Universidad Autónoma Metropolitana Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Modalidad: Proyecto Tecnológico.

Proyecto:
Estudio experimental de la aproximación de funciones con redes neuronales wavenets

José Luis Ramírez Rojano 205204774

Asesor: Dr. Oscar Herrera Alcántara

Trimestre
14-I

Fecha: 12-Marzo-2014

Resumen

Los *wavelets* [1] son funciones matemáticas que pueden usarse como bases para descomponer otras funciones, algo similar a lo que en la transformada de Fourier son las funciones seno y coseno. En el caso de los *wavelets* se representa una señal en términos de versiones trasladadas y dilatadas de una función llamada "mother *wavelet*[1]".

Las redes neuronales [2] por su parte, son un paradigma de aprendizaje y procesamiento automático inspirado en modelos conexionistas. Se trata de un sistema de interconexión de neuronas en una red que colaboran para producir un estímulo de salida.

Las redes *wavenet* [3] usan funciones *wavelet* como bases de descomposición para aproximar funciones provistas en su entrada

Los algoritmos genéticos [4] son parte de modelos evolutivos que hacen uso de operadores de mutación, cruzamiento y selección inspirados en la teoría de la evolución de los seres vivos.

Las redes neuronales han adquirido gran popularidad por la "simplicidad" al aproximar funciones como modelos de caja negra, lo cual permite analizar fenómenos o problemas de ingeniería en donde no se tiene un modelo claro o explícito.

Existe un estudio acerca del cómo y por qué funcionan las redes neuronales como aproximadores que es tema del Análisis Funcional. También relacionado con este tema está el Teorema de Aproximación Universal [2] y el Análisis Multiresolución [5] [6].

En forma general se puede decir que una función en L^2 se puede descomponer como suma de otras funciones base mediante superposición. La descomposición puede ser exacta o una aproximación. Así como un vector $\vec{v}$= (x, y) en R^2 se puede expresar como una suma de los vectores unitarios $\hat{\imath}$=(1,0) y $\hat{\jmath}$=(0,1) (que forman una base vectorial) una función f en L^2 se puede expresar como suma de funciones base.

Se puede demostrar que la proyección horizontal de $\vec{v}$ se obtiene mediante el producto punto entre $\vec{v}$ e $\hat{\imath}$, esto es $x= \vec{v}\cdot\hat{\imath}$, en tanto que la proyección vertical de $\vec{v}$ se obtiene mediante el producto punto entre $\vec{v}$ y $\hat{\jmath}$, esto es $y=\vec{v}\cdot\hat{\jmath}$, en donde se aplica que $\hat{\imath}$ y $\hat{\jmath}$ son ortogonales ($\hat{\imath}\cdot\hat{\jmath}$ = 0). Las funciones sen(x) y cos(x) son funciones ortogonales usadas en el Análisis de Fourier para descomponer funciones, pero no son las únicas funciones base.

Los *wavelets* son familias de funciones que combinan importantes propiedades, tales como localización en tiempo y frecuencia, ortogonalidad y soporte compacto.

---

[1]**Mother *Wavelet***: Quizá una traducción al español podría ser "ondita principal" en lugar de "wavelet madre", pero no se acostumbra nombrarla así en la literatura. El término se originó de un acrónimo de wavelet en francés "wave net": ondita. En este trabajo se usará wavelet, y no ondita.

Las redes neuronales basadas en *wavelets* o *wavenets*, incorporan las ventajas de la descomposición de señales mediante wavelets con la propiedad de generalización y de aproximación universal de las redes neuronales tradicionales. [7]

Anteriormente, se han explorado redes neuronales con *wavelet*s en el dominio continuo y, por otro lado, en el dominio discreto. En este trabajo se explora una combinación de ambos, además de introducir el uso de *wavelet*s de soporte compacto paramétricos.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

Introducción

Los *wavelets* son funciones matemáticas que pueden usarse como bases para descomponer otras funciones, algo similar a lo que en la transformada de Fourier son las funciones seno y coseno. En el caso de los *wavelets* se representa una señal en términos de versiones trasladadas y dilatadas de una función llamada "mother *wavelet*".

Las redes neuronales por su parte, son un paradigma de aprendizaje y procesamiento automático inspirado en modelos conexionistas. Se trata de un sistema de interconexión de neuronas en una red que colaboran para producir un estímulo de salida.

Las redes *wavenet* usan funciones *wavelet* como bases de descomposición para aproximar funciones provistas en su entrada. En cierto modelo de *wavenet*, la salida está dada por:

$$y(u) = \sum_{i=1}^{M} \varphi(a_i, b_i)$$

Donde $\varphi(a_i, b_i)$ representa la función wavelet con parámetro de escalamiento a_i y de traslación b_i.

Los algoritmos genéticos son parte de modelos evolutivos que hacen uso de operadores de mutación, cruzamiento y selección inspirados en la teoría de la evolución de los seres vivos.

Antecedentes

1. Referencias internas

En la UAM Azcapotzalco, se han desarrollado varios proyectos terminales que utilizan redes neuronales o *wavelet*s, por ejemplo:

- *Implementación de una aplicación software para procesamiento de imágenes con wavelets y bancos de filtros paramétricos de reconstrucción perfecta*, el cual fue propuesto en el trimestre 10-O [8].

Este proyecto realiza procesamiento de imágenes con *wavelet*s, y no redes neuronales *wavenets* para aproximar funciones, como se pretende en el presente trabajo.

- *Estudio experimental de la tolerancia al ruido de imágenes procesadas con la transformada wavelet redundante con árbol dual*, el cual fue propuesto en el trimes-tre 11-I [9].

Este proyecto realiza procesamiento de imágenes con *wavelet*s, pero no usa redes neuronales *wavenets* para aproximar funciones, como se pretende en el presente trabajo.

- *Reconocimiento de patrones en mamografías usando redes neuronales artificiales para detectar cáncer,* el cual fue propuesto en el trimestre 07-I [10].

En este proyecto, se reconocen patrones con redes neuronales que no son *wavenets*, como las que se propone usar en este trabajo.

- *Sistema de identificación de personas a través del iris mediante análisis de texturas por filtrado de wavelets,* el cual fue propuesto en el trimestre 09-P [11].

Se usan *wavelet*s como herramienta de procesamiento (filtrado) de texturas y no como aproximadores de funciones, como en las redes neuronales *wavenets.*

- *Estudio experimental de la aproximación de filtros digitales paramétricos para implementar la transformada wavelet discreta con polinomios evolutivos,* el cual fue propuesto en el trimestre 10-O [12].

Se trabajó con aproximación de filtros digitales paramétricos, pero éstos no se incluyen como parte de redes neuronales *wavenets* como las que se propone usar en este trabajo.

- *Clasificación de llantos de bebé con wavelets,* el cual fue propuesto en el trimestre 10-O [13].

Se emplea la transformada *wavelet* para procesar llantos de bebé, pero no se usan redes neuronales *wavenet* para aproximar funciones.4

Quizá el proyecto más parecido es el llamado *Implementación de una aplicación software para procesamiento de imágenes con wavelets y bancos de filtros paramétricos de reconstrucción perfecta* en donde se implementó una Interfaz Gráfica de Usuario (IGU) para mostrar el procesamiento de imágenes con filtros paramétricos de soporte compacto. En este trabajo se usarán también *wavelet*s de soporte compacto paramétricos, pero en lugar de procesar imágenes (que son funciones en dos dimensiones), se aproximarán funciones en una dimensión con redes neuronales *wavenets*.

Otro proyecto muy relacionado es el llamado *Estudio experimental de la tolerancia al ruido de imágenes procesadas con la transformada wavelet redundante con árbol dual*, en donde también se usa la transformada *wavelet*, pero con ella se procesan imágenes con redundancia según un árbol dual, y en este trabajo se usan *wavelet*s ortogonales de soporte compacto en redes neuronales *wavenets.*

2. Referencias externas

Existen, en general, muchos trabajos acerca de *wavelet*s que no conviene listar, al igual que de redes neuronales, sólo se mencionarán algunos.

Un trabajo relacionado con éste es el llamado *Initialization by Selection for Wavelet Network Training* [14] en donde se usan funciones *wavelet* en el dominio continuo y discreto para aproximar funciones, pero no se aplican algoritmos genéticos.

Otro trabajo relacionado es el llamado *Método Basado en Redes Neuronales Wavelet para Eliminar Ruido en Espectros Estelares* [15] en el cual se implementa una red neuronal

wavelet para eliminar ruido en espectros estelares, pero no se aproximan funciones y no se aplican algoritmos genéticos.

Otro trabajo relacionado es el llamado *Modelación de Procesos Industriales (Redes Neuronales Wavelet)* [16] en el cual se muestra el uso de *wavelets* para la obtención de modelos y la aplicación de éstos en un proceso de sedimentación de una industria de fabricación de muebles de baño, pero no se aplican algoritmos genéticos, ni se usan redes neuronales *wavenet* para aproximar funciones.

Otro trabajo relacionado es el llamado *Reconocimiento del habla mediante una Red Neuronal con pre-procesamiento wavelet* [17] en el cual se hace un estudio de reconocimiento del habla que utiliza como pre- procesamiento una función *wavelet* no ortogonal, el reconocimiento del habla se hace con una red neuronal, pero no se aplican algoritmos genéticos, ni se usan redes neuronales *wavenet* para aproximar funciones.

Justificación

El problema de la aproximación de funciones es un tema de interés matemático y de ingeniería, ya que es esencial cuando se quiere representar una función en serie de otras más sencillas, como lo son los polinomios en las series de Taylor o funciones trigonométricas en el Análisis de Fourier.
Las redes neuronales han adquirido gran popularidad por la "simplicidad" al aproximar funciones como modelos de caja negra, lo cual permite analizar fenómenos o problemas de ingeniería en donde no se tiene un modelo claro o explícito.

Existe un estudio serio acerca del cómo y por qué funcionan las redes neuronales como aproximadores que es tema del Análisis Funcional. También relacionado con este tema está el Teorema de Aproximación Universal y el Análisis Multiresolución.

En forma general se puede decir que una función en L2 se puede descomponer como suma de otras funciones base mediante superposición. La descomposición puede ser exacta o una aproximación. Así como un vector $\vec{vv}=(xx,yy)$ en R2 se puede expresar como una suma de los vectores unitarios (que forman una base vectorial) $\hat{ii}=(1,0)$ y $\hat{jj}=(0,1)$, una función f en L2 se puede expresar como suma de funciones base.

Se puede demostrar que la proyección horizontal de $\vec{vv}$ se obtiene mediante el producto punto entre $\vec{vv}$ e $\hat{ii}$, esto es $xx=\vec{vv}\cdot\hat{ii}$, en tanto que la proyección vertical de $\vec{vv}$ se obtiene mediante el producto punto entre $\vec{vv}$ y $\hat{jj}$, esto es $yy=\vec{vv}\cdot\hat{jj}$, en donde se aplica que $\hat{i}$y $\hat{jj}$ son ortogonales ($\hat{ii}\cdot\hat{jj}=0$). Las funciones sen(x) y cos(x) son funciones ortogonales usadas en el Análisis de Fourier para descomponer funciones, pero no son las únicas funciones base.

Los *wavelet*s son familias de funciones que combinan importantes propiedades, tales como localización en tiempo y frecuencia, ortogonalidad y soporte compacto.

Las redes neuronales basadas en *wavelet*s o *wavenets*, incorporan las ventajas de la descomposición de señales mediante *wavelet*s con la propiedad de generalización y de aproximación universal de las redes neuronales tradicionales.

Anteriormente, se han explorado redes neuronales con *wavelet*s en el dominio continuo y, por otro lado, en el dominio discreto. En este trabajo se explora una combinación de ambos, además de introducir el uso de *wavelet*s de soporte compacto paramétricos.

Objetivo General

Implementar algoritmos que permitan aproximar funciones matemáticas unidimensionales discretas mediante redes neuronales tipo *wavenet*.

Objetivos Particulares

- Implementar el algoritmo de cascada para aproximar el valor de wavelets ortogonales de soporte compacto paramétricos.

- Implementar una red neuronal *wavenet* para aproximar funciones matemáticas unidimensionales discretas.

- Implementar un algoritmo genético para optimizar los parámetros de escalamiento y traslación de las funciones wavelet en la red neuronal *wavenet*.

- Implementar una IGU[2] para facilitar la selección manual de parámetros libres de wavelets de soporte compacto, así como la visualización de resultados de los experimentos para aproximar funciones matemáticas unidimensionales discretas.

[2] **IGU** (Interfaz Gráfica de Usuario)

Marco Teorico

Descripción Tecnica

Para facilitar la explicación del presente trabajo a realizar se usan diagramas de bloques.

Diagrama de bloque de nivel 0. (Fig. 1).

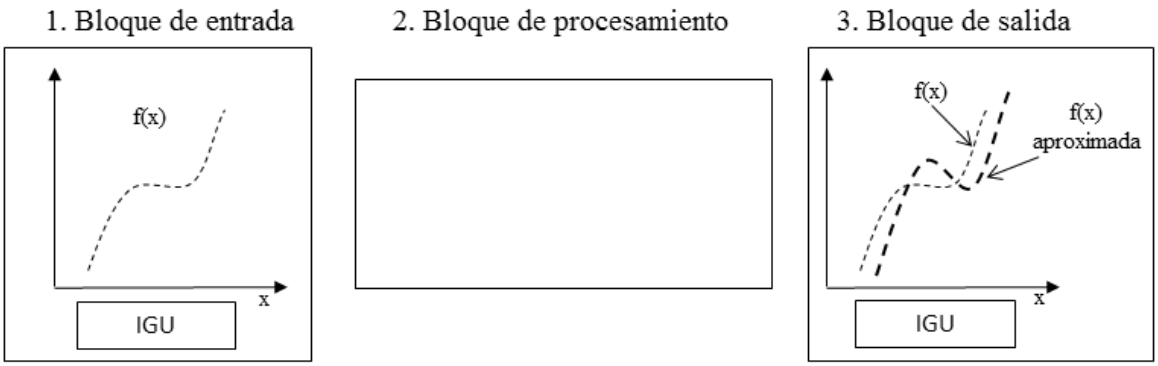


Fig. 1 Diagrama de bloque nivel 0.

1. La entrada al sistema es una función discreta *f (x)* de una variable temporal discreta x y parámetros proporcionados por el usuario a través de una IGU.

2. El procesamiento se realiza con una red neuronal tipo wavenet.

3. La salida del sistema es una función discreta que aproxima a *f (x).*

Según se aprecia en la Figura 1, en donde se numeran los bloques para identificarlos.

Diagrama de bloque de nivel 1.

Se muestra el diagrama a bloques más detallado (Fig. 2).

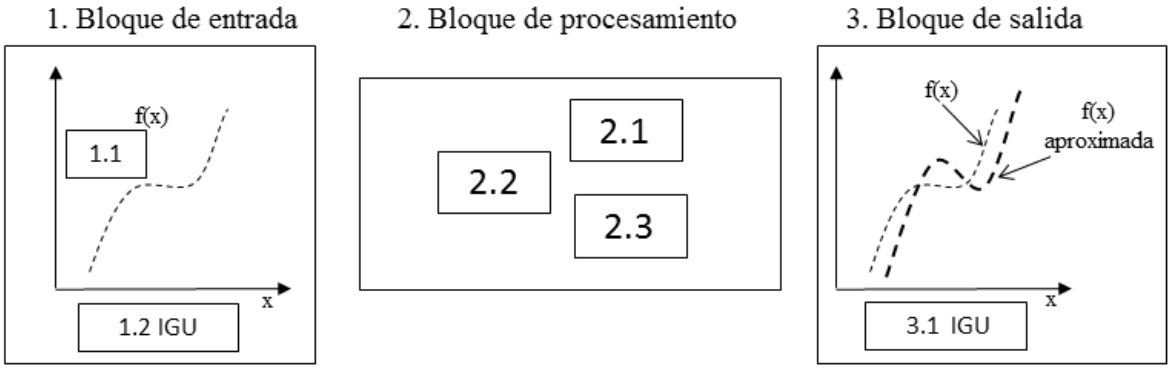


Fig. 2 Diagrama de bloque nivel 1.

El bloque de entrada incluye:

1.1. Se lee un archivo de texto, donde está la función a aproximar.

1.2. IGU. Permite que el usuario interactúe proporcionando valores de los parámetros libres para el algoritmo de cascada que permite aproximar una $\varphi(a_i, b_i)$

El bloque de procesamiento incluye los siguientes sub-bloques:

2.1. Red wavenet. Aquí se hace la aproximación de *f(x).*

2.2. Algoritmo Genético. Busca valores de (a_i, b_i) para $\varphi(a_i, b_i)$ a fin de aproximar *f(x)* mediante superposición en una red neuronal *wavenet* de los parámetros libres: α_4, α_6, β_6.

2.3. Algoritmo de Cascada [18]. Permite estimar el valor de $\varphi(a_i, b_i)$ a partir de φ(x) a la que se le aplica una traslación y una dilatación $\varphi((x - b_i)/a_i)$ cuya aproximación a su vez depende de los parámetros libres.

El bloque de salida incluye:

3.1. IGU. Permite visualizar la aproximación y los valores de los parámetros (a_i, b_i) después de la ejecución.

Especificación Técnica

Por facilidad se usa la siguiente notación para describir la comunicación entre los bloques: Bloque origen - Bloque destino. (Fig. 3 y Fig. 4).

a. Comunicación del bloque 1.1 al 2.1
b. Comunicación del bloque 2.1 al 2.2
c. Comunicación del bloque 1.2 al 2.3
d. Comunicación del bloque 2.3 al 2.1
e. Comunicación del bloque 2.2 al 2.1
f. Comunicación del bloque 2.1 al 3.1

Comunicación entre bloques.

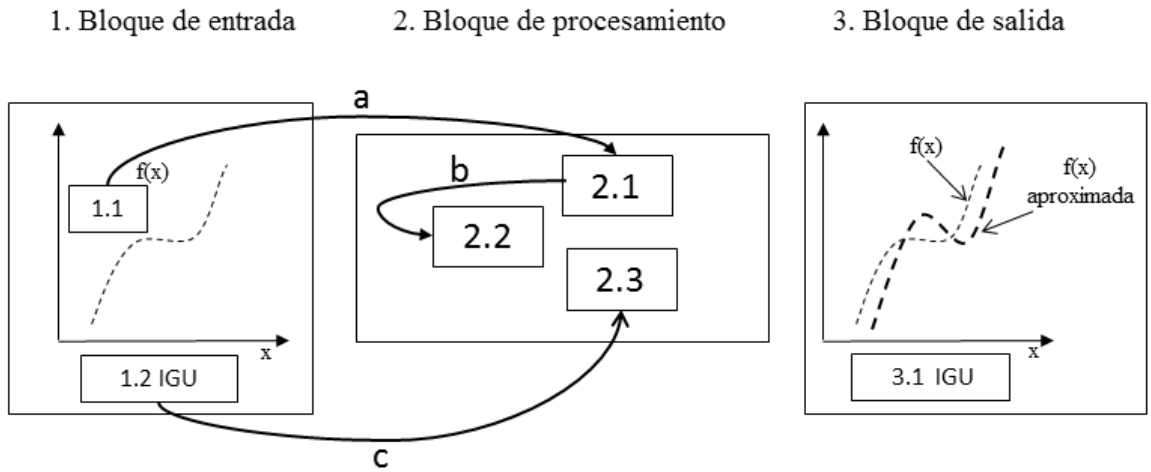


Fig. 3 Comunicación de los bloques a detalle (a, b, c).

a. La red wavenet lee los valores de la función discreta desde un archivo de texto simple, TXT.

b. La red wavenet provee la función de evaluación para el algoritmo genético cuyo fenotipo codifica los parámetros (a_i, b_i) y ambos serán programados en lenguaje Java [19].

c. La IGU permite que el usuario introduzca los valores de los parámetros de los wavelet de soporte compacto cuyas ecuaciones están dadas por:

$h4 = \{h4[0], h4[1], h4[2], h4[3]\}$

$$h_4[0] = \frac{1}{4} + \frac{1}{2\sqrt{2}} cos \propto_4 \qquad h_4[1] = \frac{1}{4} + \frac{1}{2\sqrt{2}} sin \propto_4$$

$$h_4[2] = \frac{1}{4} - \frac{1}{2\sqrt{2}} cos \propto_4 \qquad h_4[3] = \frac{1}{4} - \frac{1}{2\sqrt{2}} sin \propto_4$$

$h_6 = \{ h_6[0], h_6[1], h_6[2], h_6[3], h_6[4], h_6[5] \}$

$$h_6[0] = \frac{1}{8} - \frac{1}{4\sqrt{2}} cos \propto_6 + \frac{p}{2} cos\beta_6 \qquad h_6[1] = \frac{1}{8} - \frac{1}{4\sqrt{2}} sin \propto_6 + \frac{p}{2} sin\beta_6$$

$$h_6[2] = \frac{1}{4} - \frac{1}{2\sqrt{2}} cos \propto_6 \qquad h_6[3] = \frac{1}{4} - \frac{1}{2\sqrt{2}} sin \propto_6$$

$$h_6[4] = \frac{1}{8} + \frac{1}{4\sqrt{2}} cos \propto_6 - \frac{p}{2} cos\beta_6 \qquad h_6[5] = \frac{1}{8} + \frac{1}{4\sqrt{2}} sin \propto_6 - \frac{p}{2} sin\beta_6$$

Donde $p = \frac{1}{2}\sqrt{1 + sen\left(\propto_6 + \frac{\pi}{4}\right)}\, para\ \propto_6, \beta_6 \epsilon [0,2\pi]$

Sólo se listan las ecuaciones para los casos de longitud cuatro y seis, este proyecto se limita a usar las listadas anteriormente, si se desea conocer el resto de estas (α_8, β_8, γ_8, θ_8; α_{10}, β_{10}, γ_{10}, θ_{10}, δ_{10}), se pueden consultar: "Aplicación de algoritmos genéticos a la compresión de imágenes con evolets" [20], "Optimization of Parameterized Compactly Supported Orthogonal Wavelets for Data Compression" [21] y "On the best evolutionary wavelet based filter to compress a specific signal" [22].

Esta IGU también se implementó en Java.

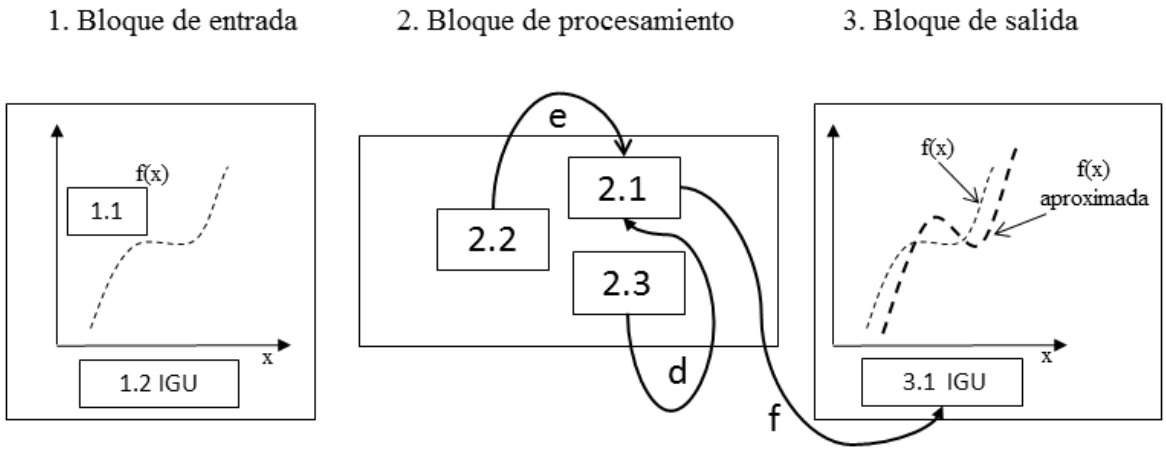


Fig. 4 Comunicación de los bloques a detalle (d, e, f).

d. El algoritmo de cascada provee una aproximación del valor de $\varphi(a_i, b_i)$ para la red *wavenet*. El algoritmo de cascada se implementó en lenguaje Java.

e. La red *wavenet* se comunica con el algoritmo genético para optimizar los parámetros (a_i, b_i) que minimizan el error de aproximación (función de aptitud) entre la función discreta dada y la provista por la red *wavenet*. La comunicación es entre objetos de Java.

f. Los datos provistos por la red *wavenet*, en formato simple en un archivo TXT, se procesarán en el bloque de salida a fin de poder visualizar:

   I. Los parámetros (a_i, b_i).
   II. La gráfica del wavelet paramétrico aproximado con el algoritmo de cascada.
   III. La aproximación de la función *f (x)* con la red neuronal.

Se calculan aproximaciones de las siguientes funciones:

I. $Sen(x)$

II. $cos(x)$

III. $$f(x) = \begin{cases} -2.186x - 12.864 & parax \in [-10,-2] \\ 4.246x & parax \in [-2,0] \\ 10exp(-0.05x - 0.5)sin\big(x(0.003x + 0.7)\big) & parax \in [0,10] \end{cases}$$

Con 10, 100 y 1000 muestras (puntos de la señal discreta).

Se reportará el error cuadrático medio y la relación señal a ruido. Al tratarse de un estudio experimental, el valor del error no determinará la culminación o no del proyecto. Aunque sí se espera un resultado aceptable.

Las funciones sen (x) y cos (x) son de particular interés porque, como ya se mencionó, son una base ortogonal en el Análisis de Fourier para descomponer otras funciones, así que si se pueden aproximar sen (x) y cos (x) se puede aproximar (al menos conceptualmente) el mismo conjunto de funciones que con el Análisis de Fourier. Respecto a la aproximación de *f (x)* es de particular interés porque es una función comúnmente utilizada para probar el desempeño de redes neuronales, toda vez que tiene un comportamiento no estacionario (tiene un abrupto cambio en su comportamiento de lineal a no lineal).

Desarrollo del proyecto

Descripcion de IGU

Se describe la IGU con cada uno de sus componentes:

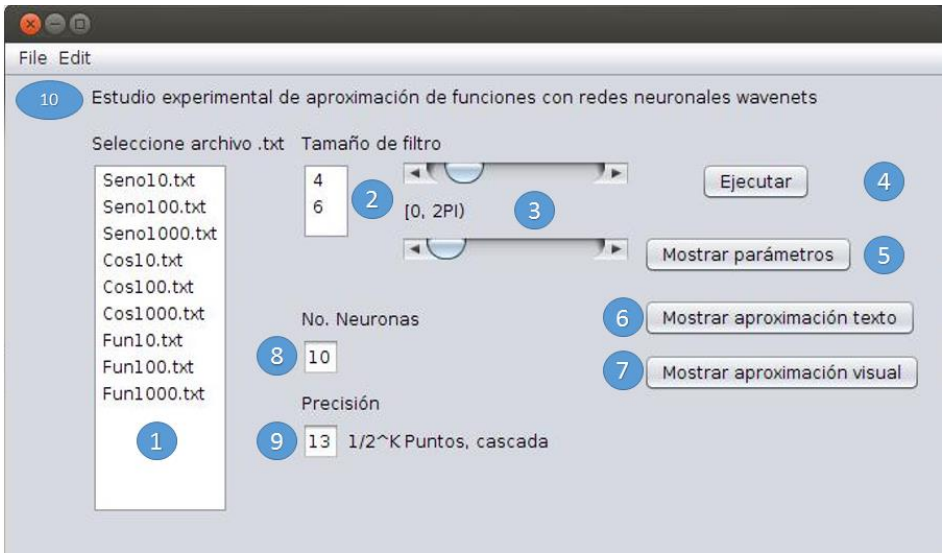


Fig. 5 Interfaz Gráfica de Usuario (IGU)

1. La IGU cuenta con una lista que permite elegir la función a aproximar, seno, coseno y Fun cada una con 10, 100 y 1000 puntos.

2. La ventana tamaño del filtro permite elegir entre un filtro tamaño 4 y uno tamaño 6.

3. Por medio de las barras de desplazamiento se puede elegir el valor de $\propto y\ \beta$ (tamaño del filtro) entre el rango 0 a 2π.

4. El botón ejecutar permite que se realice la aproximación de funciones usando las redes neuronales wavenet.

5. El botón mostrar parámetros nos muestra un archivo .txt con los parámetros a y b aproximados.

6. El botón Mostrar Aproximación Texto nos muestra un archivo .txt con los valores de la función aproximada.

7. El botón Mostrar Aproximación Visual nos despliega las gráficas de la función original vs la función aproximada.

8. La ventana No de Neuronas nos permite elegir el número de neuronas que se requieren para la red neuronal.

9. La ventana precisión nos permite introducir 2k puntos del algoritmo cascada.

10. Menú File – Exit, permite cerrar el programa.

Documentación del Código Fuente

Acontinuacion se muestra la descripción de la documentación y la estructura del código fuente.

Paquetes examples

Clases

- class **GeneticoVGA**
- class **Genetic**
- class **WVN**
- class **UsaCon**
- class **Convolucionador**
- class **IGUWavenet**
- class **ParametrosIGU**
- class **TestTxt**

Documentación de las clases

Referencia de la Clase examples.Convolucionador

Métodos públicos estáticos

- static void **convolve** (double[] out, double[] in, int dataSize, double[] kernel, int kernelSize)
- static void **convolvex** (double[] out, double[] in, int dataSize, double[] kernel, int kernelSize)

Descripción detallada

Definición en la línea 919 del archivo GeneticoVGA.java.

Documentación de las funciones miembro

static void examples.Convolucionador.convolve (double[] out*, double[]* in*, int* dataSize*, double[]* kernel*, int* kernelSize*)* `[static]`

Definición en la línea 921 del archivo GeneticoVGA.java.

static void examples.Convolucionador.convolvex (double[] out*, double[]* in*, int* dataSize*, double[]* kernel*, int* kernelSize*)* `[static]`

Definición en la línea 933 del archivo GeneticoVGA.java.

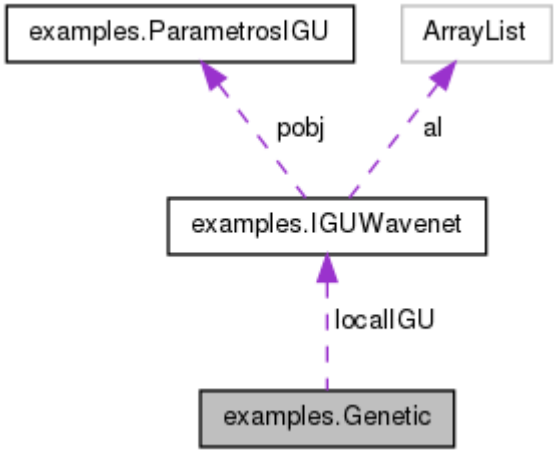
Gráfico de llamadas a esta función:

La documentación para esta clase fue generada a partir del siguiente fichero:

- **GeneticoVGA.java**

Referencia de la Clase examples.Genetic

Diagrama de colaboración para examples.Genetic:



Métodos públicos

- void **setIGU** (**IGUWavenet** iguwavenet)
- **Genetic** ()
- **Genetic** (**IGUWavenet** iguw)

Funciones del 'package'

- void **iniciaRandomPob** ()
- void **Mutacion** (int **Pmut**)
- void **Cruzamiento** (int **Pcruz**)
- double **valorpeso** (int indiv, int bitinicial, double escala)
- void **asignaFitness** (int indiv, boolean createfiles)
- void **asignarValoraPesosdeIndividuosyEvalua** ()
- void **Ordenamiento** ()

Atributos del 'package'

- boolean[][] **Poblacion**
- int **TamPob**
- int **TotalBitsPob**
- int **BitsIndiv**
- double **Fitness** []
- boolean **mostrarBits** = false
- int **NEURONAS**
- int **PRECSIGNOMAGNITUD**
- int **GENERACIONES**
- int **Pmut**
- int **Pcruz**
- int **MAGNITUD**
- double **ESCALA**
- **IGUWavenet localIGU**

Descripción detallada

Definición en la línea 32 del archivo GeneticoVGA.java.

Documentación del constructor y destructor

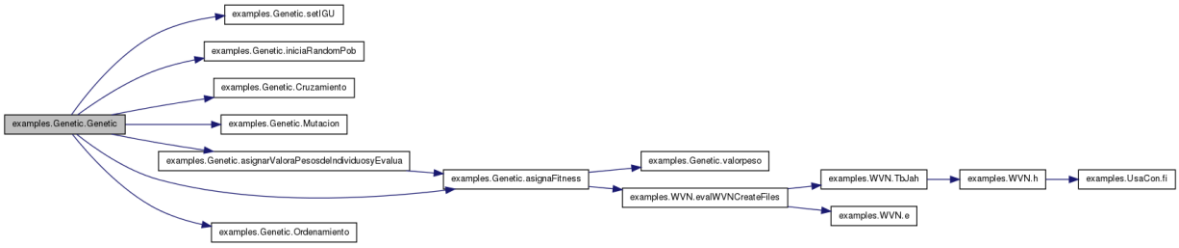
examples.Genetic.Genetic ()

Definición en la línea 55 del archivo GeneticoVGA.java.

examples.Genetic.Genetic (IGUWavenet iguw*)*

Definición en la línea 59 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:



Documentación de las funciones miembro

void examples.Genetic.asignaFitness (int indiv*, boolean* createfiles*)* `[package]`

Definición en la línea 182 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

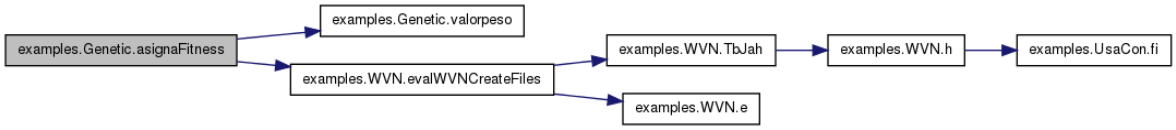
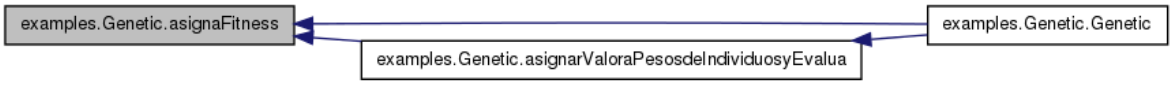


Gráfico de llamadas a esta función:

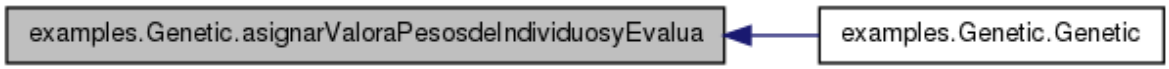


void examples.Genetic.asignarValoraPesosdeIndividuosyEvalua () `[package]`

Definición en la línea 212 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

Gráfico de llamadas a esta función:



void examples.Genetic.Cruzamiento (int Pcruz*)* *[package]*

Definición en la línea 117 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:

void examples.Genetic.iniciaRandomPob () *[package]*

Definición en la línea 92 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:

void examples.Genetic.Mutacion (int Pmut*)* *[package]*

Definición en la línea 105 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:

void examples.Genetic.Ordenamiento () *[package]*

Definición en la línea 218 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:

void examples.Genetic.setIGU (IGUWavenet iguwavenet*)*

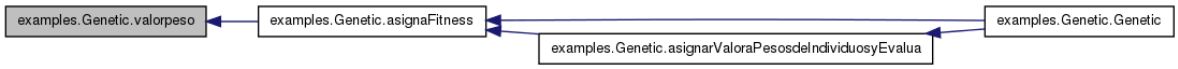
Definición en la línea 50 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:

double examples.Genetic.valorpeso (int indiv, *int* bitinicial, *double* escala*)* *[package]*

Definición en la línea 147 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:



Documentación de los datos miembro

int examples.Genetic.BitsIndiv *[package]*

Definición en la línea 37 del archivo GeneticoVGA.java.

double examples.Genetic.ESCALA *[package]*

Definición en la línea 46 del archivo GeneticoVGA.java.

double examples.Genetic.Fitness[] *[package]*

Definición en la línea 38 del archivo GeneticoVGA.java.

int examples.Genetic.GENERACIONES *[package]*

Definición en la línea 42 del archivo GeneticoVGA.java.

IGUWavenet examples.Genetic.localIGU *[package]*

Definición en la línea 48 del archivo GeneticoVGA.java.

int examples.Genetic.MAGNITUD *[package]*

Definición en la línea 45 del archivo GeneticoVGA.java.

boolean examples.Genetic.mostrarBits = false *[package]*

Definición en la línea 39 del archivo GeneticoVGA.java.

int examples.Genetic.NEURONAS *[package]*

Definición en la línea 40 del archivo GeneticoVGA.java.

int examples.Genetic.Pcruz *[package]*

Definición en la línea 44 del archivo GeneticoVGA.java.

int examples.Genetic.Pmut *[package]*

Definición en la línea 43 del archivo GeneticoVGA.java.

boolean [][] examples.Genetic.Poblacion `[package]`

Definición en la línea 34 del archivo GeneticoVGA.java.

int examples.Genetic.PRECSIGNOMAGNITUD `[package]`

Definición en la línea 41 del archivo GeneticoVGA.java.

int examples.Genetic.TamPob `[package]`

Definición en la línea 35 del archivo GeneticoVGA.java.

int examples.Genetic.TotalBitsPob `[package]`

Definición en la línea 36 del archivo GeneticoVGA.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **GeneticoVGA.java**

Referencia de la Clase examples.GeneticoVGA

Métodos públicos estáticos

- static void **main** (String[] args)

Descripción detallada

Definición en la línea 24 del archivo GeneticoVGA.java.

Documentación de las funciones miembro

static void examples.GeneticoVGA.main (String[] args) `[static]`

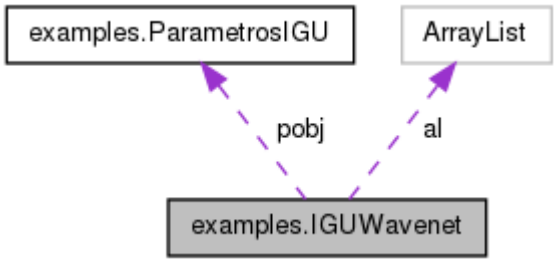
Definición en la línea 26 del archivo GeneticoVGA.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **GeneticoVGA.java**

Referencia de la Clase examples.IGUWavenet

Diagrama de colaboración para examples.IGUWavenet:



Métodos públicos

- **IGUWavenet** ()

Métodos públicos estáticos

- static void **main** (String args[])

Atributos públicos

- int **filterLength**
- double **alfa**
- int **NoNeuronas**
- int **Precision**
- double **beta**
- String **filename**

Atributos del 'package'

- ArrayList **al** = null
- **ParametrosIGU pobj** = null
- String **everything** = null

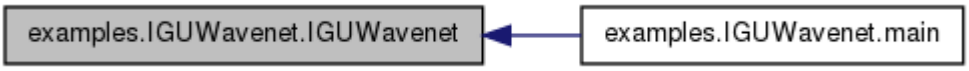
Documentación del constructor y destructor

examples.IGUWavenet.IGUWavenet ()

Creates new form **IGUWavenet**

Definición en la línea 55 del archivo IGUWavenet.java.

Gráfico de llamadas a esta función:



Documentación de las funciones miembro

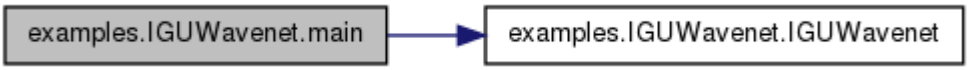
static void examples.IGUWavenet.main (String args*[])* `[static]`

Parámetros:

args	the command line arguments

Definición en la línea 528 del archivo IGUWavenet.java.

Gráfico de llamadas para esta función:



Documentación de los datos miembro

ArrayList examples.IGUWavenet.al = null `[package]`

Definición en la línea 394 del archivo IGUWavenet.java.

double examples.IGUWavenet.alfa

Definición en la línea 46 del archivo IGUWavenet.java.

double examples.IGUWavenet.beta

Definición en la línea 49 del archivo IGUWavenet.java.

String examples.IGUWavenet.everything = null `[package]`

Definición en la línea 524 del archivo IGUWavenet.java.

String examples.IGUWavenet.filename

Definición en la línea 50 del archivo IGUWavenet.java.

int examples.IGUWavenet.filterLength

Definición en la línea 45 del archivo IGUWavenet.java.

int examples.IGUWavenet.NoNeuronas

Definición en la línea 47 del archivo IGUWavenet.java.

ParametrosIGU examples.IGUWavenet.pobj = null `[package]`

Definición en la línea 523 del archivo IGUWavenet.java.

int examples.IGUWavenet.Precision

Definición en la línea 48 del archivo IGUWavenet.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **IGUWavenet.java**

Referencia de la Clase examples.ParametrosIGU

Atributos públicos

- String **filename**
- int **filtersize**
- int **neuronas**
- int **precisionK**
- double **alfa**
- double **beta**

Descripción detallada

Definición en la línea 13 del archivo ParametrosIGU.java.

Documentación de los datos miembro

double examples.ParametrosIGU.alfa

Definición en la línea 18 del archivo ParametrosIGU.java.

double examples.ParametrosIGU.beta

Definición en la línea 19 del archivo ParametrosIGU.java.

String examples.ParametrosIGU.filename

Definición en la línea 14 del archivo ParametrosIGU.java.

int examples.ParametrosIGU.filtersize

Definición en la línea 15 del archivo ParametrosIGU.java.

int examples.ParametrosIGU.neuronas

Definición en la línea 16 del archivo ParametrosIGU.java.

int examples.ParametrosIGU.precisionK

Definición en la línea 17 del archivo ParametrosIGU.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **ParametrosIGU.java**

Referencia de la Clase examples.TestTxt

Diagrama de herencias de examples.TestTxt

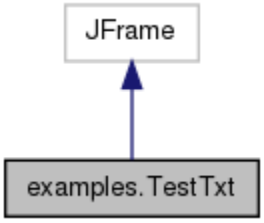
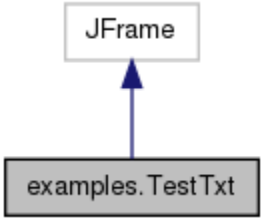


Diagrama de colaboración para examples.TestTxt:



Métodos públicos

- **TestTxt** ()
- **TestTxt** (String imagefile)

Métodos públicos estáticos

- static void **main** (String args[])

Atributos del 'package'

- String **filenameimg** = null

Descripción detallada

Definición en la línea 20 del archivo TestTxt.java.

Documentación del constructor y destructor

examples.TestTxt.TestTxt ()

Definición en la línea 28 del archivo TestTxt.java.

Gráfico de llamadas a esta función:

examples.TestTxt.TestTxt (String imagefile*)*

Definición en la línea 35 del archivo TestTxt.java.

Documentación de las funciones miembro

static void examples.TestTxt.main (String args*[])* `[static]`

Definición en la línea 23 del archivo TestTxt.java.

Gráfico de llamadas para esta función:

Documentación de los datos miembro

String examples.TestTxt.filenameimg = null `[package]`

Definición en la línea 22 del archivo TestTxt.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **TestTxt.java**

Referencia de la Clase examples.UsaCon

Métodos públicos

- double **fi** (double x)
- double **Dfi** (double x)
- **UsaCon** (int resolucion, double alfa4)
- **UsaCon** (int resolucion, double alfa6, double beta6)

Atributos del 'package'

- double[] **psi**
- double[] **phi**
- int **N**
- double **epsx**

Descripción detallada

Definición en la línea 710 del archivo GeneticoVGA.java.

Documentación del constructor y destructor

examples.UsaCon.UsaCon (int resolucion*, double* alfa4*)*

Definición en la línea 747 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

examples.UsaCon.UsaCon (int resolucion*, double* alfa6*, double* beta6*)*

Definición en la línea 833 del archivo GeneticoVGA.java.

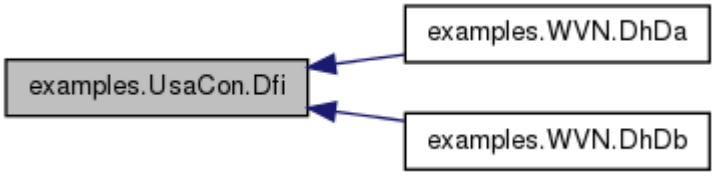
Gráfico de llamadas para esta función:

Documentación de las funciones miembro

double examples.UsaCon.Dfi (double x*)*

Definición en la línea 730 del archivo GeneticoVGA.java.

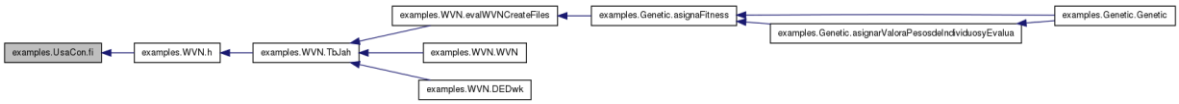
Gráfico de llamadas a esta función:



double examples.UsaCon.fi (double x)

Definición en la línea 717 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:



Documentación de los datos miembro

double examples.UsaCon.epsx `[package]`

Definición en la línea 715 del archivo GeneticoVGA.java.

int examples.UsaCon.N `[package]`

Definición en la línea 714 del archivo GeneticoVGA.java.

double [] examples.UsaCon.phi `[package]`

Definición en la línea 713 del archivo GeneticoVGA.java.

double [] examples.UsaCon.psi `[package]`

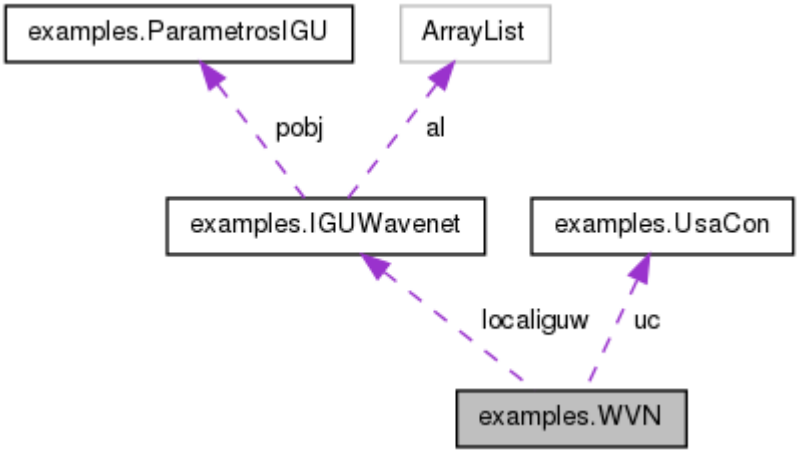
Definición en la línea 712 del archivo GeneticoVGA.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **GeneticoVGA.java**

Referencia de la Clase examples.WVN

Diagrama de colaboración para examples.WVN:



Métodos públicos

- **WVN** ()
- void **setIGUWavenet** (**IGUWavenet** iguwavenet)
- void **evalWVNCreateFiles** (int NEURONAS, double[] **w**, double[] **a**, double[] **b**, int indiv, double genalfa)
- **WVN** (**IGUWavenet** igu, int NEURONAS, double[] **w**, double[] **a**, double[] **b**, int indiv, double genalfa)

Atributos públicos

- **IGUWavenet localiguw**

Funciones del 'package'

- double **TbJah** (double bk, double ak, int x)
- double **h** (double x)
- double **DhDa** (double bk, double ak, double x)
- double **DhDb** (double bk, double ak, double x)
- double **DEDwk** (double bk, double ak, int x)
- double **e** (int t)
- double **u** (int t)

Atributos del 'package'

- int **N** = 50
- int **K**
- int **MAXITER** = 1
- double **error**
- int **iter**
- double **muw** = 0.1
- double **mua** = 0.1
- double **mub** = 0.1
- double **alfamom** = 0.1
- double **previous** = 0
- double **epsilon** = 0.0000001
- double **a** []
- double **b** []
- int **resolucion** = 13

- double **alfa4**
- double **beta6**
- **UsaCon uc** = null

Atributos Estáticos del 'package'

- static double[] **u**
- static double[] **y**
- static double[] **w**

Descripción detallada

Definición en la línea 241 del archivo GeneticoVGA.java.

Documentación del constructor y destructor

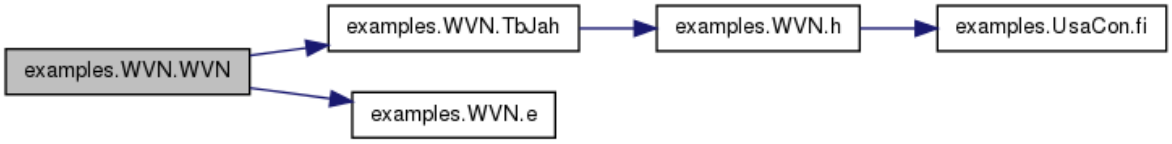
examples.WVN.WVN ()

Definición en la línea 271 del archivo GeneticoVGA.java.

examples.WVN.WVN (IGUWavenet igu, *int* NEURONAS, *double[]* w, *double[]* a, *double[]* b, *int* indiv, *double* genalfa*)*

Definición en la línea 576 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

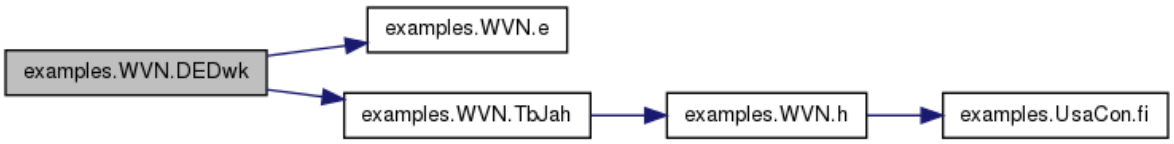


Documentación de las funciones miembro

double examples.WVN.DEDwk (double bk, *double* ak, *int* x*)* [package]

Definición en la línea 687 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:



double examples.WVN.DhDa (double bk, *double* ak, *double* x*)* [package]

Definición en la línea 676 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

double examples.WVN.DhDb (double bk*, double* ak*, double* x*)* [package]

Definición en la línea 682 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

double examples.WVN.e (int t*)* [package]

Definición en la línea 699 del archivo GeneticoVGA.java.

Gráfico de llamadas a esta función:

void examples.WVN.evalWVNCreateFiles (int NEURONAS*, double[]* w*, double[]* a*, double[]* b*, int* indiv*, double* genalfa*)*

Definición en la línea 281 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

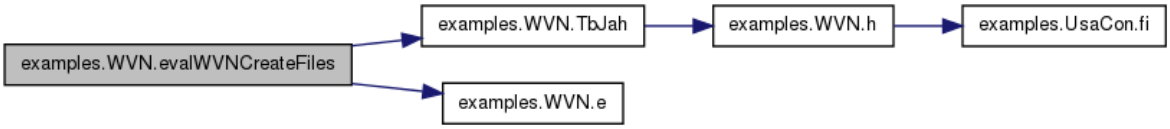
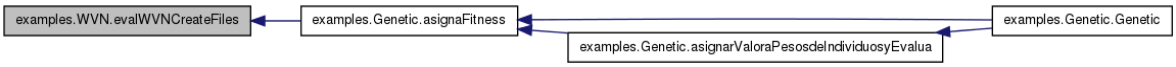


Gráfico de llamadas a esta función:

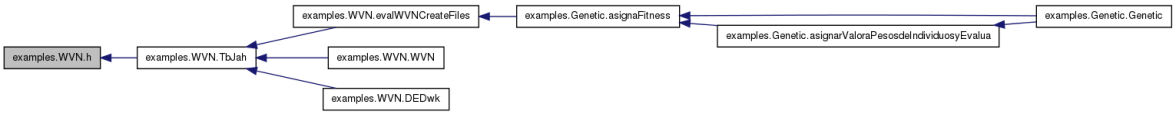


double examples.WVN.h (double x*)* [package]

Definición en la línea 669 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

Gráfico de llamadas a esta función:



void examples.WVN.setIGUWavenet (IGUWavenet iguwavenet*)*

Definición en la línea 275 del archivo GeneticoVGA.java.

double examples.WVN.TbJah (double bk*, double* ak*, int* x*)* [package]

Definición en la línea 663 del archivo GeneticoVGA.java.

Gráfico de llamadas para esta función:

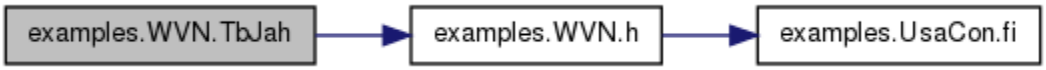
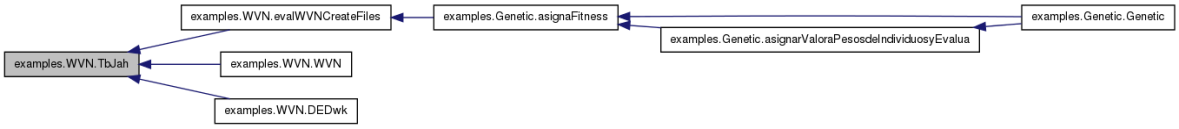


Gráfico de llamadas a esta función:



double examples.WVN.u (int t*)* [package]

Definición en la línea 703 del archivo GeneticoVGA.java.

Documentación de los datos miembro

double examples.WVN.a[] [package]

Definición en la línea 256 del archivo GeneticoVGA.java.

double examples.WVN.alfa4 [package]

Definición en la línea 262 del archivo GeneticoVGA.java.

double examples.WVN.alfamom = 0.1 [package]

Definición en la línea 252 del archivo GeneticoVGA.java.

double examples.WVN.b[] [package]

Definición en la línea 257 del archivo GeneticoVGA.java.

double examples.WVN.beta6 `[package]`

Definición en la línea 265 del archivo GeneticoVGA.java.

double examples.WVN.epsilon = 0.0000001 `[package]`

Definición en la línea 254 del archivo GeneticoVGA.java.

double examples.WVN.error `[package]`

Definición en la línea 247 del archivo GeneticoVGA.java.

int examples.WVN.iter `[package]`

Definición en la línea 248 del archivo GeneticoVGA.java.

int examples.WVN.K `[package]`

Definición en la línea 245 del archivo GeneticoVGA.java.

IGUWavenet examples.WVN.localiguw

Definición en la línea 269 del archivo GeneticoVGA.java.

int examples.WVN.MAXITER = 1 `[package]`

Definición en la línea 246 del archivo GeneticoVGA.java.

double examples.WVN.mua = 0.1 `[package]`

Definición en la línea 250 del archivo GeneticoVGA.java.

double examples.WVN.mub = 0.1 `[package]`

Definición en la línea 251 del archivo GeneticoVGA.java.

double examples.WVN.muw = 0.1 `[package]`

Definición en la línea 249 del archivo GeneticoVGA.java.

int examples.WVN.N = 50 `[package]`

Definición en la línea 244 del archivo GeneticoVGA.java.

double examples.WVN.previous = 0 `[package]`

Definición en la línea 253 del archivo GeneticoVGA.java.

int examples.WVN.resolucion = 13 `[package]`

Definición en la línea 261 del archivo GeneticoVGA.java.

double [] examples.WVN.u `[static, package]`

Definición en la línea 258 del archivo GeneticoVGA.java.

UsaCon examples.WVN.uc = null `[package]`

Definición en la línea 266 del archivo GeneticoVGA.java.

double [] examples.WVN.w `[static, package]`

Definición en la línea 260 del archivo GeneticoVGA.java.

double [] examples.WVN.y `[static, package]`

Definición en la línea 259 del archivo GeneticoVGA.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **GeneticoVGA.java**

Documentación de archivos

Referencia del Archivo GeneticoVGA.java

Clases

- class **examples.GeneticoVGA**
- class **examples.Genetic**
- class **examples.WVN**
- class **examples.UsaCon**
- class **examples.Convolucionador**

Paquetes

- package **examples**

Referencia del Archivo IGUWavenet.java

Clases

- class **examples.IGUWavenet**

Paquetes

- package **examples**

Referencia del Archivo ParametrosIGU.java

Clases

- class **examples.ParametrosIGU**

Paquetes

- package **examples**

Referencia del Archivo TestTxt.java

Clases

- class **examples.TestTxt**

Paquetes

- package **examples**

Codigo Fuente

I. GenteticoVGA.Java

```
00001 /*
00002  * Estudio experimental de la aproximación de funciones con redes
neuronales wavenets
00003  * Se muestra este código sin permiso de modificación, solo de lectura.
00004    Cualquier asunto relacionado con el tema favor de contactar a:
00005    Dr. Oscar Herrera Alcántara oha@correo.azc.uam.mx
00006    Ing. José Luis Ramírez Rojano luis861011@gmail.com
00007    Para ver la posibilidad de otorgar dichos permisos para mejoras y
nuevos proyectos.
00008  */
00009 package examples;
00010
00011 import java.io.BufferedOutputStream;
00012 import java.io.BufferedReader;
00013 import java.io.BufferedWriter;
00014 import java.io.DataOutputStream;
00015 import java.io.File;
00016 import java.io.FileOutputStream;
00017 import java.io.IOException;
00018 import java.io.InputStreamReader;
00019 import java.io.OutputStreamWriter;
00020 import java.io.PrintStream;
00021 import java.io.PrintWriter;
00022 import java.util.Random;
00023
00024 public class GeneticoVGA {
00025
00026     public static void main(String[] args) {
00027         Random rndNumbers = new Random(0);
00028         Genetic mlpgen = new Genetic();
00029     }
00030 }
00031
00032 class Genetic {
00033
00034     boolean[][] Poblacion;
00035     int TamPob;
00036     int TotalBitsPob;
00037     int BitsIndiv;
00038     double Fitness[];
00039     boolean mostrarBits = false;//true;
```

```
00040     int NEURONAS;
00041     int PRECSIGNOMAGNITUD;
00042     int GENERACIONES;
00043     int Pmut;
00044     int Pcruz;
00045     int MAGNITUD;
00046     double ESCALA;
00047     //double ESCALAWAV;
00048     IGUWavenet localIGU;
00049
00050     public void setIGU(IGUWavenet iguwavenet) {
00051         this.localIGU = iguwavenet;
00052         //System.out.println("Muestra leidas = " + localIGU.al.size());
00053     }
00054
00055     public Genetic() {
00056
00057     }
00058
00059     public Genetic(IGUWavenet iguw) {
00060         setIGU(iguw);
00061         NEURONAS = localIGU.NoNeuronas;
00062
00063         PRECSIGNOMAGNITUD = 29;
00064         MAGNITUD = PRECSIGNOMAGNITUD - 1;
00065         //NEURONAS=//10;
00066         ESCALA = 1024 * 8;
00067         //ESCALAWAV=5*Math.PI/12;
00068         //ESCALAWAV = Math.PI / 4;
00069         GENERACIONES = 200;
00070         TamPob = 40;
00071         Pcruz = 97;
00072         Pmut = 3;
00073         BitsIndiv = (PRECSIGNOMAGNITUD * 3) *
NEURONAS;//+PRECSIGNOMAGNITUD;  //abw*15 bits cada wavelet
00074         //   bit de signo *(valor/MAXVALOR)*ESCALA
00075         TotalBitsPob = TamPob * BitsIndiv;
00076         Poblacion = new boolean[2 * TamPob][BitsIndiv];  //10 individuos,
135 bits cada individuo
00077         Fitness = new double[2 * TamPob];
00078         iniciaRandomPob();
00079         for (int c = 0; c < GENERACIONES; c++) { //Generaciones
00080             Cruzamiento(Pcruz);
00081             Mutacion(Pmut);
00082             asignarValoraPesosdeIndividuosyEvalua();
```

```
00083            Ordenamiento();
00084            asignaFitness(0, false);
00085            System.out.print("GENERACION= " + c);
00086            System.out.println("\tFINDIV0= " + Fitness[0]);
00087        }
00088        asignaFitness(0, true);
00089        System.out.println("EL MEJOR FITNESS  " + Fitness[0]);
00090    } //Genetico
00091
00092    void iniciaRandomPob() {
00093        for (int i = 0; i < TamPob; i++) {
00094            for (int j = 0; j < BitsIndiv; j++) {
00095                int tmp = (int) (Math.random() * 100);
00096                if (tmp < 70) {
00097                    Poblacion[i][j] = false;
00098                } else {
00099                    Poblacion[i][j] = true;
00100                }
00101            }
00102        }
00103    }
00104
00105    void Mutacion(int Pmut) {
00106        for (int i = TamPob; i < 2 * TamPob; i++) {
00107            for (int j = 0; j < BitsIndiv; j++) {
00108                int f = (int) (Math.random() * 100);
00109                if (f < Pmut) { //Con probabilidad 5%
00110                    Poblacion[i][j] = !Poblacion[i][j];  //cambio de valor
0-->1, 1-->0
00111                    //solo mutar los descendientes
00112                }
00113            }
00114        }
00115    }
00116
00117    void Cruzamiento(int Pcruz) {
00118        for (int i = 0; i < TamPob; i++) {
00119            for (int j = 0; j < BitsIndiv; j++) {
00120                Poblacion[i + TamPob][j] = Poblacion[i][j];
00121            }
00122        }
00123
00124        for (int i = 0; i < TamPob / 2; i++) {
00125            int puntocruce1;
00126            puntocruce1 = (int) (Math.random() * BitsIndiv);
```

```
00127             int bitactual = puntocruce1;
00128             int individuoA, individuoB;
00129             individuoA = i + TamPob;
00130             individuoB = 2 * TamPob - 1 - i;
00131             for (int j = 0; j < BitsIndiv / 2; j++) {
00132                 int c = (int) (Math.random() * 100);
00133                 if (c < Pcruz) { //con probabilidad 95%
00134                     boolean tmp;
00135                     tmp = Poblacion[individuoA][bitactual];
00136                     Poblacion[individuoA][bitactual] =
Poblacion[individuoB][bitactual];
00137                     Poblacion[individuoB][bitactual] = tmp;
00138                     bitactual++;
00139                     bitactual = bitactual % BitsIndiv;
00140
00141                 }
00142             }
00143             //System.out.println("------------------");
00144         }
00145     }
00146
00147     double valorpeso(int indiv, int bitinicial, double escala) {
00148         double valor = 0;
00149         if (mostrarBits) {
00150             for (int y = bitinicial; y < bitinicial + PRECSIGNOMAGNITUD;
y++) {
00151                 if (Poblacion[indiv][y]) {
00152                     System.out.print("1");
00153                 } else {
00154                     System.out.print("0");
00155                 }
00156             }
00157             System.out.print(" ");
00158         }
00159         if (mostrarBits) {
00160             System.out.println(" ");
00161         }
00162         int bitactual = bitinicial;     //gen actual
00163         for (int i = 1; i <= MAGNITUD; i++) {
00164             bitactual++;
00165             if (Poblacion[indiv][bitactual]) {
00166                 valor += 1 << ((i - 1));
00167                 //valor += 1<<(MAGNITUD-1-(i-1));
00168             }
00169         }
```

```
00170         valor /= (1 << MAGNITUD);
00171         this.ESCALA = escala;
00172         valor *= ESCALA;
00173
00174
00175         if (Poblacion[indiv][bitinicial]) {
00176             valor = -valor;   //signo
00177         }
00178
00179         return valor;
00180     }
00181
00182     void asignaFitness(int indiv, boolean createfiles) {
00183
00184         double eav;
00185         double[] a = new double[NEURONAS];
00186         double[] b = new double[NEURONAS];
00187         double[] w = new double[NEURONAS];
00188
00189         int cur = 0;
00190         for (int n = 0; n < NEURONAS; n++) {
00191             a[n] = valorpeso(indiv, cur, ESCALA);
00192             cur += PRECSIGNOMAGNITUD;
00193             b[n] = valorpeso(indiv, cur, ESCALA);
00194             cur += PRECSIGNOMAGNITUD;
00195             w[n] = valorpeso(indiv, cur, ESCALA);
00196             cur += PRECSIGNOMAGNITUD;
00197         }
00198         double genalfa = Math.PI / 4.0;
00199
00200         WVN wvn = new WVN(localIGU, NEURONAS, w, a, b, indiv, genalfa);
00201
00202         if (createfiles) {
00203             wvn.evalWVNCreateFiles(NEURONAS, w, a, b, indiv, genalfa);
00204             System.out.println("FINAL BEST GENALFA= " + genalfa);
00205             System.out.println("FINAL BEST GENBETA= " +
this.localIGU.beta);
00206         }
00207         eav = wvn.error;
00208         Fitness[indiv] = eav;
00209
00210     }
00211
00212     void asignarValoraPesosdeIndividuosyEvalua() {
00213         for (int i = 0; i < 2 * TamPob; i++) {
```

```
00214             asignaFitness(i, false);
00215         }
00216     }
00217
00218     void Ordenamiento() {
00219         for (int i = 0; i < 2 * TamPob - 1; i++) {
00220             for (int j = i + 1; j < 2 * TamPob; j++) {
00221                 if (Fitness[j] < Fitness[i]) {
00222                     double tmp;
00223                     tmp = Fitness[j];
00224                     Fitness[j] = Fitness[i];
00225                     Fitness[i] = tmp;
00226                     for (int k = 0; k < BitsIndiv; k++) { //swap todos los
bits del indibiduo j <--> i
00227                         boolean swp = Poblacion[i][k];
00228                         Poblacion[i][k] = Poblacion[j][k];
00229                         Poblacion[j][k] = swp;
00230                     }
00231                 }
00232             }
00233         }
00234
00235
00236     }
00237
00238 }
00239
00241 class WVN {
00243
00244     int N = 50; //Number of samples of the discrete signal
00245     int K; //The number of neurons
00246     int MAXITER = 1;//1000;//25000;
00247     double error;
00248     int iter;
00249     double muw = 0.1;
00250     double mua = 0.1; //Mejor Solucion 0.009... con muw=1.1, mua=2, mub=2
00251     double mub = 0.1;
00252     double alfamom = 0.1;
00253     double previous = 0;
00254     double epsilon = 0.0000001; //El maximo error requerido como criterio
de paro
00256     double a[];
00257     double b[];
00258     static double[] u;
00259     static double[] y;
```

```
00260     static double[] w;
00261     int resolucion = 13;
00262     double alfa4;
00263     // = Math.PI / 4.0;
00264     //double alfa4=Math.PI/10;
00265     double beta6;
00266     UsaCon uc = null;
00267     //int resolucion=3;
00268     //int alfa4=Math.PI/4.0;
00269     public IGUWavenet localiguw;
00270
00271     public WVN() {
00272
00273     }
00274
00275     public void setIGUWavenet(IGUWavenet iguwavenet) {
00276         localiguw = iguwavenet;
00277         N = localiguw.al.size() - 1;
00278         System.out.print("Se leyeron " + N + "muestras del ArrayList.");
00279     }
00280
00281     public void evalWVNCreateFiles(int NEURONAS, double[] w, double[] a,
double[] b, int indiv, double genalfa) {
00282         this.K = NEURONAS;
00283         this.alfa4 = genalfa;
00284         this.beta6 = this.localiguw.beta;
00285         if (this.localiguw.filterLength == 4) {
00286             System.out.println("Se ha usado un filtro de longitud 4.");
00287             uc = new UsaCon(resolucion, alfa4);
00288
00289             u = new double[N];
00290             y = new double[N];
00291
00292             for (int i = 0; i < N; i++) {
00293
00294                 u[i] =
Double.parseDouble(this.localiguw.al.get(i).toString());
00295                 y[i] = 0;
00296             }
00297
00298             this.w = w;
00299             this.a = a;
00300             this.b = b;
00301
00302             File f;
```

```
00303             FileOutputStream fos;
00304             BufferedOutputStream bos;
00305             DataOutputStream dos;
00306             PrintStream out;
00307             f = new File("./src/examples/ItervsError.txt");
00308             try {
00309
00310                 fos = new FileOutputStream(f);
00311                 //dos = new DataOutputStream(fos);
00312                 bos = new BufferedOutputStream(fos);
00313                 out = new PrintStream(bos);
00314                 iter = 0;
00315
00316                 do {
00317                     error = 0.0;
00318                     for (int n = 0; n < N; n++) { //Por cada muestra,
variable temporal
00319                         double sumwkhtkn = 0;
00320                         for (int k = 0; k < K; k++) { //El numero de
neuronas
00321                             double tjh = TbJah(b[k], a[k], n);
00322                             sumwkhtkn += w[k] * tjh;
00323                         }
00324                         y[n] = u[n] * sumwkhtkn;
00325                         error += Math.pow(e(n), 2);
00326
00327
00328                     }//n
00329                     iter++;
00330                     error = Math.sqrt(error / N);
00331
00332                     out.println("#indiv=" + indiv + " waveiter= " + iter +
" error=" + error);
00333                     System.out.println("#indiv=" + indiv + " waveiter= " +
iter + " error= " + error);
00334
00335                 } while (error > epsilon && iter < MAXITER);
00336                 bos.flush();
00337                 out.flush();
00338                 fos.close();
00339             } catch (Exception e) {
00340                 System.out.println(e);
00341             }
00343
00345             try {
00346                 File ff;
```

```
00347                 FileOutputStream ffos;
00348                 BufferedOutputStream fbos;
00349                 DataOutputStream fdos;
00350                 PrintStream fout;
00351                 ff = new File("./src/examples/Params_ayb.txt");
00352                 ffos = new FileOutputStream(ff);
00353                 //dos = new DataOutputStream(fos);
00354                 fbos = new BufferedOutputStream(ffos);
00355                 fout = new PrintStream(fbos);
00356
00357                 fout.println("##############################");
00358
00359                 fout.println("Se ha usado un filtro de longitud 4.");
00360                 for (int i = 0; i < K; i++) {
00361
00362                     fout.println("a[" + i + "]=" + a[i] + "\tb[" + i +
"]=" + b[i]);
00363
00364                 }
00365                 fbos.flush();
00366                 fout.flush();
00367                 ffos.close();
00368             } catch (Exception e) {
00369                 System.out.println(e);
00370             }
00371             File xf;
00372             FileOutputStream xfos;
00373             BufferedOutputStream xbos;
00374             DataOutputStream xdos;
00375             PrintStream xout;
00376             try {
00377                 xf = new File("./src/examples/aproxima.txt");
00378                 xfos = new FileOutputStream(xf);
00379                 //dos = new DataOutputStream(fos);
00380                 xbos = new BufferedOutputStream(xfos);
00381                 xout = new PrintStream(xbos);
00382
00383                 for (int i = 0; i < N; i++) {
00384                     //System.
00385                     if (Math.abs(u[i] - y[i]) > 10) {
00386                         xout.println(u[i] + "\t" + 10);
00387                     } else {
00388                         xout.println(u[i] + "\t" + y[i]);
00389                     }
00390                 }
```

```
00391
00392                 xbos.flush();
00393                 xout.flush();
00394                 xfos.close();
00395             } catch (Exception ioex) {
00396                 System.out.println(ioex);
00397             } finally {
00398
00399             }
00400
00401             try {
00402                 String somecommand = "sh   /home/siul/netbeans-
7.4/IGU/src/examples/gnu.plot";
00403                 Process proc = Runtime.getRuntime().exec(somecommand);
00404                 BufferedWriter writer = new BufferedWriter(
00405                         new OutputStreamWriter(proc.getOutputStream()));
00406
00407                 System.out.println("\n\twaiting...");
00408                 proc.waitFor();
00409                 BufferedReader in = new BufferedReader(
00410                         new InputStreamReader(proc.getInputStream()));
00411                 BufferedReader err = new BufferedReader(
00412                         new InputStreamReader(proc.getErrorStream()));
00413                 String line = null;
00414                 while ((line = in.readLine()) != null) {
00415                     System.out.println(line);
00416                 }
00417                 while ((line = err.readLine()) != null) {
00418                     System.out.println(line);
00419                 }
00420
00421                 System.out.println("\n\tdone...");
00422             } catch (IOException e) {
00423                 e.printStackTrace();
00424             } catch (Exception e) {
00425                 System.out.print(e);
00426             }
00427         } else {//fin filterlength4 inicia filterlength6
00428             System.out.println("Se ha usado un filtro de longitud 6.");
00429             uc = new UsaCon(resolucion, alfa4, beta6);
00430
00431             u = new double[N];
00432             y = new double[N];
00433
00434             for (int i = 0; i < N; i++) {
```

```
00435
00436                 u[i] =
Double.parseDouble(this.localiguw.al.get(i).toString());
00437                 y[i] = 0;
00438             }
00439
00440             this.w = w;
00441             this.a = a;
00442             this.b = b;
00443
00444             File f;
00445             FileOutputStream fos;
00446             BufferedOutputStream bos;
00447             DataOutputStream dos;
00448             PrintStream out;
00449             f = new File("./src/examples/ItervsError.txt");
00450             try {
00451
00452                 fos = new FileOutputStream(f);
00453                 bos = new BufferedOutputStream(fos);
00454                 out = new PrintStream(bos);
00455                 iter = 0;
00456
00457                 do {
00458                     error = 0.0;
00459                     for (int n = 0; n < N; n++) { //Por cada muestra,
variable temporal
00460                         double sumwkhtkn = 0;
00461                         for (int k = 0; k < K; k++) { //El numero de
neuronas
00462                             double tjh = TbJah(b[k], a[k], n);
00463                             sumwkhtkn += w[k] * tjh;
00464                         }
00465                         y[n] = u[n] * sumwkhtkn;
00466                         error += Math.pow(e(n), 2);
00467
00468                     }//n
00469                     iter++;
00470                     error = Math.sqrt(error / N);
00471
00472                     out.println("#indiv=" + indiv + " waveiter= " + iter +
" error=" + error);
00473                     System.out.println("#indiv=" + indiv + " waveiter= " +
iter + " error= " + error);
00474
00475                 } while (error > epsilon && iter < MAXITER);
```

```
00476                 bos.flush();
00477                 out.flush();
00478                 fos.close();
00479             } catch (Exception e) {
00480                 System.out.println(e);
00481             }
00483
00485             try {
00486                 File ff;
00487                 FileOutputStream ffos;
00488                 BufferedOutputStream fbos;
00489                 DataOutputStream fdos;
00490                 PrintStream fout;
00491                 ff = new File("./src/examples/Params_ayb.txt");
00492                 ffos = new FileOutputStream(ff);
00493                 fbos = new BufferedOutputStream(ffos);
00494                 fout = new PrintStream(fbos);
00495
00496
00497                 fout.println("##############################");
00498
00499                 fout.println("# muw= " + muw);
00500
00501                 fout.println("# mua= " + mua);
00502
00503                 fout.println("# mub= " + mub);
00504
00505                 fout.println("# previous= " + previous);
00506                 fout.println("Se ha usado un filtro de longitud 6.");
00507                 for (int i = 0; i < K; i++) {
00508
00509                     fout.println("a[" + i + "]=" + a[i] + "\tb[" + i +
"]=" + b[i]);
00510
00511                 }
00512                 fbos.flush();
00513                 fout.flush();
00514                 ffos.close();
00515             } catch (Exception e) {
00516                 System.out.println(e);
00517             }
00518             File xf;
00519             FileOutputStream xfos;
00520             BufferedOutputStream xbos;
00521             DataOutputStream xdos;
```

```
00522            PrintStream xout;
00523            try {
00524                xf = new File("./src/examples/aproxima.txt");
00525                xfos = new FileOutputStream(xf);
00526                xbos = new BufferedOutputStream(xfos);
00527                xout = new PrintStream(xbos);
00528
00529                for (int i = 0; i < N; i++) {
00530                    //System.
00531                    if (Math.abs(u[i] - y[i]) > 10) {
00532                        xout.println(u[i] + "\t" + 10);
00533                    } else {
00534                        xout.println(u[i] + "\t" + y[i]);
00535                    }
00536                }
00537
00538                xbos.flush();
00539                xout.flush();
00540                xfos.close();
00541            } catch (Exception ioex) {
00542                System.out.println(ioex);
00543            } finally {
00544
00545            }
00546
00547            try {
00548                String somecommand = "sh /home/siul/netbeans-
7.4/IGU/src/examples/gnu.plot";
00549                Process proc = Runtime.getRuntime().exec(somecommand);
00550                BufferedWriter writer = new BufferedWriter(
00551                        new OutputStreamWriter(proc.getOutputStream()));
00552
00553                System.out.println("\n\twaiting...");
00554                proc.waitFor();
00555                BufferedReader in = new BufferedReader(
00556                        new InputStreamReader(proc.getInputStream()));
00557                BufferedReader err = new BufferedReader(
00558                        new InputStreamReader(proc.getErrorStream()));
00559                String line = null;
00560                while ((line = in.readLine()) != null) {
00561                    System.out.println(line);
00562                }
00563                while ((line = err.readLine()) != null) {
00564                    System.out.println(line);
00565                }
```

```
00566
00567                 System.out.println("\n\tdone...");
00568             } catch (IOException e) {
00569                 e.printStackTrace();
00570             } catch (Exception e) {
00571                 System.out.print(e);
00572             }
00573         }//filterlength6
00574     }
00575
00576     public WVN(IGUWavenet igu, int NEURONAS, double[] w, double[] a,
double[] b, int indiv, double genalfa) {
00577         this.localiguw = igu;
00578
00579         this.N = this.localiguw.al.size() - 1;
00580 //        System.out.println("Se usan N muestras de la señal discreta " +
N);
00581
00582         this.K = NEURONAS;
00583         this.alfa4 = genalfa;
00584
00585         if (this.localiguw.filterLength == 4) {
00586
00587             uc = new UsaCon(resolucion, alfa4);
00588
00589             u = new double[N];
00590             y = new double[N];
00591
00592             for (int i = 0; i < N; i++) {
00593                 //Leer desde al ArrayList
00594                 u[i] =
Double.parseDouble(this.localiguw.al.get(i).toString());
00595                 //u[i]= Math.cos(2*Math.PI*i/N);
00596                 //u[i]= Math.sin((double)i/N);
00597                 y[i] = 0;
00598             }
00599
00600             this.w = w;
00601             this.a = a;
00602             this.b = b;
00603
00604             iter = 0;
00605
00606             do {
00607                 error = 0.0;
```

```
00608                 for (int n = 0; n < N; n++) { //Por cada muestra, variable
temporal
00609                     double sumwkhtkn = 0;
00610                     for (int k = 0; k < K; k++) { //El numero de neuronas
00611                         double tjh = TbJah(b[k], a[k], n);
00612                         sumwkhtkn += w[k] * tjh;
00613                     }
00614                     y[n] = u[n] * sumwkhtkn;
00615                     error += Math.pow(e(n), 2);
00616
00617                 }//n
00618                 iter++;
00619                 error = Math.sqrt(error / N);
00620
00621             } while (error > epsilon && iter < MAXITER);
00622         } else {//filterlength4
00623             double beta6 = this.localiguw.beta;
00624             uc = new UsaCon(resolucion, alfa4, beta6);
00625
00626             u = new double[N];
00627             y = new double[N];
00628
00629             for (int i = 0; i < N; i++) {
00630                 //Leer desde al ArrayList
00631                 u[i] =
Double.parseDouble(this.localiguw.al.get(i).toString());
00632
00633                 y[i] = 0;
00634             }
00635
00636             this.w = w;
00637             this.a = a;
00638             this.b = b;
00639
00640             iter = 0;
00641
00642             do {
00643                 error = 0.0;
00644                 for (int n = 0; n < N; n++) { //Por cada muestra, variable
temporal
00645                     double sumwkhtkn = 0;
00646                     for (int k = 0; k < K; k++) { //El numero de neuronas
00647                         double tjh = TbJah(b[k], a[k], n);
00648                         sumwkhtkn += w[k] * tjh;
00649                     }
```

```
00650                     y[n] = u[n] * sumwkhtkn;
00651                     error += Math.pow(e(n), 2);
00652
00653                 }//n
00654                 iter++;
00655                 error = Math.sqrt(error / N);
00656
00657             } while (error > epsilon && iter < MAXITER);
00658         }//filterlength6
00660
00661     }//WVNconstructor
00662
00663     double TbJah(double bk, double ak, int x) {
00664         double TbJax;
00665         TbJax = ((double) x - bk) / ak;
00666         return h(TbJax);
00667     }
00668
00669     double h(double x) {
00670         double hx;
00671         hx = uc.fi(x);
00672
00673         return hx;
00674     }
00675
00676     double DhDa(double bk, double ak, double x) {
00677
00678         return -(x - bk) / (ak * ak) * uc.Dfi(x);
00679
00680     }
00681
00682     double DhDb(double bk, double ak, double x) {
00683
00684         return -1 / ak * uc.Dfi(x);
00685     }
00686
00687     double DEDwk(double bk, double ak, int x) {
00688         double tmp = 0;
00689         int T = u.length;
00690         for (int t = 0; t < T; t++) {
00691             tmp += e(t) * u(t) * TbJah(bk, ak, x);
00692         }
00693
00694         return -tmp;
00695     }
```

```
00696
00697
00698
00699     double e(int t) {
00700         return u[t] - y[t];
00701     }
00702
00703     double u(int t) {
00704         //u[0]=0;
00705         return u[t];
00706     }
00707 }
00708
00710 class UsaCon {
00711
00712     double[] psi;
00713     double[] phi;
00714     int N;
00715     double epsx;
00716
00717     public double fi(double x) {
00718         if (x < 0) {
00719             return 0;
00720         }
00721         if (x >= N - 1) {
00722             return 0;
00723         }
00724         double nx = x * (psi.length / (N - 1));
00725         int n = (int) nx;
00726
00727         return 1 * psi[n];
00728     }
00729
00730     public double Dfi(double x) {
00731         double deri;
00732         double nx = x * (psi.length / (N - 1));
00733         int n = (int) nx;
00734         int nmas1 = n + 1;
00735         //System.out.println("n= "+ n + "\t nmas1= "+ nmas1);
00736         if (n >= psi.length || nmas1 >= psi.length) {
00737             deri = 0;
00738         } else if (n <= 0 || nmas1 <= 0) {
00739             deri = 0;
00740         } else {
00741             deri = (psi[nmas1] - psi[n]) / epsx;
```

```
00742         }
00743
00744         return deri;
00745     }
00746
00747     public UsaCon(int resolucion, double alfa4) {
00748         N = 4;
00749         epsx = (double) (N - 1) / (double) (N * (1 << resolucion) - 1);
00750
00751         double[] h0 = new double[N];
00752         double[] h1 = new double[N];
00753
00754         h0[0] = 0.25 + (1 / (2 * Math.sqrt(2.0))) * Math.cos(alfa4); //
0.6830127;
00755         h0[1] = 0.25 + (1 / (2 * Math.sqrt(2.0))) * Math.sin(alfa4);
//1.1830127;
00756         h0[2] = 0.25 - (1 / (2 * Math.sqrt(2.0))) * Math.cos(alfa4);
//0.3169873;
00757         h0[3] = 0.25 - (1 / (2 * Math.sqrt(2.0))) * Math.sin(alfa4);  //-
0.1830127;
00758
00759
00760         for (int i = 0; i < N; i++) {
00761
00762             h0[i] *= 2.0;
00763
00764         }
00765
00766         for (int i = 0; i < N; i++) {
00767             if (i % 2 == 1) {
00768                 h1[i] = -h0[i];
00769             } else {
00770                 h1[i] = h0[i];
00771             }
00772         }
00773
00774         phi = new double[N];
00775         psi = new double[N];
00776
00777         for (int i = 0; i < N; i++) {
00778             phi[i] = h0[i];
00779             psi[i] = h1[i];
00780         }
00781         for (int j = 1; j < resolucion; j++) {
00782             int top = N * (1 << j) - 1;
00783             int le = top + 1;
```

```
00784
00785             double[] temp = new double[le / 2];
00786             double[] temp2 = new double[le / 2];
00787             for (int y = 0; y < le / 2; y++) {
00788                 temp[y] = phi[y];
00789                 temp2[y] = psi[y];
00790             }
00791             phi = null;
00792             psi = null;
00793             phi = new double[le];
00794             psi = new double[le];
00795             for (int k = 0; k < le; k++) {
00796                 phi[k] = 0;
00797                 psi[k] = 0;
00798             }
00799             int a = 0;
00800             for (int k = 0; k < top; k++) {
00801                 if (k % 2 == 0) {
00802                     phi[k] = temp[a];
00803                     psi[k] = temp2[a];
00804                     a++;
00805                 } else {
00806                     phi[k] = 0;
00807                     psi[k] = 0;
00808                 }
00809             }
00810
00811             temp = null;
00812             temp2 = null;
00813             //convolve
00814             int lon = top + N - 1;
00815             double[] out = new double[lon];
00816
00817
00818             Convolucionador.convolvex(out, phi, top, h0, N);
00819             for (int k = 0; k < le; k++) {
00820                 phi[k] = out[k];
00821             }
00822
00823             Convolucionador.convolvex(out, psi, top, h0, N);
00824             for (int k = 0; k < le; k++) {
00825                 psi[k] = out[k];
00826             }
00827             out = null;
00828         }
```

```
00829
00830     }
00831
00833    public UsaCon(int resolucion, double alfa6, double beta6) {
00834         N = 6;
00835         epsx = (double) (N - 1) / (double) (N * (1 << resolucion) - 1);
00836
00837         double[] h0 = new double[N];
00838         double[] h1 = new double[N];
00839         double p = 0.5*Math.sqrt(1+ Math.sin(alfa6+Math.PI/4));
00840         h0[0] = 1/8.0 + (1 / (4 * Math.sqrt(2.0))) * Math.cos(alfa6) +
p/2*Math.cos(beta6); // 0.6830127;
00841         h0[1] = 1/8.0 + (1 / (4 * Math.sqrt(2.0))) * Math.sin(alfa6) +
p/2*Math.sin(beta6); //1.1830127;
00842         h0[2] = 1/4.0 - (1 / (2 * Math.sqrt(2.0))) * Math.cos(alfa6);
//0.3169873;
00843         h0[3] = 1/4.0 - (1 / (2 * Math.sqrt(2.0))) * Math.sin(alfa6);  //-
0.1830127;
00844         h0[4] = 1/8.0 + (1 / (4 * Math.sqrt(2.0))) * Math.cos(alfa6) -
p/2*Math.cos(beta6); // 0.6830127;
00845         h0[5] = 1/8.0 + (1 / (4 * Math.sqrt(2.0))) * Math.sin(alfa6) -
p/2*Math.sin(beta6); //1.1830127;
00846
00847         for (int i = 0; i < N; i++) {
00848
00849             h0[i] *= 2.0;
00850
00851         }
00852
00853         for (int i = 0; i < N; i++) {
00854             if (i % 2 == 1) {
00855                 h1[i] = -h0[i];
00856             } else {
00857                 h1[i] = h0[i];
00858             }
00859         }
00860
00861         phi = new double[N];
00862         psi = new double[N];
00863
00864         for (int i = 0; i < N; i++) {
00865             phi[i] = h0[i];
00866             psi[i] = h1[i];
00867         }
00868         for (int j = 1; j < resolucion; j++) {
00869             int top = N * (1 << j) - 1;
```

```
00870            int le = top + 1;
00871
00872            double[] temp = new double[le / 2];
00873            double[] temp2 = new double[le / 2];
00874            for (int y = 0; y < le / 2; y++) {
00875                temp[y] = phi[y];
00876                temp2[y] = psi[y];
00877            }
00878            phi = null;
00879            psi = null;
00880            phi = new double[le];
00881            psi = new double[le];
00882            for (int k = 0; k < le; k++) {
00883                phi[k] = 0;
00884                psi[k] = 0;
00885            }
00886            int a = 0;
00887            for (int k = 0; k < top; k++) {
00888                if (k % 2 == 0) {
00889                    phi[k] = temp[a];
00890                    psi[k] = temp2[a];
00891                    a++;
00892                } else {
00893                    phi[k] = 0;
00894                    psi[k] = 0;
00895                }
00896            }
00897
00898            temp = null;
00899            temp2 = null;
00900            //convolve
00901            int lon = top + N - 1;
00902            double[] out = new double[lon];
00903
00904            Convolucionador.convolvex(out, phi, top, h0, N);
00905            for (int k = 0; k < le; k++) {
00906                phi[k] = out[k];
00907            }
00908
00909            Convolucionador.convolvex(out, psi, top, h0, N);
00910            for (int k = 0; k < le; k++) {
00911                psi[k] = out[k];
00912            }
00913            out = null;
00914        }
```

```
00915
00916     }
00917 }
00918
00919 class Convolucionador {
00920
00921     public static void convolve(double[] out, double[] in, int dataSize,
double[] kernel, int kernelSize) {
00922         for (int i = 0; i < dataSize + kernelSize - 1; i++) {
00923             out[i] = 0;
00924             for (int j = 0; j < kernelSize; j++) {
00925                 if (i - j >= 0) {
00926                     out[i] += in[i - j] * kernel[j];
00927
00928                 }
00929             }
00930         }
00931     }
00932
00933     public static void convolvex(double[] out, double[] in, int dataSize,
double[] kernel, int kernelSize) {
00934         //longitud datasize + kernelsize-1
00935         for (int i = kernelSize - 1; i <= dataSize; ++i) {
00936             out[i] = 0;
00937             for (int j = i, k = 0; k < kernelSize; --j, ++k) {
00938                 out[i] += in[j] * kernel[k];
00939             }
00940         }
00941
00942         for (int i = 0; i < kernelSize - 1; ++i) {
00943             out[i] = 0;
00944
00945             for (int j = i, k = 0; j >= 0; --j, ++k) {
00946                 out[i] += in[j] * kernel[k];
00947             }
00948         }
00949     }
00950
00951 }
```

II. IGUWavenet.java

```
00001 /*
00002  * Estudio experimental de la aproximación de funciones con redes
neuronales wavenets
00003  * Se muestra este código sin permiso de modificación, solo de lectura.
00004    Cualquier asunto relacionado con el tema favor de contactar a:
00005    Dr. Oscar Herrera Alcántara   oha@correo.azc.uam.mx
00006    Ing. José Luis Ramírez Rojano luis861011@gmail.com
00007    Para ver la posibilidad de otorgar dichos permisos para mejoras y
nuevos proyectos.
00008  */
00009
00010 package examples;
00011
00012 import javax.swing.JInternalFrame;
00013 import javax.swing.JDesktopPane;
00014 import javax.swing.JMenu;
00015 import javax.swing.JMenuItem;
00016 import javax.swing.JMenuBar;
00017 import javax.swing.JFrame;
00018 import javax.swing.KeyStroke;
00019
00020 import java.awt.event.*;
00021 import java.awt.*;
00022
00023 import java.io.BufferedReader;
00024 import java.io.File;
00025 import java.io.FileReader;
00026 import java.io.IOException;
00027 import java.io.BufferedOutputStream;
00028 import java.io.BufferedReader;
00029 import java.io.BufferedWriter;
00030 import java.io.DataOutputStream;
00031 import java.io.File;
00032 import java.io.FileOutputStream;
00033 import java.io.IOException;
00034 import java.io.InputStreamReader;
00035 import java.io.OutputStreamWriter;
00036 import java.io.PrintStream;
00037 import java.io.PrintWriter;
00038 import java.util.ArrayList;
00039 import javax.swing.JFrame;
00044 public class IGUWavenet extends javax.swing.JFrame {
00045     public int filterLength;
00046     public double alfa;
00047     public int NoNeuronas;
00048     public int Precision; //El número de puntos del algoritmo de cascada
```

```
00049     public double beta;
00050     public String filename;
00051
00055     public IGUWavenet() {
00056         initComponents();
00057         this.jScrollBar1.setMinimum(0);
00058         this.jScrollBar1.setMaximum(6283185);
00059         this.jScrollBar1.setBlockIncrement(6283185/12);
00060         int defalfa = 6283185/8;
00061         this.jScrollBar1.setValue(defalfa);
00062
00063         this.jScrollBar2.setMinimum(0);
00064         this.jScrollBar2.setMaximum(6283185);
00065         int defbeta = 0;
00066         this.jScrollBar2.setBlockIncrement(6283185/12);
00067     }
00068
00074     @SuppressWarnings("unchecked")
00075     // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
00076     private void initComponents() {
00077
00078         jMenuItem1 = new javax.swing.JMenuItem();
00079         jLabel1 = new javax.swing.JLabel();
00080         jScrollPane1 = new javax.swing.JScrollPane();
00081         jList1 = new javax.swing.JList();
00082         jLabel2 = new javax.swing.JLabel();
00083         jScrollPane2 = new javax.swing.JScrollPane();
00084         jList2 = new javax.swing.JList();
00085         jScrollBar1 = new javax.swing.JScrollBar();
00086         jScrollBar2 = new javax.swing.JScrollBar();
00087         jLabel3 = new javax.swing.JLabel();
00088         jLabel4 = new javax.swing.JLabel();
00089         jLabel5 = new javax.swing.JLabel();
00090         jTextField1 = new javax.swing.JTextField();
00091         jLabel6 = new javax.swing.JLabel();
00092         jTextField2 = new javax.swing.JTextField();
00093         jButton1 = new javax.swing.JButton();
00094         jButton2 = new javax.swing.JButton();
00095         jButton4 = new javax.swing.JButton();
00096         jButton5 = new javax.swing.JButton();
00097         jLabel7 = new javax.swing.JLabel();
00098         jMenuBar2 = new javax.swing.JMenuBar();
00099         jMenu3 = new javax.swing.JMenu();
00100         jMenuItem3 = new javax.swing.JMenuItem();
```

```
00101         jMenuItem4 = new javax.swing.JMenuItem();
00102         jMenu4 = new javax.swing.JMenu();
00103
00104         jMenuItem1.setText("jMenuItem1");
00105
00106
setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
00107
00108         jLabel1.setText("Seleccione archivo .txt");
00109
00110         jList1.setModel(new javax.swing.AbstractListModel() {
00111             String[] strings = { "Seno10.txt", "Seno100.txt",
"Seno1000.txt", "Cos10.txt", "Cos100.txt", "Cos1000.txt", "Fun10.txt",
"Fun100.txt", "Fun1000.txt" };
00112             public int getSize() { return strings.length; }
00113             public Object getElementAt(int i) { return strings[i]; }
00114         });
00115         jList1.addMouseListener(new java.awt.event.MouseAdapter() {
00116             public void mouseClicked(java.awt.event.MouseEvent evt) {
00117                 jList1MouseClicked(evt);
00118             }
00119         });
00120         jScrollPane1.setViewportView(jList1);
00121
00122         jLabel2.setText("Tamaño de filtro");
00123
00124         jList2.setModel(new javax.swing.AbstractListModel() {
00125             String[] strings = { "4", "6" };
00126             public int getSize() { return strings.length; }
00127             public Object getElementAt(int i) { return strings[i]; }
00128         });
00129         jList2.addMouseListener(new java.awt.event.MouseAdapter() {
00130             public void mouseClicked(java.awt.event.MouseEvent evt) {
00131                 jList2MouseClicked(evt);
00132             }
00133         });
00134         jScrollPane2.setViewportView(jList2);
00135
00136         jScrollBar1.setOrientation(javax.swing.JScrollBar.HORIZONTAL);
00137         jScrollBar1.addMouseListener(new java.awt.event.MouseAdapter() {
00138             public void mouseClicked(java.awt.event.MouseEvent evt) {
00139                 jScrollBar1MouseClicked(evt) ;
00140             }
00141             public void mousePressed(java.awt.event.MouseEvent evt) {
00142                 jScrollBar1MousePressed(evt);
00143             }
```

```
00144            public void mouseReleased(java.awt.event.MouseEvent evt) {
00145                jScrollBar1MouseReleased(evt);
00146            }
00147        });
00148        jScrollBar1.addPropertyChangeListener(new
java.beans.PropertyChangeListener() {
00149            public void propertyChange(java.beans.PropertyChangeEvent evt)
{
00150                jScrollBar1PropertyChange(evt);
00151            }
00152        });
00153        jScrollBar1.addKeyListener(new java.awt.event.KeyAdapter() {
00154            public void keyPressed(java.awt.event.KeyEvent evt) {
00155                jScrollBar1KeyPressed(evt);
00156            }
00157        });
00158
00159        jScrollBar2.setOrientation(javax.swing.JScrollBar.HORIZONTAL);
00160        jScrollBar2.addMouseListener(new java.awt.event.MouseAdapter() {
00161            public void mouseClicked(java.awt.event.MouseEvent evt) {
00162                jScrollBar2MouseClicked(evt);
00163            }
00164        });
00165
00166        jLabel3.setText("[0, 2PI)");
00167
00168        jLabel4.setText("No. Neuronas");
00169
00170        jLabel5.setText("Estudio experimental de aproximación de funciones
con redes neuronales wavenets");
00171
00172        jTextField1.setText("10");
00173        jTextField1.addMouseListener(new java.awt.event.MouseAdapter() {
00174            public void mouseClicked(java.awt.event.MouseEvent evt) {
00175                jTextField1MouseClicked(evt);
00176            }
00177        });
00178        jTextField1.addActionListener(new java.awt.event.ActionListener()
{
00179            public void actionPerformed(java.awt.event.ActionEvent evt) {
00180                jTextField1ActionPerformed (evt);
00181            }
00182        });
00183
00184        jLabel6.setText("Precisión");
00185
```

```
00186         jTextField2.setText("13");
00187         jTextField2.addMouseListener(new java.awt.event.MouseAdapter() {
00188             public void mouseClicked(java.awt.event.MouseEvent evt) {
00189                 jTextField2MouseClicked(evt);
00190             }
00191         });
00192
00193         jButton1.setText("Ejecutar");
00194         jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
00195             public void mouseReleased(java.awt.event.MouseEvent evt) {
00196                 jButton1MouseReleased(evt);
00197             }
00198         });
00199         jButton1.addActionListener(new java.awt.event.ActionListener() {
00200             public void actionPerformed(java.awt.event.ActionEvent evt) {
00201                 jButton1ActionPerformed(evt);
00202             }
00203         });
00204
00205         jButton2.setText("Mostrar aproximación visual");
00206         jButton2.addMouseListener(new java.awt.event.MouseAdapter() {
00207             public void mouseClicked(java.awt.event.MouseEvent evt) {
00208                 jButton2MouseClicked(evt);
00209             }
00210         });
00211         jButton2.addActionListener(new java.awt.event.ActionListener() {
00212             public void actionPerformed(java.awt.event.ActionEvent evt) {
00213                 jButton2ActionPerformed(evt);
00214             }
00215         });
00216
00217         jButton4.setText("Mostrar parámetros");
00218         jButton4.addActionListener(new java.awt.event.ActionListener() {
00219             public void actionPerformed(java.awt.event.ActionEvent evt) {
00220                 jButton4ActionPerformed(evt);
00221             }
00222         });
00223
00224         jButton5.setText("Mostrar aproximación texto");
00225         jButton5.addMouseListener(new java.awt.event.MouseAdapter() {
00226             public void mouseClicked(java.awt.event.MouseEvent evt) {
00227                 jButton5MouseClicked(evt);
00228             }
00229         });
00230
```

```
00231         jLabel7.setText("1/2^K Puntos, cascada");
00232
00233         jMenu3.setText("File");
00234
00235         jMenuItem3.setText("jMenuItem3");
00236         jMenu3.add(jMenuItem3);
00237
00238         jMenuItem4.setText("Salir");
00239         jMenuItem4.addActionListener(new java.awt.event.ActionListener() {
00240             public void actionPerformed(java.awt.event.ActionEvent evt) {
00241                 jMenuItem4ActionPerformed(evt);
00242             }
00243         });
00244         jMenu3.add(jMenuItem4);
00245
00246         jMenuBar2.add(jMenu3);
00247
00248         jMenu4.setText("Edit");
00249         jMenuBar2.add(jMenu4);
00250
00251         setJMenuBar(jMenuBar2);
00252
00253         javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
00254         getContentPane().setLayout(layout);
00255         layout.setHorizontalGroup(
00256
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00257             .addGroup(layout.createSequentialGroup()
00258                 .addGap(63, 63, 63)
00259
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00260                     .addGroup(layout.createSequentialGroup()
00261
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00262                             .addComponent(jLabel1)
00263                             .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
00264
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00265
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00266                             .addComponent(jLabel2)
00267                             .addGroup(layout.createSequentialGroup()
00268
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
```

```
00269
.addGroup(layout.createSequentialGroup()
00270                                           .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 36,
javax.swing.GroupLayout.PREFERRED_SIZE)
00271                                           .addGap(38, 38, 38)
00272
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
00273                                                   .addComponent(jScrollBar1,
javax.swing.GroupLayout.DEFAULT_SIZE, 164, Short.MAX_VALUE)
00274                                                   .addComponent(jScrollBar2,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
00275                                                   .addComponent(jLabel3,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)))
00276                                       .addComponent(jLabel4)
00277                                       .addComponent(jTextField1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
00278                                       .addComponent(jLabel6)
00279
.addGroup(layout.createSequentialGroup()
00280                                           .addComponent(jTextField2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
00281
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00282                                           .addComponent(jLabel7)))
00283
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00284
.addGroup(layout.createSequentialGroup()
00285                                           .addGap(53, 53, 53)
00286                                           .addComponent(jButton1)
00287                                           .addGap(0, 0, Short.MAX_VALUE))
00288
.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
00289
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
00290
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00291                                                   .addComponent(jButton5)
00292                                                   .addComponent(jButton4)
00293                                                   .addComponent(jButton2,
javax.swing.GroupLayout.Alignment.TRAILING))))))
```

```
00294
.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
00295                     .addGroup(layout.createSequentialGroup()
00296                         .addComponent(jLabel5)
00297                         .addContainerGap(92, Short.MAX_VALUE))))
00298         );
00299         layout.setVerticalGroup(
00300
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00301             .addGroup(layout.createSequentialGroup()
00302                 .addGap(8, 8, 8)
00303                 .addComponent(jLabel5)
00304                 .addGap(18, 18, 18)
00305
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00306                     .addComponent(jLabel1)
00307                     .addComponent(jLabel2))
00308
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00309
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00310                     .addGroup(layout.createSequentialGroup()
00311                         .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 250,
javax.swing.GroupLayout.PREFERRED_SIZE)
00312                         .addGap(37, 58, Short.MAX_VALUE))
00313                     .addGroup(layout.createSequentialGroup()
00314
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
00315                             .addGroup(layout.createSequentialGroup()
00316                                 .addComponent(jScrollBar1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
00317
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00318                                 .addComponent(jLabel3))
00319                             .addGroup(layout.createSequentialGroup()
00320
.addGroup(layout.createParallelGroup(javax.  swing.GroupLayout.Alignment.LEADING)
00321                                     .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 55,
javax.swing.GroupLayout.PREFERRED_SIZE)
00322
.addGroup(layout.createSequentialGroup()
00323                                         .addGap(54, 54, 54)
00324                                         .addComponent(jScrollBar2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
00325                                 .addGap(35, 35, 35)
```

```
00326                                   .addComponent(jLabel4)
00327
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00328                                   .addComponent(jTextField1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
00329
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00330                                   .addComponent(jLabel6)
00331
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
00332
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
00333                                       .addComponent(jTextField2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
00334                                       .addComponent(jLabel7)))
00335                               .addGroup(layout.createSequentialGroup()
00336                                   .addComponent(jButton1)
00337                                   .addGap(25, 25, 25)
00338                                   .addComponent(jButton4)
00339                                   .addGap(18, 18, 18)
00340                                   .addComponent(jButton5)
00341
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
00342                                   .addComponent(jButton2)))
00343                           .addGap(0, 0, Short.MAX_VALUE))))
00344         );
00345
00346         pack();
00347     }// </editor-fold>//GEN-END:initComponents
00348
00349     private void jMenuItem4ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jMenuItem4ActionPerformed
00350         // TODO add your handling code here:
00351         System.exit(0);
00352     }//GEN-LAST:event_jMenuItem4ActionPerformed
00353
00354     private void jList2MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jList2MouseClicked
00355         // TODO add your handling code here:
00356         this.filterLength =
Integer.parseInt(jList2.getSelectedValue().toString());
00357         System.out.println("FILTERLENGTH = " + filterLength);
00358         if(this.jList2.getSelectedValue().equals("4")){
00359             this.jScrollBar2.setEnabled(false);
00360         }else if(this.jList2.getSelectedValue().equals("6")){
```

```
00361             this.jScrollBar2.setEnabled(true);
00362         }
00363     }//GEN-LAST:event_jList2MouseClicked
00364
00365     private void jButton4ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jButton4ActionPerformed
00366         // TODO add your handling code here:
00367        TestParams ti = new TestParams("/home/siul/netbeans-
7.4/IGU/src/examples/Params_ayb.txt");
00368     }//GEN-LAST:event_jButton4ActionPerformed
00369
00370     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jButton1ActionPerformed
00371         // TODO add your handling code here:
00372         pobj = new ParametrosIGU();
00373
00374         pobj.filename= this.jList1.getSelectedValue().toString();
00375         System.out.println("POBJ" + pobj.filename);
00376         //examples.WVN wvn = new examples.WVN(this.NoNeuronas, this);
00377         //String[] sa = null;
00378         //GeneticoVGA gvga = new GeneticoVGA();
00379         //gvga.main(sa);
00380         this.NoNeuronas = Integer.parseInt(this.jTextField1.getText());
00381         this.Precision = Integer.parseInt(this.jTextField2.getText());
00382         this.filterLength =
Integer.parseInt(this.jList2.getSelectedValue().toString() );
00383         this.alfa = this.jScrollBar1.getValue()/1000.0 *(2*Math.PI);
00384         this.beta = this.jScrollBar2.getValue()/1000.0 *(2*Math.PI);
00385         this.filename = this.jList1.getSelectedValue().toString();
00386
00387         System.out.println("ALFA =" + this.alfa);
00388         System.out.println("BETA =" + this.beta);
00389         System.out.println("ARCHIVO LEIDO =" + this.filename);
00390
00391         Genetic mlpgen = new Genetic(this);
00392
00393     }//GEN-LAST:event_jButton1ActionPerformed
00394 ArrayList al = null;
00395     private void jList1MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jList1MouseClicked
00396         // TODO add your handling code here:
00397            BufferedReader br = null;
00398            al = new ArrayList();
00399             try {
00400                 //String ss1 = "./src/examples/txts/Cos10.txt"; //+
```

```
00401                 String ss1 = "./src/examples/txts/" +
jList1.getSelectedValue().toString();
00402                 //ss = "./src/examples/txts/" + ss.trim();
00403                 System.out.println("Leyendo SS1  " + ss1);
00404                 File ff= new File(ss1);
00405                  //br = new BufferedReader(new FileReader("Cos10.txt"));
00406                 br = new BufferedReader(new FileReader(ff));
00407                 StringBuilder sb = new StringBuilder();
00408
00409                 String line = br.readLine();
00410
00411                 while (line != null) {
00412                     sb.append(line);
00413                     sb.append('\n');
00414                     al.add(line);
00415                     line = br.readLine();
00416                 }
00417                 everything = sb.toString();
00418
00419                 br.close();
00420                 System.out.println(everything );
00421             } catch (IOException ex) {
00422                 System.out.println("CATCH everything ");
00423             }finally{
00424
00425             }
00426
00427     }//GEN-LAST:event_jList1MouseClicked
00428
00429     private void jScrollBar1MouseReleased(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jScrollBar1MouseReleased
00430         // TODO add your handling code here:
00431         System.out.println("alfa = " +
jScrollBar1.getValue()/6283185.0*(2*Math.PI));
00432     }//GEN-LAST:event_jScrollBar1MouseReleased
00433
00434     private void jScrollBar1MouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jScrollBar1MouseClicked
00435         // TODO add your handling code here:
00436         System.out.println("alfa = " +
jScrollBar1.getValue()/6283185.0*(2*Math.PI));
00437     }//GEN-LAST:event_jScrollBar1MouseClicked
00438
00439     private void jScrollBar1MousePressed(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jScrollBar1MousePressed
00440         // TODO add your handling code here:
```

```
00441         System.out.println("alfa = " +
jScrollBar1.getValue()/6283185.0*(2*Math.PI));
00442     }//GEN-LAST:event_jScrollBar1MousePressed
00443
00444     private void jScrollBar1PropertyChange(java.beans.PropertyChangeEvent
evt) {//GEN-FIRST:event_jScrollBar1PropertyChange
00445         // TODO add your handling code here:
00446         System.out.println("alfa = " +
jScrollBar1.getValue()/6283185.0*(2*Math.PI));
00447     }//GEN-LAST:event_jScrollBar1PropertyChange
00448
00449     private void jScrollBar1KeyPressed(java.awt.event.KeyEvent evt)
{//GEN-FIRST:event_jScrollBar1KeyPressed
00450         // TODO add your handling code here:
00451         System.out.println("alfa = " +
jScrollBar1.getValue()/6283185.0*(2*Math.PI));
00452     }//GEN-LAST:event_jScrollBar1KeyPressed
00453
00454     private void jTextField1MouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jTextField1MouseClicked
00455         // TODO add your handling code here:
00456         this.NoNeuronas = Integer.parseInt(jTextField1.getText());
00457
00458     }//GEN-LAST:event_jTextField1MouseClicked
00459
00460     private void jTextField2MouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jTextField2MouseClicked
00461         // TODO add your handling code here:
00462         this.Precision = Integer.parseInt(jTextField2.getText());
00463     }//GEN-LAST:event_jTextField2MouseClicked
00464
00465     private void jButton1MouseReleased(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jButton1MouseReleased
00466         // TODO add your handling code here:
00467
00468
00469     }//GEN-LAST:event_jButton1MouseReleased
00470
00471     private void jTextField1ActionPerformed(java.awt.event.ActionEvent
evt) {//GEN-FIRST:event_jTextField1ActionPerformed
00472         // TODO add your handling code here:
00473         this.NoNeuronas = Integer.parseInt(this.jTextField1.getText());
00474
00475     }//GEN-LAST:event_jTextField1ActionPerformed
00476
00477     private void jScrollBar2MouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jScrollBar2MouseClicked
```

```
00478         // TODO add your handling code here:
00479         System.out.println("beta = " +
jScrollBar2.getValue()/6283185.0*(2*Math.PI));
00480     }//GEN-LAST:event_jScrollBar2MouseClicked
00481
00482     private void jButton2MouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jButton2MouseClicked
00483         // TODO add your handling code here:
00484
00485 TestImage ti = new TestImage("/home/siul/netbeans-
7.4/IGU/src/examples/demo.png");
00486 /*
00487
00488         JFrame f = new JFrame();
00489         f.setSize(400, 300);
00490
00491
00492
00493         f.setBounds(200, 200, 100, 100);
00494
00495         double[] values = new double[this.al.size()-1];
00496         String[] names = new String[this.al.size()-1];
00497         for(int i=0; i<values.length; i++){
00498             values[i]= Double.parseDouble(this.al.get(i).toString());
00499             names[i]= "";
00500         }
00501         System.out.println("Tantos datos " + values.length);
00502
00503         String title= "TITULO";
00504         ChartPanel  cp = new ChartPanel(values, names, "title" );
00505         //f.getContentPane().add(new ChartPanel());
00506         f.getContentPane().add(cp);
00507         //jf.pack();
00508         //jf.repaint();
00509        f.show();
00510         */
00511     }//GEN-LAST:event_jButton2MouseClicked
00512
00513     private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_jButton2ActionPerformed
00514         // TODO add your handling code here:
00515     }//GEN-LAST:event_jButton2ActionPerformed
00516
00517     private void jButton5MouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_jButton5MouseClicked
00518         // TODO add your handling code here:
```

```
00519         TestTxt  tt = new TestTxt("/home/siul/netbeans-
7.4/IGU/src/examples/aproxima.txt");
00520         //TestTxt  tt = new
TestTxt(this.jList1.getSelectedValue().toString());
00521
00522     }//GEN-LAST:event_jButton5MouseClicked
00523 ParametrosIGU pobj = null;
00524 String everything = null;
00528     public static void main(String args[]) {
00529         /* Set the Nimbus look and feel */
00530         //<editor-fold defaultstate="collapsed" desc=" Look and feel
setting code (optional) ">
00531         /* If Nimbus (introduced in Java SE 6) is not available, stay with
the default look and feel.
00532          * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
00533          */
00534         try {
00535             for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
00536                 if ("Nimbus".equals(info.getName())) {
00537
javax.swing.UIManager.setLookAndFeel(info.getClassName());
00538                     break;
00539                 }
00540             }
00541         } catch (ClassNotFoundException ex) {
00542
java.util.logging.Logger.getLogger(IGUWavenet.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
00543         } catch (InstantiationException ex) {
00544
java.util.logging.Logger.getLogger(IGUWavenet.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
00545         } catch (IllegalAccessException ex) {
00546
java.util.logging.Logger.getLogger(IGUWavenet.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
00547         } catch (javax.swing.UnsupportedLookAndFeelException ex) {
00548
java.util.logging.Logger.getLogger(IGUWavenet.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
00549         }
00550         //</editor-fold>
00551
00552
00553         /* Create and display the form */
00554         java.awt.EventQueue.invokeLater(new Runnable() {
```

```
00555             public void run() {
00556                 new IGUWavenet().setVisible(true);
00557
00558             }
00559         });
00560     }
00561
00562     // Variables declaration - do not modify//GEN-BEGIN:variables
00563     private javax.swing.JButton jButton1;
00564     private javax.swing.JButton jButton2;
00565     private javax.swing.JButton jButton4;
00566     private javax.swing.JButton jButton5;
00567     private javax.swing.JLabel jLabel1;
00568     private javax.swing.JLabel jLabel2;
00569     private javax.swing.JLabel jLabel3;
00570     private javax.swing.JLabel jLabel4;
00571     private javax.swing.JLabel jLabel5;
00572     private javax.swing.JLabel jLabel6;
00573     private javax.swing.JLabel jLabel7;
00574     private javax.swing.JList jList1;
00575     private javax.swing.JList jList2;
00576     private javax.swing.JMenu jMenu3;
00577     private javax.swing.JMenu jMenu4;
00578     private javax.swing.JMenuBar jMenuBar2;
00579     private javax.swing.JMenuItem jMenuItem1;
00580     private javax.swing.JMenuItem jMenuItem3;
00581     private javax.swing.JMenuItem jMenuItem4;
00582     private javax.swing.JScrollBar jScrollBar1;
00583     private javax.swing.JScrollBar jScrollBar2;
00584     private javax.swing.JScrollPane jScrollPane1;
00585     private javax.swing.JScrollPane jScrollPane2;
00586     private javax.swing.JTextField jTextField1;
00587     private javax.swing.JTextField jTextField2;
00588     // End of variables declaration//GEN-END:variables
00589 }
00590
```

III. ParametrosIGU.java

```
00001 /*
```

```
00002  * Estudio experimental de la aproximación de funciones con redes
neuronales wavenets
00003  * Se muestra este código sin permiso de modificación, solo de lectura.
00004    Cualquier asunto relacionado con el tema favor de contactar a:
00005    Dr. Oscar Herrera Alcántara   oha@correo.azc.uam.mx
00006    Ing. José Luis Ramírez Rojano luis861011@gmail.com
00007    Para ver la posibilidad de otorgar dichos permisos para mejoras y
nuevos proyectos.
00008  */
00009
00010 package examples;
00011
00012
00013 public class ParametrosIGU {
00014     public String filename;
00015     public int filtersize;
00016     public int neuronas;
00017     public int precisionK;
00018     public double alfa;
00019     public double beta;
00020
00021 }
```

IV. TestTxt.java

```
00001 /*
00002  * Estudio experimental de la aproximación de funciones con redes
neuronales wavenets
00003  * Se muestra este código sin permiso de modificación, solo de lectura.
00004    Cualquier asunto relacionado con el tema favor de contactar a:
00005    Dr. Oscar Herrera Alcántara   oha@correo.azc.uam.mx
00006    Ing. José Luis Ramírez Rojano luis861011@gmail.com
00007    Para ver la posibilidad de otorgar dichos permisos para mejoras y
nuevos proyectos.
00008  */
00009
00010 package examples;
00011 import javax.swing.*;
00012 import java.awt. *;
00013 import java.awt.image.*;
00014 import java.io.BufferedReader;
00015 import java.io.File;
00016 import java.io.FileReader;
00017 import java.io.IOException;
00018 import java.util.ArrayList;
```

```
00019 import javax.imageio.*;
00020 public class TestTxt
00021             extends JFrame {
00022     String filenameimg=null;
00023     public static void main(String args[]) {
00024        TestTxt ti = new TestTxt("/home/siul/netbeans-
7.4/IGU/src/examples/aproxima.txt");
00025        ti.show();
00026     }
00027
00028     public TestTxt(){
00029
00030
00031
00032         this.setSize(700, 500);
00033         this.show();
00034     }
00035     public TestTxt(String imagefile) {
00036         this();
00037         this.filenameimg = imagefile;
00038
00039             String everything="";
00040            BufferedReader br = null;
00041
00042             try {
00043
00044                 String ss1 = this.filenameimg;
00045
00046                 System.out.println("Leyendo SS1  " + ss1);
00047                 File ff= new File(ss1);
00048
00049                 br = new BufferedReader(new FileReader(ff));
00050                 StringBuilder sb = new StringBuilder();
00051
00052                 String line = br.readLine();
00053
00054                 while (line != null) {
00055                     sb.append(line);
00056                     sb.append('\n');
00057
00058                     line = br.readLine();
00059                 }
00060                 everything = sb.toString();
00061
00062                 br.close();
```

```
00063                 System.out.println(everything );
00064             } catch (IOException ex) {
00065                 System.out.println("CATCH everything ");
00066             }finally{
00067
00068             }
00069         JTextArea jta = new JTextArea();
00070         jta.setText(everything);
00071
00072         JScrollPane scroll = new JScrollPane(jta);
00073
scroll.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL_SCROLLBAR_ALWAYS)
;
00074
00075         this.getContentPane().add(scroll);
00076
00077
00078     }
00079
00080 }
```

Experimentos y Resultados.

A continuación se listan los experimentos realizados con cada una de las funciones mencionadas anteriormente con 10, 100 y 1000 puntos de la señal discreta, con parámetros libres de tamaño 4 y 6:

1. Se realizan pruebas con filtro tamaño 4, se muestra el resultado en la terminal:

- ***Experimento 1 Función Seno - 10 puntos***

FILTERLENGTH = 4

alfa = 3.919208191606755

POBJSeno10.txt

ALFA =24625.11012138069

BETA =0.0

ARCHIVO LEIDO =Seno10.txt

GENERACION= 0 FINDIV0= 0.7071067824172522

GENERACION= 1 FINDIV0= 0.7071067824172522

GENERACION= 2 FINDIV0= 0.7071067824172522

GENERACION= 3 FINDIV0= 0.7071067824172522

GENERACION= 4 FINDIV0= 0.2610218396028954

GENERACION= 5 FINDIV0= 0.2610218396028954

GENERACION= 6 FINDIV0= 0.2610218396028954

GENERACION= 7 FINDIV0= 0.2610218396028954

GENERACION= 8 FINDIV0= 0.2610218396028954

GENERACION= 9 FINDIV0= 0.2610218396028954

GENERACION= 10 FINDIV0= 0.2610218396028954

…..

GENERACION= 195 FINDIV0= 8.599750569898517E-17

GENERACION= 196 FINDIV0= 8.599750569898517E-17

GENERACION= 197 FINDIV0= 8.599750569898517E-17

GENERACION= 198 FINDIV0= 8.599750569898517E-17

GENERACION= 199 FINDIV0= 8.5997505

69898517E-17

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 8.599750569898517E-17

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 8.599750569898517E-17

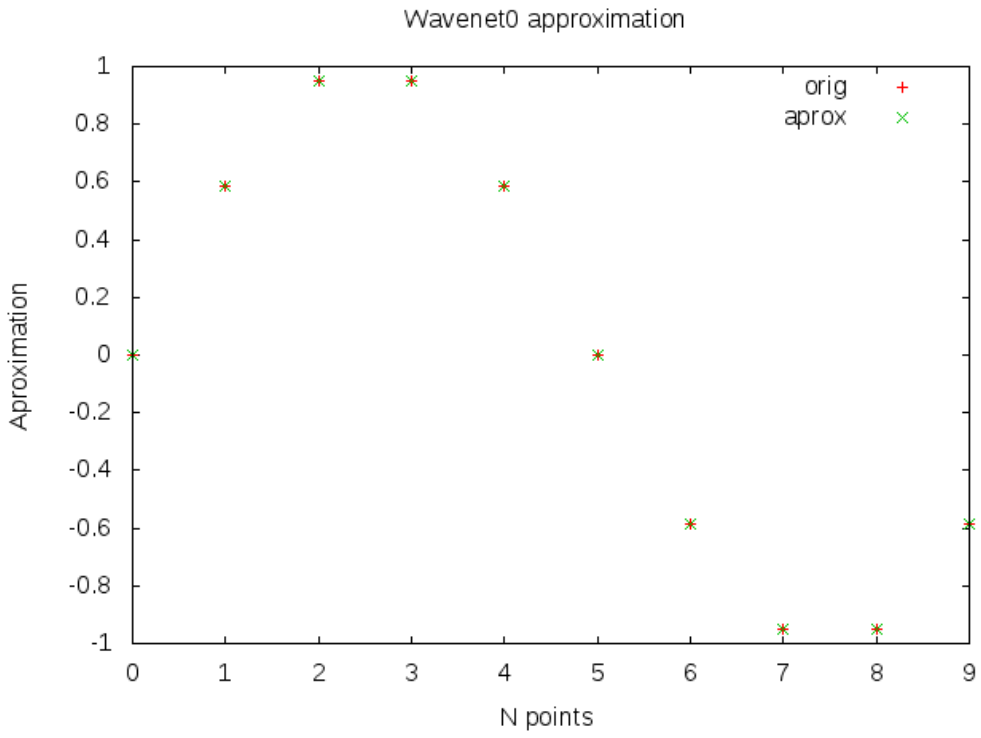


Fig. 6 Aproximación de la función Seno con 10 puntos.

- ***Experimento 2 Función Seno - 100 puntos***

FILTERLENGTH = 4

alfa = 3.048273149027482

POBJSeno100.txt

ALFA =19152.86412587224

BETA =0.0

ARCHIVO LEIDO =Seno100.txt

GENERACION= 0 FINDIV0= 0.7071067810118765

GENERACION= 1 FINDIV0= 0.7071067810115264

GENERACION= 2 FINDIV0= 0.7071067810115264

GENERACION= 3 FINDIV0= 0.7071067810115264

GENERACION= 4 FINDIV0= 0.7071067810115264

GENERACION= 5 FINDIV0= 0.7071067810115246

GENERACION= 6 FINDIV0= 0.7071067810112673

GENERACION= 7 FINDIV0= 0.7071067810112673

GENERACION= 8 FINDIV0= 0.22692672452146123

GENERACION= 9 FINDIV0= 0.22692672452146123

GENERACION= 10 FINDIV0= 0.22692672452146123

……

GENERACION= 195 FINDIV0= 9.342544893102175E-17

GENERACION= 196 FINDIV0= 9.342544893102175E-17

GENERACION= 197 FINDIV0= 9.342544893102175E-17

GENERACION= 198 FINDIV0= 9.342544893102175E-17

GENERACION= 199 FINDIV0= 9.342544893102175E-17

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 9.342544893102175E-17

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 9.342544893102175E-17

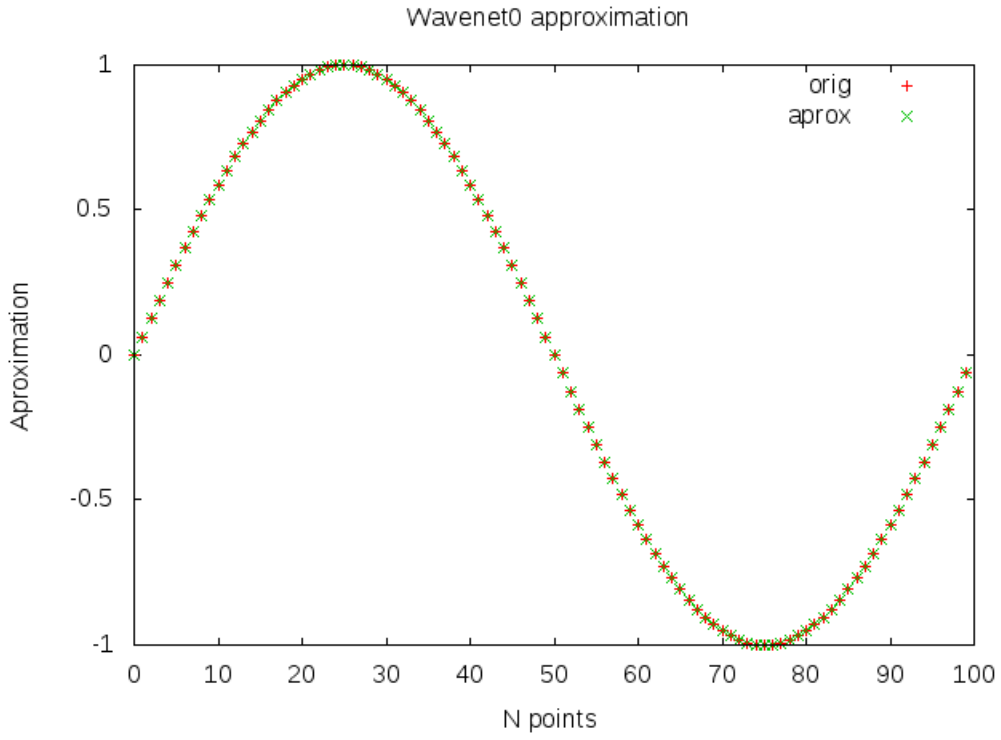


Fig. 7 Aproximación de la función Seno con 100 puntos.

- ***Experimento 3 Función Seno - 1000 puntos***

FILTERLENGTH = 4

alfa = 2.4261771186137358

POBJSeno1000.txt

ALFA =15244.119679017047

BETA =0.0

ARCHIVO LEIDO =Seno1000.txt

GENERACION= 0 FINDIV0= 162.28795180298565

GENERACION= 1 FINDIV0= 162.28795180298565

GENERACION= 2 FINDIV0= 0.7071067810639631

GENERACION= 3 FINDIV0= 0.7071067810639631

GENERACION= 4 FINDIV0= 0.7071067810639631

GENERACION= 5 FINDIV0= 0.7071067810639631

GENERACION= 6 FINDIV0= 0.7071067810639631

GENERACION= 7 FINDIV0= 0.7071067810639631

GENERACION= 8 FINDIV0= 0.7071067810639631

GENERACION= 9 FINDIV0= 0.7071067810639631

GENERACION= 10 FINDIV0= 0.7071067810639631

…..

GENERACION= 175 FINDIV0= 9.387539075475311E-17

GENERACION= 176 FINDIV0= 9.387539075475311E-17

…..

GENERACION= 197 FINDIV0= 0.0

GENERACION= 198 FINDIV0= 0.0

GENERACION= 199 FINDIV0= 0.0

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 0.0

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 0.0

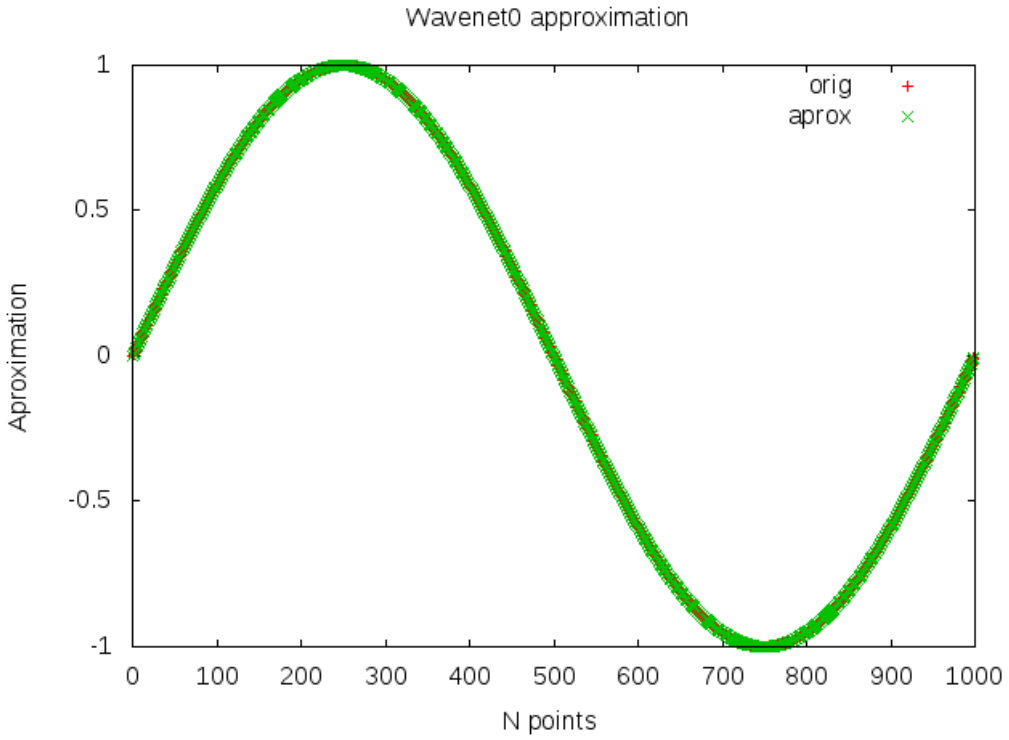


Fig. 8 Aproximación de la función Seno *con 1000 puntos.*

- ***Experimento 4 Función Cos - 10 puntos.***

FILTERLENGTH = 4

alfa = 0.7853980383974422

alfa = 2.4261771186137358

POBJCos10.txt

ALFA =15244.119679017047

BETA =0.0

ARCHIVO LEIDO =Cos10.txt

GENERACION= 0 FINDIV0= 0.7071067784195878

GENERACION= 1 FINDIV0= 0.7071067784195878

GENERACION= 2 FINDIV0= 0.7071067784195878

GENERACION= 3 FINDIV0= 0.7071067784194169

GENERACION= 4 FINDIV0= 0.7071067784194169

GENERACION= 5 FINDIV0= 0.5964487108007022

GENERACION= 6 FINDIV0= 0.5964487108007022

GENERACION= 7 FINDIV0= 0.11065806761814037

GENERACION= 8 FINDIV0= 0.11065806761814037

GENERACION= 9 FINDIV0= 0.11065806761814037

GENERACION= 10 FINDIV0= 0.11065806761814037

…..

GENERACION= 67 FINDIV0= 9.288792252416251E-17

GENERACION= 68 FINDIV0= 9.288792252416251E-17

…..

GENERACION= 196 FINDIV0= 0.0

GENERACION= 197 FINDIV0= 0.0

GENERACION= 198 FINDIV0= 0.0

GENERACION= 199 FINDIV0= 0.0

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 0.0

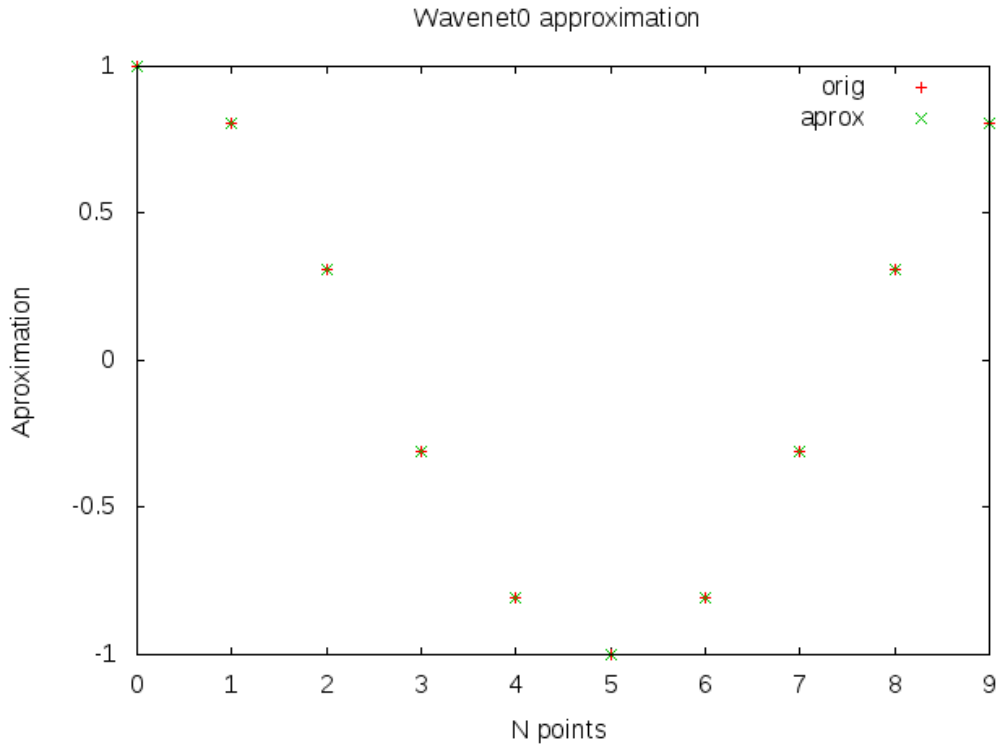
waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 0.0



}

Fig. 9 Aproximación de la función Coseno *con 10 puntos.*

- ***Experimento 5 Función Cos - 100 puntos***

FILTERLENGTH = 4

alfa = 2.737225133820609

POBJCos100.txt

ALFA =17198.491902444643

BETA =0.0

ARCHIVO LEIDO =Cos100.txt

GENERACION= 0 FINDIV0= 26.44268189115464

GENERACION= 1 FINDIV0= 0.7071067810117165

GENERACION= 2 FINDIV0= 0.7071067810117165

GENERACION= 3 FINDIV0= 0.7071067810117165

GENERACION= 4 FINDIV0= 0.707106781011493

GENERACION= 5 FINDIV0= 0.7071067810112024

GENERACION= 6 FINDIV0= 0.7071067810111352

GENERACION= 7 FINDIV0= 0.7071067810111352

GENERACION= 8 FINDIV0= 0.7071067810111352

GENERACION= 9 FINDIV0= 0.7071067810110069

GENERACION= 10 FINDIV0= 0.7071067810110069

……

GENERACION= 195 FINDIV0= 2.1579185607077513E-5

GENERACION= 196 FINDIV0= 2.1579185607077513E-5

GENERACION= 197 FINDIV0= 2.1579185607077513E-5

GENERACION= 198 FINDIV0= 2.1579185607077513E-5

GENERACION= 199 FINDIV0= 2.1579185237479277E-5

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 2.1579185237479277E-5

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 2.1579185237479277E-5

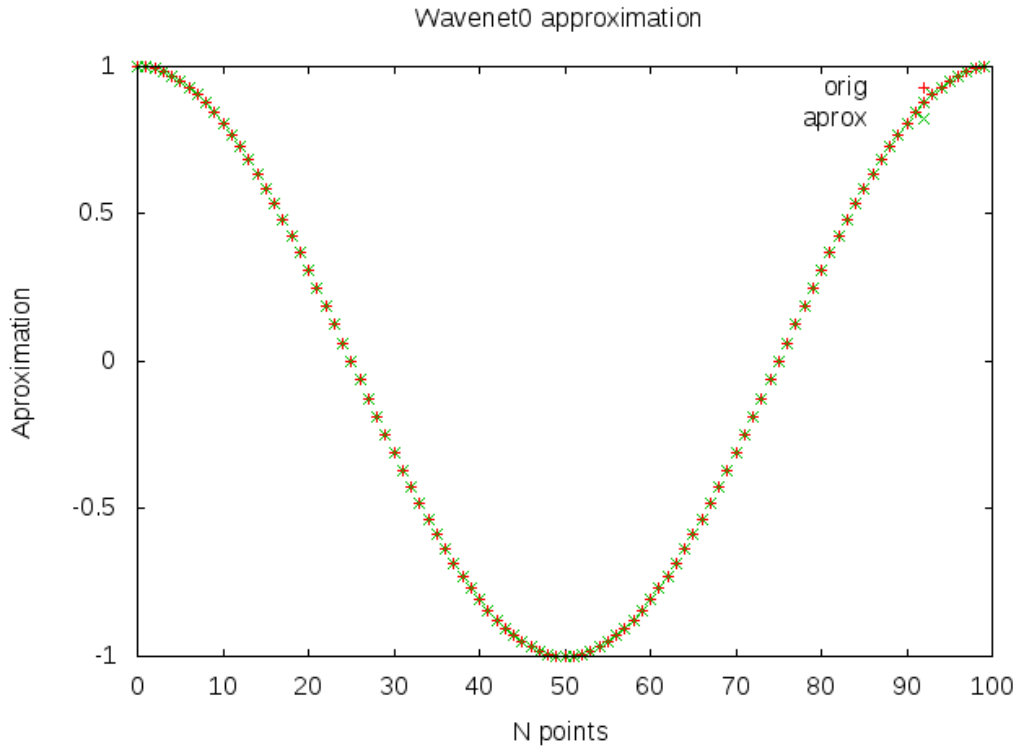


Fig. 10 Aproximación de la función Coseno *con 100 puntos.*

- ***Experimento 6 Función Cos – 1000 puntos***

POBJCos1000.txt

ALFA =8599.242809629668

BETA =0.0

ARCHIVO LEIDO =Cos1000.txt

GENERACION= 0 FINDIV0= 126.39029059215282

GENERACION= 1 FINDIV0= 0.707106781064398

GENERACION= 2 FINDIV0= 0.707106781064398

GENERACION= 3 FINDIV0= 0.7071067810642905

GENERACION= 4 FINDIV0= 0.7071067810642905

GENERACION= 5 FINDIV0= 0.7071067810642905

GENERACION= 6 FINDIV0= 0.7071067810641434

GENERACION= 7 FINDIV0= 0.7071067810641434

GENERACION= 8 FINDIV0= 0.7071067810638225

GENERACION= 9 FINDIV0= 0.7071067810638225

GENERACION= 10 FINDIV0= 0.7071067810638225

……

GENERACION= 195 FINDIV0= 2.1579183883392313E-5

GENERACION= 196 FINDIV0= 2.1579183883392313E-5

GENERACION= 197 FINDIV0= 2.1579183883392313E-5

GENERACION= 198 FINDIV0= 2.1579183883392313E-5

GENERACION= 199 FINDIV0= 2.1579183883392313E-5

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 2.1579183883392313E-5

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 2.1579183883392313E-5

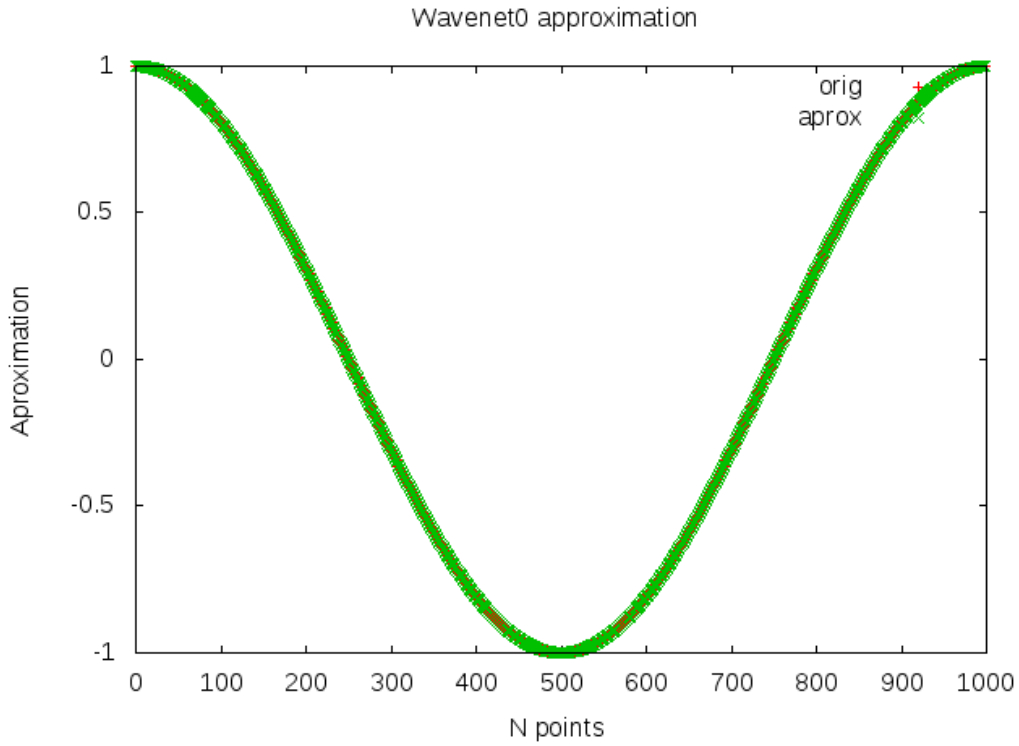


Fig. 11 Aproximación de la función Coseno *con 1000 puntos.*

- ***Experimento 7 Función f(x) – 10 puntos***

FILTERLENGTH = 4

alfa = 3.29711216119301

POBJFun10.txt

ALFA =20716.3656745255

BETA =0.0

ARCHIVO LEIDO =Fun10.txt

GENERACION= 0 FINDIV0= 5.211477846780882

GENERACION= 1 FINDIV0= 5.211477846780882

GENERACION= 2 FINDIV0= 5.211477846780369

GENERACION= 3 FINDIV0= 5.211477846780369

GENERACION= 4 FINDIV0= 5.211477846780369

GENERACION= 5 FINDIV0= 5.211477846780369

GENERACION= 6 FINDIV0= 5.211477846779883

GENERACION= 7 FINDIV0= 5.211477846779883

GENERACION= 8 FINDIV0= 5.128776171966244

GENERACION= 9 FINDIV0= 5.12623150505071

GENERACION= 10 FINDIV0= 5.126231505048871

……

GENERACION= 196 FINDIV0= 9.427094534044125E-16

GENERACION= 197 FINDIV0= 9.427094534044125E-16

GENERACION= 198 FINDIV0= 9.427094534044125E-16

GENERACION= 199 FINDIV0= 9.427094534044125E-16

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 9.427094534044125E-16

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 9.427094534044125E-16

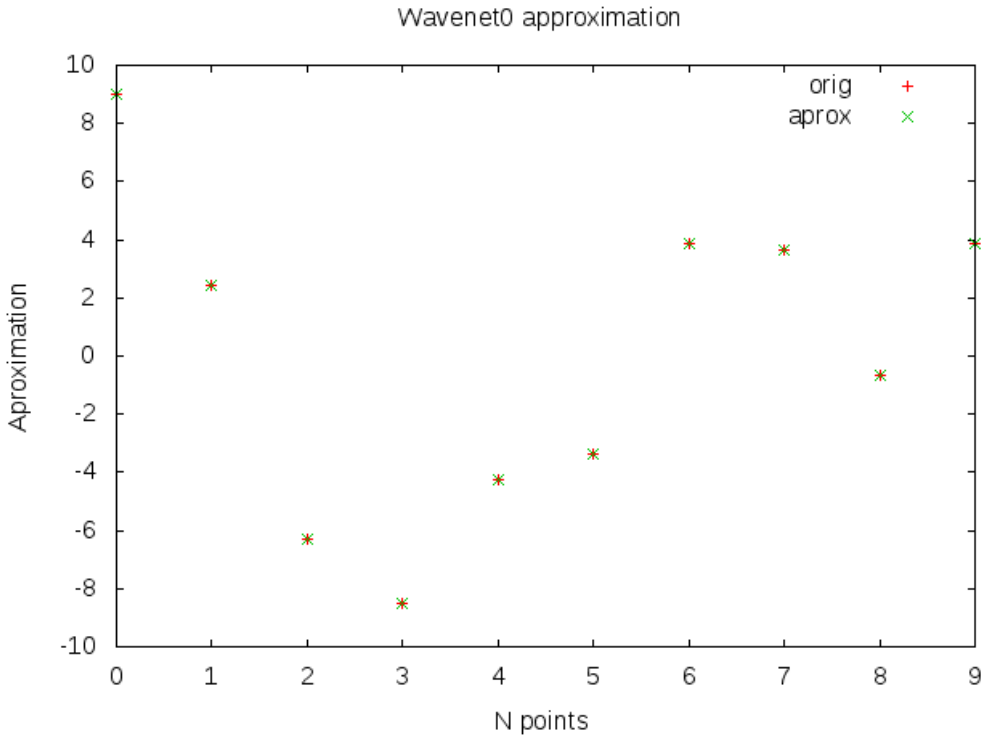


Fig. 12 Aproximación de la función f(x) *con 10 puntos.*

- ***Experimento 8 Función f(x) – 100 puntos***

FILTERLENGTH = 4

alfa = 3.29711216119301

POBJFun100.txt

ALFA =20716.3656745255

BETA =0.0

ARCHIVO LEIDO =Fun100.txt

GENERACION= 0 FINDIV0= 198.73259482064043

GENERACION= 1 FINDIV0= 4.468702036482027

GENERACION= 2 FINDIV0= 4.468702036482027

GENERACION= 3 FINDIV0= 4.4687020364785095

GENERACION= 4 FINDIV0= 4.468702036476443

GENERACION= 5 FINDIV0= 4.468702036476443

GENERACION= 6 FINDIV0= 4.468702036476443

GENERACION= 7 FINDIV0= 4.468702036475479

GENERACION= 8 FINDIV0= 4.468702036475479

GENERACION= 9 FINDIV0= 4.468702036475177

GENERACION= 10 FINDIV0= 4.4687020364750705

……

GENERACION= 195 FINDIV0= 7.16167957543428E-16

GENERACION= 196 FINDIV0= 7.16167957543428E-16

GENERACION= 197 FINDIV0= 7.16167957543428E-16

GENERACION= 198 FINDIV0= 7.16167957543428E-16

GENERACION= 199 FINDIV0= 7.16167957543428E-16

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 7.16167957543428E-16

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 7.16167957543428E-16

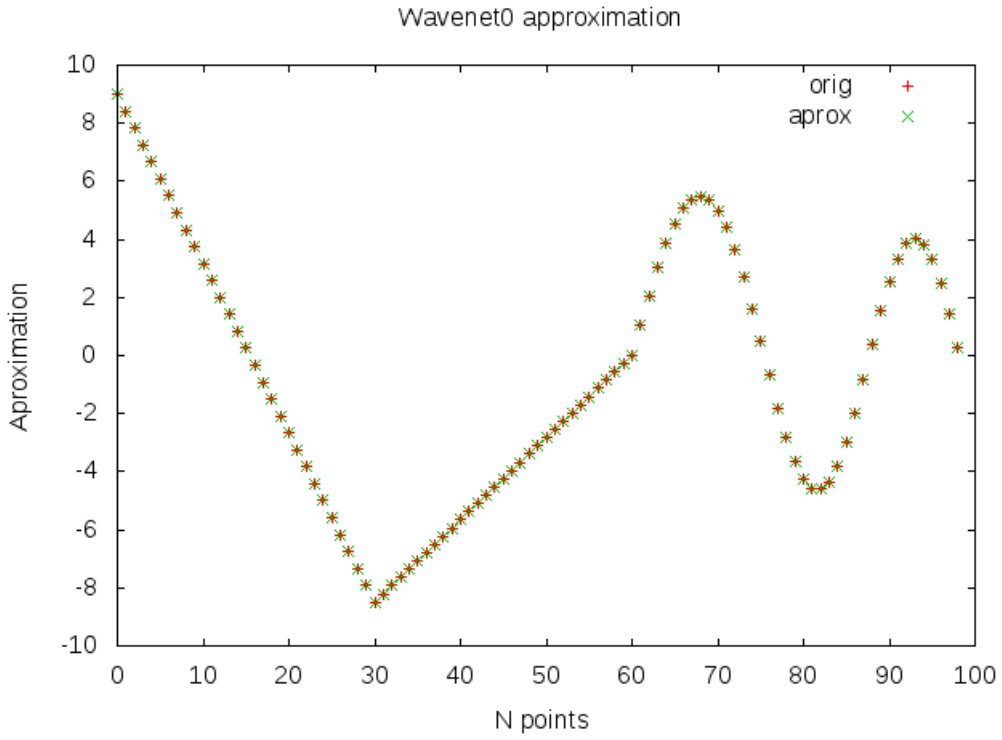


Fig. 13 Aproximación de la función f(x) *con 100 puntos.*

- ***Experimento 9 Función f(x) -1000 puntos***

FILTERLENGTH = 4

alfa = 2.115128103406814

POBJFun1000.txt

ALFA =13289.741172404145

BETA =0.0

ARCHIVO LEIDO =Fun1000.txt

GENERACION= 0 FINDIV0= 808.6367627481352

GENERACION= 1 FINDIV0= 808.6367627481352

GENERACION= 2 FINDIV0= 419.98127086115085

GENERACION= 3 FINDIV0= 4.40545630635827

GENERACION= 4 FINDIV0= 4.40545630635827

GENERACION= 5 FINDIV0= 4.40545630635827

GENERACION= 6 FINDIV0= 4.40545630635827

GENERACION= 7 FINDIV0= 4.40545630635827

GENERACION= 8 FINDIV0= 4.405456306357441

GENERACION= 9 FINDIV0= 4.405456306357366

GENERACION= 10 FINDIV0= 4.405456306355726

…….

GENERACION= 195 FINDIV0= 7.196235401519726E-16

GENERACION= 196 FINDIV0= 7.196235401519726E-16

GENERACION= 197 FINDIV0= 7.196235401519726E-16

GENERACION= 198 FINDIV0= 7.196235401519726E-16

GENERACION= 199 FINDIV0= 7.196235401519726E-16

Se ha usado un filtro de longitud 4.

#indiv=0 waveiter= 1 error= 7.196235401519726E-16

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 0.0

EL MEJOR FITNESS 7.196235401519726E-16

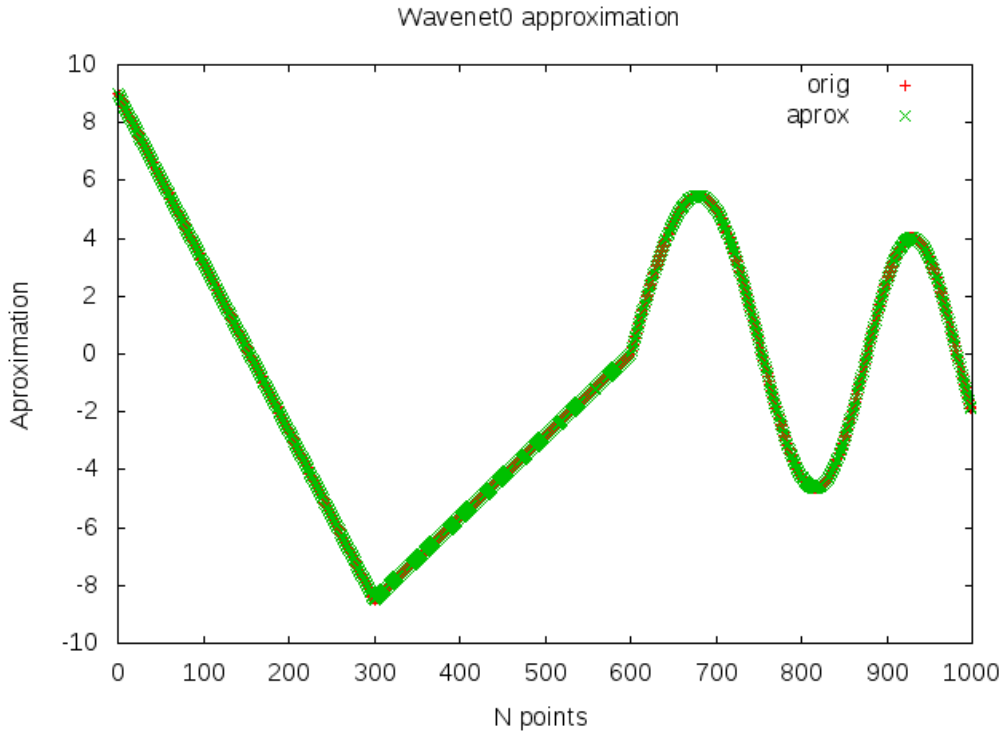


Fig. 14 Aproximación de la función f(x) *con 1000 puntos.*

2. Se realizan pruebas con filtro tamaño 6, se muestra el resultado en la terminal:

- ***Experimento 1 Función Seno - 10 puntos***

FILTERLENGTH = 6

alfa = 3.8569991885654105

POBJSeno10.txt

ALFA =24234.239446606356

BETA =13289.741172404145

ARCHIVO LEIDO =Seno10.txt

GENERACION= 0 FINDIV0= 0.5804530740105087

GENERACION= 1 FINDIV0= 0.5804530740105087

GENERACION= 2 FINDIV0= 0.5804530740105087

GENERACION= 3 FINDIV0= 0.5804530740105087

GENERACION= 4 FINDIV0= 0.5804530740105087

GENERACION= 5 FINDIV0= 0.4780558034141067

GENERACION= 6 FINDIV0= 0.11620445852345998

GENERACION= 7 FINDIV0= 0.11620445852345998

GENERACION= 8 FINDIV0= 0.013392431656082919

GENERACION= 9 FINDIV0= 0.013392431656082919

GENERACION= 10 FINDIV0= 0.013392431656082919

……..

GENERACION= 195 FINDIV0= 7.154828844670393E-7

GENERACION= 196 FINDIV0= 7.154828844670393E-7

GENERACION= 197 FINDIV0= 7.154828844670393E-7

GENERACION= 198 FINDIV0= 7.154828844670393E-7

GENERACION= 199 FINDIV0= 7.154828844670393E-7

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 7.154828844670393E-7

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 13289.741172404145

EL MEJOR FITNESS 7.154828844670393E-7

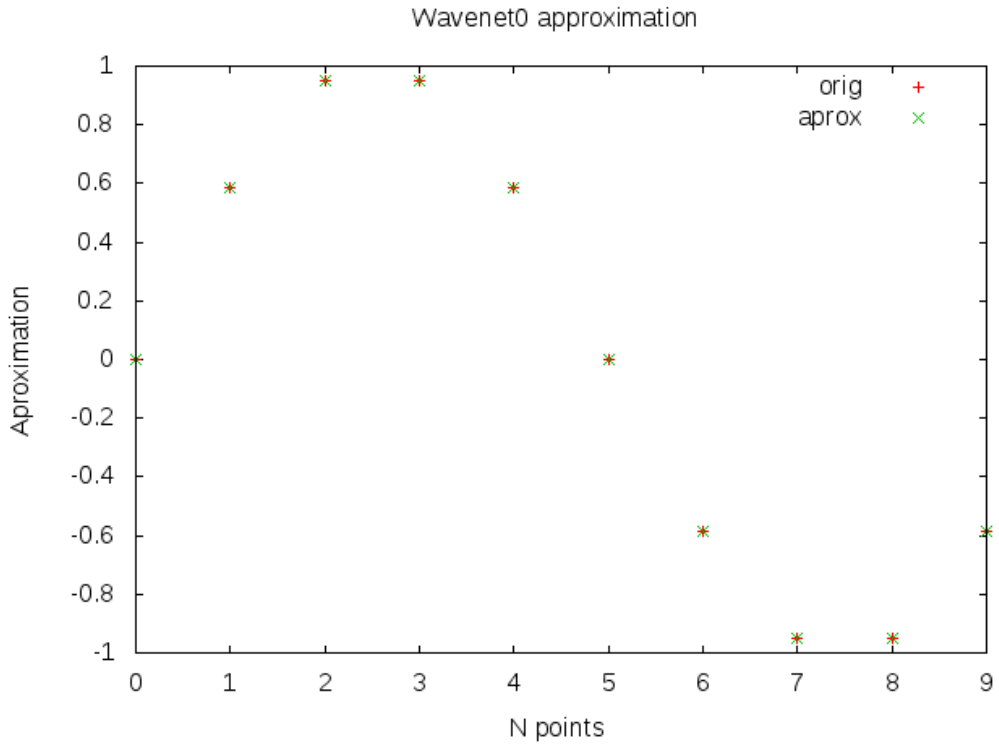


Fig. 15 Aproximación de la función Seno *con 10 puntos.*

- ***Experimento 2 Función Seno - 100 puntos***

FILTERLENGTH = 6

alfa = 4.168047203772283

alfa = 1.9907090973240749

POBJSeno100.txt

ALFA =12507.993539670168

BETA =28142.98389346155

ARCHIVO LEIDO =Seno100.txt

GENERACION= 0 FINDIV0= 0.70710678101172

GENERACION= 1 FINDIV0= 0.70710678101172

GENERACION= 2 FINDIV0= 0.70710678101172

GENERACION= 3 FINDIV0= 0.70710678101172

GENERACION= 4 FINDIV0= 0.70710678101172

GENERACION= 5 FINDIV0= 0.70710678101172

GENERACION= 6 FINDIV0= 0.70710678101172

GENERACION= 7 FINDIV0= 0.70710678101172

GENERACION= 8 FINDIV0= 0.70710678101172

GENERACION= 9 FINDIV0= 0.70710678101172

GENERACION= 10 FINDIV0= 0.70710678101172

…….

GENERACION= 195 FINDIV0= 0.031307335856700445

GENERACION= 196 FINDIV0= 0.031307335856700445

GENERACION= 197 FINDIV0= 0.031307335856700445

GENERACION= 198 FINDIV0= 0.031307335856700445

GENERACION= 199 FINDIV0= 0.031307335856700445

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 0.031307335856700445

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 28142.98389346155

EL MEJOR FITNESS 0.031307335856700445

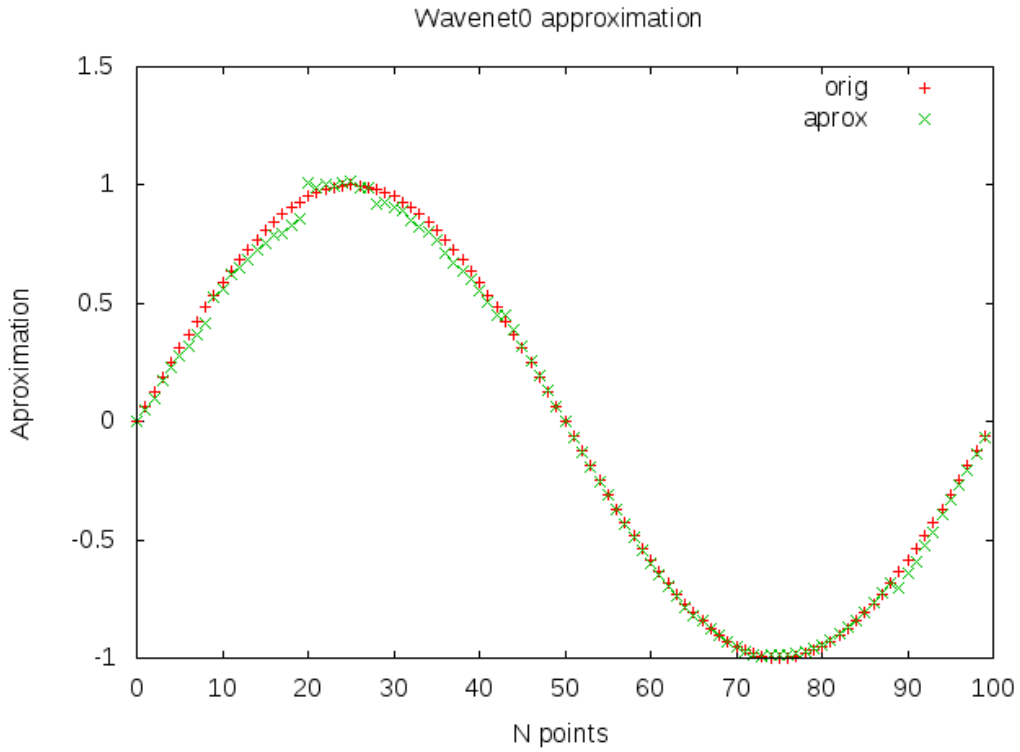


Fig. 16 Aproximación de la función Seno *con 100 puntos.*

- ***Experimento 3 Función Seno - 1000 puntos***

FILTERLENGTH = 6

POBJSeno1000.txt

ALFA =20325.48871656586

BETA =25406.87660367059

ARCHIVO LEIDO =Seno1000.txt

GENERACION= 0 FINDIV0= 0.4942996199966402

GENERACION= 1 FINDIV0= 0.4942996199966402

GENERACION= 2 FINDIV0= 0.17878115319953208

GENERACION= 3 FINDIV0= 0.09607257758640816

GENERACION= 4 FINDIV0= 0.030164485051185186

GENERACION= 5 FINDIV0= 0.030164485051185186

GENERACION= 6 FINDIV0= 0.030164485051185186

GENERACION= 7 FINDIV0= 0.030164485051185186

GENERACION= 8 FINDIV0= 0.030164485051185186

GENERACION= 9 FINDIV0= 0.030164485051185186

GENERACION= 10 FINDIV0= 0.030164485051185186

……

GENERACION= 195 FINDIV0= 5.682340764472145E-4

GENERACION= 196 FINDIV0= 5.682340764472145E-4

GENERACION= 197 FINDIV0= 5.682340764472145E-4

GENERACION= 198 FINDIV0= 5.565416389956461E-4

GENERACION= 199 FINDIV0= 5.565416389956461E-4

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 5.565416389956461E-4

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 25406.87660367059

EL MEJOR FITNESS 5.565416389956461E-4

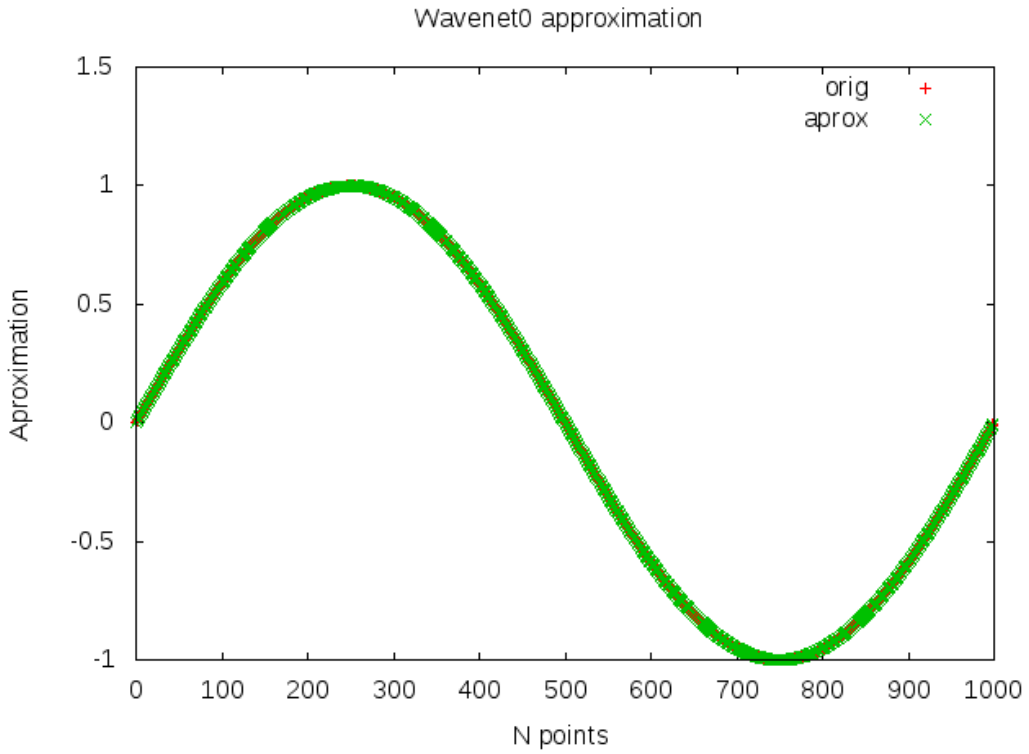


Fig. 17 Aproximación de la función Seno *con 1000 puntos.*

- ***Experimento 4 Función Cos - 10 puntos.***

FILTERLENGTH = 6

alfa = 3.2349021581516153

beta = 4.043630197689642

POBJCos10.txt

ALFA =20325.48871656586

BETA =25406.87660367059

ARCHIVO LEIDO =Cos10.txt

GENERACION= 0 FINDIV0= 0.06098972454433387

GENERACION= 1 FINDIV0= 0.05162496234117249

GENERACION= 2 FINDIV0= 0.026789928571782817

GENERACION= 3 FINDIV0= 0.01368947975637675

GENERACION= 4 FINDIV0= 0.01368947975637675

GENERACION= 5 FINDIV0= 0.01368947975637675

GENERACION= 6 FINDIV0= 0.01368947975637675

GENERACION= 7 FINDIV0= 0.01368947975637675

GENERACION= 8 FINDIV0= 0.01368947975637675

GENERACION= 9 FINDIV0= 0.01368947975637675

GENERACION= 10 FINDIV0= 0.01368947975637675

……

GENERACION= 195 FINDIV0= 4.407069114993064E-6

GENERACION= 196 FINDIV0= 4.407069114993064E-6

GENERACION= 197 FINDIV0= 2.984587392410145E-6

GENERACION= 198 FINDIV0= 2.984587392410145E-6

GENERACION= 199 FINDIV0= 2.984587392410145E-6

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 2.984587392410145E-6

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 25406.87660367059

EL MEJOR FITNESS 2.984587392410145E-6

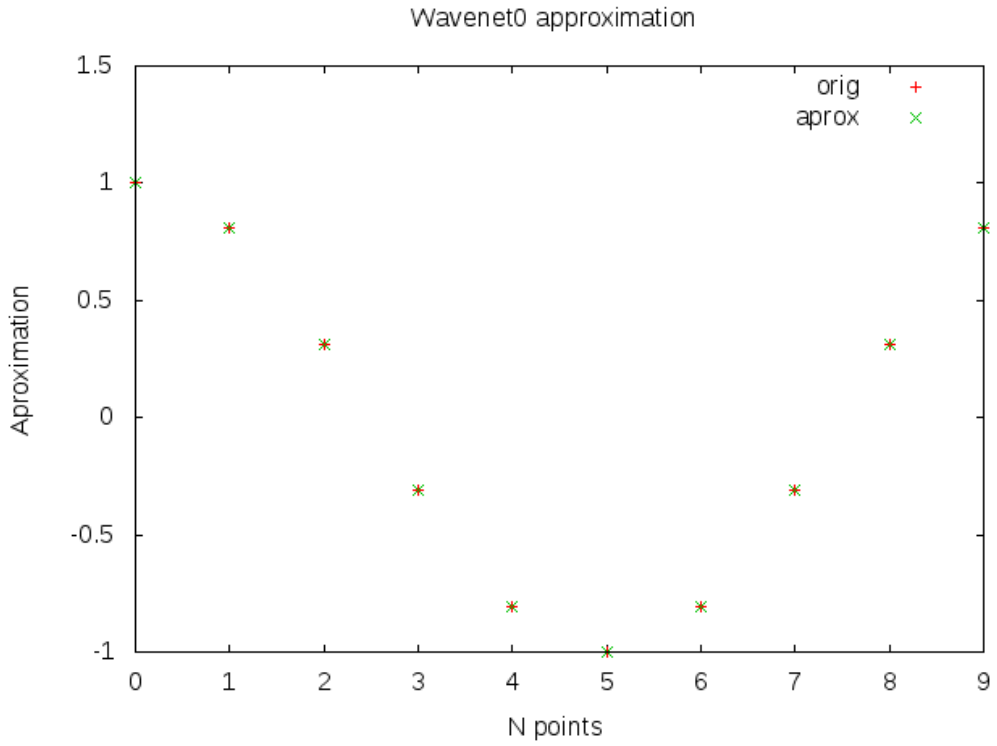


Fig. 18 Aproximación de la función Coseno *con 10 puntos.*

- ***Experimento 5 Función Cos - 100 puntos***

FILTERLENGTH = 6

alfa = 3.2349021581516153

beta = 4.043630197689642

POBJCos100.txt

ALFA =20325.48871656586

BETA =25406.87660367059

ARCHIVO LEIDO =Cos100.txt

GENERACION= 0 FINDIV0= 0.04377887091072174

GENERACION= 1 FINDIV0= 0.020528480046180752

GENERACION= 2 FINDIV0= 0.011328260211253579

GENERACION= 3 FINDIV0= 0.011328260211253579

GENERACION= 4 FINDIV0= 0.011328260211253579

GENERACION= 5 FINDIV0= 0.011328260211253579

GENERACION= 6 FINDIV0= 0.011328260211253579

GENERACION= 7 FINDIV0= 0.011077815478755656

GENERACION= 8 FINDIV0= 0.0107872596104772

GENERACION= 9 FINDIV0= 0.0107872596104772

GENERACION= 10 FINDIV0= 0.01063282335845714

…….

GENERACION= 195 FINDIV0= 1.772734501945793E-5

GENERACION= 196 FINDIV0= 1.772734501945793E-5

GENERACION= 197 FINDIV0= 1.772734501945793E-5

GENERACION= 198 FINDIV0= 1.772734501945793E-5

GENERACION= 199 FINDIV0= 1.772734501945793E-5

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 1.772734501945793E-5

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 25406.87660367059

EL MEJOR FITNESS 1.772734501945793E-5

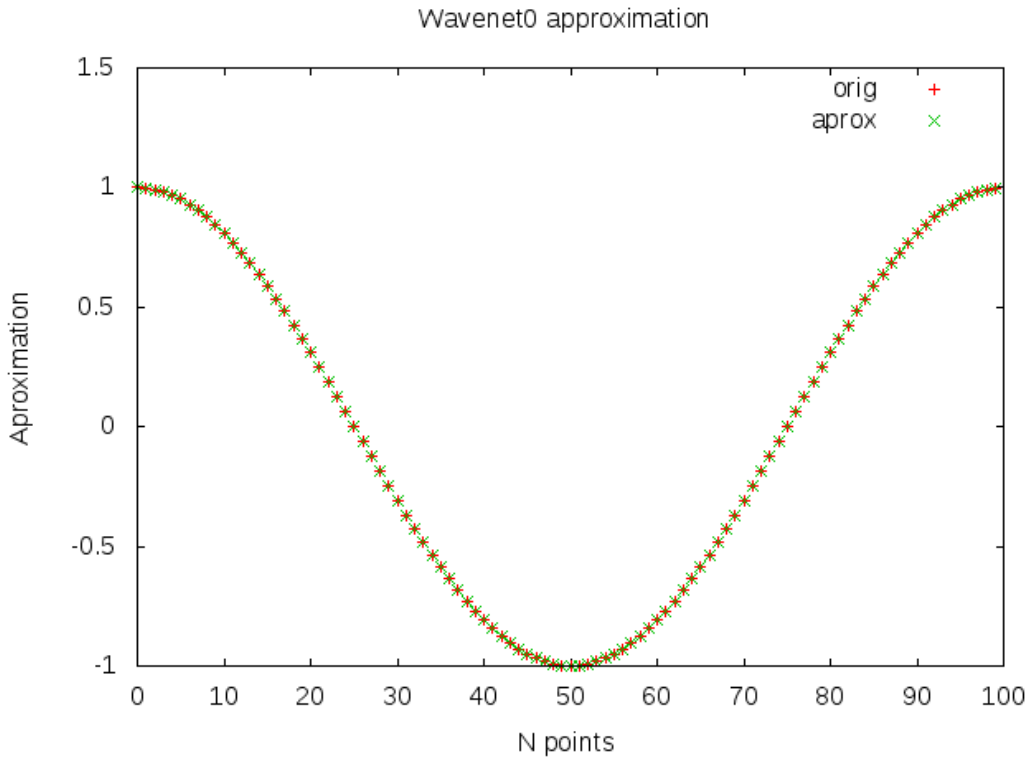


Fig. 19 Aproximación de la función Coseno *con 100 puntos.*

- ***Experimento 6 Función Cos – 1000 puntos***

FILTERLENGTH = 6

alfa = 2.3639671155723416

beta = 4.479095218979157

POBJCos1000.txt

ALFA =14853.242721057406

BETA =28142.98389346155

ARCHIVO LEIDO =Cos1000.txt

GENERACION= 0 FINDIV0= 757.2146682508944

GENERACION= 1 FINDIV0= 315.91259648148696

GENERACION= 2 FINDIV0= 315.91259648148696

GENERACION= 3 FINDIV0= 149.25009169659947

GENERACION= 4 FINDIV0= 0.707106781064398

GENERACION= 5 FINDIV0= 0.707106781064398

GENERACION= 6 FINDIV0= 0.707106781064398

GENERACION= 7 FINDIV0= 0.707106781064398

GENERACION= 8 FINDIV0= 0.707106781064398

GENERACION= 9 FINDIV0= 0.707106781064398

GENERACION= 10 FINDIV0= 0.707106781064398

…….

GENERACION= 196 FINDIV0= 0.010969045902483527

GENERACION= 197 FINDIV0= 0.010969045902483527

GENERACION= 198 FINDIV0= 0.010968149476562257

GENERACION= 199 FINDIV0= 0.010968149476562257

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 0.010968149476562257

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 28142.98389346155

EL MEJOR FITNESS 0.010968149476562257

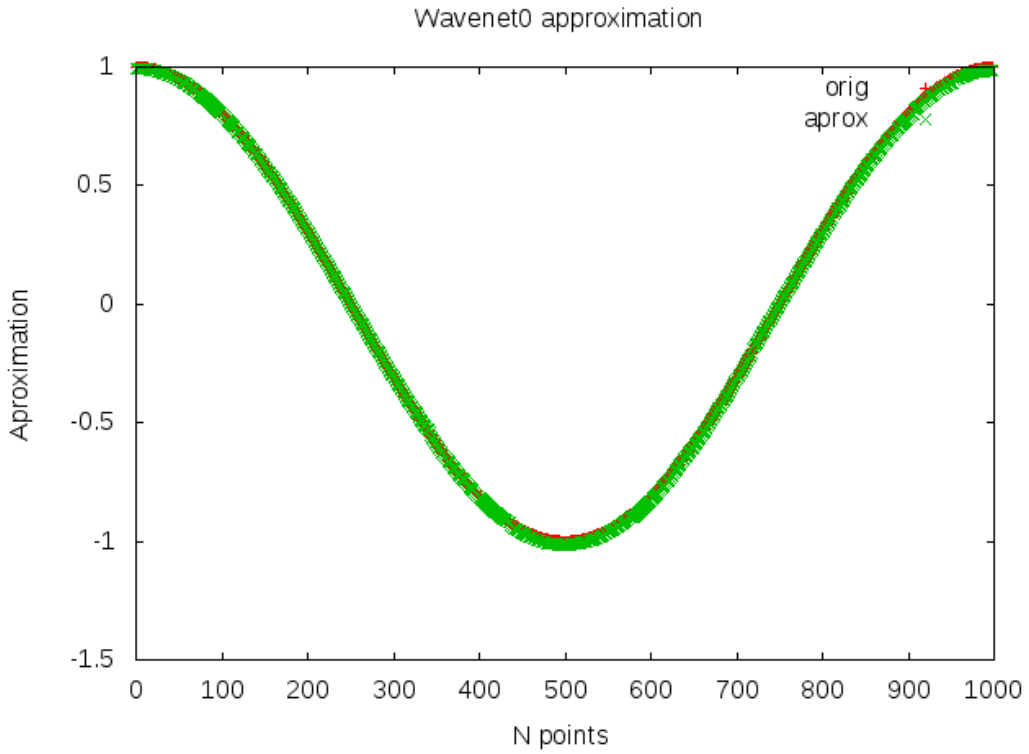


Fig. 20 Aproximación de la función Coseno *con 1000 puntos.*

- ***Experimento 7 Función f(x) – 10 puntos***

FILTERLENGTH = 6

alfa = 2.7994341368619544

alfa = 2.1773381064482087

POBJFun10.txt

ALFA =13680.618130363788

BETA =30097.362400074453

ARCHIVO LEIDO =Fun10.txt

GENERACION= 0 FINDIV0= 578.0272374083708

GENERACION= 1 FINDIV0= 12.584664984409835

GENERACION= 2 FINDIV0= 5.211477846780882

GENERACION= 3 FINDIV0= 5.211477846780882

GENERACION= 4 FINDIV0= 5.211477846780882

GENERACION= 5 FINDIV0= 3.4384659133161186

GENERACION= 6 FINDIV0= 0.5658904350880947

GENERACION= 7 FINDIV0= 0.5658904350880947

GENERACION= 8 FINDIV0= 0.5658904350880947

GENERACION= 9 FINDIV0= 0.5658904350880947

GENERACION= 10 FINDIV0= 0.5658904350880947

……

GENERACION= 195 FINDIV0= 9.838661985027478E-4

GENERACION= 196 FINDIV0= 9.838661985027478E-4

GENERACION= 197 FINDIV0= 9.838661985027478E-4

GENERACION= 198 FINDIV0= 9.838661985027478E-4

GENERACION= 199 FINDIV0= 9.838661985027478E-4

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 9.838661985027478E-4

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 30097.362400074453

EL MEJOR FITNESS 9.838661985027478E-4

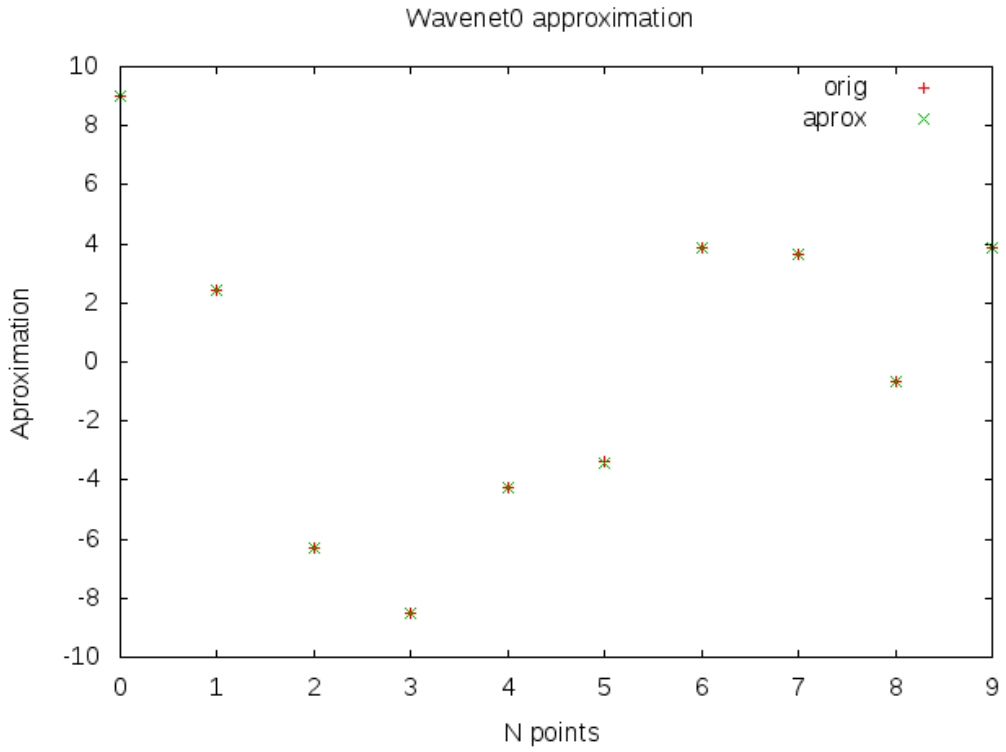


Fig. 21 Aproximación de la función f(x) *con 10 puntos.*

- ***Experimento 8 Función f(x) – 100 puntos***

FILTERLENGTH = 6

alfa = 2.1773381064482087

beta = 4.790144234186078

POBJFun100.txt

ALFA =13680.618130363788

BETA =30097.362400074453

ARCHIVO LEIDO =Fun100.txt

GENERACION= 0 FINDIV0= 4.4687020364785095

GENERACION= 1 FINDIV0= 4.4687020364785095

GENERACION= 2 FINDIV0= 4.4687020364785095

GENERACION= 3 FINDIV0= 4.4687020364785095

GENERACION= 4 FINDIV0= 4.4687020364785095

GENERACION= 5 FINDIV0= 0.8663228110261569

GENERACION= 6 FINDIV0= 0.8663228110261569

GENERACION= 7 FINDIV0= 0.8663228110261569

GENERACION= 8 FINDIV0= 0.8663228110261569

GENERACION= 9 FINDIV0= 0.8663228110261569

GENERACION= 10 FINDIV0= 0.8663228110261569

……

GENERACION= 195 FINDIV0= 5.064665665567208E-4

GENERACION= 196 FINDIV0= 5.064665665567208E-4

GENERACION= 197 FINDIV0= 5.064665665567208E-4

GENERACION= 198 FINDIV0= 5.064665665567208E-4

GENERACION= 199 FINDIV0= 5.064665665567208E-4

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 5.064665665567208E-4

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 30097.362400074453

EL MEJOR FITNESS 5.064665665567208E-4

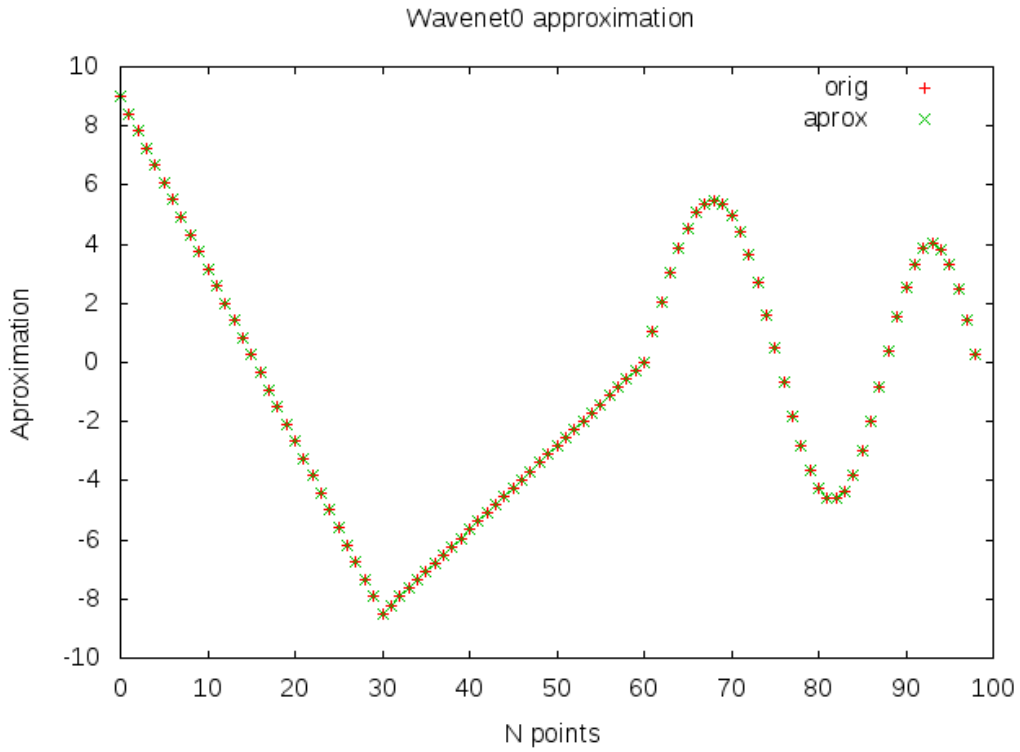


Fig. 22 Aproximación de la función f(x) *con 100 puntos.*

- ***Experimento 9 Función f(x) -1000 puntos***

FILTERLENGTH = 6

alfa = 2.3639671155723416

beta = 4.479095218979157

POBJFun1000.txt

ALFA =14853.242721057406

BETA =28142.98389346155

ARCHIVO LEIDO =Fun1000.txt

GENERACION= 0 FINDIV0= 1559.8921230754243

GENERACION= 1 FINDIV0= 1299.6190622572774

GENERACION= 2 FINDIV0= 1299.6190622572774

GENERACION= 3 FINDIV0= 1299.6190622572774

GENERACION= 4 FINDIV0= 1299.6190622572774

GENERACION= 5 FINDIV0= 384.89768806290186

GENERACION= 6 FINDIV0= 30.69706821072747

GENERACION= 7 FINDIV0= 30.69706821072747

GENERACION= 8 FINDIV0= 30.69706821072747

GENERACION= 9 FINDIV0= 4.405456306357441

GENERACION= 10 FINDIV0= 4.405456306357441

…….

GENERACION= 195 FINDIV0= 0.07668288549542031

GENERACION= 196 FINDIV0= 0.07668288549542031

GENERACION= 197 FINDIV0= 0.07668288549542031

GENERACION= 198 FINDIV0= 0.07668288549542031

GENERACION= 199 FINDIV0= 0.07668288549542031

Se ha usado un filtro de longitud 6.

#indiv=0 waveiter= 1 error= 0.07668288549542031

waiting...

done...

FINAL BEST GENALFA= 0.7853981633974483

FINAL BEST GENBETA= 28142.98389346155

EL MEJOR FITNESS 0.07668288549542031

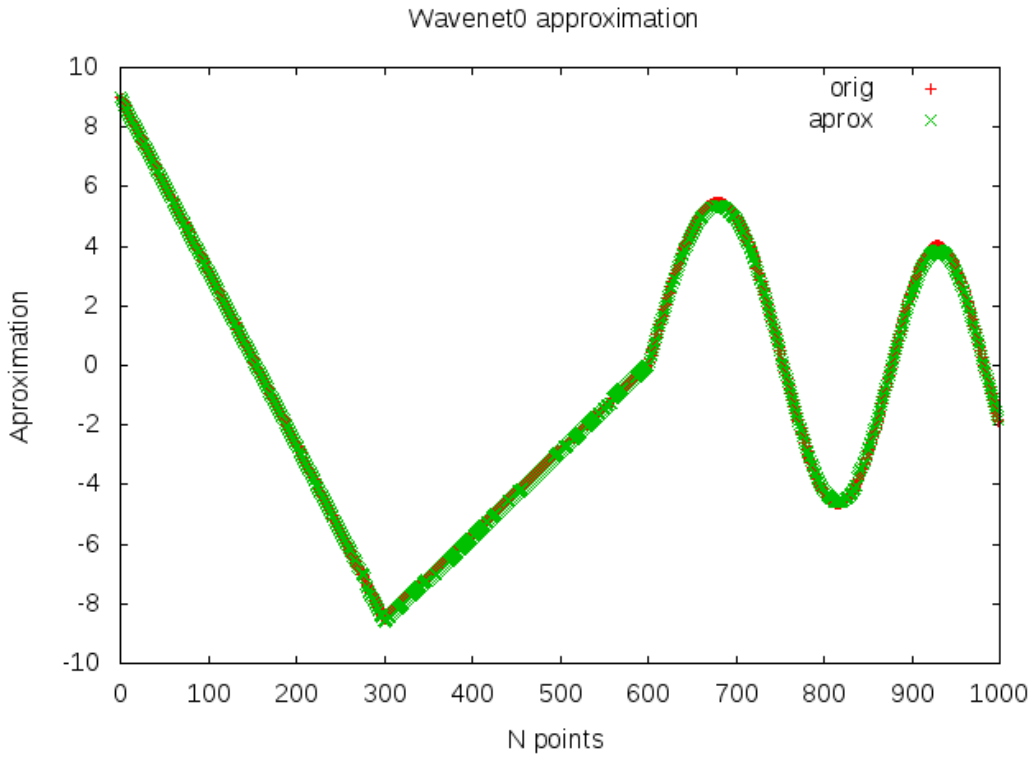


Fig. 23 Aproximación de la función f(x) *con 1000 puntos.*

Análisis, discusión de resultados y Conclusiones

El problema de la aproximación de funciones es un tema de interés matemático y de ingeniería, ya que es esencial cuando se quiere representar una función como combinación de otras más sencillas, ejemplo de ello son los monomios en las series de Taylor o las funciones trigonométricas en el Análisis de Fourier.

Con este proyecto terminal se logró el implementar algoritmos que permiten aproximar funciones matemáticas unidimensionales discretas mediante redes neuronales tipo wavenet, como son: una red neuronal que tiene como objetivo aproximar funciones unidimensionales discretas, un algoritmo cascada que sirve para aproximar el valor de wavelets ortogonales de soporte compacto paramétricos, el algoritmo genético cuya función es optimizar los parámetros de escalamiento y traslación de las funciones wavelet en la red neuronal wavenet y por ultimo una IGU que tiene como función facilitar la selección manual de parámetros libres de wavelets de soporte compacto, así como la visualización de resultados de los experimentos para aproximar funciones matemáticas unidimensionales discretas.

De la pruebas realizadas se puede observar y se concluye que los algoritmos realizan adecuadamente la aproximación con 10, 100 y 1000 puntos para cada una de las funciones discretas, con errores en el rango de 10-2 hasta 10-16.

Cabe destacar que entre más parámetros existan en los filtros paramétricos es más difícil su aproximación.

Un posible trabajo a futuro es realizar aplicaciones con estos aproximadores y trabajar en optimizar las tazas de convergencia.

Bibliografía

[1] I. Daubechies, *Ten lectures on wavelets*, 1ª edición, Philadelphia: Society For Industrial & Applied, 1992.

[2] S. Haykin, *Neural Networks and Learning Machines*, 2ª edición, Ontario, Canadá: Prentice Hall, 2009.

[3] Q. H. Zhang y A. Benveniste, “Wavelet networks”, *IEEE Trans. Neuronal Network*, vol 3, No 6, pp. 889-898, Nov. 1992.

[4] The Geek. (23/Febrero/2012). *Algoritmos genéticos y computación evolutiva* [En línea], Disponible: http://www.oracle.com/technetwork/java/javase/downloads/index.html.

[5] S. Mallat, “A theory for multiresolution signal decomposition*”, IEEE Trans. Pattern Anaysis Machine Intelligence*, vol. 11, pp. 674-693, Julio 1989.

[6] Y. Meyer, *Wavelets and Operators*, 1ª edición, Cambridge University: Cambridge, 1992.

[7] Universia. (19/febrero/2012). *Modelos de Redes Neuronales basadas en Wavelets aplicados en la Predicción y Aproximación de Series de Tiempo Caóticas* [En línea]. Disponible: http://biblioteca.universia.net/html_bura/ficha/params/title/modelos-redes-neuronales-basadas-wavelets-aplicados-prediccion-aproximacion-series tiempo/id/50860556.html.

[8] O. C. Vargas, "Implementación de una aplicación software para procesamiento de imágenes con wavelets y bancos de filtros paramétricos de reconstrucción perfecta" Propuesta de Proyecto Terminal de Ingeniería en Computación, Universidad Autónoma Metropolitana, Azcapotzalco, D.F., México, 2010.

[9] O. Cordero Pérez, "Estudio experimental de la tolerancia al ruido de imágenes procesadas con la transformada wavelet redundante con árbol dual", Propuesta de Proyecto Terminal de Ingeniería en Computación, Universidad Autónoma Metropolitana, Azcapotzalco, D.F., México, 2011.

[10] C. J. Castán Herrera, "Reconocimiento de patrones en mamografías usando redes neuronales artificiales para detectar cáncer", Propuesta de Proyecto Terminal de Ingeniería en Computación, Universidad Autónoma Metropolitana, Azcapotzalco, D.F., México, 2007.

[11] A. J. Hernández Trejo y T. D. Santiago Santiago, "Sistema de identificación de personas a través del iris mediante análisis de texturas por filtrado de wavelets", Propuesta de Proyecto Terminal de Ingeniería en Computación, Universidad Autónoma Metropolitana, Azcapotzalco, D.F., México, 2009.

[12] J. A. Garduño Martínez, "Estudio experimental de la aproximación de filtros digitales paramétricos para implementar la transformada wavelet discreta con polinomios evolutivos", Propuesta de Proyecto Terminal de Ingeniería en Computación, Universidad Autónoma Metropolitana, Azcapotzalco, D.F., México, 2010.

[13] M. S. Hernández Nieves, "Clasificación de llantos de bebé con wavelets", Propuesta de Proyecto Terminal de Ingeniería en Computación, Universidad Autónoma Metropolitana, Azcapotzalco, D.F., México, 2010.

[14] Y. Oussar y G. Dreyfus, "Initialization by selection for wavelet network training", *Proc. Neurocomputing*, vol. 34, pp. 131-143, May 2000.

[15] II Congreso Español de Informática. (25/ Marzo/ 2012). *Método Basado en Redes Neuronales Wavelet para Eliminar Ruido en Espectros Estelares* [En línea]. Disponible: http://www.lsi.us.es/redmidas/CEDI07/%5B11%5D.pdf

[16] Instituto Tecnológico Superior de Irapuato. (25/Marzo/2012). *Modelación de Procesos Industriales (Redes Neuronales Wavelet)* [En línea]. Disponible:

http://octi.guanajuato.gob.mx/octigto/formularios/ideasConcyteg/Archivos/13072006_MODELACION_PROCESOS_INDUSTRIALES.pdf

[17] Santiago. (25/Marzo/2012). *Reconocimiento del habla mediante una Red Neuronal con pre-procesamiento wavelet* [En línea]. Disponible: www.santiago.cu/hosting/linguistica/descargar.php?d=585

[18] A. Kury, "A Comprehensive Approcah to Genetic Algoritms in Optimization and Learning", *Computación y sistemas*, vol 2, No 6, pp. 218-219, 1999.

[19] Computación Aplicada al Desarrollo. (22/Febrero/2012). *Historia del lenguaje Java* [En línea]. Disponible: http://www.cad.com.mx/historia_del_lenguaje_java.htm.

[20] O. Herrera y R. Mora, "Aplicación de algoritmos genéticos a la compresión de imágenes con evolets", *Avances recientes en Sistemas Inteligentes*, Sociedad Mexicana de Inteligencia Artificial, M. González y O. Herrera, eds., pp. 157-167, México, 2011.

[21] O. Herrera y M. González, "Optimization of Parameterized Compactly Supported Orthogonal Wavelets for Data Compression", Springer LNAI 7095, *Advances in Soft Computing*, Berlin Heildelberg, Alemania, 2011, pp. 510-521. *Proceedings of Mexican International Conference on Artificial Intelligence*, Noviembre de 2011, Puebla, Pue., México.

[22] O. Herrera, "On the best evolutionary wavelet based filter to compress a specific signal", Springer LNAI 6438 *Advances in Soft Computing*, Berlin Heildelberg, Alemania, 2010, pp. 394-405. Proceedings of Mexican International Conference on Artificial Intelli-gence, MICAI, Noviembre de 2010, Pachuca Hgo., México.

Entregables

Los resultados del proyecto serán entregados en un CD con un archivo PDF, que contendrá el reporte de la "Aproximación de funciones con redes neuronales wavenets" y el código fuente documentado de cada uno de los algoritmos.