

UNIVERSIDAD DON BOSCO



**SISTEMA DE RECONOCIMIENTO DE NOTAS MUSICALES
PARA FLAUTA DULCE SOPRANO**

TRABAJO DE GRADUACIÓN
PREPARADO PARA LA FACULTAD DE INGENIERÍA
PARA OPTAR AL GRADO DE
INGENIERO EN ELECTRÓNICA

POR:

EFRÉN ANTONIO GÓMEZ

FRANZ ALEXEI MENESES OCHOA

SEPTIEMBRE DE 2004

SOYAPANGO — EL SALVADOR — AMÉRICA CENTRAL

UNIVERSIDAD DON BOSCO

RECTOR

ING. FEDERICO MIGUEL HUGUET RIVERA

SECRETARIO GENERAL

LIC. MARIO RAFAEL OLMOS

DECANO DE LA FACULTAD DE INGENIERÍA

ING. ERNESTO GODOFREDO GIRÓN

ASESOR DEL TRABAJO DE GRADUACIÓN

ING. EDUARDO RIVERA

JURADO EVALUADOR

ING. JORGE LÓPEZ

ING. JULIO RIVERA

ING. EDGARDO CRUZ ZELEDÓN

TUTOR DEL TRABAJO DE GRADUACIÓN

ING. WENCESLAO RIVAS

UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA ELECTRÓNICA

JURADO EVALUADOR DEL TRABAJO DE GRADUACIÓN

ING. JORGE LÓPEZ
JURADO

ING. JULIO RIVERA
JURADO

ING. EDGARDO CRUZ ZELEDÓN
JURADO

ING. EDUARDO RIVERA
ASESOR

ING. WENCESLAO RIVAS
TUTOR

AGRADECIMIENTOS

Efrén Antonio Gómez

En primer lugar quiero agradecer a Dios, no solo por darme las habilidades y la oportunidad de realizar satisfactoriamente esta tarea, sino también por la enorme cantidad de bendiciones y dones recibidos.

Quiero agradecer también a mi madre, Margarita Gómez, por su enorme paciencia, comprensión y apoyo incondicional.

Igualmente es mi deseo agradecer a Maricela García por todo el tiempo que me dedicó, por escucharme con genuino interés mientras le comentaba los progresos y dificultades que se presentaban, también por las veces que criticó algunas fallas y me ayudó a corregirlas, y sobre todo por las palabras de ánimo que me inspiraron a seguir adelante, haciendo siempre mi mayor esfuerzo.

Asimismo, a Eduardo Rivera, quien con sus conocimientos, experiencia y dedicación nos brindó una ayuda invaluable para concluir exitosamente este trabajo de graduación.

A todas aquellas personas que, como lo hizo Wilfredo Alfaro, colaboraron brindando sugerencias y opiniones sobre el sistema, especialmente a Gerardo Rosales, quien colaboró en la presentación de la defensa final del proyecto.

También a las personas en cuyas oraciones rogaban por la culminación exitosa del proceso de graduación, puesto que su intercesión fue importante para llevar a feliz término esta ardua tarea.

Franz Alexei Meneses Ochoa.

Quisiera agradecer en primer lugar a Dios por todo lo que me ha permitido vivir y experimentar en el desarrollo de este trabajo, por haber solucionado cuanto problema surgió y por haber permitido que todo saliera bien.

En segundo lugar quiero dar gracias a mis padres Carlos Antonio Meneses y María Lilian Ochoa de Meneses por toda la ayuda brindada. A mi padre porque desde el cielo intercedió para que todo saliera bien y a mi madre por el amor, comprensión y apoyo que me brindó en todo momento y por tenerme en cuenta siempre en sus oraciones.

A mi novia Blanca Duarte, por toda la ayuda y empuje que me dio para sacar este proyecto adelante, por sus oraciones a Dios para que todo este trabajo saliera bien. Por estar siempre ahí conmigo en los momentos difíciles.

A mis hermanos Villiam Erick y Karen Aleyda, a mi tía Sonia Estela por confiar en mí y darme ánimos para salir adelante y en especial por todas sus oraciones.

A mis hermanos de comunidad que en todo momento estuvieron rezando para que este proyecto concluyera de la mejor forma y para que Dios nos diera sabiduría para poder solventar cualquier problema que surgiera.

También quiero agradecer a Ángel Avendaño y Antonio Vides por darme todos los permisos necesarios para poder realizar este trabajo sin ponerme ningún pero, algo que les agradezco mucho.

A las personas que de una u otra forma colaboraron en este proyecto, principalmente en las pruebas finales del mismo, en especial a Gerardo Rosales por habernos colaborado en la defensa del proyecto.

A Eduardo Rivera nuestro asesor, por las horas, conocimiento y exigencias brindadas para que este proyecto resultara exitoso y superara nuestras propias metas.

Por último y nuevamente le agradezco a Dios porque si no fuera por Él nada de esto hubiera sido posible.

ÍNDICE

INTRODUCCIÓN	IX
CAPÍTULO 1. INTRODUCCIÓN A LAS REDES NEURONALES.....	1
1.1 REDES NEURONALES BIOLÓGICAS.....	2
1.2 REDES NEURONALES ARTIFICIALES.....	4
1.2.1 Analogía con las redes neuronales biológicas.....	4
1.3 LA FUNCIÓN DE ACTIVACIÓN	6
1.3.1 Función de umbral.....	6
1.3.2 Función sigmoideal o logística.....	6
1.4 REDES NEURONALES NO COMPETITIVAS.....	7
1.4.1 Red Neuronal Backpropagation (BPN).....	8
1.4.1.1 Estructura de la red Backpropagation.	8
1.4.1.2 Funcionamiento de la Backpropagation.....	9
1.4.1.3 Algoritmo de la BPN.....	10
1.4.1.4 Algoritmo de Entrenamiento. [1].....	11
1.4.1.5 Algoritmo de Aplicación.....	12
1.5 REDES NEURONALES COMPETITIVAS.....	13
1.5.1 Red ART 2.....	13
1.5.1.1 Algoritmo de entrenamiento de la red ART 2. [1].....	15
1.5.1.2 Ejemplo de ejecución del algoritmo de entrenamiento de la red neuronal ART 2.....	18
1.5.2 Kohonen Self-Organizing Maps (SOM).....	20
1.5.2.1 Estructura [1].....	20
1.5.2.2 Algoritmo de la SOM [1].....	21
1.5.2.3 Ejemplo de aplicación de una SOM.....	22
1.5.3 Learning Vector Quantization (LVQ).....	24
1.5.3.1 Estructura. [1].....	24
1.5.3.2 Algoritmo. [1]	25
CAPÍTULO 2. CONCEPTOS MUSICALES	26
2.1 EL PENTAGRAMA [11]	26
2.2 LAS CLAVES [11] [19]	27
2.3 DURACIÓN DE LAS NOTAS MUSICALES.....	28
2.4 ALTERACIONES A LAS NOTAS MUSICALES. [11]	29
2.5 EL COMPÁS. [11]	30
2.5.1 La indicación de compás.....	30

2.5.2	<i>Las líneas divisorias.....</i>	31
2.6	SIGNOS SECUNDARIOS. [11] [12]	31
2.6.1	<i>El puntillo.....</i>	32
2.6.2	<i>La ligadura</i>	32
2.7	TONO Y SEMITONO. [11] [12]	33
2.8	TRANSPOSICIÓN O TRANSPORTE. [11].....	33
CAPÍTULO 3. ADQUISICIÓN Y PROCESAMIENTO DE LA SEÑAL		34
3.1	ADQUISICIÓN DE PATRONES.....	34
3.2	PROCESAMIENTO DE LA SEÑAL.....	35
CAPÍTULO 4. PROCESO DE SELECCIÓN DE LA RED A UTILIZAR.....		37
4.1	RESULTADOS DE PRUEBAS REALIZADAS CON VALORES EN EL TIEMPO.....	37
4.1.1	<i>Red Backpropagation.....</i>	37
4.1.1.1	Resultado de pruebas realizadas con salida Sigmoidal Binaria.	37
4.1.1.2	Resultado de pruebas realizadas con salida Sigmoidal Bipolar.	41
4.1.2	<i>Resultado de las pruebas realizadas con la red ART 2.</i>	42
4.1.3	<i>Resultados obtenidos con la SOM.....</i>	47
4.1.4	<i>Resultados obtenidos con la LVQ.....</i>	49
4.2	RESULTADOS DE PRUEBAS REALIZADAS CON VALORES EN FRECUENCIA.....	51
4.2.1	<i>Red Backpropagation.....</i>	52
4.2.1.1	Resultados obtenidos con la salida de activación sigmoideal.....	52
4.2.1.2	Resultados obtenidos con salida de activación binaria.....	53
4.2.2	<i>Resultados obtenidos con la red ART 2.....</i>	55
4.3	SELECCIÓN DE LA RED.....	56
CAPÍTULO 5. DESCRIPCIÓN DEL FUNCIONAMIENTO.....		57
5.1	EJECUCIÓN DE LA FLAUTA DULCE.....	57
5.2	DIGITALIZACIÓN DE AUDIO.....	58
5.3	RECONOCIMIENTO DE LA NOTA MUSICAL CON RNA	60
5.4	DETERMINACIÓN DEL TIEMPO DE DURACIÓN DE LA NOTA MUSICAL.....	61
5.5	DESARROLLO DEL PENTAGRAMA DE LAS NOTAS EJECUTADAS	65
5.5.1	<i>Generalidades.....</i>	65
5.5.2	<i>Formato de archivo .ptg.....</i>	66
5.5.3	<i>Edición del pentagrama</i>	67
5.5.4	<i>Operación en modo normal.....</i>	68
5.5.5	<i>Operación en modo de aprendizaje supervisado.....</i>	68

CAPÍTULO 6.	MANUAL DE USUARIO SRENOM	69
6.1	PRESENTACIÓN GENERAL.....	69
6.2	BARRA DE HERRAMIENTAS.....	73
6.3	PENTAGRAMA.....	77
6.3.1	<i>Título.....</i>	77
6.3.2	<i>Pauta o Pentagrama</i>	77
6.4	BARRA DE ESTADO.....	78
6.5	CREAR UN NUEVO PENTAGRAMA.....	78
6.6	CAMBIAR EL TÍTULO DE UN PENTAGRAMA.....	79
6.7	CREAR UN PENTAGRAMA A PARTIR DE LAS NOTAS EJECUTADAS.....	80
6.8	REALIZAR TRANSPOSICIÓN DE NOTAS.....	81
6.9	REGULAR LA TOLERANCIA AL RUIDO.....	82
6.10	EDITAR LAS NOTAS EN FORMA MANUAL.....	83
6.11	UTILIZAR EL MODO DE APRENDIZAJE SUPERVISADO.....	85
6.12	REQUERIMIENTOS DEL SISTEMA.....	86
6.12.1	<i>Requerimientos de hardware y software.....</i>	86
6.12.2	<i>Características del sistema.....</i>	87
6.13	PROCESO DE INSTALACIÓN DEL SISTEMA.....	88
6.14	PRUEBAS DEL SISTEMA.....	90
CAPÍTULO 7.	CONCLUSIONES	91
CAPÍTULO 8.	RECOMENDACIONES	93
CAPÍTULO 9.	BIBLIOGRAFÍA	97
ANEXOS		99
A. CÓDIGO FUENTE.....		100
B. TABLAS ESTADÍSTICAS		150

INTRODUCCIÓN

En el presente documento se muestra el estudio y desarrollo seguido para la elaboración de un sistema para reconocimiento de notas musicales emitidas por la flauta dulce soprano, denominado SiReNoM. Para poder presentarlo, éste ha sido dividido en ocho capítulos con el objetivo de facilitar la lectura del mismo.

En el primer capítulo se hace una breve descripción de lo que son las redes neuronales artificiales (RNA), las cuales son el eje central del proyecto, las similitudes que tienen con las redes neuronales biológicas, los tipos de RNA que se utilizaron, sus características así como sus algoritmos de entrenamiento y aplicación.

En el segundo capítulo se desarrolla un breve resumen de conceptos musicales y solfa, necesarios para entender las facilidades y utilidades que brinda el sistema.

En el tercer capítulo se muestran los procesos seguidos para la adquisición y procesamiento de las señales de audio, es decir las notas emitidas por la flauta dulce soprano.

En el capítulo cuatro se presenta todos los procesos que se realizaron para poder determinar cual red neuronal era la que mejor desempeño tenía y por ende cual red sería la adecuada para desarrollar el sistema, para que este tuviera un grado alto de efectividad en su desempeño.

La descripción del funcionamiento del sistema y cada una de las partes que lo componen se describen en el capítulo cinco, estas partes son principalmente el procesamiento del audio, el reconocimiento de las notas, la determinación del tiempo de duración de cada una de ellas y la elaboración y manejo del pentagrama.

En el capítulo seis se presenta el Manual de Usuario para SiReNoM, con todas las opciones de operación del mismo, los requerimientos así como las características y limitaciones que tiene. También se presentan los pasos a seguir para la instalación del sistema, además se

incluyen las pruebas que se hicieron para ver su respuesta con diferentes usuarios y diferentes localidades.

En el capítulo siete están desarrolladas las conclusiones a las cuales llegamos después de haber finalizado el sistema y haber hecho todas las pruebas necesarias con él, para saber su efectividad y funcionalidad.

En el capítulo nueve se presentan las recomendaciones para poder realizarle mejoras a SiReNoM, para que pueda ser desarrollado como una herramienta más potente para el aprendizaje de no solo la flauta dulce soprano sino también de la música en general así como de otros instrumentos musicales.

Para finalizar, el lector podrá encontrar en los anexos el código fuente para el desarrollo de SiReNoM así como de tablas y gráficas utilizadas en el proceso de desarrollo del mismo.

Capítulo 1. INTRODUCCIÓN A LAS REDES NEURONALES.

El cerebro es un procesador de información con unas características muy notables: es capaz de procesar a gran velocidad grandes cantidades de información procedentes de los sentidos, combinarla o compararla con la información almacenada y dar respuestas adecuadas incluso en situaciones nuevas. Pero lo más impresionante de todo es su capacidad de aprender a representar la información necesaria para desarrollar tales habilidades sin instrucciones explícitas para ello.

Aunque todavía no se sabe mucho sobre la forma en que el cerebro aprende a procesar la información, se han desarrollado modelos que tratan de imitar tales habilidades denominados *redes neuronales artificiales*. La elaboración de estos modelos supone en primer lugar la deducción de los rasgos o características esenciales de las neuronas y sus conexiones, y en segundo lugar, la implementación del modelo en una computadora de forma que se pueda simular. Es obvio decir que estos modelos son idealizaciones burdas de las auténticas redes neuronales, en muchos casos de dudosa plausibilidad neurofisiológica, pero que sin embargo resultan interesantes cuando menos por sus capacidades de aprendizaje.

El progreso de las neurociencias nos está conduciendo a una comprensión cada vez mayor de la estructura física y lógica del cerebro; los avances tecnológicos ofrecen recursos cada vez mayores para representar estructuras muy complejas, realizar cálculos a gran velocidad y en paralelo, apoyando y fomentando así la investigación en este campo. Podríamos situar el origen de los modelos conexionistas con la definición de la neurona formal dada por McCulloch y Pitts en 1943 como un dispositivo binario con varias entradas y salidas. Un psicólogo, D. O. Hebb, introdujo en 1949 (Hebb 1949) dos ideas fundamentales que han influido de manera decisiva en el campo de las redes neuronales: la idea de que una

percepción o un concepto se representa en el cerebro por un conjunto de neuronas activas simultáneamente; y la idea de que la memoria se localiza en las conexiones entre las neuronas (sinápsis). Las hipótesis de Hebb, basadas en investigaciones psicofisiológicas, presentan de manera intuitiva el modo en que las neuronas memorizan información, y se plasman sintéticamente en la famosa regla de aprendizaje de Hebb (también conocida como regla del producto). Esta regla indica que las conexiones entre dos neuronas se refuerzan si ambas son activadas. Muchos de los algoritmos actuales proceden de los conceptos de este psicólogo, y a pesar de las críticas recibidas, como la existencia de conexiones inhibitorias y no sólo excitatorias, sigue teniendo una gran influencia. [14].

1.1 Redes Neuronales Biológicas

A grandes rasgos, podemos afirmar que el cerebro humano se compone de decenas de billones de neuronas interconectadas entre sí formando circuitos o redes que desarrollan funciones específicas. Una neurona típica recoge señales procedentes de otras neuronas a través de un grupo de delicadas estructuras llamadas dendritas. La neurona emite impulsos de actividad eléctrica a lo largo de una fibra larga y delgada denominada axón, que se divide en millares de ramificaciones.

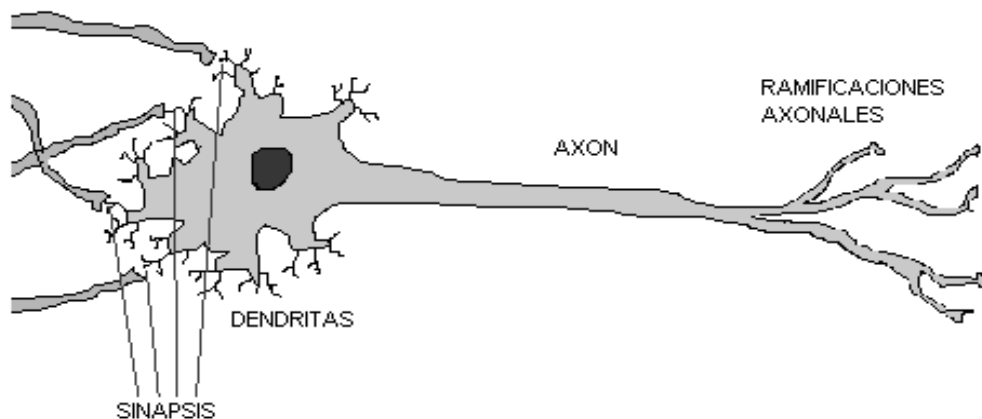


Figura 1-1. Neurona y conexiones sinápticas

Las extremidades de estas ramificaciones llegan hasta las dendritas de otras neuronas y establecen unas conexiones llamadas **sinápsis**, en las cuales se produce una transformación de los impulsos eléctricos en un mensaje neuro-químico mediante la liberación de unas sustancias llamadas neurotransmisores.

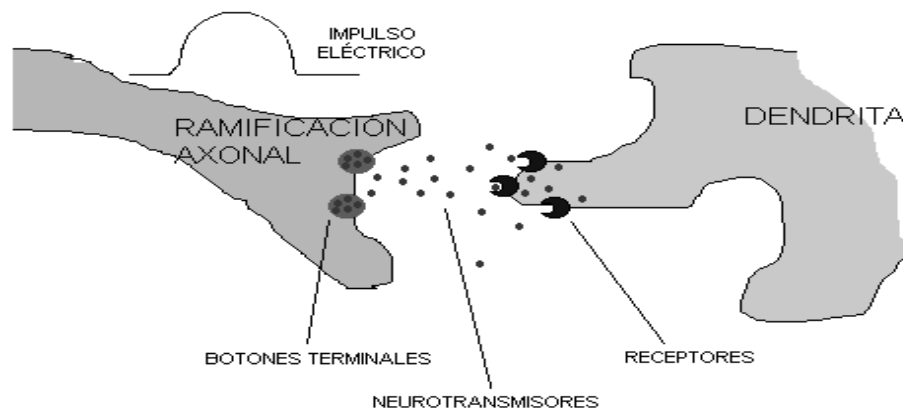


Figura 1-2. Detalle de una sinápsis

El efecto de los neurotransmisores sobre la neurona receptora puede ser excitatorio o inhibitorio, y es variable, de manera que podemos hablar de la fuerza o efectividad de una sinápsis. Las señales excitatorias e inhibitorias recibidas por una neurona se combinan, y en función de la estimulación total recibida, la neurona toma un cierto nivel de activación, que se traduce en la generación de breves impulsos nerviosos con una determinada frecuencia o tasa de disparo y su propagación a lo largo del axón hacia las neuronas con las cuales hace sinápsis.

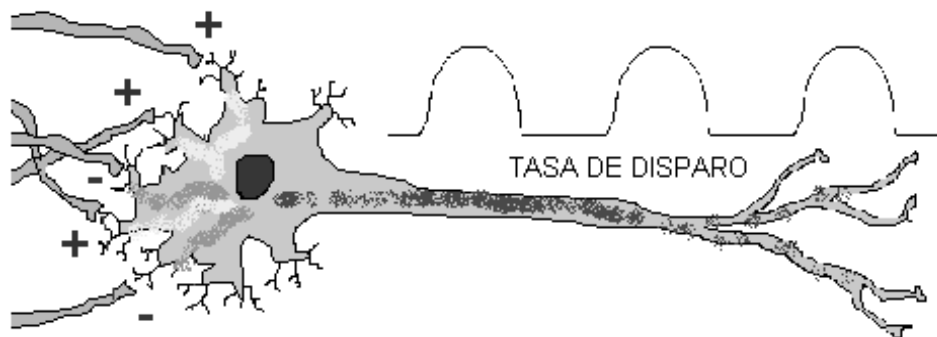


Figura 1-3. Activación y disparo de una neurona

De esta manera la información se transmite de unas neuronas a otras y va siendo procesada a través de las conexiones sinápticas y las propias neuronas. El aprendizaje de las redes neuronales se produce mediante la variación de la efectividad de las sinápsis, de esta manera cambia la influencia que unas neuronas ejercen sobre otras, de aquí se deduce que la arquitectura, el tipo y la efectividad de las conexiones en un momento dado, representan en cierto modo la memoria o estado de conocimiento de la red. [14].

1.2 Redes Neuronales Artificiales

1.2.1 Analogía con las redes neuronales biológicas

Las neuronas se modelan mediante unidades de proceso. Cada unidad de proceso se compone de una red de conexiones de entrada, una función de red, (de propagación), encargada de calcular la entrada total combinada de todas las conexiones, un núcleo central de proceso, encargado de aplicar la función de activación, y la salida, por dónde se transmite el valor de activación a otras unidades.

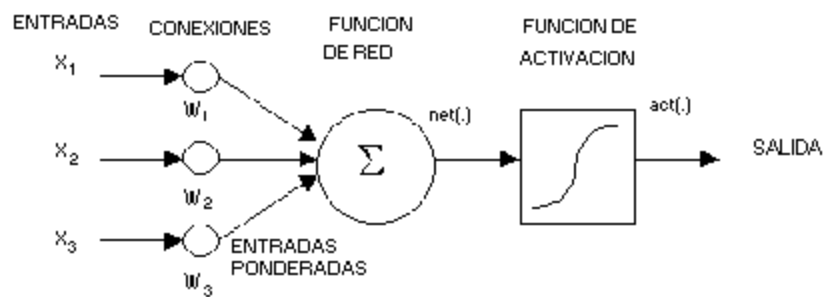


Figura 1-4. Unidad de proceso típica

La función de red es típicamente la suma ponderada, mientras que la función de activación suele ser alguna función de umbral o una función sigmoide.

- **Función de propagación o de red:** Calcula el valor de base o entrada total a la unidad, generalmente como simple suma ponderada de todas las entradas recibidas, es decir, de las entradas multiplicadas por el peso o valor de las conexiones. Equivale a la combinación de las señales excitatorias e inhibitorias de las neuronas biológicas.
- **Función de activación:** Es quizás la característica principal o definitoria de las neuronas, la que mejor define el comportamiento de la misma. Se usan diferentes tipos de funciones, desde simples funciones simples de umbral a funciones no lineales. Se encarga de calcular el nivel o estado de activación de la neurona en función de la entrada total.
- **Conexiones ponderadas:** hacen el papel de las conexiones sinápticas, el peso de la conexión equivale a la fuerza o efectividad de la sinápsis. La existencia de conexiones determina si es posible que una unidad influya sobre otra, el valor de los pesos y el signo de los mismos definen el tipo (excitatorio / inhibitorio) y la intensidad de la influencia.
- **Salida:** calcula la salida de la neurona en función de la activación de la misma, aunque normalmente no se aplica más que la función identidad, y se toma como salida el valor de activación. El valor de salida cumpliría la función de la tasa de disparo en las neuronas biológicas.

Redes Neuronales Biológicas	Redes Neuronales Artificiales
Neuronas	Unidades de proceso
Conexiones sinápticas	Conexiones ponderadas
Efectividad de las sinapsis	Peso de las conexiones
Efecto excitatorio o inhibitorio de una conexión	Signo del peso de una conexión
Efecto combinado de las sinapsis	Función de propagación o de red
Activación - - tasa de disparo	Función de activación - - Salida

Tabla 1-1 Comparación entre las neuronas biológicas reales y las unidades de proceso artificiales

1.3 La Función De Activación

Se suele distinguir entre funciones lineales, en las que la salida es proporcional a la entrada, funciones de umbral, en las cuales la salida es un valor discreto (típicamente binario 0/1) que depende de si la estimulación total supera o no un determinado valor de umbral; y funciones no lineales, donde las salidas no son proporcionales a la entrada.

1.3.1 Función de umbral

En un principio se pensó que las neuronas usaban una función de umbral, es decir, que permanecían inactivas y se activaban sólo si la estimulación total superaba cierto valor límite; esto se puede modelar con una función escalón: la más típica es el escalón unitario: la función devuelve 0 por debajo del valor crítico (umbral) y 1 por encima. Después se comprobó que las neuronas emitían impulsos de actividad eléctrica con una frecuencia variable, dependiendo de la intensidad de la estimulación recibida, y que tenían cierta actividad hasta en reposo, con estimulación nula. Estos descubrimientos llevaron al uso de funciones no lineales con esas características, como la función sigmoideal, con un perfil parecido al escalón de una función de umbral pero continuo.

1.3.2 Función sigmoideal o logística

Es probablemente la función de activación más empleada en la actualidad.

$$act_j(net_j) = \frac{1}{(1 + e^{-net_j/\sigma})}$$

Se trata de una función continua no lineal con bastante plausibilidad fisiológica. La función sigmoideal posee un rango comprendido entre 0 y 1. Esto, aplicado a las unidades de proceso de una red neuronal artificial significa que, sea cual sea la entrada, la salida estará comprendida entre 0 y 1. Esta función depende del parámetro σ , que usualmente toma el valor 1.

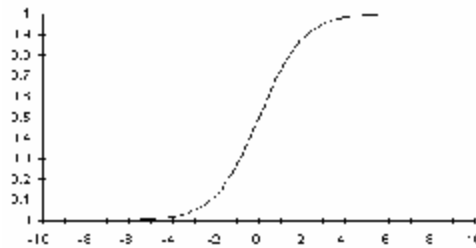


Figura 1-5. Función sigmoideal con $s = 1$

La salida de una unidad vale 0.5 cuando la entrada es nula, esto significa que la unidad tiene cierta actividad aún en ausencia de estimulación. Al aumentar la estimulación la unidad aumenta su activación, y la disminuye si la estimulación es inhibitoria, de forma parecida a como se comportan las neuronas reales.

Presenta las siguientes características deseables.

1. Acomodación de señales muy intensas sin producir saturación.
2. Admite señales débiles sin excesiva atenuación
3. Fácilmente derivable, ya que $f'(x) = f(x) \cdot (1 - f(x))$

La principal limitación de esta función es que no sirve para expresar polaridades, da siempre valores positivos. Una función alternativa con cualidades parecidas pero con un rango entre -1 y 1 es la función sigmoideal bipolar. Desde un punto de vista fisiológico, el signo negativo se puede interpretar como una disminución de la tasa de disparo de una neurona por debajo de la tasa de disparo en reposo. [14].

1.4 Redes Neuronales No Competitivas

De las diferentes topologías existentes entre las redes no competitivas se buscó una que fuese del tipo heteroasociativa, es decir que su salida consistiese en un patrón diferente al de la entrada, la que presentaba mayores posibilidades en cuanto a capacidad de aprendizaje fue la red Backpropagation, gracias a su enorme habilidad de representar casi cualquier tipo de función.

1.4.1 Red Neuronal Backpropagation (BPN)

1.4.1.1 Estructura de la red Backpropagation.

Una red backpropagation (BPN de sus siglas en ingles Backpropagation Network) es una red multicapa y está formada por tres grupos básicos de neuronas: Entrada, Ocultas y Salida. Las neuronas de entrada están directamente conectadas a las neuronas ocultas, las neuronas ocultas pueden ser de varias capas, las cuales están conectadas a las neuronas de salida.

Nomenclatura:

- X_n : Unidades de entrada.
- V_{0m} : Unidades de tendencia o umbral de la capa oculta
- V_{nm} : Pesos de conexión entre capa de entrada y capa oculta
- Z_m : Unidades de capa oculta
- W_{0k} : Unidades de tendencia o umbral de la capa de salida
- V_{mk} : Pesos de conexión entre capa oculta y la capa de salida
- Y_k : Unidades de capa de salida

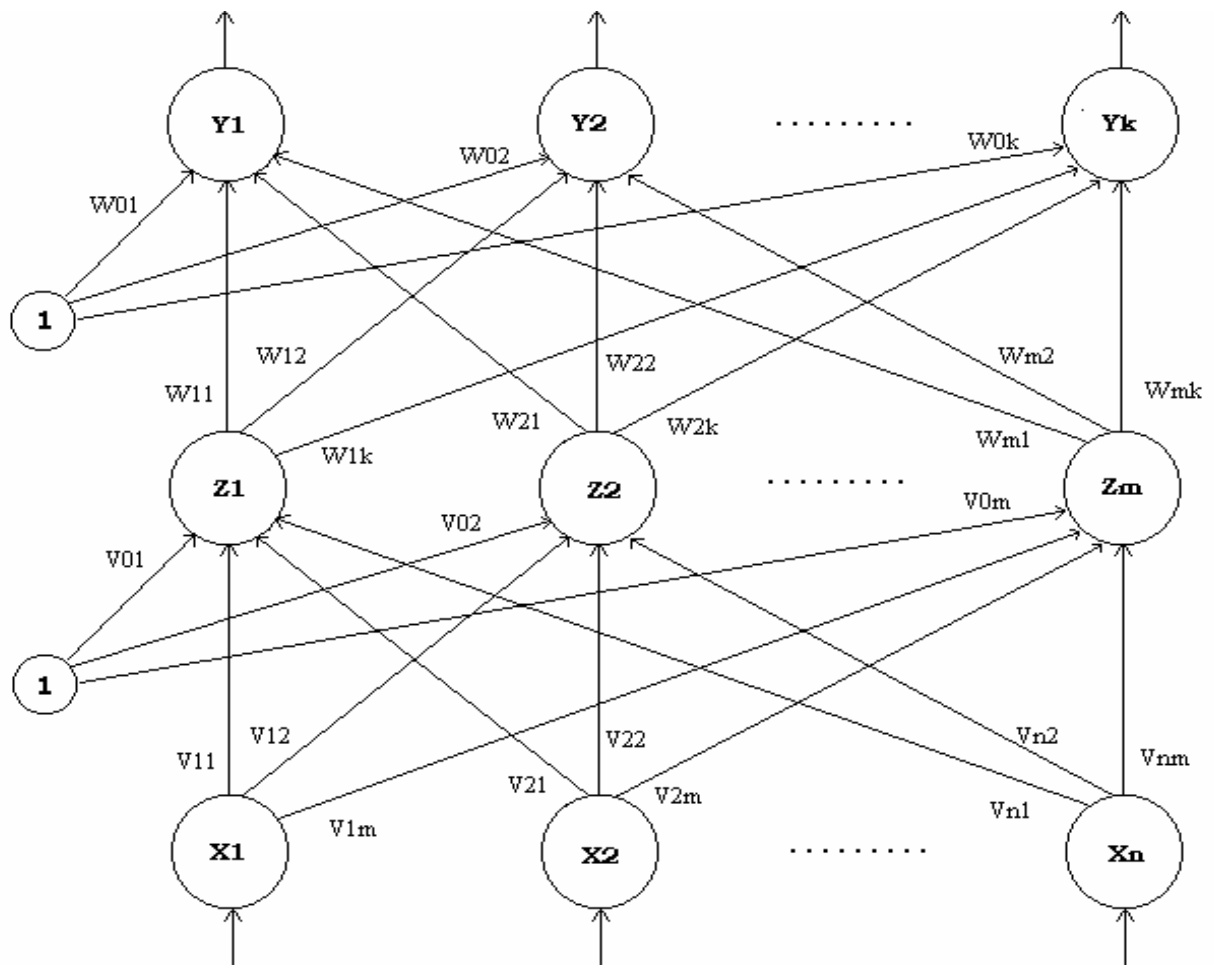


Figura 1-6 Estructura de la red Backpropagation

Para la conexión de las neuronas se utilizan los pesos, también se tienen las unidades de tendencia o umbral, pero que también se conectan a las neuronas ocultas y de salida con un peso que influye en el proceso de la red. En la figura 1-6 se presenta un diagrama de la estructura de la BPN en una forma muy general. [1]

1.4.1.2 Funcionamiento de la Backpropagation.

La red backpropagation aprende un grupo predeterminado de entradas y salidas deseadas para esas entradas, utilizando un ciclo de propagación-adaptación de tres fases: la

propagación hacia delante del patrón de entrada, el cálculo y propagación hacia atrás del error asociado y por último el ajuste de los pesos de cada capa.

Para realizar el proceso de entrenamiento de la BPN, primero se aplica el vector de entrada a la primera capa de la red (neuronas de entrada), éste se va propagando hacia adelante por todas las capas de la red hasta generar una salida. Esta señal de salida se compara con la salida deseada y se calcula una señal de error para cada unidad de salida.

La señal de error obtenida se transmite hacia atrás, partiendo de la capa de salida hacia todas las neuronas de la capa oculta, es de hacer notar que el error no se transmite igual a todas las neuronas de la capa oculta sino solamente una fracción del error, en base a la contribución que cada neurona tiene con el error total de salida. Así mismo, las neuronas de la capa oculta transmiten una fracción del error a las neuronas de entrada. Con estos errores transmitidos a cada neurona de cada capa se actualizan los pesos de las conexiones entre capas y se repite nuevamente el proceso para otro patrón de entrada hasta que el error total sea lo suficientemente pequeño.

1.4.1.3 Algoritmo de la BPN

Como se mencionó, el entrenamiento de la BPN incluye tres etapas o fases, durante la primera etapa cada unidad de entrada X_i recibe una señal de entrada y propaga la señal a cada una de las neuronas ocultas ($Z_1... Z_m$). Cada unidad de la capa oculta calcula su activación y envía su señal de salida (Z_m) a cada neurona de la capa de salida (Y_k) las cuales calculan su salida (Y_k) para obtener la respuesta de la red para el patrón de entrada.

En la propagación hacia atrás del error, cada unidad de la capa de salida compara su salida (Y_k) con la salida deseada (T_k) para obtener el error asociado de cada neurona con ese patrón de entrada. Basados en ese error se calcula el factor d_j ($j = 1, 2... k$), este factor es usado para distribuir el error de la unidad de salida a todas las unidades de la capa anterior (capa oculta) y también para calcular los nuevos pesos entre la capa de salida y la capa oculta (W_{mk}). De igual forma se calcula el factor d_c ($c = 1, 2... m$) para cada unidad de la capa oculta. Este factor ya no se propaga a la capa de entrada pero sí se utiliza para calcular los nuevos pesos entre la capa oculta y la capa de entrada (V_{nm}).

1.4.1.4 Algoritmo de Entrenamiento. [1]

Paso0. Inicializar Pesos (W_{ij} , V_{ij} , W_{bias0j} , V_{bias0j})

Paso1. Mientras (Error Cuadrado > Error mínimo) hacer pasos del 2 al 11

Paso2. Para cada patrón de entrada hacer pasos del 3 al 10

Propagación hacia delante:

Paso3. Introducir patrón de entrada

Paso4. Calcular entrada de cada neurona de la capa oculta

$$z_in_j = v_{0j} + \sum_{i=1}^n x_i v_{ij}$$

Ecuación 1-1

Paso5. Calcular la salida para cada neurona de la capa oculta

$$z_j = f(z_in_j)$$

Ecuación 1-2

Paso6. Calcular la entrada de cada neurona de la capa de salida

$$y_in_k = w_{0k} + \sum_{j=1}^k z_j w_{jk}$$

Ecuación 1-3

Paso7. Calcular la salida para cada neurona de la capa de salida

$$y_k = f(y_in_k)$$

Ecuación 1-4

Propagación hacia atrás del error:

Paso8. Calcular factor d_k para cada neurona de la capa de salida

$$d_k = (t_k - y_k) f'(y_in_k)$$

Ecuación 1-5

Paso9. Calcular el factor de error propagado a cada neurona de la capa oculta

$$d_in_j = \sum_{k=1}^m d_k w_{jk}$$

Ecuación 1-6

$$d_j = d_in_j (f'(z_in_j))$$

Ecuación 1-7

Actualización de pesos:

Paso10. Actualizar pesos de capa de salida y de capa oculta

$$W_{jk}(t+1) = W_{jk}(t) + \mathbf{ad}_k z_j \quad \text{Ecuación 1-8}$$

$$Wbias_{0k}(t+1) = Wbias_{0k}(t) + \mathbf{ad}_k \quad \text{Ecuación 1-9}$$

$$V_{ij}(t+1) = v_{ij}(t) + \mathbf{ad}_j x_i \quad \text{Ecuación 1-10}$$

$$Vbias_{0j}(t+1) = Vbias_{0j}(t) + \mathbf{ad}_j \quad \text{Ecuación 1-11}$$

Paso11. Verificar condición de parada

1.4.1.5 Algoritmo de Aplicación

Para utilizar una BPN después de que esta ha sido entrenada solamente se hace uso de la etapa de propagación hacia adelante, la misma que se usó en el proceso de entrenamiento.

Paso0. Para cada patrón de entrada hacer pasos del 1 al 4

Paso1. Calcular entrada de cada neurona de la capa oculta

$$z_in_j = v_{0j} + \sum_{i=1}^n x_i v_{ij} \quad \text{Ecuación 1-12}$$

Paso2. Calcular la salida para cada neurona de la capa oculta

$$z_j = f(z_in_j) \quad \text{Ecuación 1-13}$$

Paso3. Calcular la entrada de cada neurona de la capa de salida

$$y_in_k = w_{0k} + \sum_{j=1}^k z_j w_{jk} \quad \text{Ecuación 1-14}$$

Paso4. Calcular la salida para cada neurona de la capa de salida

$$y_k = f(y_in_k) \quad \text{Ecuación 1-15}$$

1.5 Redes Neuronales Competitivas

Se buscó también utilizar este tipo de redes para ver si se obtenían mejores resultados para nuestra aplicación. Se utilizaron tres tipos de redes competitivas que son las siguientes: *ADAPTIVE RESONANCE THEORY (ART 2)*, *SELF ORGANIZING MAPS (SOM)* y *LEARNING VECTOR QUANTIZATION (LVQ)*. Sus algoritmos y los procesos de entrenamiento se muestran en los siguientes apartados.

1.5.1 Red ART 2.

La red ART 2 es una red heteroasociativa de aprendizaje no supervisado que acepta valores continuos en su entrada. A fin de conseguir clasificar patrones de valor continuo debe realizar una serie de cálculos que vuelven más complejo el algoritmo de entrenamiento. Para comenzar la capa de entrada es una combinación de funciones de normalización y supresión de ruido.

La arquitectura típica de la red ART 2 se muestra en la figura 1-7. La capa F1 esta formada por seis tipos de unidades (W, X, U, V, P y Q) de modo que por cada una de las entradas de la red hay seis unidades involucradas en la generación de las señales enviadas a la siguiente capa. Un elemento adicional entre W y X recibe señales de todas las unidades W y calcula la norma o magnitud del vector \mathbf{w} y envía ese resultado a todas la unidades de la subcapa X. Mecanismos similares de normalización se encuentran entre las unidades $P \rightarrow Q$ y $V \rightarrow U$. Las unidades P actúan como interfaces hacia la capa F2 de esta red, y cada una de las unidades P_i está conectada a todas las neuronas de salida por medio de dos conexiones de pesos, una ascendente (bottom \rightarrow up) y una descendente (top \rightarrow down) y es en estas conexiones dobles que se almacenan los prototipos de cada una de los grupos formados durante el funcionamiento de la red. La acción de la capa F2 es de competencia del tipo *ganador-toma-todo* es decir que solamente una de las neuronas de salida (la ganadora) tendrá su salida activa a la vez.

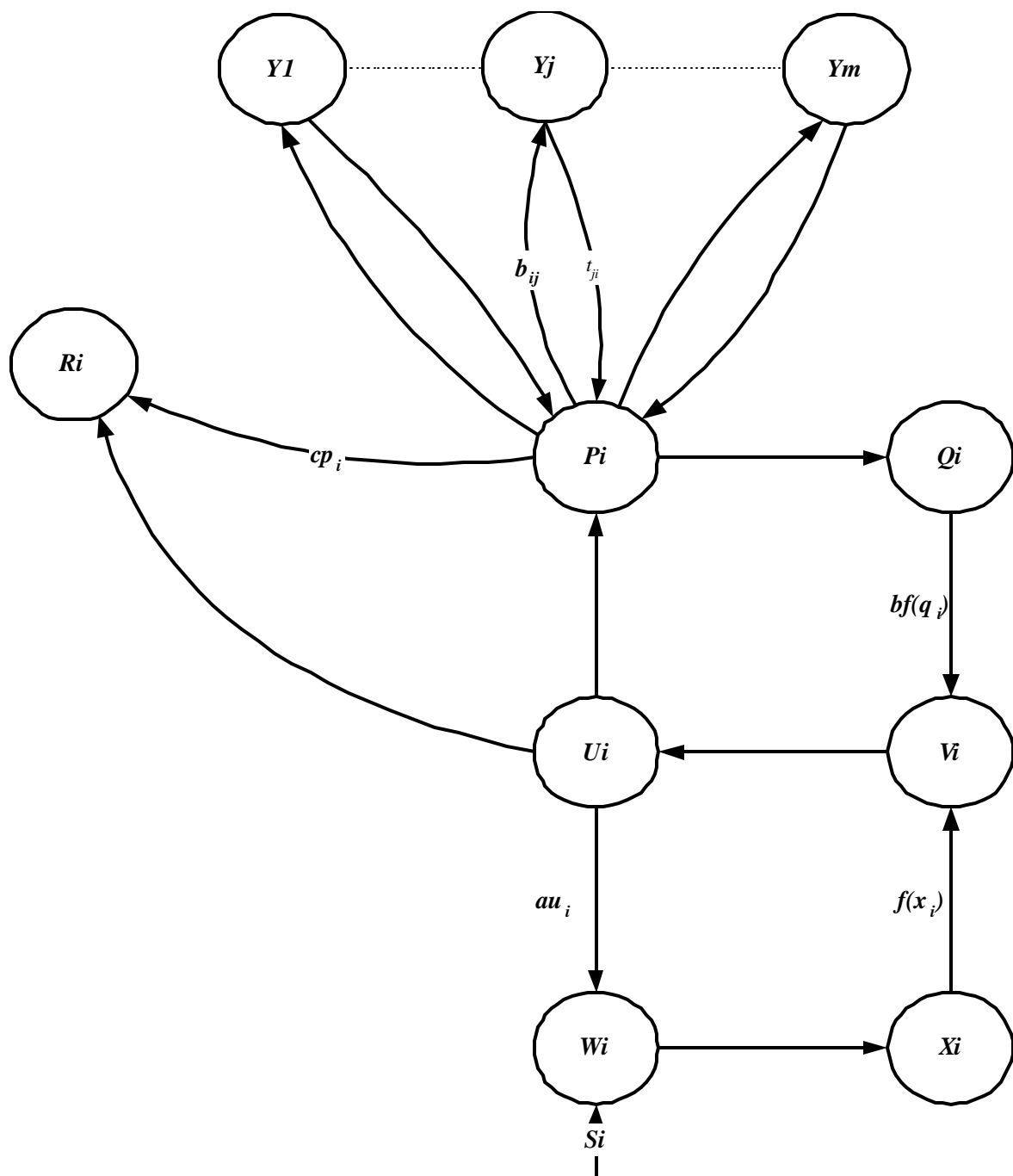


Figura 1-7 Estructura de la ART 2

1.5.1.1 Algoritmo de entrenamiento de la red ART 2. [1]

El ciclo de aprendizaje para un patrón de entrada inicia con el cálculo de la función de activación de la subcapa U (que no es más que la salida normalizada de la subcapa V). Cada elemento U_i envía una señal a sus respectivos elementos W_i y P_i , el nivel de activación de estas unidades es entonces calculado. Los elementos W_i suman la señal recibida de U_i y se S_i (la señal de entrada). P_i suma las señales provenientes de U_i y la señal descendente generada en caso de hallarse activa alguna de las neuronas de la capa F2. La activación de X_i y Q_i son, respectivamente, las versiones normalizadas de W_i y P_i . Una función de activación es aplicada a estas unidades y el resultado es enviado a las unidades V_i , donde es realizada la adición de las señales provenientes de la subcapa X y Q, con lo que se completa el ciclo de actualización de la capa F1.

La función de activación utilizada en todas las pruebas realizadas fue la siguiente:

$$f(x) = \begin{cases} x & ; \Leftrightarrow x \geq q \\ 0 & ; \Leftrightarrow x < q \end{cases} \quad \text{Ecuación 1-16}$$

La razón que justifica el uso de esa función es que, de acuerdo a los creadores de esta red (Carpenter & Grossbreg 1987), las unidades de las capas U y P alcanzan el equilibrio después de tan solo 2 ciclos de actualización de la capa F1, y es además la recomendada por los autores que han realizado publicaciones sobre el tema (cf. Fausett 1994, Hilera)

Una vez que la capa F1 ha alcanzado el equilibrio, las unidades P envían sus señales a la capa F2, donde tiene lugar la competencia para escoger al candidato a aprender el patrón de entrada. También las unidades de las subcapas U y P envían señales hacia las unidades del bloque de control de reset y es en base a la respuesta de ese bloque que la neurona de salida de un grupo o cluster puede ser aceptada o rechazada como no válida por presentar un poco grado de semejanza entre los patrones. Si ese es el caso la unidad rechazada es inhibida, (es decir, su salida es fijada en -1 para excluirla de futuras competencias) y la unidad que tenga la siguiente salida más grande será propuesta como candidata a aprender el patrón de entrada. Ese proceso se repite hasta que el patrón es admitido en un clúster o cuando ya se ha agotado la cantidad máxima de clusters permitidos. El aprendizaje ocurre únicamente si un patrón es aceptado como parte de un clúster de la red.

Descrito en forma más clara el proceso de aprendizaje de una red ART 2 es como sigue:

Paso 0. Inicializar parámetros de la red:

n: número de entradas de la red (capa F1)

m: número máximo de clusters (capa F2)

a, b: pesos fijos de la capa F1. Inicializarlos a cero produce inestabilidad

c: Peso fijo usado para verificar Reset. Un valor pequeño de c proporciona un rango mayor de operación al parámetro de vigilancia ?.

d: activación de unidad ganadora de la capa F2. Los valores escogidos de c y d deben satisfacer la siguiente inecuación: $\frac{c \cdot d}{1 - d} \leq 1$ **Ecuación 1-17**

e: un valor muy pequeño que se usa para prevenir la división entre cero

θ : parámetro de supresión de ruido utilizado en la función de activación de las neuronas. Un valor típico es $1/\sqrt{n}$

ρ : parámetro de vigilancia. Junto con los valores de los pesos ascendentes, b_{ij} , determina la cantidad de clusters que se forman durante el aprendizaje. Valores por debajo de 0.7 no difieren de los resultados obtenidos al fijarlo a valores tan bajos como $\rho = 0$.

t_{ji} : pesos descendentes ($F2 \rightarrow F1$), deben ser inicializados a 0 para asegurarse que no ocurrirá un Reset para el primer patrón admitido en un clúster.

b_{ij} : pesos ascendentes ($F1 \rightarrow F2$), deben ser escogidos cumpliendo la siguiente desigualdad: $b_{ij} \leq 1 / ((1 - d) \cdot \sqrt{n})$ **Ecuación 1-18**

Paso 1. Realizar pasos del 2 al 12 el número especificado de veces de acuerdo a la cantidad de patrones de ejemplo disponibles.

Paso 2. Actualizar los valores de las unidades de la capa F1 con $u_i = q_i = p_i = 0$

$$x_i = s_i / (e + |s_i|)$$

$$w_i = s_i \quad v_i = f(x_i)$$

Paso 3. Actualizar nuevamente los valores en la capa F1 con los resultados del paso anterior

$$u_i = v_i / (e + |v_i|)$$

$$w_i = s_i + a \cdot u_i \quad p_i = u_i + d \cdot t_{ji}$$

$$x_i = w_i / (e + |w_i|) \quad q_i = p_i / (e + |p_i|)$$

$$v_i = f(x_i) + b \cdot f(q_i)$$

Paso 4. Calcular las señales de la capa F2:

$$y_j = \sum_i b_{ij} p_i$$

Ecuación 1-19

Paso 5. Mientras *ResetFlag* sea igual a “verdadero” repetir los pasos 6→7

Paso 6. Encontrar la unidad ganadora, Y_J , de la capa F2

; donde el valor de J es tal que $y_J = y_j$ para $j = 1, 2, \dots, m$)

Paso 7. Verificar el Reset

$$u_i = v_i / (e + |v|)$$

$$p_i = u_i + d * t_{ji}$$

$$r_i = (u_i + c * p_i) / (e + |u| + c * |p|)$$

Si $|r| < \theta - e$, entonces

Resetflag = “verdadero”

$Y_J = -1$. Repetir Paso 6.

Si $|r| < \theta - e$, entonces

$$w_i = s_i + a * u_i$$

$$x_i = w_i / (e + |w|)$$

$$q_i = p_i / (e + |p|)$$

$$v_i = f(x_i) + b * f(q_i)$$

Resetflag = “falso”; continuar con el paso 8

Paso 8. Actualizar los pesos de la unidad ganadora J

$$t_{ji} = u_i / (1 - d)$$

$$b_{ij} = u_i / (1 - d)$$

Paso 9. Verificar si se ha alcanzado el número de iteraciones previstas

1.5.1.2 Ejemplo de ejecución del algoritmo de entrenamiento de la red neuronal ART 2

Paso 0. Se escogen los valores de los diferentes parámetros de la red

$$a = b = 10$$

$$c = 0.1$$

$$d = 0.9$$

$$e = 0 \text{ (valor usual en cálculos manuales)}$$

$$\rho = 0.9$$

$$\rho = 0.7$$

$$\mathbf{b}_j = (7.0, 7.0) \text{ para } j = 1, \dots, m$$

$$\mathbf{t}_j = (0.0, 0.0) \text{ para } j = 1, \dots, m$$

Paso 1 y 2. El primer patrón de entrada es: $\mathbf{s} = (0.8, 0.6)$

$$\mathbf{u}_i = \mathbf{v}_i / (e + \|\mathbf{v}_i\|) \rightarrow (0.0, 0.0)$$

$$\mathbf{w}_i = \mathbf{s}_i + a * \mathbf{u}_i \rightarrow (0.8, 0.6)$$

$$\mathbf{p}_i = \mathbf{u}_i \rightarrow (0.0, 0.0)$$

$$\mathbf{x}_i = \mathbf{w}_i / (e + \|\mathbf{w}_i\|) \rightarrow (0.8, 0.6)$$

$$\mathbf{q}_i = \mathbf{p}_i / (e + \|\mathbf{p}_i\|) \rightarrow (0.0, 0.0)$$

$$\mathbf{v}_i = f(\mathbf{x}_i) + b * f(\mathbf{q}_i) \rightarrow (0.8, 0.0)$$

Paso 3. Actualizar nuevamente la capa F1 con los resultados anteriores

$$\mathbf{u}_i = \mathbf{v}_i / (e + \|\mathbf{v}_i\|) \rightarrow (1.0, 0.0)$$

$$\begin{aligned} \mathbf{w}_i &= \mathbf{s}_i + a * \mathbf{u}_i \rightarrow (0.8, 0.6) + 10 * (1.0, 0.0) \\ &= (10.8, 0.6) \end{aligned}$$

$$\mathbf{p}_i = \mathbf{u}_i \rightarrow (1.0, 0.0)$$

$$\mathbf{x}_i = \mathbf{w}_i / (e + \|\mathbf{w}_i\|) \rightarrow (0.998, 0.055)$$

$$\mathbf{q}_i = \mathbf{p}_i / (e + \|\mathbf{p}_i\|) \rightarrow (1.0, 0.0)$$

$$\begin{aligned} \mathbf{v}_i &= f(\mathbf{x}_i) + b * f(\mathbf{q}_i) \rightarrow (0.998, 0.0) + 10 * (1.0, 0.0) \\ &= (10.998, 0.0) \end{aligned}$$

Como tras dos actualizaciones de la capa F1 se alcanza el equilibrio las señales son enviadas a la capa F2

$$y_j = \sum_i b_{ij} p_i$$

Paso 4. Calcular las señales de la capa F2: **Ecuación 1-20**

Pasos 5 y 6. Dado que no se han modificado los pesos ascendentes todas las neuronas de salida proporcionarían el mismo resultado $y_j = 1.0 * 7.0 + 0.0 * 7.0 = 7 \forall j$

Se escoge entonces como ganadora aquella con el menor subíndice: $J = 1$

Paso 7. Verificar si existen las condiciones de Reset

$$u_i = v_i / (e + |v_i|) \rightarrow (1.0, .0)$$

$$p_i = u_i + d * t_{ji} \rightarrow (1.0, 0.0) + 0.9 * (0.0, 0.0)$$

$$r_i = (u_i + c * p_i) / (e + |u_i| + c * |p_i|) = 1$$

Como se cumple que $|r_i| < \theta - e$, entonces

$$\begin{aligned} w_i = s_i + a * u_i &\rightarrow (0.8, 0.6) + 10 * (1.0, 0.0) \\ &= (10.8, 0.6) \end{aligned}$$

$$x_i = w_i / (e + |w_i|) \rightarrow (0.998, 0.055)$$

$$q_i = p_i / (e + |p_i|) \rightarrow (1.0, 0.0)$$

$$\begin{aligned} v_i = f(x_i) + b * f(q_i) &\rightarrow (0.998, 0.0) + 10 * (1.0, 0.0) \\ &= (10.998, 0.0) \end{aligned}$$

Resetflag = “falso”; continuar con el paso 8

Paso 8. Actualizar los pesos de la unidad ganadora J

$$t_{1i} = u_i / (1 - d) \rightarrow (10.0 \ 0.0)$$

$$b_{i1} = u_i / (1 - d) \rightarrow (10.0 \ 0.0)$$

Paso 9. El proceso anterior debe repetirse para cada uno de los patrones de entrada

1.5.2 Kohonen Self-Organizing Maps (SOM)

1.5.2.1 Estructura [1]

La forma básica de una SOM se muestra en la figura 1-8.

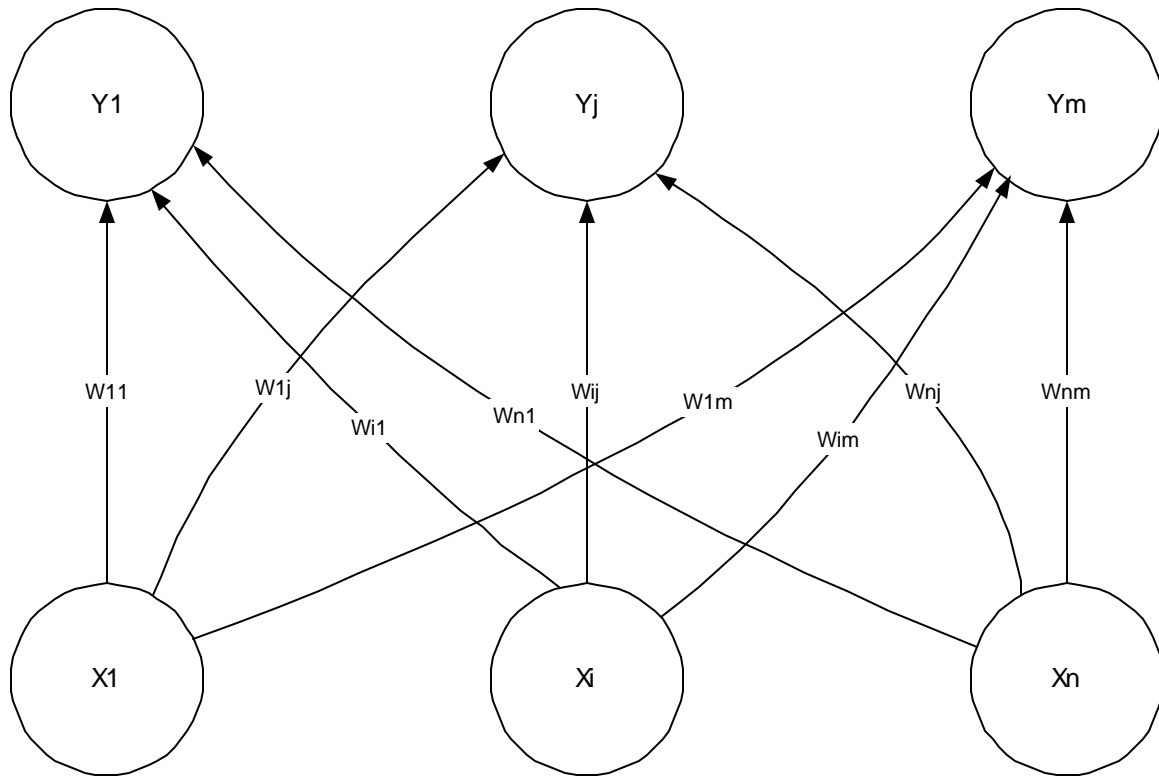


Figura 1-8 Estructura de la SOM

En esta topología los pesos no se multiplican con los vectores de entrada para obtener la salida, muy al contrario sirven como un patrón con el cual asociar o comparar el patrón de entrada, la neurona ganadora es aquella que presenta mayor similitud entre el vector de entrada y su vector de pesos. Es de hacer mencionar que en este tipo de red se utiliza un concepto de vecindad el cual es delimitado por un radio alrededor de la neurona ganadora.

1.5.2.2 Algoritmo de la SOM [1]

En esta red primero se inicializan los pesos, se determinan los parámetros de vecindad y los parámetros de aprendizaje. Se determina también el número total de iteraciones a desarrollar para el aprendizaje.

Para cada vector de entrada se efectúa una comparación con el vector de pesos de las neuronas, y la neurona ganadora es aquella que tiene la mayor similitud entre ellos, en base a la neurona ganadora y al radio de vecindad definido al inicio del proceso se calculan los nuevos pesos para todas las neuronas de la vecindad. Luego se actualiza el factor de aprendizaje y se reduce el radio de la vecindad cada cierto tiempo, hasta que se realizan todas las iteraciones para el aprendizaje.

En una forma más específica se presenta a continuación paso a paso el algoritmo de aprendizaje de la red:

Paso 0. Inicializar pesos W_{ij} , inicializar radio R y definir factor de aprendizaje inicial α

Paso 1. Verificar si es la última iteración a realizar o continuar

Paso 2. Para cada vector de entrada \mathbf{X} realizar pasos del 3 al 5

Paso 3. Para cada j calcular:

$$D(j) = \sum_i (w_{ij} - x_i)^2$$

Ecuación 1-21

Paso 4. Determinar la j para la cual $D(j)$ sea la menor.

Paso 5. Para todas las neuronas de la vecindad de la j ganadora actualizar pesos:

$$w_{ij}(new) = w_{ij}(old) + \alpha [x_i - w_{ij}(old)]$$

Ecuación 1-22

Paso 6. Actualizar α

Paso 7. Reducir R cada cierto número de iteraciones

Paso 8. Verificar iteraciones

1.5.2.3 Ejemplo de aplicación de una SOM.

Los siguientes vectores de entrada van a ser entrenados

$$(1, 1, 0, 0); (0, 0, 0, 1); (1, 0, 0, 0); (0, 0, 1, 0)$$

con un total de dos neuronas de salida ($m=2$) y un factor de aprendizaje de 0.6 y un radio de vecindad de 0.

Paso 0. Inicialización de pesos

$$\begin{bmatrix} 0.2 & 0.8 \\ 0.6 & 0.4 \\ 0.5 & 0.7 \\ 0.9 & 0.3 \end{bmatrix}$$

$$R=0$$

$$\alpha = \alpha * 0.6$$

Paso 1. Iniciar entrenamiento para cada vector de entrada realizar pasos del 2 al 5

Paso 2. Para el primer vector de entrada (1,1,0,0) realizar pasos 3-5

$$\text{Paso 3 } D(1) = (0.2 - 1)^2 + (0.6 - 1)^2 + (0.5 - 0)^2 + (0.9 - 0)^2 = 1.86$$

$$D(2) = (0.8 - 1)^2 + (0.4 - 1)^2 + (0.7 - 0)^2 + (0.3 - 0)^2 = 0.98$$

Paso 4. El vector de entrada es más parecido a la salida D(2) por lo que $J=2$.

Paso 5. Se actualizan los pesos en la neurona ganadora

$$w_{i2}(\text{new}) = w_{i2}(\text{old}) + 0.6[x_i - w_{i2}(\text{old})]$$

Obteniendo la siguiente matriz de pesos:

$$\begin{bmatrix} 0.2 & 0.92 \\ 0.6 & 0.76 \\ 0.5 & 0.28 \\ 0.9 & 0.12 \end{bmatrix}$$

Paso 2. Para el segundo vector de entrada (0,0,0,1) realizar pasos 3-5

$$\text{Paso 3. } D(1) = (0.2 - 0)^2 + (0.6 - 0)^2 + (0.5 - 0)^2 + (0.9 - 1)^2 = 0.66$$

$$D(2) = (0.92 - 0)^2 + (0.76 - 0)^2 + (0.28 - 0)^2 + (0.12 - 1)^2 = 2.276$$

Paso 4. El vector de entrada es más parecido a la salida D(1) por lo que $J=1$.

Paso 5. Se actualizan los pesos en la neurona ganadora, obteniendo la siguiente matriz de pesos:

$$\begin{bmatrix} 0.08 & 0.92 \\ 0.24 & 0.76 \\ 0.20 & 0.28 \\ 0.96 & 0.12 \end{bmatrix}$$

Paso 2. Para el tercer vector de entrada (1,0,0,0) realizar pasos 3-5

$$D(1) = (0.08 - 1)^2 + (0.24 - 0)^2 + (0.2 - 0)^2 + (0.96 - 0)^2 = 1.865$$

Paso 3. $D(2) = (0.92 - 1)^2 + (0.76 - 0)^2 + (0.28 - 0)^2 + (0.12 - 0)^2 = 0.676$

Paso 4. El vector de entrada es más parecido a la salida D(2) por lo que $J=2$.

Paso 5. Se actualizan los pesos en la neurona ganadora, obteniendo la siguiente matriz de pesos:

$$\begin{bmatrix} 0.08 & 0.968 \\ 0.24 & 0.304 \\ 0.20 & 0.112 \\ 0.96 & 0.048 \end{bmatrix}$$

Paso 2. Para el cuarto vector de entrada (0,0,1,1) realizar pasos 3-5

Paso 3.

$$D(1) = (0.08 - 0)^2 + (0.24 - 0)^2 + (0.2 - 1)^2 + (0.96 - 1)^2 = 0.706$$

$$D(2) = (0.968 - 0)^2 + (0.304 - 0)^2 + (0.112 - 1)^2 + (0.048 - 1)^2 = 2.2724$$

Paso 4. El vector de entrada es más parecido a la salida D(1) por lo que $J=1$.

Paso 5. Se actualizan los pesos en la neurona ganadora, obteniendo la siguiente matriz de pesos:

$$\begin{bmatrix} 0.032 & 0.968 \\ 0.096 & 0.304 \\ 0.68 & 0.112 \\ 0.984 & 0.048 \end{bmatrix}$$

Paso 6. Reducir el factor de aprendizaje $\alpha = 0.5 \times 0.6 = 0.3$, volver a presentar nuevamente todos los patrones de entrada hasta un valor previamente determinado de iteraciones.

1.5.3 Learning Vector Quantization (LVQ)

1.5.3.1 Estructura. [1]

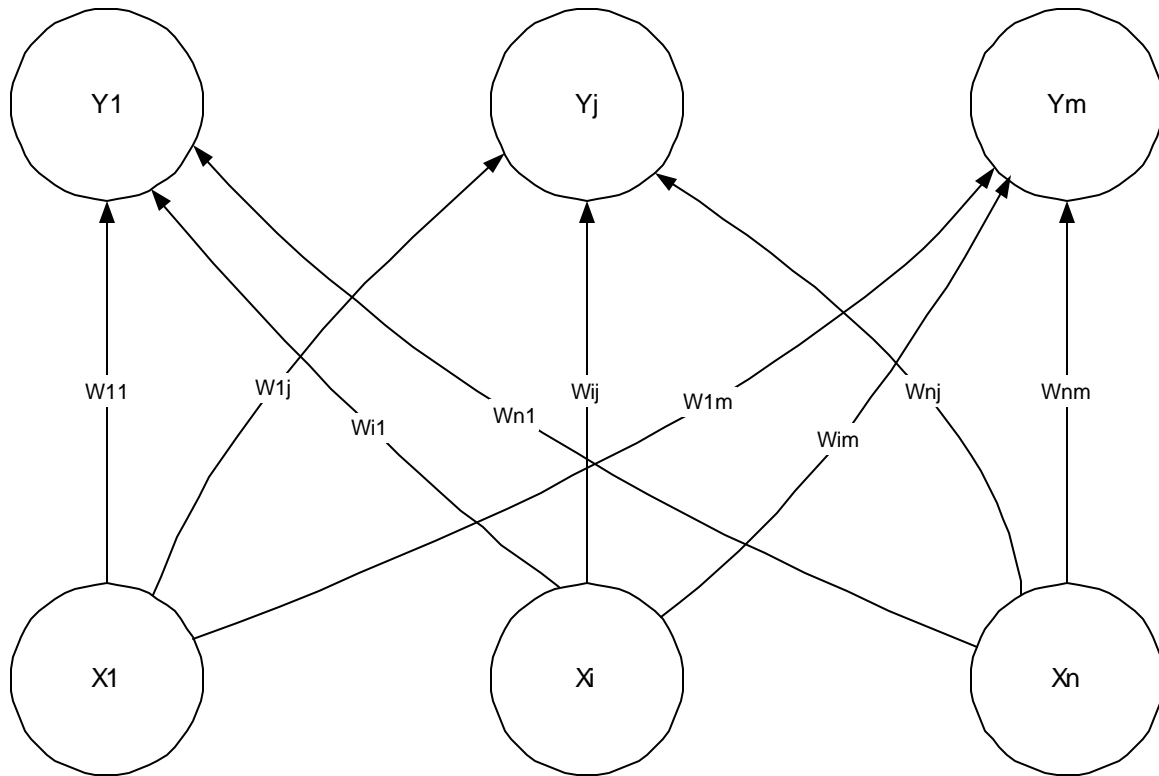


Figura 1-9 Estructura de la red LVQ

La estructura básica de una LVQ se muestra en la figura 1-9, y es esencialmente igual que la estructura de la SOM, con la diferencia de que cada neurona de salida conoce una clase o categoría que ella representa.

Para el presente caso las categorías asignadas a cada neurona de salida son igual que las utilizadas en la red BPN, es decir que se asignó para la primera nota, Do 5ª, la categoría 1 en la primera neurona y así sucesivamente hasta Sol 6ª, la categoría 20 y la última neurona.

1.5.3.2 Algoritmo. [1]

La idea principal de este algoritmo es de encontrar la neurona de salida que más similar sea al vector de entrada. Para el caso en que tanto el vector de los pesos de la neurona de salida como el vector de entrada pertenezcan a una misma categoría se ajustan los pesos acercándolos al vector de entrada; y si la neurona de salida no pertenece a la misma categoría que el vector de entrada se ajustan los pesos alejándolos del vector de entrada. A continuación se presenta la nomenclatura y los pasos a seguir para realizar el aprendizaje de la red.

T = categoría de vector de entrada

C_j = categoría representada por la neurona de salida j-esima

W_j = peso de la neurona de salida j-esima

X = vector de entrada

Paso 0. Determinar categorías para los vectores de entrada, obtener pesos, inicializar factor de aprendizaje α

Paso 1. Realizar procedimiento hasta ultima iteración

Paso 2. Para cada vector de entrada hacer pasos 3 y 4

Paso 3. Encontrar **J** para el cual $\| \mathbf{x} - \mathbf{w}_j \|$ sea el menor.

Paso 4. Actualizar \mathbf{w}_j de acuerdo a las siguientes condiciones.

Si T = C_j entonces

$$w_j(\text{new}) = w_j(\text{old}) + \alpha(x - w_j(\text{old}))$$

Ecuación 1-23

Si T \neq C_j entonces

$$w_j(\text{new}) = w_j(\text{old}) - \alpha(x - w_j(\text{old}))$$

Ecuación 1-24

Paso 5. Reducir factor de aprendizaje α

Paso 6. Verificar condición de parada. (Número de iteraciones)

Capítulo 2. CONCEPTOS MUSICALES

El objetivo principal de un pentagrama es el brindar un medio de representación simbólica de una melodía, incluyendo todas y cada una de las notas ejecutadas, a fin de que cualquier individuo que vea un pentagrama, y que conozca la simbología utilizada, sea capaz de reproducir exactamente la melodía allí representada, aun cuando no la haya escuchado con anterioridad. Para poder comprender e interpretar correctamente un pentagrama es indispensable conocer cada una de las partes que lo conforman, así como las diferentes representaciones usadas para indicar las notas y sus alteraciones, además de la ausencia de las mismas. Por ello se incluyen en éste capítulo las partes del pentagrama, las claves, la forma en que se indican los diferentes tipos de compás y la forma en que se representa una nota musical, tanto su tonalidad como su duración.

2.1 El Pentagrama [11]

La palabra *pentagrama* es de origen griego: *penta* significa cinco y *grama* escrito. Es un conjunto de cinco líneas paralelas que se utiliza para asignar el nombre de las notas, a las figuras musicales que se representan sobre él. Es decir según el espacio o línea en que se encuentre una figura musical dentro del pentagrama, así recibirá un nombre u otro.

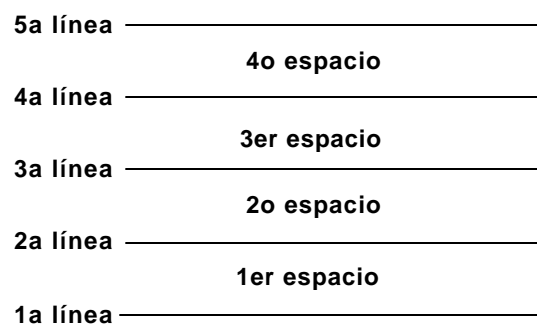


Figura 2-1 El Pentagrama

Como todas las notas de la escala no pueden caber en las cinco líneas y cuatro espacios del pentagrama, a veces es necesario agregarle líneas adicionales por debajo y encima del mismo. En ellas se colocan las notas de la misma forma que en el pentagrama.

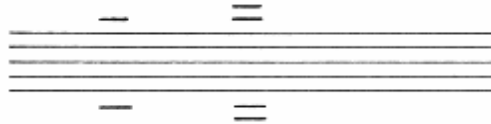


Figura 2-2 líneas adicionales

Al utilizar estas líneas adicionales en un pentagrama se obtienen resultados similares a los que se muestran a continuación:



Figura 2-3 Ejemplo del uso de líneas adicionales en un pentagrama

2.2 Las Claves [11][19]

En todo pentagrama el primer símbolo que encontraremos es una clave, la importancia de las claves es que nos permiten conocer el nombre de las notas que están representadas en el pentagrama. Existen varias claves. Las más utilizadas son la **clave de sol**:



Figura 2-4 clave de Sol

y la **clave de fa**:

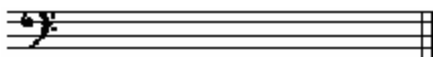






Figura 2-5 clave de Fa

La clave de *sol* es utilizada por instrumentos agudos como el violín, la flauta, la trompeta y por otros no tan agudos como la guitarra. La clave de *fa* es utilizada por instrumentos graves como el contrabajo, el violonchelo y el trombón.

Frecuentemente se usan los símbolos  y  para indicar los compases de 4/4 () y 2/2 ()

2.3 Duración De Las Notas Musicales

La duración de una nota musical se indica por medio de varias figuras, de las cuales presentamos en la tabla 2-1 las usadas con mayor frecuencia en las melodías escritas para flauta dulce soprano [10] [18]






Figura	Nombre	Valor en tiempo
	Redonda	El doble de una blanca
	Blanca	El doble de una negra Mitad de una redonda
	Negra	El doble de una corchea Mitad de una blanca
	Corchea	El doble de una semicorchea Mitad de una negra
	semicorchea	El doble de una fusa Mitad de una corchea

Tabla 2-1. Representación y duración de notas musicales

Cuando se desea indicar ausencia de sonido se utilizan algunos símbolos conocidos como *silencios* entre los que se encuentran:






Tipo de silencio	Símbolo	Nombre
Silencio de Redonda		Pausa
Silencio de Blanca		Media Pausa
Silencio de Negra		Aspiración
Silencio de Corchea		Media Aspiración
Silencio de Semicorchea		Cuarto de Aspiración

Tabla 2-2. Tipos de silencios.

2.4 Alteraciones A Las Notas Musicales. [11]

Las alteraciones son signos que modifican el sonido de una nota, cualquiera de las siete notas se puede alterar de forma ascendente o descendente por medio de las alteraciones. Existen tres tipos de alteraciones las cuales se muestran en la tabla 2-3:




Alteración	Nombre	Efecto
	sostenido	Altera ascendentemente la nota 1 semitono
	bemol	Altera descendentemente la nota 1 semitono
	becuadro	Anula cualquier alteración y devuelve la nota a su estado natural.

Tabla 2-3. Tipos de alteraciones.

Las alteraciones se anteponen a la nota a la cual van a afectar (figura2-6), y su efecto se mantiene hasta el fin del compás en el cual aparecen, recibiendo el nombre de alteraciones de paso, o pueden ir en la armadura (figura 2-7), que es el espacio que hay entre la clave y entre el compás, y en ese caso afectarán a las mismas notas durante toda la pieza.



Figura 2-6 Ejemplo de Alteraciones individuales



Figura 2-7 Ejemplo de Alteraciones en armadura

2.5 El Compás. [11]

Es el patrón rítmico de la música que se escucha. Generalmente se pueden agrupar los tiempos o pulsaciones en grupos de 2, 3 ó 4. Para indicar el compás se usan dos cosas: la indicación de compás y las líneas divisorias.

2.5.1 La indicación de compás

El compás se indica al principio de una obra musical usando dos cifras como se muestra en la figura 2-8:

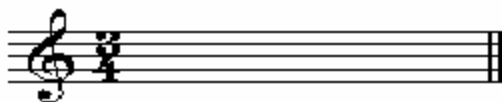


Figura 2-8 Ejemplo de indicación de compás

La cifra superior indica el número de tiempos que tiene el compás, 3 en este caso. La cifra inferior indica la figura que ocupa cada tiempo. En la tabla 2-4 se pueden ver la relación entre figuras y cifras:






Cifra	Figura
1	
2	
4	
8	
16	

Tabla 2-4. Relación entre figuras y cifras.

Es decir que, la notación $\frac{3}{4}$ indica que el compás tiene tres tiempos y que cada tiempo lo representa una negra.

2.5.2 Las líneas divisorias

Para facilitar la lectura, se separan los compases con líneas verticales las cuales se llaman **líneas divisorias**, tal como se muestra en la figura 2-9:



Figura 2-9 Ejemplo de líneas divisorias

En esta figura se observan dos compases de dos tiempos donde cada tiempo es ocupado por una negra y cada compás está separado por una línea divisoria.

2.6 Signos Secundarios. [11] [12]

Los signos secundarios son herramientas auxiliares para determinar la duración de la nota o notas a las cuales afectan.

2.6.1 El puntillo

Si se añade un puntillo a una figura, su duración aumentará de la mitad de su valor, como se muestra en la tabla 2-5:

















Sin puntillo		Con puntillo	
Figura	Equivalencia	Figura	Equivalencia
			
			
			
			

Tabla 2-5. Efecto del puntillo.

2.6.2 La ligadura

La ligadura es un signo que une dos notas de un mismo sonido y casi siempre del mismo nombre, sea cual fuere su duración, es decir que una ligadura aumenta el valor de la primera nota con el valor de la segunda a la cual está ligada



Figura 2-10 Ejemplo de ligadura.

En la figura 2-10, el efecto de la ligadura en el primer compás es igual al efecto del puntillo en el segundo compás.

2.7 Tono y Semitono. [11] [12]

Las notas no guardan entre si las mismas distancias, entre unas la distancia es más grande entre otras la distancia es más corta.

A la distancia más grande de separación entre notas se le llama **tono** y a la distancia más pequeña se le llama **semitono**. En la figura siguiente se puede ver la separación que existe entre las notas.



Figura 2-11 Ejemplo de Tono y Semitono.

Un tono se divide en dos semitonos. Entre dos notas que están separadas por un tono se puede hacer oír una nota que está a un semitono de las dos, es decir en medio de ellas, esto se logra agregando una alteración a una de las notas. Por ejemplo Do y Re están separadas un tono pero agregando un sostenido (#) a Do se tiene una nota que está a un semitono de Re, la cual es Do# y así con todas las notas que están separadas por un tono.

2.8 Transposición o Transporte. [11]

Transportar es ejecutar o transcribir un trozo de música en otro tono distinto de aquel en que está originalmente escrito. El objetivo de la transposición es poner en una tonalidad conveniente a un instrumento, una pieza musical escrita muy alta o muy baja para este instrumento.

Hay dos formas de realizar la transposición que son:

1. Cambiando la posición de las notas en el pentagrama (Escritura).
2. Cambiando la llave del pentagrama (Lectura)

Capítulo 3. ADQUISICIÓN Y PROCESAMIENTO DE LA SEÑAL

El primer paso necesario para iniciar la adquisición de patrones es determinar el tipo de información que será analizada. Para los objetivos del presente sistema basta con definir que en total son veinte notas musicales provenientes de la flauta dulce soprano que se pretenden reconocer; de las cuales se tomaron muestras para realizar los entrenamientos y reconocimientos, estas notas van desde Do 5ª (562 Hz) hasta Sol 6ª (1582 Hz). La frecuencia de muestreo para la adquisición de datos está determinada por el formato wav utilizado, permitiendo adquirir desde 8000 hasta 44100 muestras por segundo.

3.1 Adquisición De Patrones.

No se utilizó ningún tipo de filtrado para eliminar ruido, puesto que el filtrado tendría que ser pasa-banda para el rango de frecuencias de las notas a reconocer, teniéndose la dificultad de que no se podrían eliminar los ruidos con frecuencias en el mismo rango que el de las notas, haciendo inútil el filtro para dichas interferencias.

Para adquirir los patrones necesarios para el entrenamiento y prueba de las diferentes redes fue necesario elaborar una aplicación en Visual Basic que permitiese leer información de la tarjeta de sonido presente en la computadora. Después de una serie de pruebas consistentes en comparaciones entre el funcionamiento de funciones de la API de Windows® y el uso de los objetos incorporados en la colección de VB se determinó que se utilizaría el Control Multimedia de la colección Microsoft Windows Common Controls 6.0 dada la flexibilidad y relativa sencillez encontrada en el mismo.

Los patrones capturados fueron almacenados en formato wav, que es el más simple de los formatos existentes para almacenamiento de sonido ya que no es necesario realizar ningún

tipo de procesamiento de conversión para obtener el sonido almacenado. Cada muestra se grabó por un período de cuatro segundos para contar con una cantidad suficiente de muestras para facilitar el proceso de entrenamiento.

3.2 Procesamiento De La Señal.

Para la ejecución de las diferentes pruebas de entrenamiento con las redes neuronales se elimina la cabecera de los archivos wav (los primeros 45 Bytes) y se busca el inicio de un semiciclo positivo en la forma de onda y a partir de ese punto se tomaban la cantidad de muestras necesarias para completar el número de entradas seleccionadas para las diferentes redes. A menos que se indique lo contrario debe asumirse que se emplearon 80 patrones de entrada, con dos diferentes tipos de agrupamiento: en la primera forma de agrupamiento se presentaba un ejemplo de todas las notas a entrenar, de modo que los patrones correspondientes a la misma nota se repetían cada 20 ejemplos, esto debido a que son veinte el total de notas a reconocer y hasta que se presentaba la última nota se volvía a presentar de nuevo todas las notas. El segundo tipo reunía cuatro ejemplos continuos de la misma nota.

Para utilizar las muestras capturadas y almacenadas en formato wav como entradas para la red neuronal se les sometió a un pequeño procesamiento para mejorar su presentación. En primer lugar, dado que el formato wav es de tipo binario, los datos se almacenan en valores entre 0 y 255¹ siendo 128 el centro o línea base de los datos, por lo que se pasaron a valores de entre -1 y 1 con la siguiente fórmula:

$$ValorNuevo = \frac{ValorBinario - 128}{128}$$

Ecuación 3-1

¹ Rango de valores válido para adquisición de datos con 8 bits. Véase la sección 5.2.

Esta modificación es necesaria para evitar el desbordamiento de las variables en el proceso de entrenamiento. Otro proceso necesario para reducir el efecto de las variaciones de amplitud (o de volumen) de las señales adquiridas es normalizar los vectores de entrada, de modo que los datos presentados a la red sean los vectores unitarios de los patrones.

Otro tipo de procesamiento realizado fue la aplicación del algoritmo de la transformada rápida de Fourier, que permite extraer las componentes en frecuencia de la señal analizada. Esta operación es necesaria para obtener la información correspondiente a la cantidad y magnitud de armónicos presentes en la música capturada.

Capítulo 4. PROCESO DE SELECCIÓN DE LA RED A UTILIZAR.

Para poder determinar cual Red Neuronal utilizar se realizaron diversas pruebas con cuatro diferentes redes, Backpropagation, ART2, SOM y LVQ, teniendo con ellas diversos resultados los cuales fueron tomados en cuenta para la decisión final.

A continuación se presentan todas las pruebas y resultados obtenidos, tanto en el tiempo como en frecuencia, con las diferentes redes.

4.1 Resultados De Pruebas Realizadas Con Valores En El Tiempo

4.1.1 Red Backpropagation

Para esta red se hicieron pruebas con dos funciones de activación distintas que son:

Sigmoidal Binaria

Sigmoidal Bipolar

Los resultados obtenidos con cada una de ellas se muestran a continuación

4.1.1.1 Resultado de pruebas realizadas con salida Sigmoidal Binaria.

Para la BPN que se usaría en el proyecto, para el cual se necesita el reconocimiento de 20 notas musicales, que van desde Do 5ª hasta Sol 6ª, se tomaron 20 neuronas para la capa de salida, las cuales responden en el mismo orden que las notas musicales, es decir que para Do 5ª se activa la primera neurona y así sucesivamente con las demás notas y neuronas hasta Sol 6ª para la cual responde la neurona de salida número 20. En la tabla 4-1 se presenta un diagrama de cómo responden las neuronas de salida para cada patrón de entrada o nota musical. Cada columna representa una neurona de salida, y cada fila corresponde a un patrón de entrada. La “X” indica cual fue la neurona que se activó en cada caso.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Do	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Do#	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Re	-	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Re#	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Mi	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Fa	-	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Fa#	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-
Sol	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	-
Sol#	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-
La	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-
La#	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	-
Si	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-
Do	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-
Do#	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-
Re	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-
Re#	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-
Mi	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-
Fa	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-
Fa#	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-
Sol	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X

Tabla 4-1 Comportamiento de las salidas de la BPN

Se realizaron 8 entrenamientos en total con la BPN con diferentes números de patrones de entrenamiento y diferente número de muestras por patrón como se muestra en la tabla 4-2.

Entrenamiento	Nº Patrones de entrada	Nº Muestras / Patrón
1	20	1024
2	20	200
3	20	200
4	80	33
5	80	33

6	200	33
7	200	33
8	600	15

Tabla 4-2 Patrones de entrada para BPN

En estos ocho experimentos también se utilizó diferentes estructuras para la red con diferentes neuronas de entrada, ocultas y de salida tal y como se muestra en la tabla 4-3.

Entrenamiento	Neuronas		
	Entrada	Ocultas	Salida
1	1024	1024	20
2	200	200	20
3	200	200	20
4	33	33	20
5	33	33	20
6	33	33	20
7	33	33	20
8	12	12	20

Tabla 4-3 Estructuras utilizadas con la BPN

En los ocho entrenamientos se tuvieron diferentes tiempos de duración y también diferentes números de iteraciones por entrenamiento, tal como se muestra en la tabla 4-4.

Entrenamiento	No de iteraciones	Tiempo de entrenamiento
1	12600	01:30
2	660	00:12
3	1563	00:30
4	7324	00:05
5	27620	00:16
6	26840	00:20
7	107730	01:15
8	317220	01:20

Tabla 4-4. Características de tiempo de entrenamiento de la BPN

En la tabla 4-5 se presentan los diferentes porcentajes de aprendizaje que se obtuvieron, así como el error cuadrado promedio (MSE) que se utilizó como condicionante para detener el proceso de aprendizaje en cada entrenamiento.

Entrenamiento	% de Aprendizaje	% de Reconocimiento	MSE
1	20	0	0.025
2	80	10	0.02
3	90	40	0.015
4	98	50	0.0005
5	100	70	0.0001
6	100	77	0.0005
7	100	80	0.0001
8	100	69	0.0006

Tabla 4-5 Efectividad de Entrenamiento y reconocimiento de la BPN con diferentes MSE

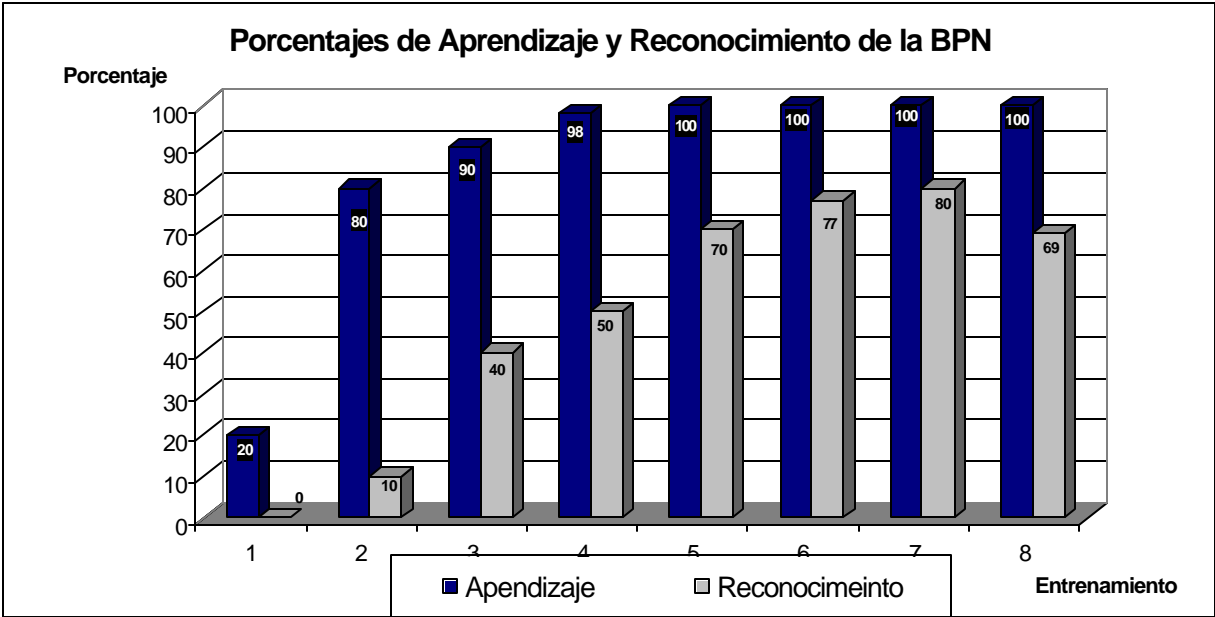


Gráfico 4-1 Porcentajes de Aprendizaje y Reconocimiento de la BPN

En el gráfico 4-1 se presentan los porcentajes obtenidos en cada entrenamiento. Tanto los porcentajes de aprendizaje como los de reconocimiento.

4.1.1.2 Resultado de pruebas realizadas con salida Sigmoidal Bipolar.

Las primeras pruebas de entrenamiento con la red Backpropagation se realizaron con 1024 neuronas de entrada, 128 neuronas ocultas y 20 neuronas en la capa de salida. La tasa de aprendizaje tomó valores desde 0.5 hasta 0.8 manteniéndose constante durante cada intento. El promedio de iteraciones realizadas para esas condiciones fue de aproximadamente 1500, las cuales tomaron cerca de 2:30 horas cada prueba. A pesar del tiempo empleado el error medio se mantuvo por encima de 0.95.

Dado que el algoritmo tradicional de entrenamiento no proporcionaba los resultados esperados se optó por agregar *momentum* a las ecuaciones de actualización de pesos, además de inicializar estos últimos utilizando la regla de Nguyen-Widrow. La prueba con la misma topología (1024-128-20) y el momentum presentaron un desempeño similar al entrenamiento tradicional, por lo que se modificó la estructura de la red hasta aumentar el número de neuronas en la capa oculta a 1024. La cantidad de iteraciones necesarias para que se estabilizara el error medio se redujo a 200, pero aún se estancaba en valores demasiado elevados para nuestros propósitos.

La siguiente modificación a la estructura de la red fue basada en el teorema de Hecht-Nielsen, que establece que una función cualquiera $R^n = f(I^n)$ puede ser representada por una red neuronal de alimentación hacia delante n entradas, $2n+1$ neuronas ocultas y m salidas. Las topologías o estructuras escogidas fueron 511-1203-20, 255-51-20 y 31-65-20 siendo ésta última la que mejores resultados presentó, alcanzando niveles de error de tan solo 0.046 que seguía decreciendo después de 4000 iteraciones. Un resumen de los resultados obtenidos puede verse en la tabla 4-6, cabe mencionar que solo el primer experimento se realizó sin momentum ni pesos inicializados con Nguyen-Widrow:

Experimento	Estructura	No de iteraciones	Tiempo de entrenamiento	MSE
1	1024-128-20	1550	02:30	0.909
2	1024-128-20	1250	02:30	0.809
3	1024-1024-20	230	01:47	0.627
4	511-1023-20	175	01:20	0.525
5	255-511-20	931	00:54	0.437
6	31-65-20	2410	00:41	0.054

Tabla 4-6 Resultados de entrenamiento de BPN con salida Sigmoidal bipolar

4.1.2 Resultado de las pruebas realizadas con la red ART 2.

La red ART 2 presenta un algoritmo complejo pero rápido, ya que su aprendizaje es en línea, con el inconveniente que es del tipo no supervisado, de modo que queda a discreción de la misma red la asignación de clústeres para los patrones recibidos. Los resultados obtenidos varían de acuerdo al parámetro de vigilancia y al orden de presentación de los patrones. Se utilizaron dos formas de ordenar los ejemplos destinados al aprendizaje de la red. En unas pruebas se colocaban sucesiones de una muestra de cada nota, mientras que en otras se aplicaban grupos de 4 patrones correspondientes a la misma nota en forma consecutiva. Para ejemplificar más claramente la forma en que se realizó la asignación se realizó una pequeña aplicación que indicaba en un formulario gráfico la forma en que la red efectuaba la clasificación y los resultados más significativos se muestran a continuación.

Las filas representan cada uno de los grupos formados por la red, mientras que las columnas son cada una de las muestras usadas para el entrenamiento. Hay que notar que cada figura posee 20 filas y 80 columnas, con líneas de referencia en color negro cada 5 filas y cada 10 columnas. El factor de vigilancia empleado aparece en el cuadro de texto de la misma aplicación. La neurona ganadora se indica mediante un rectángulo rosado, y los patrones no identificables se representan con una columna azul negra.

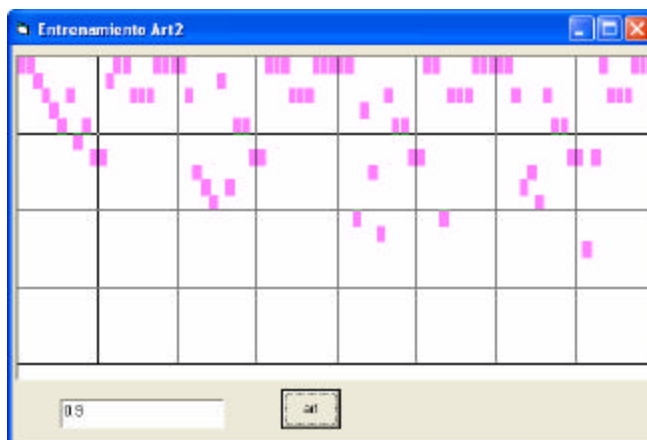


Figura 4-1 Aprendizaje con $\rho=0.9$. 13 grupos

Puede notarse que solo se formaron 13 diferentes categorías y que la mayor parte de la señales fueron asignadas a la primera de ellas. A fin de incrementar el número de categorías se incremento el factor de vigilancia, obteniéndose el siguiente resultado:

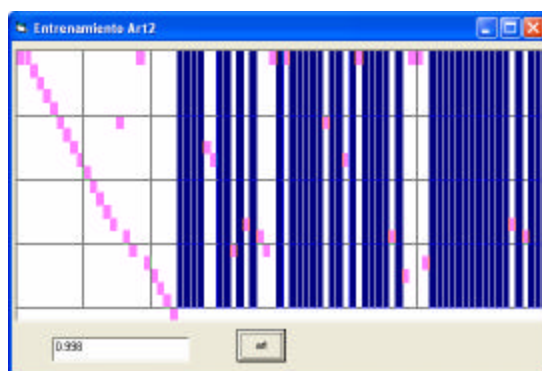


Figura 4-2 Aprendizaje con $\rho=0.998$. 20 grupos.

Es indudable que la clasificación de las notas ha mejorado con respecto al valor presentado anteriormente, pero también se presentaron varios patrones que no fueron aceptados como válidos, lo que se evidencia en la cantidad de columnas azules de la figura 4-3. Esto a pesar de que aun no se ha alcanzado un nivel óptimo de categorización.

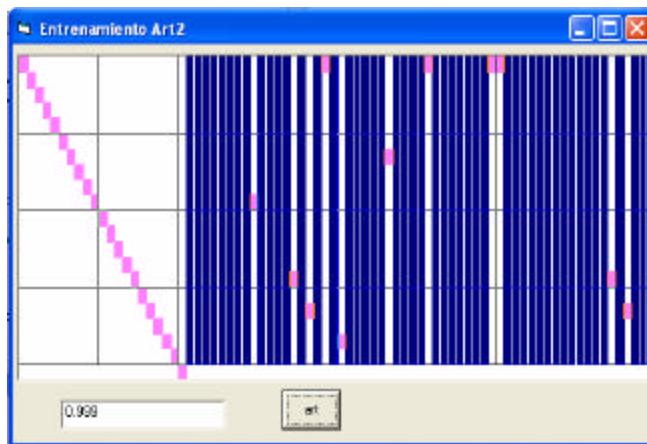


Figura 4-3 Alto nivel de discriminación entre patrones de entrada

Aquí puede verse que con un valor de vigilancia de 0.999 se obtiene una clasificación perfecta de los primeros 20 ejemplos pero el nivel de rechazo del resto de patrones vuelve inadecuados los valores aprendidos por la red.

Al cambiar el orden de presentación de los patrones se obtuvieron los siguientes resultados:

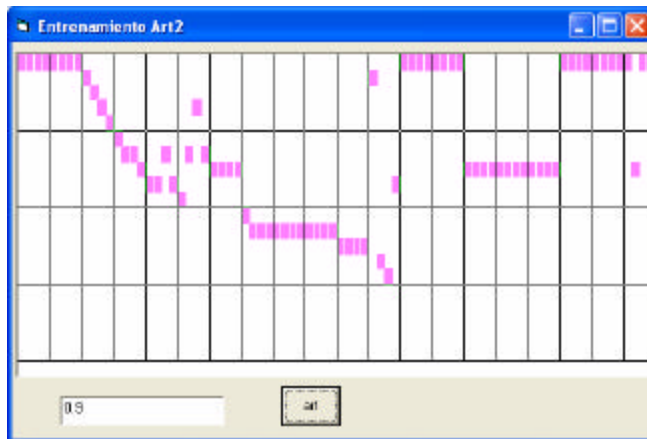


Figura 4-4 Clasificación obtenida al modificar el orden de presentación

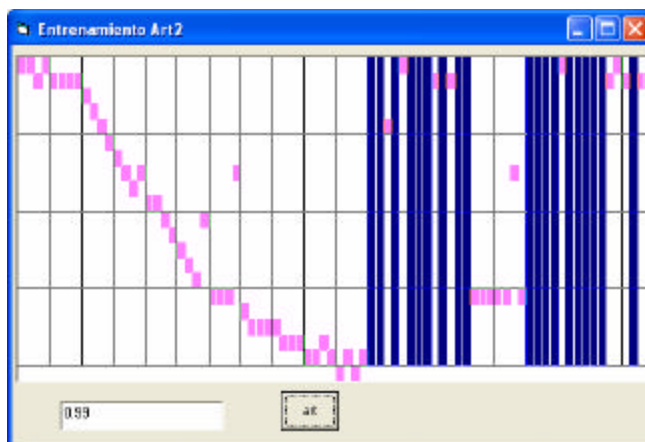
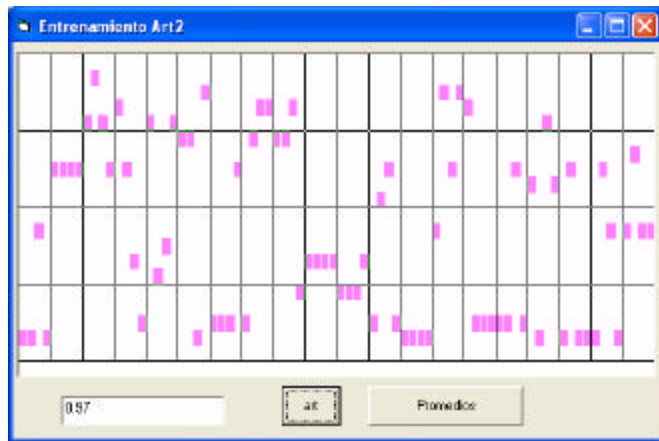


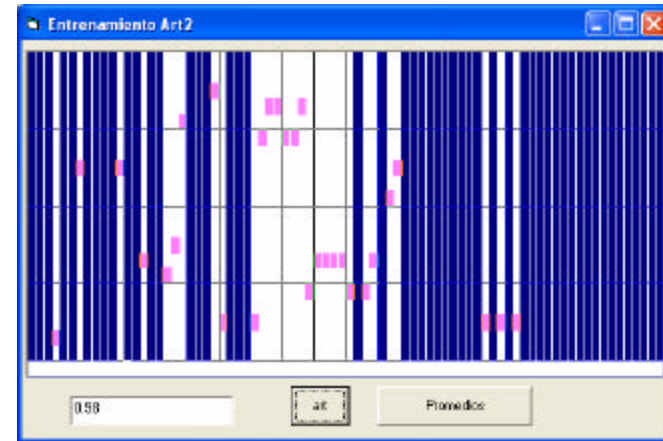
Figura 4-5 Clasificación obtenida con notas agrupadas

Es evidente la mejoría en el nivel de diferenciación obtenido, pero aún está lejos de ser ideal. Una gráfica sin ninguna desviación habría sido una escalinata formada por pasos de cuatro muestras en el mismo cluster de salida.

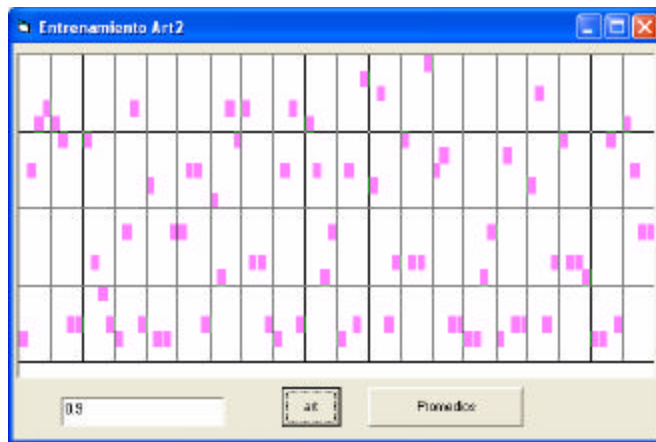
Tomando en cuenta que los pesos de una red ART 2 son al final de cuentas un promedio de los vectores normalizados pertenecientes a cada categoría se procedió a calcular el promedio de varias muestras y fueron esos vectores de promedios los que se presentaron a la red. La idea era que se entrenaba con los valores que teóricamente debía memorizar no debería existir dificultad alguna para clasificar correctamente los patrones de prueba. Sin embargo los resultados obtenidos distan casi tanto como los anteriores de los valores de salida deseados, como puede verificarse en las siguientes imágenes (Figura 4-6. a-d)



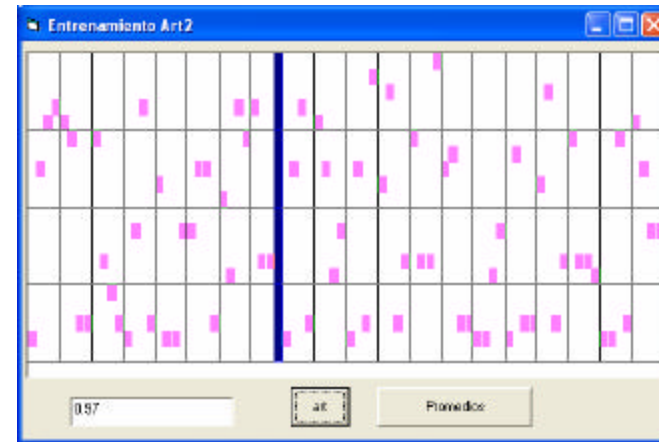
a)



b)



c)



d)

Figura 4-6 Red entrenada con promedios a) y b) notas mezcladas c) y d) notas separadas

4.1.3 Resultados obtenidos con la SOM.

En nuestra red SOM se utilizaron 20 unidades de salida, una para cada tipo de nota musical, esperando que la red asigne a cada neurona de salida la nota correspondiente. Para todos los procesos de entrenamiento que se realizaron con las SOM se utilizó un valor inicial del factor de aprendizaje igual a 0.6, iba disminuyendo cada iteración en relación a la siguiente función $\alpha = 0.9 (\alpha)$. También se tomó un radio inicial igual a 4 e iba disminuyendo en uno cada cierto número de iteraciones. En la tabla 4-7 se presentan los patrones de entrada por entrenamiento y el número de muestras por patrón, y en la tabla 4-8 se presenta la estructura que se utilizó para cada entrenamiento

Entrenamiento	Nº Patrones de entrada	Nº Muestras / Patrón
1	20	500
2	20	200
3	20	200
4	20	200

Tabla 4-7 Patrones de entrada para entrenar la SOM

Entrenamiento	Neuronas	
	Entrada	Salida
1	500	20
2	200	20
3	200	20
4	200	20

Tabla 4-8 Estructuras utilizadas por entrenamiento para la SOM

En la primera prueba de entrenamiento, con un número total de 5000 iteraciones, el radio se fue disminuyendo, en 1, cada 1000 iteraciones. El tiempo total de aprendizaje fue de

aproximadamente dos minutos, los resultados obtenidos fueron malos puesto que no reconoció ninguno de los patrones de entrada que posteriormente se le aplicaron a la red. Debido a estos resultados se decidió disminuir el número de muestras por patrón de entrada. Utilizando un total de 1000 iteraciones para el aprendizaje y disminuyendo el radio en 1 cada 200 iteraciones, los resultados obtenidos fueron en este caso mejor que en el primer entrenamiento aunque no satisfactorios para nuestras necesidades. El proceso de aprendizaje duró aproximadamente un minuto, un poco menos, y logró clasificar 9 de las 20 notas musicales al momento de probar la red. Se entrenó nuevamente la red pero ahora con un número mayor de iteraciones (5000) y el radio se iba disminuyendo siempre en 1, pero ahora cada 1000 iteraciones. El proceso de aprendizaje duró aproximadamente un minuto y medio y la red logró reconocer 10 de las 20 notas musicales al probarla. Se aumentó nuevamente el número de iteraciones, ahora a 10000 y disminuyendo el radio cada 2000 iteraciones, el tiempo del proceso de aprendizaje fue de 2 minutos aproximadamente y logró reconocer 10 de las 20 notas musicales.

Como una observación a los últimos tres procesos de aprendizaje se puede decir que, las notas que no reconocía correctamente la red las asignaba siempre en los tres casos a las mismas notas musicales, por ejemplo una nota que no reconoció en ninguna ocasión fue Fa 5ª y siempre la red la reconocía como si fuera Re 5ª, y así con las otras notas que no reconocía correctamente, por lo que se puede decir que la red consideraba que eran muy similares y las clasificaba como si fuesen la misma nota musical.

Entrenamiento	N° de iteraciones	Tiempo de entrenamiento	a ($a = 0.9*a$)	R ($R = R-1$)	% de Aprendizaje
1	5000	2:15	0.6	4	10
2	1000	0:50	0.6	4	10
3	5000	1:25	0.6	4	50
4	10000	1:55	0.6	4	50

Tabla 4-9 Resultados de entrenamiento de SOM

4.1.4 Resultados obtenidos con la LVQ.

Para el entrenamiento de esta red se utilizó un factor de aprendizaje, α , con una tendencia de reducción lineal, según la siguiente relación $\alpha = \alpha \times 0.9$, y los pesos iniciales para cada neurona de salida fueron iguales al vector de entrada de la misma categoría, es decir que para la primera neurona de salida sus pesos iniciales fueron las muestras de un patrón de Do 5ª, para la segunda fueron las muestras de un patrón de Do 5ª # y así sucesivamente hasta que para la última neurona sus pesos iniciales fueron las muestras del patrón correspondiente a Sol 6ª. Para el proceso de entrenamiento se utilizaron un total de 80 patrones de entrada, cuatro por cada nota musical, pero el primer vector de cada nota musical se utilizó como peso inicial, por lo que solamente quedaron 60 patrones para entradas de entrenamiento de la red.

Entrenamiento	Nº Patrones de entrada	Nº Muestras / Patrón
1	60	200
2	60	200
3	60	33
4	60	33

Tabla 4-10 Patrones de entrada para entrenar la LVQ

Entrenamiento	Neuronas	
	Entrada	Salida
1	200	20
2	200	20
3	33	20
4	33	20

Tabla 4-11 Estructuras utilizadas por entrenamiento para la LVQ

En el primer proceso de entrenamiento de la red cada patrón de entrada tenía 200 muestras y se realizaron 1000 iteraciones, tardándose un tiempo aproximado de 4 minutos para el aprendizaje. Los resultados obtenidos no fueron buenos puesto que solamente reconoció una de las 20 notas musicales, utilizando 20 patrones de los 60 utilizados para el entrenamiento.

Se hicieron ajustes a la primera prueba y se decidió solamente incrementar el número de iteraciones a 5000 para ver si mejoraba el aprendizaje, se tardó en el entrenamiento aproximadamente 29 minutos y los resultados fueron iguales que en la primera prueba, es decir que solo reconoció una nota de las 20. Tomando en cuenta que para este número de muestras por patrón no importaba si se aumentaban el número de iteraciones decidimos disminuir el número de muestras por patrón para observar que sucedía. Cabe mencionar que en las primeras dos pruebas las notas musicales fueron identificadas, no correctamente, pero si por las mismas neuronas de salida en ambas pruebas, asignando en más de una ocasión varias notas a una sola neurona de salida.

Con la disminución del número de muestras por patrón a 33 y con un número total de 2000 iteraciones se realizó una nueva prueba, tardándose en el aprendizaje aproximadamente 1 minuto y medio obteniendo mejores resultados, la red reconoció 19 de 20 notas al utilizar 20 vectores de entrada de los 60 utilizados en el proceso de entrenamiento, pero al usar vectores de entrada diferentes solamente reconoció el 50 % de las notas, es decir 10 notas de las 20 totales.

Se observó que los resultados habían mejorado con la reducción del número de muestras por patrón se decidió incrementar el número de iteraciones para el aprendizaje a 5000 el tiempo total del proceso de entrenamiento fue en este caso de aproximadamente 3 minutos y los resultados obtenidos fueron idénticos a los de la prueba anterior.

Para las últimas pruebas las notas musicales reconocidas fueron las mismas en ambas redes y las que no reconocieron correctamente también las asignaron a las mismas neuronas de salida.

Tomando en cuenta los resultados anteriores se puede decir que la red, por la similitud de los patrones de las notas musicales, hacía un correcto reconocimiento de algunas notas musicales y las otras que no reconocía correctamente las clasificaba como una de las que ya había conocido, por lo que se puede decir que la red creaba categorías en vez de diferenciar cada nota de forma individual.

Entrenamiento	Nº de iteraciones	Tiempo de entrenamiento	a ($a = 0.9*a$)	% de Aprendizaje
1	1000	4:20	0.1	10
2	5000	28:50	0.1	10
3	2000	1:37	0.1	50
4	5000	2:57	0.1	50

Tabla 4-12 Resultados de entrenamiento de LVQ

4.2 Resultados De Pruebas Realizadas Con Valores En Frecuencia

Basados en los resultados obtenidos en las pruebas de topologías en el dominio del tiempo se decidió tomar únicamente dos topologías de redes para las pruebas en frecuencia las cuales fueron la BPN, con las dos salidas de activación, y la ART 2 .

Ya elegidas las redes con las cuales se iba a trabajar en el dominio de la frecuencia se optó por realizar un procesamiento más intenso de las señales captadas, a fin de obtener características que permitiesen identificar en forma única e inequívoca cada nota presentada a las redes. El proceso consistió en tomar los patrones de entrada y realizar una transformada rápida de Fourier, y el resultado de esa transformación se presenta como patrón de entrada.

4.2.1 Red Backpropagation.

4.2.1.1 Resultados obtenidos con la salida de activación sigmoideal

Para las primeras pruebas se introdujo la totalidad de los resultados de la FFT a la red (1024 muestras), con lo que los resultados en cuanto a grado de aprendizaje y desempeño fueron prácticamente idénticos a los que se obtenían cuando no se les aplicó la FFT, por lo que resulta impropia la documentación de los mismos. Como segunda variación de condiciones de entrada se presentaron a la red las posiciones de los tres puntos del espectro donde se presentaba la más alta componente armónica. Tales pruebas fueron todas fallidas, sin importar los valores de la tasa de aprendizaje ni si se usa o no momentum para el entrenamiento, los patrones asociados presentaban tal grado de similitud que la red era incapaz de diferenciarlos, de modo que ningún patrón fue correctamente aprendido.

Luego de eso se modificaron las condiciones de entrada, dejando únicamente los valores de las componentes de frecuencia donde se presentaba la mayor concentración de energía, es decir de 485 Hz a 1650 Hz, suprimiendo las componentes por debajo y por encima de ese rango. El porcentaje de reconocimiento se incrementó notablemente bajo estas condiciones, llegando hasta valores superiores al 95 %.El consolidado de las pruebas realizadas se observa en la tabla 4-13, NW= inicialización de pesos con Nguyen-Widrow [1]

Estructura	No Iteraciones	Tiempo de entrenamiento	MSE	Condiciones	NW
1024-1024-20	3300	02:40	0.997	todas las frecuencias	No
1024-1024-20	2050	01:25	0.97	todas las frecuencias	Si
512-1024-20	5000	02:05	0.89	todas las frecuencias	No
512-1024-20	3700	01:20	0.91	todas las frecuencias	Si
3-7-20	10000	00:37	0.94	3 puntos máximos	Si
110-110-20	1200	00:07	0.00001	485 Hz a 1650 Hz	Si

Tabla 4-13 Resultados de pruebas con la BPN y salida de activación sigmoideal

4.2.1.2 Resultados obtenidos con salida de activación binaria

Para este tipo de salida de activación se realizaron cuatro pruebas en el dominio de la frecuencia obteniéndose mejores resultados que los obtenidos en el dominio del tiempo. Los entrenamientos se hicieron con diferentes números de patrones de entrenamiento y diferente número de muestras por patrón de entrada tal como se muestra en la tabla 4-14.

Entrenamiento	Nº Patrones de entrada	Nº Muestras / Patrón
1	20	1024
2	200	3
3	200	110
4	200	110

Tabla 4-14 Patrones de entrada para BPN

En estos cuatro experimentos también se utilizó diferentes estructuras para la red con diferentes neuronas de entrada, ocultas y de salida tal y como se muestra en la tabla 4-15.

Entrenamiento	Neuronas		
	Entrada	Ocultas	Salida
1	1024	1024	20
2	3	3	20
3	110	110	20
4	110	110	20

Tabla 4-15 Estructuras utilizadas con la BPN

En los cuatro entrenamientos se tuvieron diferente número de iteraciones por entrenamiento y tiempo de duración de cada entrenamiento tal como se muestra en la tabla 4-16.

Entrenamiento	No de iteraciones	Tiempo de entrenamiento
1	5210	00:35
2	8890	00:18
3	1000	01:05
4	15000	01:20

Tabla 4-16 características de tiempo de entrenamiento de la BPN

En la tabla 4-17 se muestran los porcentajes de aprendizaje obtenidos en los entrenamientos y reconocimientos, así como el error cuadrado promedio que se utilizó para cada entrenamiento.

Entrenamiento	% de Aprendizaje	% de Reconocimiento	MSE
1	25	0	0.01
2	0	0	0.01
3	100	90	0.00015
4	100	93	0.000075

Tabla 4-17 Efectividad de Entrenamiento y reconocimiento de la BPN con diferentes MSE

En el gráfico 4-2 se presentan los porcentajes obtenidos en cada entrenamiento. Tanto los porcentajes de aprendizaje como los de reconocimiento.

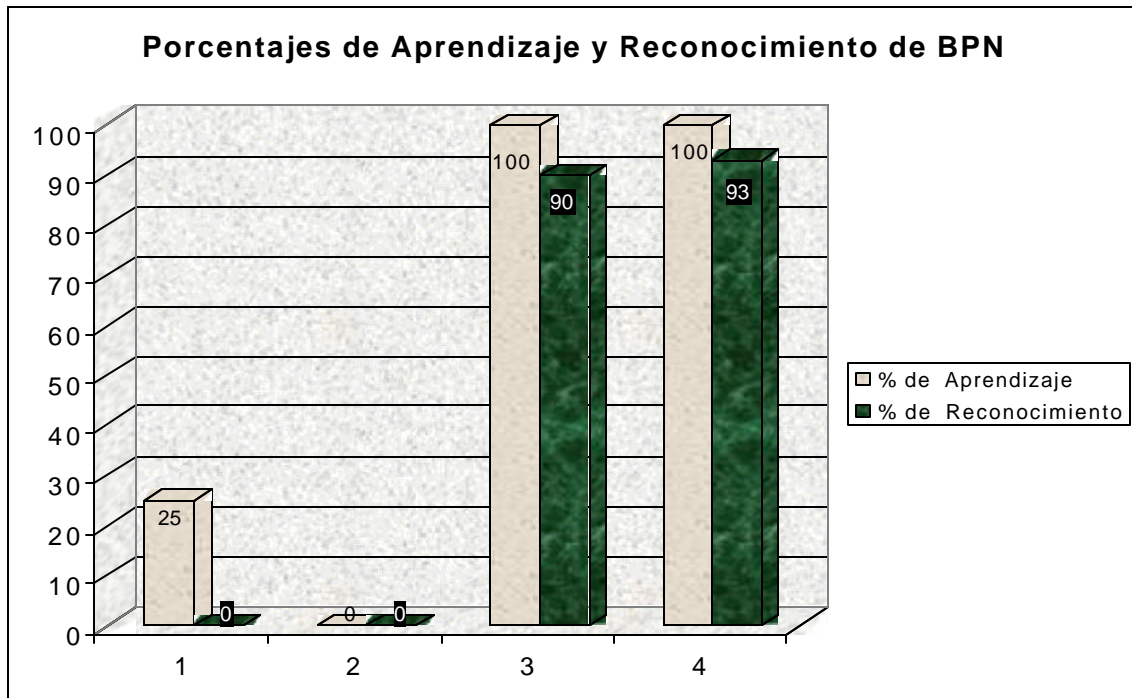


Gráfico 4-2 Porcentajes de aprendizaje y reconocimiento de la BPN

4.2.2 Resultados obtenidos con la red ART 2.

En base a las pruebas anteriores, y que no se obtuvo un 100% de efectividad en el reconocimiento de los patrones, se inició el entrenamiento de la red ART 2 utilizando los promedios de 400 patrones de entrada, con un factor de vigilancia igual a 0.8621 se formaron los 20 grupos que representan a cada una de las notas. Los patrones eran los obtenidos de la FFT después de eliminar la componentes fuera del rango de 485 Hz a 1650 Hz. Una vez entrenada la red se procedió a realizar las pruebas correspondientes obteniéndose resultados sorprendentemente mejores que los proporcionados por la red Backpropagation. En la figura 4-7 se presentan de forma gráfica estos resultados; se le aplicaron a la red 80 patrones de entrada cuatro por cada nota musical, los cuales están representados por los cuadros rosados. Cada grada en la gráfica representa el inicio de otros 20 patrones diferentes de reconocimiento. El no reconocimiento de una nota (o patrón de entrada) sería representado por una columna azul negro en vez de rosado.

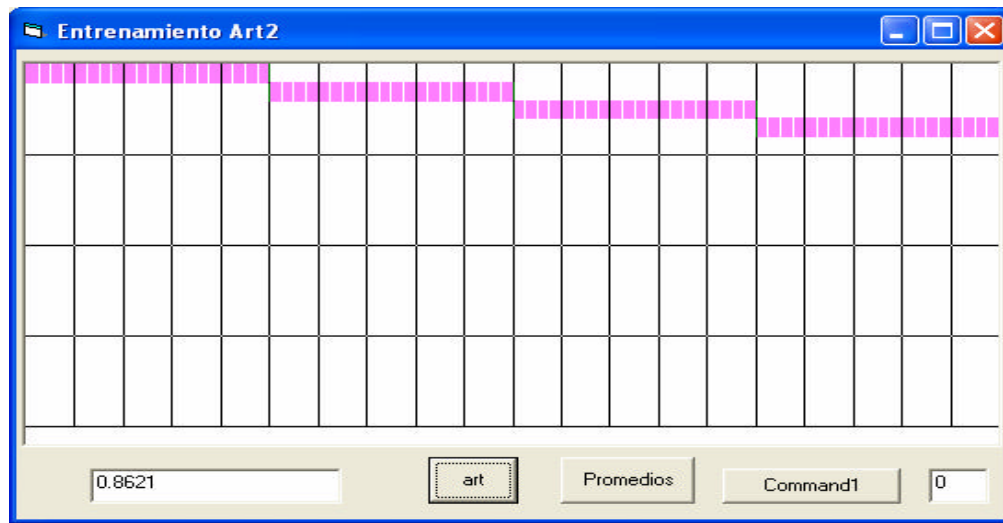


Figura 4-7 Resultados obtenidos con la ART 2

4.3 Selección De La Red

Para las pruebas de topologías de redes en el dominio del tiempo, puede decirse que la red Backpropagation fue la que mostró un mejor desempeño al momento de realizar la identificación de las notas presentadas, ya que, debido al elevado nivel de semejanza entre los patrones, la categorización efectuada por las otras tres topologías se mostraba demasiado exigente y discriminatoria, o por el contrario demasiado general haciendo totalmente inadecuada la respuesta del sistema, mientras que para las pruebas en frecuencia se observó que la red ART2 fue la que obtuvo un mejor desempeño comparada con la otra red con la cual también se hicieron pruebas en el dominio de la frecuencia.

Al analizar todos los resultados obtenidos con todas las topologías de las redes y con los diferentes procesamiento de las señales de entrada se decidió tomar la red ART 2 con patrones de entrada en el dominio de la frecuencia como la red adecuada para el desarrollo de la aplicación debido esto a que logro un porcentaje de reconocimiento de un 100%, que es lo que pretendíamos o buscábamos obtener, de una red neuronal, para lograr una máxima eficiencia en nuestra aplicación final.

Capítulo 5. DESCRIPCIÓN DEL FUNCIONAMIENTO.

El sistema posee una interfaz gráfica altamente intuitiva que facilita su uso, pues la mayor parte de tareas son las mismas que se encuentran en prácticamente todas las aplicaciones compatibles con Windows®, tales como Guardar, Cerrar, Guardar como..., etcétera.

El proceso que se realiza durante la ejecución normal del programa puede dividirse en cinco actividades o subprocesos, cada uno encargado de una tarea específica que contribuye al correcto funcionamiento del conjunto, y cuya relación se presenta en el siguiente diagrama de bloques:

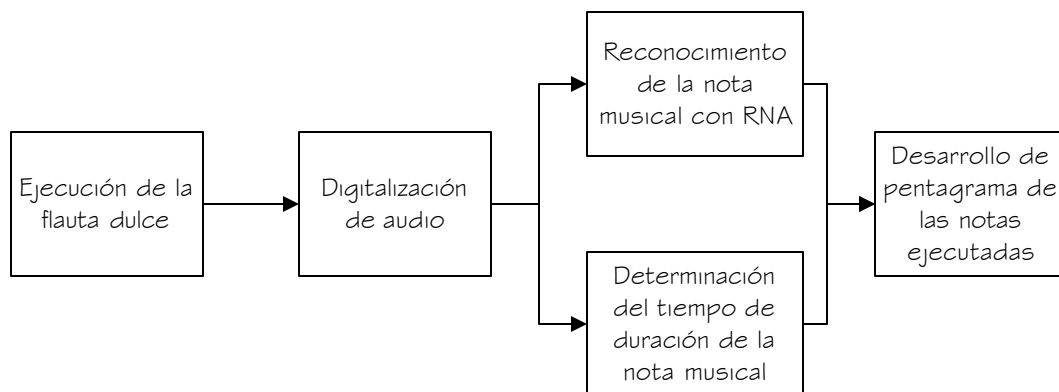


Figura 5-1 Diagrama de bloques del sistema.

5.1 Ejecución De La Flauta Dulce

Este segmento depende por completo de las habilidades y características del usuario del sistema, por tanto ningún proceso interno es llevado a cabo en la computadora. La condición más importante en esta etapa es contar con un ambiente tranquilo, sin perturbaciones de ningún tipo.

5.2 Digitalización De Audio

La captura y digitalización de audio se efectúa mediante la tarjeta de sonido de la PC, lo que le otorga un alto grado de comodidad al usuario, pues elimina la necesidad de adquirir hardware adicional para poder utilizar el sistema.

El sonido es almacenado en archivos de formato WAV, que presenta la forma más simple de registrar sonido en forma digital. La estructura de este tipo de archivos se muestra en la tabla 5-1.

Bytes	Contenido Usual	Descripción
1-4	"RIFF"	Bloque de identificación (sin comillas).
5-8		Tamaño del archivo en bytes, incluyendo cabecera.
9-12	"WAVE"	identificador
13-16	"fmt "	identificador
17-20	16	Tamaño de la cabecera hasta este punto.
21-22	1	versión del tipo de formato utilizado
23-24	1	Número de canales (2 si es estéreo).
25-28	11025 ²	Frecuencia de muestreo (muestras/segundo).
29-32	11025 ²	Número medio de bytes/segundo.
33-34	1	Alineamiento de bloque.
35-36	8 ²	Número de Bits por muestra (normalmente 8, 16 ó 32).
37-40	"data"	Marcador que indica el comienzo de los datos
41-44		Número de bytes muestreados.
45->		Muestras (cuantificación uniforme)

Tabla 5-1. Formato de los archivos wav

² Valores varían de acuerdo al fabricante y a la configuración de la tarjeta de sonido utilizada

La adquisición se realiza utilizando dos archivos wav diferentes. La forma en que se realiza puede describirse como se indica a continuación

1. Indicación de inicio de la captura de notas por parte del usuario.
2. Inicio de grabación de audio en el primer archivo durante un segundo
3. Cierre del primer archivo e inicio de grabación en el segundo archivo
4. Procesamiento del primer archivo
5. Cierre del segundo archivo e inicio de grabación en el primer archivo
6. Procesamiento del segundo archivo
7. Repetición de los pasos 3 al 6 hasta que se detenga la operación de captura de notas

La principal limitante que se encuentra al llevar a cabo adquisición de audio es la amplia gama de tarjetas de sonido que se existen actualmente, teniendo cada una de ellas diferentes capacidades en cuanto a frecuencia de muestreo, número de bits por muestra y número de canales utilizados. Para alcanzar un desempeño adecuado ante las diferentes combinaciones posibles fue necesario incluir procedimientos que identificaran los tres aspectos mencionados anteriormente. En total pueden presentarse varios casos diferentes, de los cuales fueron seleccionados los 16 que se encuentran típicamente, tal como se indica en la tabla 5-2. De acuerdo al caso que se presente se deben realizar procesos de adecuación o transformación de los datos, ya que, por ejemplo, si el archivo es de 8 bits/muestra los datos están entre 0 y 255, pero si son 16 bits/muestra tomarán valores entre 0 y 65535, de modo que el ajuste necesario diferirá un poco del expresado en la ecuación 3-1. En caso de encontrarse un archivo grabado en dos canales se toma solamente uno de ellos para efectuar la identificación. Las variaciones en la frecuencia de muestreo se compensan incrementando o disminuyendo el número de muestras leídas del archivo, de modo que el resultado de la fft corresponda siempre a 10.76 Hz/muestra.

No	Frec de muestreo (Hz)	No de canales	bits/muestra
1	8000	1	8
2	8000	1	16
3	8000	2	8
4	8000	2	16
5	11025	1	8
6	11025	1	16
7	11025	2	8
8	11025	2	16
9	22050	1	8
10	22050	1	16
11	22050	2	8
12	22050	2	16
13	44100	1	8
14	44100	1	16
15	44100	2	8
16	44100	2	16

Tabla 5-2. Combinaciones de parámetros wav soportados

5.3 Reconocimiento De La Nota Musical Con RNA

El proceso de reconocimiento de las notas musicales es, básicamente, llevado a cabo mediante la implementación de una Red Neuronal Artificial basada en la Teoría de Resonancia Adaptativa, mejor conocida como red ART2³. Para llevar a cabo el reconocimiento de las notas se divide cada archivo de audio en segmentos de una duración

³ El algoritmo detallado del funcionamiento de la red ART2 se encuentra en el capítulo 1 y las razones que motivaron la selección de esta red son presentadas en el capítulo 4.

de 125 milisegundos, cada uno de estos segmentos es sometido a un proceso de transformación al dominio de la frecuencia, a fin de extraer la información de su contenido armónico para luego ser presentado a la red neuronal para que sea ésta la que identifique la nota ejecutada durante ese intervalo. Por si mismos, los resultados obtenidos de esta etapa no son concluyentes para la elaboración del pentagrama, siendo, por tanto, indispensable que se combinen con los de la etapa de detección del tiempo para producir un resultado coherente.

5.4 Determinación Del Tiempo De Duración De La Nota Musical

El tiempo durante el cual se ejecuta una nota es un dato muy importante en el proceso de construcción de un pentagrama, dado que de eso depende el tipo de figura utilizada para su representación, lo que convierte a este bloque, junto con el de la etapa basada en la RNA, en el centro de operaciones del sistema.

El proceso utilizado consiste en reunir los datos provenientes de la red neuronal y verificar la presentación de una misma nota durante dos o más segmentos consecutivos, procediendo entonces a calcular la duración de una nota multiplicando el número de segmentos consecutivos en los que está presente por la duración de un segmento. Cada uno de esos segmentos posee una extensión de 125 ms, que equivale a la duración correspondiente a una fusa, es decir, la mitad de una semicorchea.

Para obtener un resultado satisfactorio el proceso debe superar dos obstáculos de solución antagónica que vuelven muy compleja la tarea de calcular la duración de una nota. El primero de ellos es el elevado nivel de plasticidad de la red neuronal, característica indispensable para reconocer con facilidad patrones que no concuerden completamente con los almacenados en la red, que da lugar a la aparición de valores de salida inapropiados ante la presencia de perturbaciones externas (como ruido ambiental, voces, vibraciones, etc.) o propias del ejecutante (variaciones del flujo de aire, presión excesiva o escasa de los

dedos). La respuesta ante esas falsas salidas de la red es establecer un mínimo de duración de una nota para ser considerado válido, ignorando aquellos valores que no alcancen el tiempo requerido, para el presente proyecto esa duración es la de dos segmentos consecutivos, que equivalen a una semicorchea. El segundo obstáculo lo constituye la identificación de dos notas consecutivas que posean el mismo tono como dos entidades diferentes, dado que la separación entre ellas es muy pequeña y variable. La solución sería obviamente tomar en cuenta las variaciones más pequeñas que sea posible, pero eso no es compatible con la medida tomada para reducir los efectos de la elevada plasticidad de la red, siendo, por tanto necesario encontrar un punto de equilibrio entre ambas propuestas. El procedimiento seguido para superar esta situación fue verificar el nivel energético del primer armónico de la señal identificada en cada uno de los segmentos procesados, y si éste quedaba por debajo de cierto nivel de umbral se daba por terminada la nota actual y se inicializaban los contadores de la duración. El nivel de umbral varía con cada nota, y fue determinado después de una serie de pruebas y análisis de las notas que se identificarían.

Para otorgar un mayor nivel de flexibilidad al sistema para poder ser usado bajo diferentes condiciones de ruido ambiental se agregó un control que permite modificar los valores usados como umbral. De modo que un nivel bajo es apropiado en un ambiente silencioso y un nivel elevado es adecuado en ambientes donde la tranquilidad no esté completamente garantizada, ya que esos valores también son utilizados para determinar si en un segmento hay en realidad una nota o si es un sonido tan débil que debe ser considerado silencio. Es importante notar que un valor elevado también incrementa el nivel de sensibilidad respecto a las variaciones en la ejecución de una nota, especialmente en aquellas de duración prolongada, por lo que es posible que si no se ejecuta uniformemente una nota pueda ser reconocida como dos notas del mismo tono tocadas una después de la otra, razón por la cual es preferible el uso de niveles bajos de umbral en ambientes silenciosos.

Antes de presentar el flujograma del algoritmo utilizado para determinar la duración de las notas es necesario presentar algunas de las variables utilizadas para llevar a cabo dicha tarea, a fin de volver más sencilla su comprensión.

Duración: en esta variable se almacena la duración de las notas ejecutadas

Nnew: valor devuelto por la RNA justo antes de ejecutar este algoritmo.

Nant: última nota ejecutada e identificada

Ncand: nota que se perfila como posible candidata a ser la siguiente nota a incluir en el pentagrama

Duracand: duración de la nota candidata.

La primera vez que se recibe un resultado proveniente de la red neuronal solamente se inicializan las variables, la siguiente nota recibida entra al proceso de selección, primero se compara si es igual a la nota del segmento anterior ($N_{ant}=N_{new}$) si eso es así se incrementa en 1 la cantidad de segmentos consecutivos en los cuales aparece la nota en cuestión ($Duracand=Duracand+1$). En caso de no cumplirse lo anterior puede ser que la nueva nota sea igual a la nota candidata (N_{cand}), es decir, aquella nota que probablemente sea la siguiente en la melodía, si eso es así se incrementa la duración de N_{cand} , si no, se establece la nueva nota como candidata. Si una nota nueva aparece en al menos dos segmentos consecutivos se da por terminada la nota anterior y se inicializa la verificación y conteo de segmentos para la nota candidata, iniciando con la duración que ya se ha registrado en $Duracand$. El uso de ésta nota candidata es un artificio para reducir la posibilidad de efectuar falsas identificaciones, ocasionadas principalmente por breves variaciones en el tono o por ruidos del ambiente, que normalmente no suelen durar más de unas fracciones de segundo.

Después de eso el ciclo se repite con los nuevos valores hasta que se proporciona la orden de parar el proceso de captura de datos.

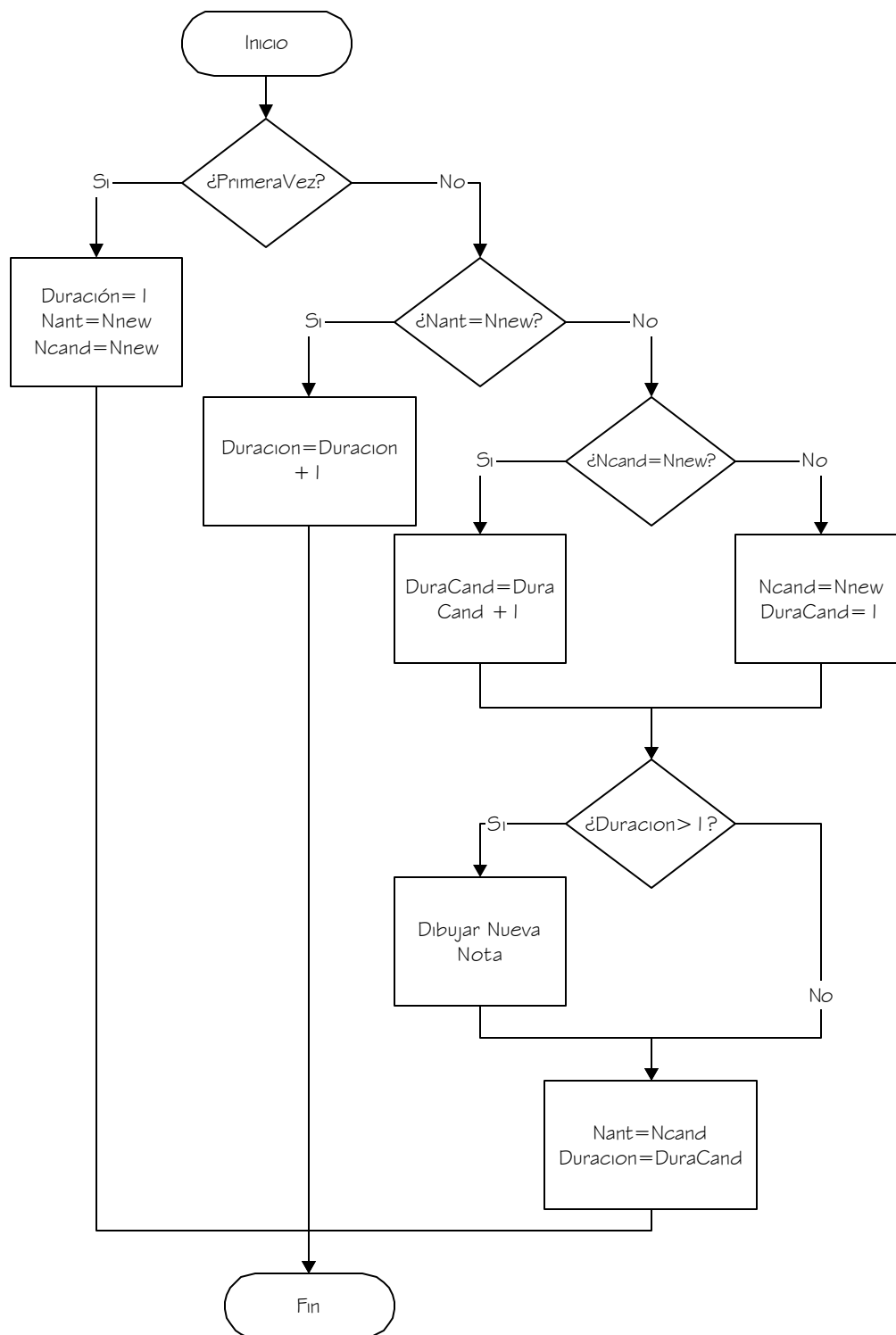


Figura 5-2 Algoritmo usado para determinar la duración de una nota musical

5.5 Desarrollo Del Pentagrama De Las Notas Ejecutadas

Cada vez que una nota está completamente identificada es enviada a las rutinas de creación del pentagrama, que la presentan en la posición y forma apropiada. Para la construcción y visualización en pantalla del pentagrama se utilizan una serie de métodos gráficos a fin de ofrecer un nivel apropiado de calidad del dibujo.

5.5.1 Generalidades

Cada nota identificada es almacenada en una matriz bidimensional que alberga tanto a la nota como a la duración de la misma, para la codificación del tono se utiliza la tabla 5-3

Nota	Valor	Nota	Valor
silencio	0	la#	11
do	1	si	12
do#	2	do 2	13
re	3	do 2#	14
re#	4	re 2	15
mi	5	re 2#	16
fa	6	mi 2	17
fa#	7	fa 2	18
sol	8	fa 2#	19
sol#	9	sol 2	20
la	10		

Tabla 5-3. Representación de las notas musicales

El tiempo de duración se indica en dieciseisavos de una redonda, es decir el número de semicorcheas que sería necesario unir para completar la duración de la nota. Por ejemplo una nota blanca en la posición de “la” se representa mediante (10,8) donde el 10 es el código correspondiente a “la” y 8 es el número de dieciseisavos que forman una blanca.

La información de la nota determina la ubicación del símbolo dibujado sobre el pentagrama, y su duración indica cual deberá ser ese símbolo. Es importante resaltar que, a excepción de la figura de la clave de sol, todo cuanto aparece en la hoja del pentagrama es creado a través de métodos gráficos, usándose principalmente los métodos *line* y *circle* para la mayor parte de símbolos. Para determinar la posición exacta de la nota dentro de la hoja se utilizan dos registros, el primero almacena la distancia horizontal desde el borde izquierdo de la página, el cual se utiliza además para pasar a la siguiente línea del pentagrama una vez que se ha agotado el espacio disponible en la línea actual, y el segundo permite saber en cual de los nueve bloques del pentagrama se dibujará la siguiente nota y en que momento es necesario cargar una nueva página para no interrumpir la captura y presentación de las notas.

Además de dibujar las notas es también necesario verificar cuando termina cada compás, para ello se acumulan en una sola variable las duraciones de cada nota y cuando esta suma alcanza el límite establecido para el compás de 4/4 se dibuja la línea divisoria y se verifica además si no se excederá ese límite, en caso de que se sobrepasen los 16/16 que puede albergar un compás es necesario el uso de una ligadura para dividir la última nota registrada entre dos compases diferentes.

5.5.2 Formato de archivo .ptg

Además de presentar el pentagrama en pantalla se ofrece la opción de almacenarlo en forma permanente en un archivo con un formato diseñado específicamente para esta aplicación, ya que incluye secciones que contienen toda la información presentada en pantalla, además de datos adicionales que brindarán la oportunidad de agregar mayor versatilidad en caso de que la aplicación sea utilizada como punto de partida para algún proyecto futuro. El formato empleado en los archivos creados por la aplicación se muestra en la tabla 5-4, junto con una breve descripción de cada uno de sus campos.

Bytes	Nombre	Contenido Usual	Descripción
1-2	t_cabecera	11	Tamaño de la cabecera del archivo
3	versión	1	Versión del formato del archivo
4	clave	“G”	Clave mostrada en el pentagrama
5-7	compás	“4/4”	Compás de la melodía
8-10	armadura	0	Información para la armadura
11	tempo	60	Velocidad de aparición de una negra expresado en BPM
12-13	t_titulo	?	Longitud del titulo expresado en caracteres
?	Titulo	?	Titulo de la pieza musical
?	t_penta	?	Cantidad de símbolos musicales presentes
?	penta	?	Notas almacenadas

Tabla 5-4. Formato de los archivos de pentagrama (.ptg)

5.5.3 Edición del pentagrama

El pentagrama puede sufrir modificaciones provenientes de tres diferentes fuentes, cada una de ellas con sus propias características.

1. Captura de notas. Eso ocurre cuando esta activada la detección e identificación de notas musicales. Al reconocer cada nota ésta es agregada al final de la matriz de pentagrama y el símbolo correspondiente es dibujado en el lugar que le corresponde.
2. Transposición de notas. En este caso todas las notas del pentagrama son modificadas a la vez, ya sea subiendo o bajando el tono de la melodía. Para ello basta con sumar o restar el número de semitonos que se desea transponer al código almacenado para cada nota.
3. Edición manual de notas. Éste es la forma que presenta la mayor cantidad de variaciones posibles, ya que cada nota puede ser modificada en forma individual, además es posible insertar y eliminar notas. Para efectuar todas esas acciones la

matriz completa es limpiada y rellena con los nuevos valores indicados por el usuario en el formulario de edición manual.

Un detalle que debe ser tomado en cuenta es que tanto para la transposición como para la edición manual el pentagrama debe ser dibujado desde el principio, a diferencia de la captura de notas que solo realiza modificaciones al final del pentagrama.

5.5.4 Operación en modo normal

La operación en modo normal es básicamente la captura e identificación continua de todas las notas registradas por el micrófono, según los procesos descritos en éste capítulo. Se adecua principalmente para el reconocimiento de música fluida, es decir sin pausas o vacilaciones excesivas.

5.5.5 Operación en modo de aprendizaje supervisado

El modo de aprendizaje supervisado abre un archivo de pentagrama existente que será usado como modelo y lo dibuja en un tono gris, para servir de indicación al usuario sobre qué se espera que ejecute. Luego se inicia la captura de notas, con la única diferencia de que el pentagrama no registra todas las notas identificadas, sino solamente aquellas que coincidan con el modelo elegido. Cuando eso ocurre la nota es agregada a la matriz de pentagrama y dibujada en color negro para indicar así el avance de la melodía.

Este modo es especialmente útil para aquellas personas que se encuentran aprendiendo a tocar la flauta dulce soprano, ya que permite al usuario verificar su forma de interpretar la flauta, además de la lectura de solfa, sin la necesidad de supervisión de alguien externo.

Capítulo 6. MANUAL DE USUARIO SiReNoM

6.1 *Presentación General.*

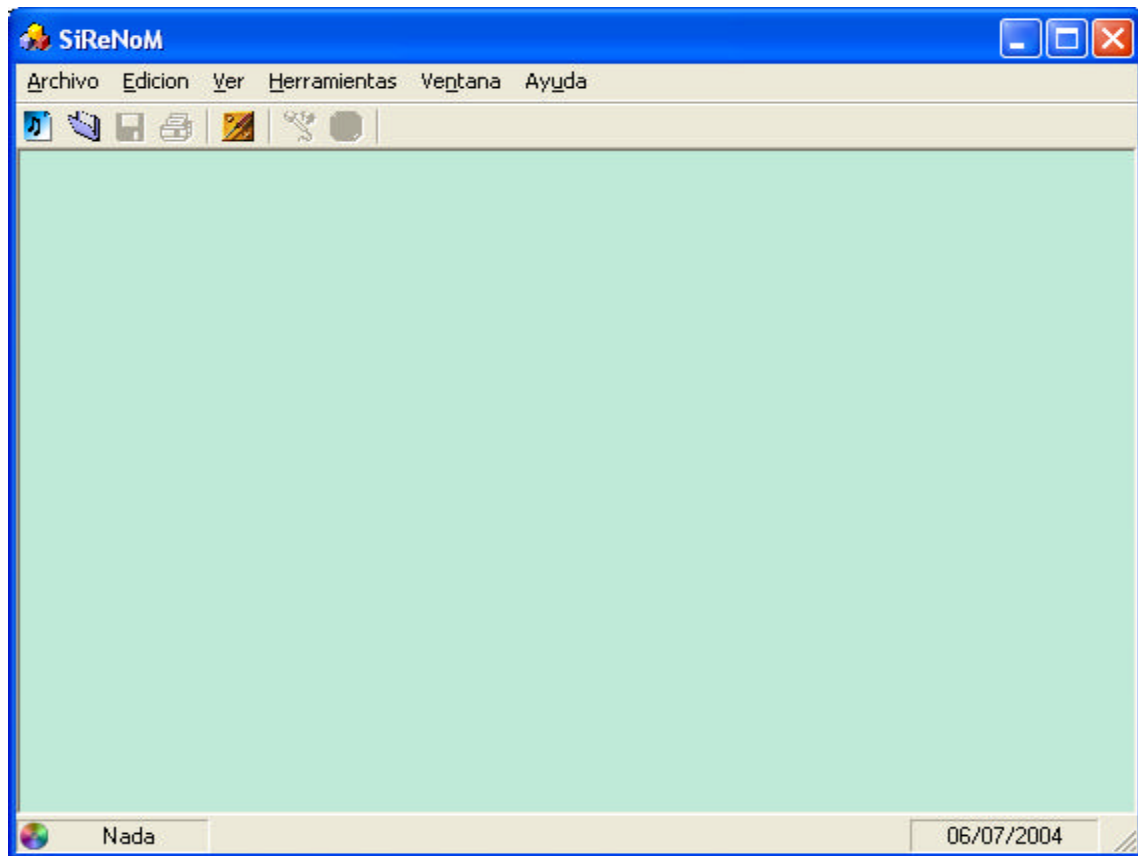


Figura 6-1 Pantalla principal del sistema

En la figura 6-1 se presenta la pantalla inicial de SiReNoM con las siguientes características principales:

- Barra de título
- Barra de menús
- Barra de herramientas
- Pentagrama
- Barra de estado

Barra de Título



Figura 6-2 Barra de título

La barra de título contiene el nombre del programa SiReNoM seguido del título asignado al pentagrama, que por defecto es Sin Titulo, en el extremo derecho de esta barra se encuentran los iconos para control de la ventana que sirven para minimizar, restaurar y cerrar.

Barra de Menús

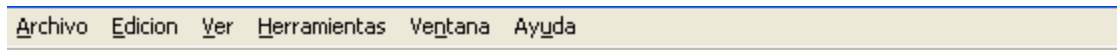


Figura 6-3 Barra de menús

En la barra de menús se encuentran todos los comandos que se pueden usar en el SiReNoM para su completo desarrollo, en el extremo derecho se tienen también los iconos para control de documento con los cuales se minimiza, restaura y se cierra solamente el pentagrama activo, en el dado caso de tener abiertos más de un pentagrama. Con estos controles no se puede cerrar el programa.

Menú Archivo

Contiene las opciones para manejar el archivo, abrir uno nuevo, abrir uno existente, guardar uno, imprimir y otras opciones tal y como se muestra en la figura 6-4

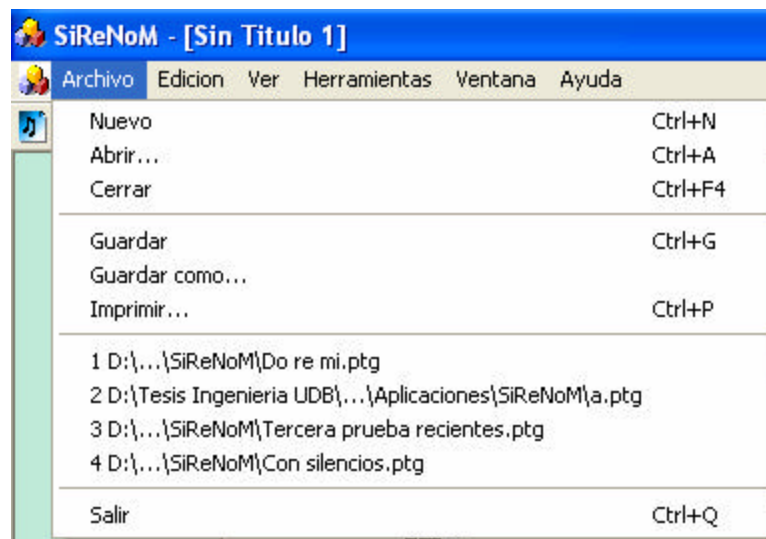


Figura 6-4. Menú Archivo

Menú Edición

En este menú se encuentran las opciones para hacer funcionar el proceso de grabación de audio y para detenerlo tal y como se ve en la figura 6-5

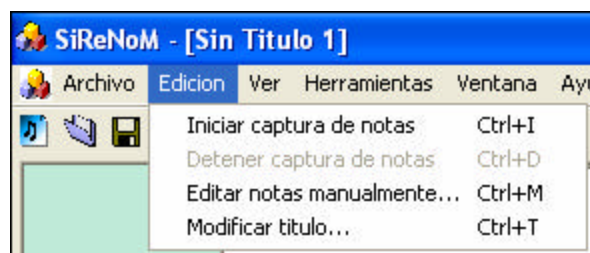


Figura 6-5. Menú edición

Menú Ver

En este menú se pueden controlar las barras que pueden estar presentes en la pantalla del programa tal y como se ve en la figura 6-6, la barra estará activa si la opción de ella está marcada con un cheque y estará oculta si no tiene el cheque marcado

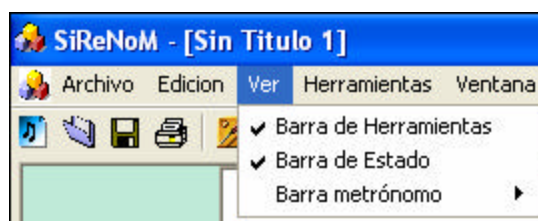


Figura 6-6. Menú Ver

Menú Herramientas

Este menú tiene la opción de poder subir y bajar tonos, tal como se ve en la figura 6-7, al seleccionar esta opción aparecerá un cuadro de diálogo como el de la figura 6-14 para poder realizar el cambio de tonos en las notas, subirlos o bajarlos. También permite tener acceso al modo de aprendizaje supervisado, herramienta especialmente útil para quien se inicia en la flauta dulce.



Figura 6-7. Menú Herramientas

Menú Ventana

En este menú se tienen las opciones para controlar las diferentes ventanas de los pentagramas que estén activos en un determinado momento como se ven en la figura 6-8, permitiendo ver un documento a la vez o tener varias ventanas abiertas al mismo tiempo



Figura 6-8. Menú Ventana

Menú Ayuda

En este menú se manejan las opciones de ayuda y de créditos del programa, tal y como se muestra en la figura 6-9.

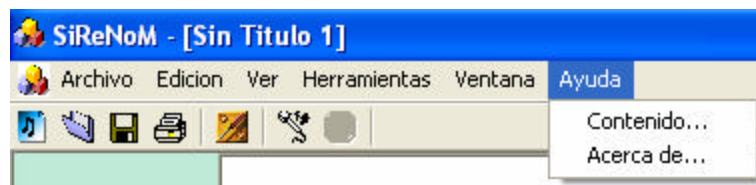


Figura 6-9. Menú Ayuda

6.2 Barra De Herramientas



Figura 6-10 Barra de herramientas

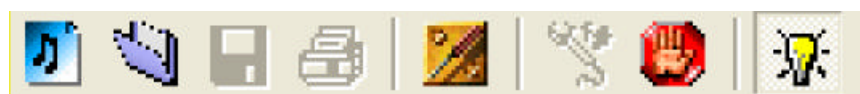


Figura 6-11 Barra de herramientas durante la adquisición

La barra de herramientas tiene todos los iconos para trabajar con el pentagrama, como se puede ver en la figura 6-10, en la figura 6-11 tiene un icono más que aparece cuando se está adquiriendo audio y desaparece cuando se detiene la grabación de audio.

NUEVO



Este icono se utiliza para abrir un nuevo pentagrama y permite abrir varios pentagramas a la vez, cada uno con el nombre Sin Título seguido de un número que indica cuantos pentagramas se han abierto.

ABRIR



Este icono se utiliza para abrir un pentagrama previamente guardado. Al dar clic en el se abre una ventana como la que se muestra en la figura 6-12

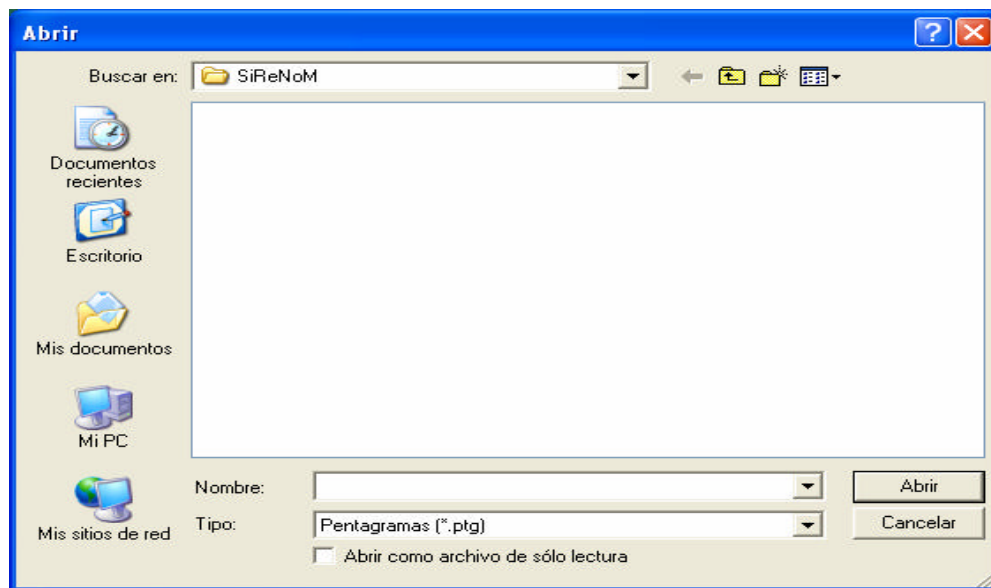


Figura 6-12. Ventana para abrir pentagramas previamente guardados

GUARDAR



Este icono se utiliza para guardar un pentagrama como un archivo, para poder usarlo posteriormente. Al dar clic sobre este icono aparece una ventana como la que se muestra en la figura 6-13, para poder guardar los pentagramas.

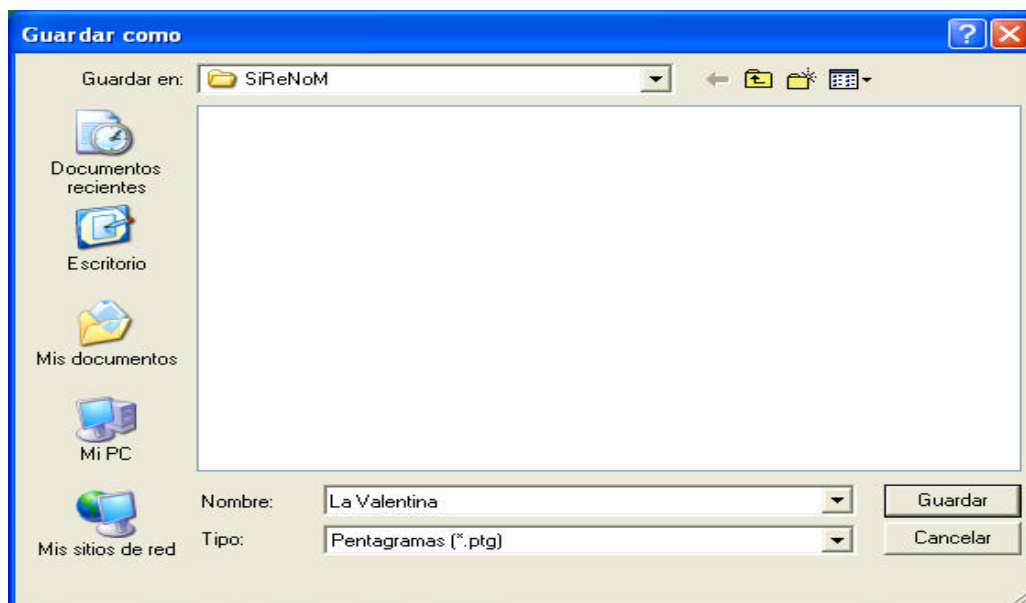


Figura 6-13. Ventana para guardar pentagramas



IMPRIMIR

Este icono se utiliza para imprimir el pentagrama que está en primer plano en la pantalla.



SUBIR/BAJAR TONOS

Al dar clic sobre este icono aparece una ventana como la de la figura 6-14 con la cual se pueden seleccionar el número de tonos que se quieren subir o bajar en el pentagrama.

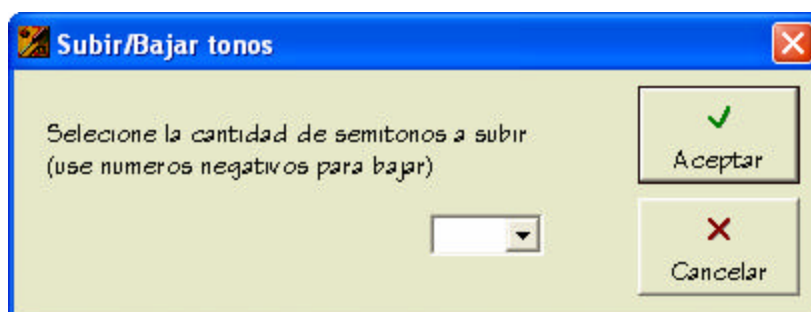
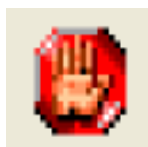


Figura 6-14. Cuadro de diálogo para subir o bajar tonos



INICIO GRABACIÓN

Este icono se utiliza para iniciar el proceso de adquisición de audio para reconocimiento de las notas musicales. Al dar clic en este icono se activa el icono para detener el proceso de grabación y aparece un nuevo icono, el icono del metrónomo.



DETENER GRABACIÓN

Este icono sirve para parar el proceso de grabación de audio, permanece inactivo hasta que se da clic sobre el icono de iniciar grabación.



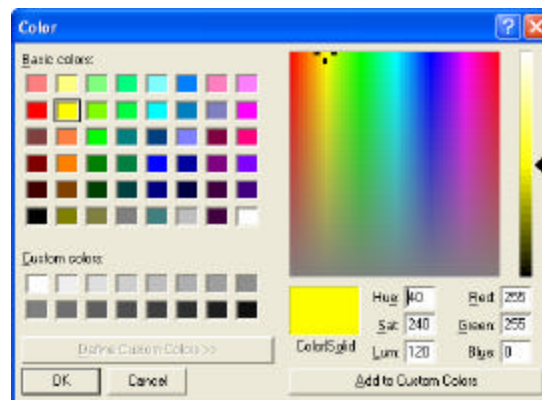
METRÓNOMO

Este icono tiene como funcionamiento proporcionar herramienta para medir el tiempo. Cuando se inicia la grabación de audio automáticamente este icono aparece y empieza a parpadear en intervalos de un segundo, es decir el tiempo de duración de una negra.

Junto con ese icono el sistema presenta un marco que proporciona una forma mucho más sencilla de percibir el ritmo sin perder de vista el pentagrama. Los lados de ese marco pueden ocultarse y mostrarse desde el menú *Ver*→*Barra metrónomo*, tal como se aprecia en la figura 6-15 a.



a)



b)

Figura 6-15 Barras de metrónomo a) Visualización b) Color a usarse

Si se desea modificar el color de las barras se puede hacer desde el menú *Herramientas*→*Opciones* y escoger el botón *color de barras de metrónomo*, luego se selecciona el color deseado en el cuadro de dialogo mostrado en la figura 6-15 b.

6.3 Pentagrama

El cuerpo del documento tiene dos partes que son:

Título

Pauta o Pentagrama

6.3.1 Título



Figura 6-16 Título del pentagrama

En el título del documento (Figura 6-16) se define el nombre que se le asigna al pentagrama, para cambiarlo se hace doble-clic sobre el título del mismo, apareciendo un cuadro de diálogo (Figura 6-17) en el cual se puede introducir el nuevo nombre para identificar de mejor forma el pentagrama.

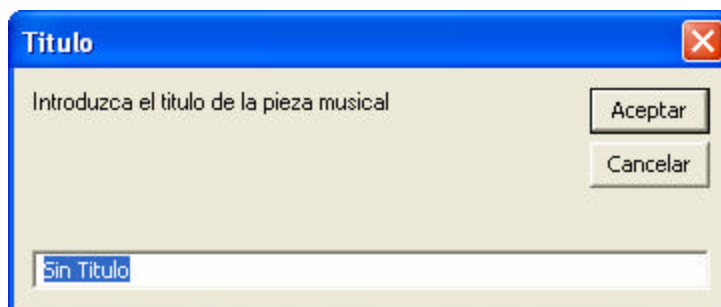


Figura 6-17. Cuadro de diálogo para cambiar nombre de pentagrama

6.3.2 Pauta o Pentagrama

En la pauta o pentagrama (Figura 6-18) es donde se irán colocando las diferentes notas musicales que el usuario esté ejecutando, siempre y cuando se haya iniciado el proceso de grabación de audio. El pentagrama se puede modificar por ejecución de la flauta, por uso de la opción de bajar o subir tonos o a través de la edición manual de las notas.

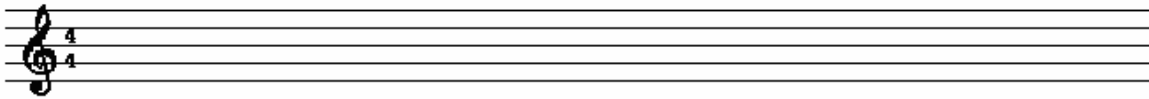


Figura 6-18. Pauta o pentagrama

6.4 Barra De Estado

Muestra el estado actual del sistema. Es un medio de brindarle información al usuario de lo que está sucediendo en ese momento.

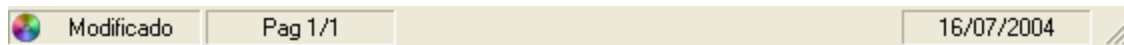
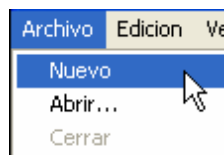


Figura 6-19. Barra de estado

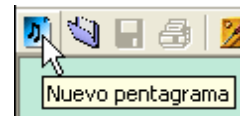
6.5 Crear Un Nuevo Pentagrama

Una de las tareas más comunes que se llevan a acabo con este sistema es la creación de un nuevo pentagrama en blanco, para, posteriormente, iniciar la captura de las notas ejecutadas. La forma de hacerlo se muestra a continuación:

1º Seleccione *Archivo* → *Nuevo* o presione el botón Nuevo en la barra de herramientas como se muestra en la figura 6-20, o bien presione *ctrl.+N*



a)



b)

Figura 6-20 Nuevo pentagrama a) Menú archivo b) Barra de herramientas

Con cualquiera de las alternativas presentadas aparecerá un pentagrama en blanco listo para ser utilizado.

6.6 Cambiar El Título De Un Pentagrama

Por defecto aparece la frase “Sin título” en la parte superior de la primera hoja de un nuevo pentagrama, pero eso puede no ser muy descriptivo para fines de almacenamiento o referencia futura, por ello es conveniente colocar un título descriptivo de la melodía ejecutada. Para modificar el título del pentagrama actual vaya a la barra de menú y seleccione *Edición* → *Modificar título*

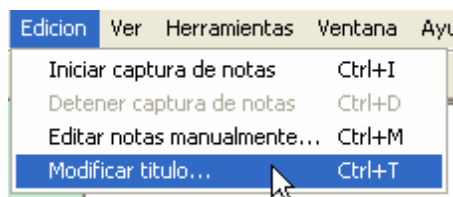


Figura 6-21 Cambiar el título de un pentagrama

O haga doble-clic sobre el título que aparece en el pentagrama, o bien, presione ctrl. +T. Con ello aparecerá un cuadro de diálogo donde podrá introducir el título de la pieza musical que está registrando.

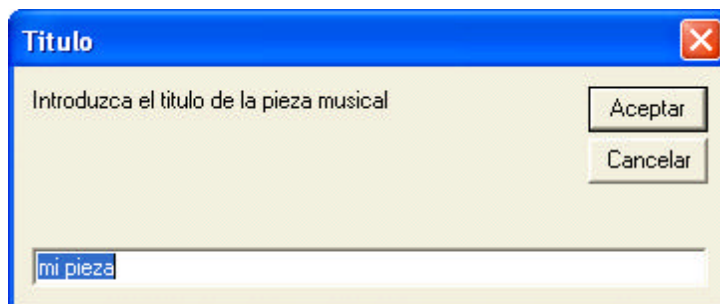


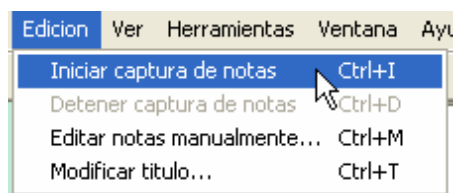
Figura 6-22 Cuadro de diálogo Cambiar el título de un pentagrama

6.7 Crear Un Pentagrama A Partir De Las Notas Ejecutadas

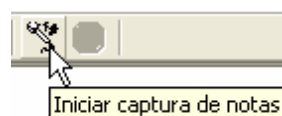
Para poder registrar en forma de pentagrama el sonido de las notas emitidas por la flauta dulce debe seguir estos pasos:

1º- Cree un documento en blanco o abra uno ya existente

2º- Seleccione *Edición* → *Iniciar captura de notas...* (Figura 6-23.a) o presione el botón Captura de notas en la barra de herramientas (Figura 6-23.b)



a)



b)

Figura 6-23 Crear un pentagrama a partir de notas ejecutadas

O bien, presione *ctrl.+I*.

3º- Ejecute la melodía deseada, ayudándose del ritmo marcado por el metrónomo para obtener mejores resultados

4º- Una vez que haya concluido la ejecución de la melodía debe detener la captura de sonido, para ello seleccione *Edición* → *Detener captura de notas* (Figura 6-23a) o presione el botón Detener captura en la barra de herramientas (Figura 6-24) o bien, presione *ctrl. +D*

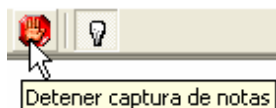


Figura 6-24 Detener captura de notas

5º- Ahora su pentagrama aparecerá en la pantalla y puede modificarlo, imprimirlo o almacenarlo para usarlo posteriormente.

6.8 Realizar Transposición De Notas

Transportar es ejecutar o transcribir un trozo de música en otro tono distinto de aquel en que está originalmente escrito. El objetivo de la transposición es poner en una tonalidad conveniente a un instrumento una pieza musical escrita muy alta o muy baja para este instrumento. Dentro del Sistema de Reconocimiento de Notas Musicales ésta es una tarea muy sencilla, basta con seleccionar *Herramientas*→ *Subir/Bajar tonos...* o presione el botón de Transposición en la barra de herramientas (Figura 6-25)

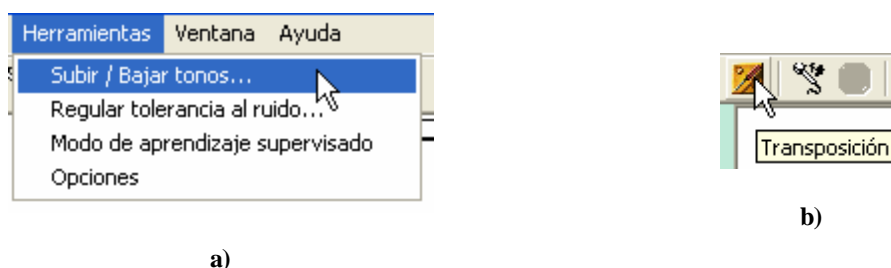


Figura 6-25 Subir y Bajar tonos a) menú herramientas b) barra de herramientas

Aparecerá entonces un formulario con un cuadro de lista desplegable, donde podrá determinar el número de semitonos a subir toda la pieza musical, si su objetivo es bajar determinada cantidad de semitonos bastará con escoger una cifra negativa en la lista que aparece en el formulario.

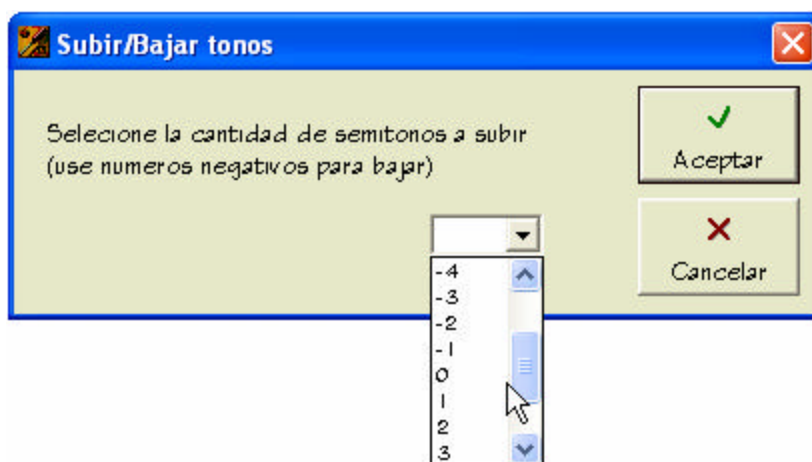


Figura 6-26 Transposición de notas

Luego presione Aceptar y la transposición se realizará en forma automática.

6.9 Regular La Tolerancia Al Ruido

El funcionamiento de este sistema es muy sensible al ruido del ambiente, por lo que deberá utilizarse en un recinto preferiblemente aislado, o por lo menos, sin sonidos estridentes o irregulares. También es una buena opción regular el nivel esperado de ruido que será captado por el micrófono. Para ello vaya a la barra de menú y seleccione *Herramientas*→*Regular tolerancia al ruido...* como se indica en la figura 6-27, con lo que aparecerá un control deslizante idéntico al de la figura 6-28, a través del cual puede controlar el valor deseado. El nivel se indica como un porcentaje del nivel predeterminado, el valor por defecto es 100%.

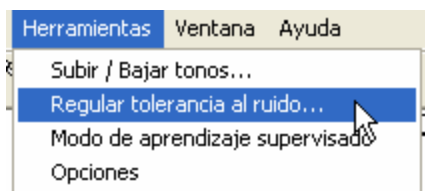


Figura 6-27 Regular la tolerancia al ruido

Al presionar el botón Aceptar se establece el nuevo valor, y con el botón Cancelar se ignora cualquier cambio efectuado.



Figura 6-28 Regulador de tolerancia al ruido

6.10 Editar las notas en forma manual

En algunas ocasiones el tiempo de duración de una nota señalado por el sistema, o bien, el mismo valor de la nota ejecutada, no coincide completamente con lo que se pretendía tocar, situación especialmente cierta si durante la ejecución de la melodía se percibieron sonidos ajenos a los emitidos por la flauta, por lo que pueden observarse pequeñas incongruencias entre las notas interpretadas y el pentagrama resultante.

Esas inconsistencias pueden ser fácilmente eliminadas gracias a la poderosa herramienta de edición manual del pentagrama. Para acceder a ella diríjase al menú *Edición* y ubique el ítem *Editar notas manualmente...* o presione *ctrl. +M* desde el teclado.

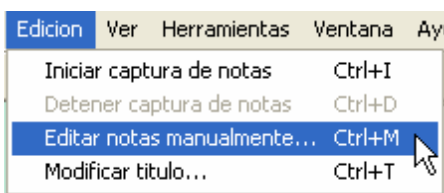


Figura 6-29 Editar notas manualmente

Aparecerá un formulario de edición como el siguiente:

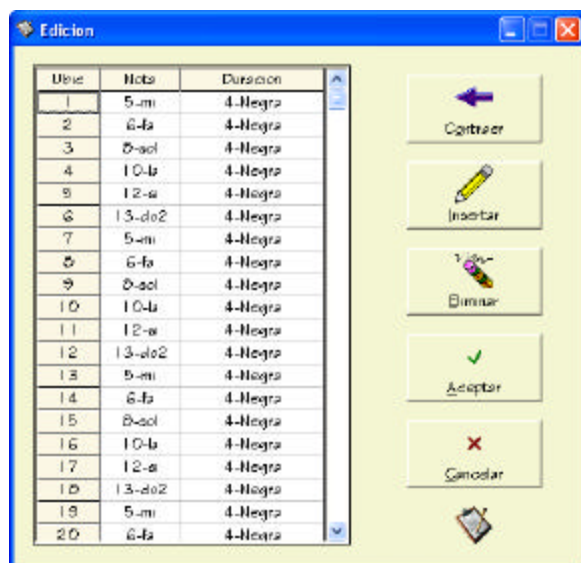


Figura 6-30 Cuadro de edición manual de notas

Este formulario posee varios botones que facilitan la tarea de edición



Contraer: Este botón reduce el tamaño del formulario de edición a fin de visualizar un área mayor del pentagrama que se está editando.



Insertar: Con el botón Insertar se añade un espacio en blanco en la lista de notas del pentagrama en el punto donde se encuentra el cursor.



Eliminar: Permite eliminar la nota actual.



Aceptar: Vuelve permanentes todos los cambios realizados



Cancelar: Descarta todos los cambios realizados y deja el pentagrama tal como estaba antes de iniciar el proceso de edición

Para modificar una nota cualquiera es necesario hacer clic sobre la fila en la que se encuentra y se escogen los valores adecuados al tipo de nota y su duración en los cuadros de lista desplegable que aparecen sobre la nota escogida.

5	1 2-si	4-Negra
6	1 3-do2	1-Semicorchea
7	5-mi	2-Corchea
8	6-fa	3-CorcheaPuntillo
9	8-sol	4-Negra
10	1 0-la	6-NegraPuntillo
11	1 2-si	8-Blanca
12	1 3-do2	1 2-BlancaPuntillo
		1 6-Redonda

Figura 6-31 Edición manual de una nota

Una vez que haya realizado los cambios necesarios presione Aceptar.

6.11 Utilizar El Modo De Aprendizaje Supervisado

En algunas ocasiones se puede ser deseable verificar la exactitud de las notas ejecutadas, tarea no tan sencilla para un individuo que no cuenta con la ayuda de un profesor de música al momento de realizar sus ejercicios. Por tanto se ha incluido una herramienta de modo supervisado que permite al estudiante de la flauta dulce alcanzar un nivel aceptable en la forma de ejecución del instrumento.

El principio básico de este modo es que todo estudiante de música se repite una y otra vez melodías ya existentes hasta alcanzar un alto grado de habilidad. Por ello se usa un pentagrama ya existente que se toma como modelo a seguir, de modo que si las notas ejecutadas no coinciden con el modelo propuesto, el nuevo pentagrama no avanzará y esperará hasta que se detecte en forma correcta la nota esperada.

Para acceder a este modo basta con dirigirse al menú *Herramientas* → *Aprendizaje supervisado* como se muestra en la figura 6-32

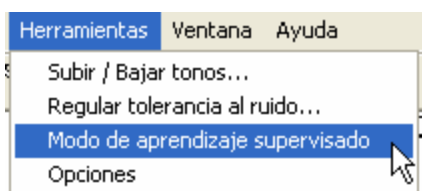


Figura 6-32 Modo de aprendizaje supervisado

Luego debe seleccionarse el pentagrama que se usará como modelo, en un cuadro similar al de Abrir archivos. El pentagrama escogido se presentará en pantalla en forma atenuada a fin de que sirva de guía visual sobre lo que debe tocarse. El sistema activará en forma automática la captura de audio, y a medida que las notas vayan siendo correctamente identificadas serán añadidas en la forma tradicional al pentagrama, tal como se muestra en la figura 6-33, donde puede observarse el progreso de la identificación de las notas, que hasta ese momento ha llegado al segundo tiempo del tercer compás.

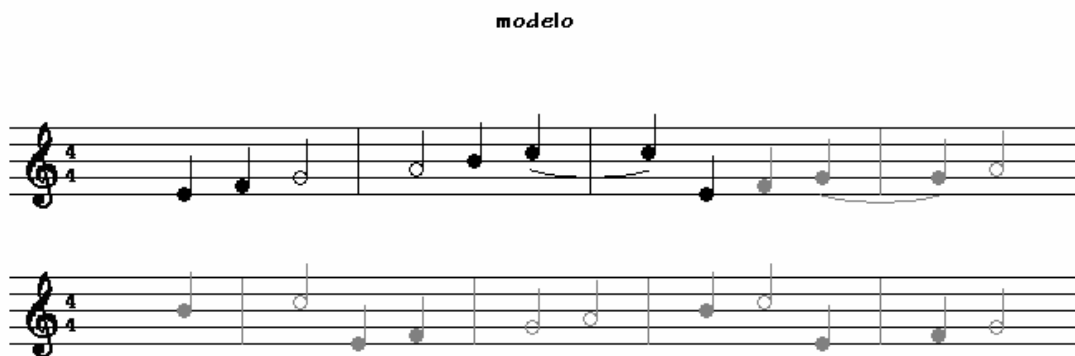


Figura 6-33 Ejecución en modo de aprendizaje supervisado

Esta modalidad permite al usuario verificar sus progresos en la ejecución del instrumento sin la necesidad de la presencia constante de un maestro de música que le indique cuando no está ejecutando correctamente una melodía.

NOTA: Si el pentagrama con el que se quiere trabajar no se encuentra entre los guardados anteriormente, se puede editar manualmente el nuevo pentagrama tal y como se describe en el numeral 6.10, guardarlo y después utilizarlo siguiendo los pasos descritos en este mismo numeral.

6.12 Requerimientos Del Sistema

Para conseguir un desempeño adecuado del sistema se establecen los siguientes requisitos mínimos tanto en hardware como en software

6.12.1 Requerimientos de hardware y software

Windows 2000 pro

100 MB libres en HD

128 MB de RAM

Microprocesador 1 GHz

Monitor SVGA

Tarjeta de sonido con capacidad de grabación.

Micrófono monocanal

6.12.2 Características del sistema

- SiReNoM está diseñado para trabajar únicamente a 60 bpm en un compás de 4/4.
- La duración mínima para que una nota sea reconocida como tal es de 250 milisegundos, es decir la duración de una semicorchea.
- La duración máxima que puede registrar el sistema es de un poco más del tiempo de dos redondas u 8 segundos, pero la edición manual tiene como límite superior una redonda, que equivale a 4 segundos exactos.
- La separación mínima entre dos notas consecutivas del mismo tono varia entre la duración de una semifusa y una fusa, 62.5 y 125 milisegundos respectivamente, el valor exacto dependerá de las condiciones acústicas del sitio donde se use y del nivel a que se haya establecido la tolerancia al ruido.
- El silencio más corto que es registrado como tal en el pentagrama es el silencio de corchea, 500 milisegundos, y el más largo es el de una redonda.
- El modo de aprendizaje supervisado es el que presenta el mejor desempeño al momento de identificar notas aisladas o con pausas al momento de su ejecución, ya que no toma en cuenta aquellas notas o silencios si éstos no concuerdan con la figura presentada en el pentagrama que se toma como modelo.
- El modo normal de operación está pensado para el reconocimiento de piezas musicales ejecutadas de manera fluida, es decir es una herramienta para personas que hayan alcanzado un nivel aceptable de habilidad en la interpretación de música para flauta dulce soprano.
- Si el programa se ejecuta en presencia de ruido en la hoja del pentagrama aparecerán notas que no corresponden a las ejecutadas sino a las coincidencias armónicas del ruido percibido y las notas de la flauta dulce, independientemente de la presencia de verdaderas notas musicales. Esto es especialmente cierto para el modo normal de funcionamiento, donde el desempeño del sistema puede verse disminuido hasta en un 50 % o incluso más en los casos donde el ruido o las voces tienen presencia excesiva.

6.13 Proceso De Instalación Del Sistema

La instalación del programa de aplicación es una tarea muy sencilla, que no requiere ningún tipo de configuración compleja o un número elevado de opciones. Esta tarea es similar a la que se lleva a cabo para la mayor parte de las aplicaciones en Windows, presentando inicialmente una recomendación al usuario antes proceder al proceso de instalación, tal como se muestra en la figura 6-34

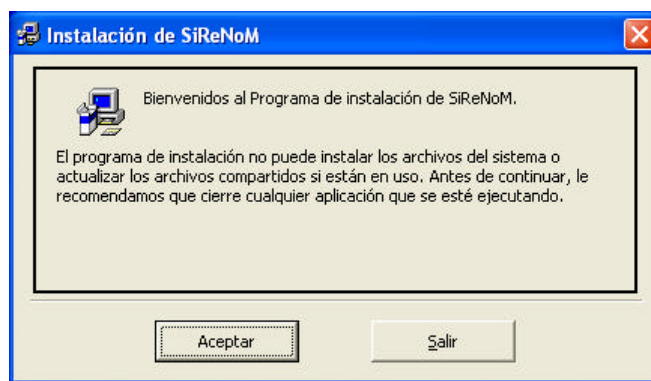


Figura 6-34 Pantalla de inicio de instalación

Luego se solicita al usuario la ubicación en la cual desea instalar el programa. Para ello se da la opción de modificar la ruta por defecto, o bien, de continuar con la que el sistema propone en la carpeta C:\Archivos de Programa\SiReNoM.

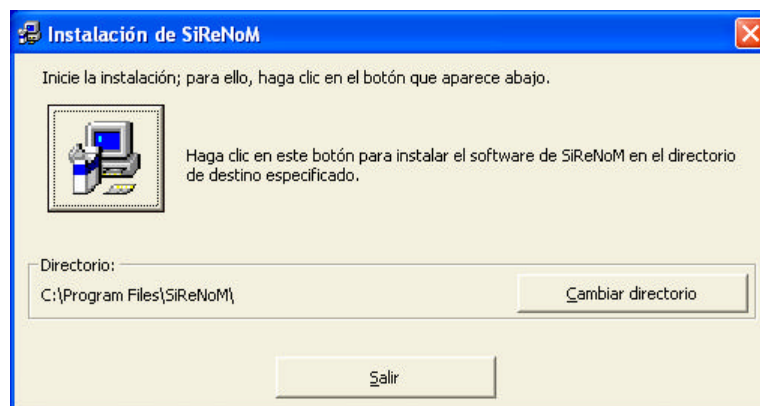


Figura 6-35 Ruta de instalación

A continuación se pregunta al usuario en que grupo de programas del menú Inicio desea que se cree el acceso directo al sistema. Se recomienda crear un grupo nuevo con el mismo nombre de la aplicación.

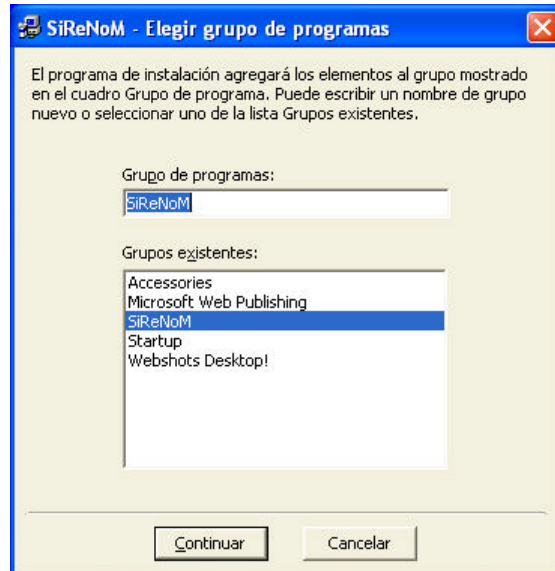


Figura 6-36 Elegir grupo de programas

Después de eso se copian automáticamente todos los archivos necesarios para el funcionamiento del sistema, indicando en forma visual el progreso de los mismos.

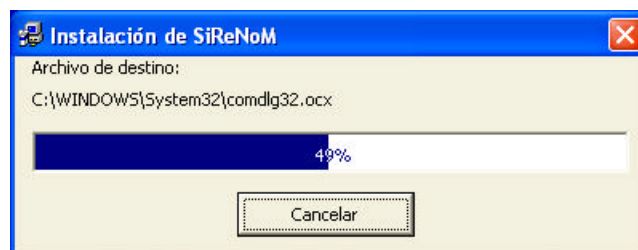


Figura 6-37 Progreso de la instalación

Al concluir todo el proceso se despliega un mensaje indicando que no se presentaron problemas, y en algunos casos, se pedirá que se reinicie el equipo.

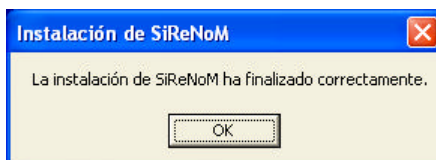


Figura 6-38 Fin de la instalación

6.14 Pruebas Del Sistema

Para poder saber que tan bien responde el sistema a los diferentes usuarios que este pudiera tener, se hicieron pruebas con diferentes personas observando el comportamiento del sistema ante cada una de ellas, en este caso se utilizaron individuos que podían tocar la flauta dulce soprano, al mismo tiempo estas pruebas tuvieron diferentes entornos ambientales para ver como respondía también ante ellos. Dentro de estos entornos se pueden mencionar salas y cuartos de casa, aulas, estudio de música. La prueba constaba de dos partes la primera era probar el sistema en modo normal y la segunda era probar el sistema en el modo supervisado.

Para las pruebas además se utilizaron diferentes tipos de micrófono para la adquisición de la música. Los resultados obtenidos se muestran en la tabla 6.1

LOCAL	TIPO MICRÓFONO	% EFECTIVIDAD MODO NORMAL	% EFECTIVIDAD MODO SUPERVISADO
SALA	PC	40%	85%
SALA	MUSICAL	65%	95%
CUARTO	PC	80%	95%
CUARTO	MUSICAL	95%	100%
AULA	PC	80%	95%
AULA	MUSICAL	-	-
ESTUDIO	PC	95%	100%
ESTUDIO	MUSICAL	100%	100%

Tabla 6-1 Resultados de validación del sistema

Capítulo 7. CONCLUSIONES

El formato de archivos wav ofrece un bajo nivel de procesamiento para almacenar y recuperar las notas musicales, sin embargo existen diferencias en el tamaño y velocidad de adquisición de las muestras de audio obtenidas al efectuar grabaciones, que dependen tanto de las características de la tarjeta de sonido como de la configuración y capacidades del sistema, por lo que se vuelve indispensable determinar la forma en que se ha dado el muestreo antes de poder procesar la información contenida en ellos.

El uso de redes neuronales para la identificación de las notas otorga un alto grado de plasticidad a la aplicación (como fue comprobado con diferentes fuentes de sonido), lo que es una característica deseable dado el amplio rango de variación que puede presentarse en la captación y digitalización de sonido.

En base a los resultados obtenidos con los diferentes tipos de redes neuronales se seleccionó la red ART2 con patrones de entrada en el dominio de la frecuencia por ser la topología que presentó el más alto grado de eficacia al momento de reconocer las notas.

Una enorme cantidad de sonidos interferentes está presente en el espectro de frecuencias que corresponde a las de la flauta dulce soprano, por lo que la tarea de identificación de una nota se vuelve altamente compleja, pues se deben suprimir las variaciones producidas por el sonido ambiente sin, por ello, crear un sistema rígido que no responda adecuadamente a las más sutiles variaciones en la ejecución musical.

Para reducir las interferencias provenientes del espectro de frecuencias que está fuera del rango de las notas musicales, que serán identificadas, se limitó el procesamiento de los datos a las componentes de frecuencia correspondientes a la zona de mayor concentración energética para el segmento de la escala musical sujeto de reconocimiento.

La definición de un nuevo formato de archivo no es una tarea sencilla, ya que es deseable ofrecer la posibilidad de incluir información que al momento del diseño no estaba contemplada, lo que permitirá, en el futuro, incluir mayores capacidades al sistema, tales como diferentes tipos de compás, velocidades de ejecución, etcétera.

El intervalo de separación entre notas repetidas no es un valor fijo, sino que varía y depende totalmente del ejecutante, por lo que ofrece una enorme dificultad al momento de su reconocimiento, lo que se traduce en una pérdida de precisión en el proceso de definición del pentagrama.

El hecho de que haya una relación exponencial (base 2) en los posibles valores que puede tomar la duración de una nota y haber definido previamente el tempo de las melodías permitió manejar un pequeño margen de tolerancia en la fase de determinación del tiempo de duración de las notas ejecutadas, con lo que se logra una mayor eficacia en el reconocimiento de la duración de las notas

El sonido presente en el ambiente donde se utiliza el sistema influye en gran medida en la capacidad de distinguir una nota de otra, por lo que es preferible que sea utilizado en un local silencioso, libre de perturbaciones externas tales como voces, golpes y vibraciones.

En base a las pruebas realizadas en diferentes ambientes y tipos de micrófono se puede afirmar que la utilización de un micrófono especial para instrumentos musicales mejora la eficiencia del sistema, sobre todo en ambientes silenciosos y cerrados.

Capítulo 8. RECOMENDACIONES

1. La separación de fuentes independientes de audio de una muestra que es una mezcla de datos es un desafío muy importante en el procesamiento de señales, en muchas aplicaciones, así como en la presente, es necesario separar una o más señales deseadas de una mezcla de señales presentes en la muestra de datos. Es por eso que se presentan las siguientes opciones para eliminar interferencias de audio que se encuentran en la misma frecuencia que nuestro rango de frecuencias utilizables
 - a. Desarrollar un sistema que permita eliminar el ruido. En base a un algoritmo que permita reconocer y discriminar los sonidos emitidos por cualquier sistema diferente a la flauta dulce. En base a consideraciones estadísticas como con el algoritmo Blind Signal Separation (BASS). Este algoritmo cubre muchas aplicaciones como separación de orígenes musicales de alta calidad, mejoramiento de señal de habla, reconocimiento de habla y otras. Para mayor información sobre este algoritmo se puede visitar las siguientes páginas Web:
http://www.egr.msu.edu/annweb/papers/blind_separation
<http://bach.ece.jhu.edu/ica/iscas99/tutorial.html>
<http://www.irisa.fr/metiss/gribonval/Conf/2003/ICA/0085.pdf>
 - b. Otra posible forma de tratar el problema sería basarse en el timbre de las notas emitidas por la flauta y en base a otro sistema de red neuronal se pueda diferenciar los timbres de las notas musicales, el timbre es la cualidad que permite distinguir los sonidos producidos por los diferentes instrumentos o por la voz, y en base a ese reconocimiento pasar al reconocimiento de que tipo de nota musical se trata. Para lograr hacer esto es necesario hacer un estudio de las formas de onda de cada nota musical en función del tiempo puesto que para cada una se tiene una forma de onda diferente y que también es diferente a la misma nota musical pero emitida por otro instrumento o por la voz.

Para obtener mayor información sobre esta propuesta se puede visitar las siguientes páginas Web:

<http://www.xtec.es/centres/a8019411/caixa/>

<http://waynesweb.ualr.edu/MUSICFUND.htm>

2. Para permitir la universalidad y transportabilidad de los archivos de pentagrama puede modificarse el formato actual (.ptg) y utilizarse un formato normalizado como el XML. Un formato sugerido para ello se muestra a continuación. El ejemplo muestra un archivo con su definición de tipo de documento contenida en el interior del mismo. Los elementos *bemol* y *sostenido* deben poder aparecer ninguna, una o más veces, ya que las diferentes escalas musicales poseen indicaciones de bemol y sostenido en distintas posiciones dentro de su armadura, y están ya sea en una línea o en un espacio del pentagrama. El archivo mostrado contiene la misma información señalada en la sección 5.5.2. Además se incluyen, a manera de ejemplo, los valores de tres notas, do re mi, con una duración equivalente a una negra.

```
<?xml version = "1.0" standalone = "yes"?>
<!-- formato sugerido para almacenar pentagramas -->
<!-- generados por SiReNoM 1.0.0 -->

<!DOCTYPE pentagrama [
    <!ELEMENT version_archivo ( #PCDATA )>
    <!ELEMENT clave ( #PCDATA )>
    <!ELEMENT armadura (bemol*, sostenido* )>
    <!ELEMENT bemol ( #PCDATA )*>
        <!ATTLIST bemol linea_espacio ( LINEA | ESPACIO ) "LINEA">
    <!ELEMENT sostenido ( #PCDATA )*>
        <!ATTLIST sostenido linea_espacio ( LINEA | ESPACIO ) "LINEA">
    <!ELEMENT compas EMPTY >
        <!ATTLIST compas n_tiempos CDATA #REQUIRED>
        <!ATTLIST compas figura CDATA #REQUIRED>
    <!ELEMENT tempo EMPTY>
        <!ATTLIST tempo bpm CDATA #REQUIRED>
    <!ELEMENT titulo_melodia ( #PCDATA )*>
    <!ELEMENT figuras ( simbolo+ )>
    <!ELEMENT simbolo EMPTY>
        <!ATTLIST simbolo nota CDATA #REQUIRED>
        <!ATTLIST simbolo duracion CDATA #REQUIRED>
]>
```



```

<pentagrama>

  <version_archivo>1</version_archivo>

  <clave>G</clave>

  <armadura>
    <bemol></bemol>
    <sostenido></sostenido>
  </armadura>

  <compas n_tiempos = "4" figura = "4"/>
  <tempo bpm = "60"/>

  <titulo_melodia>do re mi</titulo_melodia>

  <figuras>
    <simbolo nota = "1" duracion = "4"/>
    <simbolo nota = "2" duracion = "4"/>
    <simbolo nota = "3" duracion = "4"/>
  </figuras>

</pentagrama>

```

Para obtener información adicional sobre el formato xml pueden visitarse los siguientes sitios relacionados:

<http://www.w3.org/XML>

<http://www.xml.org>

<http://msdn.microsoft.com/xml>

<http://www.oasis-open.org/cover>

3. Para poder utilizar diferentes tipos de compás, a fin de adaptarlo a diferentes ritmos musicales, haría falta cambiar el tamaño de los archivos de audio capturados de modo que coincidan con la figura indicada por la cifra inferior del compás, así como el valor límite usado para determinar la duración de un compás.
4. La velocidad del metrónomo puede ser ajusta en forma independiente de la captura de audio, pero para garantizar que cumpla con su función debe estar sincronizado con ese proceso, esto es especialmente cierto si se desea reducir el periodo de operación del mismo.

5. Para proporcionar un uso más cómodo del metrónomo al usuario final se puede incluir un efecto sonoro que indique cada uno de los tiempos dentro del compás, asegurándose de que el sonido empleado no sea registrado por el micrófono del sistema ya sea por el uso de una onda pura que esté alejada de las notas de la flauta o bien mediante la utilización de audífonos.

Capítulo 9. BIBLIOGRAFÍA

- [1] Fausett L. Fundamentals of Neural Networks. Prentice Hall. USA. 1994.
- [2] Hilerá José R. y Martínez Víctor J. Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones. Editorial Addison-Wesley Iberoamericana 1995
- [3] James Freeman / David Skapura. Redes Neuronales. Algoritmos, aplicaciones y técnicas de Programación. Editorial Addison-Wesley 1993. Primera Edición
- [4] Eduardo Rivera, Francisco Rodríguez y Jonathan Rodríguez Reconocimiento de Patrones Cardíacos Utilizando Redes Neuronales Documento de Tesis año 2000
- [5] Chen C. H. Fuzzy Logic and Neural Network Handbook. McGraw Hill. USA. 1996.
- [6] Principe J. C. Neil E. Neural and Adaptive Systems: Fundamentals through Simulations. Editorial John Wiley. USA. 2000.
- [7] Kamas A. Lee E. Digital Signal Processing Experiments. Prentice Hall. USA. 1989.
- [8] Foularikas A. Seely S. Signals and Systems. Press publisher. USA. 1985.
- [9] Introduction to Digital Signal Processing.
- [10] Bodenmann, H. Pahlen, K. El ABC de la flauta dulce. Editorial Melodie. Suiza.
- [11] Danhauser, A. Teoría de la música. Ediptasol. México. 2001

[12] Monzon Herrera, M. Monzon Herrera, L. Juguemos a tocar la flauta dulce.
Editapsol. México 1999

[13] Pahlen, K. Síntesis del saber musical. Editorial Emece. Argentina 1957

[14] http://www.iiia.csic.es/~mario/rna/tutorial/RNA_intro.html.
(Última visita: abril 2004)

[15] <http://www.emsl.pnl.gov:2080/proj/neuron/neural/courses.html>
(Última visita Abril 2004)

[16] <http://www.monografias.com/trabajos/redesneuro/redesneuro.html>
(Última visita: abril 2004)

[17] <http://usuarios.lycos.es/sonymusica>. (Última visita: junio 2004)

[18] <http://www.guitarraonline.com.ar>. (Última visita: junio 2004)

[19] <http://www.cancionero.com.ar>. (Última visita: mayo 2004)

[20] <http://venus.javeriana.edu.co/~cproy/DisPro/Nuevos/srtm/default.htm>.
(Última visita: julio 2004)

[21] <http://www.ii.uam.es/~taao/practica/practica3.html>.
(Última visita: julio 2004)

[22] <http://www.upv.es/protel/index.html>. (Última visita: mayo 2004)

[23] <http://www.fortunecity.com/tinpan/supergrass/657/articulos.htm>.
(Última visita: abril 2004)

ANEXOS

A. CÓDIGO FUENTE

```
about
Const READ_CONTROL = &H20000
Const KEY_QUERY_VALUE = &H1
Const KEY_SET_VALUE = &H2
Const KEY_CREATE_SUB_KEY = &H4
Const KEY_ENUMERATE_SUB_KEYS = &H8
Const KEY_NOTIFY = &H10
Const KEY_CREATE_LINK = &H20
Const KEY_ALL_ACCESS = KEY_QUERY_VALUE + KEY_SET_VALUE + _
                        KEY_CREATE_SUB_KEY + KEY_ENUMERATE_SUB_KEYS + _
                        KEY_NOTIFY + KEY_CREATE_LINK + READ_CONTROL

Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1
Const REG_DWORD = 4

Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"

Private Declare Function RegOpenKeyEx Lib "advapi32" Alias
"RegOpenKeyExA" (ByVal hKey As Long, ByVal lpSubKey As String, ByVal
ulOptions As Long, ByVal samDesired As Long, ByRef phkResult As Long) As
Long
Private Declare Function RegQueryValueEx Lib "advapi32" Alias
"RegQueryValueExA" (ByVal hKey As Long, ByVal lpValueName As String,
ByVal lpReserved As Long, ByRef lpType As Long, ByVal lpData As String,
ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long)
As Long

Private Sub cmdSysInfo_Click()
    Call StartSysInfo
End Sub

Private Sub cmdOK_Click()
    Unload Me
End Sub

Private Sub Form_Load()
    Me.Caption = "Acerca de " & App.Title
    lblVersion.Caption = "Versión " & App.Major & "." & App.Minor & "." &
App.Revision
    lblTitle.Caption = App.Title
    lblDescription.Caption = App.FileDescription
End Sub
```

```

Public Sub StartSysInfo()
    On Error GoTo SysInfoErr
    Dim rc As Long
    Dim SysInfoPath As String
    If GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFO, gREGVALSYSINFO,
SysInfoPath) Then
        ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC,
gREGVALSYSINFOLOC, SysInfoPath) Then archivo
            If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
                SysInfoPath = SysInfoPath & "\MSINFO32.EXE"

            Else
                GoTo SysInfoErr
            End If
        Else
            GoTo SysInfoErr
        End If

        Call Shell(SysInfoPath, vbNormalFocus)

    Exit Sub
SysInfoErr:
    MsgBox "La información del sistema no está disponible en este
momento", vbOKOnly
End Sub

papel
Dim Yi As Integer, LineaBase(1 To 9) As Single
Public Pag As Integer, Grabado As Boolean, ActualPage As Integer,
ValorInicial As Single
Private Const DistanciaLineaLinea As Integer = 10
Private Const DistanciaPtg As Integer = 40
Private Const DistanciaEnX As Integer = 35
Dim Xmin As Single, Xmax As Single, NoClaves As Long
Dim xx As Single, yy As Integer, Duracioncompas As Integer
Dim Pentagrama() As Integer, Sostenidos() As Boolean, texto As String,
PentagramaDeseado() As Integer
Public DuracionMinima As Integer, DuraCandMinima As Integer, Estricto As
Boolean, Modelo As Boolean
Dim Largo As Single

Dim EsPar As Boolean, RutaArchivo As String, mVectorEntrada As Single
Dim Nin As Integer, ValMaxEnFrec As Single, PrimeraVez As Boolean
Dim Rho As Single, Minimo(0 To 20) As Single 0.999894
Dim Nnew As Integer, Nant As Integer, Ncand As Integer, Duracion As
Integer, DuraCand As Integer
Dim bu() As Single, td() As Single, DuracionSilencio As Integer
Dim HoraIni

Const a As Single = 10, Bb As Single = 10, c As Single = 0.01, d As
Single = 0.98, e As Single = 1E-32
Const Alpha As Single = 0.5, Theta As Single = 0.1, Lfrec As Integer = 45
Const MaxSal As Integer = 21

```

```

Private Sub Command2_Click()
    DibujarNuevaNota 5, Negra
    DibujarNuevaNota 6, Negra *
    DibujarNuevaNota 8, Blanca
    DibujarNuevaNota 10, Blanca
    DibujarNuevaNota 12, Negra
    DibujarNuevaNota 13, Blanca *
    DibujarNuevaNota 11, BlancaPuntillo
    DibujarNuevaNota 12, Negra *
    DibujarNuevaNota 13, Blanca
    DibujarNuevaNota 14, Blanca *
End Sub

Private Sub Command3_Click()
    DeleteSetting "SiReNoM"
    Dim a As Long, ruta As String
    ruta = App.Path & "\e.exe"
    a = FileLen(ruta)
    MsgBox a
End Sub

Private Sub FlatScrollBar1_Change()
    Dim Npaginas As Long, i As Long
    Dim l, t, h
    Npaginas = picHoja.UBound
    l = picHoja(Pag).Left
    t = picHoja(Pag).Top
    h = picHoja(Pag).Height
    For i = 1 To Npaginas
        picHoja(i).Move picHoja(i).Left, -FlatScrollBar1.Value * 25 +
        picHoja(i).Height * (i - 1) * 1.05
        picHoja(i).Move picHoja(i).Left, -FlatScrollBar1.Value * 25 +
        picHoja(i).Height * (i - 1) * 1.1
    Next i
    ActualPage = (FlatScrollBar1.Value + 11.176 - ValorInicial) /
    11.176 12.3
    ActualPage = (25 * FlatScrollBar1.Value + Me.ScaleHeight) / Largo
    frmMain.StatusBar1.Panels("NPaginas").Text = "Pag " & ActualPage &
    "/" & Pag
End Sub

Private Sub FlatScrollBar1_Scroll()
    Call FlatScrollBar1_Change
End Sub

Private Sub Form_Activate()
    FlatScrollBar1_Change
    If frmMain.Toolbar1.Buttons("tbStop").Enabled Then
        Me.SetFocus
    End If
    FlatScrollBar1.Move Me.ScaleWidth - FlatScrollBar1.Width, 0,
    FlatScrollBar1.Width, Me.ScaleHeight
End Sub

```



```

Private Sub Form_Deactivate()
MsgBox "deactivate"
    If frmMain.Toolbar1.Buttons("tbStop").Enabled Then
        Me.SetFocus
    End If
frmMain.VerificarForms
End Sub

Private Sub Form_Load()
Dim i As Integer, X As Single, Xpent As Single
Dim fid As Integer, ruta As String
Me.WindowState = vbMaximized
Pag = 1
ActualPage = 1
Me.Move x, 0, 12240, 15840 2 * frmMain.ScaleHeight
Me.Move 0, 0, frmMain.ScaleWidth, frmMain.ScaleHeight
Me.Show
Me.Refresh
DoEvents
FlatScrollBar1.Move Me.ScaleWidth - FlatScrollBar1.Width, 0,
FlatScrollBar1.Width, Me.ScaleHeight
picHoja.Move 0, 0, Me.ScaleWidth - FlatScrollBar1.Width, Me.ScaleHeight
Me.ScaleMode = vbInches
picHoja(Pag).Move 0, 0, 8.5, 11
Me.ScaleMode = vbTwips

picHoja(Pag).ScaleMode = vbInches
lblTitulo.Move (picHoja(Pag).ScaleWidth - lblTitulo.Width) / 2, 1
picHoja(Pag).ScaleMode = vbPixels
lblPagina(Pag).Caption = Pag
lblPagina(Pag).Move 0.9 * picHoja(Pag).ScaleWidth, 0.93 *
picHoja(Pag).ScaleHeight
frmMain.StatusBar1.Panels("NPaginas").Text = "Pag " & ActualPage & "/" &
Pag
FlatScrollBar1.Max = (15840 - Me.ScaleHeight) / 1000

Me.ScaleMode = vbMillimeters
Largo = picHoja(1).Height
FlatScrollBar1.Max = (279.4 - Me.ScaleHeight) / 25
FlatScrollBar1.Max = (Largo - Me.ScaleHeight) / 25
ValorInicial = FlatScrollBar1.Max
Me.Refresh
X = (frmDoc.ScaleWidth - picHoja(Pag).Width) / 2
picHoja(Pag).Move X, 0
MsgBox Me.Caption
Xpent = picHoja(Pag).ScaleWidth * 0.11
Yi = 0.15 * picHoja(Pag).ScaleHeight
For i = 1 To 9
    Load imgPentagrama(i)
    imgPentagrama(i).Move Xpent, Yi + 5
    imgPentagrama(i).Visible = True
    LineaBase(i) = Yi + DistanciaLineaLinea * 5
    Call Dibujar5lineas
Next i

```

```

NoClaves = 9
Me.Refresh
lblTitulo.Caption = Me.Caption
    yy = 1
    xx = Xmin
    Mago.Show True
Mago.Play "pleased"
Mago.Speak "Espero que sepas tocar"
Mago.Hide
Mago.Play "processing"
ReDim Pentagrama(2, 1)
ReDim PentagramaDeseado(2, 1)
ReDim Sostenidos(20)
EsPar = False

With MMControll
    .Notify = False
    .Wait = True
    .Shareable = False
    .DeviceType = "Waveaudio"
End With
    Nin = 110
    ReDim bu(Nin, MaxSal) As Single, td(MaxSal, Nin) As Single,
entrada(1024) As Single
    fid = FreeFile
    ruta = App.Path & "\Pr1_110in.bin"
    Open ruta For Binary As #fid
    Get fid, 1, td
    Get fid, , bu
    Close fid
    Rho = 0.8621
lblNivelRuido.Caption = sldOffset.Value / 100
Estricto = True
Modelo = True
PrimeraVez = True
DuracionMinima = 1
DuraCandMinima = 1
Minimo(1) = 10 7           do
Minimo(2) = 19.9 10.9
Minimo(3) = 21.7 19.7       re
Minimo(4) = 22.4 10.4
Minimo(5) = 9.9 6.9        mi
Minimo(6) = 4.9           fa
Minimo(7) = 9.6
Minimo(8) = 29.4 25.4      sol
Minimo(9) = 75
Minimo(10) = 250 248.4     la
Minimo(11) = 255.7 250.7
Minimo(12) = 100 99.9      si
Minimo(13) = 160 152.7     do2
Minimo(14) = 231.8
Minimo(15) = 230 211.3     re2
Minimo(16) = 49.2
Minimo(17) = 276.2        mi2

```

```

Minimo(18) = 129.5    fa2
Minimo(19) = 99.7
Minimo(20) = 278.1    sol2
Minimo(0) = 30000
End Sub

Public Sub Dibujar5lineas()
Dim i As Integer
Dim inicio As Integer, fin As Integer

inicio = picHoja(Pag).ScaleWidth * 0.1
fin = picHoja(Pag).ScaleWidth * 0.9
Xmin = inicio + DistanciaEnX * 3
Xmax = fin - DistanciaEnX
For i = 1 To 5
    Yi = Yi + DistanciaLineaLinea
    picHoja(Pag).Line (inicio, Yi)-(fin, Yi)
Next i
    Yi = Yi + DistanciaPtg
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
Dim X As Single, t As Single, t2 As Single
Dim Resp As Integer
If Me.Name = "frmDoc" Then Exit Sub
If Me.Tag = "no" Then Exit Sub
If Me.Tag <> "saved" Then
    Resp = MsgBox("Desea guardar los cambios en " & Me.Caption,
vbYesNoCancel + vbQuestion)
    If Resp = vbYes Then
        frmMain.Toolbar1.Buttons("tbSave").Value = tbrPressed
        If frmMain.Toolbar1.Buttons("tbStop").Enabled Then
            frmMain.mnuDetener_Click
        End If
        frmMain.mnuGuardar_Click
    ElseIf Resp = vbCancel Then
        Cancel = -1
        Exit Sub
    Else
        If frmMain.Toolbar1.Buttons("tbStop").Enabled Then
            frmMain.mnuDetener_Click
        End If
    End If
End If
If Rnd() < 0.1 And Grabado Then MostrarMago
End Sub

Private Sub MostrarMago()
Dim X As Integer, t As Single, t2 As Single
Mago.Show

X = Int(1000 * Rnd())

Select Case X

```

```

Case 0 To 99
    Mago.Play "congratulate_2"
    Mago.Speak "¡Al fin aprendiste!"
    Mago.Speak "Hello world"
Case 100 To 199
    Mago.Play "surprised"
    Mago.Speak "Guau!!! eras vos"
    Mago.Speak "Pensé que era alguien que si podia..."
Case 200 To 299
    Mago.Play "GetAttention"
    Mago.Speak "Hey!"
    Mago.Play "GetAttention"
    Mago.Play "sad"
    Mago.Speak "Deberias dedicarte a otra cosa..."
Case 300 To 399
    Mago.Play "WriteContinued"
    Mago.Speak "Ya me fijé quien sos..."
    Mago.Speak "...no pienso dejate tocar de nuevo"
Case 400 To 499
    Mago.Play "Decline"
    Mago.Speak "¿por qué no pruebas con algo más acorde a tus
capacidades...?"
    Mago.Play "suggest"
    Mago.Speak "¡Una maraca!"
Case 500 To 599
    Mago.Play "Announce"
    Mago.Speak "Hay un burro flautista!!!"
    Mago.Play "pleased"
Case 600 To 699
    Mago.Play "Confused"
    Mago.Speak "Necesitas ayuda profesional..."
    Mago.Play "explain"
    Mago.Speak "de un Terapeuta profesional"
Case 700 To 799
    Mago.Play "Alert"
    Mago.Speak "Por lo menos pareces perseverante"
    Mago.Speak "Haré que aprendas a tocar..."
    Mago.Play "DoMagic1"
    Mago.Play "Think"
    Mago.Speak "Mmm..." + vbCrLf + "veo que eres un caso dificil"
    Mago.MoveTo 0, 0
    Mago.Play "Read"
    Mago.MoveTo 400, 400
    Mago.Play "GestureUp"
    Mago.Play "explain"
    Mago.Play "GestureUp"
    Mago.Play "DoMagic1"
    Mago.Play "DoMagic2"
    Mago.Play "Think"
    Mago.Play "Decline"
    Mago.Speak "Corrección: no eres un caso dificil"
    Mago.Speak "Eres un caso perdido"
Case 800 To 899
    Mago.Speak "Puedo ver tu futuro..."

```

```

Mago.Play "Searching"
t = Timer
Do
    t2 = Timer
    Loop While (t2 - t) < 10
Mago.Stop
Mago.Play "GestureUp"
Mago.Speak "Como flautista estás perdido"
Mago.Play "greet"
Case Else
    Mago.Play "Idle3_1"
    Mago.Speak "ya me aburrí de oírte..."
    Mago.Play "Idle3_1"
    Mago.Speak "por qué no salís a dar un paseo..."
    Mago.Speak "... y no te apresures en regresar"
End Select
Mago.Play "Wave"
Mago.Hide
End Sub

Private Sub Form_Terminate()
If (frmMain.ActiveForm Is Nothing) Then frmMain.HabilitarBotones ("Nada")
frmMain.VerificarForms
frmMain.StatusBar1.Panels("NPaginas").Text = ""
End Sub

Private Sub Form_Unload(Cancel As Integer)
If frmMain.ActiveForm Is Nothing Then frmMain.HabilitarBotones "Nada"
frmMain.VerificarForms
End Sub

Private Sub lblTitulo_DblClick()
Dim TituloAnterior As String
TituloAnterior = lblTitulo.Caption
lblTitulo.Caption = InputBox("Introduzca el titulo de la pieza musical",
"Titulo", lblTitulo.Caption)
If Trim(lblTitulo.Caption) = "" Then
    lblTitulo.Caption = TituloAnterior
Else
    Me.Caption = lblTitulo.Caption
End If
If Me.Tag <> "new" Then
    Me.Tag = "mod"
Else
    Me.Caption = lblTitulo.Caption
End If
frmMain.HabilitarBotones "Mod"
End Sub

Public Sub DibujarNuevaNota(Nota As Integer, ByVal Duracion As TipoNota,
Optional Posicion As Integer = 0, Optional Save As Boolean = True)
Dim Dura, Durb, Xpent As Single, separacionY As Single, separacionX As
Single
Dim sinInversion As Boolean, temporal As Single

```

```

Dim Posy As Single, t As Integer, Segunda As Single, i As Integer
On Error Resume Next
picHoja(Pag).ForeColor = RGB(0, 0, 0)
picHoja(Pag).FillColor = RGB(0, 0, 0)
If Nota = 0 Then
    Duracion = VerificarDuracionSilencio(Duracion)
    If Duracion = 0 Then Exit Sub
End If
Duracion = VerificarDuracion(Duracion)
If yy = 10 Then Exit Sub
Pag = Pag + 1
Load picHoja(Pag)
Segunda = picHoja(Pag).Height * (Pag - 1) * 1.05
picHoja(Pag).Move picHoja(Pag).Left, Segunda
picHoja(Pag).Visible = True
Load lblPagina(Pag)
Set lblPagina(Pag).Container = picHoja(Pag)
lblPagina(Pag).Caption = Pag
lblPagina(Pag).Move 0.9 * picHoja(Pag).ScaleWidth, 0.93 *
picHoja(Pag).ScaleHeight
lblPagina(Pag).Visible = True

Yi = 0.15 * picHoja(Pag).ScaleHeight
Xpent = picHoja(Pag).ScaleWidth * 0.11
For i = 1 To 9
    LineaBase(i) = Yi + DistanciaLineaLinea * 5
    NoClaves = NoClaves + 1
    Load imgPentagrama(NoClaves)
    Set imgPentagrama(NoClaves).Container = picHoja(Pag)
    imgPentagrama(NoClaves).Move Xpent, Yi + 5
    imgPentagrama(NoClaves).Visible = True
    Call Dibujar5lineas
Next i
picHoja(Pag).Refresh
picHoja(Pag).Move 0, Segunda
yy = 1
If Not Estricto Then
    FlatScrollBar1.Max = 1.1 * (Pag - 1) / 25 + ValorInicial 17.4
15.84 * 2
    temporal = (picHoja(Pag).Height * (1.05 * Pag - 0.05) -
Me.ScaleHeight) / 25
    FlatScrollBar1.Max = Round(temporal)
End If
FlatScrollBar1.Value = FlatScrollBar1.Value + 0.6 * Largo / 25 12.3
/ 2
ActualPage = (FlatScrollBar1.Value + ValorInicial) Mod (Largo *
1.05)
ActualPage = (25 * FlatScrollBar1.Value + Me.ScaleHeight) / Largo
frmMain.StatusBar1.Panels("NPaginas").Text = "Pag " & ActualPage &
"/" & Pag
End If

If Not Estricto Then
    If Modelo Then

```

```

        picHoja(Pag).ForeColor = RGB(128, 128, 128)
        picHoja(Pag).FillColor = RGB(128, 128, 128)
    End If
    If Modelo And Save Then
        t = UBound(PentagramaDeseado, 2)
        ReDim Preserve PentagramaDeseado(2, t + 1)
        PentagramaDeseado(1, t) = Nota
        PentagramaDeseado(2, t) = Duracion
        picHoja(Pag).ForeColor = RGB(128, 128, 128)
        picHoja(Pag).FillColor = RGB(128, 128, 128)
    ElseIf Save Then
        If Posicion = 0 Then
            t = UBound(Pentagrama, 2)
            ReDim Preserve Pentagrama(2, t + 1)
            Pentagrama(1, t) = Nota
            Pentagrama(2, t) = Duracion
        Else
            Pentagrama(1, Posicion) = Nota
            Pentagrama(2, Posicion) = Duracion
        End If
    End If
Else
    t = UBound(Pentagrama, 2)
    If PentagramaDeseado(1, t) = Nota And PentagramaDeseado(2, t) =
Duracion Then
        If Save Then
            If PentagramaDeseado(1, t) = Nota Then
                ReDim Preserve Pentagrama(2, t + 1)
                Pentagrama(1, t) = Nota
                Duracion = PentagramaDeseado(2, t)
                Pentagrama(2, t) = Duracion
            Else
                Exit Sub
            End If
        End If
    End If
End If
Select Case Nota
Case 0 silencio
    Posy = LineaBase(yy) - 2.5 * DistanciaLineaLinea
Case 1 do
    Posy = LineaBase(yy) + DistanciaLineaLinea
    picHoja(Pag).Line (xx - 8, Posy)-(xx + 8, Posy)
    If Sostenidos(Nota + 1) Then
        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 2 do#
    Posy = LineaBase(yy) + DistanciaLineaLinea
    picHoja(Pag).Line (xx - 8, Posy)-(xx + 8, Posy)
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 3 re
    Posy = LineaBase(yy) + DistanciaLineaLinea * 0.5
    If Sostenidos(Nota + 1) Then

```

```

        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 4    re#
    Posy = LineaBase(yy) + DistanciaLineaLinea * 0.5
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 5    mi
    Posy = LineaBase(yy)
Case 6    fa
    Posy = LineaBase(yy) - DistanciaLineaLinea * 0.5
    If Sostenidos(Nota + 1) Then
        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 7    fa#
    Posy = LineaBase(yy) - DistanciaLineaLinea * 0.5
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 8    sol
    Posy = LineaBase(yy) - DistanciaLineaLinea * 1
    If Sostenidos(Nota + 1) Then
        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 9    sol#
    Posy = LineaBase(yy) - DistanciaLineaLinea * 1
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 10   la
    Posy = LineaBase(yy) - DistanciaLineaLinea * 1.5
    If Sostenidos(Nota + 1) Then
        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 11   la#
    Posy = LineaBase(yy) - DistanciaLineaLinea * 1.5
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 12   si
    Posy = LineaBase(yy) - DistanciaLineaLinea * 2
Case 13   do2
    Posy = LineaBase(yy) - DistanciaLineaLinea * 2.5
    If Sostenidos(Nota + 1) Then
        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 14   do2#
    Posy = LineaBase(yy) - DistanciaLineaLinea * 2.5
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 15   re2
    Posy = LineaBase(yy) - DistanciaLineaLinea * 3
    If Sostenidos(Nota + 1) Then

```



```

        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 16 re2#
    Posy = LineaBase(yy) - DistanciaLineaLinea * 3
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 17     mi2
    Posy = LineaBase(yy) - DistanciaLineaLinea * 3.5
Case 18 fa2
    Posy = LineaBase(yy) - DistanciaLineaLinea * 4
    If Sostenidos(Nota + 1) Then
        DibujarBecuario xx, Posy
        Sostenidos(Nota + 1) = False
    End If
Case 19 fa2#
    Posy = LineaBase(yy) - DistanciaLineaLinea * 4
    If Not Sostenidos(Nota) Then DibujarSostenido xx, Posy
    Sostenidos(Nota) = True
Case 20     sol2
    Posy = LineaBase(yy) - DistanciaLineaLinea * 4.5
End Select

If Nota >= 12 Then
    sinInversion = False
    separacionY = 20
    separacionX = 4
Else
    sinInversion = True
    separacionY = 0
    separacionX = 0
End If
Duracioncompas = Duracioncompas + Duracion
If Duracioncompas > 16 Then
    Dura = 16 - (Duracioncompas - Duracion)
    Durb = Duracioncompas - 16
    If Nota = 0 Then
        Dura = VerificarDuracionSilencio(Dura)
        DibujarSilencio xx, Posy, Duracion, Pag
    Else
        DibujarNota xx, Posy, Dura, Pag, sinInversion
    End If
    xx = xx + DistanciaEnX
    picHoja(Pag).Line (xx, LineaBase(yy))-(xx, LineaBase(yy) -
DistanciaLineaLinea * 4) , RGB(255, 0, 0)
    ReDim Sostenidos(20)
    Duracioncompas = 0
    If Nota <> 0 Then
        If sinInversion Then
            picHoja(Pag).Circle (xx - separacionX, Posy + separacionY), 50,
, 1.25 * PI, 1.75 * PI, 0.3
        Else
            picHoja(Pag).Circle (xx - separacionX, Posy + separacionY), 50,
, 0.25 * PI, 0.75 * PI, 0.3

```

```

        End If
        Duracioncompas = Durb
        If Nota <> 0 Then
            Duracion = VerificarDuracion(Durb)
            xx = xx + DistanciaEnX
            DibujarNuevaNota Nota, Duracion, , False
        Else
            xx = xx + DistanciaEnX
        End If
    End If

Else
    If Nota = 0 Then
        DibujarSilencio xx, Posy, Duracion, Pag
    Else
        DibujarNota xx, Posy, Duracion, Pag, sinInversion
    End If
    xx = xx + DistanciaEnX
End If

If Duracioncompas = 16 Then
    picHoja(Pag).Line (xx, LineaBase(yy))-(xx, LineaBase(yy) -
DistanciaLineaLinea * 4)
    xx = xx + DistanciaEnX
    Duracioncompas = 0
    ReDim Sostenidos(20)
End If

If xx >= Xmax Then
    xx = Xmin
    yy = yy + 1
    If yy = 5 Then FlatScrollBar1.Value = FlatScrollBar1.Max Value + 6.15
End If

If UBound(Pentagrama, 2) = UBound(PentagramaDeseado, 2) And Estricto And
Save Then
    frmMain.mnuDetener_Click
    MsgBox "Felicidades" & vbCrLf & "Terminaste correctamente la ejecución
de la pieza", vbInformation, "Aprobaste"
End If
End Sub

Private Function VerificarDuracion(DuracionPrueba) As TipoNota
1, 2, 3., 4, 6., 8, 12., 16
Select Case DuracionPrueba
Case 5
    VerificarDuracion = Negra
Case 7
    VerificarDuracion = NegraPuntillo
Case 9, 10
    VerificarDuracion = Blanca
Case 11, 13, 14
    VerificarDuracion = BlancaPuntillo
Case 15, 17, 18

```

```

        VerificarDuracion = Redonda
Case Else
    VerificarDuracion = DuracionPrueba
End Select
End Function

Private Function VerificarDuracionSilencio(DuracionPrueba) As TipoNota
1, 2, 3., 4, 6., 8, 12., 16
Select Case DuracionPrueba
Case 2
    VerificarDuracionSilencio = Corchea
Case 3 To 5
    VerificarDuracionSilencio = Negra
Case 7
    VerificarDuracion = NegraPuntillo
Case 6, 7, 8, 9, 10, 11
    VerificarDuracionSilencio = Blanca
Case 11, 13, 14
    VerificarDuracion = BlancaPuntillo
Case 12 To 22 15, 17, 18
    VerificarDuracionSilencio = Redonda
Case Else
    VerificarDuracion = DuracionPrueba
    VerificarDuracionSilencio = 0
End Select
End Function

Public Sub ConstruirArchivo(NombreArchivo As String)
Dim TamañoTitulo As Integer, Titulo As String, TamañoPentagrama As Integer
Dim fid As Integer, InicioTitulo As Integer, InicioPentagrama As Integer
Dim Version As Byte, Clave As String, Compas As String, Armadura As Integer, Tempo As Integer
Dim TamañoCabecera As Integer
TamañoCabecera = 11
Version = &H1
Clave = "G"
Compas = "4/4"
Armadura = 0
Tempo = 60
TamañoTitulo = Len(Trim(lblTitulo.Caption))
TamañoPentagrama = UBound(Pentagrama, 2)
InicioPentagrama = TamañoCabecera + TamañoTitulo + 5
InicioTitulo = TamañoCabecera + 3
Titulo = Trim(lblTitulo.Caption)
fid = FreeFile
Open NombreArchivo For Binary As #fid
Put fid, 1, TamañoCabecera
Put fid, , Version
Put fid, , Clave
Put fid, , Compas
Put fid, , Armadura
Put fid, , Tempo
Put fid, , TamañoTitulo

```

```

Put fid, , Titulo
Put fid, , TamañoPentagrama
Put fid, InicioPentagrama, Pentagrama
Put fid, , Pentagrama
Close fid
Me.Caption = NombreArchivo
frmMain.ActualizarRecientes NombreArchivo
End Sub

Public Sub SubirTonos(NumeroSemitonos As Integer)
Dim i As Integer, t As Integer
Dim Nota As Integer, Dura As TipoNota
picHoja(1).Cls
If Pag > 1 Then
    For i = NoClaves To 10 Step -1
        Unload imgPentagrama(i)
    Next i
    For i = 2 To Pag
        picHoja(i).Cls
        Unload lblPagina(Pag)
        Unload picHoja(i)
    Next i
    NoClaves = 9
    Pag = 1
    FlatScrollBar1.Max = (279.4 - Me.ScaleHeight) / 25
    FlatScrollBar1.Value = 1
End If

Yi = 0.15 * picHoja(Pag).ScaleHeight
For i = 1 To 9
    LineaBase(i) = Yi + DistanciaLineaLinea * 5
    Call Dibujar5lineas
Next i
Me.Refresh
lblTitulo.Caption = Me.Caption
yy = 1
xx = Xmin
t = UBound(Pentagrama, 2) - 1
Duracioncompas = 0

For i = 1 To t
    Pentagrama(1, i) = Pentagrama(1, i) + NumeroSemitonos
    Nota = Pentagrama(1, i)
    Dura = Pentagrama(2, i)
    DibujarNuevaNota Nota, Dura, i
Next i
End Sub

Public Sub BuscarMinMax()
Dim t As Integer, i As Integer, X As Integer, Y As Integer
Dim VectorTemporal() As Single
t = UBound(Pentagrama, 2)
ReDim VectorTemporal(t - 1)
For i = 1 To t - 1

```

```

        VectorTemporal(i) = Pentagrama(1, i)
    Next i
    X = Pentagrama(1, PrimerMinimo(VectorTemporal))
    Y = Pentagrama(1, PrimerMaximo(VectorTemporal))
    frmSubirTonos.cmbTonos.Clear
    frmSubirTonos.Aabajo = 1 - X
    frmSubirTonos.Arriba = 20 - Y
End Sub

Public Sub LeerArchivo(NombreArchivo As String)
    Dim TamañoTitulo As Integer, Titulo As String
    Dim fid As Integer, InicioTitulo As Integer, InicioPentagrama As Integer
    Dim Version As Byte, Clave As Byte, Compas As String, Armadura As
    Integer, Tempo As Integer
    Dim TamañoCabecera As Integer, TamañoPentagrama As Integer
    TamañoCabecera = 11
    Version = &H1
    Clave = "G"
    Compas = "4/4"
    Tempo = 60
    TamañoTitulo = Len(Trim(lblTitulo.Caption))
    TamañoPentagrama = UBound(Pentagrama, 2)
    fid = FreeFile
    Open NombreArchivo For Binary As fid
    Get fid, 1, TamañoCabecera
    Get fid, , Version
    Get fid, , Clave
    Get fid, , Compas
    Get fid, , Armadura
    Get fid, , Tempo
    InicioTitulo = TamañoCabecera + 1
    Get fid, InicioTitulo, TamañoTitulo
    Titulo = Space(TamañoTitulo)
    Get fid, , Titulo
    Get fid, , TamañoPentagrama
    InicioPentagrama = TamañoCabecera + TamañoTitulo + 5
    ReDim Pentagrama(2, TamañoPentagrama)
    Get fid, InicioPentagrama, Pentagrama
    Close fid
    lblTitulo.Caption = Titulo
    Me.Caption = NombreArchivo
    SubirTonos 0
End Sub

Private Sub tmrTD_Timer()
    Static Contador As Integer
    frmMain.Toolbar1.Buttons("tbLight").Image = "imtLightOff"
    frmMain.picMetronomo.BackColor = &HF0F0F0

    If EsPar = True Then
        MMControll1.Command = "Save"
        MMControll1.Command = "Close"
        RutaArchivo = App.Path & "\Uno.wav"
        MMControll1.FileName = RutaArchivo
        MMControll1.Command = "Open"
    End If
End Sub

```

```

        MMControll.Command = "Record"
        aqui debe ir al procesado de las notas de par.wav
        IdentificarNota_T False
        LeerWav 8 15 29 en 32 partes
        ProcesarArchivo
        Shapel.BackColor = QBColor(4)
        EsPar = False
Else
    MMControll.Command = "Save"
    MMControll.Command = "Close"
    RutaArchivo = App.Path & "\Dos.wav"
    MMControll.FileName = RutaArchivo
    MMControll.Command = "Open"
    MMControll.Command = "Record"
    aqui debe ir al procesado de las notas de impar.wav
    IdentificarNota_T True
    LeerWav 8 15 29 en 32 partes
    ProcesarArchivo
    Shapel.BackColor = QBColor(14)
    EsPar = True
End If
Contador = Contador + 1
If Contador = 100 Then
    cmdDetener_Click
    MsgBox CStr(Timer - HoraIni)
    Contador = 0
End If
End Sub

Public Function LeerWav(Nveces As Integer)
    Dim fid As Integer, t As Integer, inicio As Integer
    Dim tempbuff() As Byte, temp() As Single, TempDos() As Single
    Dim Tempbuff2B() As Integer, BanderaCortar As Boolean
    Dim i As Integer, PosMax As Integer, ValMax As Single, Indos() As Single
    Dim k As Long, Iteracion As Integer, Resultados() As Single
    Dim NinFft As Integer, Tipo As Integer, Mensaje As String
    Dim FactorFrec As Integer, Fsampling As Integer, Channels As Integer
    Dim entrada() As Single, Nbits As Integer
    fid = FreeFile
    If EsPar = True Then
        RutaArchivo = App.Path & "\Dos.wav"
        If Nveces = 1 Then RutaArchivo = App.Path & "\MagDos.wav"
    Else
        RutaArchivo = App.Path & "\Uno.wav"
        If Nveces = 1 Then RutaArchivo = App.Path & "\MagUno.wav"
    End If
    If Nveces = 1 Then
        If EsPar = True Then
            RutaArchivo = App.Path & "\MagDos.wav"
        Else
            RutaArchivo = App.Path & "\MagUno.wav"
        End If
    End If
End Function

```

```

Open RutaArchivo For Binary As #fid
Get fid, 23, Channels
Get fid, 25, Fsampling
Get fid, 35, Nbits
Tipo = Nbits + Channels
FactorFrec = Fsampling / 11025
Debug.Print FactorFrec

Seek fid, 45
k = 45
Do While Not EOF(fid)
Close fid
ReDim Resultados(0 To 21)
For Iteracion = 1 To Nveces
    Select Case Tipo
    Case 9 8 bits mono
        ReDim tempbuff(1378 * FactorFrec)
        Get fid, k, tempbuff
        t = UBound(tempbuff)
        ReDim temp(t)

        For i = 1 To t
            temp(i) = (tempbuff(i) - 128) / 128
            texto = texto & CStr(temp(i)) & vbCrLf
        Next i
    Case 17 16 bits mono
        ReDim Tempbuff2B(1378 * FactorFrec)
        Get fid, k, Tempbuff2B
        t = UBound(Tempbuff2B)
        ReDim temp(t)
        For i = 1 To t
            Tempbuff2B(i) = Tempbuff2B(i) / 257
            temp(i) = (Tempbuff2B(i) - 128) / 128
        Next i
    Case 10 8 bits stereo
        ReDim tempbuff(2, 1378 * FactorFrec)
        Get fid, k, tempbuff
        t = UBound(tempbuff, 2)
        ReDim temp(t)
        For i = 1 To t
            temp(i) = (tempbuff(1, i) - 128) / 128
        Next i
    Case 18 16 bits stereo
        ReDim Tempbuff2B(2, 1378 * FactorFrec)
        Get fid, k, Tempbuff2B
        t = UBound(Tempbuff2B, 2)
        ReDim temp(t)
        For i = 1 To t
            Tempbuff2B(1, i) = Tempbuff2B(1, i) / 257
            temp(i) = (Tempbuff2B(1, i) - 128) / 128
        Next i
    Case Else
        Mensaje = "No de canales: " & CStr(Channels) & vbCrLf & "Frec de
muestreo: " & CStr(Fsampling) & vbCrLf & "Bits/muestra: " & CStr(Nbits)

```

```

    MsgBox Mensaje, vbCritical, "No es un formato compatible"
End Select

k = k + t / 2
Inicio = Zero2PosIndex(temp)
If Inicio > (353 * FactorFrec) Then Inicio = 353 * FactorFrec
inicio = 1
NinFft = 1024 * FactorFrec
If NinFft < 1024 Then
    ReDim Preserve temp(1024)
    NinFft = 1024
End If
ReDim Indos(NinFft)
ReDim Entrada(1024)
ReDim entrada(NinFft)
For i = 1 To NinFft
    entrada(i) = 1 + temp(inicio + i - 1)
    Indos(i) = temp(inicio + i - 1)
Next i
mVectorEntrada = Magnitud(Indos)
lblMag.Caption = mVectorEntrada
temp = fft(entrada)
temp = fft(Indos)
Nin = 110
ReDim TempDos(Nin)
For i = 1 To Nin
    TempDos(i) = temp(i + Lfrec)
Next i
PosMax = PrimerMaximo(TempDos)
ValMax = TempDos(PosMax)
ValMaxEnFrec = temp(PosMax + Lfrec)
If ValMax = 0 Then ValMax = 1
ReDim entrada(Nin)
lleva la entrada a valores entre 1 y 0
For i = 1 To Nin
    entrada(i) = TempDos(i) / ValMax
Next i
LeerWav = Entrada
Nnew = CalculosART2(entrada)
If ValMaxEnFrec < (Minimo(Nnew) * sldOffset.Value / 100) Then
If ValMaxEnFrec < (Minimo(Nnew) * frmMain.Offset / 100) Then
    If Nnew = Nant Then
        BanderaCortar = True
    Else
        Nnew = 0
        BanderaCortar = False
    End If
End If

If PrimeraVez Then
    PrimeraVez = False
    Duracion = 1
    Nant = Nnew
    Ncand = Nnew

```



```

        DuraCand = 0
    Else
        If Nant = Nnew Then
            Duracion = Duracion + 1
            If BanderaCortar And Duracion > 1 And Nnew <> 0 Then *
                Text1.Text = Text1.Text + DesplegarNota(Nant) + " =" +
CStr(Duracion) + vbTab
                DibujarNuevaNota Nant, Duracion
                Duracion = 0
            End If
        Else
            If Ncand = Nnew Then
                DuraCand = DuraCand + 1
            Else
                Ncand = Nnew
                DuraCand = 1
            End If
            If (DuraCand > 1 And Ncand <> 0) Or (DuraCand > 4) Then
                If (DuraCand > 1 And Ncand <> 0) Or (DuraCand > 4) Or
(BanderaCortar) Then
                    If (DuraCand > DuraCandMinima And Ncand <> 0) Or (DuraCand > 4)
Or (BanderaCortar) Then
                        If Duracion > 1 Then 2 Then
                            If Duracion > DuracionMinima Then
                                Text1.Text = Text1.Text + DesplegarNota(Nant) + " =" +
CStr(Duracion) + vbTab
                                DibujarNuevaNota Nant, Round(Duracion / 2)
                                Colum = Colum + 1
                                If Colum = 5 Then
                                    Colum = 0
                                    RichTextBox1.Text = RichTextBox1.Text + vbCrLf
                                End If
                            End If
                            Nant = Ncand
                            Duracion = DuraCand
                        End If
                    End If
                End If
            End If
        Next Iteracion
    Close fid
End Function

Private Function Zero2PosIndex(Matriz() As Single) As Integer
Dim M As Integer, i As Integer, p As Integer
M = UBound(Matriz, 1)
p = 1
For i = 2 To M
    If Sgn(Matriz(i - 1)) = -1 And Sgn(Matriz(i)) >= 0 Then
        p = i
        Exit For
    End If
Next i
Zero2PosIndex = p
End Function

```

```

Private Function PrimerMaximo(Vector() As Single) As Integer
Dim i As Integer, t As Integer, MaxPos As Integer
t = UBound(Vector)
MaxPos = 1
For i = 2 To t
    If Vector(i) > Vector(MaxPos) Then
        MaxPos = i
    End If
Next i
PrimerMaximo = MaxPos
End Function

Private Function Magnitud(Vector() As Single) As Single
Dim temp As Single, t As Integer, i As Integer, t2 As Single
t = UBound(Vector)
For i = 1 To t
    temp = temp + Vector(i) ^ 2
Next i
Magnitud = Sqr(temp)
End Function

Private Function CalculosART2(entrada() As Single) As Integer
Dim W() As Single, u() As Single, q() As Single, p() As Single
Dim X() As Single, V() As Single, Y(MaxSal) As Single, r() As Single
Dim Mw As Single, Mv As Single, Mp As Single, Mu As Single
Dim i As Integer, j As Integer, f As Integer, Jmax As Integer
Dim ResetFlag As Boolean, seg As Integer
Jmax = 0
ReDim W(Nin), V(Nin), r(Nin)
ReDim u(Nin), q(Nin), p(Nin), X(Nin)
For i = 1 To Nin
    W(i) = entrada(i)
Next i
Mw = Magnitud(W)
For i = 1 To Nin
    X(i) = W(i) / (Mw + e)
    If X(i) >= Theta Then
        V(i) = X(i)
    Else
        V(i) = 0
    End If
Next i
Mv = Magnitud(V)
For i = 1 To Nin
    u(i) = V(i) / (Mv + e)
    W(i) = entrada(i) + a * u(i)
    p(i) = u(i)
Next i
Mp = Magnitud(p)
For i = 1 To Nin
    X(i) = W(i) / (Mw + e)
    q(i) = p(i) / (Mp + e)
    V(i) = 0

```

```

        If X(i) >= Theta Then V(i) = X(i)
        If q(i) >= Theta Then V(i) = V(i) + Bb * q(i)
Next i
For j = 1 To MaxSal - 1
    Y(j) = 0
    For i = 1 To Nin
        Y(j) = Y(j) + bu(i, j) * p(i)
    Next i
Next j
seg = 0
Do
    Jmax = PrimerMaximo(Y)
    seg = seg + 1
    Mv = Magnitud(V)
    For i = 1 To Nin
        u(i) = V(i) / (Mv + e)
        p(i) = u(i) + d * td(Jmax, i)
    Next i
    Mu = Magnitud(u)
    Mp = Magnitud(p)
    For i = 1 To Nin
        r(i) = (u(i) + c * p(i)) / (Mu + c * Mp + e)
    Next i
    If Magnitud(r) < Rho Then
        Y(Jmax) = -1
        ResetFlag = True
    Else
        For i = 1 To Nin
            W(i) = entrada(i) + a * u(i)
        Next i
        Mw = Magnitud(W)
        For i = 1 To Nin
            X(i) = W(i) / (Mw + e)
            q(i) = p(i) / (Mp + e)
            V(i) = 0
            If X(i) >= Theta Then V(i) = X(i)
            If q(i) >= Theta Then V(i) = V(i) + Bb * q(i)
        Next i
        ResetFlag = False
    End If
Loop While ResetFlag = True And seg < 21
If ResetFlag = False And Jmax < 21 Then
    se identifico
    CalculosART2 = Jmax
    lblNota.Caption = DesplegarNota(Jmax)
    lblNota.Refresh
Else
    no se identifico
    lblNota.Caption = ""
    lblNota.Refresh
    CalculosART2 = 0
End If
End Function

```

```

Public Sub Iniciar()
EsPar = False
cmdIniciar.Enabled = False
cmdDetener.Enabled = True
If Not Estricto Then
    DuracionMinima = 1
    DuraCandMinima = 1
End If
HoraIni = Timer
    RutaArchivo = App.Path & "\Uno.wav"
    MMControll.FileName = RutaArchivo
    MMControll.Command = "Open"
    MMControll.Command = "Record"
    EsPar = False
    tmrTD.Enabled = True
End Sub

Public Sub Detener()
Dim fid As Integer
tmrTD.Enabled = False
MMControll.Command = "Save"
MMControll.Command = "Close"
If Rnd() < 0.2 Then MostrarMago
Grabado = True
fid = FreeFile
Open "d:\texto.txt" For Binary As fid
Put fid, 1, texto
Close fid
End Sub

Public Sub EditarPentagrama()
Dim t As Long, i As Long, tiempo As Integer
t = UBound(Pentagrama, 2)
frmEdicion.Grid.Rows = t
Load frmEdicion
For i = 1 To t - 1
    frmEdicion.cmbNota.ListIndex = Pentagrama(1, i)
    tiempo = Pentagrama(2, i)
    Select Case tiempo Grid.Text
    Case 1
        frmEdicion.cmbDuracion.ListIndex = 0
    Case 2
        frmEdicion.cmbDuracion.ListIndex = 1
    Case 3
        frmEdicion.cmbDuracion.ListIndex = 2
    Case 4
        frmEdicion.cmbDuracion.ListIndex = 3
    Case 5, 6
        frmEdicion.cmbDuracion.ListIndex = 4
    Case 7, 8, 9
        frmEdicion.cmbDuracion.ListIndex = 5
    Case 10, 11, 12, 13
        frmEdicion.cmbDuracion.ListIndex = 6
    Case 14, 15, 16, 17, 18, 19

```

```

        frmEdicion.cmbDuracion.ListIndex = 7
    Case Else
        cmbDuracion.Text = ""
        Sinduracion = True
    End Select
    frmEdicion.Grid.TextMatrix(i, 0) = i
    frmEdicion.Grid.TextMatrix(i, 1) = frmEdicion.cmbNota.Text
    frmEdicion.Grid.TextMatrix(i, 2) = frmEdicion.cmbDuracion.Text
Next i
frmEdicion.Editando = True
frmEdicion.Show vbModal
End Sub

Public Sub ActualizarPentagrama()
Dim t As Long, i As Long, tiempo As Integer, pos As Integer
t = frmEdicion.Grid.Rows
ReDim Pentagrama(2, t)
For i = 1 To t - 1
    pos = InStr(frmEdicion.Grid.TextMatrix(i, 1), "-")
    If cmbNota.ListIndex = 0 Then VerificarSilencio
        Pentagrama(1, i) = Val(Left(frmEdicion.Grid.TextMatrix(i, 1), pos -
1)) Left(cmbNota.Text, pos - 1)
        pos = InStr(frmEdicion.Grid.TextMatrix(i, 2), "-")
        Pentagrama(2, i) = Val(Left(frmEdicion.Grid.TextMatrix(i, 2), pos -
1))
    End If
Next i
SubirTonos 0
If Me.Tag <> "new" Then Me.Tag = "mod"
frmMain.HabilitarBotones "Mod"
End Sub

Public Sub lblTituloCambiar()
lblTitulo_DblClick
End Sub

Public Function LeerWavEstricto(Nveces As Integer)
Dim fid As Integer, t As Integer, inicio As Integer
Dim tempbuff() As Byte, temp() As Single, TempDos() As Single
Dim Tempbuff2B() As Integer, BanderaCortar As Boolean
Dim i As Integer, PosMax As Integer, ValMax As Single, Indos() As Single
Dim k As Long, Iteracion As Integer, Resultados() As Single
Dim NinFft As Integer, Tipo As Integer, Mensaje As String
Dim FactorFrec As Integer, Fsampling As Integer, Channels As Integer,
Nbits As Integer
Dim entrada() As Single
fid = FreeFile
If EsPar = True Then
    RutaArchivo = App.Path & "\Dos.wav"
    If Nveces = 1 Then RutaArchivo = App.Path & "\MagDos.wav"
Else
    RutaArchivo = App.Path & "\Uno.wav"
    If Nveces = 1 Then RutaArchivo = App.Path & "\MagUno.wav"
End If

```

```

If Nveces = 1 Then
    If EsPar = True Then
        RutaArchivo = App.Path & "\MagDos.wav"
    Else
        RutaArchivo = App.Path & "\MagUno.wav"
    End If
End If

Open RutaArchivo For Binary As #fid
Get fid, 23, Channels
Get fid, 25, Fsampling
Get fid, 35, Nbits
Tipo = Nbits + Channels
FactorFrec = Fsampling / 11025
Debug.Print FactorFrec

Seek fid, 45
k = 45
Do While Not EOF(fid)
    Close fid
    ReDim Resultados(0 To 21)
    For Iteracion = 1 To Nveces
        Select Case Tipo
            Case 9 8 bits mono
                ReDim tempbuff(1378 * FactorFrec)
                Get fid, k, tempbuff
                t = UBound(tempbuff)
                ReDim temp(t)

                For i = 1 To t
                    temp(i) = (tempbuff(i) - 128) / 128
                    texto = texto & CStr(temp(i)) & vbCrLf
                Next i
            Case 17 16 bits mono
                ReDim Tempbuff2B(1378 * FactorFrec)
                Get fid, k, Tempbuff2B
                t = UBound(Tempbuff2B)
                ReDim temp(t)
                For i = 1 To t
                    Tempbuff2B(i) = Tempbuff2B(i) / 257
                    temp(i) = (Tempbuff2B(i) - 128) / 128
                Next i
            Case 10 8 bits stereo
                ReDim tempbuff(2, 1378 * FactorFrec)
                Get fid, k, tempbuff
                t = UBound(tempbuff, 2)
                ReDim temp(t)
                For i = 1 To t
                    temp(i) = (tempbuff(1, i) - 128) / 128
                Next i
            Case 18 16 bits stereo
                ReDim Tempbuff2B(2, 1378 * FactorFrec)
                Get fid, k, Tempbuff2B
                t = UBound(Tempbuff2B, 2)

```

```

    ReDim temp(t)
    For i = 1 To t
        Tempbuff2B(1, i) = Tempbuff2B(1, i) / 257
        temp(i) = (Tempbuff2B(1, i) - 128) / 128
    Next i
    Case Else
        Mensaje = "No de canales: " & CStr(Channels) & vbCrLf & "Frec de
muestreo: " & CStr(Fsampling) & vbCrLf & "Bits/muestra: " & CStr(Nbits)
        MsgBox Mensaje, vbCritical, "No es un formato compatible"
    End Select

    k = k + t / 2
    Inicio = Zero2PosIndex(temp)
    If Inicio > (353 * FactorFrec) Then Inicio = 353 * FactorFrec
    inicio = 1
    NinFft = 1024 * FactorFrec
    ReDim Indos(NinFft)
    ReDim Entrada(1024)
    ReDim entrada(NinFft)
    For i = 1 To NinFft
        entrada(i) = 1 + temp(inicio + i - 1)
        Indos(i) = temp(inicio + i - 1)
    Next i
    mVectorEntrada = Magnitud(Indos)
    lblMag.Caption = mVectorEntrada
    temp = fft(entrada)
    temp = fft(Indos)
    Nin = 110
    ReDim TempDos(Nin)
    For i = 1 To Nin
        TempDos(i) = temp(i + Lfrec)
    Next i
    PosMax = PrimerMaximo(TempDos)
    ValMax = TempDos(PosMax)
    ValMaxEnFrec = temp(PosMax + Lfrec)
    If ValMax = 0 Then ValMax = 1
    ReDim entrada(Nin)
    lleva la entrada a valores entre 1 y 0

    For i = 1 To Nin
        entrada(i) = TempDos(i) / ValMax
    Next i
    LeerWav = Entrada
    Nnew = CalculosART2(entrada)
    If ValMaxEnFrec < (Minimo(Nnew) * sldOffset.Value / 100) Then
    If ValMaxEnFrec < (Minimo(Nnew) * frmMain.Offset / 100) Then
        If Nnew = Nant Then
            BanderaCortar = True
        Else
            Nnew = 0
            BanderaCortar = False
        End If
    End If

```

```

    If PrimeraVez Then
        PrimeraVez = False
        Duracion = 1
        Nant = Nnew
        Ncand = Nnew
        DuraCand = 0
    Else
        If Nant = Nnew Then
            Duracion = Duracion + 1
            If BanderaCortar And Duracion > 1 And Nnew <> 0 Then *
                Text1.Text = Text1.Text + DesplegarNota(Nant) + " =" +
CStr(Duracion) + vbTab
                DibujarNuevaNota Nant, Duracion
                Duracion = 0
            End If
        Else
            If Ncand = Nnew Then
                DuraCand = DuraCand + 1
            Else
                Ncand = Nnew
                DuraCand = 1
            End If
            If (DuraCand > 1 And Ncand <> 0) Or (DuraCand > 4) Then
                If (DuraCand > 1 And Ncand <> 0) Or (DuraCand > 4) Or
(BanderaCortar) Then
                    If Duracion > 1 Then 2 Then
                        Text1.Text = Text1.Text + DesplegarNota(Nant) + " =" +
CStr(Duracion) + vbTab
                        DibujarNuevaNota Nant, Round(Duracion / 2)
                        Colum = Colum + 1
                        If Colum = 5 Then
                            Colum = 0
                            RichTextBox1.Text = RichTextBox1.Text + vbCrLf
                        End If
                    End If
                    Nant = Ncand
                    Duracion = DuraCand
                End If
            End If
        End If
    End If
Next Iteracion
Close fid
End Function

Public Sub ConfigurarModoAprendizaje()
    Modelo = False
    Estricto = True
    Me.Tag = "new"
    FlatScrollBar1.Value = 0
    yy = 1
    xx = Xmin
    Pag = 1
    DuracionMinima = 1
    DuraCandMinima = 2

```



```

Duracioncompas = 0
frmMain.mnuCaptura_Click
ReDim Pentagrama(2, 1)
End Sub

Public Sub MoverScroll()
FlatScrollBar1.Move Me.ScaleWidth - FlatScrollBar1.Width, 0,
FlatScrollBar1.Width, Me.ScaleHeight
End Sub

edicion
Dim Sinduracion As Boolean, SinNota As Boolean
Public Editando As Boolean

Private Sub cmbDuracion_Click()
Dim Numero As String, pos As Integer
If Not Editando Then Exit Sub

pos = InStr(cmbDuracion.Text, "-")
If cmbNota.ListIndex = 0 Then VerificarSilencio
Grid.Col = 2
Grid.Text = cmbDuracion.Text Left(cmbDuracion.Text, pos - 1)
Sinduracion = False
If SinNota Then cmbNota_Click
End Sub

Private Sub cmbNota_Click()
Dim Numero As String, pos As Integer
If Not Editando Then Exit Sub

pos = InStr(cmbNota.Text, "-")
If cmbNota.ListIndex = 0 Then VerificarSilencio
Grid.Col = 1
Grid.Text = cmbNota.Text Left(cmbNota.Text, pos - 1)
SinNota = False
If Sinduracion Then cmbDuracion_Click
End Sub

Private Sub cmdAceptar_Click()
If FilasVacias > 0 Then

    MsgBox "Debe llenar los espacios en blanco"
    Grid.Row = FilasVacias
    Exit Sub
End If
frmMain.ActiveForm.ActualizarPentagrama
Unload Me
Grid.Rows = 1
Me.Hide
Editando = False
Unload Me
End Sub

```

```

Private Sub cmdCancelar_Click()
Grid.Rows = 1
Editando = False
Unload Me
End Sub

Private Sub cmdContraer_Click()
Grid.Move 50, Grid.Top
cmdExpander.Visible = True
cmdInsertar.Move 4100, cmdInsertar.Top, 500
cmdEliminar.Move 4100, cmdEliminar.Top, 500
cmdAceptar.Move 4100, cmdAceptar.Top, 500
cmdCancelar.Move 4100, cmdCancelar.Top, 500
Me.Width = 4800
Me.Tag = "c"

cmdInsertar.Caption = ""
cmdEliminar.Caption = ""
cmdAceptar.Caption = ""
cmdCancelar.Caption = ""
Grid_Scroll
End Sub

Private Sub cmdEliminar_Click()
Dim i As Long

For i = Grid.Row To Grid.Rows - 2
    Grid.TextMatrix(i, 1) = Grid.TextMatrix(i + 1, 1)
    Grid.TextMatrix(i, 2) = Grid.TextMatrix(i + 1, 2)
Next i
Grid.Rows = Grid.Rows - 1
Grid_Click
End Sub

Private Sub cmdExpander_Click()
Grid.Move 240, Grid.Top
cmdExpander.Visible = False
cmdInsertar.Move 4920, cmdInsertar.Top, 1695
cmdEliminar.Move 4920, cmdEliminar.Top, 1695
cmdAceptar.Move 4920, cmdAceptar.Top, 1695
cmdCancelar.Move 4920, cmdCancelar.Top, 1695
Me.Tag = "e"
Me.Width = 7245
cmdInsertar.Caption = "&Insertar"
cmdEliminar.Caption = "&Eliminar"
cmdAceptar.Caption = "&Aceptar"
cmdCancelar.Caption = "&Cancelar"
Grid_Scroll
End Sub

Private Sub cmdInsertar_Click()
Dim i As Long
Grid.Rows = Grid.Rows + 1
For i = Grid.Rows - 1 To Grid.Row + 1 Step -1

```

```

        Grid.TextMatrix(i, 1) = Grid.TextMatrix(i - 1, 1)
        Grid.TextMatrix(i, 2) = Grid.TextMatrix(i - 1, 2)
    Next i
    Grid.TextMatrix(Grid.Rows - 1, 0) = Val(Grid.TextMatrix(Grid.Rows - 2,
    0)) + 1
    Grid.TextMatrix(Grid.Row, 1) = ""
    Grid.TextMatrix(Grid.Row, 2) = ""
    cmbNota.Visible = False
    cmbDuracion.Visible = False
End Sub

Private Sub Form_DblClick()
    cmdExpander_Click
    Me.Height = 7000
    MsgBox Grid.CellHeight
End Sub

Private Sub Form_Resize()
    If Me.Height > 1000 Then Grid.Height = Me.Height - 1000
    If Me.Tag = "c" Then
        If Me.Width > 4800 Then Me.Width = 4800
    Else
        If Me.Width > 7245 Then Me.Width = 7245
    End If
End Sub

Private Sub Grid_Click()
    Dim X As Single, Y As Single, Nota As String, pos As Integer
    Dim tiempo As String
    X = Grid.Left + Grid.ColPos(2) + 30
    Y = Grid.Top + Grid.CellTop
    cmbDuracion.Move X, Y
    cmbNota.Visible = True
    cmbDuracion.Visible = True
    Grid.Col = 2
    pos = InStr(Grid.Text, "-")

    If pos > 0 Then tiempo = Left(Grid.Text, pos - 1)
    Select Case tiempo Grid.Text
    Case "1"
        cmbDuracion.ListIndex = 0
    Case "2"
        cmbDuracion.ListIndex = 1
    Case "3"
        cmbDuracion.ListIndex = 2
    Case "4"
        cmbDuracion.ListIndex = 3
    Case "6"
        cmbDuracion.ListIndex = 4
    Case "8"
        cmbDuracion.ListIndex = 5
    Case "12"
        cmbDuracion.ListIndex = 6
    Case "16"

```

```

        cmbDuracion.ListIndex = 7
Case Else
    cmbDuracion.Text = ""
    Grid.TextMatrix(Grid.Row, 2) = cmbDuracion.Text
    Sinduracion = True
End Select
X = Grid.Left + Grid.ColPos(1) + 30
cmbNota.Move X, Y
Grid.Col = 1
pos = InStr(Grid.Text, "-")
If pos > 0 Then
    Nota = Left(Grid.Text, pos - 1)
    cmbNota.ListIndex = Val(Nota)
Else
    Grid.TextMatrix(Grid.Row, 1) = cmbNota.Text
    SinNota = True
End If
End Sub

Private Sub Form_Load()
Dim i As Integer

With Grid
    .FixedAlignment(1) = flexAlignCenterCenter
    .ColAlignment(1) = flexAlignCenterCenter
    .ColAlignment(2) = flexAlignCenterCenter
    .ColAlignment(0) = flexAlignCenterCenter
    .Col = 1
    .Row = 0
    .Text = "Nota"
    .Col = 2
    .Text = "Duracion"
    .Col = 0
    .Text = "Ubic"
    .ColWidth(2) = 1800
    .ColWidth(1) = 1000
    .ColWidth(0) = 800
End With

cmbDuracion.AddItem "1-Semicorchea" 0
cmbDuracion.AddItem "2-Corchea" 1
cmbDuracion.AddItem "3-CorcheaPuntillo" 2
cmbDuracion.AddItem "4-Negra" 3
cmbDuracion.AddItem "6-NegraPuntillo" 4
cmbDuracion.AddItem "8-Blanca" 5
cmbDuracion.AddItem "12-BlancaPuntillo" 6
cmbDuracion.AddItem "16-Redonda" 7

cmbNota.AddItem "0-Silencio"
cmbNota.AddItem "1-do"
cmbNota.AddItem "2-do#"
cmbNota.AddItem "3-re"
cmbNota.AddItem "4-re#"
cmbNota.AddItem "5-mi"

```

```

cmbNota.AddItem "6-fa"
cmbNota.AddItem "7-fa#"
cmbNota.AddItem "8-sol"
cmbNota.AddItem "9-sol#"
cmbNota.AddItem "10-la"
cmbNota.AddItem "11-la#"
cmbNota.AddItem "12-si"
cmbNota.AddItem "13-do2"
cmbNota.AddItem "14-do2#"
cmbNota.AddItem "15-re2"
cmbNota.AddItem "16-re2#"
cmbNota.AddItem "17-mi2"
cmbNota.AddItem "18-fa2"
cmbNota.AddItem "19-fa2#"
cmbNota.AddItem "20-sol2"

For i = 1 To Grid.Rows - 1
    Grid.TextMatrix(i, 0) = i
Next i
End Sub

Private Sub Grid_Scroll()
Grid.TextMatrix(Grid.Row, 1) = cmbNota.Text
cmbNota.Visible = False
cmbDuracion.Visible = False
End Sub

Private Sub VerificarSilencio()
Select Case cmbDuracion.ListIndex
    Case 0, 1
        cmbDuracion.ListIndex = 1
    Case 2, 3
        cmbDuracion.ListIndex = 3
    Case 4, 5
        cmbDuracion.ListIndex = 5
    Case 6, 7
        cmbDuracion.ListIndex = 7
    Case Else
        cmbDuracion.ListIndex = 3
End Select
End Sub

Private Function FilasVacias() As Long
Dim i As Long
FilasVacias = 0
For i = 1 To Grid.Rows - 1
    If Grid.TextMatrix(i, 1) = "" Or Grid.TextMatrix(i, 2) = "" Then
        FilasVacias = i
        Exit For
    End If
Next i
End Function

```

```

main
Public NumTonos As Integer
Dim WithEvents MyAgent As Agent
Dim ini
Public Offset As Single, lDocumentCount As Long
Private Declare Function OSWinHelp% Lib "user32" Alias "WinHelpA" (ByVal
hwnd&, ByVal HelpFile$, ByVal wCommand%, dwdata As Any)

Private Sub MDIForm_Load()
Dim llave As String, valor As String
llave = "Porcentaje"
valor = GetSetting(Aplicacion, ToleranciaAlRuido, llave)
If valor = Empty Then
Offset = 100
Else
Offset = Val(valor)
End If
llave = "ColorMetronomo"
valor = GetSetting(Aplicacion, ColorMetronomo, llave)
SaveSetting Aplicacion, ToleranciaAlRuido, llave,
CStr(CommonDialog1.Color)

If valor = Empty Then
CommonDialog1.Color = RGB(16, 255, 255) &H10FFFF
Else
CommonDialog1.Color = Val(valor)
End If

Create an instance of the control using New
Set MyAgent = New Agent
Open a connection to the server
MyAgent.Connected = True
MyAgent.Characters.Load "merlin",
"C:\WINDOWS\msagent\chars\merlin.acs"
Set Mago = MyAgent.Characters("merlin")
Mago.MoveTo 400, 400
MyAgent.Characters("merlin").Show True

Mago.Play "processing"
Randomize
CommonDialog1.InitDir = App.Path
CommonDialog1.HelpFile = App.HelpFile
HabilitarBotones "Nada"
LeerRecientes
End Sub

Private Sub MDIForm_QueryUnload(Cancel As Integer, UnloadMode As Integer)
Dim llave As String
Set Mago = Nothing
MyAgent.Characters.Unload "merlin"
Set MyAgent = Nothing
llave = "ColorMetronomo"
SaveSetting Aplicacion, ColorMetronomo, llave, CStr(CommonDialog1.Color)
End Sub

```

```

Private Sub mnuAbrir_Click()
Dim frmD As frmDoc
On Error GoTo a
CommonDialog1.DialogTitle = "Abrir"
CommonDialog1.ShowOpen
Set frmD = New frmDoc
frmD.Show
frmD.SetFocus
frmD.Tag = "saved"
ActiveForm.LeerArchivo CommonDialog1.FileName
HabilitarBotones ("Abrir")
frmDoc.Tag = "saved"
Unload frmDoc
ActualizarRecientes CommonDialog1.FileName
Exit Sub
a:
    If (ActiveForm Is Nothing) Then HabilitarBotones ("Nada")
End Sub

Private Sub mnuAcercaDe_Click()
frmAbout.Show vbModal, frmMain
End Sub
Private Sub mnuAprendizajeSupervisado_Click()
Dim frmD As frmDoc
On Error GoTo a
CommonDialog1.DialogTitle = "Seleccione el pentagrama modelo..."
CommonDialog1.ShowOpen
Set frmD = New frmDoc
frmD.Show
frmD.SetFocus
frmD.Tag = "saved"
frmD.Modelo = True
frmD.Estricto = False
ActiveForm.LeerArchivo CommonDialog1.FileName
ActiveForm.ConfigurarModoAprendizaje
HabilitarBotones ("Abrir")
lDocumentCount = lDocumentCount + 1
frmD.Caption = "Sin Titulo " & lDocumentCount
frmD.Tag = "new"
Unload frmDoc
mnuCaptura_Click
ActualizarRecientes CommonDialog1.FileName
Exit Sub
a:
    If Err.Number = cdlCancel Then
        MsgBox "Esta acción cancelará el modo de aprendizaje supervisado",
vbInformation
    End If
    If (ActiveForm Is Nothing) Then HabilitarBotones ("Nada")
End Sub

Private Sub mnuArrange_Click()
Me.Arrange vbArrangeIcons
End Sub

```

```

Public Sub mnuCaptura_Click()
HabilitarBotones ("Play")
tmrLight.Enabled = True
If ActiveForm.Tag <> "new" Then ActiveForm.Tag = "mod"
ini = Timer
frmMain.ActiveForm.Iniciar
End Sub

Private Sub mnuCascada_Click()
Me.Arrange vbCascade
End Sub

Private Sub mnuCerrar_Click()
If Not (ActiveForm Is Nothing) Then Unload ActiveForm
VerificarForms
End Sub

Private Sub mnuContenido_Click()
Dim nRet, Ar As String
Ar = App.Path & "\Ayuda SiReNoM.chm"
    If Len(App.HelpFile) = 0 Then
        MsgBox "No se puede mostrar el contenido de la Ayuda. No hay
Ayuda asociada a este proyecto.", vbInformation, Me.Caption

    Else
        On Error Resume Next
        nRet = OSWinHelp(Me.hwnd, App.HelpFile, 3, 0)
        hwnedayudahtml = Me.hWnd
        h = HtmlHelp(hwnedayudahtml, "c:\ayudalexo\ayuda.chm" +
"::\html\index.html", HH_DISPLAY_TOPIC, 0&)

        h = HtmlHelp(Null, Ar, HH_DISPLAY_TOPIC, 0&)
        h = HtmlHelp(hwnedayudahtml, Ar, HH_DISPLAY_TOPIC, 0&)

        h = HtmlHelp(Me.hwnd, App.HelpFile, HH_DISPLAY_TOPIC, 0&)

        h=htmlhelp(hwnedayudahtml,app.HelpFile)+::\html
        Shell Ar, vbNormalFocus
        If Err Then
            MsgBox Err.Description
        End If
    End If
End Sub

Public Sub mnuDetener_Click()
HabilitarBotones ("Stop")
tmrLight.Enabled = False
frmMain.ActiveForm.Detener
End Sub

Private Sub mnuEdicionManual_Click()
frmMain.ActiveForm.EditarPentagrama
End Sub

```



```

Public Sub mnuGuardar_Click()
If ActiveForm.Tag = "new" Then
    mnuGuardarComo_Click
Else
    grabar archivo binario
    ActiveForm.ConstruirArchivo ActiveForm.Caption CommonDialog1.FileName
    ActiveForm.Tag = "saved"
End If
HabilitarBotones ("Grabar")
End Sub

Private Sub mnuGuardarComo_Click()
On Error Resume Next
CommonDialog1.FileName = ActiveForm.lblTitulo.Caption
CommonDialog1.ShowSave
HabilitarBotones ("Grabar")
    ActiveForm.ConstruirArchivo CommonDialog1.FileName
    ActiveForm.Tag = "saved"
End Sub

Private Sub mnuImprimir_Click()
Dim Ncopias As Integer, i As Integer, j As Integer
Dim Sx As Single, Sy As Single
If Not (frmMain.ActiveForm Is Nothing) Then
    With CommonDialog1
        .Flags = .Flags Or cdlPDNoPageNums Or cdlPDNoSelection
        .ShowPrinter
    End With
    If Err = 0 Then
        For Ncopias = 1 To CommonDialog1.Copies
            With Printer
                .PaintPicture ActiveForm.picHoja(1).Image, 0, 0
                .ScaleMode = vbInches
                .CurrentX = (.ScaleWidth -
                .TextWidth(ActiveForm.lblTitulo.Caption)) / 2
                .CurrentY = 1
                .Font = ActiveForm.lblTitulo.Font
                .Font.Bold = True
            End With
            Printer.Print ActiveForm.lblTitulo.Caption
            Sx = 0.935
            For j = 1 To 9
                Sy = 0.07 * Printer.ScaleHeight + 0.94 * (j)
                Printer.PaintPicture
                ActiveForm.imgPentagrama(0).Picture, Sx, Sy
            Next j
            If frmMain.ActiveForm.Pag > 1 Then
                For i = 2 To frmMain.ActiveForm.Pag
                    Printer.NewPage
                    Printer.PaintPicture ActiveForm.picHoja(1).Image, 0, 0
                    Printer.Print Printer.Page

                    For j = 1 To 9
                        Sy = 0.07 * Printer.ScaleHeight + 0.94 * (j)

```

```

        Printer.PaintPicture
ActiveForm.imgPentagrama(0).Picture, Sx, Sy
        Next j
        Next i
    End If
    Printer.Page = 1
    Printer.EndDoc
    Next Ncopias
    Clipboard.SetData frmMain.ActiveForm.picHoja.Picture
End If
End If
e:
End Sub

Private Sub mnuNuevo_Click()
NuevoDoc
HabilitarBotones ("Nuevo")
End Sub

Private Sub NuevoDoc()
Static lDocumentCount As Long
Dim frmD As Form frmDoc
Dim frmD As New frmDoc
lDocumentCount = lDocumentCount + 1
Set frmD = New frmDoc
frmD.Tag = 1
frmD.Visible = True
frmD.Caption = "Sin Titulo " & lDocumentCount
frmD.Tag = "new"
Load frmD show
frmDoc.Tag = "saved"
Unload frmDoc
frmMain.ActiveForm.picHoja.lblTitulo.Caption = ActiveForm.Caption
End Sub

Private Sub mnuOffset_Click()
frmOffset.SSTab1.Tab = 0
frmOffset.Show , Me
End Sub

Private Sub mnuOpciones_Click()
frmOffset.SSTab1.Tab = 1
frmOffset.Show
End Sub

Private Sub mnuReciente_Click(Index As Integer)
Dim frmD As frmDoc, Nombre As String, TenBytes As Long
If mnuReciente(Index).Caption = "r" Then Exit Sub

On Error GoTo a
CommonDialog1.ShowOpen
Nombre = mnuReciente(Index).Tag
TenBytes = FileLen(Nombre)

```

```

If TenBytes > 13 Then
    CommonDialog1.FileName = Nombre
    Set frmD = New frmDoc
    frmD.Show
    frmD.SetFocus
    frmD.Tag = "saved"
    ActiveForm.LeerArchivo Nombre CommonDialog1.FileName
    HabilitarBotones ("Abrir")
    frmDoc.Tag = "saved"
    Unload frmDoc
    ActualizarRecientes CommonDialog1.FileName
End If
Exit Sub
a:
    If (ActiveForm Is Nothing) Then HabilitarBotones ("Nada")
    If Err.Number = 53 Then
        MsgBox "No existe el archivo indicado", vbCritical
    End If
End Sub

Private Sub mnuSalir_Click()
    Unload Me
End Sub

Private Sub mnuSubirTono_Click()
    On Error GoTo aa
    frmSubirTonos.Abajo = -5
    frmSubirTonos.Arriba = 5
    frmSubirTonos.cmbTonos.Clear
    frmMain.ActiveForm.BuscarMinMax
    frmSubirTonos.Show vbModal
    If frmSubirTonos.cmbTonos.Text <> "0" Then
    If NumTonos <> 0 Then
        MsgBox NumTonos
        frmMain.ActiveForm.SubirTonos NumTonos
        If frmMain.ActiveForm.Tag <> "new" Then frmMain.ActiveForm.Tag = "mod"
    End If
    End If
Exit Sub
aa:
End Sub

Private Sub mnuTileHorizontal_Click()
    Me.Arrange vbTileHorizontal
End Sub

Private Sub mnuTileVertical_Click()
    Me.Arrange vbTileVertical
End Sub

Private Sub mnuTitulo_Click()
    If Not (ActiveForm Is Nothing) Then ActiveForm.lblTituloCambiar
End Sub
Private Sub mnuVerBarrasMetronomo_Click(Index As Integer)

```

```

mnuVerBarrasMetronomo(Index).Checked = Not
mnuVerBarrasMetronomo(Index).Checked
picMetronomo(Index).Visible = mnuVerBarrasMetronomo(Index).Checked
ActiveForm.MoverScroll
End Sub

Private Sub mnuVerStatusBar_Click()
mnuVerStatusBar.Checked = Not mnuVerStatusBar.Checked
StatusBar1.Visible = mnuVerStatusBar.Checked
End Sub

Private Sub mnuVerToolBar_Click()
mnuVerToolBar.Checked = Not mnuVerToolBar.Checked
ToolBar1.Visible = mnuVerToolBar.Checked
End Sub

Private Sub Option1_Click(Index As Integer)
Clase = 2 ^ (Index)
End Sub

Private Sub tmrLight_Timer()
Static a As Integer
Dim i As Integer , j, k
If Toolbar1.Buttons("tbLight").Image = "imtLightOn" Then
    Toolbar1.Buttons("tbLight").Image = "imtLightOff"
    For i = 0 To 3
        picMetronomo(i).BackColor = vbButtonFace &HF0F0F0
    Next i
Else
    Toolbar1.Buttons("tbLight").Image = "imtLightOn"
    For i = 0 To 3
        picMetronomo(i).BackColor = CommonDialog1.Color &H10FFFF
        &H8000000F
    Next i
End If
a = a + 1
For i = i To 10000
    For k = 1 To 10
        j = j + 0.001
    Next k
Next i

If a = 1000 Then
    tmrLight.Enabled = False
    MsgBox CStr(Timer - ini)
End If
End Sub

Public Sub HabilitarBotones(Modo As String)
Select Case Modo
    Case "Nuevo", "Stop", "Mod"
        Toolbar1.Buttons("tbSave").Enabled = True
        Toolbar1.Buttons("tbPrint").Enabled = True
        Toolbar1.Buttons("tbPlay").Enabled = True

```

```

Toolbar1.Buttons("tbStop").Enabled = False
Toolbar1.Buttons("tbLight").Visible = False
StatusBar1.Panels(1).Text = "Modificado"
mnuGuardar.Enabled = True
mnuGuardarComo.Enabled = True
mnuImprimir.Enabled = True
mnuCerrar.Enabled = True
mnuCaptura.Enabled = True
mnuDetener.Enabled = False
mnuEdicionManual.Enabled = True
mnuAprendizajeSupervisado.Enabled = True
mnuSubirTono.Enabled = True
mnuTitulo.Enabled = True
Case "Play"
    Toolbar1.Buttons("tbSave").Enabled = False
    Toolbar1.Buttons("tbPrint").Enabled = False
    Toolbar1.Buttons("tbPlay").Enabled = False
    Toolbar1.Buttons("tbStop").Enabled = True
    Toolbar1.Buttons("tbLight").Visible = True
    StatusBar1.Panels(1).Text = "Grabando"
    mnuGuardar.Enabled = False
    mnuGuardarComo.Enabled = False
    mnuImprimir.Enabled = False
    mnuCerrar.Enabled = False
    mnuCaptura.Enabled = False
    mnuDetener.Enabled = True
    mnuEdicionManual.Enabled = False
    mnuAprendizajeSupervisado.Enabled = False
    mnuSubirTono.Enabled = False
    mnuTitulo.Enabled = False
Case "Abrir", "Grabar"
    Toolbar1.Buttons("tbSave").Enabled = False
    Toolbar1.Buttons("tbPrint").Enabled = True
    Toolbar1.Buttons("tbPlay").Enabled = True
    Toolbar1.Buttons("tbStop").Enabled = False
    Toolbar1.Buttons("tbLight").Visible = False
    StatusBar1.Panels(1).Text = "Archivo"
        mnuGuardar.Enabled = False
    mnuGuardarComo.Enabled = True
    mnuImprimir.Enabled = True
    mnuCerrar.Enabled = True
    mnuCaptura.Enabled = True
    mnuDetener.Enabled = False
    mnuEdicionManual.Enabled = True
    mnuAprendizajeSupervisado.Enabled = True
    mnuSubirTono.Enabled = True
    mnuTitulo.Enabled = True
Case "Nada"
    Toolbar1.Buttons("tbSave").Enabled = False
    Toolbar1.Buttons("tbPrint").Enabled = False
    Toolbar1.Buttons("tbPlay").Enabled = False
    Toolbar1.Buttons("tbStop").Enabled = False
    Toolbar1.Buttons("tbLight").Visible = False
    StatusBar1.Panels(1).Text = "Nada"

```

```

        StatusBar1.Panels("NPaginas").Text = ""
        mnuGuardar.Enabled = False
        mnuGuardarComo.Enabled = False
        mnuImprimir.Enabled = False
        mnuCerrar.Enabled = False
        mnuCaptura.Enabled = False
        mnuDetener.Enabled = False
        mnuEdicionManual.Enabled = False
        mnuAprendizajeSupervisado.Enabled = False
        mnuSubirTono.Enabled = False
        mnuTitulo.Enabled = False

End Select
End Sub

Private Sub Toolbar1_ButtonClick(ByVal Button As MSComctlLib.Button)
Select Case Button.Key
    Case "tbNew"
        mnuNuevo_Click
    Case "tbOpen"
        mnuAbrir_Click
    Case "tbSave"
        mnuGuardar_Click
    Case "tbPrint"
        mnuImprimir_Click
    Case "tbFix"
        mnuSubirTono_Click
    Case "tbTool"
        mnuNuevo_Click
    Case "tbPlay"
        mnuCaptura_Click
    Case "tbStop"
        mnuDetener_Click
End Select
End Sub

Public Sub VerificarForms()
If (ActiveForm Is Nothing) Then HabilitarBotones "Nada"
End Sub

Public Sub ActualizarRecientes(NombreArchivo As String)
Dim i As Integer, Indice As Integer, EnLista As Boolean, Clave As String
For i = 1 To 4
    If NombreArchivo = mnuReciente(i).Tag Then
        Indice = i
        EnLista = True
    End If
Next i
If Not EnLista Then Indice = 4
If Indice > 1 Then
    For i = Indice To 2 Step -1
        mnuReciente(i).Caption = "&" & i & " " & mnuReciente(i - 1).Tag
        mnuReciente(i).Tag = mnuReciente(i - 1).Tag
        If mnuReciente(i).Caption <> "r" Then mnuReciente(i).Visible = True
    Next i
End If

```

```

        Next i
    End If
    mnuReciente(1).Caption = "&l " & NombreArchivo
    mnuReciente(1).Tag = NombreArchivo
    mnuReciente(1).Visible = True
    mnuReciente(0).Visible = True
    If Len(mnuReciente(1).Caption) > 40 Then
        mnuReciente(1).Caption = Recortar(mnuReciente(1).Caption)
    End If
    For i = 1 To 4
        Clave = "ArchivoReciente" & i
        If mnuReciente(i).Caption <> ("r") Then
            SaveSetting Aplicacion, Seccion, Clave, mnuReciente(i).Tag
            If Len(mnuReciente(i).Caption) > 40 Then
                mnuReciente(i).Caption = Recortar(mnuReciente(i).Caption)
            End If
        End If
    Next i
End Sub

Public Sub LeerRecientes()
    Dim ListaArchivos As Variant, i As Integer

    If GetSetting(Aplicacion, Seccion, "ArchivoReciente1") = Empty Then Exit Sub

    ListaArchivos = GetAllSettings(Aplicacion, Seccion)
    mnuReciente(0).Visible = True
    For i = 0 To UBound(ListaArchivos, 1)
        mnuReciente(i + 1).Caption = "&" & i + 1 & " " & ListaArchivos(i, 1)
        mnuReciente(i + 1).Tag = ListaArchivos(i, 1)
        If ListaArchivos(i, 1) <> "r" Then
            If Len(mnuReciente(i + 1).Caption) > 40 Then
                mnuReciente(i + 1).Caption = Recortar(mnuReciente(i +
1).Caption)
            End If
            mnuReciente(i + 1).Visible = True
        End If
    Next i
End Sub

Private Function Recortar(Archivo As String) As String
    Dim ff As String
    Dim P_1 As Integer, P_2 As Integer, Traslape As Boolean
    Dim TamTotal As Integer, Temp1 As Integer, Temp2 As Integer
    ff = "C:\WINDOWS\system\system32\oobe\images\qmark.acs"
    ff = Archivo
    TamTotal = Len(ff)

    P_1 = InStr(ff, "\")
    P_2 = InStrRev(ff, "\")

    Do While Traslape = False
        If TamTotal - P_2 + P_1 > 37 Then
            Traslape = True

```

```

Else
    If P_1 < P_2 Then
        Temp2 = P_2
        P_2 = InStrRev(ff, "\", Temp2 - 1)
        If P_1 < P_2 Then
            Temp1 = P_1
            P_1 = InStr(Temp1 + 1, ff, "\")
        Else
            P_2 = Temp2
            Traslape = True
        End If
    Else
        P_1 = Temp1
        Traslape = True
    End If
End If

Loop
Recortar = Left(ff, P_1) & "...\" & Right(ff, TamTotal - P_2)
End Function

offset
Private Sub cmdAceptar_Click()
Dim llave As String
llave = "Porcentaje"
SaveSetting Aplicacion, ToleranciaAlRuido, llave, CStr(sldOffset.Value)
Unload Me
End Sub

Private Sub cmdCancelar_Click()
Dim llave As String, valor As String

llave = "Porcentaje"
valor = GetSetting(Aplicacion, ToleranciaAlRuido, llave)
If valor = Empty Then
    frmMain.Offset = 100
Else
    frmMain.Offset = Val(valor)
End If
Unload Me
End Sub

Private Sub cmdColorMetronomo_Click()
On Error GoTo sa
frmMain.CommonDialog1.Flags = &H2 cdCCFullOpen
frmMain.CommonDialog1 = frmMain.CommonDialog1.Color &H10FFFF
frmMain.CommonDialog1.ShowColor
Exit Sub
sa:
End Sub

Private Sub Form_Load()
200     E8 E6&
200 232 230

```



```

sldOffset.Value = frmMain.Offset
lblNivelRuido.Caption = sldOffset.Value & " %"
SSTab1.TabEnabled(1) = False
End Sub

Private Sub Image1_Click()
sldOffset.Value = 100
End Sub

Private Sub sldOffset_Change()

lblNivelRuido.Caption = sldOffset.Value & " %"
frmMain.Offset = sldOffset.Value
End Sub

Private Sub sldOffset_Scroll()
sldOffset_Change
End Sub

subirtonos
Public Abajo As Integer, Arriba As Integer

Private Sub cmbTonos_GotFocus()
cmbTonos.Clear
Dim i As Integer

For i = Abajo To Arriba
    cmbTonos.AddItem i
Next i
End Sub

Private Sub cmdAceptar_Click()
Unload Me
End Sub

Private Sub cmdCancelar_Click()
cmbTonos.Text = 0
Unload Me
End Sub

Private Sub Form_Load()
cmbTonos.Clear
Dim i As Integer
For i = Abajo To Arriba
    cmbTonos.AddItem i
Next i
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
frmMain.NumTonos = Val(cmbTonos.Text)
frmMain.hwnd
End Sub

```

```

splash

Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
End Sub

Private Sub Form_Load()
    lblVersion.Caption = "Versión " & App.Major & "." & App.Minor & "." &
App.Revision
    lblProductName.Caption = App.Title
    fraMainFrame.Move (Me.ScaleWidth - fraMainFrame.Width) / 2,
(Me.ScaleHeight - fraMainFrame.Height) / 2

End Sub

Private Sub Frame1_Click()
    Unload Me
End Sub

fft
Const PI As Double = 3.14159265358979

Public Type Complex
    Real As Single
    Imag As Single
End Type

Private Sub Bitrev(n As Integer, ByRef Point() As Complex)
    Dim i As Integer, j As Integer, k As Integer
    Dim temp As Complex
    j = n / 2 + 1
    For i = 2 To n
        If i < j Then
            temp = Point(i)
            Point(i) = Point(j)
            Point(j) = temp
        End If
        k = n / 2
        Do While k < j And k <> 0
            j = j - k
            k = k / 2
        Loop
        j = j + k
    Next i
End Sub

Public Function fft(Vtiempo() As Single) As Single()
    Dim a As Integer, b As Integer, d As Integer, e As Integer
    Dim n As Integer, f As Integer, i As Integer, t As Single, tempResp() As
Single
    Dim u As Complex, W As Complex, temp As Complex, Point() As Complex
    d = 1
    n = UBound(Vtiempo)
    ReDim Point(n)

```

```

ReDim tempResp(n)
For i = 1 To n
    Point(i).Real = Vtiempo(i)
Next i
t = Log(n) / Log(2)
If t - Round(t) <> 0 Then
    MsgBox n
    Stop
End If
Bitrev n, Point

Do While d < n
    e = d
    d = d * 2
    u.Real = 1
    u.Imag = 0
    W.Real = Cos(PI / e)
    W.Imag = -Sin(PI / e) =sin(pi/e) para ifft

    For b = 1 To e
        For a = b To n Step d
            f = a + e
            temp.Real = Point(f).Real * u.Real - Point(f).Imag * u.Imag
            temp.Imag = Point(f).Real * u.Imag + Point(f).Imag * u.Real
            Point(f).Real = Point(a).Real - temp.Real
            Point(f).Imag = Point(a).Imag - temp.Imag
            Point(a).Real = Point(a).Real + temp.Real
            Point(a).Imag = Point(a).Imag + temp.Imag
        Next a
        temp.Real = u.Real
        u.Real = u.Real * W.Real - u.Imag * W.Imag
        u.Imag = temp.Real * W.Imag + u.Imag * W.Real
    Next b
Loop

For a = 1 To n / 2
    tempResp(a) = Point(a).Real ^ 2 + Point(a).Imag ^ 2
    tempResp(a) = Sqr(tempResp(a))
Next a
fft = tempResp
End Function

inicio
Public Mago As Object
Dim Pl As Object
Public Clase As TipoNota
Public Enum TipoNota
    Semicorchea = 1
    Corchea = 2
    CorcheaPuntillo = 3
    Negra = 4
    NegraPuntillo = 6
    Blanca = 8
    BlancaPuntillo = 12

```

```

    Redonda = 16
End Enum
Public Const Aplicacion As String = "SiReNoM"
Public Const Seccion As String = "Recientes"
Public Const ToleranciaAlRuido As String = "Offset"
Public Const ColorMetronomo As String = "ColorMetronomo"

Private Const radio As Integer = 4
Private Const Largo As Integer = 24
Private Const arco As Integer = 6
Private Const arco2 As Integer = 5
Public Const PI As Double = 3.14159265358979

Public Const HH_DISPLAY_TOPIC = &H1
Public Const hh_display_index = &H2
Public hwndayudahtml As Long, h As Long
Public Declare Function HtmlHelp Lib "HHCTRL.OCX" Alias "HtmlHelpA"
(ByVal hwndcaller As Long, ByVal pszfile As String, ByVal ucommand As
Long, ByVal dwdata As Long) As Long

Sub Main()
Dim i As Long, j As Long, k As Double
    frmSplash.Show
    frmSplash.ZOrder 0
    frmSplash.Refresh
    Set fMainForm = New frmMain
    Load frmMain
    frmMain.Show
    frmSplash.Show
    frmSplash.ZOrder 0
    frmSplash.Refresh
    Unload frmSplash
    Load frmEdicion
End Sub

Public Sub DibujarNota(X As Single, Y As Single, ByVal Tipo As TipoNota,
Pagina As Integer, Optional sinInvertir As Boolean = True)
Set P1 = frmMain.ActiveForm.picHoja(Pagina)
Select Case Tipo
Case Redonda, 14, 15, Is > 17                                Redonda 16
    P1.Circle (X, Y), radio, QBColor(0), , , 1
    P1.Circle (X, Y), radio, QBColor(0), , , 2
    P1.Circle (X, Y), radio, , , , 1
    P1.Circle (X, Y), radio, , , , 2
Case BlancaPuntillo, 10, 11, 13                                BlancaPuntillo 12
    P1.Circle (X, Y), radio
    If sinInvertir Then
        P1.Line (X + radio, Y)-(X + radio, Y - Largo)
    Else
        P1.Line (X - radio, Y)-(X - radio, Y + Largo)
    End If
    P1.DrawWidth = 3
    P1.PSet (X + 2 * radio, Y)
    P1.DrawWidth = 1

```

```

Case Blanca, 7, 9
    Pl.Circle (X, Y), radio
    Pl.Line (X + radio, Y)-(X + radio, Y - largo)
    If sinInvertir Then
        Pl.Line (X + radio, Y)-(X + radio, Y - Largo)
    Else
        Pl.Line (X - radio, Y)-(X - radio, Y + Largo)
    End If
Case NegraPuntillo
    Call DibNegra(X, Y, sinInvertir)
    Pl.DrawWidth = 3
    Pl.PSet (X + 2 * radio, Y)
    Pl.DrawWidth = 1
Case Negra, 5
    Call DibNegra(X, Y, sinInvertir)
Case CorcheaPuntillo
    Call DibNegra(X, Y, sinInvertir)
    Call colochito(X, Y, sinInvertir)
    Pl.DrawWidth = 3
    Pl.PSet (X + 2 * radio, Y)
    Pl.DrawWidth = 1
Case Corchea
    Call DibNegra(X, Y, sinInvertir)
    Call colochito(X, Y, sinInvertir)
Case Semicorchea
    Call DibNegra(X, Y, sinInvertir)
    Call colochito(X, Y, sinInvertir)
    If sinInvertir Then
        Call colochito(X, Y + 4, sinInvertir)
    Else
        Call colochito(X, Y - 4, sinInvertir)
    End If
Case Else
    MsgBox ("ninguno " & Tipo)

End Select
End Sub

Public Sub DibujarSilencio(X As Single, Y As Single, ByVal Tipo As
TipoNota, Pagina As Integer)
Dim
Set Pl = frmMain.ActiveForm.picHoja(Pagina)
Select Case Tipo
    Case Corchea
        Pl.DrawWidth = 2
        Pl.FillStyle = 0
        Pl.Line (X + 5, Y - 3)-(X - 4, Y + 12)
        Pl.DrawWidth = 1
        Pl.Circle (X, Y), 3
        Pl.FillStyle = 1
    Case Negra
        Pl.DrawWidth = 2
        Pl.FillStyle = 0
        Pl.Line (X - 5, Y - 3)-(X + 4, Y + 12)

```

```

        Pl.DrawWidth = 1
        Pl.Circle (X, Y), 3
        Pl.FillStyle = 1
    Case Blanca
        Pl.Line (X - 4, Y)-(X + 4, Y + 4), , BF
    Case Redonda
        Pl.Line (X - 4, Y)-(X + 4, Y - 4), , BF
End Select
End Sub

Private Sub colocchito(X As Single, Y As Single, sinInvertir)
Dim xx As Integer, yy As Integer
If sinInvertir Then
    Pl.Circle (X + radio, Y - Largo + arco), arco, , 0, PI / 2
    xx = X + radio + arco + arco2
    yy = Y - Largo + arco
    Pl.Circle (xx, yy), arco2, , PI, (3 * PI / 2)
Else
    Pl.Circle (X - radio + 1, Y + Largo - arco), arco, , 3 * PI / 2, 0
    xx = X - radio + arco + arco2 + 1
    yy = Y + Largo - arco
    Pl.Circle (xx, yy), arco2, , PI / 2, PI
End If
End Sub

Private Sub DibNegra(X As Single, Y As Single, sinInvertir As Boolean)
    Pl.FillStyle = 0
    Pl.Circle (X, Y), radio
    Pl.Line (X + radio, Y)-(X + radio, Y - largo)
    If sinInvertir Then
        Pl.Line (X + radio, Y)-(X + radio, Y - Largo)
    Else
        Pl.Line (X - radio, Y)-(X - radio, Y + Largo)
    End If
    Pl.FillStyle = 1
End Sub

Public Sub DibujarSostenido(X As Single, Y As Single)
verticales
If Pl Is Nothing Then
    Set Pl = frmMain.ActiveForm.picHoja(1)
End If
    Pl.Line (X - 16, Y - 5)-(X - 16, Y + 6)
    Pl.Line (X - 12, Y - 6)-(X - 12, Y + 5)
horizontales
    Pl.Line (X - 19, Y - 3)-(X - 9, Y - 4)
    Pl.Line (X - 19, Y + 4)-(X - 9, Y + 3)
End Sub

Public Sub DibujarBecuadro(X As Single, Y As Single)
verticales
    Pl.Line (X - 16, Y - 6)-(X - 16, Y + 3)
    Pl.Line (X - 12, Y - 3)-(X - 12, Y + 6)
horizontales

```

```

        Pl.Line (X - 16, Y - 2)-(X - 12, Y - 3)
        Pl.Line (X - 16, Y + 3)-(X - 12, Y + 2)
    End Sub
    Public Function PrimerMaximo(Vector() As Single) As Integer
    Dim i As Integer, t As Integer, MaxPos As Integer
    t = UBound(Vector)
    MaxPos = 1
    For i = 2 To t
        If Vector(i) > Vector(MaxPos) Then
            MaxPos = i
        End If
    Next i
    PrimerMaximo = MaxPos
    End Function

    Public Function PrimerMinimo(Vector() As Single) As Integer
    Dim i As Integer, t As Integer, MinPos As Integer
    t = UBound(Vector)
    MinPos = 1
    For i = 2 To t
        If Vector(i) < Vector(MinPos) Then
            MinPos = i
        End If
    Next i
    PrimerMinimo = MinPos
    End Function

```

B. TABLAS ESTADÍSTICAS

En esta sección se presentan las tablas obtenidas después de realizar una serie de pruebas bajo diferentes condiciones a fin de determinar los valores medios que debían ser identificadas por la red neuronal.

La tabla B-1 muestra la máxima energía obtenida en el dominio de la frecuencia, así como su ubicación en el espectro. Para dar una idea del comportamiento registrado también se tomo en cuenta la media, la desviación media y la moda para cada una de las notas que serían identificadas.

Nota	Parámetro	frec	E max
do	promedio	539.0	10.96
	media	538.3	11.06
	desv. media	1.3	1.26
	moda	538.3	
do#	promedio	560.6	12.36
	media	559.9	12.09
	desv. media	1.3	0.79
	moda	559.9	
re	promedio	602.9	15.97
	media	602.9	15.53
	desv. media	0.0	1.24
	moda	602.9	
re#	promedio	630.9	6.17
	media	635.2	5.89
	desv. media	5.7	1.30
	moda	635.2	

Nota	Parámetro	frec	E max
mi	promedio	662.9	5.96
	media	667.5	5.75
	desv. media	5.3	0.99
	moda	667.5	
fa	promedio	704.1	8.76
	media	699.8	7.93
	desv. media	5.2	2.28
	moda	699.8	
fa#	promedio	747.2	10.12
	media	742.9	8.08
	desv. media	5.2	4.39
	moda	742.9	
sol	promedio	788.1	26.19
	media	786.0	26.33
	desv. media	3.4	4.08
	moda	786.0	
sol#	promedio	844.1	35.10
	media	839.8	29.89
	desv. media	5.2	8.76
	moda	839.8	
la	promedio	882.9	60.96
	media	882.9	62.24
	desv. media	0.0	5.67
	moda	882.9	
la#	promedio	938.1	45.74
	media	936.7	44.50
	desv. media	2.5	10.96

Nota	Parámetro	frec	E max
	moda	936.7	
si	promedio	1001.3	32.33
	media	1001.3	34.22
	desv. media	0.0	5.91
	moda	1001.3	
do2	promedio	1055.1	62.93
	media	1055.1	66.66
	desv. media	0.0	7.02
	moda	1055.1	
do2#	promedio	1129.1	68.50
	media	1130.5	73.31
	desv. media	2.5	14.72
	moda	1130.5	
re2	promedio	1190.8	112.33
	media	1195.1	112.15
	desv. media	5.2	12.21
	moda	1195.1	
re2#	promedio	1260.4	16.53
	media	1259.7	14.22
	desv. media	1.3	4.53
	moda	1259.7	
mi2	promedio	1335.8	87.81
	media	1335.1	92.61
	desv. media	2.7	9.80
	moda	1335.1	
fa2	promedio	1416.9	69.91
	media	1421.2	61.30

Nota	Parámetro	frec	E max
	desv. media	5.2	18.92
	moda	1421.2	
fa2#	promedio	1488.7	44.81
	media	1485.8	41.79
	desv. media	4.2	8.66
	moda	1485.8	
sol2	promedio	1584.1	116.98
	media	1582.7	120.09
	desv. media	6.2	20.55
	moda	1582.7	

Tabla B-1. Energía máxima en el dominio de la frecuencia

Estos valores fueron usados para establecer el límite usado para diferenciar la presencia de una nota real a la aparente presencia de una nota producida por la elevada sensibilidad de la red neuronal, aún en presencia de un moderado nivel de ruido ambiente.

En la figura B-1 puede notarse la enorme variación en la magnitud de las señales analizadas, lo cual se traduce en un mayor grado de dificultad al momento realizarse el proceso de entrenamiento de la red, ya que mientras unos patrones estaban acercándose al límite de la memorización, con su consecuente pérdida de generalidad, habían otros que aún no estaban completamente diferenciados.

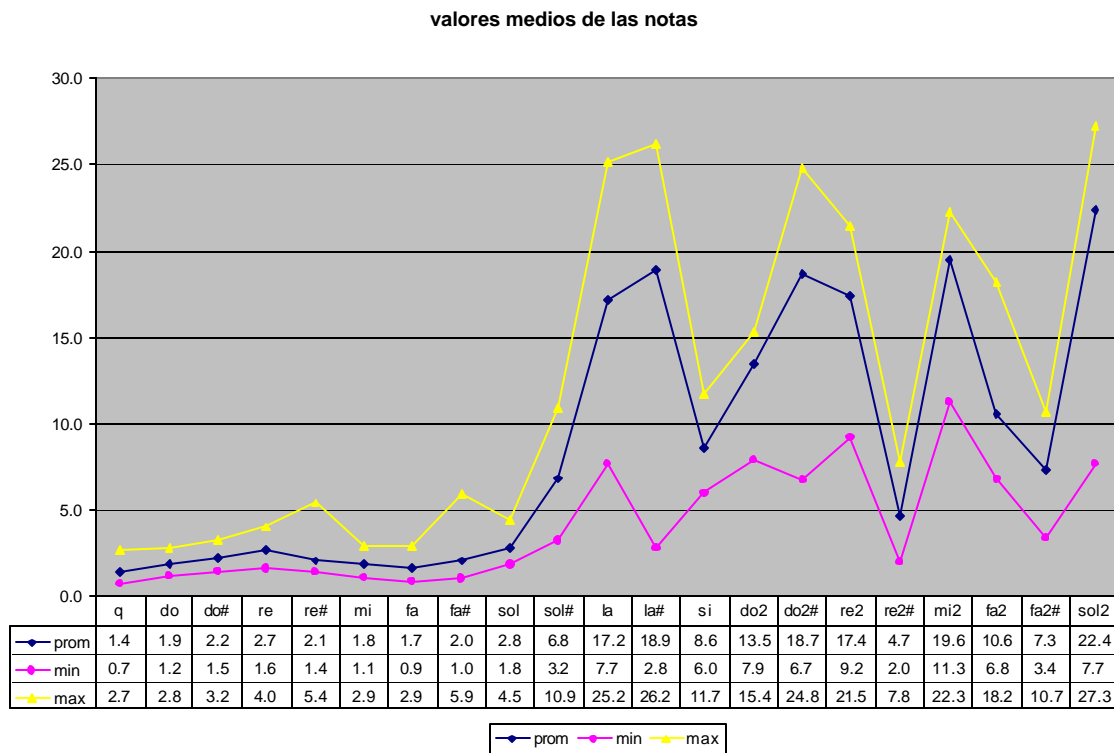


Figura B-1. Magnitud media de las notas digitalizadas

Como dificultad adicional se encuentra el hecho de que esas variaciones también son visibles dentro de la misma nota, ya que el nivel energético de las mismas no se mantiene constante durante todo el tiempo, sino que presenta diversos valores durante la ejecución de la nota, tal como puede verse en forma general en la figura B-2

Nivel energético de las notas

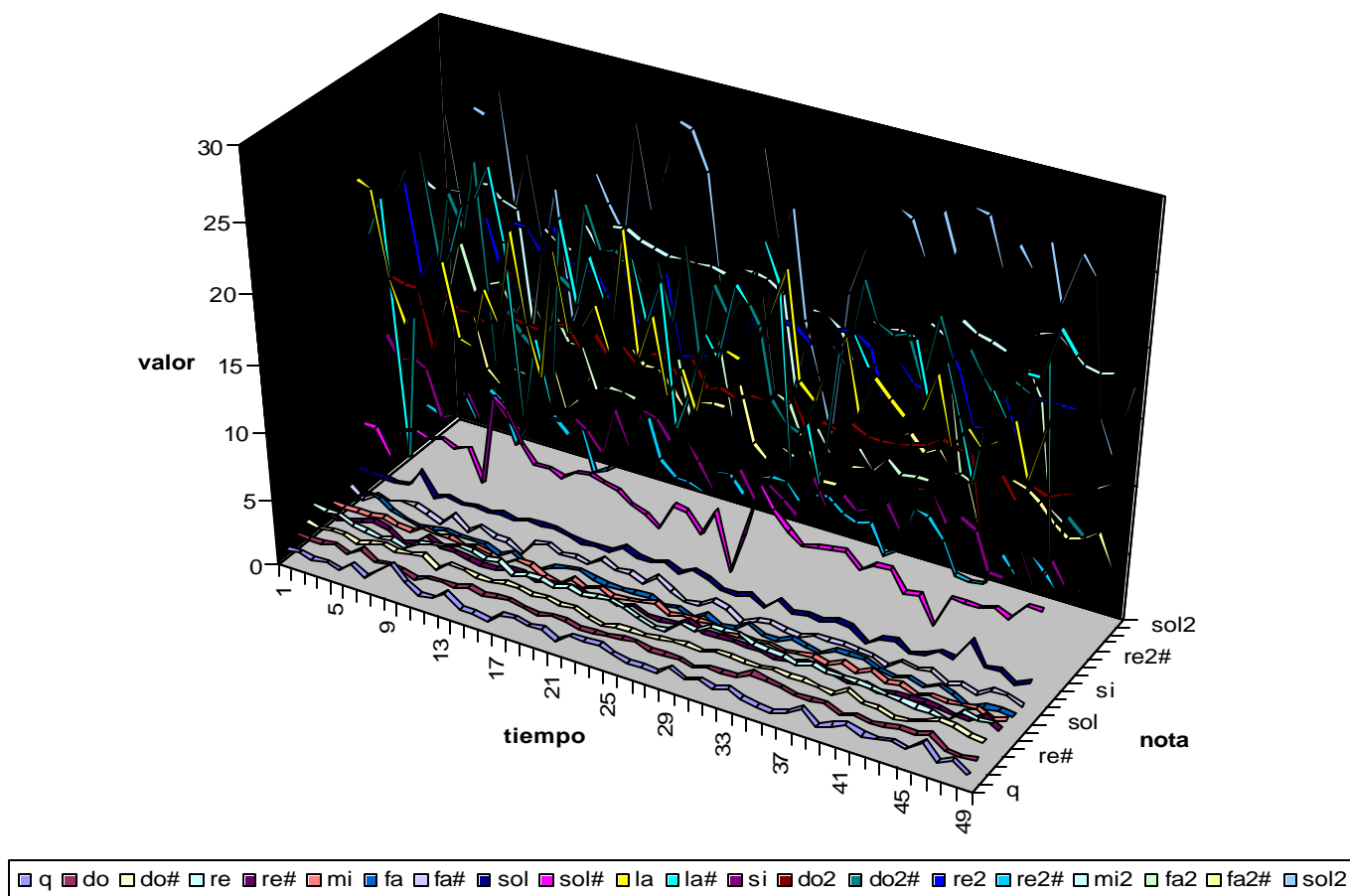


Figura B-2. Variaciones energéticas de las notas digitalizadas

