Universidad Central de Venezuela
Facultad de Ciencias

Escuela de Física

UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE FÍSICA

AUTOMATIZACIÓN DEL PROCESO DE MINERÍA DE DATOS PARA LA UTILIZACIÓN EN SISTEMAS EXPERTOS

Trabajo Especial de Grado presentado por
Jackson J. Cabezas M.
ante la Facultad de Ciencias de la
Ilustre Universidad Central de Venezuela
como requisito parcial para optar al título
de: **Licenciado en Física**
Con la tutoría de: Dra. Nuri Hurtado

Marzo-2017

Caracas-Venezuela

UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE FÍSICA

AUTOMATIZACIÓN DEL PROCESO DE MINERÍA DE DATOS PARA LA UTILIZACIÓN EN SISTEMAS EXPERTOS

Trabajo Especial de Grado presentado por
Jackson J. Cabezas M.
ante la Facultad de Ciencias de la
Ilustre Universidad Central de Venezuela
como requisito parcial para optar al título
de: **Licenciado en Física**
Con la tutoría de: Dra. Nuri Hurtado

Marzo-2017

Caracas-Venezuela

UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE FÍSICA

VEREDICTO

Quienes suscriben, miembros del Jurado designado por el Consejo de la Escuela de Física de la Facultad de Ciencias de la Universidad Central de Venezuela, para examinar el Trabajo Especial de Grado presentado por **Jackson Javier Cabezas Montañez**, Cédula de Identidad **V-16558265**, bajo el título "**Automatización del Proceso de minería de datos para la utilización en sistemas expertos**", a los fines de cumplir con el requisito legal para optar al grado de **Licenciado en Física**, dejan constancia de lo siguiente:

1. Leído como fue dicho trabajo por cada uno de los miembros del Jurado, éste fijó el día 24 de marzo de 2017, a las 2:00 pm, para que el autor lo defendiera en forma pública, lo que éste hizo en la Sala de Seminarios Guillermo Ruggeri de la Escuela de Física, mediante un resumen oral de su contenido, luego de lo cual respondió satisfactoriamente a las preguntas que le fueron formuladas por el jurado; todo ello conforme a los artículos 20, 21, 22, 25, 26 y 28 de la Normativa de Trabajo Especial de Grado de la Licenciatura en Física de la Facultad de Ciencias de la UCV vigente.

2. Finalizada la defensa pública del trabajo, el jurado decidió declararlo aprobado por considerar que se ajusta a lo dispuesto y exigido en la Normativa de Trabajo Especial de Grado de la Licenciatura en Física de la Facultad de Ciencias de la UCV vigente en sus artículos 1, 5 y 6.

Se levanta la presente acta a los 24 días del mes de marzo de 2017, dejándose también constancia de que, conforme a la normativa jurídica vigente, actuó como coordinadora del jurado la tutora del Trabajo Especial de Grado Profa. Nuri Janil Hurtado Villasana.

Firma del jurado evaluador

Prof Alberto José Bellorín
V-8382750
UCV

Prof. Levi García
E-81093009
UCV

Profa. Nuri Janil Hurtado
V-6180156
UCV

REPÚBLICA BOLIVARIANA DE VENEZUELA
Facultad de Ciencias
Escuela de Física
DIRECCIÓN
UNIVERSIDAD CENTRAL DE VENEZUELA

A mis hijos, Ender Javier y Elismar Isabella,
mis fuentes de inspiración.
A mi esposa Elizabeth Torres,
por su compañía durante el desarrollo de este trabajo.
A mis padres, Blanca y Raúl,
por ser ejemplos de constancia y dedicación.

Agradecimientos

Gracias a Dios por iluminarme y guiarme en todo momento.

Gracias a mi Profesora/Tutora Dra. Nuri Hurtado por darme la oportunidad de ser parte de su equipo, desarrollando este trabajo.

Agradezco a mis padres por todo el apoyo incondicional durante mis años de formación en esta casa de estudios. A mis hermanos, Blendys Yorbelis, Raúl Junior, Roy Ender, Enzoni Ismael, Raúl Roger y Nayobi, por la compañía. A Margaret Piña por apoyarme en todo momento. Los quiero.

Gracias a mis amigos y profesores que tuve a lo largo de mi carrera, siempre los llevo presentes en mi mente y mi corazón. Gracias a todos ustedes por acompañarme, creer en mí, por su amistad, por su dedicación, por enseñarme y formarme en esta casa de estudios que considero mi segundo hogar, la Facultad de Ciencias de la Universidad Central de Venezuela.

RESUMEN

En el siguiente trabajo se aplicaron procedimientos de programación para automatizar las distintas etapas del proceso de minería de datos utilizando modelos: matemáticos, como el método de los mínimos cuadrados; estadísticos, empleando variables como la correlación, la desviación estándar y el RMSE; y de inteligencia artificial, como el sistema neuro-difuso (SND) ANFIS. Para lograrlo, se diseñó una aplicación empleando herramientas de la plataforma de MatLab, con la interfaz gráfica de usuario (GUI). La aplicación cuenta con varias interfaces, módulos de controles y entornos gráficos (2D) en distintas escalas lineal-logarítmica para la visualización, comparación y análisis de los datos. Además la aplicación fue evaluada con datos provenientes de fenómenos físicos, como los preestablecidos en trabajos especiales de grado (TEG) ya culminados en el Laboratorio de Física (UCV), donde el procedimiento asociado a la minería de datos es ahora automático.

Palabras claves: Minería de Datos, Redes Neuronales Artificiales (RNA), Lógica Difusa, Sistemas Neuro-Difusos (SND), ANFIS, Interfaz Gráfica de Usuario (GUI) de MatLab.

Índice general

Índice de figuras

Capítulo 1

Introducción

Vivimos en una época donde la cantidad de datos crece diariamente de manera acelerada y muchas veces, sin percatarnos, no hemos logrado procesarlos en su totalidad, tal vez por falta de recursos humanos, financieros, logísticos, tecnológicos o de conocimientos. Es posible que estas y otras faltas nos impidan avanzar a la misma velocidad con la que van los datos. Es por ello que muchas organizaciones y empresas invierten gran parte de su potencial y recursos por mantenerse al día con el procesamiento de sus datos.

En general, procesar los datos no es suficiente, sino obtenemos de ello información clara con la cual podamos describir el problema que nos estamos planteando y al recurrir a métodos manuales o programas no automatizados nos llevaría a generar pérdidas algo significativas de algunos recursos. Estas incidencias materializan en la década de los 90, el término *Data Mining* o *Minería de Datos* [1], con el que se busca extraer información útil y valiosa de una gran cantidad de datos, identificando patrones y tendencias revelados a través de determinados modelos.

En el presente trabajo se pretende desarrollar un sistema automatizado lo suficientemente robusto y fácil de manejar, capaz de preparar y procesar cualquier tipo de dato, empleando modelos estadísticos [2], matemáticos [3] y de inteligencia artificial [4, 5, 6, 7, 8, 9, 10, 11, 12] para el estudio lineal o no de los datos, y poder ser implementados por sistemas expertos. Matemáticamente usaremos el método de los mínimos cuadrados [3], estadísticamente usaremos variables como la correlación, la desviación estándar y el error cuadrático medio (RMSE) [2], y en el caso de inteligencia artificial describiremos las redes neuronales artificiales [4, 5, 6, 7], la lógica difusa [8, 9, 10] y el sistema neuro-difuso ANFIS [9, 11, 12], siendo este último el que utilizamos.

Para el diseño de la aplicación se utilizaron herramientas de la plataforma de MatLab, como la interfaz gráfica de usuario (GUI) [13], desarrollando varias interfaces y controles para cada una de las etapas de la minería de datos [1]. Adicionalmente se muestran entornos gráficos 2D en distintas escalas lineal-logarítmica para visualizar, comparar e

interpretar los resultados.

Algunos de los TEGs desarrollados en el Laboratorio de Física Teórica del Sólido, basados en áreas como: petrofísica [14, 15, 16, 17, 18, 19, 20], magnetismo de roca [21, 22, 23], paleoclimatología [24, 25, 26], han utilizado herramientas de inteligencia artificial para el estudio de los datos de manera manual. En estos trabajos se han utilizado y generado una gran cantidad de datos (minería de datos), que ahora podrán ser procesados de manera automatizada con el desarrollo de un software adecuado.

Este trabajo está esquematizado de la siguiente manera: En el capítulo II se describen las distintas etapas de la minería de datos, en qué consisten los modelos matemáticos, estadísticos y de inteligencia artificial, así como las distintas herramientas utilizadas de la plataforma de cálculo MatLab. En el capítulo III se presenta la metodología detallándose paso a paso como se construyeron, desarrollaron y programaron los controles de cada interfaz gráfica, con una breve descripción de las funciones de MatLab implementadas en el código. En el capítulo IV se muestran los resultados obtenidos tras procesar un archivo de datos con la aplicación desarrollada. En el capítulo V finalizamos con las conclusiones a las cuales hemos llegado tras los resultados durante el desarrollo e implementación de la aplicación diseñada.

Capítulo 2

Marco teórico

En este capítulo presentaremos los conceptos más importantes, asociados al desarrollo de este trabajo. La organización del mismo es conforme a los lineamientos de lo que se conoce como el proceso de minería de datos, donde resaltaremos algunas herramientas de la estadística y algunos modelos de sistemas expertos que pueden ser los asociados al reconocimiento de patrones. Adicionalmente presentaremos herramientas de la plataforma de MatLab, como la interfaz gráfica de usuario (GUI), que nos permitirá diseñar y automatizar todo lo que involucra el proceso de minería de datos.

2.1. Minería de Datos

En la minería de datos se abarcan distintos procesos y métodos donde se trabaja con grandes cantidades de datos haciendo uso de herramientas como base de datos, estadística, sistemas y programas computacionales, con la finalidad de generar información.

Podemos definir la minería de datos como el proceso de percibir información de un gran conjunto de datos, el cual puede mostrar ciertos patrones y tendencias después de realizar sobre este conjunto un análisis matemático. Estos patrones y tendencias se pueden recopilar y definir como un modelo de minería de datos, el cual forma parte de un proceso dinámico e iterativo que puede representarse en un esquema como el mostrado en la figura 2.1 [1]. A continuación se detalla cada etapa.

- **Definir el problema**

Primeramente se debe analizar y definir cada una de las variables involucradas que se evaluarán en el modelo, considerando las formas de usar los datos y concretar los objetivos del proyecto de minería de datos que proporcionarán una respuesta al problema planteado.

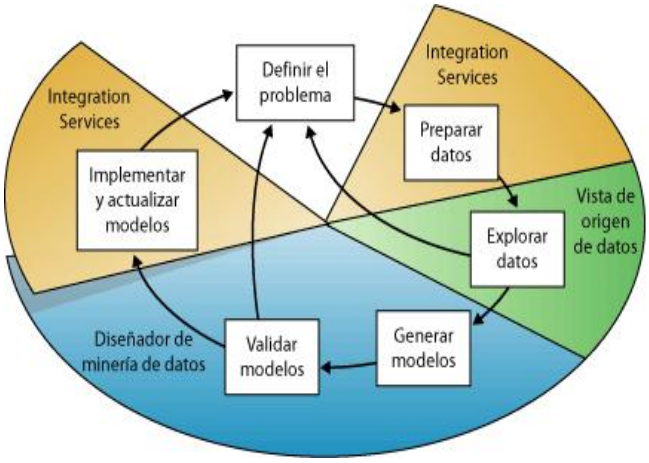


Figura 2.1: Pasos del proceso de minería de datos (Modificado de: Microsoft SQL Server 2014).

- **Preparar los datos**

Los datos que se usan no necesariamente provienen de una base de datos. Se puede realizar minería de datos con cualquier tipo de dato de origen definido, por ejemplo, archivos de texto, libros de Excel, figuras, archivos de Matlab, entre otros. Es decir, los datos pueden estar dispersos y almacenados en formatos distintos, pueden contener incoherencias como entradas que faltan, de apariencias independientes o incorrectas, es por ello que debemos limpiar y filtrar los datos identificados en el paso anterior. Adicionalmente debemos buscar las correlaciones ocultas en los datos, identificar sus orígenes y determinar qué columnas son las más adecuadas para el análisis [1].

- **Explorar los datos**

Para conocer más el problema debemos explorar los datos y poder decidir si el conjunto de datos contienen datos defectuosos, para luego crear una estrategia de corrección y así obtener una descripción más profunda de los comportamientos de los datos [1].

Las herramientas estadísticas, nos permiten agrupar, ordenar, clasificar y representar gráficamente, los datos que se deriven de diferentes fenómenos, con la intención de analizarlos mediante métodos apropiados a fin de crear modelos que nos orienten en la toma de decisiones, el reconocimiento de patrones, la predicción o inferencia.

- **Generar modelos**

Como se mencionó anteriormente, los datos con los que se trabaje pueden regirse por modelos lineales, no lineales o no tener ningún modelo que lo caracterice.

Un modelo de minería de datos es un contenedor que especifica las columnas que se usan para la entrada, el atributo que se está infiriendo y los parámetros que indican al algoritmo cómo procesar los datos. El procesamiento a menudo se denomina entrenamiento, el cual hace referencia al proceso de aplicar un algoritmo matemático concreto a los datos de la estructura para extraer patrones. Los patrones que se encuentre en el proceso de entrenamiento dependerá de la selección de los datos de entrenamiento, el algoritmo que elija y cómo se haya configurado el algoritmo [1].

Como punto de partida podemos buscar la posible existencia de una relación lineal entre variables. Graficando las variables podemos establecer cualitativa y cuantitativamente si la relación entre ellas es lineal o no y su grado de dispersión, como se muestra en la figura 2.2. Los datos estudiados podrían venir descritos por algunos de los ajustes que el graficador usado posea, en cuyo caso, se contará con una ecuación de inferencia del comportamiento del conjunto de datos a estudiar.

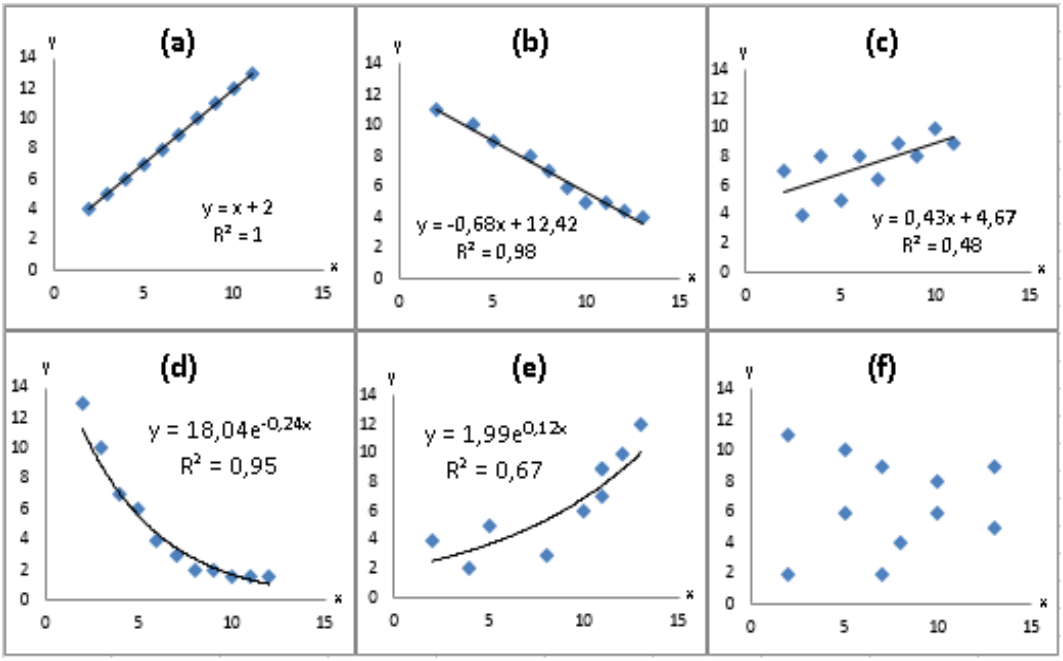


Figura 2.2: Buen ajuste lineal de los datos: (a) sin dispersión, (b) con poca dispersión, (c) el ajuste lineal no es bueno. Buen ajuste exponencial: (d) con poca dispersión, (e) con dispersión. (f) Los datos no pueden ser correlacionados.

De esta manera, se inician los primeros pasos para abordar temas de estadística descriptiva e inferencial, donde el objetivo primordial en el análisis de regresión es obtener una ecuación que represente la relación entre las variables. En general se pueden encontrar estos tipos de ajuste en casi todos los graficadores comerciales:

- Lineal,
$$f(x) = ax + b \tag{2.1}$$
- Polinómico,
$$f(x) = ax^n + ... + bx + c \tag{2.2}$$
- Exponencial,
$$f(x) = a(b^x) \tag{2.3}$$
- Logarítmica,
$$f(x) = alog_b x \tag{2.4}$$

De igual manera podemos extendernos a tres o más variables haciendo un estudio de regresión lineal múltiple, donde existen dos o más variables independientes $(x_1, x_2, ..., x_n)$ y una variable dependiente (y). La curva de ajuste lineal sería de la forma:

$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \tag{2.5}$$

donde el problema radica en conseguir cada uno de los coeficientes β_i.

- **Validar Modelos**

Al generar un modelo es importante realizar varias pruebas con la finalidad de ver cual ofrece mejores resultados al problema planteado. Una técnica es separar los datos en conjuntos de datos de entrenamiento y prueba, para poder evaluar el rendimiento de todos los modelos. El conjunto de datos de entrenamiento se utiliza para generar el modelo y el conjunto de datos de prueba para comprobar la precisión del modelo mediante la creación de consultas de inferencia [1].

Si resulta que ninguno de los modelos generados funciona correctamente, se recomienda regresar al paso anterior, redefinir el problema planteado o revisar el origen de los datos. Para validar los modelos empleados en el presente trabajo utilizaremos algunos TEGs desarrollados en el Laboratorio de Física Teórica del Sólido, basados en áreas como: petrofísica, magnetismo de roca, climatología, propagación de ondas, etc., con la finalidad de comparar todos los resultados obtenidos manualmente en dichos trabajos y el proceso que se está empleando de manera automatizada.

- **Implementar y Actualizar Modelos**

En este último paso del proceso de minería de datos como se ilustró en la figura 2.1 se pueden llevar acabo diferentes tareas, dependiendo del problema planteado, entre las cuales podemos especificar:

- El uso de los modelos nos permitirá crear inferencia para luego tomar de decisiones.
- El uso de herramientas estadísticas, reglas o fórmulas de los modelos, nos permitirá consultar ciertas tendencias y patrones que puedan generar los datos.
- Incrustar la funcionalidad del proceso de minería de datos directamente en una aplicación.
- Crear un informe que permita a los usuarios realizar consultas directamente en un modelo de minería de datos existente.
- Actualizar los modelos después de la revisión y análisis.
- Actualizar dinámicamente los modelos, cuando entre más datos, y realizar modificaciones constantes para mejorar la efectividad de la solución.

2.2. Modelos

Al procesar un conjunto de datos los patrones o tendencias que estos puedan generar, pueden ser descritos a través de modelos lineales, modelos no-lineales, o no poder ser descritos por ningún modelo. Es por ello que en el desarrollo de este trabajo nos basaremos en algunas herramientas estadísticas y describiremos varios modelos de sistemas expertos con la intención de familiarizarnos finalmente con los sistemas neuro-difusos.

2.2.1. Modelo lineal

Para un conjunto de N datos $\{x_i,y_i\}$, podemos determinar numéricamente con el coeficiente de correlación de Pearson (R) y el coeficiente de determinación (R^2), el cual establece el grado de confiabilidad de la tendencia, que se pueda establecer con dichos datos, como se muestra en la figura 2.2. El coeficiente de correlación lineal de Pearson puede determinarse a través de la siguiente relación [2]:

$$R = \frac{N\sum_{i=1}^{N}(x_i y_i) - (\sum_{i=1}^{N} x_i)(\sum_{i=1}^{N} y_i)}{\sqrt{[N(\sum_{i=1}^{N} x_i^2) - (\sum_{i=1}^{N} x_i)^2][N(\sum_{i=1}^{N} y_i^2) - (\sum_{i=1}^{N} y_i)^2]}} \tag{2.6}$$

Si el tipo de ajuste obtenido es lineal (ver ec. 2.1), el problema se reduce a calcular estadísticamente el término independiente o punto de corte (b), y el coeficiente que acompaña la variable independiente o pendiente de la curva (a), usando el método de mínimos cuadrados [3] podemos obtener:

- La pendiente

$$a = \frac{N\sum_{i=1}^{N}(x_i y_i) - (\sum_{i=1}^{N} x_i)(\sum_{i=1}^{N} y_i)}{N\sum_{i=1}^{N}(x_i^2) - (\sum_{i=1}^{N} x_i)^2} \tag{2.7}$$

- El punto de corte

$$b = \frac{\sum_{i=1}^{N}(y_i) - a\sum_{i=1}^{N}(x_i)}{N} \tag{2.8}$$

Un parámetro estadístico de gran relevancia es la desviación estándar (σ), ya que es una medida de dispersión, por lo que nos indica la separación de los datos respecto a la media, es decir, entre más pequeña sea la desviación estándar, mayor será la concentración de los puntos alrededor de la media $\bar{y}$. Podemos definir a la desviación estándar de la siguiente manera:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{2.9}$$

donde $\bar{y} = \dfrac{\sum_{i=1}^{N} y_i}{N}$ es la media aritmética.

Otro parámetro importante en el análisis estadístico de los datos, en el proceso de inferencia de valores con una ecuación específica es el $RMSE$ (Root Mean Square Error) también conocida como raíz cuadrada de la varianza de los residuos. Este parámetro es una medida relativa de ajuste y es importante si el propósito principal del modelo es la inferencia. Puede calcularse a partir de la ecuación 2.10:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - y_{inf})^2} \tag{2.10}$$

donde y_{inf} representa los datos inferidos y y_i, los datos reales.

Como se indicó inicialmente, no siempre el conjunto de datos tiene un comportamiento lineal, por lo que en muchas ocasiones es necesario emplear técnicas no lineales para reconocer el tipo de patrón que estos datos tienen, en caso de tener alguno.

2.2.2. Modelo No-Lineal

El modelo no-lineal que utilizaremos en este trabajo es un híbrido entre las Redes Neuronales Artificiales (RNA) y la Lógica Difusa (LD), conocido como Sistema Neuro-Difuso (SND). En tal sentido se describirán dichos modelos. Es importante aclarar que existen otros modelos no lineales, aunque no los vayamos a describir en este trabajo.

- **Redes Neuronales Artificiales (RNA)**

Las Redes Neuronales Artificiales es una técnica de aprendizaje y procesamiento automático inspirado en el comportamiento biológico de las neuronas y de la estructura del cerebro. Entre sus características tenemos [4]:

- La capacidad de aprendizaje: adquieren el conocimiento a través de la experiencia.
- La capacidad de generalizar: pueden presentar salidas consistentes a pequeñas variaciones de las entradas.
- La capacidad de abstracción: ciertas estructuras pueden capturar la información útil de un grupo de entradas.

Un primer acercamiento a la estructura cerebral, es la descripción de una neurona típica, la cual posee el aspecto y las partes que se muestran en la figura 2.3. Esta figura no está a escala, ya que el axón alcanza un largo típico de centímetros y a veces de metros, las dendritas también son más largas y las terminales sinápticas, son más largas, numerosas y tupidas. De igual manera vemos que las dendritas son las zonas receptivas de una neurona, siendo el axón una línea de transmisión y las terminales sinápticas comunican los impulsos a otras neuronas [5].

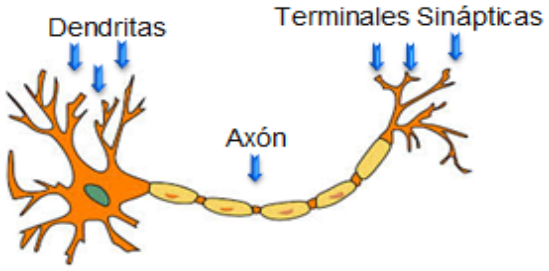


Figura 2.3: Neurona Típica (Modificado de: Valencia, 2013).

Las RNA no pretenden modelar exactamente el comportamiento fisiológico de una neurona, sino más bien algunas de sus características más relevantes. En la figura 2.4 se presenta un modelo de una neurona artificial donde se pueden apreciar los siguientes elementos [6]:

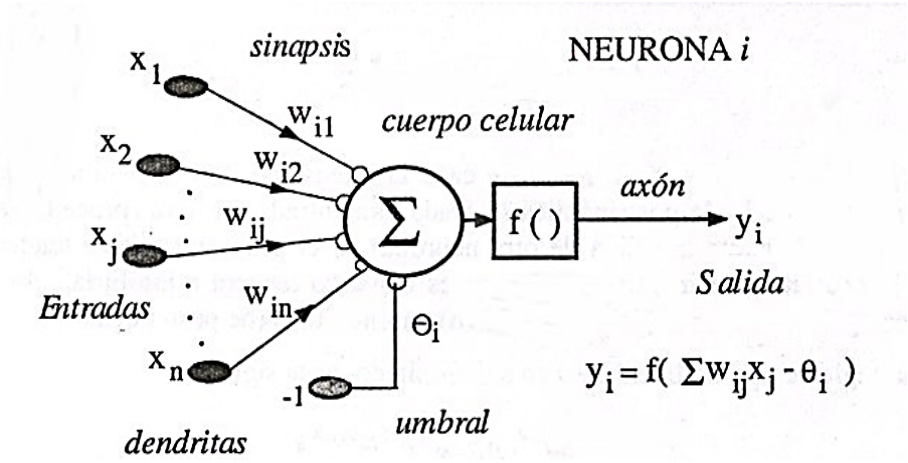


Figura 2.4: Modelo computacional de una neurona de una sola capa, que simula una neurona biológica (Tomado de: http://proton.ucting.udg.mx)

- Las n dendritas están recibiendo señales de entradas, que son valores numéricos o datos.

- Un conjunto de sinapsis o conexiones, cada una de ellas caracterizada por su fuerza o peso sináptico (w_{ij}) que depende de la información contenida en la data entrante, x_i. Estos pesos sinápticos pueden ser excitadores o inhibidores. Observando la notación, el primer índice denota la neurona hacia donde se dirige la información, y el segundo índice denota de qué dendrita procede la información.

- Una combinación lineal (Σ), que produce la suma ponderada de las entradas de acuerdo a los correspondientes pesos sinápticos de las conexiones.

- Es habitual la inclusión en el modelo un umbral o polarización (θ), cuya misión es controlar el nivel a partir del cual la neurona produce su salida.

- Una función de activación o transferencia (f), que tiene como misión limitar la amplitud de la salida generada por la neurona.

La salida de la i-ésima neurona dependerá de cada uno de estos elementos, cuya salida puede expresarse de la siguiente forma:

$$y_i = f(\Sigma w_{ij} x_j - \theta_i) \tag{2.11}$$

Como se puede detallar en la ecuación (2.11), las señales de entrada (x_1, x_2, ..., x_j) a una neurona artificial (i) pasan cada una a través de una ganancia o peso sináptico (w_{ij}) que pueden ser positivos (excitadores) o negativos (inhibidores). La neurona se activa si la entrada total ($\Sigma w_{ij} x_j$) supera un cierto umbral (θ_i). Luego se aplica una función de activación o transferencia (f) que puede ser por ejemplo, una función tipo escalón o

sigmoidea, como la tangente hiperbólica.

En la figura 2.5 se muestran algunas funciones de activación. Cabe destacar que existen varios tipos de funciones de activación y la elección de uno de estos tipos depende fuertemente del algoritmo de aprendizaje que se vaya a utilizar, la experiencia del usuario y el rango de salida que espera obtener. Normalmente los elementos de proceso de una RNA se organizan como una secuencia de capas con un determinado patrón de interconexión entre los diferentes elementos que lo forman. Uno de los rasgos que puede ayudar a definir una capa es el hecho de que todos los elementos que la forman usan la misma función de transferencia.

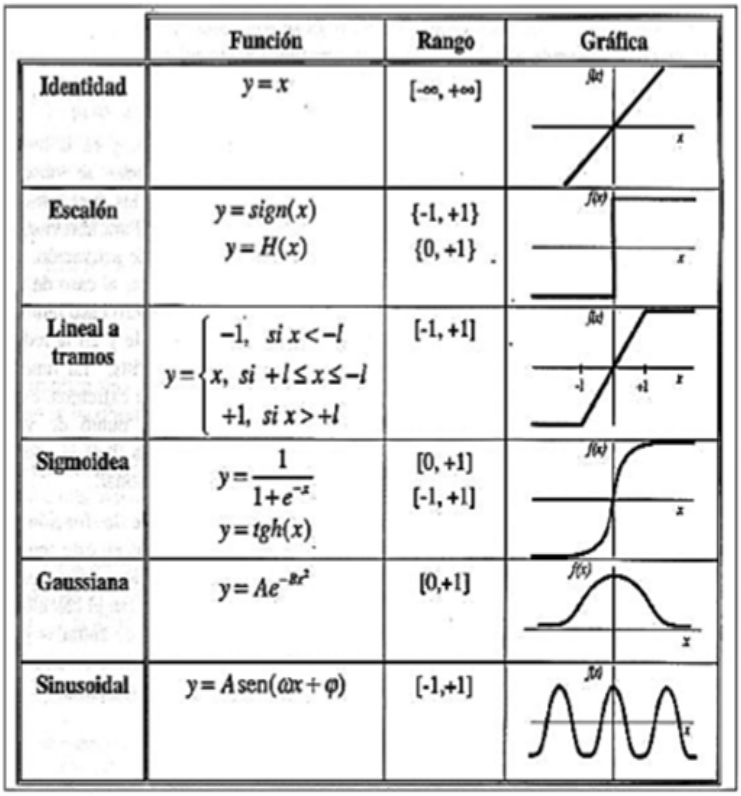


Figura 2.5: Funciones de Activación más utilizadas para las redes neuronales artificiales (Tomado de: http://redesneuralesramsesm.blogspot.com)

En muchas arquitecturas de redes neuronales se puede hacer la siguiente distribución entre capas [6]:

- Capa de entrada: Es la capa que recibe los estímulos del entorno. No suele tener asociado un mecanismo de aprendizaje, es decir, sus pesos se mantienen constantes, y su misión simplemente es la de distribuir dicha entrada al resto de los elementos de proceso que constituyen la red.

- Capa de salida: Es la capa sobre la que se forman las salidas de la red.
- Capas ocultas: Son las capas que se encuentran entre la capa de entrada y la de salida.

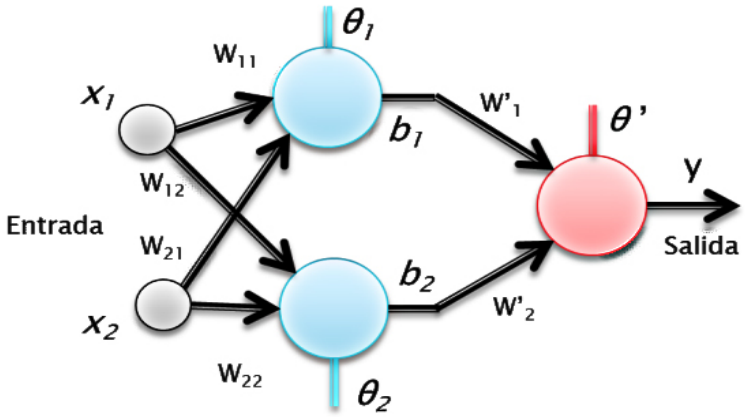


Figura 2.6: Ejemplo de Red Multicapa

Debemos tener en consideración que existe una gran relación entre la arquitectura de una RNA y el o los algoritmos de aprendizaje que pueda usar, de tal modo que diferentes arquitecturas de RNAs requieren diferentes algoritmos de aprendizaje (procedimiento mediante el cual se modifican o adaptan los pesos de la red de acuerdo al entorno en que esta se encuentra). Adicionalmente existe la función de entrenamiento que es el algoritmo general utilizado para entrenar a la red neuronal para reconocer una determinada entrada y asignarla a una salida.

Analizando la red de 3 capas, mostrada en la figura 2.6, la cual posee 2 entradas x_1 y x_2, 2 neuronas en la capa oculta con salidas b_1 y b_2, y una neurona en la capa de salida, podemos decir que:

- Para la neurona 1 de la capa oculta, con salida b_1:

$$b_1 = f(w_{11}x_1 + w_{21}x_2 + \theta_1) \tag{2.12}$$

- Para la neurona 2 de la capa oculta, con salida b_2:

$$b_2 = f(w_{12}x_1 + w_{22}x_2 + \theta_2) \tag{2.13}$$

- Para la neurona de salida:

$$y = f(w'_1 b_1 + w'_2 b_2 + \theta') \tag{2.14}$$

Conociendo la salida de cada neurona podemos obtener el resultado general de cómo se modifican todos los pesos (w) y los umbrales (θ) para cada neurona, a través de un desarrollo matemático, empleando el tipo de algoritmo de Retropropagación (algoritmo básico de aprendizaje que usa el perceptrón multicapa). El resultado general es como sigue:

- Para las neuronas en la capa oculta:

$$w_{ij}(n) = w_{ij}(n-1) + \alpha(s_j(n) - b_j(n))f'(\Sigma w_{ij}x_i + \theta_j)x_i \tag{2.15}$$

  donde n se refiere al peso modificado y $(n-1)$ al peso sin modificar, el parámetro α se define como la tasa de aprendizaje, y $s(n)$ es la salida patrón (aplica también para la neurona de salida).

$$\theta_j(n) = \theta_j(n-1) + \alpha(s_j(n) - b_j(n))f'(\Sigma w_{ij}x_i + \theta_j) \tag{2.16}$$

  donde n se refiere al umbral modificado y $(n-1)$ al umbral sin modificar, el parámetro α se define igualmente como la tasa de aprendizaje, y $s(n)$ es la salida patrón (aplica también para la neurona de salida).

- Para la neurona de salida:

$$w'_j(n) = w'_j(n-1) + \alpha(s(n) - y(n))f'(\Sigma w_j b_j + \theta')b_j \tag{2.17}$$

$$\theta'(n) = \theta'(n-1) + \alpha(s(n) - y(n))f'(\Sigma w_j b_j + \theta') \tag{2.18}$$

- **Lógica Difusa (LD)**

La lógica difusa o borrosa, es una técnica diseñada para tratar de emular el razonamiento humano basado en reglas imprecisas (los humanos razonan eficientemente con definiciones difusas). La lógica difusa es una técnica flexible que le permite a un computador clasificar información del mundo real en una escala infinita acotada por los valores falso y verdadero, creando aproximaciones matemáticas para la resolución de ciertos problemas y producir resultados exactos a partir de la observación del entorno, la formulación de reglas lógicas y de los mecanismos de toma de decisión. Las bases de la teoría en lógica difusa se establecen por el Profesor Lofti, A. Zadeh de la Universidad de Berkeley [7]. Algunos conceptos básicos de lógica difusa:

- **Conjuntos difusos**

Los conjuntos clásicos, tienen limitaciones, se define un universo de discurso que contiene a conjuntos cuyos bordes están bien definidos, un elemento puede o no pertenecer a cierto conjunto, algo es verdadero o falso, no se definen situaciones intermedias. Los conjuntos difusos son una extensión de los clásicos, donde se añade una función de pertenencia, definida ésta como un número real entre 0 y 1. Así se introduce el concepto de conjunto o subconjunto borroso y se le asocia a un determinado valor lingüístico, definido por una palabra o etiqueta lingüística, donde ésta es el nombre del conjunto o subconjunto. Un conjunto difuso se define matemáticamente como [7]:

$$A = \{(x, \mu_{A(x)}/x \epsilon U)\} \tag{2.19}$$

donde por cada conjunto difuso, A, se define una función de pertenencia o membresía $\mu_{A(x)}$, la cual indica el grado en que la variable x está incluida en el concepto representado por la etiqueta A ($0 \leq \mu_{A(x)} \leq 1$), si esta función toma el valor 0 significa que tal valor de x no está incluido en A y si toma el valor 1 el correspondiente valor de x está absolutamente incluido en A, U es una colección de objetos (Universo de Discurso), expresados en forma genérica por la variable x, entonces, un conjunto difuso A en U, se define como un conjunto de pares ordenados.

Para el caso en el que U es continuo, un conjunto difuso A se puede representar como:

$$A = \int_x \frac{\mu_{A(X)}}{x} \tag{2.20}$$

Ahora, para el caso en que U es discreto, el conjunto difuso A se puede representar como:

$$A = \sum_{i=1}^{N} \frac{\mu_{A(x_i)}}{x_i} \tag{2.21}$$

- **Funciones de pertenencia o membresía**

Las funciones de membresía representan el grado de pertenencia de un elemento a un subconjunto definido por una etiqueta. Para cualquier función de pertenencia en general es recomendado el uso de funciones simples, buscando simplificar los cálculos matemáticos y no perder exactitud, ya que precisamente se está definiendo un concepto difuso. Existe una gran variedad de formas para las funciones de pertenencia, la escogencia de la función de pertenencia va a depender del tipo de dato que se esté analizando. Entre las funciones de pertenencia más comunes tenemos, la triangular, la gamma, la S, la trapezoidal, la singleton y la gaussiana, cuyas representaciones se pueden detallar en las siguientes figuras [7]:

$$\mu_A(x) = \begin{cases} 0 & \text{si} \quad x \leq a \\ (x-a)/(m-a) & \text{si} \quad x\epsilon(a,m] \\ (b-x)/(b-m) & \text{si} \quad x\epsilon(m,b) \\ 0 & \text{si} \quad x \geq b \end{cases}$$

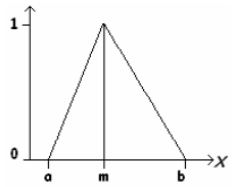


Figura 2.7: Función de pertenencia Triangular

$$\mu_A(x) = \begin{cases} 0 & \text{si} \quad x \leq a \\ 1 - e^{-k(x-a)^2} & \text{si} \quad x > a \end{cases}$$

$$\mu_A(x) = \begin{cases} 0 & \text{si} \quad x \leq a \\ \dfrac{k(x-a)^2}{1+k(m-a)^2} & \text{si} \quad x > a \end{cases}$$

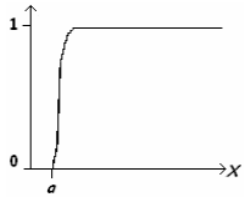


Figura 2.8: Función de pertenencia gamma

$$\mu_A(x) = \begin{cases} 0 & \text{si} \quad x \leq a \\ 2[(x-a)/(b-a)]^2 & \text{si} \quad x\epsilon(a,m] \\ 1 - 2[(x-a)/(b-a)]^2 & \text{si} \quad x\epsilon(m,b) \\ 1 & \text{si} \quad x \geq b \end{cases}$$

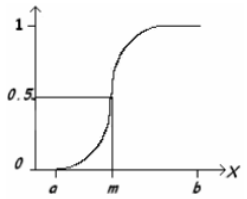


Figura 2.9: Función de pertenencia S

$$\mu_A(x) = \begin{cases} 0 & \text{si} \quad (x \leq a) o (x \geq d) \\ (x-a)/(b-a) & \text{si} \quad x\epsilon(a,b] \\ 1 & \text{si} \quad x\epsilon(b,c) \\ (d-x)/(d-c) & \text{si} \quad x\epsilon[c,d) \end{cases}$$

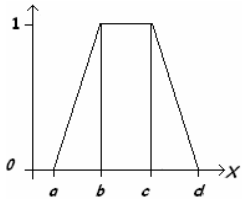


Figura 2.10: Función de pertenencia trapezoidal

$$\mu_A(x) = \begin{cases} 1 & x = a \\ 0 & x \neq a \end{cases}$$

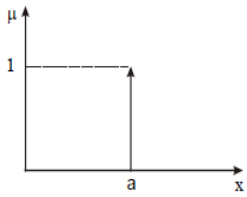


Figura 2.11: Función de pertenencia Singleton

$$\mu_A(x) = e^{-k(x-m)^2}$$

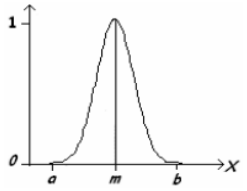


Figura 2.12: Función de pertenencia Gaussiana

- **Operaciones con conjuntos difusos**

A los subconjuntos difusos se les pueden aplicar determinados operadores lógicos o bien se pueden realizar operaciones entre ellos. Al aplicar un operador lógico sobre un solo conjunto difuso, se obtendrá otro conjunto difuso, lo mismo sucede cuando se realiza una operación entre conjuntos difusos. Para ilustrar las operaciones con conjuntos difusos, consideremos dos conjuntos difusos A y B con funciones de pertenencia $\mu_A(x)$ y $\mu_B(x)$ respectivamente (ver figura 2.13) [7]:

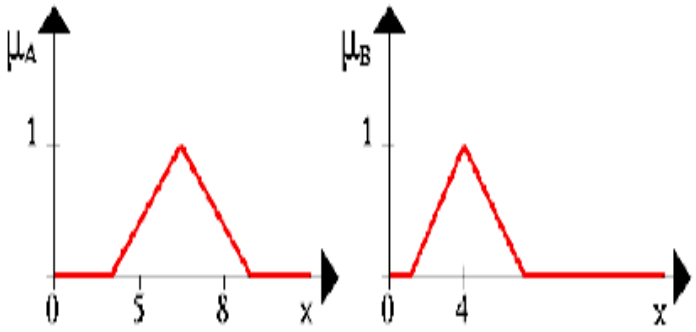


Figura 2.13: Conjuntos difusos A y B con sus funciones de pertenencia $\mu_A(x)$ y $\mu_B(x)$.

Veamos algunas operaciones básicas con los dos conjuntos difusos [8]:

- Complemento

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x)$$

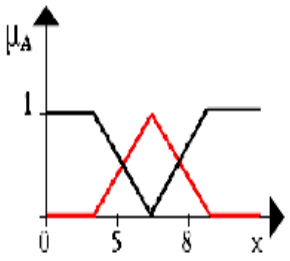


Figura 2.14: Complemento de A

- Unión, operador lógico OR (T-COnorma) [8]

Operador lógico OR de Zadeh (max)

$$\mu_{A\cup B}(x) = max[\mu_A(x), \mu_B(x)]$$

Operador lógico OR de Lukasiewicz (min)

$$\mu_{A\cup B}(x) = min[1, \mu_A(x) + \mu_B(x)]$$

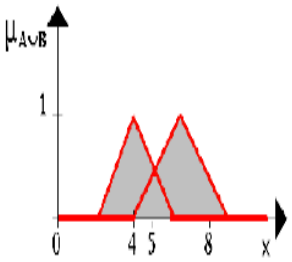


Figura 2.15: Unión entre A y B

- Intersección, operador lógico AND (T-norma) [8]

Operador lógico AND de Zadeh (min)

$$\mu_{A\cap B}(x) = min[\mu_A(x), \mu_B(x)]$$

Operador lógico AND del producto

$$\mu_{A\cap B}(x) = \mu_A(x).\mu_B(x) = max[0, \mu_A(x) + \mu_B(x) - 1]$$

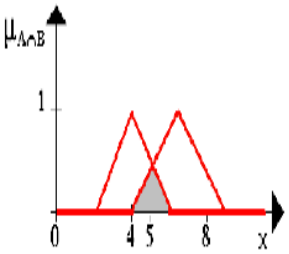


Figura 2.16: Intersección entre A y B

- Inclusión $A \subseteq B$

$$\mu_{\bar{A}}(x) \leq \mu_B(x) \tag{2.22}$$

- Norma

$$\mu_{NORMA(A)}(x) = \frac{\mu_A(x)}{max[\mu_A(x)]} \tag{2.23}$$

- Concentración

$$\mu_{con(A)}(x) = (\mu_A(x))^2 \tag{2.24}$$

- Dilatación

$$\mu_{Dila(A)}(x) = (\mu_A(x))^{0,5} \tag{2.25}$$

- Igualdad

$$\mu_A(x) = \mu_B(x) \tag{2.26}$$

- **Fusificación**

Operación que se realiza en todo instante de tiempo, y es la puerta de entrada al sistema de inferencia difusa. La fusificación es un procedimiento matemático en el que un elemento del universo del discurso (entrada definida, no difusa) intersecta las funciones de pertenencia convirtiéndose en las entradas difusas. Mediante este procedimiento, el fusificador establece una relación entre los puntos de entrada no difusos y sus correspondientes conjuntos difusos en el universo de discurso, U [9].

- **Reglas Difusas**

Son reglas que combinan uno o más conjuntos borrosos de entrada llamados antecedentes o premisas y le asocian un conjunto borroso de salida llamado consecuente o consecuencia. A estas reglas se les llama reglas borrosas o difusas. Son afirmaciones del tipo SI-ENTONCES. Los conjuntos borrosos del antecedente se asocian mediante operaciones lógicas borrosas AND, OR, etc [7]. Las reglas borrosas son proposiciones que permiten expresar el conocimiento que se dispone sobre la relación entre antecedentes y consecuentes.

Las reglas difusas más utilizadas son conocidas como las reglas difusas de Mamdani o Takagi-Sugeno.

- Reglas difusas tipo Mamdani, en este caso la salida es lingüística. La regla se puede expresar como:

$$\text{IF } x_1 \textbf{ is A AND } x_2 \textbf{ is B AND } x_3 \textbf{ is C}$$
$$\text{THEN } u_1 \textbf{ is D, } u_2 \textbf{ is E}$$

  donde x_1, x_2 y x_3 son las variables de entrada, A, B y C son funciones de membresía de entrada, u_1 y u_2 son las acciones de control en sentido genérico son todavía variables lingüísticas (no toman valores numéricos), D y E son las funciones de membresía de la salida, y AND es el operador lógico difuso. La primera parte de la sentencia (IF x_1 is A AND x_2 is B AND x_3 is C) es el antecedente y la restante es el consecuente [7].

- Reglas difusas tipo Takagi-Sugeno, donde la salida puede ser una función o una constante numérica. La regla en este caso en que la salida es una función se puede expresar como:

$$\text{IF } x_1 \textbf{ is A AND } x_2 \textbf{ is B AND } x_3 \textbf{ is C}$$
$$\text{THEN } u_1 = f(x_1, x_2, x_3), u_2 = g(x_1, x_2, x_3)$$

  la primera parte de la sentencia (antecedente) se escribe como en la regla tipo Mamdani pero el consecuente son las funciones.

- **Inferencia Difusa**

Las reglas difusas representan el conocimiento y la estrategia de control, pero cuando se asigna información específica a las variables de entrada en el antecedente, la inferencia difusa es necesaria para calcular el resultado de las variables de salida del consecuente, este resultado es en términos difusos, es decir que se obtiene un conjunto difuso de salida de cada regla, que posteriormente junto con las demás salidas de reglas se obtendrá la salida del sistema [7].

- **Agregado**

Cuando se evalúan las reglas difusas se obtienen tantos conjuntos difusos como reglas existan, entonces es necesario agrupar estos conjuntos, esta etapa se conoce como agregado y existen varios criterios para realizar este paso. Un criterio muy utilizado es el de agrupar los conjuntos inferidos mediante la operación max [7].

- **Defusificación**

La defusificación (defuzzyfication) es un proceso matemático usado para convertir un conjunto difuso en un número real. El sistema de inferencia difusa obtiene una conclusión a partir de la información de la entrada, pero es en términos difusos. Esta conclusión o salida difusa es obtenida por la etapa de inferencia borrosa, la cual genera un conjunto borroso pero el dato de salida del sistema debe ser un número real y debe ser representativo de todo el conjunto obtenido en la etapa de agregado, es por eso que existen diferentes métodos de defusificación y arrojan resultados distintos, el más común es el centroide. Con el método de defusificación del centroide se transforma la salida difusa en un número real el cual es la coordenada equis (x) del centro de gravedad de tal conjunto difuso de salida [10]

$$y_d = \frac{\int_S y\mu_Y(y)dy}{\int_S \mu_Y(y)dy} \tag{2.27}$$

donde μ_Y es la función de pertenencia del conjunto de salida Y, cuya variable de salida es y, S es el dominio o rango de integración. Este método trae mucha carga computacional, por lo que se emplean en general otros esquemas con menor carga.

Otro defusificador muy utilizado es el centro de área (COA, center of area) también llamado altura, el centro de gravedad es aproximado por el centro de gravedad de un arreglo de masas puntuales, las cuales son el centro de gravedad de cada conjunto de salida correspondiente a cada regla, con masa igual al grado de pertenencia en ese punto de su centro de gravedad. Si se le llama δ_1 al centro de gravedad del conjunto difuso de

salida B_l de la l-ésima regla, el centro de gravedad queda determinado por [10]:

$$y_d = \frac{\sum_{l=1}^{R} \delta_l \mu_{B_l(\delta l)}}{\sum_{l=1}^{R} \mu_{B_l(\delta l)}} \tag{2.28}$$

donde R es el número de reglas.

De manera resumida las figuras 2.17 y 2.18 muestran los pasos llevados a cabo en un sistema de inferencia difusa según el modelo utilizado [9].

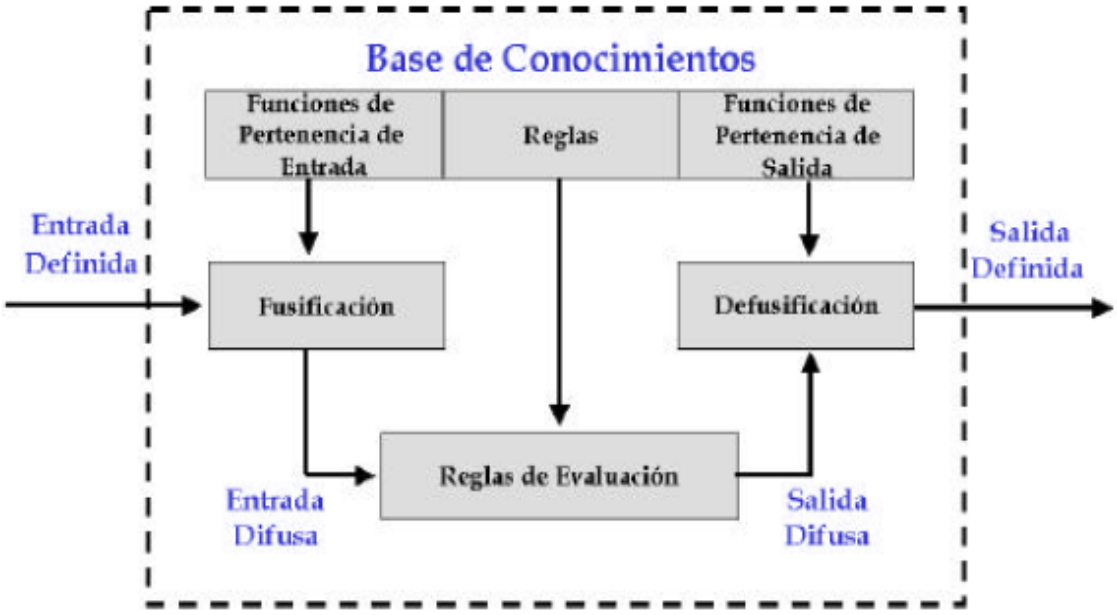


Figura 2.17: Sistema de inferencia difusa de Mamdani (Modificado de: MatLab Fuzzy Inference Systems)

Es importante resaltar que la fusificación y la defusificación son el nexo del sistema difuso con el mundo real, donde las reglas difusas definen la estrategia de conocimiento y las operaciones que se realizan entre los conjuntos difusos.

- **Sistemas Neuro-Difusos (SND)**

Como se ha mencionado, la lógica difusa y las redes neuronales artificiales tienen propiedades particulares que las hacen adecuadas para ciertos problemas. Por ejemplo, mientras las redes neuronales ofrecen ventajas como el aprendizaje, adaptación, tolerancia a fallas, paralelismo y generalización, no son buenas para explicar cómo han alcanzado

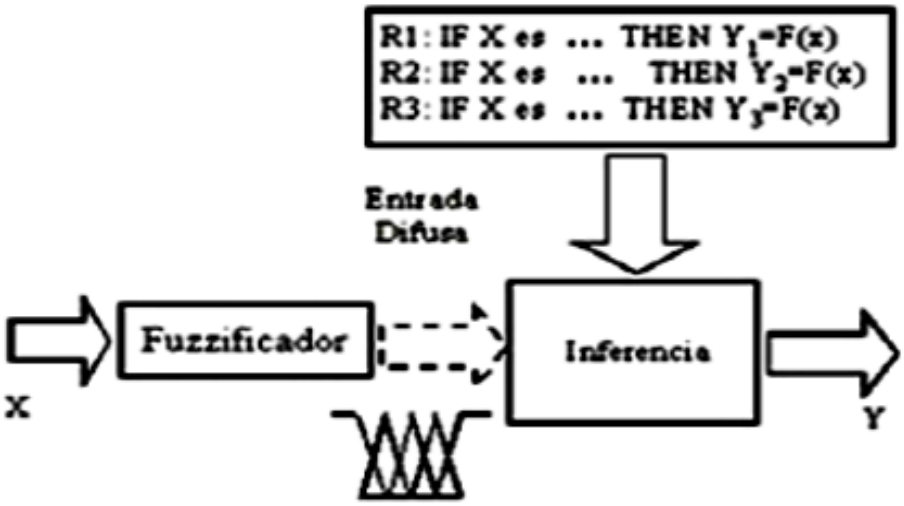


Figura 2.18: Sistema de inferencia difusa Takagi-Sugeno (Modificado de: MatLab Sugeno-Type Fuzzy Inference)

sus decisiones. En cambio los sistemas difusos, los cuales razonan con la información imprecisa a través de un mecanismo de inferencia bajo incertidumbre lingüística, son buenos explicando sus decisiones pero no pueden adquirir automáticamente las reglas que se usan para tomarlas. Uniendo la capacidad de aprendizaje de las RNAs con el poder de interpretación lingüística de la lógica difusa, se crean los sistemas neuro-difusos, entre los cuales podemos citar:

- ANFIS (Adaptive Neuro Fuzzy Inference System) es un método que permite sintonizar o crear la base de reglas de un sistema difuso utilizando el algoritmo de entrenamiento de retropropagación a partir de la recopilación de datos de un proceso. Su arquitectura es funcionalmente equivalente a una base de reglas de tipo Sugeno [9].

- FSOM (Fuzzy Self-Organizing Maps) consiste en un sistema difuso optimizado a partir de la técnica de los mapas auto-organizados de Kohonen [9].

- NEFCLASS (Neuro Fuzzy Classification) está basado en la estructura del perceptrón multicapa cuyos pesos son modelados por conjuntos difusos y además la activación, la salida y las funciones de propagación son adaptadas apropiadamente. Así, se preserva la estructura de una red neuronal, pero se permite la interpretación del sistema resultante por el sistema difuso asociado, es decir, la RNA deja de ser una caja negra [11].

- FuzzyTech es un software que propone un método de desarrollo de sistemas neuro-difuso similar a ANFIS [9].

Los sistemas de inferencia difusa son conocidos como sistemas basados en reglas difusas, modelos difusos, memorias asociativas difusas (FAM), o controladores difusos cuando son usados como controladores. Un sistema de inferencia difusa básicamente está compuesta por cinco bloques (ver figura 2.19), los cuales se pueden describir como [11]:

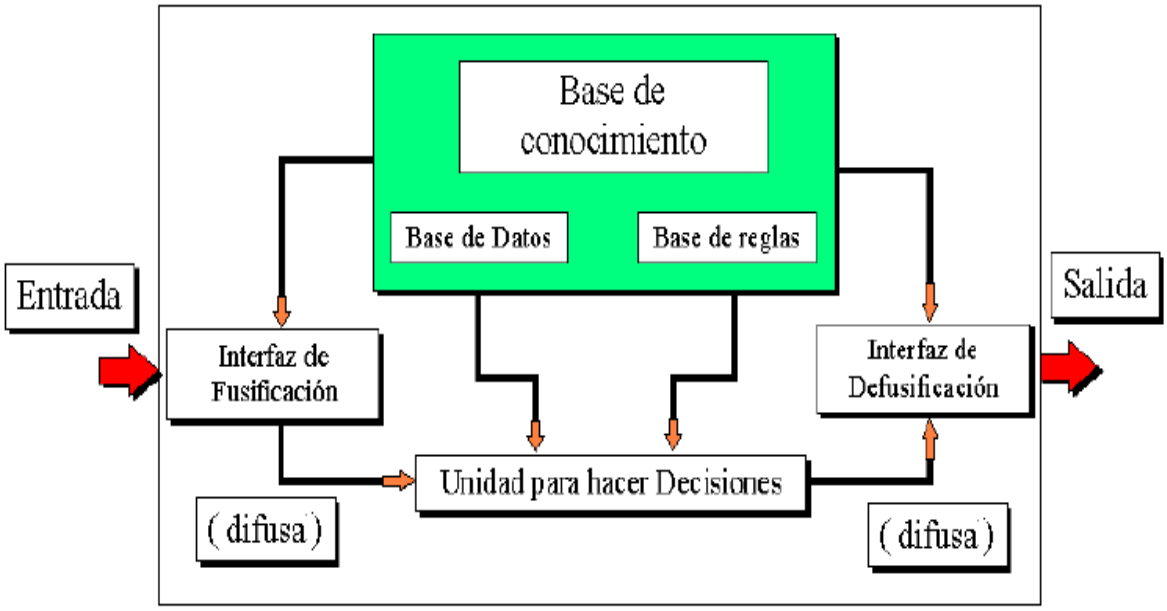


Figura 2.19: Sistema de Inferencia Difusa (Modificado de: R. Jang [12])

- Una base de regla conteniendo un número de reglas if-then difusas.
- Una base de datos la cual define las funciones de membresía de los conjuntos difusos usados en las reglas difusas.
- Una unidad de toma de decisión la cual ejecuta las operaciones de inferencias sobre las reglas.
- Una interface de fusificación la cual transforma las entradas nítidas en grados de igualdad con los valores lingüísticos.
- Una interface de defusificación la cual transforma los resultados difusos de la inferencia en una salida nítida.

El sistema de inferencia difusa que estaremos trabajando es la clase de red adaptativa ANFIS, bajo la consideración de que tiene dos entradas x y y, una salida f, y la base de reglas if-then según Takagi-Sugeno. Comencemos enunciando la base de reglas de inferencia Takagi-Sugeno como sigue (ver ecuaciones 2.29 y 2.30, y figura 2.20):

- Regla 1: Si x es A_1 and y es B_1, entonces

$$f_1 = p_1 x + q_1 y + r_1 \tag{2.29}$$

- Regla 2: Si x es A_2 and y es B_2, entonces

$$f_2 = p_2 x + q_2 + r_2 \tag{2.30}$$

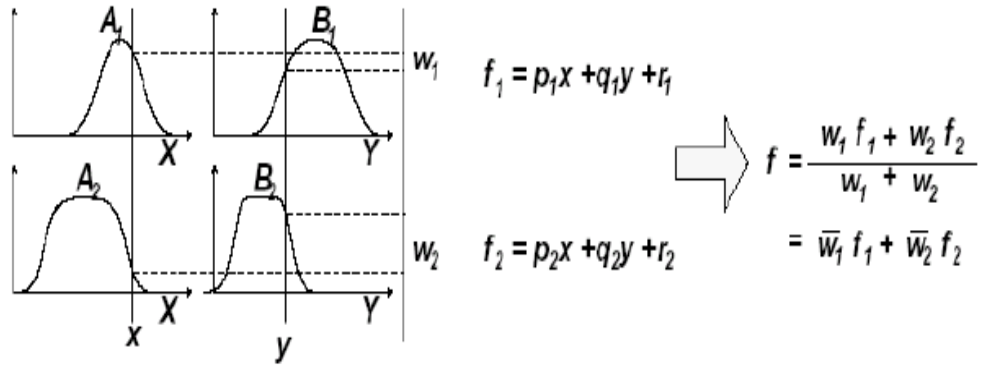


Figura 2.20: Razonamiento Difuso Takagi-Sugeno (Modificado de: R. Jang [12])

donde por cada conjunto difuso A_1, A_2, B_1 y B_2 se define una función de pertenencia $\mu_{A_1}(x)$, $\mu_{A_2}(x)$, $\mu_{B_1}(y)$ y $\mu_{B_2}(y)$, respectivamente.

La arquitectura de un sistema ANFIS se muestra en la figura 2.21, donde explicaremos brevemente las funciones que cumplen cada una de las capas que componen dicho sistema [12]:

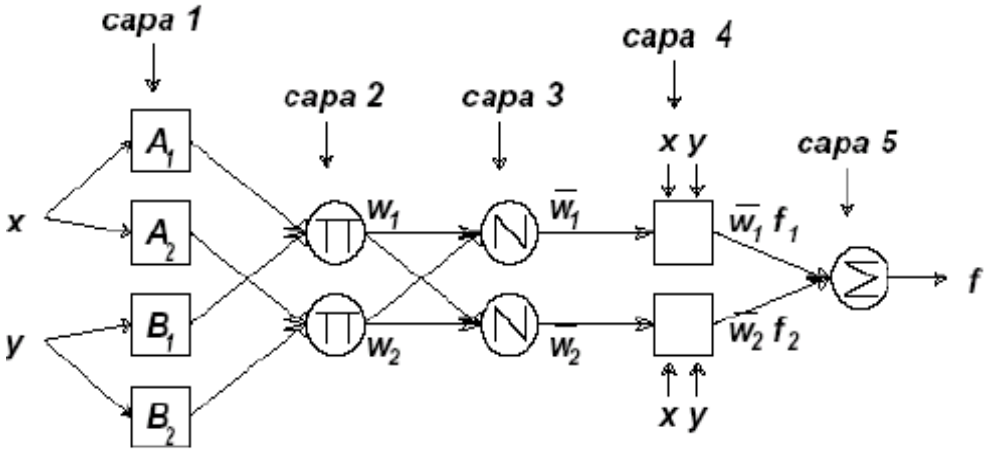


Figura 2.21: Arquitectura de un sistema ANFIS (Modificado de: R. Jang [12])

- **Capa 1**: En cada nodo i de esta capa (nodo cuadrado) se implementa una regla difusa a través de una función de pertenencia.

$$O_i^1 = \mu_{A_i}(x) \tag{2.31}$$

La salida es el grado de pertenencia para la cual la variable de entrada x satisface el término lingüístico μ_{A_i} asociado al nodo i. Los parámetros de dichas funciones (constantes que las caracterizan) se conocen como parámetros del antecedente.

- **Capa 2**: En cada nodo i de esta capa (nodo circular) etiquetado por $\prod$, se multiplican las señales entrantes, usando cualquier T-norma para modelar la operación lógica AND.

$$O_i^2 = w_i = \mu_{A_i}(x).\mu_{B_i}(y) \tag{2.32}$$

con $i = 1, 2$.

- **Capa 3**: En cada nodo i de esta capa (nodo circular) etiquetado por N, se normalizan las funciones de pertenencias para actualizar las entradas.

$$O_i^3 = \bar{w}_i = \frac{w_i}{w_1 + w_2} \tag{2.33}$$

con $i = 1, 2$.

- **Capa 4**: En cada nodo i de esta capa (nodo cuadrado) se realiza el producto de la salida de la capa 3 $\bar{w}_i$ por una función de nodo f_i.

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i(p_i x + q_i y + r_i) \tag{2.34}$$

con $i = 1, 2.$, donde p, q, r forman un conjunto de parámetros conocidos como parámetros del consecuente, y se van ajustando durante el proceso de entrenamiento.

- **Capa 5**: El único nodo en esta capa (nodo circular) etiquetado por Σ, calcula la salida final del sistema realizando la sumatoria de todas las señales de salida de la capa 4.

$$O^5 = f = \bar{w}_1 f_1 + \bar{w}_2 f_2 \tag{2.35}$$

Mientras está en el proceso de aprendizaje, ANFIS va a ir ajustándose para ir en cada iteración asemejándose a la salida real y así minimizar el error, para lo cual se van ajustando los pesos de los parámetros del antecedente y del consecuente [12].

2.3. Plataforma de Cálculo

La idea principal de este trabajo es desarrollar una interfaz gráfica para el procesamiento automatizado de conjunto de datos, sin la necesidad de recurrir a un método manual o el empleo de hojas de cálculo tradicionales. Para esta tarea desarrollaremos un software en la plataforma de cálculo MatLab (Matrix Laboratory). MatLab permite programar de forma eficiente y sencilla, basado en cálculo vectorial y matricial. La idea es generar un sistema de uso simple, amigable al usuario, por lo que aprovecharemos algunas de las herramientas de MatLab, como por ejemplo la interfaz gráfica de usuario que esta plataforma posee.

2.3.1. Interfaz Gráfica de Usuario

MatLab posee una interfaz gráfica de usuario conocida como GUI (Graphical User Interface) sencillo de manejar y adecuada para este trabajo. Las GUIs utilizan un conjunto de imágenes y objetos gráficos que permiten al usuario realizar tareas y crear controles en forma interactiva. A través de llamadas (callbacks) se puede hacer que los controles realicen las tareas deseadas por el usuario.

Una manera de crear una GUI en MatLab es escribiendo GUIDE (Graphical User Interface Development Environment) desde el Command Window (ventana de comando), la cual genera un cuadro de diálogo GUIDE Quick Start, como se muestra en la figura 2.22. Para comenzar se recomienda usar la plantilla predeterminada vacía de GUI (Blank GUI (Default)).

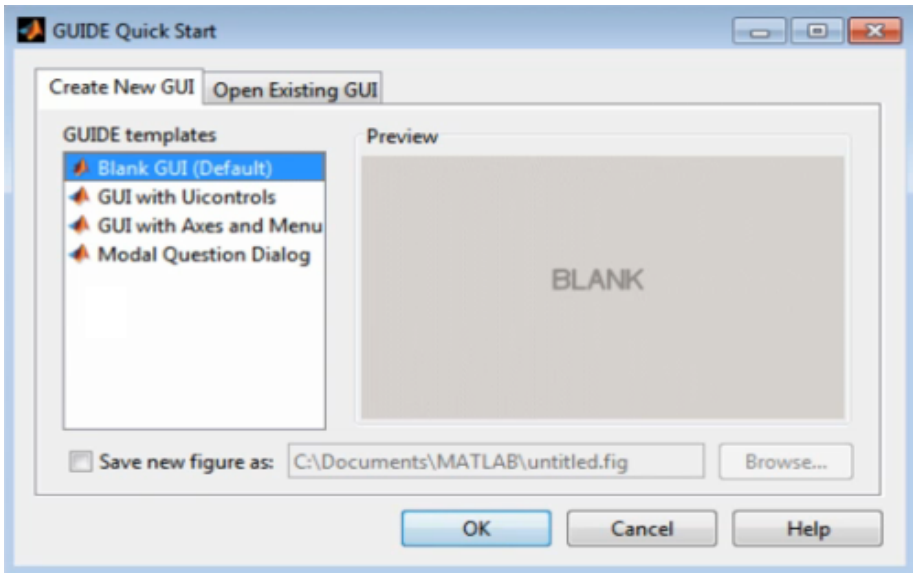


Figura 2.22: Ventana de inicio de GUI (Tomado de: MatLab GUIDE Quick Start)

Toda aplicación GUIDE trabaja con dos tipos de archivo:

- Un archivo **.fig** donde está contenido todo el diseño gráfico y los controles utilizados en la GUI.
- Un archivo **.m** donde se almacena el código fuente de la aplicación. Contiene la codificación de inicialización y funciones callback que controlan la GUI.

El entorno gráfico de trabajo para diseñar una GUI posee una barra de título, una de menús, una de herramientas, y una de controles, una ventana de figuras (área de diseño donde se colocan los controles a utilizar), una barra Tag y dos barras de posiciones, como podemos ver en la figura 2.23. La barra de herramientas de una GUIDE cuenta con varios íconos (ver figura 2.24). Una breve descripción sobre la barra de controles de GUIDE se muestra en la figura 2.25.

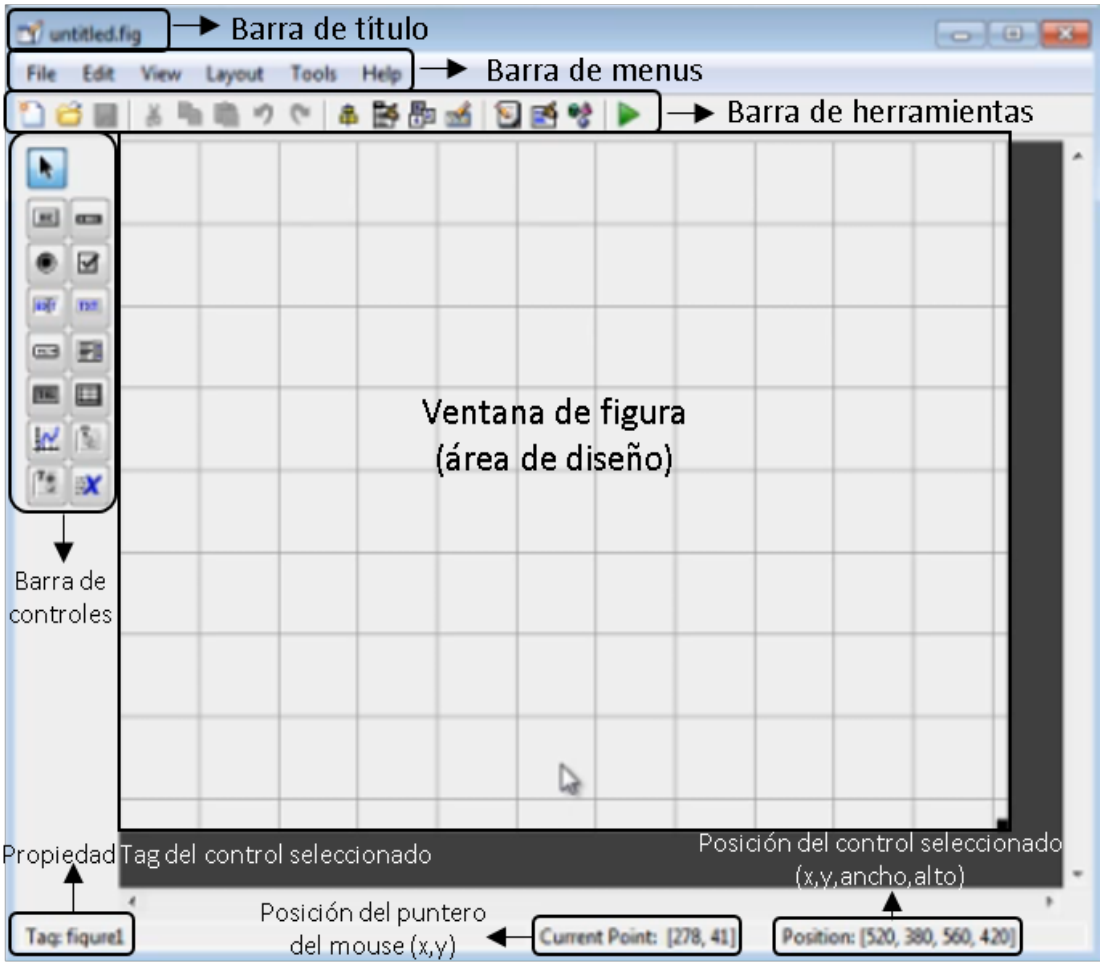


Figura 2.23: Entorno gráfico de una GUIDE (Modificado de: MatLab GUIDE).

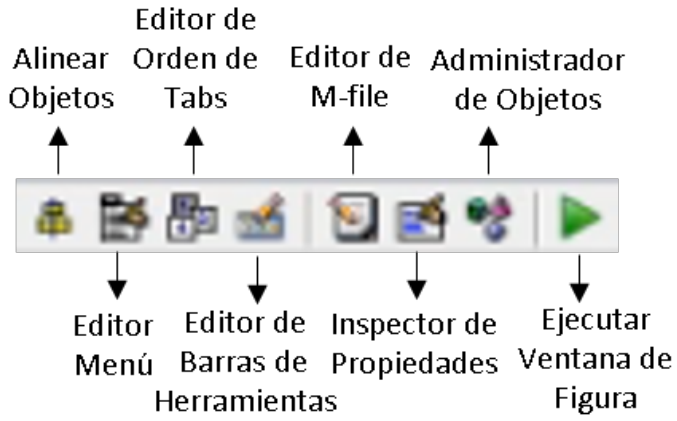


Figura 2.24: Barra de herramientas en GUIDE (Modificado de: MatLab GUIDE).

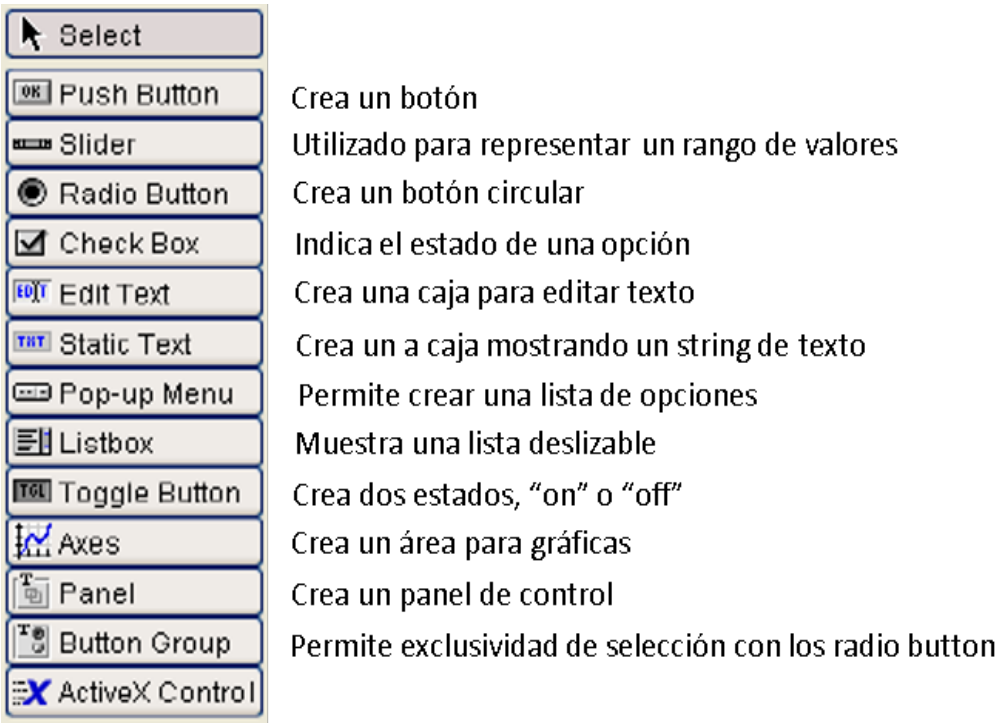


Figura 2.25: Barra de controles de GUIDE (Modificado de: MatLab GUIDE).

Es importante señalar que cada control de GUIDE posee propiedades que pueden modificarse a través de la herramienta Property Inspector. Por ejemplo, la propiedad Tag permite referenciar al control seleccionado dentro del código fuente; es decir, todos los Tags de una GUIDE cuentan con una estructura llamada handles a través de la cual se hace referencia al control en el código fuente.

2.3.2. ANFIS de MatLab

ANFIS deriva de Sistema de Inferencia Neuro-Difuso Adaptativo, la cual usa un conjunto de datos de entrada/salida, donde el módulo de ANFIS combina un sistema de inferencia borrosa (FIS, por sus siglas en inglés) con un sistema de redes neuronales. La interfaz gráfica del módulo de ANFIS, de MatLab, se muestra en la figura 2.26. Para comenzar a trabajar con el módulo de ANFIS debemos conocer las funciones que cumplen cada área de trabajo, como podemos exponer a continuación [13]:

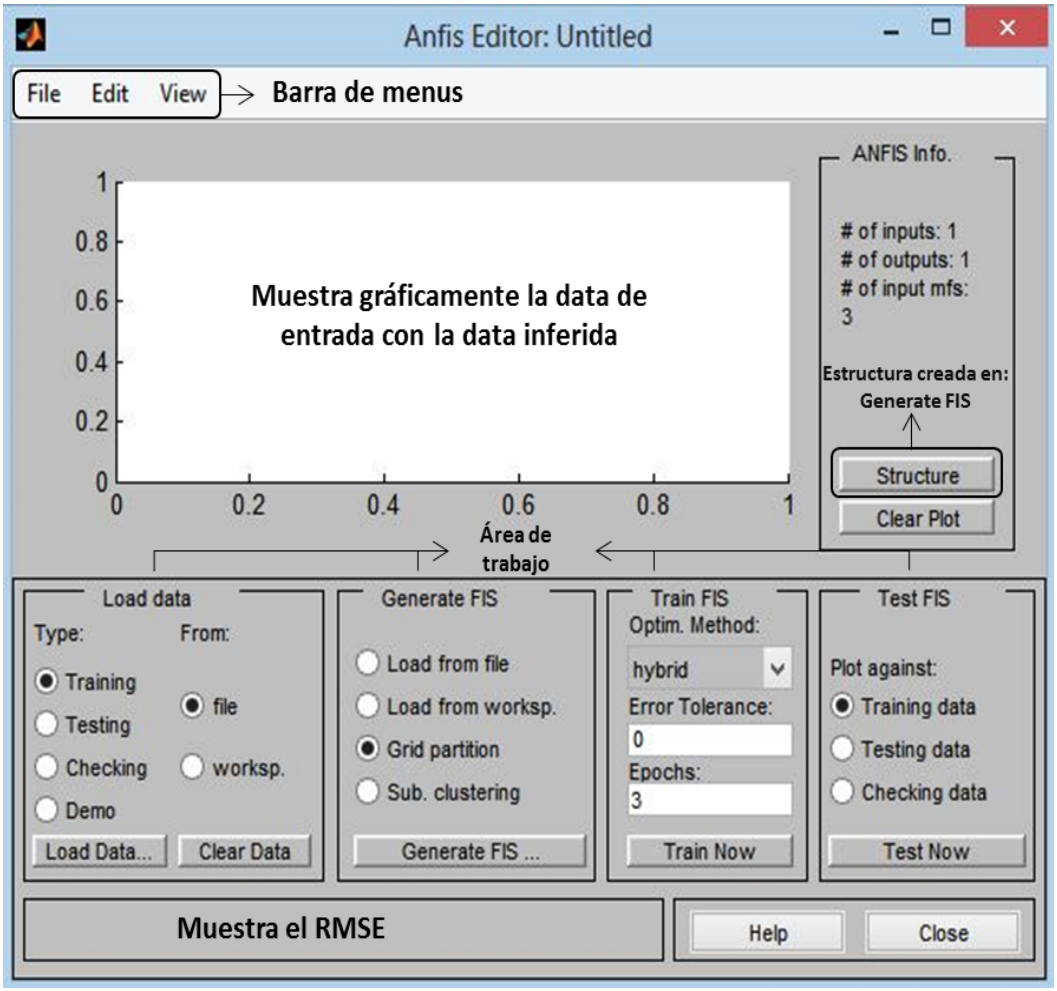


Figura 2.26: Interfaz gráfica de ANFIS. (Modificado de: MatLab ANFISEDIT).

- **Load data**
  En esta sección se puede cargar un conjunto de datos que pueden provenir de un archivo ubicado en cualquier lugar (file) o en el espacio de trabajo de MatLab (worksp). Los datos seleccionados pueden usarse para: entrenamiento (Training), prueba (Testing), chequeo (Checking) o demostración (Demo).

- **Generate FIS**
  Cargados los datos debemos especificar un modelo de estructura inicial de la FIS. Para ello podemos cargar una estructura tipo Sugeno previamente guardada en un archivo (Load from file) o en el espacio de trabajo de MatLab worksp (Load from worksp.). Sin embargo, se puede generar el modelo inicial FIS al elegir algunas de las técnicas:

  - (Grid partition) en donde se puede colocar el número de reglas (Number of MFs) y elegir el tipo de función (MF Type) como entrada de la FIS, y por último el tipo de salida (MF Type) de la FIS. Técnica aplicada para pocas variables de entrada.
  - (Sub. clustering) en donde se aplica la agrupación de sustracción de los datos. Técnica aplicada para varias variables.

  Para ver una representación gráfica de la estructura del modelo inicial hacemos clic en Structure ubicado en ANFIS Info (ver fig. 2.26).

- **Train FIS**
  Para comenzar a entrenar el FIS debemos elegir el método de optimización (Optim. Method), donde tenemos dos opciones: el hybrid (combinación de mínimos cuadrados) o el backpropaga (método de propagación hacia atrás). Luego se introduce el error de tolerancia (Error Tolerance), el cual puede ser por ejemplo 0.01 para un mejor ajuste y por último colocamos la cantidad de iteraciones (Epochs) que deseamos tener durante el entrenamiento.

  Cuando comience el entrenamiento las iteraciones y el error de entrenamiento se van registrando en el espacio de trabajo donde se indica RMSE de la figura 2.26.

- **Test FIS**
  Finalmente en esta área de trabajo podemos probar el desempeño del FIS, dependiendo del conjunto de datos cargados, eligiendo los datos para entrenamiento (Training data), los datos para prueba (Testing data), o los datos para chequeo (Checking data). Una vez que se haga clic en probar ahora (Test now), aparecerá gráficamente los datos originales frente a los datos de salida de la FIS.

Capítulo 3

Metodología

En este capítulo fundamentamos el desarrollo del sistema automatizado del proceso de minería de datos, que hemos desarrollado, el cual permitirá al usuario preparar y procesar los datos de algún fenómeno físico, visualizar y comparar gráficamente en dos dimensiones los resultados obtenidos, en escalas normales y logarítmicas, obtener valores estadísticos como la dispersión y correlación de los datos, la ecuación de regresión lineal asociada a dichos datos la cual permitirá a través de sus coeficientes generar valores inferidos (y_{inf}) para su posterior comparación con los datos reales (y) y así obtener valores como el RMSE, con los cuales es posible saber si los ajustes han sido los adecuados. Es importante resaltar, que durante este proceso el usuario podrá utilizar algunas herramientas de los modelos lineal y logarítmico.

La interfaz gráfica desarrollada, también ofrece al usuario la posibilidad de procesar los datos empleando el Sistema Neuro-Difuso ANFIS como modelo no lineal. En esta etapa el usuario dispone de las herramientas y bondades que posee el módulo de ANFIS de MatLab, adicionales a las ventajas que posee la interfaz, para facilitar el proceso.

Para el diseño del sistema automatizado utilizamos la interfaz gráfica de usuario (GUIDE) de la plataforma de MatLab versión 8.5 R2015a previamente instalado en un procesador Intel(R) Core(TM) i5, con 8GB de memoria RAM, disponible en el Laboratorio de Física Teórica del Sólido, en la Escuela de Física de la UCV, donde se desarrollaron las respectivas pruebas con trabajos anteriores donde el procedimiento no era automatizado.

3.1. Automatización

Uno de los aspectos importantes de este software es el diseño y la forma en cómo se presenta cada uno de los elementos o componentes, refiriéndonos a botones, cajas de textos, paneles de control, gráficos, entre otros, así como las diferentes interfaces con las cuales el usuario podrá interaccionar. Cada elemento o componente de este software está

programado con funciones propias de MatLab y otras funciones que fueron creadas para cumplir con ciertos objetivos. Para ingresar al software el usuario debe escribir en la ventana de comandos de MatLab la palabra:

>> **guide**

y al presionar la tecla intro o enter aparecerá la venta de inicio de GUI (ver figura 2.22) donde deberá señalar Open Existing GUI, para luego presionando el botón Browse podrá buscar la dirección donde se aloja la aplicación con el nombre de Interfaz. Otra manera de ingresar a la aplicación es escribiendo en la ventana de comandos:

>> **Interfaz**

siempre y cuando esté ubicado en la carpeta donde se aloja dicho programa. Con esta instrucción se ejecutará la ventana principal de la aplicación (ver figura 3.1), en donde se podrá apreciar partes fundamentales del proceso de minería de datos. En esta hemos definido dos paneles de control, uno para los modelos lineal y logarítmico, y otro para el modelo no-lineal, cuatro diagramas de dispersión en escalas lineal y logarítmica, y una tabla con valores estadísticos.

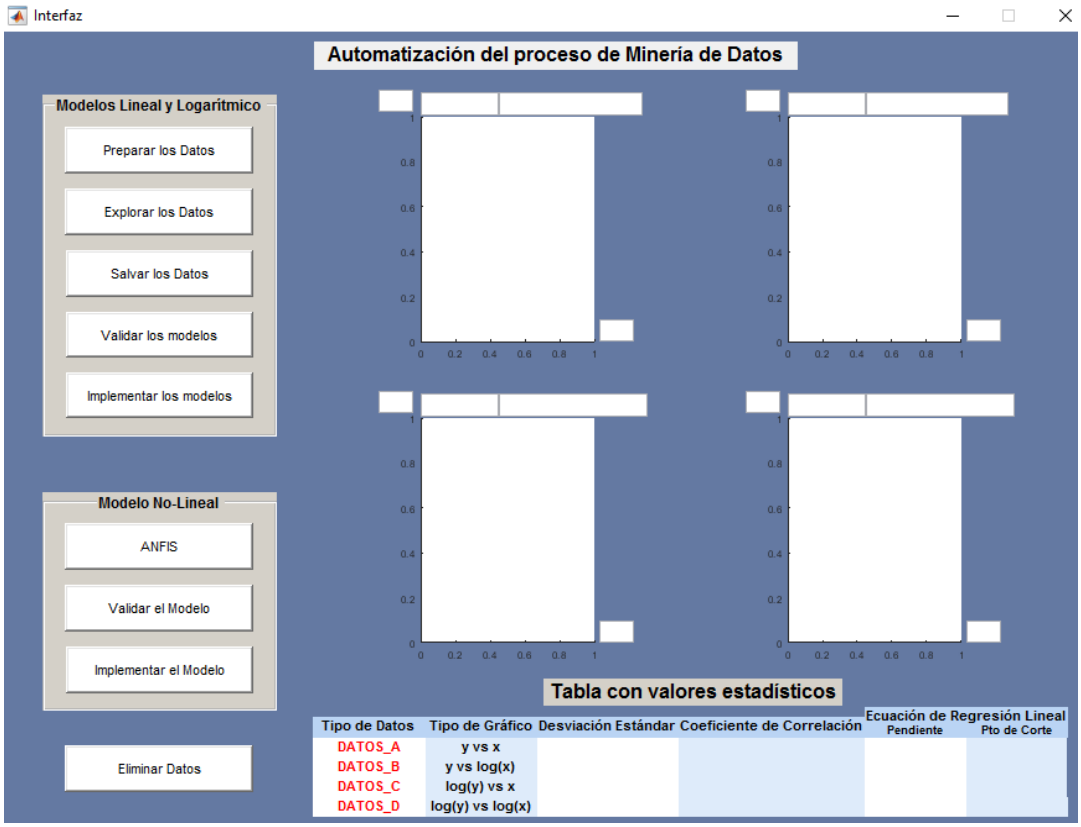


Figura 3.1: Automatización del proceso de minería de datos. Ventana principal

3.1.1. Modelos Lineal y Logarítmico

En este primer apartado mostraremos como automatizamos todos los pasos que involucra el proceso de minería de datos (ver figura 2.1), donde haremos uso de los modelos lineal y logarítmico. Para ello se ha dispuesto al usuario el panel de control que se muestra en la figura 3.2, extraído de la parte superior izquierda de la ventana principal del software diseñado.

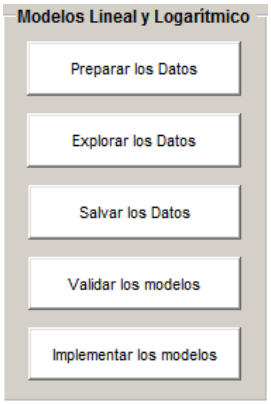


Figura 3.2: Panel de control de los Modelos Lineal y Logarítmico

A continuación se presentarán las funciones implementadas y asociadas a cada botón, algunas propias de MatLab y otras diseñadas, configuradas de manera tal que cada uno de los parámetros de entrada y salida suministren la información necesaria al usuario a través de las interfaces involucradas.

- **Preparación de los datos**

Este control creado con un Push Button y que lleva por nombre "Preparar los datos" (ver figura 3.2) permite al usuario seleccionar un archivo de datos, ubicado en cualquier lugar de la computadora o dispositivo de almacenamiento extraíble como se muestra en la figura 3.3.

La función de MatLab implementada para esta tarea es *uigetfile*, programada para utilizar como argumentos de entrada los formatos de archivos .txt, .xls, .doc, .mat, y .dat, devolviendo a la salida el nombre del archivo y la dirección donde se encuentra. A esta salida se le aplica la función de MatLab *fullfile* y se asigna a una variable, donde se alojará el nombre y la dirección del archivo. Luego, a esta variable se le aplica la función de MatLab *importdata* generándose finalmente la matriz asociada a dicho archivo. Además, si el usuario no selecciona ningún archivo, aparecerá un cuadro de diálogo de error como el de la figura 3.4, para la cual se utilizó la función de MatLab *errordlg*. Veamos cómo está escrito esta parte del programa en la figura 3.5:

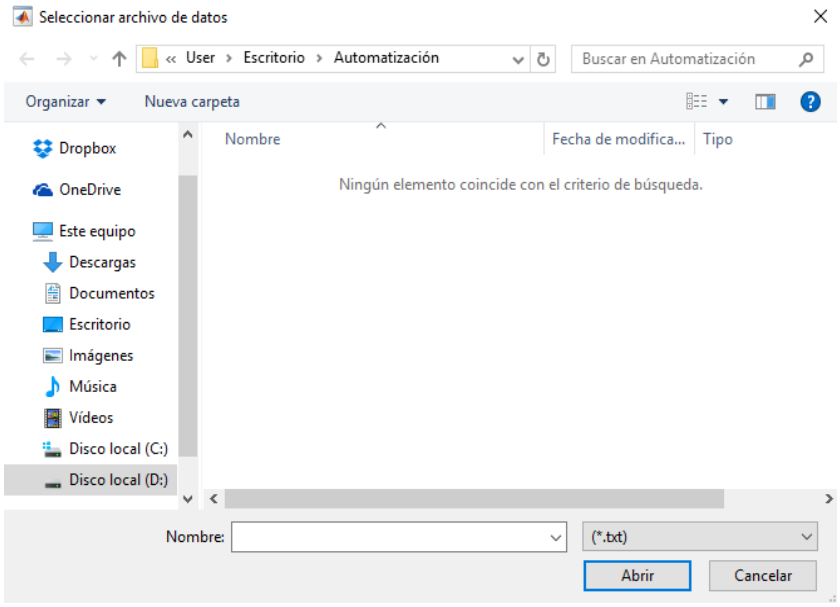


Figura 3.3: Ventana para buscar y seleccionar el archivo de datos

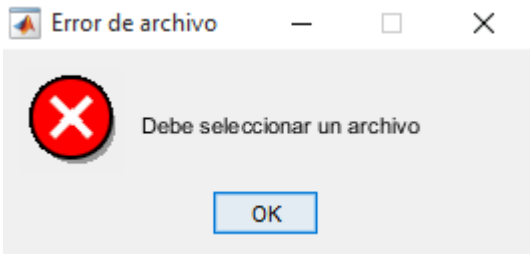


Figura 3.4: Cuadro de diálogo de error al no seleccionar un archivo de datos

```
%Función para abrir archivos
[Filename Path]=uigetfile({'*.txt';'*.xls';'*.doc';'*.mat';'*.dat'},
                          'Seleccionar archivo de datos');
if isequal(Filename,0)
    %Cuadro de diálogo de error
    errordlg('Debe seleccionar un archivo','Error de archivo');
    return
else
%Función con el nombre del archivo completo
    datos=fullfile(Path,Filename);
%Carga datos del archivo en la variable data como una matriz
    datainicial=importdata(datos);
```

Figura 3.5: Código para la selección del archivo de datos

Al generarse la matriz asociada al archivo de datos el programa apertura otra interfaz donde el usuario podrá llevar a cabo la limpieza, filtrado y selección de los datos que serán

procesados en los próximos pasos de la minería de datos. En la figura 3.6 se muestra la interfaz cuyo nombre es "Seleccion", la cual dispone cuatro módulos con filtros específicos, cuatro gráficos asociados a cada uno de ellos, y un módulo adicional que le permitirá al usuario seleccionar y guardar las columnas de datos que usará posteriormente.

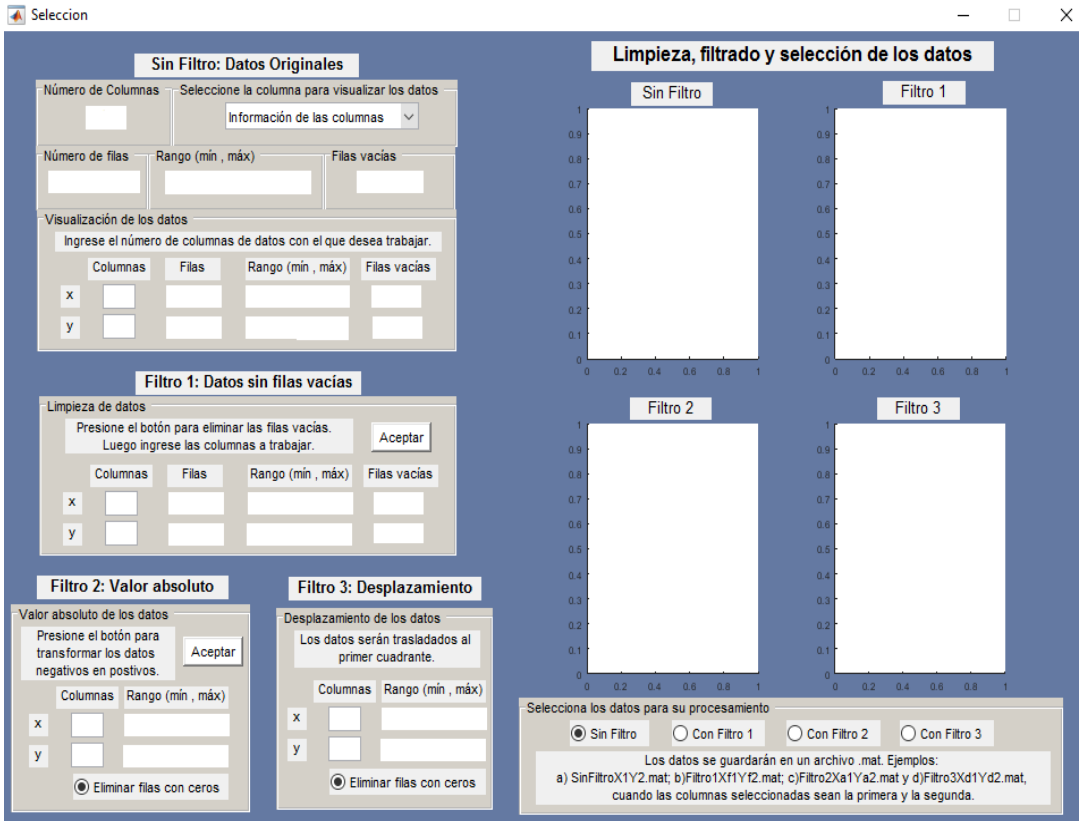


Figura 3.6: Interfaz para la limpieza, filtrado y selección de la matriz de los datos originales

- **Datos: sin filtro**

En la parte superior izquierda de la interfaz "Seleccion" tenemos un primer módulo en donde se muestra en un *Static Text* el número de columnas de datos que posee el archivo (ver figura 3.7). Al seleccionar una columna de la lista desplegable que se encuentra al lado derecho, creada con un *Pop-up Menu*, se muestra el número de filas, los valores mínimos y máximos, y el número de filas vacías, en cuatro *Static Text* por debajo. Posteriormente, el usuario podrá elegir cuáles son los datos más adecuados para el análisis del problema planteado, con tan solo escribir el número de la columna con la que desea trabajar, teniendo en cuenta, que el primero será considerado por el programa como la variable independiente y el segundo número de columna como la variable dependiente. De igual manera se mostrarán el número de filas, los valores mínimos y máximos, y las filas vacías, de las columnas seleccionadas.

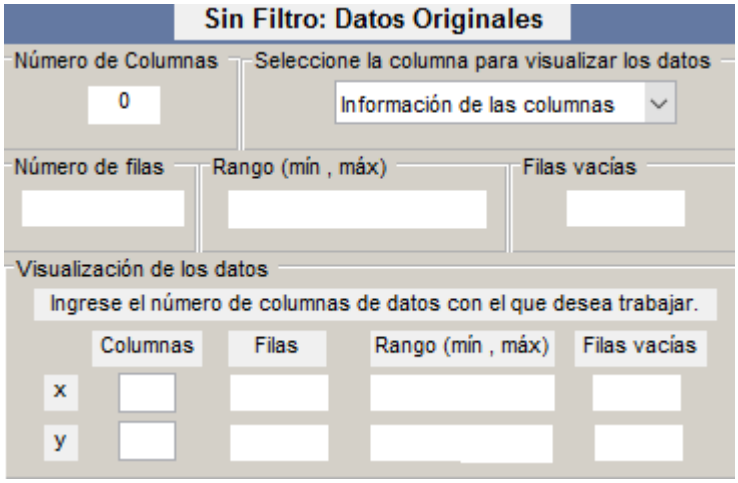


Figura 3.7: Características de los datos originales

En esta primera etapa sin aplicarle un filtro a los datos se utilizaron las funciones *size*, *set*, *get*, *min*, *max*, *isnan* y *sum* de MatLab.

Las funciones *size*, *min*, *max*, *isnan* y *sum* utilizan como argumento de entrada la matriz de datos del archivo seleccionado. La función *size* devuelve a la salida el número de filas y el número de columnas que hay en la matriz de datos, en cambio las otras dos funciones que continúan generan a la salida los valores mínimos y máximos de la columna seleccionada de la lista desplegable. Por otra parte, la función *isnan* detecta las filas que se encuentran vacías, generando a la salida una matriz de datos con 0 y 1, haciendo 1 (true) cuando es vacía (NAN). Luego, aplicando la función *sum* sumamos la cantidad de NAN contenida en cada columna de la matriz anterior.

Con esta información, hacemos uso de la función *set* para mostrar al usuario, los valores correspondientes a los distintos parámetros que se muestran en la figura 3.7, en los diferentes *Static Text*. Es importante resaltar, que la función *get* toma los valores ingresados por el usuario, por ejemplo cuando selecciona de la lista desplegable la columna con la cual desea tener información. Otro ejemplo del uso de esta función se hace notable cuando el usuario ingresa el número de columnas con las que desea trabajar, en el apartado de "Visualización de los datos", generando la ejecución de las funciones anteriores.

Sin embargo, cuando el usuario ingresa el número de columna correspondiente a la variable dependiente se mostrará de inmediato el gráfico de ambas variables en el primer *Axes* titulado "Sin Filtro"(ver figura 3.6). Para generar la gráfica en dos dimensiones se usó la instrucción *plot* de MatLab. Cabe resaltar que si el usuario eligiera la misma columna como variable dependiente e independiente el programa arrojaría un mensaje de error y no ejecutaría ninguna acción (ver figura 3.8).

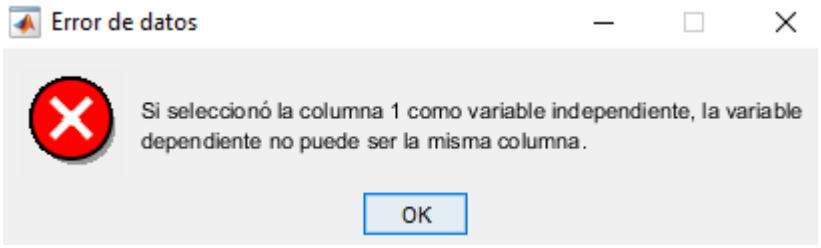


Figura 3.8: Cuadro de diálogo de error al seleccionar la misma columna

Hasta el momento solo hemos mostrado algunas funciones que se repetirán a lo largo del programa y que servirán como referencia en el uso y manejo de otros controles. No obstante, en algunos casos el programa internamente hace uso de arreglos de sentencias **if... elseif... else** como se muestra en la figura 3.9. En otros casos se utilizó la sentencia condicional **switch** la cual permite agrupar el conjunto de sentencias anteriores (ver figura 3.10). Una combinación entre ambas sentencias también se encuentran dispuesta en el programa (ver figura 3.12).

```
if condicion_1
    sentencias_1
elseif condicion_2
    sentencias_2
else
    sentencias_3
end
```

Figura 3.9: Sentencias if... elseif... else

```
switch expresion
    case valor_1
        sentencias_1
    case valor_2
        sentencias_2
    case valor_3
        sentencias_3
end
```

Figura 3.10: Sentencia switch

- **Datos: filtro 1**

Este módulo le permite al usuario limpiar aquellas filas vacías que se encuentran en la matriz del archivo de datos original, con tan solo presionar el Push Button *Aceptar* (ver figura 3.11). Para ello eliminamos las filas que contienen NAN usando la siguiente instrucción: $MatrizInicial(any(isnan(MatrizInicial)'), :) = [\,];$.

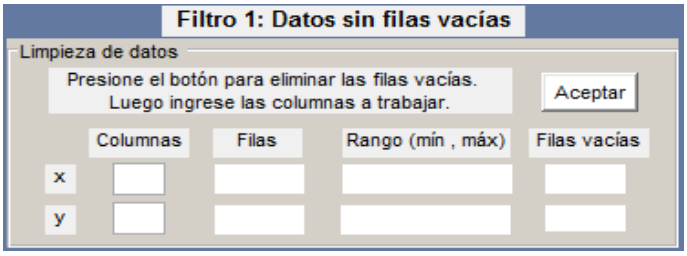


Figura 3.11: Módulo que permite eliminar las filas vacías del archivo de datos

```
if condicion_1
    switch expresion_1
        case valor_1
            sentencias_1
        case valor_2
            sentencias_2
        case valor_3
            sentencias_3
    end
elseif condicion_2
     switch expresion_2
        case valor_4
            sentencias_4
        case valor_5
            sentencias_5
        case valor_6
            sentencias_6
     end
else
     switch expresion_3
        case valor_7
            sentencias_7
        case valor_8
            sentencias_8
        case valor_9
            sentencias_9
     end
end
```

Figura 3.12: Combinación de las sentencias if... switch... elseif... else

Si el usuario, por casualidad ingresará el número de columnas que desea trabajar sin presionar el botón *Aceptar*, el programa no ejecutaría ninguna acción, disponiendo únicamente del módulo anterior (Datos Sin Filtro). Sin embargo, al presionar el botón internamente se genera una matriz, con la misma cantidad de columnas que la matriz original, pero sin NAN. A partir de este momento cuando el usuario ingrese las columnas (x,y) el módulo mostrará la cantidad de filas, los valores mínimo y máximo, y la cantidad de filas vacías igual a cero (0), haciendo uso de las mismas funciones del módulo anterior, y graficando en el *Axes* "Filtro 1" de la figura 3.6.

- **Datos: filtro 2**

Este módulo (ver figura 3.13) es posible ejecutarse solo cuando la limpieza de los datos con filas vacías es accionada, es decir, este módulo no ejecutaría ninguna acción sin haber eliminado las filas con NAN. Por otro lado, si el usuario intentara ingresar el número

de columnas (x,y) sin presionar el Push Button *Aceptar* tampoco se ejecutaría ninguna acción.

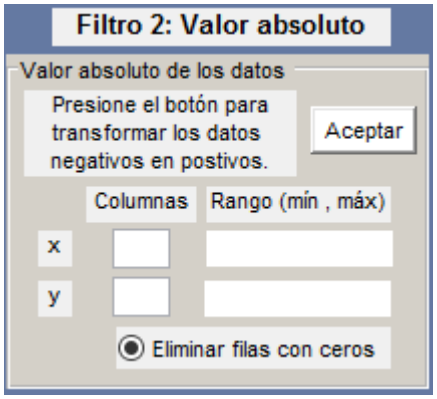


Figura 3.13: Módulo que permite aplicarle el valor absoluto al archivo de datos sin NAN

Cuando el usuario presione el botón *Aceptar*, internamente a la matriz sin NAN se le aplica el valor absoluto por medio de la función *abs* de MatLab, generándose otra matriz con valores positivos únicamente. Con ello, el usuario solo podrá observar los valores mínimos y máximos de las columnas (x,y) ingresadas, con su respectiva representación gráfica en el *Axes* que lleva por nombre "Filtro 2" (ver figura 3.6).

- **Datos: filtro 3**

Tal vez el usuario solo requiera trasladar todos los datos al primer cuadrante sin la necesidad de aplicarle el valor absoluto. Este módulo (ver figura 3.14) tiene la particularidad de trabajar solo con aquellas columnas que se hayan seleccionado en el "Filtro 1: Datos sin filas vacías", de lo contrario el programa no ejecutaría ninguna acción.

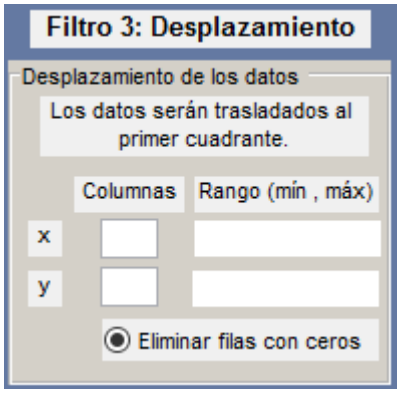


Figura 3.14: Módulo que permite desplazar los datos al primer cuadrante

Para trasladar los datos al primer cuadrante el valor mínimo de cada columna seleccionada debe ser menor a cero (0), de lo contrario el par ordenado no tendría que trasladarse.

Si resulta que el valor mínimo es un número negativo, le sacamos el valor absoluto y el resultado se lo sumamos a cada entrada del vector columna, generándose un nuevo vector columna desplazado al primer cuadrante con el valor mínimo igual a cero 0. De igual manera que en el módulo anterior, el usuario solo podrá observar los valores mínimos y máximos de las columnas (x,y) seleccionadas, pero con su respectiva representación gráfica en el *Axes* que lleva por nombre "Filtro 3" (ver figura 3.6).

Finalmente, después de limpiar y filtrar los datos, el usuario dispone de un módulo para seleccionar y guardar los datos, tal y como se muestra en la figura 3.15. Este panel de control se construye principalmente con cuatro *Radio Button* dispuestos y programados para cada módulo de filtro, el cual permite guardar en formato .mat los datos seleccionados, a través de la función *save* que se explicará más adelante.

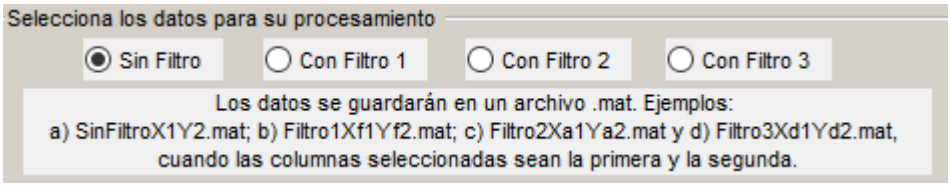


Figura 3.15: Módulo que permite guardar el archivo de datos filtrado o no

Sin embargo cuando el usuario desea seguir trabajando, por ejemplo, con el valor absoluto de la cuarta columna y la tercera columna correspondiente al "Filtro 2: Valor absoluto", solo debe presionar el *Radio Button* "Con filtro 2". En ese momento le aparecerá un cuadro de diálogo de pregunta (*questdlg*) para que confirme, o no, que las columnas seleccionadas y dispuestas internamente en el programa son las columnas que seguirán trabajándose durante el proceso de minería de datos (ver figura 3.16). En esta ocasión el archivo se guardará con el nombre Filtro2Xa4Ya3.mat, donde Xa4 es la cuarta columna con su valor absoluto seleccionada como variable independiente y Ya3 es la tercera columna con su valor absoluto seleccionada como variable dependiente.

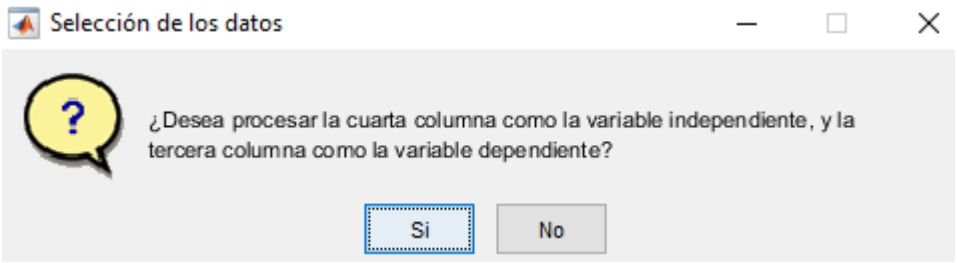


Figura 3.16: Cuadro de diálogo que le permite reafirmar las columnas seleccionadas

Cuando el usuario este de acuerdo con su elección de columnas, automáticamente regresará a la interfaz principal (figura 3.1) donde continuará con el proceso de minería de datos.

- **Exploración de los datos**

Este control creado con un Push Button e identificado como "Explorar los Datos" (ver figura 3.2) también le permite al usuario seleccionar un archivo de datos como se hace en el primer control "Preparar los Datos".

Al momento de seleccionar el archivo de datos, todos los pasos son idénticos a como se mostraron en las figuras 3.3 y 3.4, salvo que el usuario debe considerar la ruta donde se ubica el archivo de datos, guardado en la sección anterior, al momento de instalar la aplicación. Es decir, para la ruta de acceso debe considerarse la unidad donde se alojó y la carpeta donde está contenida la aplicación con el archivo de datos, antes de usar este software. Por ejemplo: 'F:\Automatización\Filtro2Xa4Ya3.mat'. Dicha configuración deberá hacerla solo la primera vez que use este software.

Es importante resaltar que el programa al leer la matriz de datos hace uso de cada columna, como vectores columnas asignándolas a las variables x y y. Continuemos con el ejemplo que estamos tratando, de donde se extraen las columnas x y y del archivo Filtro2Xa4Ya3.mat como se muestra a continuación:

```
x=data.xa4;
y=data.ya3;
```

El usuario podrá visualizar y comparar gráficamente los datos en dos dimensiones y en escalas lineal, semi-logarítmicas y logarítmica en los cuatro *Axes* dispuestos en la interfaz principal como aparece en la figura 3.1. Cada gráfico esta referenciado como: **DATOS_A**, **DATOS_B**, **DATOS_C** y **DATOS_D** correspondientes a los gráficos: y en función de x, y en función de $log(x)$, $log(y)$ en función de x, y $log(y)$ en función del $log(x)$. Además, se muestra en el título de cada gráfico y la forma de la ecuación que mejor ajusta los datos a una linealidad (ecuación de inferencia Y_{inf}), donde cada eje de coordenadas también está identificado como se muestra en la figura 3.17. Los pares ordenados se graficarán en los distintos *Axes* con círculos de color rojo, y la línea de tendencia inferida que ajusta los datos a una linealidad será de color azul.

Los títulos de cada *Axes* son identificados con cuatro *Edit Text* cada uno, los cuales son etiquetados internamente con distintos nombres haciendo uso del *Property Inspector* de la barra de herramientas de GUIDE (ver figura 2.24). Al momento de la exploración de los datos los títulos pueden ser renombrados por el usuario si así lo considera. En la figura 3.18 se detalla el código para los títulos de los gráficos, donde la función *set* en su primer argumento utiliza el apuntador **handles** para referirse al *Edit Text* identificado con un nombre en particular, el segundo argumento va dirigido al tipo de variable que queremos mostrar, en estos casos un 'String', y un último argumento para escribir el título que se quiere mostrar al usuario.

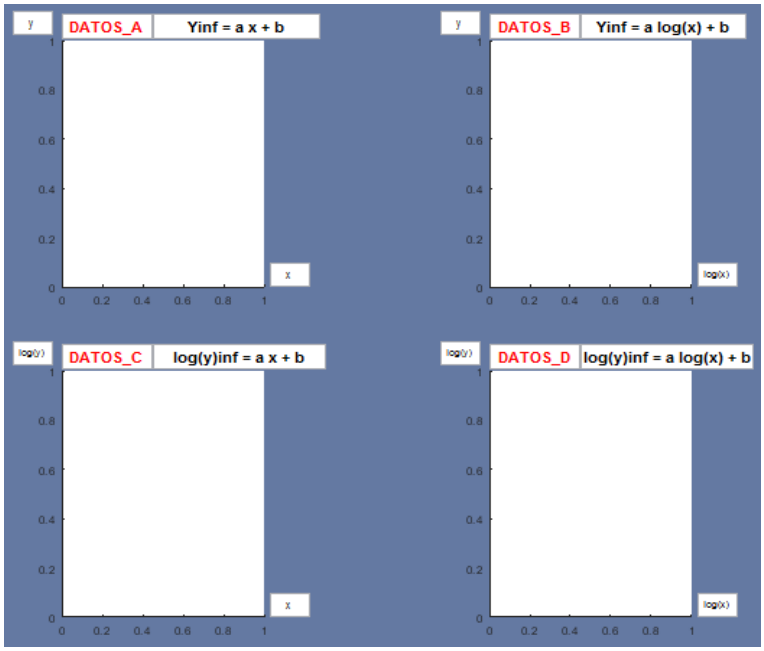


Figura 3.17: Diagrama de dispersión en escala lineal, semi-logarítmica y logarítmica

```
%Título de los gráficos
set(handles.editxy_A,'String','DATOS_A');
set(handles.editxy,'String','Yinf = a x + b');
set(handles.y1,'String','y');
set(handles.x1,'String','x');
set(handles.editlogxy_B,'String','DATOS_B');
set(handles.editlogxy,'String','Yinf = a log(x) + b');
set(handles.y2,'String','y');
set(handles.x2,'String','log(x)');
set(handles.editxlogy_C,'String','DATOS_C');
set(handles.editxlogy,'String','log(y)inf = a x + b');
set(handles.y3,'String','log(y)');
set(handles.x3,'String','x');
set(handles.editlogxlogy_D,'String','DATOS_D');
set(handles.editlogxlogy,'String','log(y)inf = a log(x) + b');
set(handles.y4,'String','log(y)');
set(handles.x4,'String','log(x)');
```

Figura 3.18: Código para los títulos de los gráficos

Este control ejecuta una segunda acción que tiene que ver con los valores estadísticos que se muestran en la tabla de la figura 3.1, como: la desviación estándar, el coeficiente de correlación y los parámetros de la ecuación lineal. Dichos valores se mostrarán con una sola cifra significativa, salvo el coeficiente de correlación que se mostrará con dos cifras significativas. En general, se han utilizado las funciones *set, std, fitlm, polyfit, polyval, plot, log10* y las sentencias de la figura 3.9.

La función *std* usa como argumento de entrada la variable independiente (y), devolviendo a la salida el valor de la desviación estándar, la cual es mostrada al usuario empleando la función *set* a través de un *Static Text* en la tabla de valores estadísticos, con una sola cifra significativa. Veamos cómo está dispuesto el código del estudio estadístico de los datos cuando nos referimos a **DATOS_A** (ver figura 3.19).

```
%Estadística
%Declaración de las variables globales
global yinfa mdla x y
%Cálculo de la desviación estándar
desva=std(y);
%Asigna valores al edit text Desva
%num2str función que convierte número a texto
%'%10.2e' imprime el valor en formato exponencial con
%una cifra significativa
set(handles.Desva,'String',num2str(desva,'%10.1e'));
%Función que genera el modelo de regresión lineal
mdla=fitlm(x,y);
%Valor del coeficiente de correlación
ra2=mdla.Rsquared.Ordinary;
set(handles.Correlaciona,'String',num2str(ra2,'%10.2f'));
%Parámetros de la ecuación de regresión lineal
%Pendientea de la recta
a1=mdla.Coefficients.Estimate(2,1);
set(handles.Pendientea,'String',num2str(a1,'%10.1e'));
%Punto de corte con el eje de las ordenadas
b1=mdla.Coefficients.Estimate(1,1);
set(handles.Ptocortea,'String',num2str(b1,'%10.1e'));
%Ecuación de regresión lineal
ma=polyfit(x,y,1);
%Generación de yi
yinfa=polyval(ma,x);
```

Figura 3.19: Código de las funciones utilizadas para el estudio estadístico de x vs y

Podemos observar que la función *fitlm* de MatLab permite ajustar a un modelo de regresión lineal usando como argumento de entrada las variables (x,y). Esta función genera una estructura llena de información variada, de donde nos interesa en estos momentos extraer los valores del coeficiente de correlación a través de la extensión *Rsquared.Ordinary*, y los de una ecuación lineal a través de las extensiones *Coefficients.Estimate(2,1)* para la pendiente de la recta y *Coefficients.Estimate(1,1)* para el punto de corte con el eje de las ordenadas (ver figura 3.19).

Por otro lado, por simplicidad hemos usado la función *polyfit* de MatLab indicándole en la entrada, las variables y el grado del polinomio del cual queremos obtener sus coeficientes, en nuestro caso de grado uno, con la intención de obtener la pendiente y el punto

de corte de la recta con el eje de las ordenadas en una sola matriz. A pesar que la función *fitlm* tiene dentro de su estructura de salida los valores de los parámetros de la ecuación lineal, la función *polyfit* facilita generar los valores inferidos Y_{inf} usando la función *polyval* de MatLab.

Una tercera acción que realiza este control es la transformación del par ordenado (x, y) a escala logarítmica de base diez, donde se utilizó la función *log10* de MatLab. El usuario debe considerar que en la preparación de los datos, la limpieza, el filtrado y la selección de los datos, se presentan opciones para que las variables (x, y) sean mayores y distintos de cero. Sin embargo, si al momento de explorar los datos el usuario obvia las restricciones que debe tener el argumento del *log10* el programa está dispuesto a solo presentar gráficamente y con sus valores estadísticos, aquellos resultados que sean posibles.

Para trabajar con la función *log10* el código se ha condicionado como se muestra en la figura 3.20. Cuando la condición de las variables son valores mayores que cero, se ejecuta la sentencia de la función logarítmica acompañado del código de la figura 3.19, con la correspondiente variable según el tipo de dato.

```
%Generación de valores logarítmicos de la variable x
if(x<0)
elseif(x==0)
elseif(x>0)
    logx=log10(x);
end
%Generación de valores logarítmicos de la variable y
if(y<0)
elseif(y==0)
elseif(y>0)
    logy=log10(y);
end
```

Figura 3.20: Código empleado para generar los valores logarítmicos de las variables (x, y)

Veamos las posibles acciones que pueden suceder:

- Si los valores de ambas variables (x, y) son mayores que cero, se ejecutan todos los tipos de datos con sus respectivos gráficos y valores estadísticos.
- Si algún valor de la variable x es menor o igual que cero, considerando que la variable y es mayor que cero, se ejecutan los tipos de datos **DATOS_A** y **DATOS_C**.
- Si algún valor de la variable y es menor o igual que cero, considerando que la variable x es mayor que cero, se ejecutan los tipos de datos **DATOS_A** y **DATOS_B**.
- Si los valores de ambas variables (x, y) son menores que cero, solo se ejecutaría el tipo de dato, **DATOS_A**.

- **Salvar los Datos**

Este control creado con otro Push Button e identificado como "Salvar los Datos" (ver figura 3.2) permite al usuario guardar en varios archivos con la extensión *.mat* los valores de las variables utilizadas en pares ordenados para los distintos tipos de datos, y los mostrados en la tabla con valores estadísticos. Para ello utilizamos la función *save* de MatLab, cuyos parámetros de entrada son: ('Nombre del Archivo' , 'Variable a guardar'). En la figura 3.21 se muestra explícitamente como está configurada esta parte del código.

```
% Guarda los valores como par ordenados de las variables (x,y)
% en las distintas escalas lineal y logarítmica
save ('DATOS_A','DATOS_A');
save ('DATOS_B','DATOS_B');
save ('DATOS_C','DATOS_C');
save ('DATOS_D','DATOS_D');
% Se crea una matriz con los valores de la tabla estadística
Exploracion_Estadistica=[desva ra2 a1 b1;
                         desva rb2 a2 b2;
                         desvb rc2 a3 b3;
                         desvb rd2 a4 b4];
%Guarda la matriz con los valores de la tabla estadística
save('Exploracion_Estadistica','Exploracion_Estadistica');
```

Figura 3.21: Código para salvar los resultados obtenidos en la exploración de los datos

El código facilita al usuario manejar los tipos de datos, distribuidos en cuatro archivos, los cuales pueden ser renombrados si así lo considera, pero teniendo en cuenta que al momento de usar los mismos debe escribir en el Workspace la función *importdata* cuyo argumento es el 'Nombre del Archivo.mat'. Esto mostrará en el Command Window de MatLab los valores del archivo en una matriz de 2 columnas, para el caso de las variables independiente y dependiente, y en una matriz $4x4$ los valores contenidos en la tabla estadística.

- **Validación de los modelos**

Este control creado con un Push Button e identificado como "Validar los Datos" (ver figura 3.2) permite al usuario ver qué tan bueno es el ajuste de los datos cuando se comparan los valores inferidos Y_{inf} con los valores de la variable dependiente y en las distintas escalas lineal y logarítmica. Esto se mostrará gráficamente con círculos de color azul, además de mostrar una línea de tendencia que mejor ajusta los datos también en color azul, y una línea negra que representará la identidad.

Para ello se ha dispuesto una interfaz adicional titulada "Inferencia" como se muestra en la figura 3.22, la cual está diseñada con cuatro *Axes*, cada uno con sus títulos y la identificación de sus ejes respectivamente, usando *Edit Text*. Además muestra al usuario una

tabla con valores estadísticos como: el RMSE, el coeficiente de correlación y la pendiente de la recta.

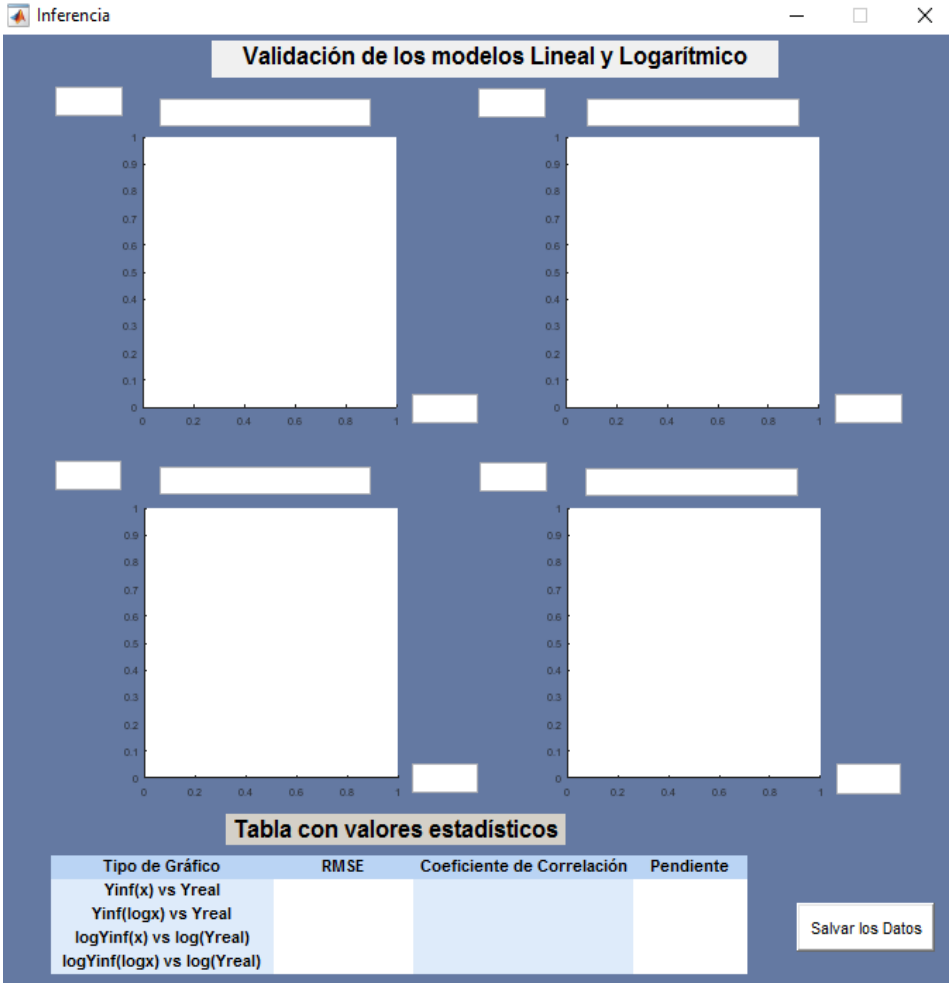


Figura 3.22: Interfaz que permite la validación de los modelos lineal y logarítmico

Las funciones implementadas en este control son las mismas que las del control exploración de los datos, pero con diferentes etiquetas dispuestas dentro de las líneas de código. Lo nuevo aparece cuando extraemos de la estructura de datos generada con la función *fitlm* el valor RMSE, y la manera en como una sola gráfica permite hacer tres representaciones distintas usando la función *plot*, tal y como se muestra en las líneas de código de la figura 3.23.

Para el proceso de validación de las demás ecuaciones de inferencia, se desarrolla el código de la misma manera, considerando las distintas escalas lineal y logarítmica según sea el caso. Por otra lado, el botón ubicado en la parte inferior derecha de esta interfaz permite asociar internamente los valores contenidos en la tabla estadística en una matriz $4x3$, que luego son guardados en un archivo con el nombre de 'Validacion_Estadistica' usando la función save con en el caso de la figura 3.21.

```
%Estadística
%Estudio de Yinfa(x,y) vs Yreal
%Modelo de regresión lineal entre Yinfa y Yreal
mdlinfa=fitlm(y,yinfa);
%RMSE de yinfa
RMSEa=mdlinfa.RMSE;
set(handles.RMSEa,'String',num2str(RMSEa,'%10.1e'));
%Coeficiente de correlación
rinfa2=mdlinfa.Rsquared.Ordinary;
set(handles.Correlacioninfa,'String',num2str(rinfa2,'%10.2f')); %decimal
%Pendiente de la recta
ainfa=mdlinfa.Coefficients.Estimate(2,1);
set(handles.Pendienteinfa,'String',num2str(ainfa,'%10.1e'));
%Ecuación de regresión lineal, modelo inferido a
minfa=polyfit(y,yinfa,1);
%Generación de yvalidacion
yvalidaciona=polyval(minfa,y);
%Graficación
plot(uno,y,yinfa,'bo',y,yvalidaciona,'--',y,y,'k');
```

Figura 3.23: Código que permite validar la inferencia de los **DATOS_A** en función de la variable dependiente y

- **Implementación de los modelos**

Este control creado con un Push Button e identificado como "Implementar los modelos" (ver figura 3.2) permite al usuario disponer de los parámetros obtenidos en el proceso de exploración de los datos, como la pendiente de la recta y el punto de corte de los diferentes tipos de datos, para ser utilizados en la generación de nuevos valores inferidos ajustados a una tendencia lineal.

Para lograr esto, la variable independiente que alimentará la ecuación deberá ser seleccionada de un conjunto de datos medidos por el usuario y transformados en un vector columna. Esta selección la podrá realizar a través de los Push Button que se muestran en los ejes de las abscisas de cada gráfico, de la interfaz "Implementacion" (ver figura 3.24). Adicionalmente se dispone de un botón para guardar en un archivo los valores inferidos.

Las funciones que se emplearon en esta sección para la selección del archivo de datos son las mismas del código de la figura 3.5, y para la generación de los valores inferidos la función *polyval* como se han comentado anteriormente. Se recomienda al usuario preparar los datos antes de la implementación, teniendo en cuenta los valores negativos y cero para las distintas escalas lineal y logarítmica. Para guardar los valores inferidos de cada caso se generó una matriz de cuatro columnas con el nombre de 'Implementacion_Estadistica' para la cual se utilizó la función *save* de MatLab.

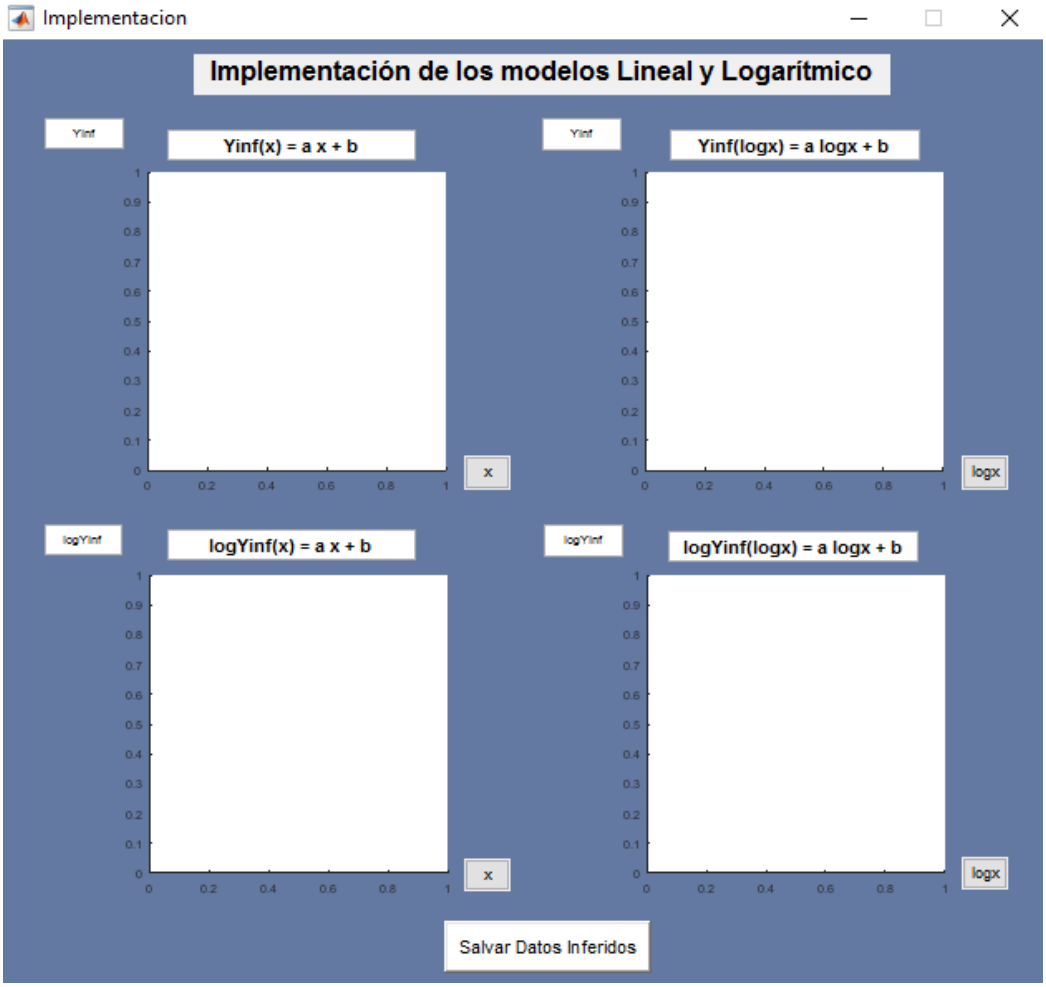


Figura 3.24: Interfaz gráfica para la implementación de los modelos lineal y logarítmico

3.1.2. Modelo No-Lineal

Si resulta que el Modelo Lineal-Logarítmico empleado para inferir los datos no es el adecuado, entonces el usuario dispondrá de este módulo hasta obtener, o no, algún tipo de tendencia o patrón entre los datos. Veamos cómo están las funciones implementadas y asociadas en cada botón del panel de control de la figura 3.25, el cual es obtenido de la parte inferior izquierda de la ventana principal del este software (ver figura 3.1).

Figura 3.25: Panel de control del Modelo No-Lineal

- **Módulo de ANFIS**

La intención de este control es explorar los tipos datos que se procesaron en el modelo anterior, para poder comparar ambos resultados y tomar la decisión del modelo que mejor ajusta los datos a una linealidad. Este control creado con un Push Button y que lleva por nombre "ANFIS" (ver figura 3.25) permite al usuario utilizar el módulo de ANFIS a través de la función *anfisedit*, la cual apertura la interfaz gráfica que se muestra en la figura 2.26.

Ubicándose en **Load data** en el área de trabajo de esta interfaz, podrá seleccionar alguna de las opciones: *Training*, *file* y pulsar el botón *Load Data...*, ó, *Training*, *worksp.* y pulsar el botón *Load Data...*, con la finalidad de entrenar los datos ubicados en un archivo o en el espacio de trabajo de MatLab. Considerando que solo se puede procesar un solo tipo de dato a la vez, el usuario tiene la libertad de procesar los cuatro tipos de datos, uno a uno, o tomar el que mejor corresponda según sus criterios.

Para el uso de los demás controles del área de trabajo de esta interfaz, el usuario debe considerar los ítems tratados en la sesión 2.3.2, teniendo en cuenta que al generar el modelo FIS a la salida debe escoger una tendencia de tipo lineal.

Al finalizar con todo el procesamiento de datos y presionar el botón *close* aparecerá un cuadro de diálogo tipo pregunta como el de la figura 3.26. Al presionar *Yes* se abrirá una ventana donde podrá guardar el archivo generado por este módulo con la extensión *.fis*.

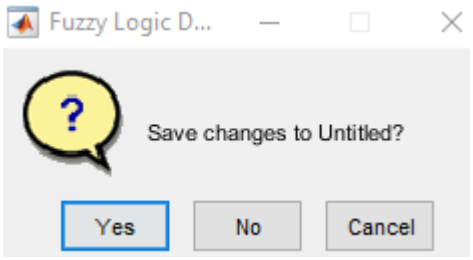


Figura 3.26: Cuadro de diálogo que le permite guardar el archivo generado por el módulo de ANFIS

- **Validación del modelo**

Una vez procesados los datos con el módulo de ANFIS, podemos utilizar la información necesaria para visualizar, comparar y analizar, de igual manera como en los modelos anteriores, distintos tipos de gráficos y valores estadísticos de los datos inferidos Y_{inf} en función de los datos reales y, en otra ventana GUIDE. Este control creado con un Push

Button e identificado como "Validar el Modelo" (ver figura 3.25) permite al usuario trabajar con el archivo de datos generado anteriormente en el módulo de ANFIS.

La aplicación de las funciones son iguales a como hemos explicado anteriormente a la hora de seleccionar el archivo de datos, cambiando solamente la función *importdata* por la función *readfis* de MatLab, la cual permite leer los datos de un archivo *.fis*. Al seleccionar el archivo de datos, se apertura la interfaz de la figura 3.27.

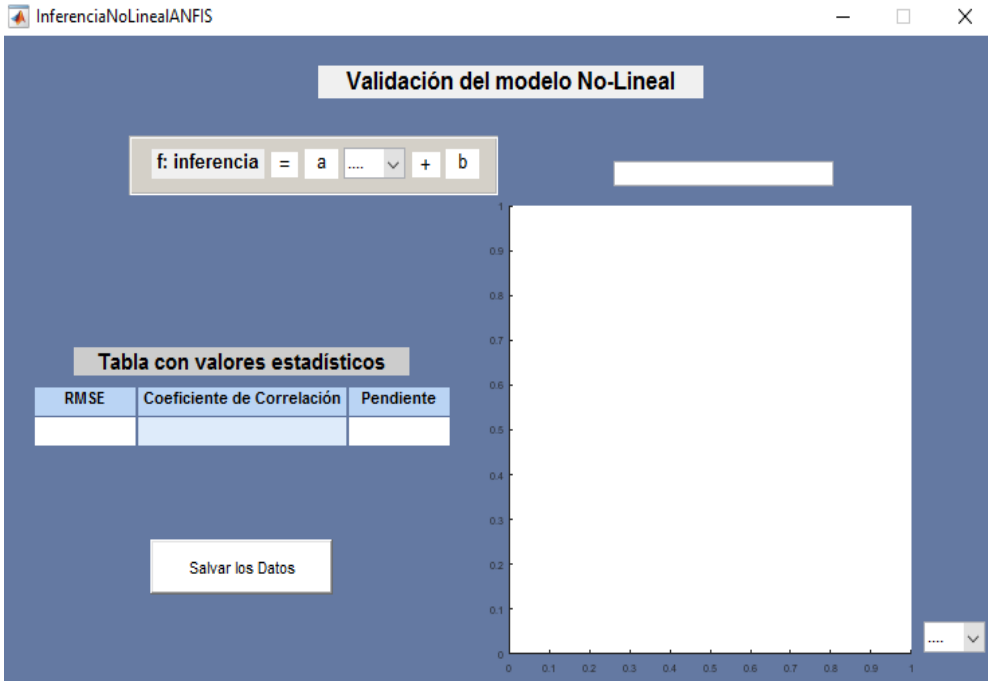


Figura 3.27: Interfaz que permite validar el Modelo No-Lineal

En la leyenda del eje de las ordenadas se muestra la forma que tendrá la función de inferencia, cuyos parámetros a y b están contenidos en el archivo *.fis*. La variable independiente solo puede tomar dos posibles valores x ó $logx$ a través de la pestaña desplegable que aparece al lado del parámetro a.

El usuario debe tener claridad qué tipo de datos procesó para obtener la función de inferencia adecuada al momento de validar el modelo. Para ello usamos la función *evalfis* de MatLab, que contiene dos argumentos de entrada, (Variable_Independiente , Archivo.fis), arrojando a la salida los valores inferidos de ANFIS.

Adicionalmente el eje de las abscisas también presenta una pestaña desplegable con dos condiciones y ó $logy$, que son los valores reales de la variable dependiente que permitirán validar el modelo. Al momento de seleccionar la opción correspondiente al tipo de datos, se graficarán la función de inferencia en círculos de color azul, la línea de tenden-

cia en color azul después de realizar el ajuste lineal, y de color negro se trazará la identidad.

De igual manera aparece una tabla con algunos valores estadísticos que hemos trabajado anteriormente, y un botón que permite guardarlos en una matriz de 3 columnas, cuyo nombre del archivo será 'Validacion_EstadisticaNL'. Esto con la intención de hacer uso posteriormente de los parámetros de la ecuación lineal en la etapa de implementación de los datos, de este módulo.

- **Implementación de los modelos**

Este control creado con un Push Button e identificado como "Implementar el Modelo" (ver figura 3.25) permite al usuario hacer uso del archivo *.fis* generado en el módulo de ANFIS, con la intención de implementar los parámetros contenidos en el mismo a un conjunto de datos seleccionados a través de la lista desplegable que aparece en el eje de las abscisas de la figura 3.28 y ajustarlos a una linealidad.

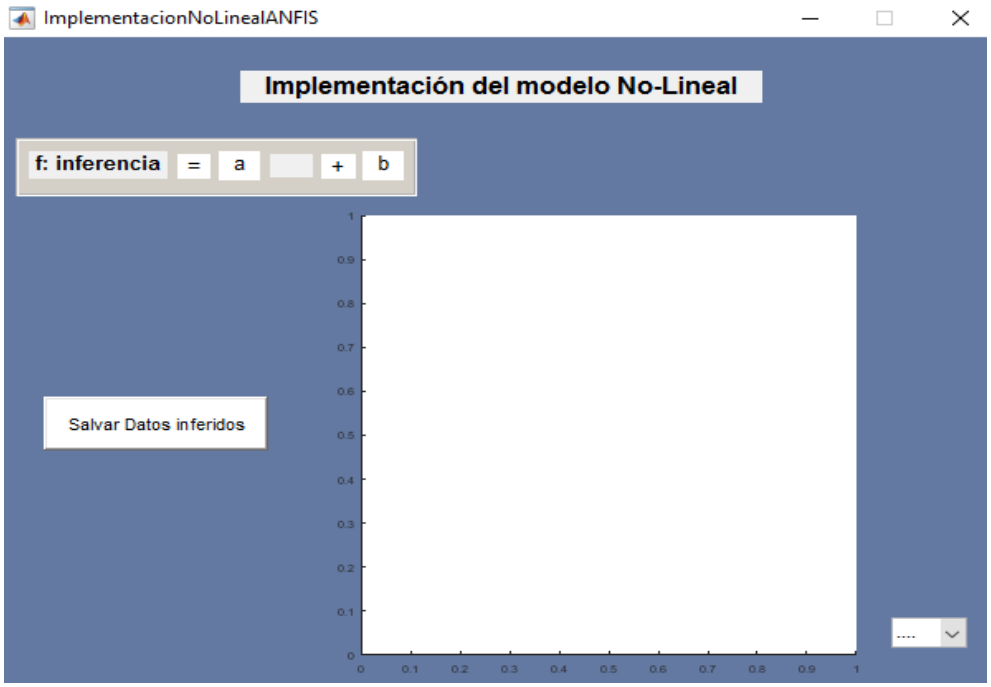


Figura 3.28: Interfaz que permite implementar el Modelo No-Lineal

Esta interfaz tiene la particularidad de procesar solo un tipo de datos de los 4 posibles casos que se pueden estar trabajando, y esto es debido a que el módulo de ANFIS solo trabaja con un par ordenado de datos. Sin embargo, como se explicó anteriormente esto no limita al usuario a procesar o trabajar con todas las posibles combinaciones tratadas en las distintas escalas.

Así pues, se ofrece un Push Button adicional para guardar en un archivo los datos inferidos por este modelo con el nombre de 'Implementacion_NoLineal' en una matriz

vector columna.

Finalmente, si partimos con la posibilidad de que el usuario seleccione un archivo de datos con el cual no desea seguir trabajando, se le ofrece la opción de eliminar todos los datos y variables que se hayan procesado. Para lograr esto se dispone de un Push Button con el nombre de 'Eliminar Datos' en la interfaz principal de este software (ver figura 3.1). Al presionar dicho botón se apertura un cuadro de diálogo tipo pregunta como aparece en la figura 3.29.

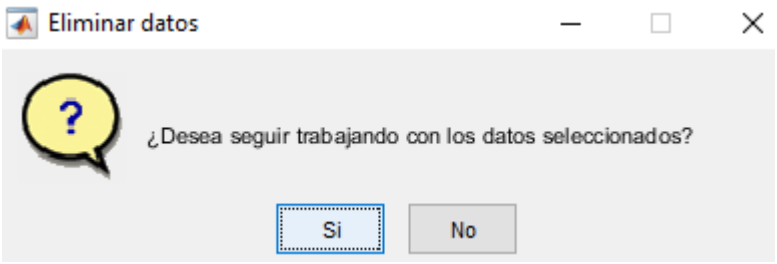


Figura 3.29: Cuadro de diálogo que permite eliminar los datos trabajados

Las funciones utilizadas para este botón son: *clearvars*, *clear* y *close*. Las dos primeras permiten borrar y limpiar todos los datos y variables generados en el espacio de trabajo. La función *close* cierra todas las interfaces abiertas, salvo la ventana principal 'Interfaz'.

Por otro lado la instrucción ' ' juega un papel importante al momento de eliminar toda la información mostrada en los Axes y en la tabla con valores estadísticos a través del uso de la función *set* y de los apuntadores *handles*.

Capítulo 4

Resultados

Cada panel de control fue evaluado con datos provenientes de algunos ejemplos prácticos que se emplean en el laboratorio de física uno y de algunos trabajos realizados en el Laboratorio de Física Teórica del Sólido.

Como punto de partida, procesamos los datos en hojas de cálculo de Microsoft Excel de la experiencia donde se mide la energía total radiada (R) por unidad de tiempo y por unidad de área de la superficie de un alambre de tungsteno como función de su temperatura (T), y otra experiencia donde se mide la diferencia de potencial en los extremos de un condensador (Vc) en función del tiempo de descarga (t). Con estas referencias se comenzaron a realizar pruebas en cada uno de los controles diseñados en el software con la intención de compararlos e ir ajustando todos los parámetros involucrados, logrando que ambas aplicaciones en sus resultados dieran exactamente lo mismo (ver Apéndice A).

En este capítulo solo presentaremos los resultados obtenidos después de someter la aplicación a rigurosas pruebas con un archivo de datos provenientes de un fenómeno físico, el cual contenía una variedad de valores tanto negativos como positivos, incluyendo al cero y con algunos espacios vacíos.

4.1. Modelos Lineal y Logarítmico

Al ubicarnos en la interfaz gráfica principal a través de la línea de comandos, como se indicó en el capítulo anterior, presionamos el botón 'Preparar los Datos' para seleccionar un archivo de datos en particular con el nombre 'VaciosConNegativos' con la cual se apertura la interfaz 'Seleccion'.

Haciendo un barrido de las columnas a través de la lista desplegable, se pueden ver los valores máximos y mínimos de cada una, así como la cantidad de filas vacías. Ingresando los valores del par de columnas con la que queremos trabajar, se muestra de igual manera

como están dispuesto los datos en cada columna graficándose al momento de escribir el valor de la variable dependiente. En la figura 4.1 se muestran los resultados de las acciones después de escribir la columna 4, como la variable x, y la columna 3 como la variable y.

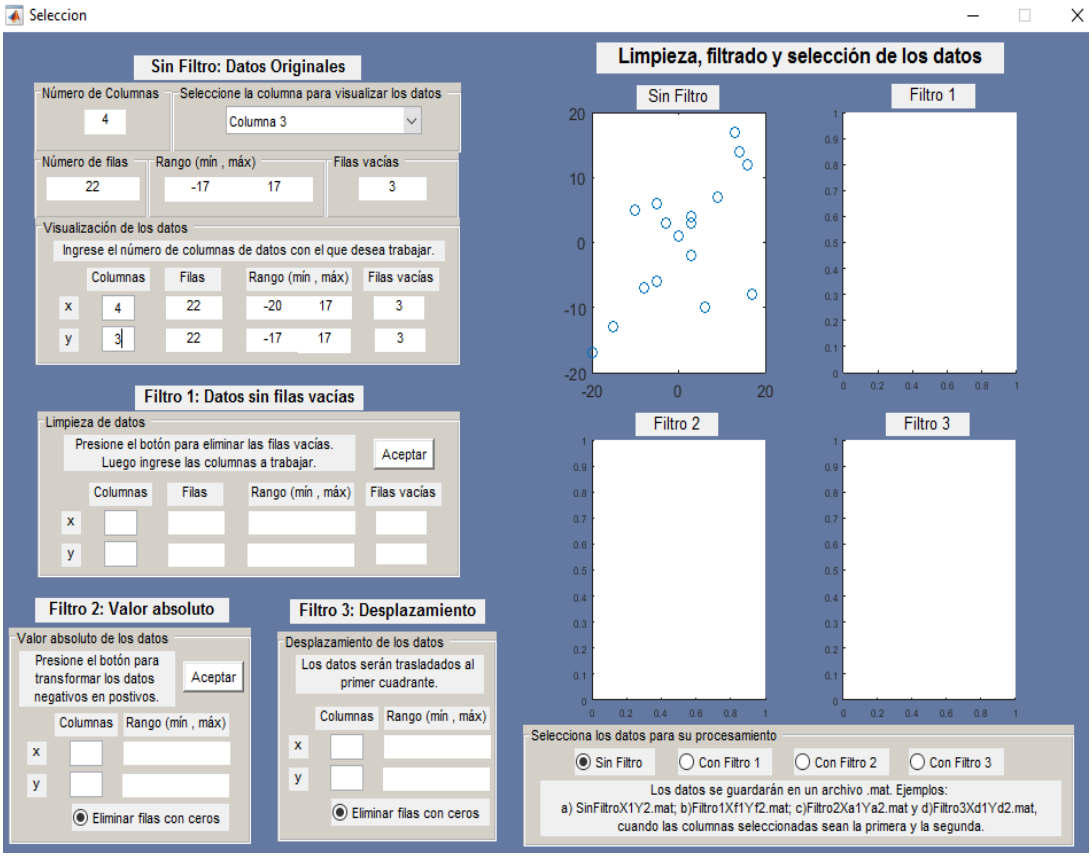


Figura 4.1: Muestra de parámetros y gráfica de las columnas seleccionadas sin aplicarle filtro

Veamos en la figura 4.2 la particularidad de aplicarle los filtros al par de columnas de nuestro ejemplo, considerando que después de presionar el botón 'Aceptar' del Filtro 1, se eliminarán todas las filas vacías. No obstante, si el usuario no presionará dicho botón, al tratar de insertar alguna columna el programa no ejecutaría ninguna acción. Aplica de igual manera para el Filtro 2. Por otro lado, existe la posibilidad de colocar otras columnas a las anteriores en todos los filtros, pero teniendo en cuenta que no se lograría una comparación entre los datos.

Así pues, el usuario podría realizar un barrido por todas las columnas, visualizando, comparando y eligiendo la que mejor se relaciona según lo observado. Para finalizar con este primer proceso de minería de datos, seleccionaremos los datos arrojados por el Filtro 2 haciendo clic en el Radio Button 'Con Filtro 2'. Al aceptar en el cuadro de diálogo, como el de la figura 3.16, regresaríamos a la interfaz principal del software.

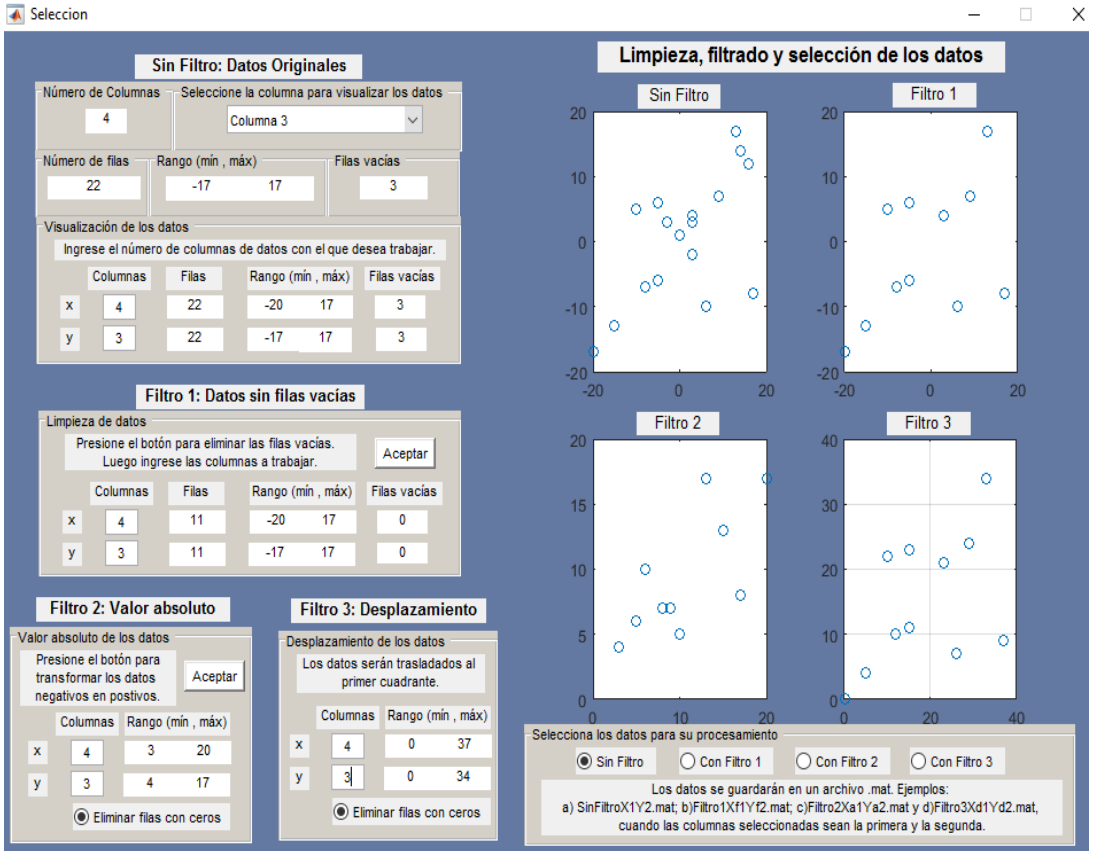


Figura 4.2: Limpieza y filtrado de las columnas seleccionadas

Por el momento no se observaría ningún cambio en la interfaz hasta presionar el botón 'Explorar los Datos'. En ese momento se muestra la opción para elegir el archivo guardado con se muestra en la figura 4.3.

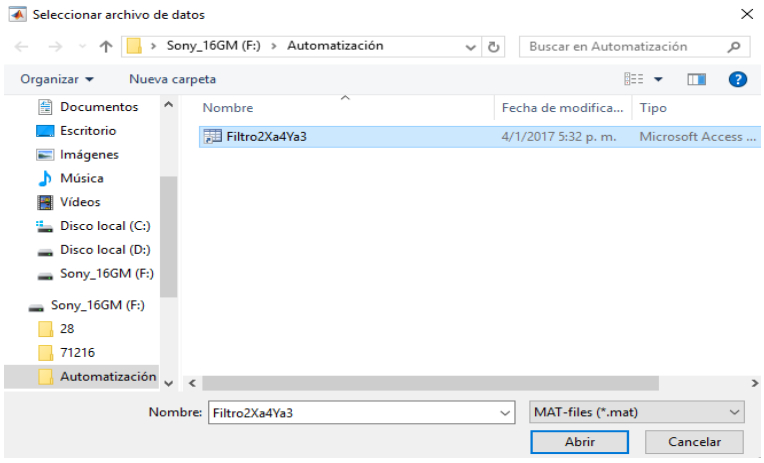


Figura 4.3: Ventana para buscar y seleccionar el archivo de datos preparados

Esperando tan solo un instante el programa ejecutaría todas las acciones correspon-

dientes a la exploración de los datos, arrojando todos los resultados en la interfaz principal tal y como se muestra en la figura 4.4.

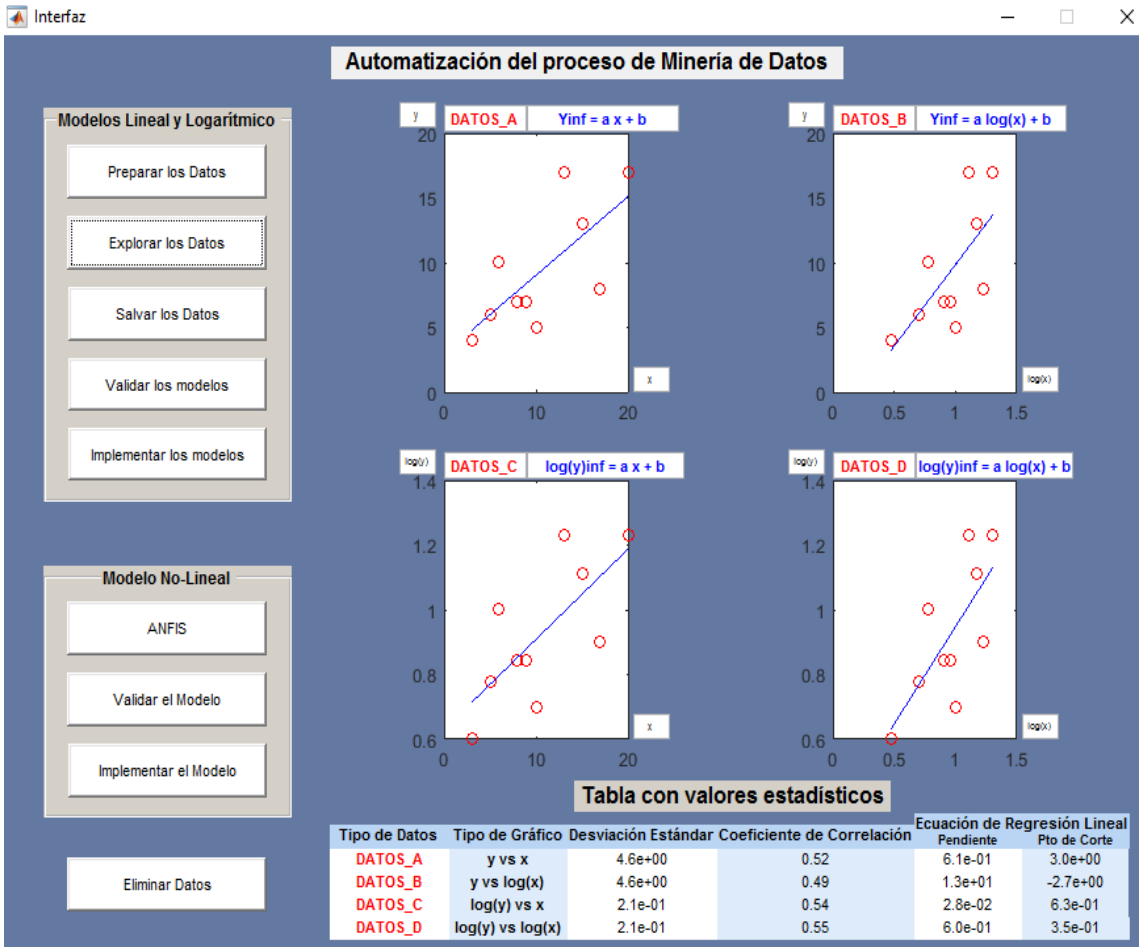


Figura 4.4: Exploración de los datos preparados

Como hemos discutido, las representaciones gráficas en círculos de color rojo corresponden a los datos preparados en sus distintas escalas, y la línea de tendencia de los datos inferidos en color azul. Además la tabla con algunos valores estadísticos para los distintos tipos de gráficos.

Para guardar los resultados de la exploración de los datos le damos un clic al botón 'Salvar los Datos'. Con ello se guardarán en la carpeta donde se aloja el programa los archivos que se muestran en la figura 4.5, como son los cuatro archivos de las posibles combinaciones lineal y logarítmica, y un archivo adicional el cual contiene los valores arrojados en la tabla estadística.

Al presionar el botón para 'Validar los modelos', automáticamente se apertura la ventana 'Inferencia' (ver figura 4.6) donde se describe en círculos de color azul los valores

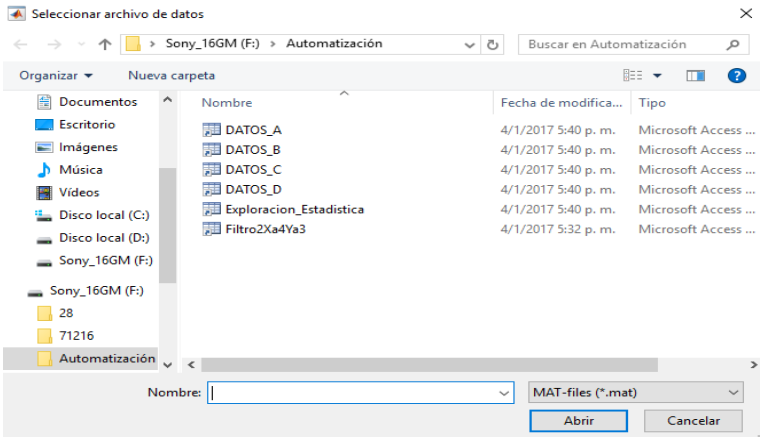


Figura 4.5: Archivos que se guardan de la exploración de los datos

inferidos Y_{inf} en función de la variable y, una línea de tendencia en azul y la identidad en color negro, en los distintos tipos de gráficos. De igual manera se muestran ciertos valores estadísticos en la tabla correspondiente a cada caso.

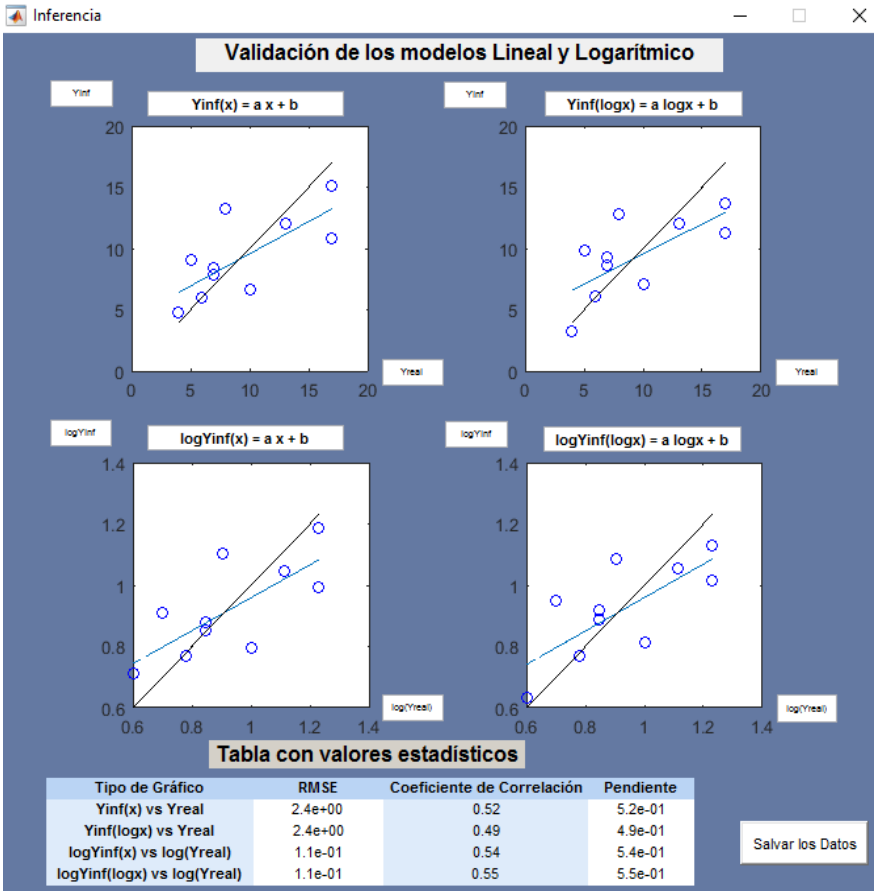


Figura 4.6: Validación de los modelos lineal y logarítmico con los datos explorados

Al presionar el botón 'Salvar los Datos' se guardaría en una matriz los valores estadísticos con las dimensiones $4x3$, con el nombre de 'Validacion_Estadistica' como se muestra en la figura 4.7.

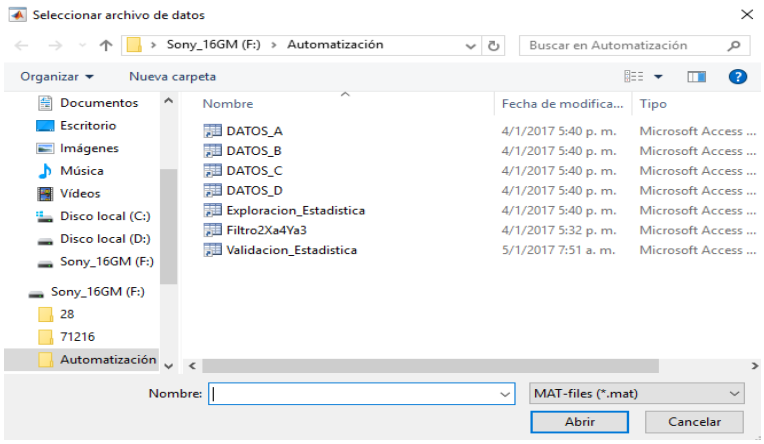


Figura 4.7: Archivo que se guarda en la validación de los modelos

Para finalizar con todos los pasos que implica el proceso de minería de datos, nos tocaría presionar el botón para implementar los modelos en la interfaz principal. Con ello se apertura la interfaz 'Implementacion' y haciendo clic en el botón ubicado en los ejes de las abscisas de cada gráfico, en sus distintas escalas si así lo estima conveniente el usuario, permitirá elegir un archivo de datos que evaluará la función con cada uno de los parámetros validados para la inferencia.

Para visualizar las acciones de esta interfaz hemos diseñado como ejemplo un archivo de datos de una sola columna con el nombre de 'ParaImplementar' tal y como se muestra en la figura 4.8.

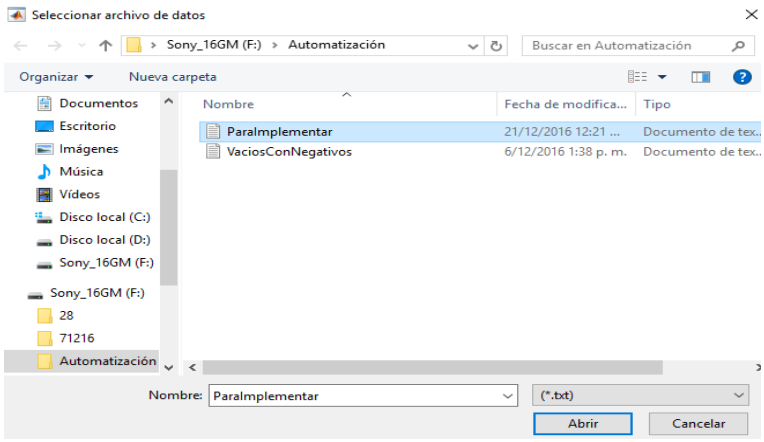


Figura 4.8: Archivo para implementar los modelos en sus distintas escalas

Los resultados después de presionar los botones de cada eje en los gráficos se pueden

visualizar en la figura 4.9, donde se representan en círculos azules la linealidad de los datos inferidos en sus distintas escalas. Al presionar el botón para salvar los datos inferidos, el usuario dispondrá de un archivo llamado 'Implementacion_Inferida' con la extensión *.mat* en donde se ubican los datos inferidos en una matriz de cuatro columnas.

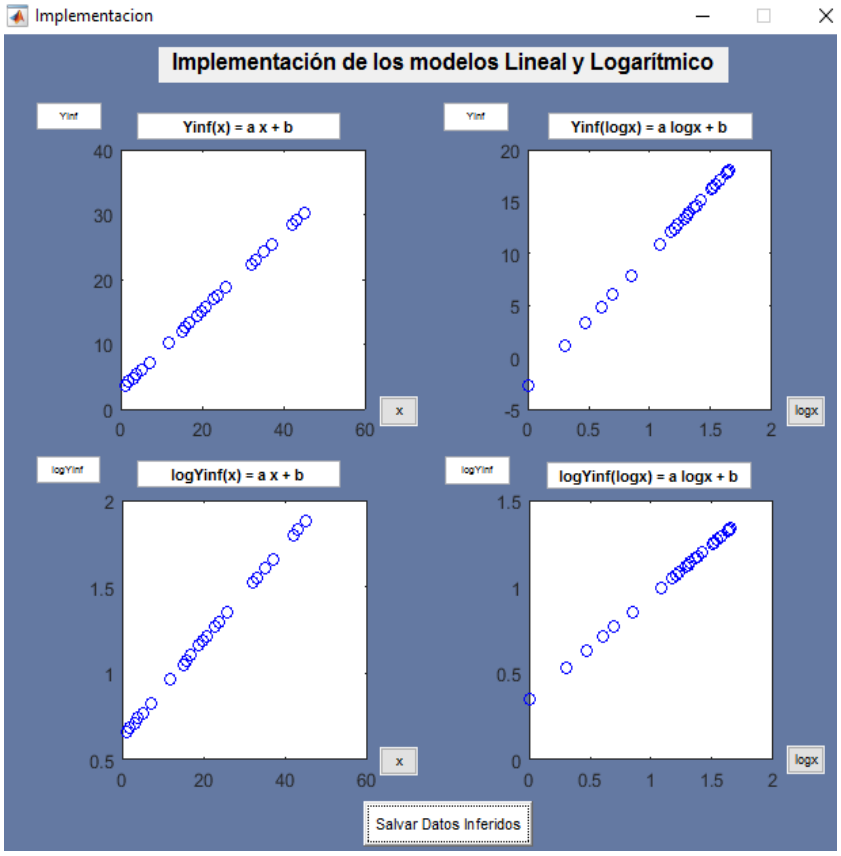


Figura 4.9: Implementación de datos asociados al mismo fenómeno físico

4.2. Modelo No-Lineal

Hemos discutido anteriormente que para hacer uso del módulo de ANFIS, basta con que el usuario presione el botón 'ANFIS' en la interfaz principal de este software. Por otro lado recordemos que el primer paso del proceso de minería de datos se realizará siempre en el panel de control anterior. Sin embargo, cuando se selecciona el archivo de datos a explorar se crean los distintos tipos de datos, que no es más que los valores (x, y) en las distintas escalas.

Bajo este esquema podemos explorar los distintos tipos de datos con el módulo de ANFIS de uno en uno. Tomemos como ejemplo los **DATOS_D**, ya que la correlación entre $log(x)$ y $log(y)$ es la más cercana al valor 1.

Para ello seleccionamos en el área de trabajo *Load data* las opciones *Training* de Type, y *worksp.* de From. Luego presionando el botón 'Load Data...' aparecerá una ventana donde escribiremos el nombre del tipo de dato que trabajaremos, en nuestro caso particular DATOS_D (ver figura 4.10). Presionando el botón 'OK' se graficarían los datos (ver figura 4.11).

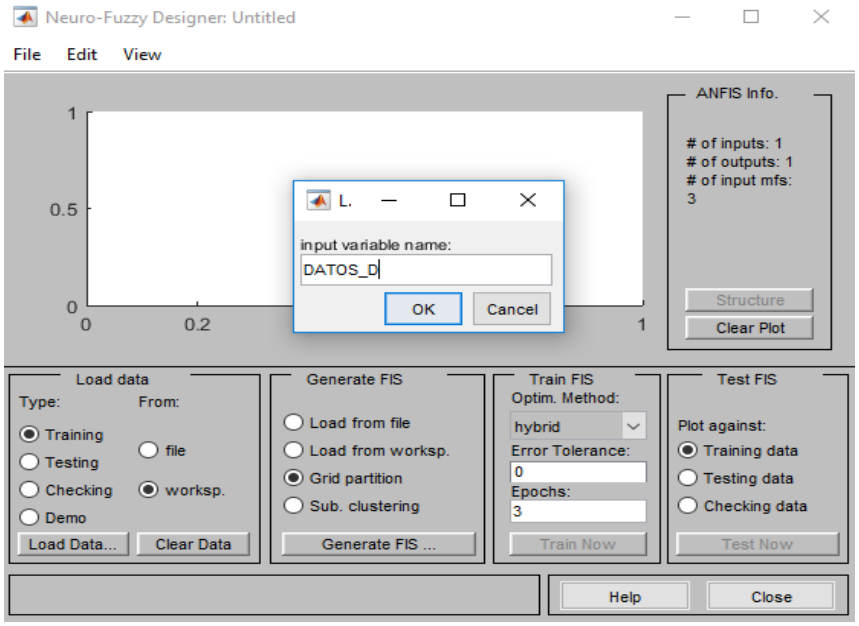


Figura 4.10: Tipo de dato a usar para el entrenamiento en el módulo de ANFIS

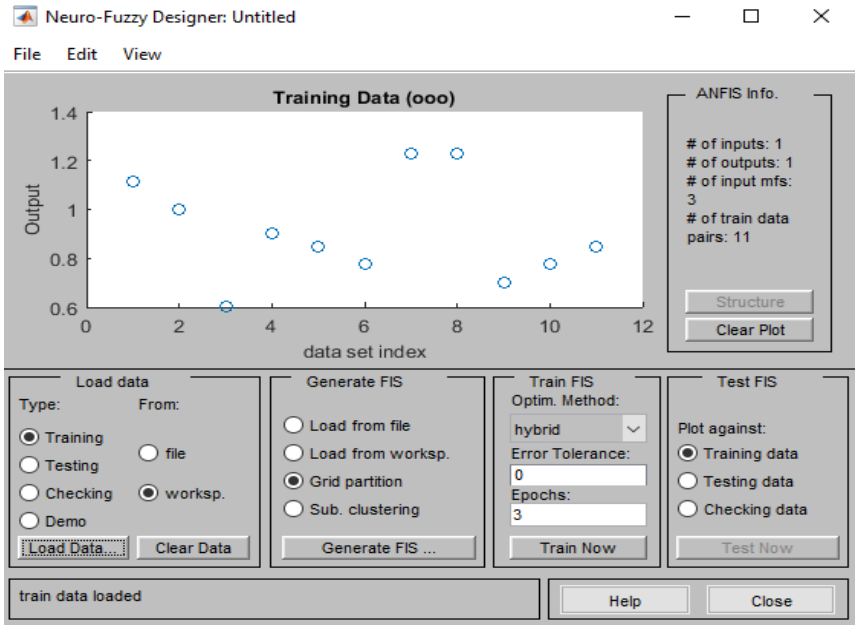


Figura 4.11: Gráfico de los DATOS_D en el módulo de ANFIS

Para generar el modelo FIS, seleccionamos del área de trabajo *Generate FIS* la opción 'Grid partition'. Luego haciendo clic en el botón 'Generate FIS...' escribimos el número de reglas en 'Number of MFs:' en nuestro caso 2, seleccionando como función de pertenencia la función *gauss2mf* en 'MF Type:', todo esto se aplicará a los datos de entrada (INPUT). A la salida (OUTPUT) la función de pertenencia elegida en 'MF Type:' es *linear* (ver figura 4.12). Con estas elecciones presionamos el botón 'OK'.

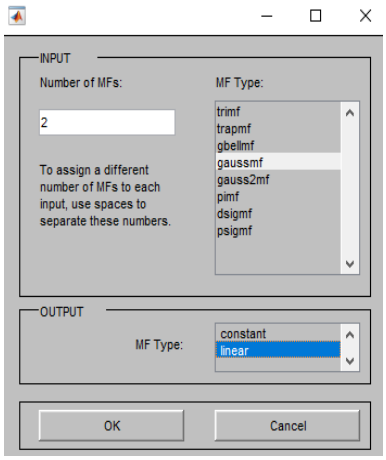


Figura 4.12: Número de reglas y funciones de pertenencia de entrada y salida del modelo FIS

Para poder utilizar el modelo FIS generado nos ubicamos en el área de trabajo *Train FIS*. Como primer paso elegimos el método de optimización *hybrid* (combinación de mínimos cuadrados) en 'Optim. Method:'. Luego escribimos el 'Error Tolerance:' y la cantidad de iteraciones para el entrenamiento. Presionando el botón 'Train Now' se puede observar cómo se van procesando los datos de manera gráfica, así como va cambiando el valor RMSE durante las iteraciones (ver figura 4.13).

Por otro lado se puede ir visualizando como en el workspace de MatLab se van mostrando una serie de parámetros durante el entrenamiento. El módulo de ANFIS te permite comparar los datos del entrenamiento con los datos seleccionados originalmente. Para ello elegimos la opción *Training data* en 'Plot against:' del área de trabajo 'Test FIS' y presionando el botón 'Test Now', obteniendo la gráfica de la figura 4.14.

Finalmente al presionar el botón 'Close' y seleccionar la opción *Yes* del cuadro del diálogo, guardamos el archivo con el nombre 'Exploracion_ANFIS_D01.fis'.

A continuación presionamos el botón 'Validar el Modelo' de la interfaz principal, con el que podemos elegir el archivo de datos *.fis*. Sabemos qué en la interfaz para validar el modelo No-Lineal al seleccionar en el eje de las ordenadas la variable independiente obtendríamos los valores inferidos de este modelo con los parámetros obtenidos del módulo de ANFIS a través del archivo seleccionado.

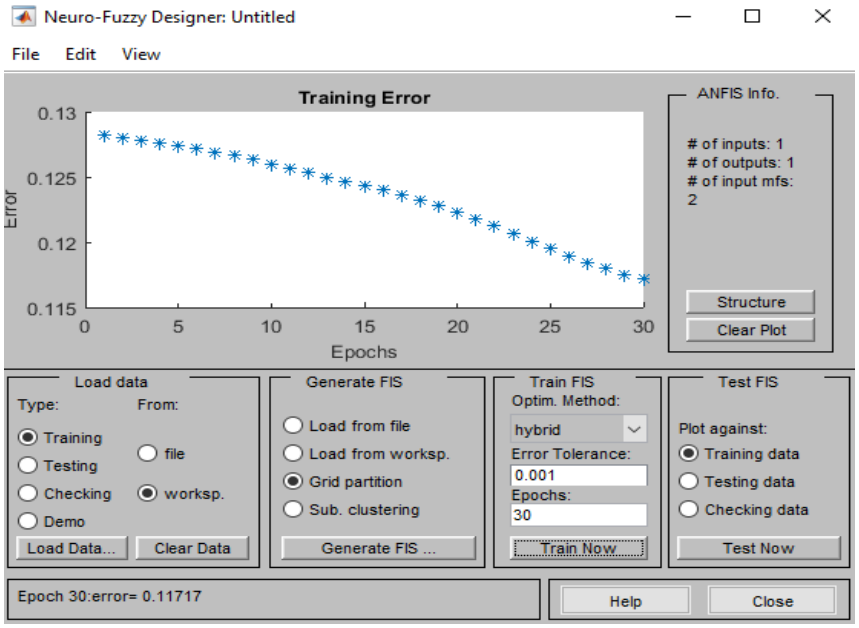


Figura 4.13: Entrenamiento del modelo FIS generado en el módulo de ANFIS

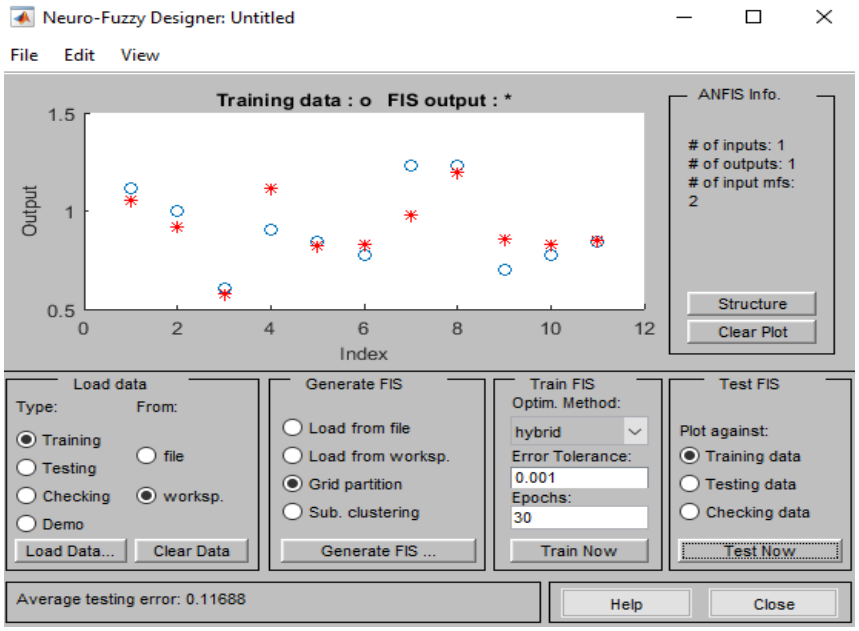


Figura 4.14: Comparación de los datos entrenados y los datos seleccionados en el módulo de ANFIS

En nuestro caso la función de inferencia tiene como variable independiente el $log(x)$. Luego evaluamos los valores inferidos escogiendo en el eje de las abscisas el $log(y)$ con el cual se describe, al igual que en la validación del modelo anterior, gráficamente y estadísticamente la linealidad o no de los datos procesados (ver figura 4.15). Para guardar los valores estadísticos presionamos el botón 'Salvar los Datos' los cuales se guardarán en

el archivo 'Validacion_EstadisticaNL'.

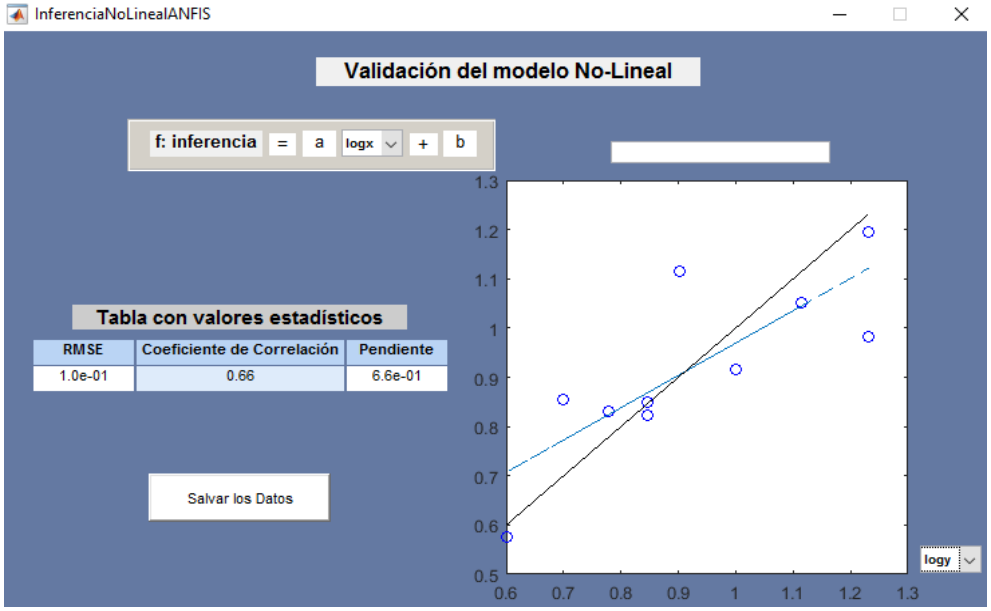


Figura 4.15: Validación de los datos generados en el módulo de ANFIS

Terminando con todo el proceso de minería de datos en este modelo le damos un clic al botón 'Implementar el Modelo' en la interfaz principal. Luego seleccionamos de la lista desplegable, en el eje de las abscisas, la variable *logx* para nuestro caso en particular. Con ello se apertura la ventana para elegir el archivo de datos que utilizados para el modelo anterior llamado 'ParaImplementar'.

En la figura 4.16 se muestra la interfaz gráfica para la implementación del modelo No-Lineal. De igual manera, para salvar los datos inferidos basta con presionar el botón 'Salvar los Datos' y se guardará un archivo vector columna con el nombre de 'Implementación_NoLineal'.

Si el usuario se percata que los datos elegidos no logran concretar información valiosa después de realizar el proceso de minería de datos, o en cualquier fase de la misma, tiene la opción de eliminar los datos y cerrar todas las interfaces gráficas abiertas hasta ese punto, con tan solo presionar el botón 'Eliminar Datos' de la interfaz principal. Entonces solo quedaría disponible para el usuario lo mostrado en la figura 4.17.

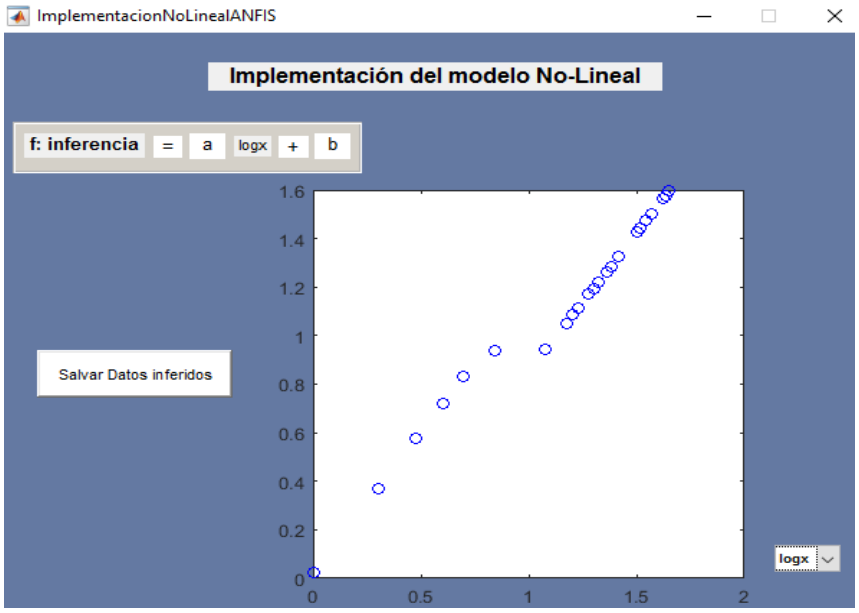


Figura 4.16: Implementación del modelo No-Lineal

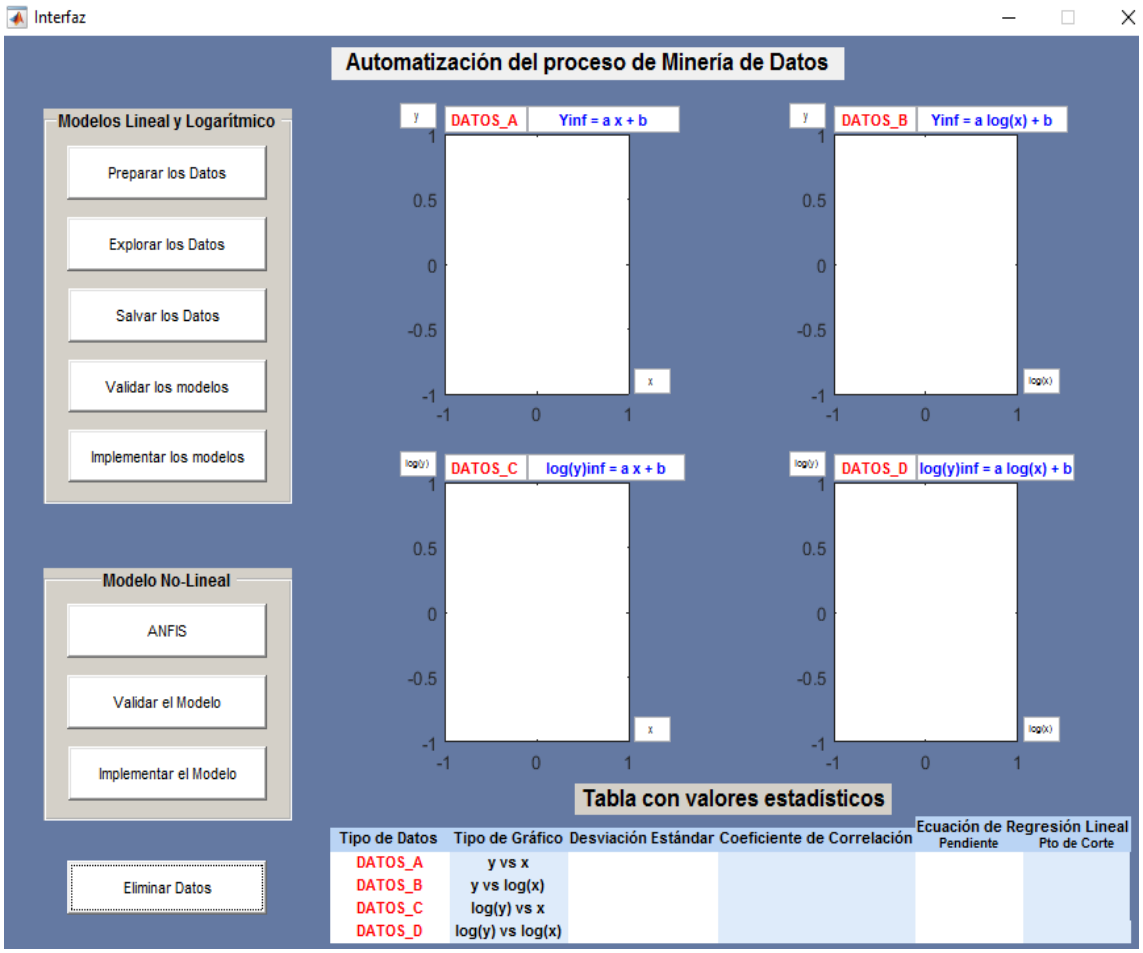


Figura 4.17: Interfaz principal después de eliminar los datos

Capítulo 5

Conclusiones

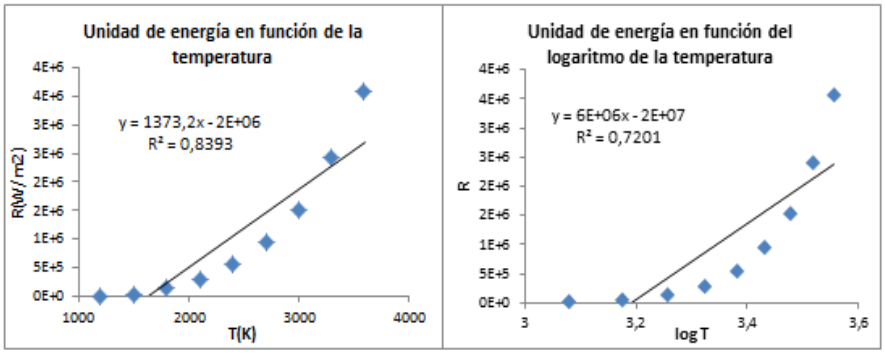
El objetivo principal de este trabajo es el desarrollo de una aplicación en donde se automatizan las distintas etapas del proceso de minería de datos, logrando de esta manera que el usuario pueda limpiar, filtrar y seleccionar datos provenientes de cualquier sistema o fenómeno físico, que luego podrá procesar utilizando modelos Lineal, Logartímico, No-Lineal, para finalmente ser usados por sistemas expertos. Con el desarrollo de esta aplicación hemos llegado a las siguientes conclusiones:

- Con herramientas de la interfaz gráfica de usuario (GUI) de MatLab se lograron diseñar siete módulos con distintos paneles de control que permitieron estudiar la tendencia y patrón de los datos procesados. Para ello cada módulo dispone de entornos gráficos que producen resultados estadísticos, entre los cuales se encuentran la correlación, la desviación estándar y el error cuadrático medio (RMSE). Cada interfaz permite guardar los resultados obtenidos.
- La aplicación permite preparar los datos que serán procesados a través de un módulo que posee cuatro filtros con su representación gráfica respectivamente.
- Los modelos utilizados en la aplicación son el método de los mínimos cuadrados para cuando el estudio de los datos posea una relación lineal simple o con escalas logarítmicas, y el módulo de ANFIS de MatLab para estudiar los datos, cuya relación sea no-lineal.
- En está aplicación se podrán incluir otros módulos que permitan al usuario incorporar otros modelos, como redes neuronales artificiales, lógica difusa y teoría fractal, entre otros.
- Cada una de las interfaces dispone de entornos gráficos en 2D y en escalas lineales o logarítmicas, que permiten visualizar y comparar los resultados obtenidos. Adicionalmente en los gráficos se cuenta con la posibilidad de determinar la tendencia y realizar los ajustes lineales.

Apéndice A

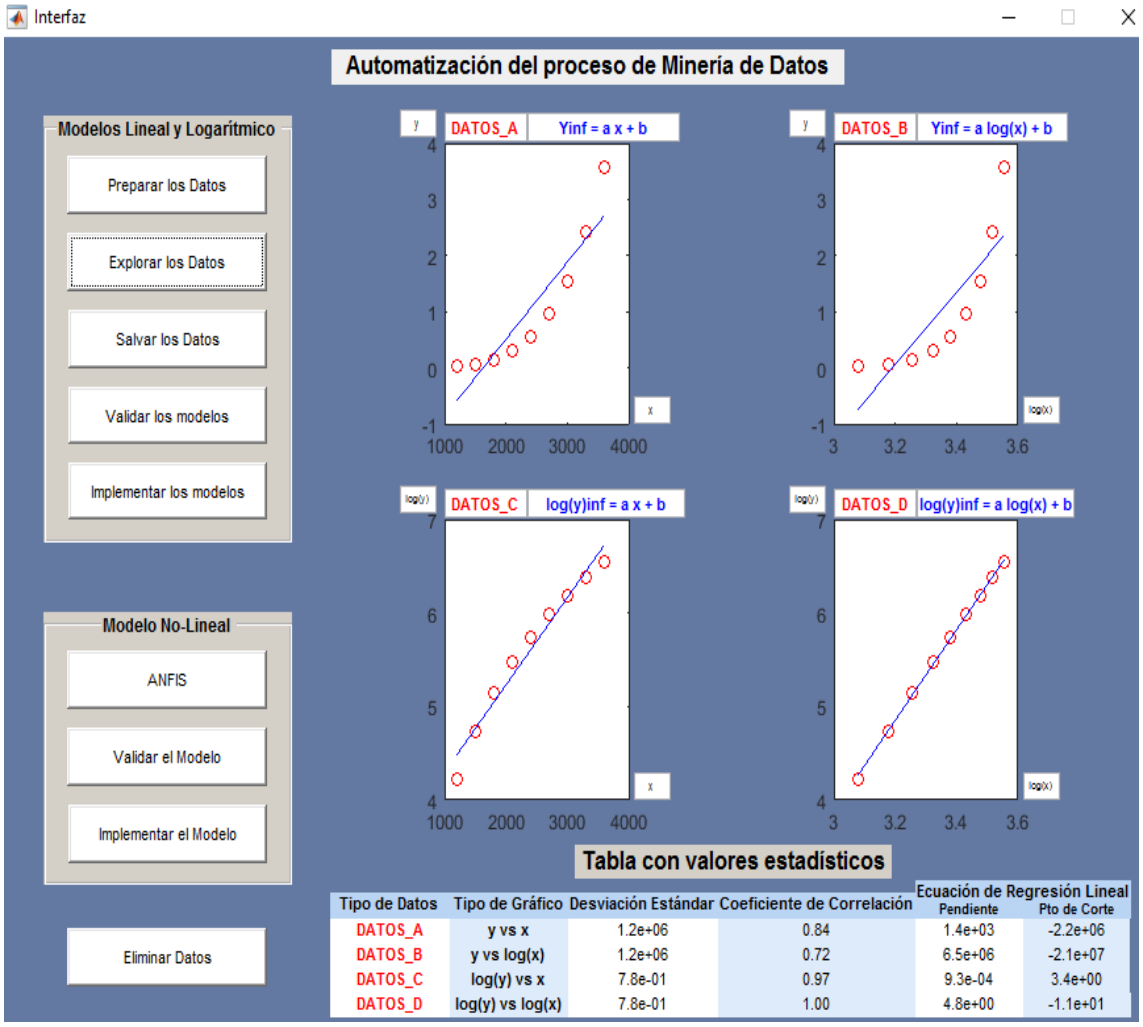
Aplicación evaluada

Resultados de datos procesados con hojas de cálculo de Microsoft Excel y con la aplicación desarrollada para su comparación, de la experiencia donde se mide la energía total radiada (R) por unidad de tiempo y por unidad de área de la superficie de un alambre de tungsteno como función de su temperatura (T):



Tipo de Gráfico	Desviación Estándar	Coeficiente de Correlación	Ecuación de regresión
x vs y	1,2E+6	0,84	(1,4E+3)x - 2E+06
log(x) vs y	1,2E+6	0,72	(6E+06)x - 2E+07
x vs log(y)	7,8E-1	0,97	(9E-4)x + 3,4
log(x) vs log(y)	7,8E-1	1,00	(4,8)x - 1,1E+1

Figura A.1: Datos procesados con Microsoft Excel en escalas Lineal-Logarítmica



Tipo de Datos	Tipo de Gráfico	Desviación Estándar	Coeficiente de Correlación	Ecuación de Regresión Lineal	
				Pendiente	Pto de Corte
DATOS_A	y vs x	1.2e+06	0.84	1.4e+03	-2.2e+06
DATOS_B	y vs log(x)	1.2e+06	0.72	6.5e+06	-2.1e+07
DATOS_C	log(y) vs x	7.8e-01	0.97	9.3e-04	3.4e+00
DATOS_D	log(y) vs log(x)	7.8e-01	1.00	4.8e+00	-1.1e+01

Figura A.2: Datos procesados con la aplicación desarrollada

Bibliografía

[1] Microsoft SQL Server (2014). *Conceptos de Minería de Datos.* https://msdn.microsoft.com/es-es/library/ms174949.aspx.

[2] A. Rosas y J. Zúñiga. *Correlación y Regresión Lineales.* Colegio de Bachilleres.

[3] I. Escalona y P. Chocron (2002). *Cálculo de errores gráficos.* UCV-Facultad de Ciencias, Departamento de Física.

[4] J. Valencia (2013). *Desarrollo de Software de Usuario para Sistemas de Control Predictivo No-Lineal Basado en Modelo Borroso.* Universidad Nacional de Colombia - Sede Medellín.

[5] F. Izaurieta y C. Saavedra. *Redes Neuronales Artificiales.* Departamento de Física, Universidad de Concepción, Chile.

[6] S. Haykin (2001). *Neural Networks A Comprehensive Foundation.* Prentice Hall PTR Upper Saddle River, NJ, USA. Segunda Edición.

[7] B. Martín y A. Sanz (2002). *Redes neuronales y sistemas difusos.* 2ª Edición. ©RA-MA Editorial.

[8] I. Alson y N. Hurtado (2010). *Ecuaciones de inferencia entre cociente S y susceptibilidad magnética del pozo Saltarín 1A, usando redes neuronales difusas.* Escuela de Física, Universidad Central de Venezuela, Venezuela.

[9] J. Chahuara (2005). *Control Neuro-Difuso Aplicado a una Grúa Torre.* Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería Electrónica, Perú.

[10] M. Ousslaah, H. Nguyen y V. Kreinovich. *A new derivation of centroide defuzzification*

[11] O. Galindo (2005). *Sistema de clasificación neurodifusa en imágenes de envases.* Universidad Centroccidental Lisandro Alvarado, Decanato de Ciencias y Tecnología.

[12] R. Jang (1993). *ANFIS: Adaptative-Network-Based Fuzzy Inference System.* IEEE Transactions on Systems, Man and Cybernetics, **23**.

[13] MatLab (2014). *Neuro-Adaptive Learning and ANFIS*

[14] R. Coronado, N. Hurtado y M. Aldana (2015). *Inferencia de Velocidades de Ondas S, mediante la Técnica de Redes Neuronales Difusas*. Boletín de Geología, **37**(2): 83-87

[15] E. Henriquez y N. Hurtado (2014). *Estudio numérico de permeabilidad utilizando redes neuronales artificiales*. Escuela de Física, Universidad Central de Venezuela, Venezuela.

[16] D. Zabaleta y N. Hurtado (2014). *Ecuaciones de inferencia de permeabilidad empleando lógica difusa y paris*. Escuela de Física, Universidad Central de Venezuela, Venezuela.

[17] J. Diaz y N. Hurtado (2014). *Inferencia de la permeabilidad en función de la porosidad y saturación de agua utilizando sistema neuro-difuso ANFIS*. Escuela de Física, Universidad Central de Venezuela, Venezuela.

[18] R. Diaz y N. Hurtado (2013). *Inferencia de permeabilidad en pozo mediante un sistema neuro-difuso*. Escuela de Física, Universidad Central de Venezuela, Venezuela.

[19] N. Hurtado, M. Aldana and J. Torres (2009). *Comparison between Neuro-Fuzzy and Fractal Models for Permeability Prediction*. Computational Geosciences **13**(2): 181-186

[20] J. Torres, N. Hurtado y M. Aldana (2007). *Comparación de tres técnicas distintas con datos reales de pozo, en la determinación de la permeabilidad*. Ciencia **15**(4): 433-437

[21] V. Camacho, D. López, V. Costanzo, M. Aldana, G. Bayona and N. Hurtado (2013). *A neuro fuzzy approach to recognize rock magnetic and lithological patterns in a stratigraphic well from the Llanos Foreland Basin (Colombia)*. Studia Geophysica et Geodetica **57**: 669-691

[22] N. Hurtado, V. Costanzo, D. López, M. Aldana and G. Bayona (2013). *Characterization of Lithostratigraphic Units using Neuro Fuzzy System Analyses Applied to Rock Magnetic Data*. In Proceedings of the 2nd International Conference on Pattern Recognition Applications and Methods 686-689

[23] D. Lopez y N. Hurtado (2009). *Caracterización de litofacies en el pozo saltarín 1A a través de la determinación experimental de parámetros magnéticos y su inferencia por medio de la técnica de redes neuronales difusas*. Escuela de Física, Universidad Central de Venezuela, Venezuela.

[24] D. Perez y N. Hurtado (2013). *Estudio neuro-difuso* $\delta^1 8/^O$ *y* $\delta^1 3/^C$ *en el límite triásico / jurásico*. Escuela de Física, Universidad Central de Venezuela, Venezuela.

[25] A. Da Silva, V. Costanzo, N. Hurtado, A. Milagrosa, G. Bayona, O. Guzmán and D. López (2010). *Possible correlation between miocene global climatic changes and magnetic proxies, using neuro fuzzy logic analysis in a Stratigraphic Well at the Llanos Foreland Basin, Colombia*. Studia Geophysica et Geodetica **54**: 607-631

[26] A. Da Silva, V. Costanzo, N. Hurtado, M. Aldana, G. Bayona (2010). *Correlation between miocene global climatic changes ($\delta^{18}O$) and magnetic properties using neuro fuzzy logic analysis*. Barcelona'10 Conference Proceeding's & Exhibitors Catalogue, P502 CDROM