITEM_X, 5 asi como **PANEL_ITEM_Y, 200,** indica las coordenadas, ahora la variable que habilita al panel para la recepción de mensajes es **PANEL_LABEL_BOLD** con la etiqueta TRUE. **PANEL_VALUE_STORED_LENGTH** va a almacenar a la variable indicando en el número que la acompaña la longitud, en este ejemplo el valor es de 15.

PANEL_VALUE_DISPLAY_LENGTH al igual que en la variable anterior está acompañada por un número que indica la longitud de la linea a ser desplegada, la cual será asignada a la variable a ser salvada ó impresa.

PANEL_NOTIFY_PROC indica el procedimiento a ser referenciado y por último **PANEL_VALUE** recibe el nombre de la variable, ésta debe ser declarada de tipo char.

En la siguiente declaración se crea un botón con la etiqueta "ASIGNAR" dentro de la ventana presentada, dicho botón al ser presionado llama al procedimiento cerrar_archivo, al momento de la llamada la ventana desaparece dando paso a la siguiente acción a realizar.

```
(void) xv_create(panel, PANEL_BUTTON,
       PANEL_ITEM, X,              60,
       PANEL_ITEM, Y,              6,
       PANEL_LABEL_STRING,   "ASIGNAR",
       PANEL_NOTIFY_PROC,    cerrar_archivo,
       NULL);
```

<u>Sección de graficación.</u>

Para la definición del área asignada a datos y/ó impresión de gráficas, es necesario indicar que porción de la ventana principal le será asignada, el Canvas ó lienzo es declarado y almacenado en una variable para su posterior uso . Se debe declarar asi.

```
canvas=(Canvas)xv_create(frame, CANVAS,          Se indica que la variable corresponde al lienzo
        WIN_X,                  0,               Posición del frame en x.
        WIN_Y,                  40,              Posición del frame en y.
        WIN_HEIGHT,             800,
        CANVAS_WIDTH,           1200,            Tamaño del área en pixeles.
        CANVAS_HEIGHT,          800,
        CANVAS_X_PAINT_WINDOW,  TRUE,            Habilita la impresión en esta área.
        NULL);
```

En la variable Canvas se almacenan las caracteristicas del lienzo que posteriormente podrán ser llamadas por las diferentes instrucciones de graficación como por ejemplo:

```
Display   *dpy
Window    xwin                                   Variables de ambiente
Gc         gc

gc=DefaultGC(dpy,DefaultScreen(dpy))             Asignación de valores al parametro gc
XDrawLine(dpy, xwin, gc, X1, Y1, X2, Y2)         Impresión de una linea.
XDrawString(dpy, xwin, gc, X, Y, cadena, strlen(cadena))  Impresión de una cadena de caracteres
```

donde:

X1,Y1	posición del primer punto en pixeles.
X2,Y2	posición del segundo punto en pixeles
X, Y	posición de la cadena en pantalla
dpy, xwin, gc	variables de ambiente.

Para habilitar el área del lienzo e indicar que se va a trabajar en él , se lleva a cabo la declaración siguiente.

```
xv_set(canvas,
        WIN_X,                  0,
        WIN_Y,                  40,
        WIN_HEIGHT,             altura,
        WIN_WIDTH,              ancho,
        CANVAS_X_PAINT_WINDOW,  TRUE,
        CANVAS_REPAINT_PROC,    dibuja,
        NULL);
```

Con el xv_set se asignan nuevos valores al frame para modificarlo.

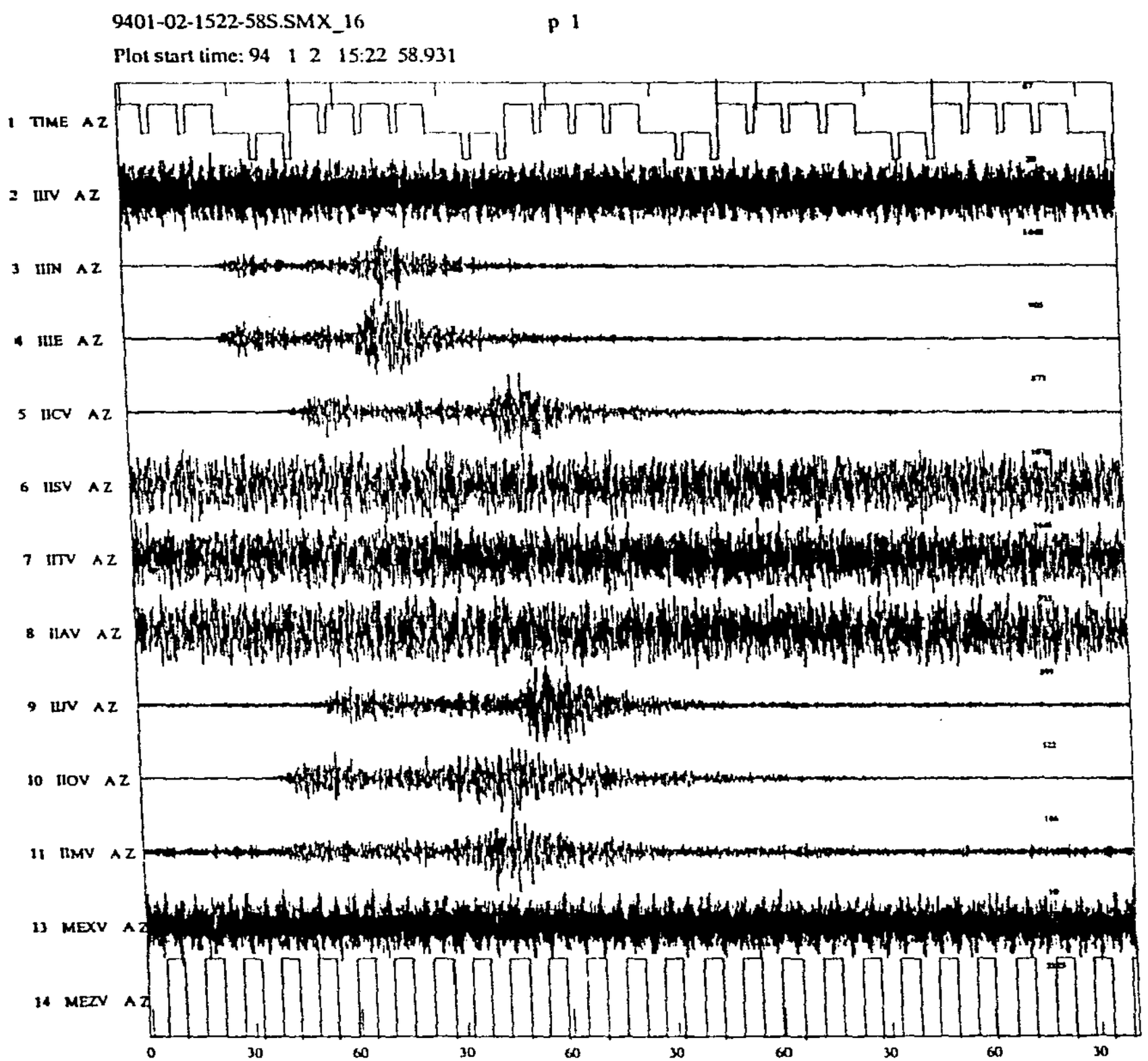
IV. 5. 2. 5. Sección de mensajes.

Para habilitar esta sección se utiliza la estructura antes mencionada. xv_set con el mensaje que se quiere aparezca al final del frame, precisamente dentro del área de mensajes.

APENDICE B.

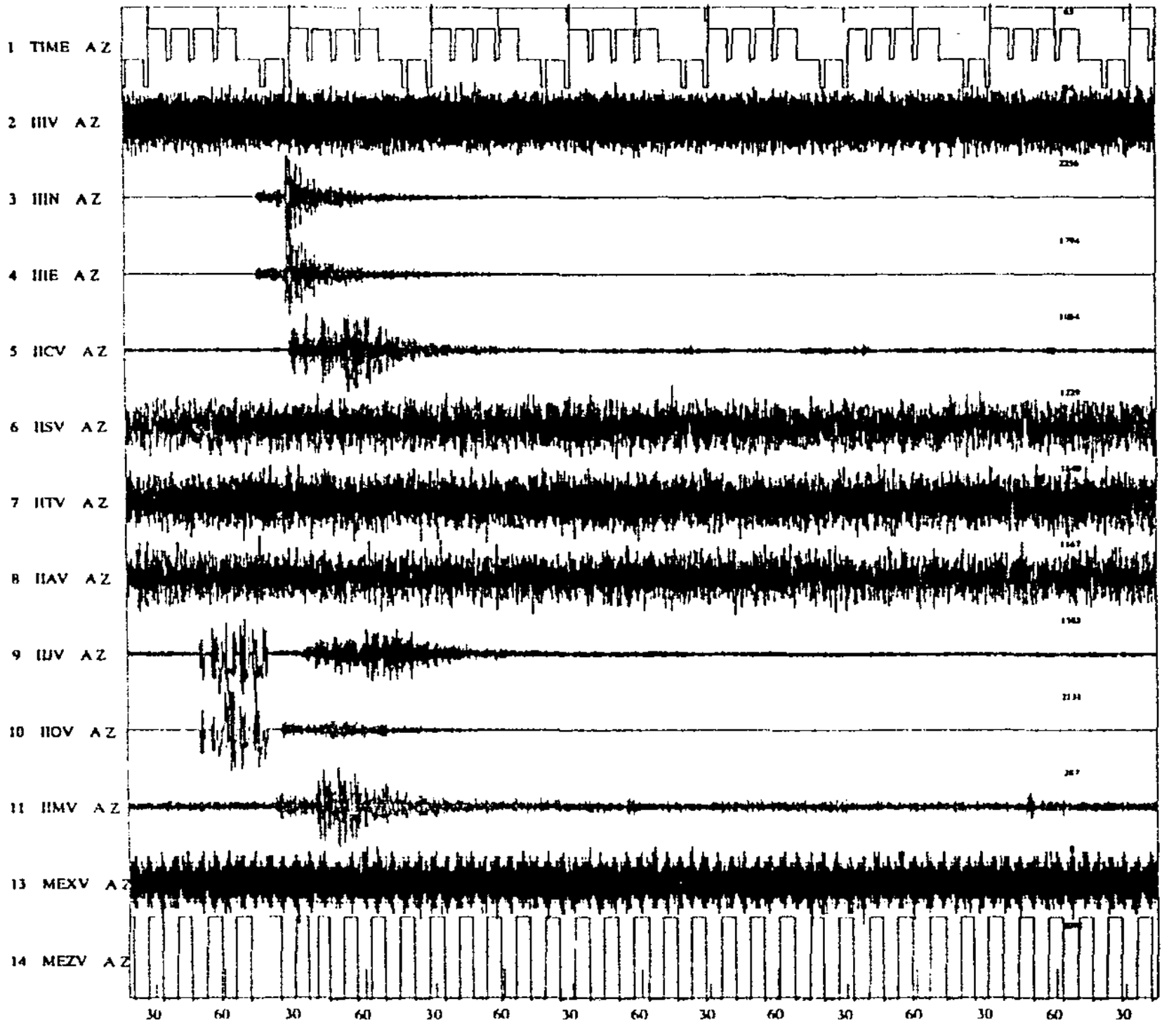
En esta sección se presentarán los 14 sismos que constituyen tanto el grupo de Entrenamiento como el de Reconocimiento.

En las gráficas se pueden apreciar las 16 trazas correspondientes a cada estación de la Red de Sismex. El primer canal es la referencia de tiempo, la cual no es tomada en cuenta dentro del sistema, por lo que se empieza con la numeración de las trazas a partir del Canal IIIV.


9401-02-1522-58S.SMX_16
p 1
Plot start time: 94 1 2 15:22 58.931
1 TIME AZ
2 IIIV AZ
3 IIIN AZ
4 IIIE AZ
5 IICV AZ
6 IISV AZ
7 IITV AZ
8 IIAV AZ
9 IIJV AZ
10 IIOV AZ
11 IIMV AZ
13 MEXV AZ
14 MEZV AZ
0
30
60
30
60
30
60
30
60
30

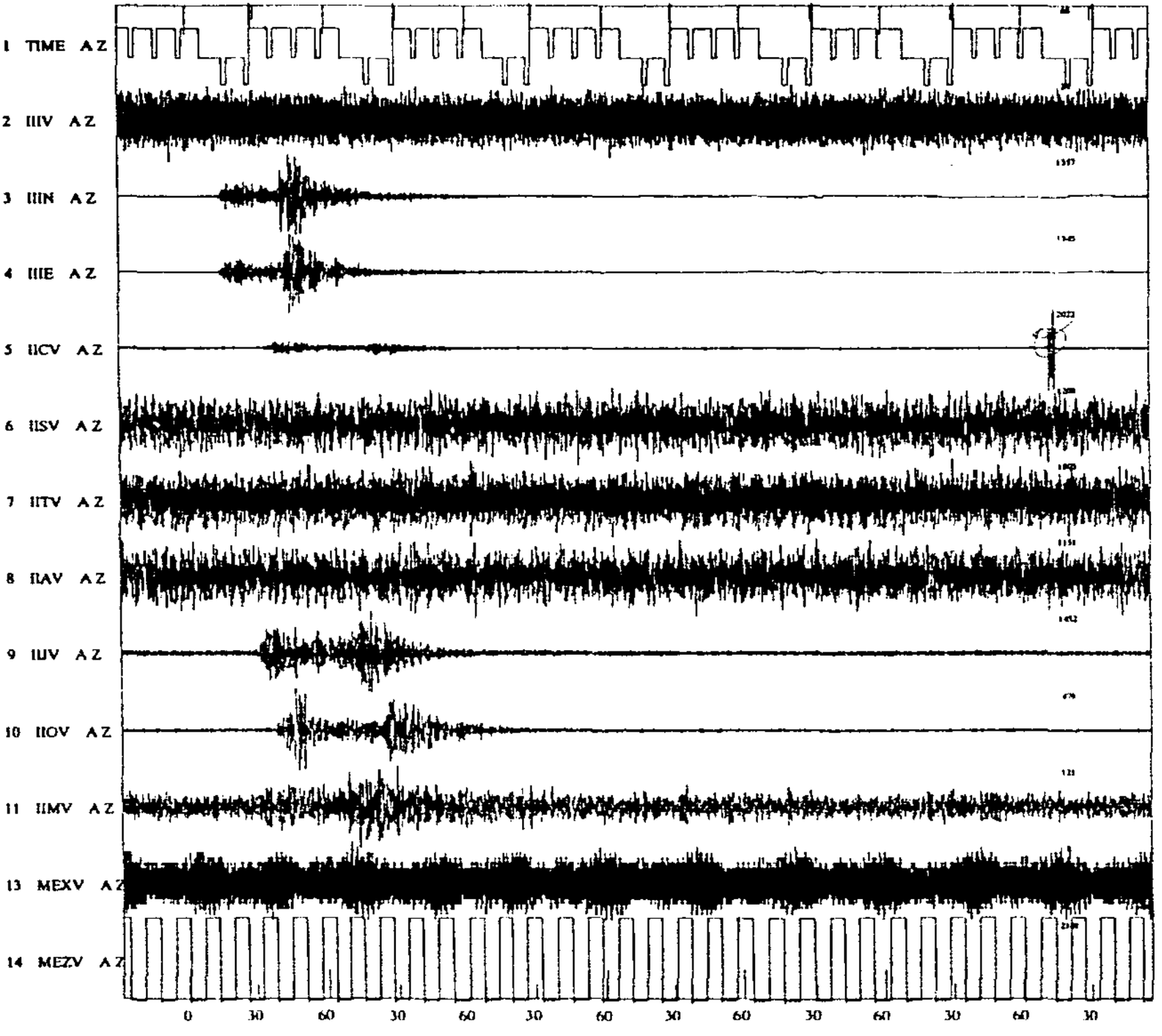
9401-04-1905-18S.SMX_16 p 1

Plot start time: 94 1 4 19: 5 18.213



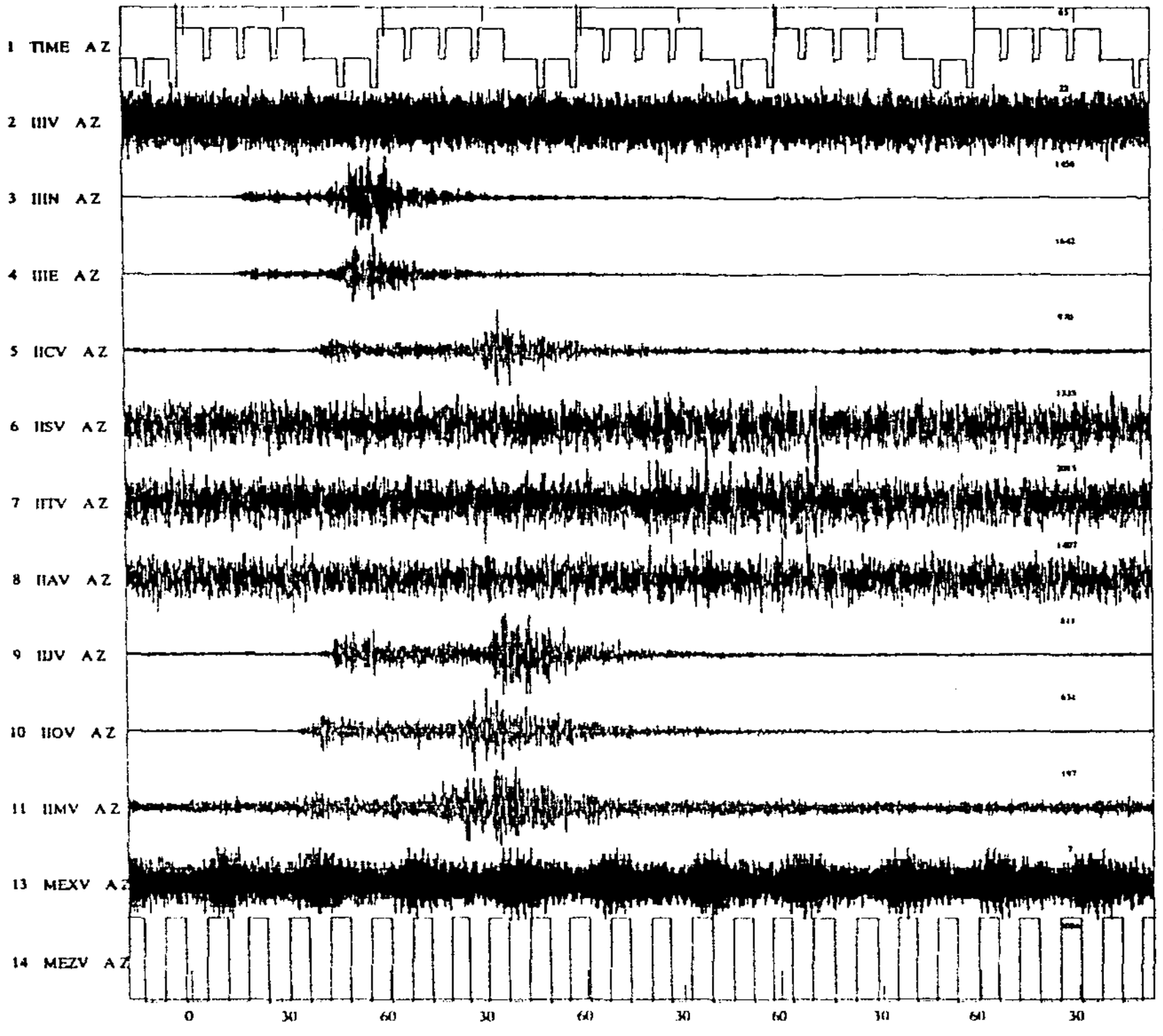
9401-05-1853-31S.SMX_16 p 1

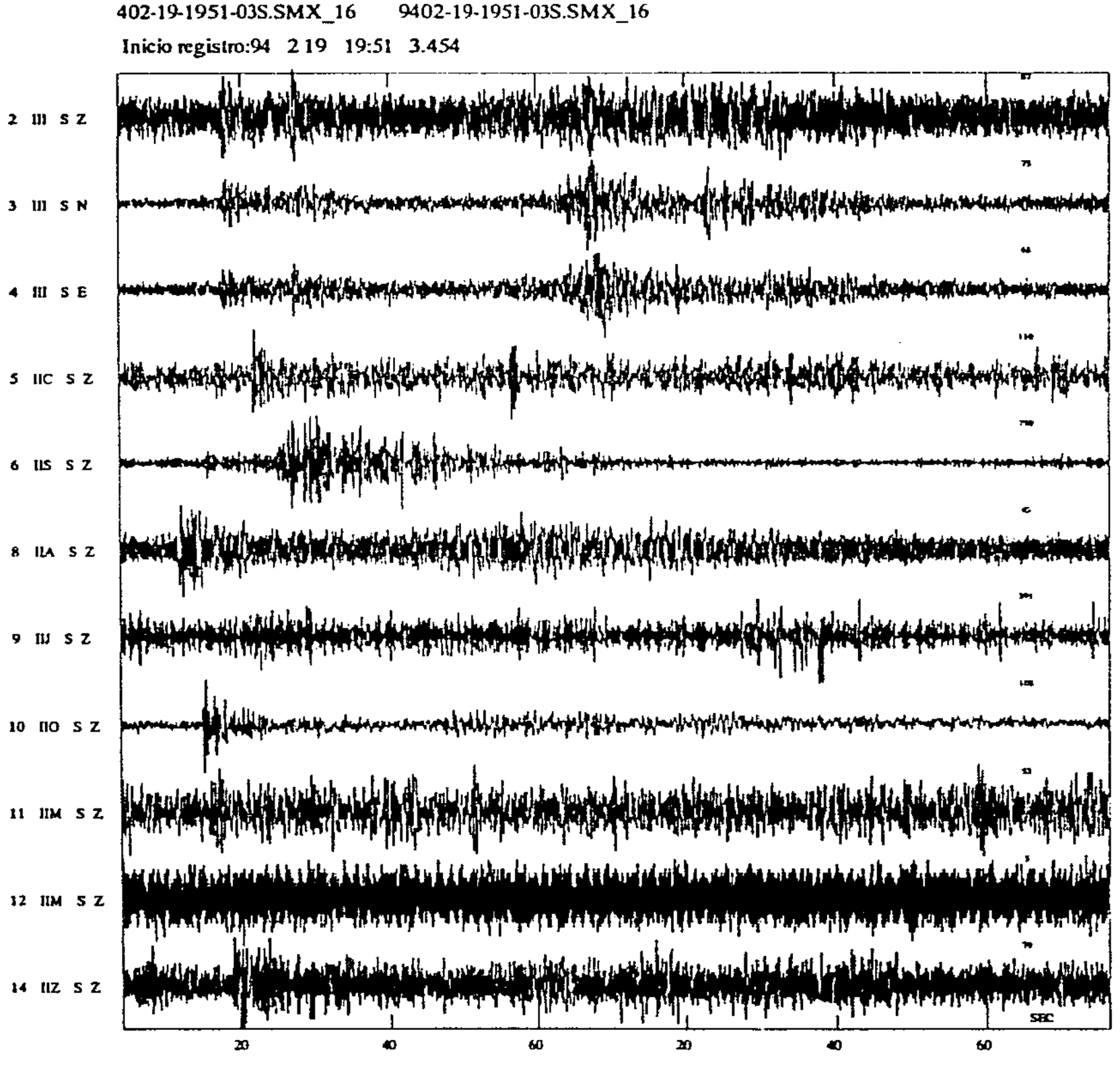
Plot start time: 94 1 5 18:53 31.101



9401-12-0246-41S.SMX_16 p 1

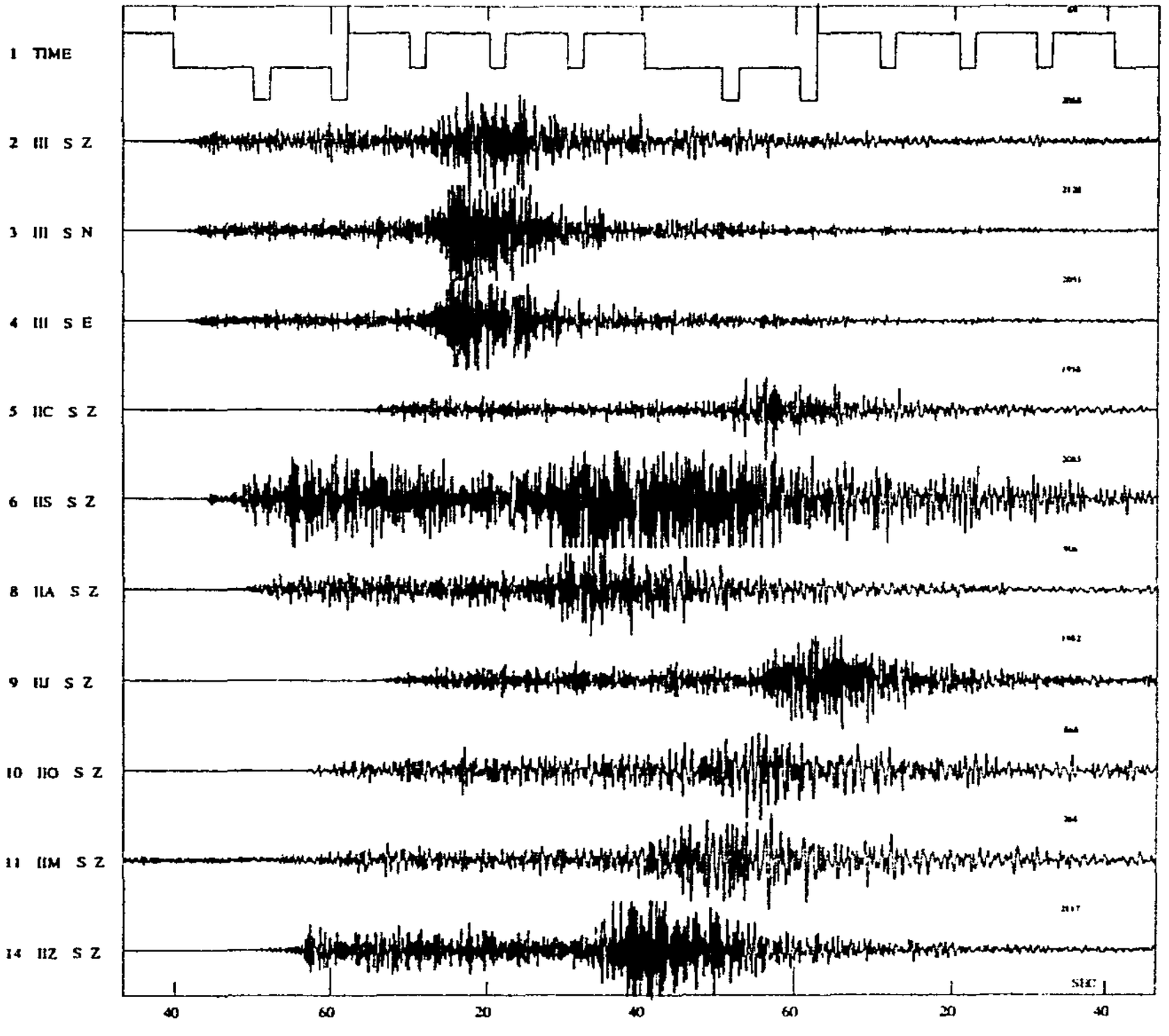
Plot start time: 94 1 12 2:46 41.000




402-19-1951-03S.SMX_16
9402-19-1951-03S.SMX_16
Inicio registro:94 2 19 19:51 3.454
2 III S Z
3 III S N
4 III S E
5 IIC S Z
6 IIS S Z
8 IIA S Z
9 IIJ S Z
10 IIO S Z
11 IIM S Z
12 IIM S Z
14 IIZ S Z
SEC
20
40
60
20
40
60

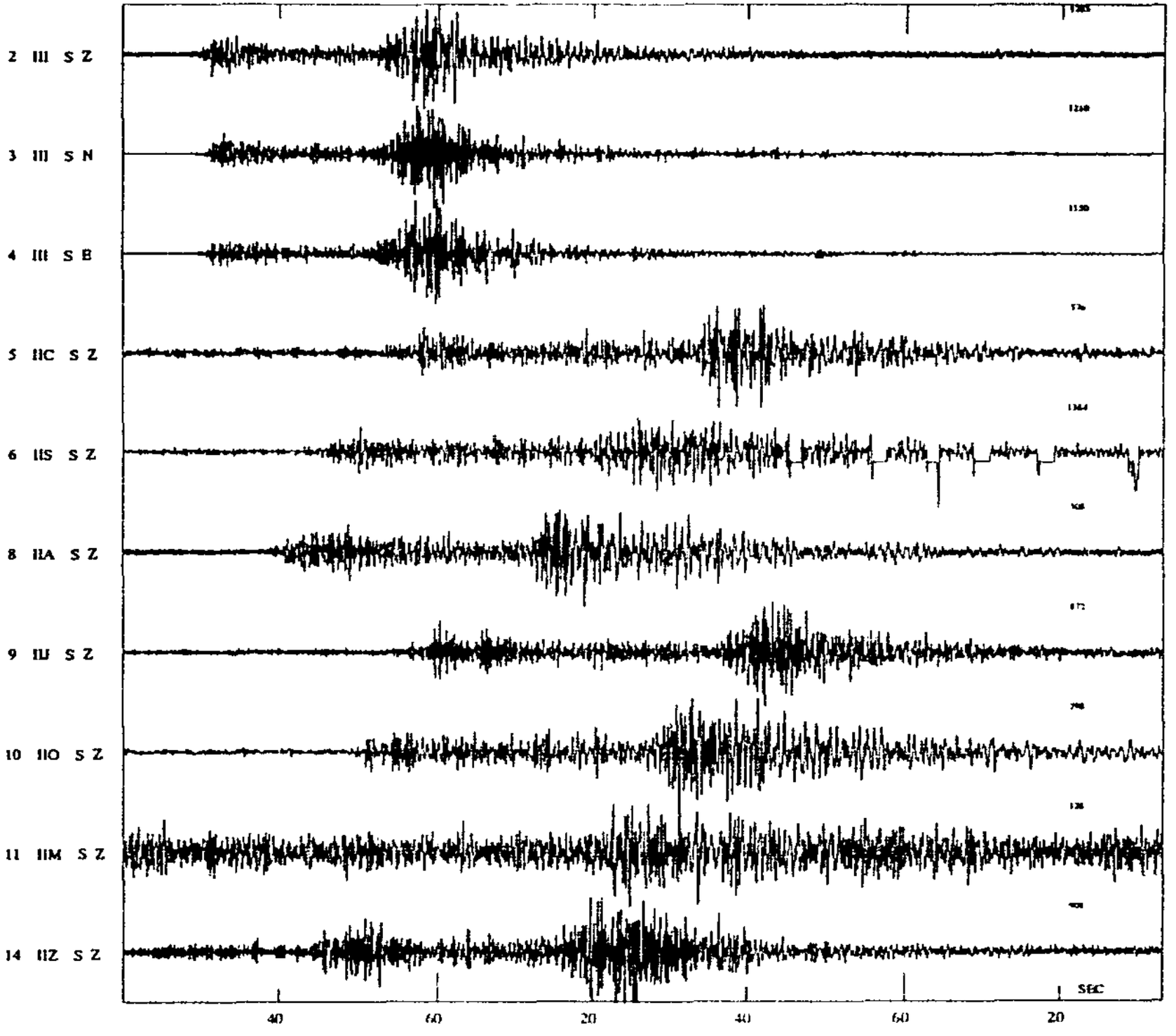
402-27-0407-33S.SMX_16 9402-27-0407-33S.SMX_16

Inicio registro:94 2 27 4: 7 33.362



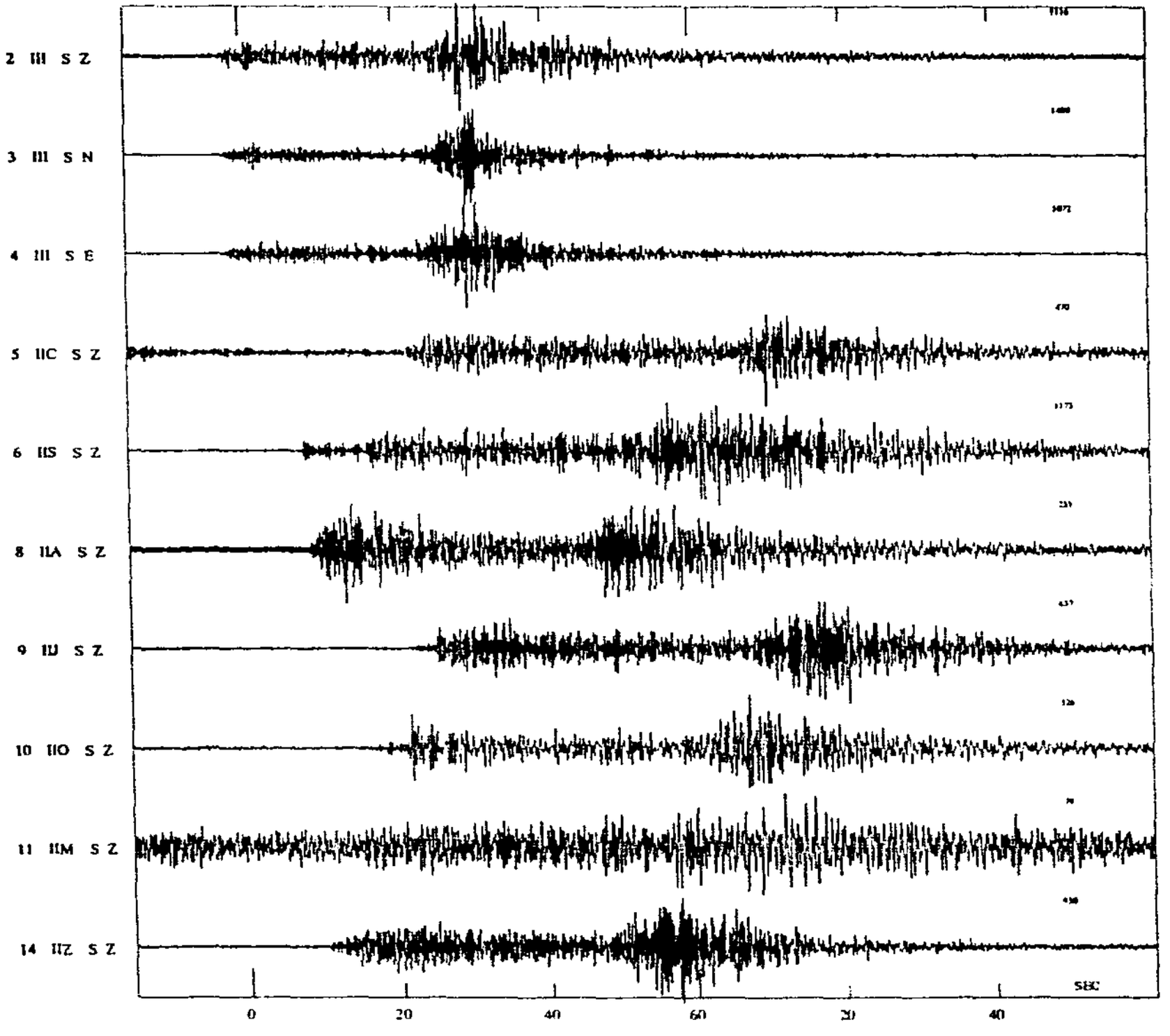
403-03-1420-20S.SMX_16 9403-03-1420-20S.SMX_16

Inicio registro:94 3 3 14:20 20.137



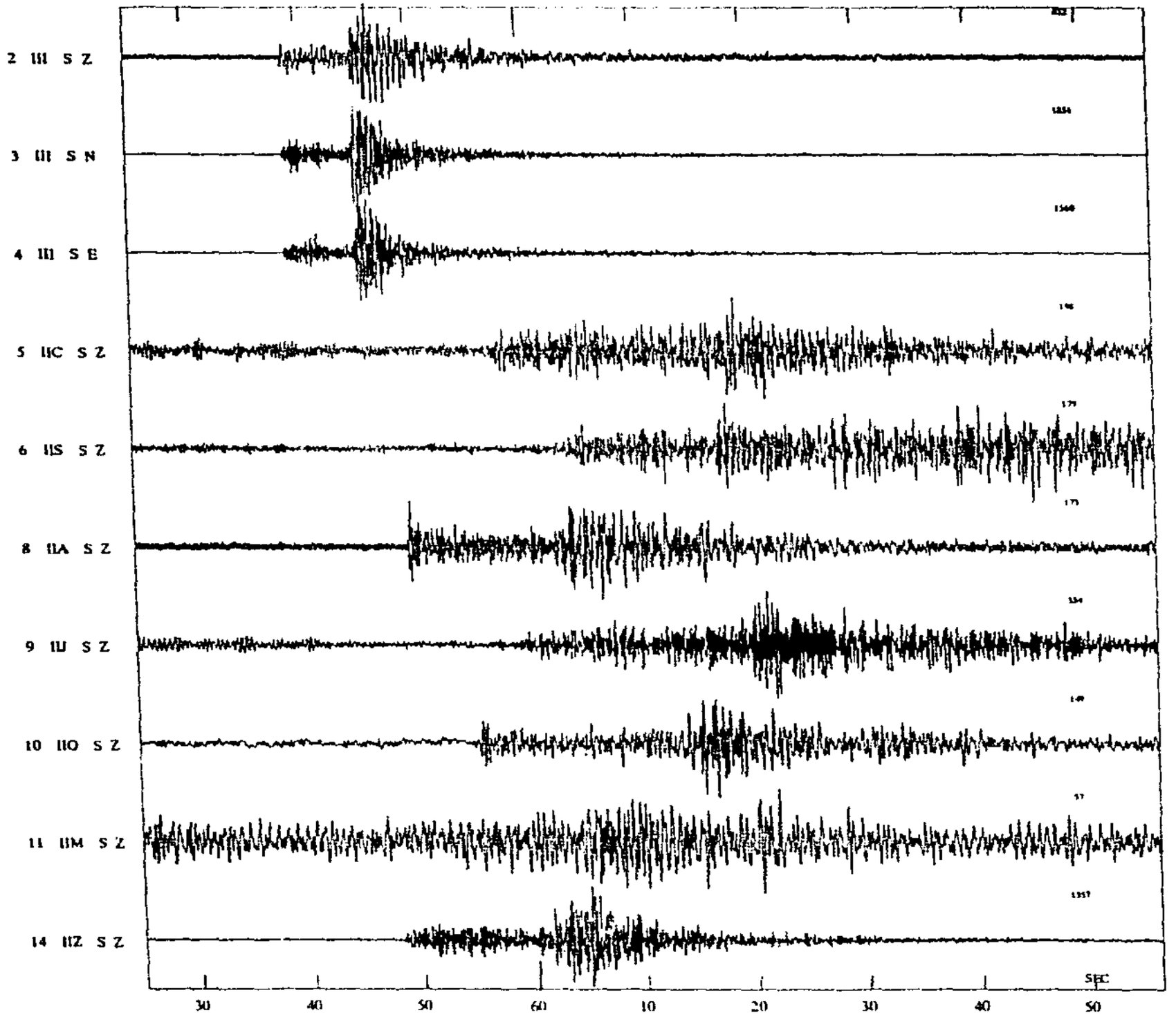
403-19-0421-44S.SMX_16 9403-19-0421-44S.SMX_16

Inicio registro:94 3 19 4:21 44.396



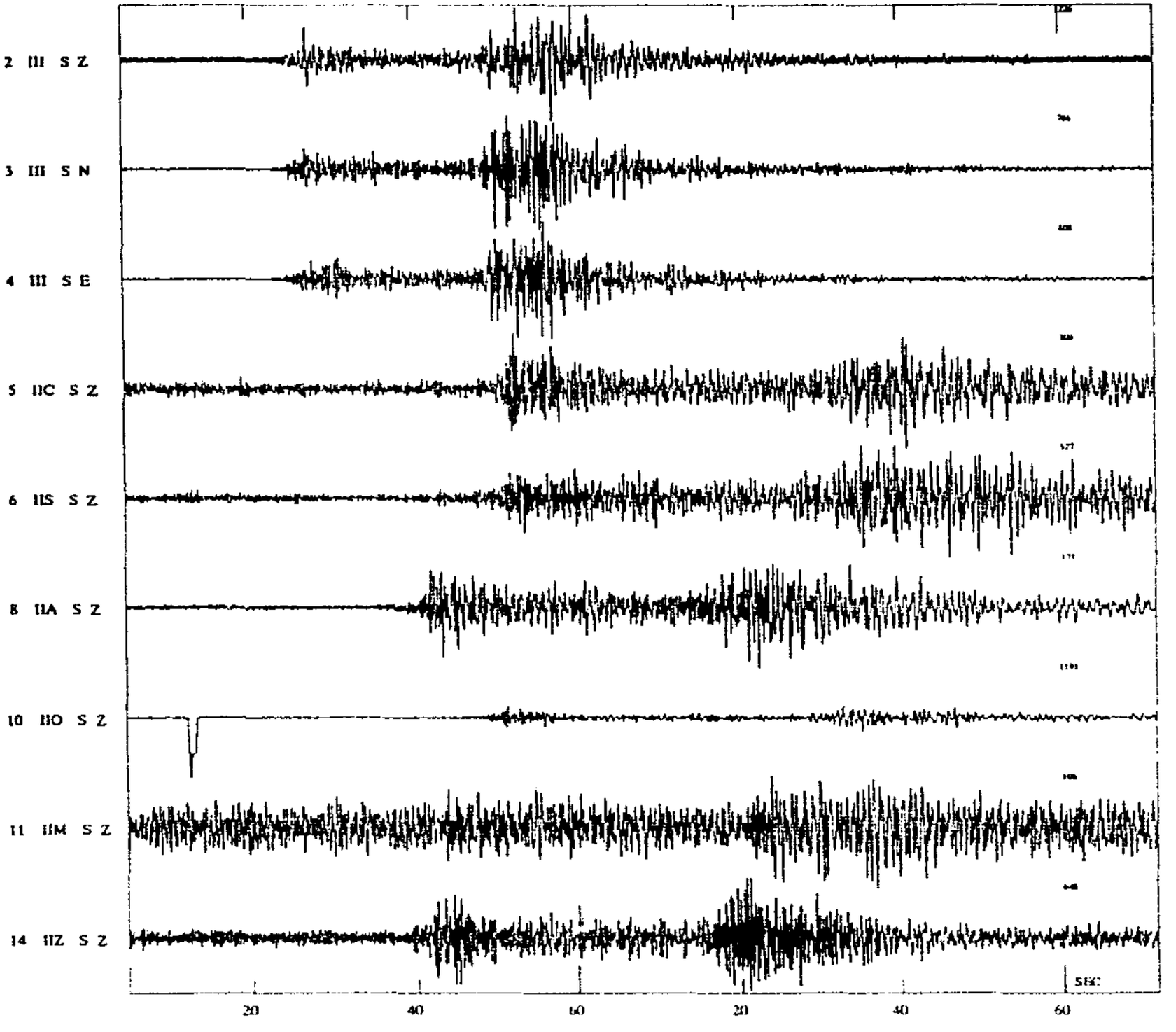
403-20-0506-25S.SMX_16 9403-20-0506-25S.SMX_16

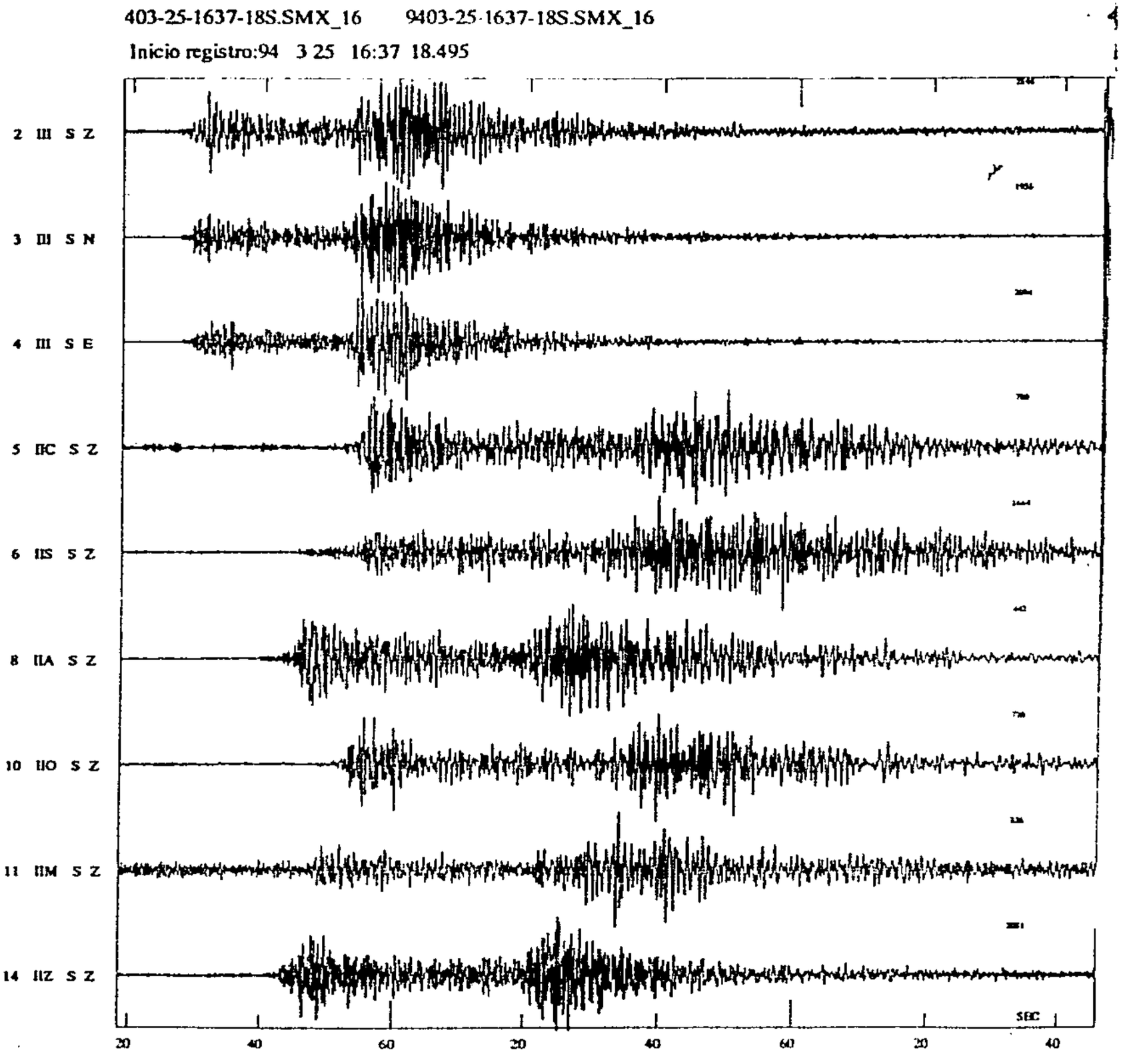
Inicio registro:94 3 20 5: 6 25.032



403-25-1633-04S.SMX_16 9403-25-1633-04S.SMX_16

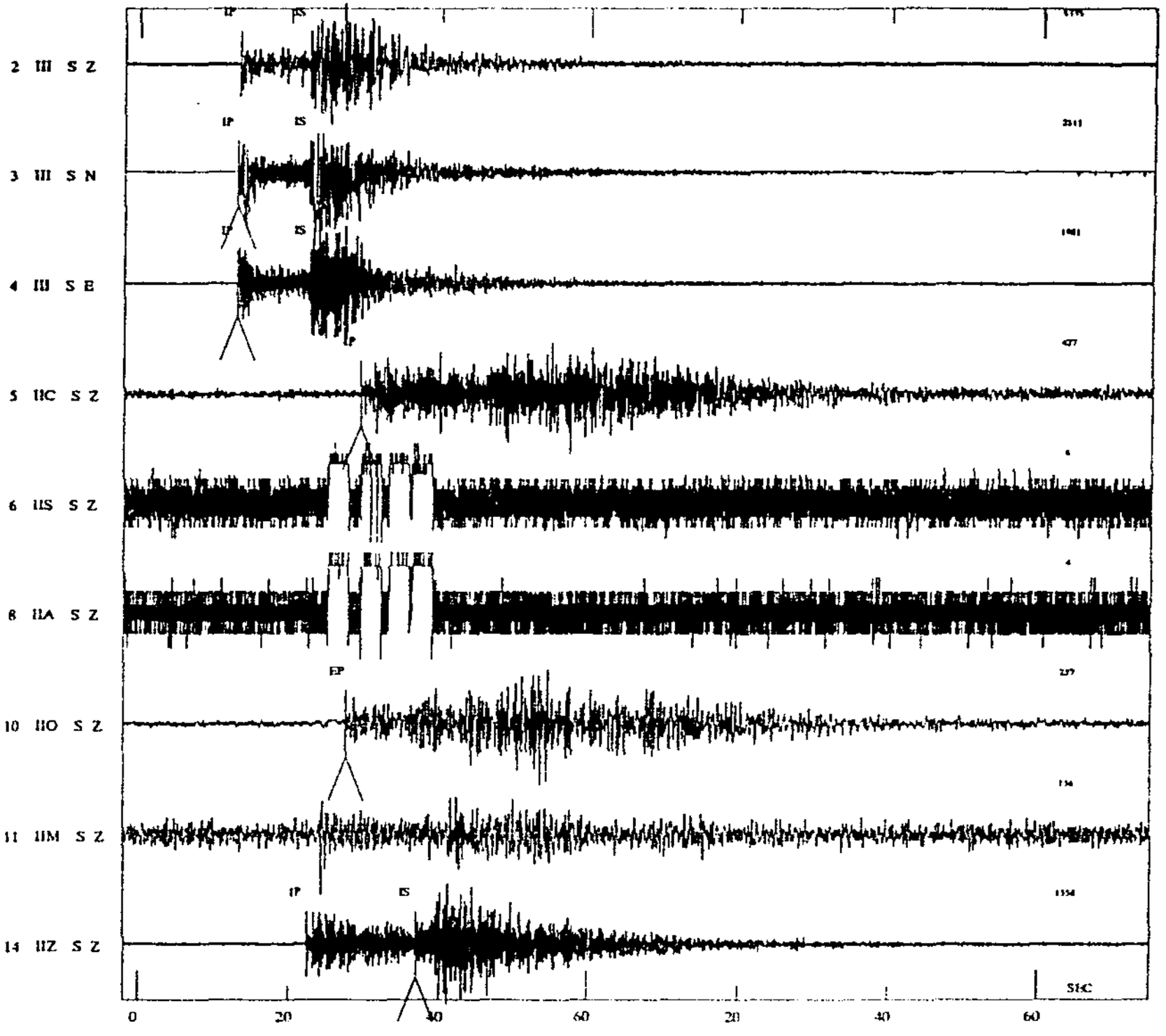
Inicio registro:94 3 25 16:33 4.537




403-25-1637-18S.SMX_16 9403-25-1637-18S.SMX_16
Inicio registro:94 3 25 16:37 18.495
2 III S Z
3 III S N
4 III S E
5 IIC S Z
6 IIS S Z
8 IIA S Z
10 IIO S Z
11 IIM S Z
14 IIZ S Z
SEC
20
40
60
20
40
60
20
40

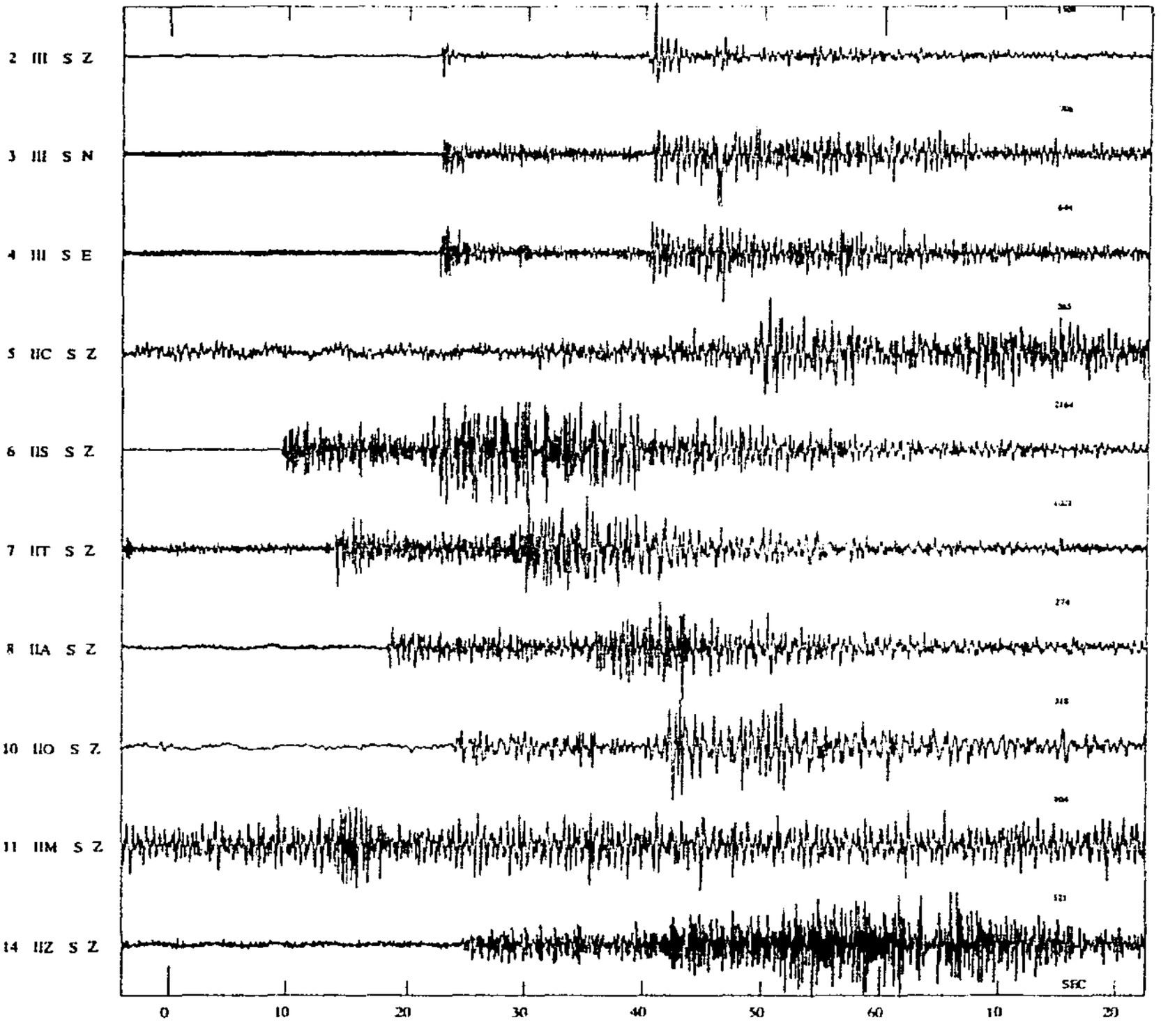
409-17-1916-58S.SMX_16 9409-17-1916-58S.SMX_16

Inicio registro:94 9 17 19:16 58.046



410-04-2002-56S.SMX_16 9410-04-2002-56S.SMX_16

Inicio registro:94 10 4 20: 2 56.141



412-25-0409-28S.SMX_16 9412-25-0409-28S.SMX_16

Inicio registro:94 12 25 4: 9 28.613