PREDICCIÓN DE DEMANDA DE CORTO PLAZO EMPLEANDO REDES NEURONALES

CÉSAR YOBANY ACEVEDO ARENAS
Ingeniero Electricista

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
MAESTRÍA EN POTENCIA ELÉCTRICA

BUCARAMANGA
2004

PREDICCIÓN DE DEMANDA DE CORTO PLAZO EMPLEANDO REDES NEURONALES

CÉSAR YOBANY ACEVEDO ARENAS
Ingeniero Electricista

Trabajo de Investigación para optar al título de
Magister en Potencia Eléctrica

Director
GERARDO LATORRE BAYONA
Ingeniero Electricista, Doctor Ingeniero Industrial

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
MAESTRÍA EN POTENCIA ELÉCTRICA

BUCARAMANGA
2004

Nota de Aceptación

_____
_____
_____
_____
_____
_____

_____
Presidente del Jurado

_____
Jurado

_____
Jurado

Bucaramanga, Junio de 2004

A Dios,
A mi abuela Helena (q.e.p.d),
A mis padres David y Elsa,
A mi hijo Camilo Andrés.

AGRADECIMIENTOS

Al Doctor Gerardo Latorre Bayona, director del trabajo de investigación por su respaldo, confianza, paciencia y colaboración.

Al Doctor Jorge Luis Grosso Vargas, quien con su ejemplo y apoyo, me animó y fortaleció en la culminación de este trabajo de investigación.

Al Ingeniero John Jairo Sanabria, por su ayuda y paciencia en el desarrollo del prototipo.

A mis compañeros de la Maestría en Potencia Eléctrica, con quienes compartí, tanto momentos alegres como difíciles.

A mis familiares y amigos, quienes me motivaron de una u otra manera, a culminar con éxito esta etapa en mi vida.

César Yobany Acevedo Arenas

CONTENIDO

pág.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

LISTA DE ANEXOS

pág.

GLOSARIO

ANÁLISIS DE SERIES DE TIEMPO: Proceso mediante el cual una serie de tiempo es desagregada en sus cuatro componentes principales: Tendencia, ciclo, irregularidad y estacionalidad.

ALGORITMOS GENÉTICOS: Son algoritmos matemáticos aplicables a problemas de optimización, basados en la teoría de la evolución de Darwin, operando en un ciclo simple de selección y reproducción, implicando una recombinación y mutación del "material genético" de las soluciones. En ellos, una "población" de posibles soluciones se genera al azar, se evalúan con respecto a un objetivos y las más aptas se combinan entre sí para producir nuevas soluciones. El ciclo se repite hasta llegar a una solución aceptable o al determinarse el óptimo de una función.

CICLO DE LA SERIE DE TIEMPO: Es la componente de corto plazo, representada por un patrón que se repite a si mismo en forma periódica, durante toda la serie histórica. Su forma se debe al comportamiento en patrones de consumo de corto plazo (horas, días, semanas e incluso meses) y su fluctuación está alrededor de la tendencia.

CND: Centro Nacional de Despacho.

CREG: Comisión de Regulación de Energía y Gas.

ESSA: Empresa Electrificadora de Santander S.A.

ESTACIONALIDAD DE LA SERIE DE TIEMPO: Es un patrón de cambio regularmente recurrente a través del tiempo, el cual se presenta por lo general de

manera anual. (p.e. estaciones climáticas).

INTELIGENCIA ARTIFICIAL: Es el campo de la ciencia de la computación dedicado a analizar y desarrollar sistemas que reproduzcan e imiten los procesos de pensamiento y razonamiento del hombre.

IRREGULARIDAD DE LA SERIE DE TIEMPO: Es la variabilidad de la serie de tiempo una vez se retiran las componentes de tendencia, ciclo y estacionalidad. Está compuesta por sucesos "impredecibles" o no periódicos, asociándose en el caso de señales al ruido.

LÓGICA DIFUSA: Es un conjunto de técnicas matemáticas para la representación y tratamiento de datos que no tienen una precisión definida y concreta. La evaluación de este tipo de datos es a menudo una cuestión relativa (p.e. alto / bajo)

MÉTODO HEURÍSTICO: Resolución de problemas, probando diferentes métodos y comparando cual es el que ofrece la mejor solución.

MINERÍA DE DATOS: Son sistemas de búsqueda de conocimiento, tratando de determinar relaciones y patrones de comportamiento en bases de datos, de una forma inteligente y automática.

REDES NEURONALES (NEURALS NETWORKS): Son programas de Inteligencia Artificial capaz de simular algunas de las funciones de aprendizaje del ser humano. Sin reglas convencionales, una red neuronal obtiene experiencia analizando automática y sistemáticamente una cantidad de datos, para determinar reglas de comportamiento. Con base en estas reglas, puede realizar predicciones sobre nuevos casos. Estas técnicas se aplican a problemas de clasificación y series de tiempo. y ofrecen el potencial de identificar conexiones que otras técnicas no pueden, porque utiliza relaciones lineales y no-lineales entre los datos, puede trabajar con cualquier tipo de distribución (no solamente distribución

normal) y maneja datos con redundancia y/o inconsistencia en la información

SERIE DE TIEMPO: Variable cuyo registro en el tiempo se realiza de manera periódica y regular.

SISTEMAS EXPERTOS: Los sistemas expertos permiten el desarrollo de otros sistemas que representan el conocimiento como una serie de reglas. Las distintas relaciones, conexiones y afinidades sobre un tema pueden ser compilados en un sistema experto pudiendo incluir relaciones altamente complejas y con múltiples interacciones.

TENDENCIA DE LA SERIE DE TIEMPO: Es la componente de largo plazo que constituye la base del crecimiento (o declinación) de la serie histórica. A nivel general, las fuerzas básicas que producen o afectan la tendencia de la serie pueden ser: cambios en la población, inflación, cambio tecnológico e incremento en la productividad.

SÍNTESIS DE SERIES DE TIEMPO: Proceso mediante el cual se combinan las componentes de tendencia, ciclo, irregularidad y estacionalidad, para obtener la serie de tiempo.

RESUMEN

TÍTULO:
PREDICCIÓN DE DEMANDA DE CORTO PLAZO EMPLEANDO REDES NEURONALES*

AUTOR:
César Yobany Acevedo Arenas**

PALABRAS CLAVE:
Predicción de demanda de corto plazo, Series de tiempo, Identificación de sistemas, Redes neuronales

DESCRIPCIÓN:

El trabajo de investigación plantea una solución al problema de predicción de la demanda de potencia eléctrica horaria, a través de una metodología consistente en la descomposición de la serie de tiempo en sus cuatro factores principales (tendencia, ciclo, irregularidad y estacionalidad). Esta metodología se basa en el uso de la componente irregular de la serie de tiempo como dato de entrada al sistema de predicción. Una vez predicho el valor de la componente irregular, ésta vuelve a ser multiplicada con la extrapolación de la tendencia y el ciclo de la serie, las cuales son fácilmente determinables a través de procedimientos estadísticos y de identificación de sistemas.
A fin de probar las posibilidades que ofrece la metodología se realizaron pruebas con técnicas convencionales de predicción basadas en la identificación de sistemas tales como: Modelos Autorregresivos, Modelos Autorregresivos de Promedio Móvil, Espacio de estados y Metodología de Box & Jenkins. Los resultados obtenidos en tales pruebas, además de establecer ventajas comparativas de éstas técnicas convencionales frente a la utilización de redes neuronales en el problema de predicción de series de tiempo de corto plazo, sirvieron de base para el planteamiento de la metodología de predicción.
El trabajo de investigación presenta un estudio de las diferentes formas de modelado de las series de tiempo, técnicas y procesos de predicción, un análisis de las características de demanda y el planteamiento de un sistema de predicción soportado en una herramienta software que permite comprobar las hipótesis planteadas y solucionar deficiencias comunes encontradas en la revisión bibliográfica del estado del arte. La propuesta metodológica permitirá aprovechar al máximo la capacidad de la red neuronal en la predicción de la componente irregular (ruido) de la curva de demanda, dejando las otras componentes para ser predichas por separado mediante la simple extrapolación de una función matemática cuya determinación es más sencilla.

*Trabajo de Investigación.
**Facultad de Ingenierías Físico-Mecánicas, Escuela de Ingeniería Eléctrica, Electrónica y Telecomunicaciones, Ing. Ph.D. Gerardo Latorre Bayona.

SUMMARY

TITLE:
SHORT-TERM LOAD FORECASTING USING NEURAL NETWORKS*

AUTHOR:
César Yobany Acevedo Arenas**

KEY WORDS:
Short-term Load Forecasting, Time Series, Systems Identification, Artificial Neural Networks.

DESCRIPTION:

The research work outlines a solution to the problem of Short-term load forecasting in electric power systems, through a methodology that consist in the decomposition of the time series in its four main factors (tendency, cycle, irregularity and seasonality). This methodology is based on the use of the irregular component of the time series, like entrance to the prediction system. Once the value of the irregular component has been predicted, it is multiplied with the extrapolation of the tendency and the cycle of the series again, which are easily determined through statistical procedures and systems identification.

In order to demonstrate the possibilities that the methodology offers, tests were carried out with conventional techniques of forecasting based on the system identification such as: Autoregressive Model, Autoregressive Moving Average Model, State-Space Model, and Box-Jenkins Methodology. The obtained results in such tests, besides of establishing comparative analysis of the use of these conventional techniques versus the use of Neural Nets, provided the base for the proposition of the forecasting methodology.

The research work presents an analysis of the different ways of modeling of the time series, the forecasting techniques, the prediction process and load characteristics. In addition it proposes a prediction system supported in a software tool that makes it possible to check the outlined hypotheses and to solve common deficiencies that were found in the bibliographical revision of the state of the art. The proposed methodology takes the best advantage of the neural network capacity in the prediction of the irregular component (noise) of the load curve, leaving the other components to be predicted separately by means of the extrapolation of a mathematical function whose determination is simpler.

*Research Work.
**Physical-Mechanical Engineerings Faculty, School of Electrical, Electronic and Telecommunication Engineerings, Eng. Ph.D. Gerardo Latorre Bayona.

INTRODUCCIÓN

Planear en el corto plazo la operación de los recursos de generación, transmisión, interconexión y distribución en forma económica, teniendo como propósito la atención de la demanda con los criterios de seguridad, confiabilidad y calidad establecidos en la reglamentación vigente, es parte de la misión de los diferentes operadores de los sistemas eléctricos de potencia, en cada uno de los niveles de tensión.

Dentro de este proceso, la previsión de los valores de energía y potencia demandados por los clientes del sistema de energía eléctrica, es una de las variables de entrada de cualquier modelo de planeación operativa y comercial, tanto en el corto como en el largo plazo.

La motivación para el desarrollo de modelos que permitan una previsión del comportamiento de la demanda lo más acertadamente posible dentro del horizonte del corto plazo, se ha acentuado en las dos últimas décadas, dada la transformación liberalizadora de la mayoría de mercados de energía eléctrica del mundo (incluyendo el colombiano). En este entorno, las empresas comercializadoras compran la energía que consumen sus clientes, ya sea mediante contratos con las empresas generadoras (en horizontes de tiempo cada vez más cortos), y/o mediante transacciones en la bolsa de energía donde el precio depende del despacho horario de unidades generadoras.

Para un comercializador, las oportunidades de gestión en búsqueda de transacciones de energía a precios favorables, están dadas por la toma de decisiones adecuadas en la compra de energía dentro de las modalidades anteriores (en contratos y/o en bolsa de energía). Dado un precio fijo al cliente final, el margen de comercialización dependería entonces de la manera como éste "juegue" dentro de estas dos modalidades de compra de energía; si el

comercializador conoce exactamente la demanda de sus clientes, podrá sacar mayor provecho en sus transacciones, ya que en algunos momentos el precio de los contratos puede ser superior al de la bolsa de energía, siendo aconsejable comprar la mayor cantidad de energía en contratos; o bien, ocurrir el fenómeno contrario, siendo entonces aconsejable comprar la mayor cantidad posible de energía en la bolsa.

Otro aspecto que justifica la predicción de la demanda de potencia horaria, es de carácter operativo en la red de distribución. Opciones como la reconfiguración, automatización del sistema de distribución, y otras alternativas de gestión del sistema de distribución como la gestión de carga y la gestión de demanda, está íntimamente relacionadas con la manera como se consume la energía por parte de los clientes, reduciendo de esta manera la incertidumbre en la efectividad de las acciones tomadas, asociada a los flujos de potencia reales por la red. Además de lo anterior es un requerimiento regulatorio que cada una de las empresas dentro de las diferentes áreas operativas del Sistema Interconectado Nacional (SIN) envíen información al Centro Nacional de Despacho (CND) declarando la demanda diaria para la semana siguiente, lo que hace necesaria la predicción por razones mandatorias.

Estas razones, hacen que la predicción de demanda se convierta en una actividad cuyo interés se centra principalmente en el respaldo informativo a la operación técnica y comercial de los sistemas eléctricos de potencia. El trabajo de investigación realizado en torno al tema de predicción de demanda de la potencia horaria planteado en este documento, presenta una alternativa que emplea técnicas de predicción basadas en redes neuronales, cuya novedad dentro del estado del arte radica en la utilización del concepto de descomposición de series de tiempo y el empleo de unidades de pre-procesamiento de datos, las cuales se basan en técnicas estadísticas e identificación de sistemas, permitiendo de esta manera generar un prototipo que permite realizar predicciones de demanda, acordes a las necesidades de predicción de las empresas del Sector

Eléctrico Colombiano, e incluya aspectos propios de nuestra cultura como los patrones de consumo y las festividades (tanto religiosas como por Ley Emiliani [1]).

El trabajo de investigación propuesto, intenta en primera instancia, realizar un análisis comparativo de las técnicas convencionales de predicción e identificación de sistemas, frente a las redes neuronales artificiales, con el fin de encontrar ventajas o desventajas de cada una de estas técnicas, que permitan hacer un uso más adecuado de ellas en los procedimientos o metodologías de predicción posteriores.

Dadas las ventajas reportadas en la bibliografía recopilada de las Redes Neuronales Artificiales (RNAs), tales como su capacidad de aproximación de funciones “matemáticamente desconocidas”, su alta tolerancia al ruido, adaptabilidad y capacidad de generalización ante nuevos ejemplos, se plantea la posibilidad de emplear esta técnica en la predicción de demanda de energía eléctrica en el corto plazo, como alternativa ante métodos convencionales de predicción, planteando un esquema de solución, acorde con las necesidades de predicción de las empresas del sector eléctrico colombiano.

Finalmente se presentará una herramienta computacional prototipo que permita aplicar la técnica de predicción desarrollada, probando el modelo con datos de demanda agregada regulada y no regulada de la Electrificadora de Santander S.A. E.S.P (ESSA).

[1] Por "Ley Emiliani" se conoce la modificación del Honorable Consejo de Estado al artículo 62 de la Ley 4ª de 1913 (Art. 70 C.C.), en auto de febrero 26 de 1982, donde se traslada al siguiente lunes los días festivos.

1. ASPECTOS GENERALES SOBRE LA PREDICCIÓN DE LA DEMANDA HORARIA DE POTENCIA ELÉCTRICA

La predicción de la demanda de potencia en el corto plazo cubre intervalos que van desde una hora hasta una semana. Su aplicación dentro de los procedimientos de planeación operativa y toma de decisiones en las transacciones comerciales del mercado de energía en bloques, hacen que la estimación de la carga del sistema en cada hora del día, lo más acertadamente posible, sea fundamental para una operación económica del sistema tanto técnica y comercialmente hablando. Algunas de las razones por las cuales es necesario tener esquemas de predicción con exactitud, ya han sido expuestas en la introducción de este documento y son complementadas a continuación.

1.1 JUSTIFICACIÓN

El problema de predicción de la curva de demanda de potencia hora a hora durante el día varía según el grado de agregación. En Colombia la aplicabilidad principal de modelos de predicción en el corto plazo, están enfocados en tres sentidos: Estimación de la carga total del Sistema Interconectado Nacional (SIN) para el despacho económico, seguro y confiable de unidades de generación; Previsión de la demanda atendida por un comercializador para efecto de sus transacciones comerciales de energía y potencia; y Previsión de la demanda a nivel de alimentadores dentro del sistema de distribución, para implementación de opciones de gestión técnica en la red por parte del operador de la red de distribución como reconfiguración, automatización y gestión de carga.

Desde el punto de vista del operador del sistema, el conocimiento a priori de la

carga con un margen de error mínimo permite, en primera instancia, establecer de una forma más acertada los requerimientos de generación para cumplir con las condiciones de seguridad y confiabilidad del mismo, ayudando de esta manera, al establecimiento de puntos vulnerables del sistema y las posibles acciones correctivas para una condición de demanda dada. Adicionalmente, dentro de las funciones del despacho de generación, el conocimiento preciso de la demanda permite estimar con menor incertidumbre las necesidades de reservas operativas y para un determinado nivel de confiabilidad del sistema, enganche de unidades generadoras y horarios de mantenimiento. En este orden de ideas, la información obtenida de previsiones de la demanda en el horizonte operativo de corto plazo con alto grado de certeza, tendrá efectos benéficos en la reducción de los riesgos asociados con la incertidumbre y en la disminución de las diferencias entre el despacho ideal y el real, permitiendo entonces, un despacho más económico y confiable, evitando sobrecostos para el sistema en conjunto por: redespacho de unidades, escasez de combustibles, uso no planeado de unidades de alto costo o arranque innecesario de unidades generadoras por sobreestimación de la carga del sistema.

En este nivel, el comportamiento de la demanda para labores de despacho es de carácter agregado, y el efecto del consumidor individual, lo cual genera una componente aleatoria en la serie de tiempo puede llegar incluso a ser despreciable. Adicionalmente variables estacionales como la época lluviosa o la época de sequía pueden llegar a tener influencia en la predicción, así como comportamientos de variables económicas, dado el aporte del sector industrial.

Desde el punto de vista del comercializador, las reglas del mercado de energía colombiano, hacen que la mayoría de agentes comercializadores encargados de la compra o venta de energía, tiendan a suplir las necesidades de energía de sus clientes a través de contratos libres con los agentes generadores, dejando la mayoría de veces los excedentes (por encima o por debajo de su demanda pronosticada) para ser transados en la Bolsa de energía donde la volatilidad en los

precios aumenta el riesgo de las transacciones comerciales. Según la resolución 024 de 1.995 de la CREG[2], existen tres modalidades de contratos que los comercializadores pueden realizar con los generadores, estos son: Pague lo contratado, Pague lo contratado-condicional y Pague lo demandado. En el primero, el comercializador se compromete a pagar toda la energía contratada, independiente de que ésta sea consumida o no. Si el consumo es mayor que la energía contratada, la diferencia se paga al precio de la Bolsa. Si el consumo es menor que la energía contratada, este excedente se le paga al comercializador al precio de la Bolsa. En la modalidad Pague lo contratado-condicional se da el mismo tratamiento que en el caso anterior, con la diferencia de que este contrato sólo se "despacha" si, con base en el precio (orden de méritos), se requiere total o parcialmente para atender la demanda del comercializador. Por último, en la modalidad Pague lo demandado, el comercializador solamente paga (a precio de contrato) su consumo, siempre y cuando éste sea inferior o igual a la cantidad de energía contratada (Tope máximo). Si el consumo es superior, la diferencia se liquida al precio de la Bolsa. Lo anterior implica que para un agente comercializador es benéfico prever su demanda de energía, ya que esto permite realizar sus transacciones con una menor incertidumbre y tener mejores márgenes de comercialización al "jugar" aventajadamente con los precios de la bolsa y los de contratos de corto plazo con agentes generadores individuales.

No obstante, la dinámica regulatoria tiende hacia un mercado mucho más dinámico, en donde cada vez los contratos se realizan en horizontes de tiempo más corto y el agente regulador genera penalizaciones cuando la diferencia entre la demanda estimada y la demanda real, sobrepase límites establecidos. Esto permite una mayor transparencia y economía en el despacho del sistema interconectado nacional, ya que por una parte se evita que los comercializadores declaren una demanda inferior o superior a la demanda que realmente tendrán,

[2] CREG: Comisión de Regulación de Energía y Gas

para tomar ventaja de la diferencia entre el precio de bolsa y contratos; y por otra, al declarar la demanda real, el despacho se realiza de acuerdo a los requerimiento reales de energía en el sistema, permitiendo un despacho óptimo de unidades generadoras y menor incertidumbre en el precio de bolsa.

El trabajo de investigación propuesto, plantea un esquema general de predicción de curvas de demanda diaria, que podría ser empleado en diferentes ámbitos del negocio del sector eléctrico colombiano. La propuesta debe entonces enfocarse en las necesidades comunes de predicción de las empresas del sector eléctrico colombiano, las cuales plantean diferencias respecto a las encontradas en el estado del arte a nivel internacional.

En el corto plazo, el comportamiento de la demanda de energía eléctrica depende principalmente de tres factores: socioculturales, ambientales y aleatorios; el último se atribuye al consumidor individual, y tiene menor incidencia en la medida que el grado de agregación aumenta.

Una revisión de la bibliografía pertinente al tema de predicción de demanda, evidencia una concordancia entre los autores, acerca de las características deseables que los mecanismos empleados por las empresas para realizar sus estimaciones, deben poseer. Estas características son:

- Adaptabilidad a los cambios en el comportamiento de la demanda.
- Identificación de condiciones anormales de operación del sistema eléctrico dentro de la información de entrada.
- Realimentación continua con los valores reales medidos para mejorar el modelo de estimación.
- Rapidez de cálculo (especialmente en la toma de decisiones de operación tanto del despacho de unidades generadoras como de acciones sobre la topología de la red).

- Manejo de variables externas de incidencia en el consumo (como temperatura, duración del brillo solar en la zona, etc.).

- Consideración de efectos en la demanda por situaciones socioculturales tales como temporadas vacacionales o días festivos.

- Filtrado de señales de demanda no acordes con el comportamiento natural (ruido).

No obstante, uno de los aspectos comunes del desarrollo de esta temática a nivel internacional, radica en que los modelos desarrollados responden a problemáticas específicas en cada uno de los entornos de predicción y su aplicabilidad dentro del sistema. Al centrar el estudio dentro del contexto de las empresas del sector eléctrico colombiano, surgen expectativas adicionales que el estado del arte no soluciona, o que por el contrario no tienen incidencias en la demanda de nuestro entorno. Por ejemplo:

- Por ser desarrollados en países con estaciones, en la mayoría de modelos de predicción la componente estacional de las series de tiempo es fundamental, debido al cambio en el patrón de consumo en épocas de verano e invierno predominantemente. Colombia no las posee, y el efecto de las épocas secas y lluviosas es despreciable en cuanto a la correlación con la variable demanda.

- La consideración de aspectos sociales dentro del consumo son específicos del país donde se desarrolle el modelo. En nuestro caso el calendario litúrgico que define la semana santa y otras fiestas religiosas, así como, el traslado de la mayoría de festividades al siguiente Lunes, le adiciona particularidades al modelo planteado, ya que es necesario establecer de antemano, si el día que se va a predecir es festivo o hábil.

- En el caso específico de la demanda de la Electrificadora de Santander S.A., sobre la cual se basó el modelo de predicción, variables climáticas como la

temperatura, el brillo solar y la humedad, no tienen correlación con los datos de la demanda, como se demostró en [Rodriguez-2000]. Esto quiere decir que la demanda responde más a un modelo autorregresivo que a uno de regresión múltiple. Una explicación al respecto, es que la incidencia de la temperatura y humedad sobre la demanda de energía, está básicamente relacionada con la presencia de sistemas de aire acondicionado. Por otra parte, el brillo solar en la región de Santander es bastante constante durante todo el año, lo cual hace que sea un patrón cíclico y por tanto no produzca variaciones en el perfil.

- Aún si la incidencia de las variables climáticas no fuera despreciable, la obtención de esta información en nuestro entorno es complicada, ya que no siempre se poseen puntos de medición donde se tienen los centros de carga, además de la demora en la obtención de datos, que para efectos de trabajos anteriores, ha llegado a ser de casi tres meses debido a tramites burocráticos entre las entidades. Esta razón motivó a plantear un modelo predominantemente autorregresivo, el cual solo depende de datos históricos de demanda, más aún considerando que el modelo puede ser empleado en cualquier región e incluso a nivel nacional (donde la demanda agregada tendría efectos de diferentes zonas climáticas).

Otras características a tener en cuenta en el planteamiento de esquemas de predicción, pueden surgir cuando éstos son aplicados en el contexto del mercado de energía colombiano donde, por ejemplo: según la Resolución CRG-025/1995, la predicción horaria de la demanda para el Despacho Económico se efectúa por áreas operativas y para cada una de las 24 horas de cada día de la semana. Esta predicción de demanda de potencia la efectúa el CND y la envía a las empresas semanalmente el día miércoles y recibe comentarios o modificaciones hasta el día Viernes a las 13:00 horas. Esto implica intervalos semanales de predicción, iniciando en cierto día de la semana. Además, puede ocurrir que la información necesaria para la etapa de producción, llegue en forma tardía, luego el esquema de predicción debe tener en cuenta estos retardos cuando se ingresa la

información.

El desarrollo de diferentes técnicas de predicción, basadas en conceptos estadísticos, determinísticos, de identificación de sistemas y de inteligencia artificial, han hecho del tema de investigación, un área de amplia variedad de esquemas a ser implementados en herramientas computacionales que ofrezcan algunas de las características mencionadas anteriormente. Adicionalmente, consideraciones propias de este tipo de problemas, tales como horizonte de predicción, datos disponibles, inclusión de variables externas dentro del modelo y exactitud requerida entre otras, hacen que algunas de estas técnicas tengan mayor o menor efectividad en el resultado final, una vez se han implementado. Este conocimiento, alrededor de las diferentes técnicas de predicción plantea la posibilidad de emplear esquemas híbridos, que aprovechen características individuales de otras técnicas. A continuación se presentan algunos avances en el estado del arte que permiten explorar posibilidades en cuanto al planteamiento metodológico de este trabajo de investigación.

1.2 ESTADO DEL ARTE

Hasta antes de 1927, los pronósticos se realizaban simplemente extrapolando la serie en el tiempo. El principio de lo que se puede llamar “predicción moderna” de series temporales se puede fijar en este año, cuando Yule inventó la técnica autorregresiva para pronosticar el número anual de manchas solares [Yule-1927]. Su modelo pronosticaba el siguiente valor como una suma ponderada de las observaciones previas de la serie.

Para poder obtener un comportamiento interesante de sistemas lineales como ese, era necesario asumir la intervención de un factor externo (ruido), que afectaba al sistema lineal. Durante el medio siglo siguiente, se asumía que las series temporales eran generadas por sistemas lineales afectados por un ruido, y

todas las investigaciones culminaron en la metodología ARIMA[3] de Box-Jenkins [Box-1970].

Sin embargo, existen casos simples para los que esta última metodología es poco adecuada. El hecho de que series temporales aparentemente complicadas puedan ser generadas por ecuaciones muy simples, hace necesario un marco teórico mucho más general para el análisis y predicción de series temporales. Así fueron surgiendo trabajos en los que se trataban series no estacionarias y/o no lineales, con nuevos métodos: modelos bilineales, biespectrales, de umbral, etc. [Priestley-1988];[Tong-1990].

En los años 80, ocurrieron dos acontecimientos cruciales en la evolución de los estudios sobre series temporales. Por un lado, el incremento en la potencia de los computadores personales, permitió el estudio de series temporales mucho más largas, la aplicación de algoritmos más complejos, y la visualización interactiva tanto de los datos como de los resultados. El segundo hecho fue el desarrollo de las técnicas de aprendizaje automático (inteligencia artificial) y concretamente de las Redes Neuronales Artificiales.

Las Redes Neuronales Artificiales (RNAs) son modelos matemáticos inspirados en la organización y el funcionamiento de las neuronas biológicas. Existen númerosas variantes de RNAs que están relacionadas con la naturaleza de la tarea que se ha asignado. De la misma manera, también existen distintas variaciones sobre cómo modelar la neurona; en algunos casos se asemejan mucho a las neuronas biológicas mientras que en otros, los modelos son muy diferentes.

La literatura sugiere algunas características de las RNAs que las hacen especialmente interesantes en su aplicación a la previsión de series temporales. Fundamentalmente se señalan dos:

---

[3] ARIMA: Autoregresive Inegrated Moving Average - Modelo autoregresivo integrado con promedio movil.

- la capacidad de las RNAs de aproximar prácticamente cualquier función (incluso las no lineales).

- la posibilidad de hacer aproximaciones “piece-wise” o por trozos, de las funciones.

Desde el punto de vista matemático, las RNAs se pueden considerar como aproximadores universales de funciones. Esto significa que pueden automáticamente aproximar la función que mejor se ajuste a los datos, permitiendo de esta manera extraer relaciones cuando las funciones son muy complejas. Además las RNAs son intrínsecamente no lineales [Rumelhart-1986], lo cual implica no sólo que pueden estimar correctamente funciones no lineales, sino que también pueden extraer elementos no lineales de los datos, una vez extraídos los términos lineales.

Por otro lado, una RNA con una o más capas ocultas, puede dividir el espacio muestral automáticamente y construir diferentes funciones en diferentes porciones del espacio. Esto significa que las redes neuronales poseen la capacidad de construir modelos no lineales “piece-wise”. En [Collopy-1992], se revisaron las opiniones de expertos en previsión, que coinciden en afirmar la importancia de contar con modelos de estas características, es decir, capaces de identificar y tratar cambios abruptos en los patrones de la serie temporal.

Algunos métodos estadísticos de series temporales tienen limitaciones debidas a la forma en que los modelos son estimados, por esta razón, la estimación de muchos tipos de modelos de series temporales requieren la intervención y supervisión humana. Además, la mayoría de modelos estadísticos deben ser re-estimados periódicamente cuando se dispone de nuevos datos. Por el contrario, la estimación con RNAs puede ser automatizada [Hoptroff-1993], y no es necesario revisar los modelos puesto que las redes aprenden de forma incremental.

En lo que respecta a la predicción de la demanda horaria de potencia eléctrica, el avance en cuanto al trabajo investigativo de la temática en las últimas dos décadas, se debe en primera instancia, a la transformación liberalizadora de la mayoría de los mercados de energía eléctrica del mundo [Gómez-1999], y en segunda, al auge de modelos basados en inteligencia artificial e identificación de sistemas, apoyados a su vez por el desarrollo en la capacidad de cálculo de los computadores de hoy en día.

El estado del arte en cuanto a aplicación de las redes neuronales al estudio de diferentes tópicos en sistemas de potencia es bastante amplio [ElSharkawi-1996], entre ellos, la predicción de demanda, la evaluación de la estabilidad del sistema y la detección de fallas, ocupan los primeros lugares en cuanto al desarrollo de investigaciones que empleen redes neuronales artificiales para la solución de los problemas en estas áreas.

En cuanto a la predicción de demanda, las RNAs se presentan como una variante de gran aplicabilidad a la solución de problemas de predicción, debido a su capacidad intrínseca como aproximadora de funciones matemáticamente desconocidas y clasificadora de patrones. Adicionalmente, su alta inmunidad al ruido, capacidad de auto-adaptación y tolerancia a fallos, las convierten en una herramienta con ventajas frente a los métodos convencionales de predicción [Lippmann-1987], [Hammerstrom-1993], [Haykin-1999].

Son muchas las publicaciones referentes al tema de predicción de la demanda en el corto plazo. Como punto de inicio en el estudio del tema, en [Gross-1987] se presenta un estado del arte en la predicción de demanda de corto plazo y una descripción detallada de los diferentes tipos de modelado de la carga del sistema y técnicas de predicción; adicionalmente, se presentan una serie de recomendaciones prácticas y aspectos claves en el diseño de esquemas de predicción. En cuanto a las técnicas de predicción, la bibliografía recopilada muestra avances en métodos convencionales como en [Juberias-1999] y [Perry-

1999], donde se trabaja con modelos autorregresivos integrados con promedio móvil (ARIMA – AutoRegressive Integrated Moving Average) y modelos de regresión lineal múltiple (MLR – Multiple linear regression) para predicción de demanda de corto plazo. Lo anterior muestra que a pesar del gran desarrollo de técnicas basadas en inteligencia artificial, de uso generalizado en nuestros días gracias a las capacidades computacionales, es notorio también el desarrollo de los esquemas convencionales de identificación de sistemas que los ponen en un nivel de competitividad similar a las anteriores. Un aporte al mejoramiento de estas técnicas de identificación de sistemas lo constituye el alto desarrollo alcanzado por herramientas matemáticas como el MATLAB [Ljung-1994],[Ljung-1997].

Sin embargo, es indudable, que la mayor atención en cuanto a técnicas de predicción de demanda, la han logrado los esquemas que emplean inteligencia artificial. Su desarrollo en las últimas décadas ha sido notable dadas las aplicaciones posibles en este campo y el desarrollo de los equipos de computación. Por lo general, al hablar de inteligencia artificial aplicada a predicción de demanda, se piensa en Redes Neuronales dadas las ventajas presentadas al emplearlas dentro de esquemas de predicción, las cuales fueron mencionadas anteriormente. No obstante, otros esquemas de inteligencia artificial tales como sistemas expertos [Ho-1990], [Sharaf-1993], o esquemas híbridos pueden ser encontrados en la bibliografía referente al tema. Entre estos esquemas híbridos se pueden encontrar combinaciones de Redes neuronales con otras técnicas, como por ejemplo, sistemas de lógica difusa [Srinivasan-1996],[Shan-1997] o algoritmos genéticos [Heny-1998].

Algunos textos de consulta como [Granger-1989] y [Hanke-1996], para predicción de demanda a nivel general, así como [Freeman-1993] y [Haykin-1999] para redes neuronales, son una buena base de estudio para el trabajo de conceptualización inicial que requiere el desarrollo de esta investigación. Sin embargo, algunas ideas claves para el desarrollo del trabajo, tales como tratamiento de la información de entrada, metodologías empleadas para el entrenamiento de la red

neuronal, son propuestas en artículos IEEE (Institute of Electrical and Electronics Engineers). Un ejemplo lo constituye el tratamiento previo de la información de entrada a la red a través de un “pre-procesador”, como se plantea en [Sharaf-1993]; este pre-procesador puede ser otra red neuronal como es el caso de esquemas de predicción con redes neuronales que emplean mapas auto-organizativos de Kohonen (como clasificadores de patrones de consumo) [Kohonen-1990], [Hsu-1991] y [Baumann-1993], u otros esquemas de pre-procesamiento tales como sistemas expertos o lógica difusa como en [Ho-1990], [Sharaf-1993], [Srinivasan-1996] y [Shan-1997].

En cuanto a la forma de abordar el problema, diversas metodologías son encontradas en la bibliografía del tema de predicción de demanda con redes neuronales. La heurística presentada en el problema, debido a que la mayoría de implementaciones con redes neuronales surgen de procesos empíricos apoyados lógicamente en una teoría básica, hace que sean diversas las formas de enfrentar el problema de predicción; ejemplos de esto lo constituyen artículos como [Lee-1991], [AlFuhaid-1997], [Choueiki-1997] o [Rewagad-1998]. Algunos aspectos claves, que pueden servir como base para el desarrollo de una metodología para la predicción de demanda en el corto plazo, en lo concerniente al manejo de variables climatológicas, se presentan en [Xiao-1995], [Wang-1998] y [Kandil-1999]; otras alternativas interesantes, entre otras, como la de [Park-1993] consistente en la predicción de demanda a partir de la variación del consumo con respecto a un perfil de demanda patrón, o el empleo de redes neuronales recurrentes como en [Srivastava-1997] o el entrenamiento dinámico de la red (durante su etapa de producción) como en [Morioka-1993], amplían el espectro de posibilidades investigativas en el tema.

A nivel de trabajos desarrollados en la UIS con respecto al tema de predicción de demanda, se inició con la elaboración de un proyecto de grado denominado “Técnicas de Modelado y Predicción de demanda” [León-1994], en el cual se hizo un estudio del estado del arte las principales técnicas de predicción de demanda

convencionales tanto en el corto como en el largo plazo. Posteriormente se inició el trabajo de predicción a través de esquemas que emplean redes neuronales; un gran aporte en este aspecto, por ser el primer trabajo desarrollado en el tema, lo constituye el proyecto de grado “Predicción de Demandas de Carga Mediante Redes Neuronales” [Becerrra-1998], en el cual se propone una metodología para la selección de la arquitectura “óptima” de red neuronal a través de la monitorización de los errores de entrenamiento y validación, mediante un barrido sistemático de un conjunto de arquitecturas, en las cuales se varía el número de unidades en la entrada y el número de neuronas ocultas. Como resultado de la metodología propuesta se realizaron predicciones de la energía diaria demandada con errores porcentuales absolutos del 2,02% en promedio. Más recientemente se ha terminado el proyecto de grado “Predicción de Demanda Usando Redes Neuronales Entrenadas con el Algoritmo de Correlación en Cascada” [Portela-2000], el cual tiene como aporte fundamental el realizar predicciones de demanda de potencia hora a hora, a través de redes neuronales construidas mediante el algoritmo de correlación en cascada. Este algoritmo de aprendizaje constructivo supervisado, tiene la capacidad de modificar la topología de la red mientras la entrena. Como complemento en el entrenamiento se empleó el algoritmo Quickpro, el cual es un método de segundo orden basado en el método de Newton, con un alto componente heurístico; adicionalmente para el desarrollo del proyecto se empleó el programa de simulación de redes neuronales STUTTGART en sistema operativo LINUX. Otro aporte sobresaliente a emplear en el desarrollo del trabajo de investigación lo constituye el proyecto de grado: “Predicción de Caudales en el Mediano Plazo Mediante Redes Neuronales“ [Alvarado-1998] el cual, a pesar de no estar relacionado en sí con la predicción de demanda de energía eléctrica, se establece como una base importante para la implementación de esquemas de predicción mediante redes neuronales artificiales, debido a su aplicación en variables que tienen un alto componente aleatorio, lo cual es de gran importancia dado el planteamiento de descomposición de series de tiempo presentado en este trabajo de investigación el cual se describirá en el marco

teórico de este documento.

Adicionalmente, se han concluido dos proyectos de grado encaminados principalmente a la predicción de aspectos identificativos en cuanto a la forma de la curva de demanda diaria; estos son, el pico y valle de la curva de demanda diaria [Acevedo-2000], y la variación en el perfil de demanda [Rodriguez-2000]. Los resultados de estos proyectos de grado, además del aporte heurístico al tema, serán de importancia en el planteamiento del esquema de predicción empleado en este trabajo de investigación.

2. ANÁLISIS Y PREDICCIÓN DE SERIES TEMPORALES

A una variable cuyo registro se hace en forma periódica y regular se le conoce como SERIE DE TIEMPO y su ANÁLISIS (descomposición) es un factor clave en el proceso de predicción.

Considerando la metodología para la predicción de la demanda horaria de potencia empleada en este trabajo de investigación, es importante abordar el tema del Análisis de series de tiempo, entendido éste, como aquel proceso que permite separar la serie en sus componentes principales (Tendencia, Ciclo, Estacionalidad e Irregularidad), a fin de realizar predicciones más sencillas mediante la proyección por separado de las componentes, para luego reunirlas mediante la SÍNTESIS, y obtener el comportamiento futuro de la serie.

El análisis de series de tiempo permite identificar el comportamiento de la serie de acuerdo a patrones de datos y establecer el modelo de predicción más adecuado al requerimiento del pronóstico; para esto se recurre a la descomposición de la serie en las "fuerzas" que rigen su comportamiento. En [Hanke-1996] se presentan metodologías apropiadas para el análisis de series de tiempo, a fin de detectar en los datos históricos las componentes de la serie de tiempo definidas a continuación:

- **Tendencia:** Es la componente de largo plazo que constituye la base del crecimiento (o declinación) de la serie histórica. A nivel general, las fuerzas básicas que producen o afectan la tendencia de la serie pueden ser: cambios en la población, inflación, cambio tecnológico e incremento en la productividad.

- **Ciclo:** Es la componente de corto plazo, representada por un patrón que se repite a si mismo en forma periódica, durante toda la serie histórica. Su forma se debe al comportamiento en patrones de consumo de corto plazo (horas,

días, semanas e incluso meses) y su fluctuación está alrededor de la tendencia.

- **Estacionalidad:** Es un patrón de cambio regularmente recurrente a través del tiempo, el cual se presenta por lo general de manera anual. (p.e. estaciones climáticas).

- **Irregularidad:** Es la variabilidad de la serie de tiempo una vez se retiran las anteriores. Está compuesto por sucesos "impredecibles" o no periódicos.

Es importante destacar que dependiendo de el modelo de serie de tiempo a emplear se tendría diferente forma de establecer las componentes de la serie. Algunos autores tratan la serie de tiempo como la suma de las componentes que la caracterizan, es decir:

$$Y(t) = T + C + E + I. \quad \text{Ec. 1}$$

Sin embargo, la mayoría prefieren trabajar la serie en forma de producto, dada su facilidad de manejo:

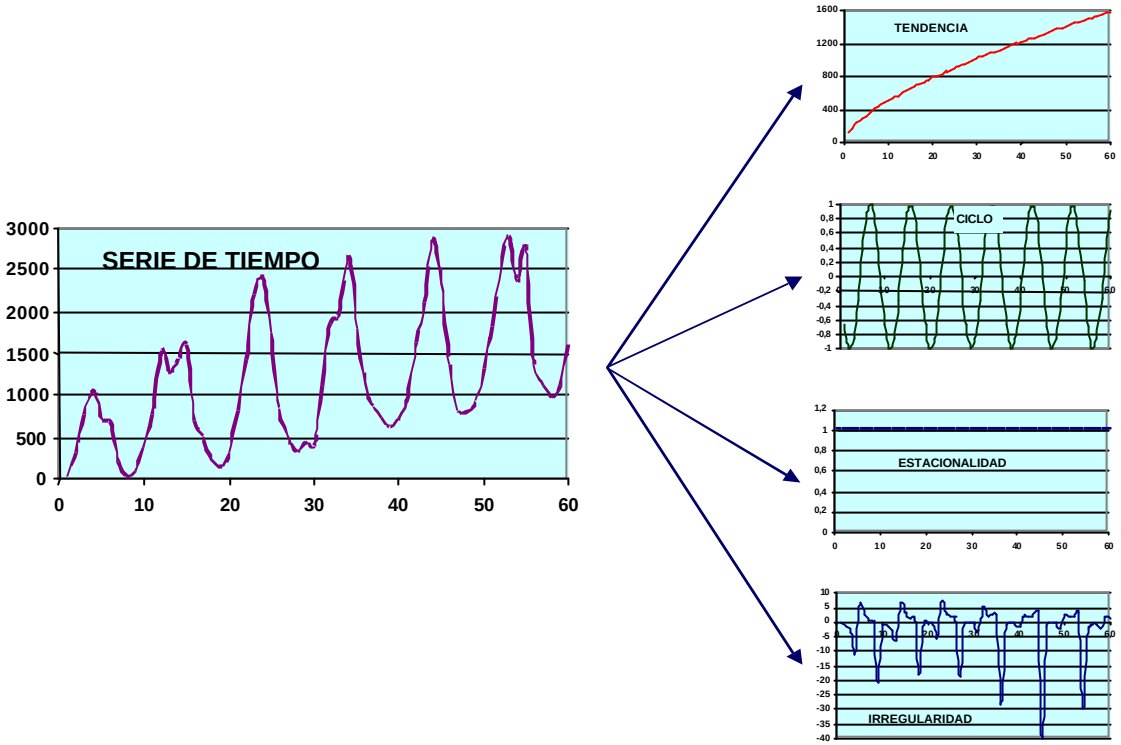
$$Y(t) = T \times C \times E \times I \quad \text{Ec. 2}$$

Inclusive, se encuentran planteamientos híbridos de la serie de tiempo a partir de las anteriores como se muestra a continuación, en la cual, la irregularidad se toma como un residual aditivo:

$$Y(t) = (T \times C \times E) + I \quad \text{Ec. 3}$$

A manera de ejemplo, en la figura 1 se muestra una serie de tiempo a la cual se le han separado sus cuatro componentes, tendencia (***T***), ciclo (***C***), estacionalidad (***E***) e irregularidad (***I***).

Figura 1. Descomposición de la serie de tiempo



2.1 EXPLORACIÓN DE PATRONES DE DATOS

El planteamiento metodológico de este trabajo de investigación, se basa en la descomposición de la serie de tiempo en las "fuerzas" que rigen su comportamiento, siendo por lo tanto indispensable la exploración de los patrones de datos a fin de establecer la existencia de cada una de las componentes mencionadas anteriormente (Tendencia, ciclo, estacionalidad e irregularidad), ya que esto permite identificar el tipo de modelo de predicción a emplear según la componente predominante dentro de la serie y la manera como se realizará la separación de las componentes que estén presentes.

Como se plantea en [Hanke-1996], una de las mejores maneras de explorar patrones de datos, es la Autocorrelación. Cuando se mide una variable a través del tiempo, con frecuencia está correlacionada consigo misma cuando se desfasa uno o más periodos. El coeficiente de autocorrelación es un indicador de la correlación que existe entre una variable desfasada uno a más periodos y la misma variable. En la ecuación 4 se muestra la manera de calcular el coeficiente de autocorrelación (r_k) de orden k (entre observaciones separadas k periodos: Y_t y Y_{t-k}).

$$r_k = \frac{\sum_{t=1}^{n-k} (Y_t - \bar{Y})(Y_{t-k} - \bar{Y})}{\sum_{t=1}^{n} (Y_t - \bar{Y})^2} \quad \text{Ec. 4}$$

Donde:

Y_t : Observación en el periodo de tiempo t.

$\bar{Y}$: Media de los valores de la serie.

Y_{t-k} : Observaciones en los k periodos anteriores o en el periodo $t-k$.

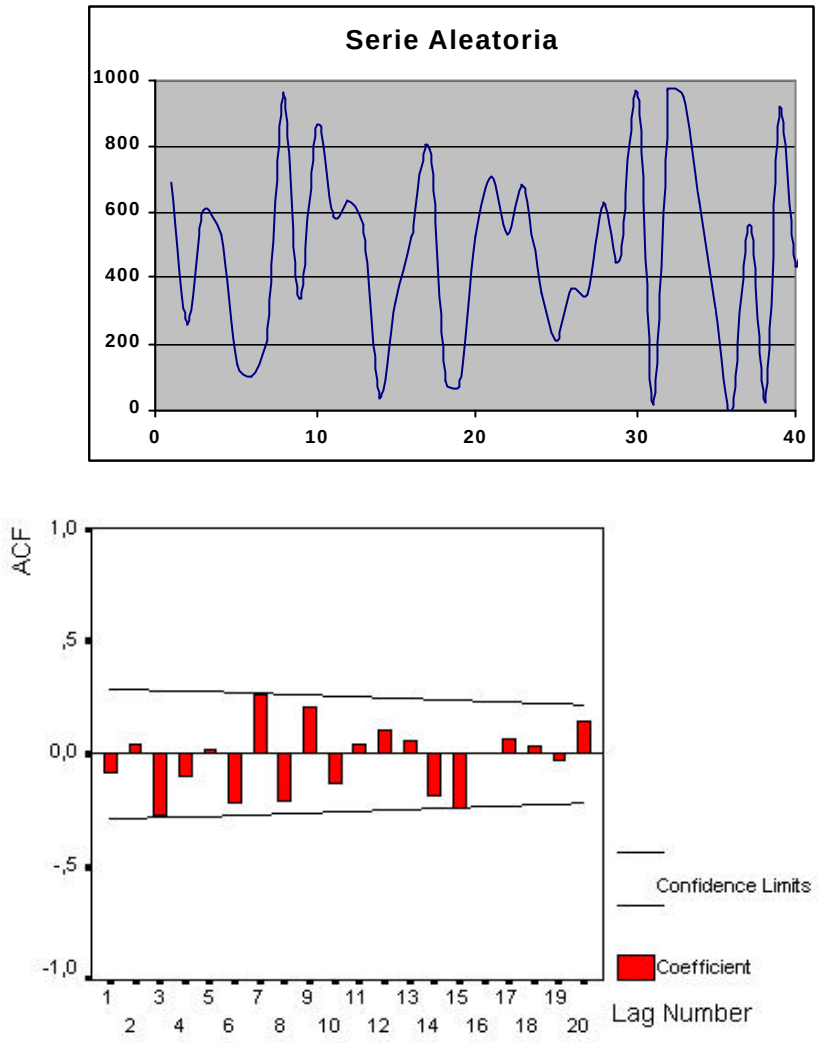
Los patrones de datos que incluyen todas o algunas de las componentes mencionadas anteriormente, se pueden estudiar empleando el enfoque del análisis de autocorrelación. Este permite resolver preguntas como:

- ¿Los datos son aleatorios?
- ¿Los datos tienen tendencia?
- ¿Los datos son estacionarios?
- ¿Los datos son estacionales?

Si una serie es aleatoria, la correlación entre Y_t y Y_{t-k} es cercana a cero y sus

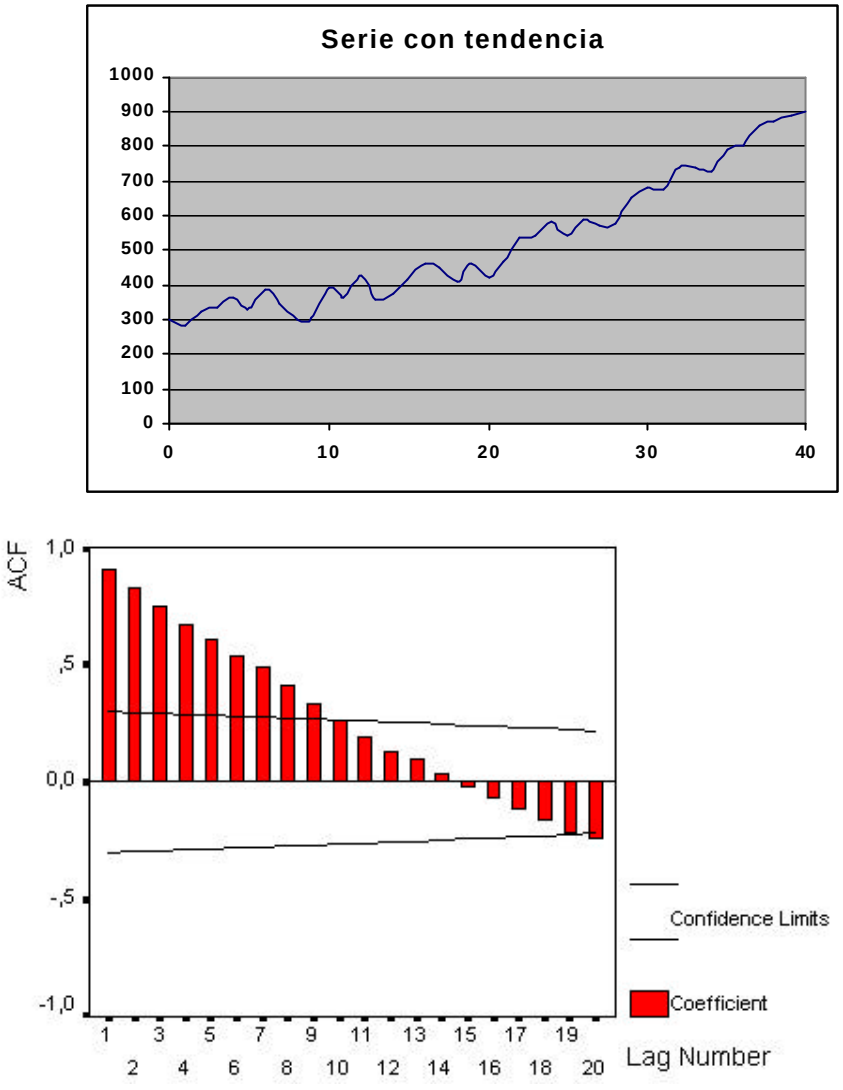
valores sucesivos de desfase no guardan relación entre sí. En la figura 2 se muestra a manera de ejemplo, una serie de tiempo generada aleatoriamente y su correspondiente correlograma, obtenido mediante el paquete estadístico SPSS.

Figura 2. Serie aleatoria ejemplo y su correlograma



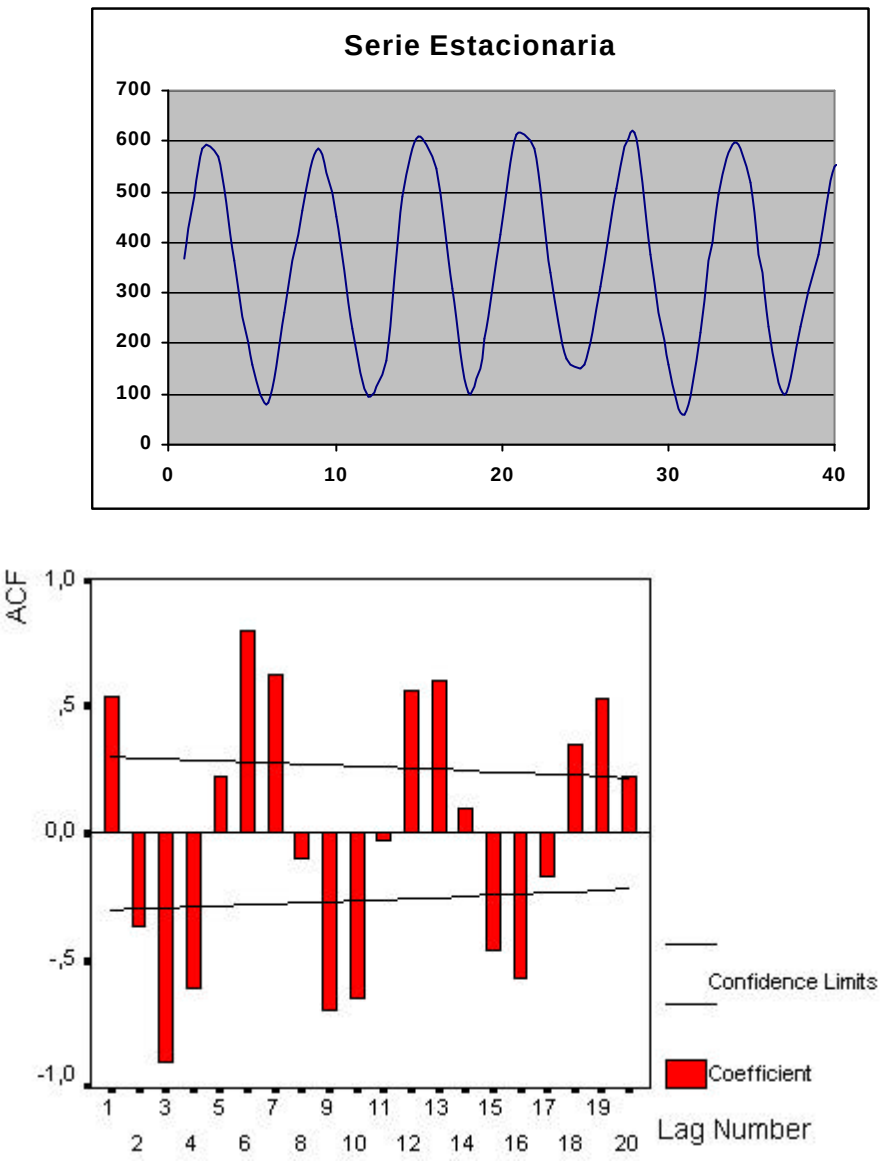
Cuando la serie tiene una componente predominante de largo plazo, es decir presenta como mayor componente su tendencia, la correlación entre Y_t y Y_{t-k} es alta (cercano a 1) en los primeros períodos de desfase y desciende en forma gradual. En la figura 3 se muestra una serie de tiempo con tendencia generada a manera de ejemplo y su correspondiente correlograma.

Figura 3. Serie con tendencia ejemplo y su correlograma



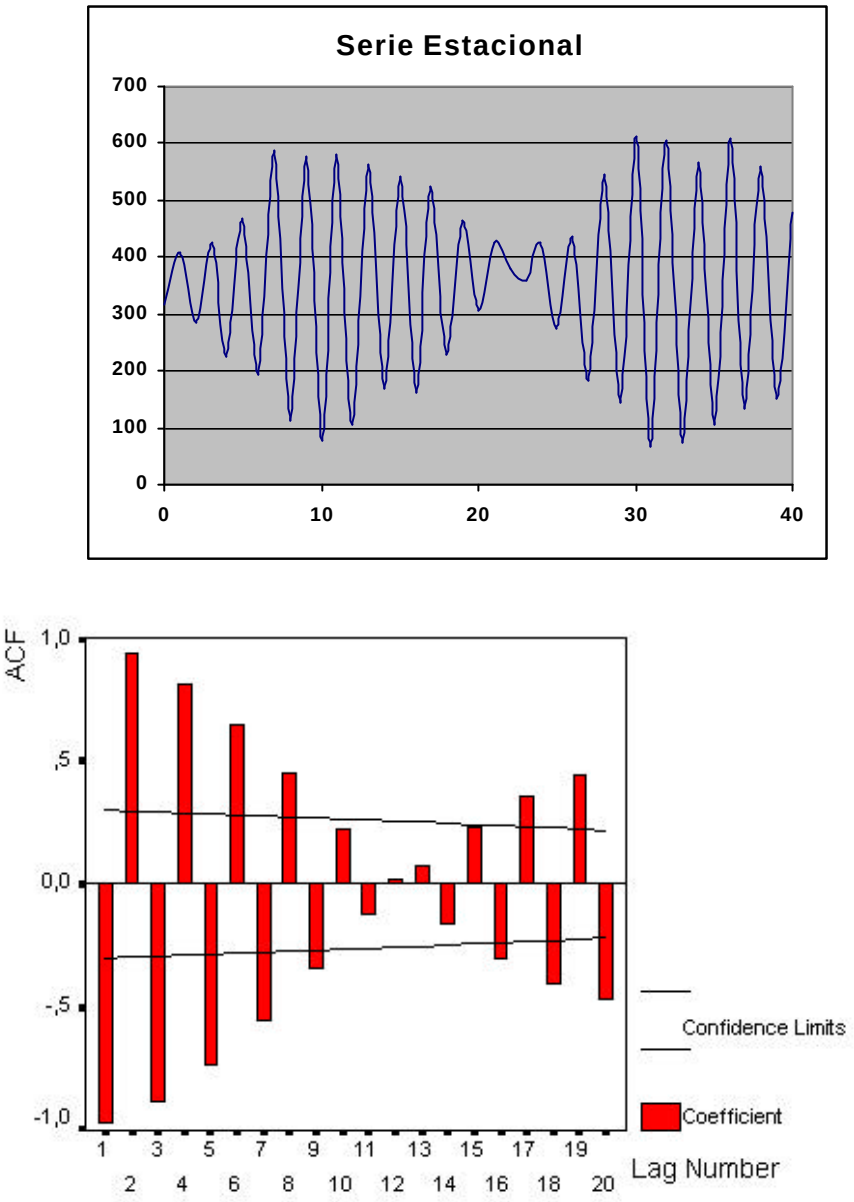
Si la serie es de corto plazo con característica estacionaria (sus propiedades estadísticas básicas como media y varianza permanecen constante con el tiempo), los patrones de autocorrelación se repiten en iguales periodos de desfase. La figura 4 muestra una serie estacionaria generada como ejemplo y su correspondiente correlograma.

Figura 4. Serie estacionaria ejemplo y su correlograma



Finalmente, cuando la serie tiene un patrón estacional, se presentará un coeficiente de autocorrelación significativo en el período de desfasamiento correspondiente a la repetición del efecto estacional. La figura 5 muestra una serie estacional generada como ejemplo y su correspondiente correlograma.

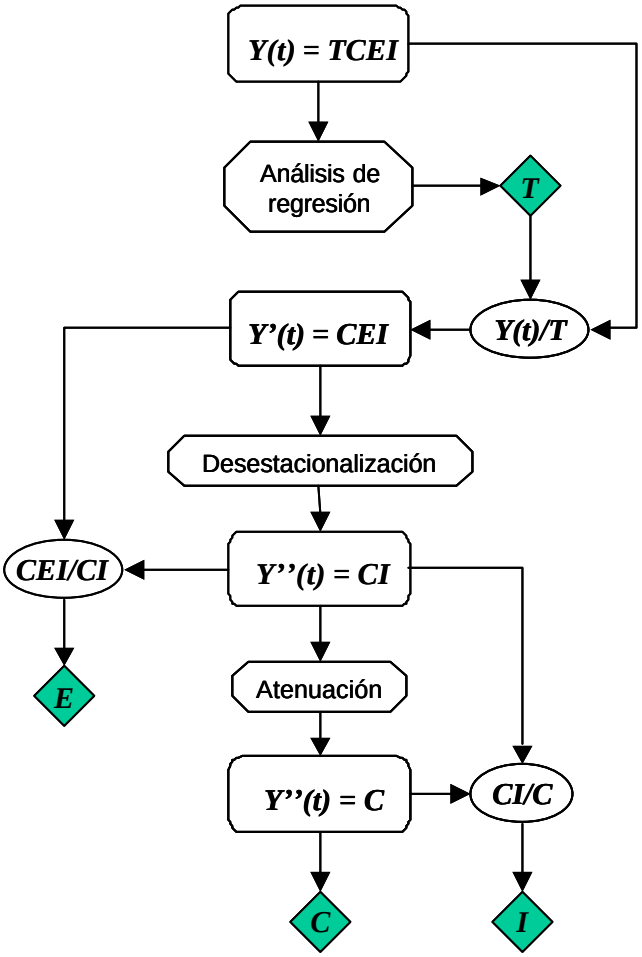
Figura 5. Serie estacional ejemplo y correlograma



Es de aclarar que las series ejemplo mostradas anteriormente, fueron generadas con cierto patrón aleatorio, pero con una fuerte componente de la característica que representan. En la práctica es muy probable, encontrar efectos combinados, lo cual hace que el correlograma requiera experiencia para su análisis.

Bajo el enfoque de una serie de tiempo multiplicativa ($Y=TCEI$) el procedimiento de análisis (separación de las componentes) de la serie de tiempo consistirá en la realización de operaciones sucesivas de división término a término de la serie de tiempo.

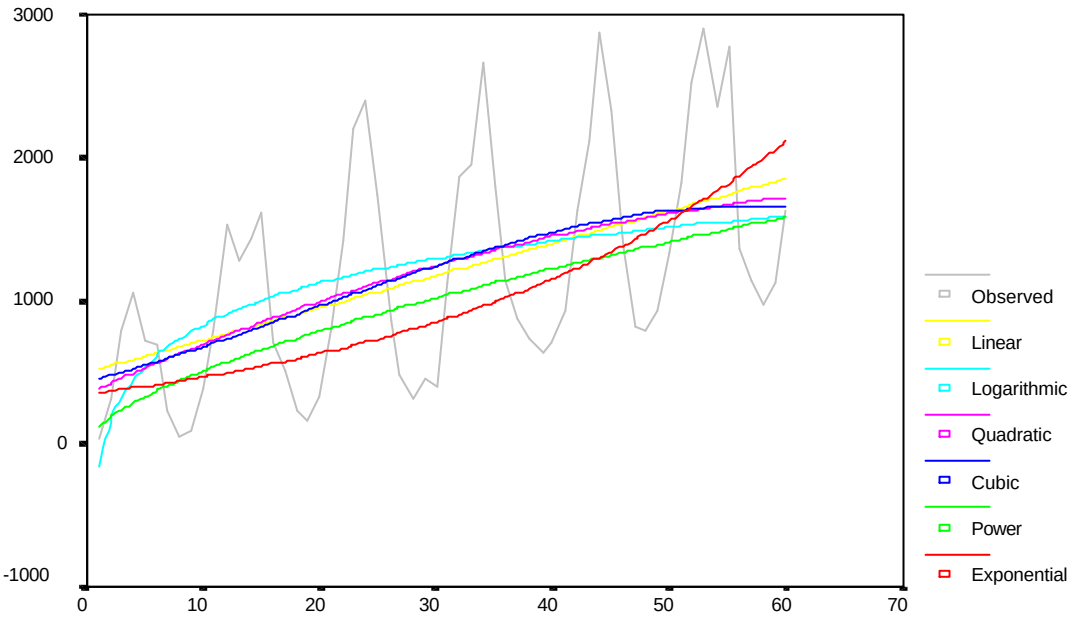
Figura 6. Procedimiento de descomposición de las series de tiempo



Como se muestra en la figura 6, el inicio del proceso de descomposición es el establecimiento de la tendencia de la serie de tiempo. Para esto, se recurre a un análisis de regresión, el cual permite encontrar una línea que minimice la suma de los cuadrados de las desviaciones alrededor de ella de todos los datos de la serie.

En el análisis de regresión, la línea de "mejor ajuste" no necesariamente es una línea recta; aparte del análisis de regresión lineal, existen análisis de regresión de potencia, logarítmicos, exponenciales, entre otros. En la figura 7 se muestran las diferentes líneas de regresión para la serie de tiempo mostrada como ejemplo en la figura 1 (en color gris), obtenidas mediante el paquete estadístico SPSS.

Figura 7. Diferentes lineas de regresión para la serie ejemplo



Para seleccionar la mejor línea de regresión, se observan los coeficientes de correlación. Un coeficiente de correlación cercano a 1, quiere decir que la línea representa el mejor ajuste al conjunto de datos de la serie.

Una vez se obtiene una ecuación dependiente del tiempo de la tendencia de la

serie, se procede a estimar las otras componentes por simple división, lo que también se puede denominar como remoción de la tendencia de la serie: $CEI = Y(t) / T$. Esta serie resultante CEI, puede ser desestacionalizada, es decir, se procede a encontrar patrones de cambio que se repitan a sí mismos en la serie cada intervalo de tiempo y se eliminan para convertirla en una serie CI. Para esto, se procede a realizar promedios móviles con intervalos de tiempo que van de acuerdo a correlogramas que indican dónde pueden presentarse las estacionalidades y corregir la serie con factores de estacionalidad.

Si se atenúa la serie CI a través de promedios simples o móviles o por medio de atenuación exponencial, se puede llegar al ciclo de la serie. Para tener por separado cada una de las componentes se realizan divisiones simples entre los términos de las series de tiempo resultantes, siguiendo el proceso que muestra la figura 7.

Es importante aclarar, que por ser un modelo de producto, solo una de las componentes de la serie de tiempo tendrá las unidades de la serie completa (para el caso de estudio unidades de potencia como kW , MW o GW). Según el planteamiento presentado anteriormente, la Tendencia sería la componente con las unidades de potencia, ya que es la primera que se separa de la serie y mostraría el crecimiento promedio de la demanda con el tiempo (comportamiento de largo plazo).

No obstante, es de mayor interés para un modelo de predicción en el corto plazo, que la componente que lleve las unidades del modelo y las magnitudes del comportamiento, sea la componente de corto plazo, es decir la que se separe primero de la serie, sea el ciclo. En este orden de ideas, es necesario plantear un nuevo esquema de descomposición, en el cual se separe primero el ciclo de la serie de tiempo.

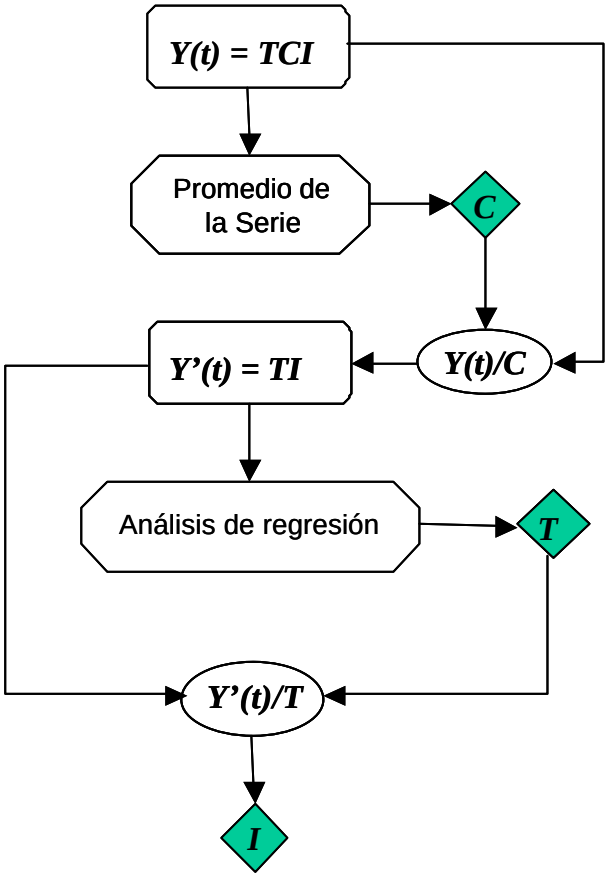
También se descartó el patrón estacional en estas series mediante un análisis previo de los datos y resultados de trabajos de grado anteriores en torno al tema

de predicción de la demanda de corto plazo, lo cual es razonable al considerar que en Colombia no hay estaciones que afecten el comportamiento de la demanda de manera apreciable. Esto quiere decir que las series de estudio tienen la forma:

$$Y(t) = TCI \quad \text{Ec. 5}$$

En la figura 8 se presenta el esquema de descomposición de la serie de tiempo empleado en el planteamiento de este trabajo de investigación, en el cual se considera que el ciclo es la componente que da la característica principal al comportamiento de la serie.

Figura 8. Esquema de descomposición a emplear en el prototipo



Al comparar con la figura 6, el esquema planteado en la figura 8 separa en primer lugar el ciclo de la serie (***TCI***) mediante el promedio de los datos, en cada una de las horas. Esto dará como resultado una serie de tiempo solo de 24 horas, siendo el dato de cada una de esas horas el promedio de todos los datos correspondientes a la hora donde se promedió.

Posteriormente se divide cada dato de la serie original (***TCI***), en el ciclo de 24 horas repetido n veces, según sea el tamaño de los datos de la serie original, dando como resultado una serie de tendencia e irregularidad (***TI***).

Encontrando la tendencia mediante un análisis de regresión de la serie resultante del proceso anterior (***TI***), se realiza de nuevo la división, esta vez entre la serie de tendencia, dando como resultado los valores de la serie de irregularidad.

Vale la pena destacar que si se conocen por separado, cada una de las componentes anteriores, se pueden realizar pronósticos de una manera sencilla, ya que ***T*** y ***C*** son perfectamente extrapolables por ser funciones (en el caso de ***T***) o patrones que se repiten dependientes del tiempo (en el caso de ***C***). No obstante, la irregularidad es un patrón no extrapolable debido a su predominante aleatoriedad, siendo necesario emplear modelos matemáticos más avanzados para su proyección en el tiempo. Aquí es donde se pueden aprovechar potencialidades de técnicas no convencionales de predicción como las Redes Neuronales.

2.2 TÉCNICAS DE PREDICCIÓN

A nivel general, existen dos grupos de técnicas de predicción: Las técnicas cualitativas, que se basan en el juicio humano y en la intuición, y las técnicas cuantitativas que se utilizan cuando existen suficientes datos históricos disponibles, los cuales son representativos del futuro.

Las técnicas cuantitativas pueden ser agrupadas en tres subgrupos: estadísticas, determinísticas y no-convencionales. Las técnicas estadísticas se enfocan completamente en patrones, cambios en ellos y perturbaciones causadas por influencias "aleatorias"[4]. Las técnicas estadísticas se basan principalmente en dos enfoques: El primero consiste en la suposición de que los datos pueden ser descompuestos en sus cuatro componentes básicas (tendencia, ciclo, estacionalidad e irregularidad), realizándose la predicción mediante la proyección en el tiempo de cada una de las componentes por separado y reuniéndolas luego (ya sea por suma o producto). El segundo enfoque se basa principalmente en conceptos de identificación de sistemas, y a diferencia del anterior, no supone a la variable dependiente (valor a predecir), como el efecto de componentes separadas, sino un determinado modelo matemático producto de un análisis sobre la misma serie de tiempo en el pasado.

Las técnicas determinísticas se basan en la identificación y determinación de las influencias de otras variables independientes (y de la misma variable a predecir en su comportamiento histórico) sobre la variable a predecir (variable dependiente).

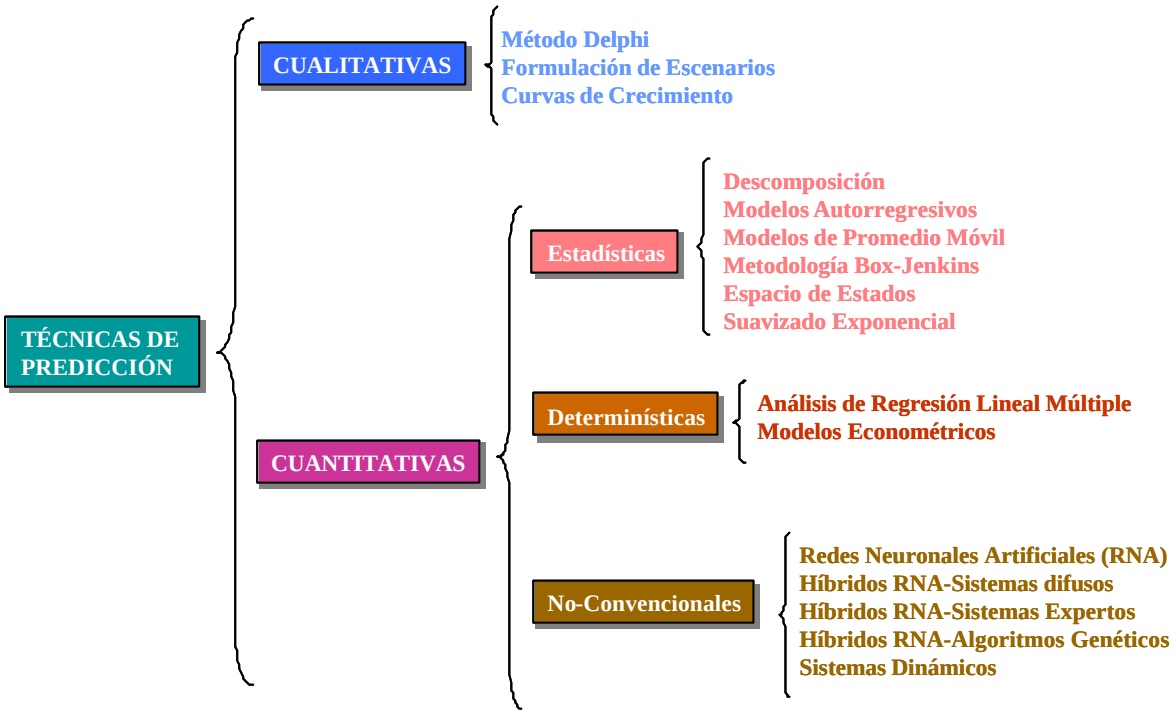
En las técnicas determinísticas, conceptos estadísticos como la regresión entre variables y otros conceptos como estimación de promedios móviles, autorregresión, son empleados en un proceso tendiente a estimar un modelo matemático de la variable a predecir como función de un conjunto de variables explicativas, las cuales pueden ser dependientes (Caso de modelos autorregresivos) o independientes (Caso de regresión múltiple).

Como técnicas no-convencionales se conocen aquellas técnicas que no se pueden clasificar dentro de las anteriores o emplean esquemas de predicción híbridos entre las diferentes técnicas. La mayoría de estos esquemas emplean

[4] En el caso de las técnicas estadísticas, lo "aleatorio" se refiere a lo que sobra entre el valor verdadero y el patrón modelo. En el caso de un enfoque determinístico, gran parte de lo que sobra, considerado dentro del modelo estadístico, puede ser explicado o tiene alguna causa, dejando de ser entonces "aleatorio".

procesos de inteligencia artificial o sistemas dinámicos. A continuación, en la figura 9, se presenta un resumen de las técnicas mas comúnmente usadas.

Figura 9. Técnicas de predicción de uso común



Debido a las características del trabajo a realizar, es decir la predicción de la demanda en el corto plazo, las técnicas cualitativas de predicción pierden interés debido a su aplicabilidad en escenarios del largo plazo, de carácter económico principalmente.

En cuanto a las técnicas cuantitativas, se realizaron dentro del trabajo de investigación, pruebas con poca cantidad de datos tendientes a establecer, en primer lugar, las ventajas y desventajas de la aplicación de estas técnicas para la predicción de la demanda de energía, y en segundo, las posibilidades que pueden

ofrecer estas técnicas dentro de modelos híbridos de predicción, con redes neuronales artificiales.

A continuación se presentará un resumen de las técnicas de predicción convencionales, dejando a las no convencionales, representadas por las redes neuronales, para tratarlas mas adelante de manera específica. Trabajos relacionados con la identificación de sistemas como el [Ljung-1994], y en forma más aplicada al proceso de predicción de demanda de corto plazo el de [Gross-1987], así como textos de consulta como [Hanke-1996] y [Granger-1989] en el tema de los pronósticos de series de tiempo, han sido tomados para generar el mencionado resumen.

Las técnicas convencionales seleccionadas para realizar este análisis son las que en la bibliografía recopilada, presentan mejor rendimiento en la predicción o son aptas para la predicción en el corto plazo. Estas son: Metodología Box-Jenkins, Suavizado Exponencial, Espacio de Estados, Análisis de Regresión Lineal Múltiple. A pesar de que la técnica de descomposición, tratada anteriormente es una técnica de predicción válida en el corto plazo, se hará uso de ella como ayuda dentro del proceso de predicción, mas no como predictor propiamente dicho.

2.2.1 Metodología de Box-Jenkins.

Esta metodología es en realidad la composición de dos técnicas de predicción: El modelo Auto-Regresivo (AR - AutoRegressive) y el Promedio Móvil (MA – Moving Average). Para entender mejor esta metodología, es necesario tratar por separado cada uno de los modelos anteriores.

2.2.1.1 Modelo autorregresivo.

Esta técnica emplea la regresión de datos como mecanismo para encontrar el valor a predecir de una serie de tiempo. Para

esto se retrasa la variable dependiente $Y(t)$[5] uno a más valores de tiempo y se utiliza como variable independiente. En otras palabras, el modelo autorregresivo expresa una predicción como una función de valores previos de la misma serie de tiempo. Un modelo autorregresivo de orden p, donde p es el número de valores en el tiempo que se retrasa la serie para calcular los coeficientes de regresión[6], posee la siguiente forma:

$$\hat{Y}_t = f_0 + f_1 Y_{t-1} + f_2 Y_{t-2} + \ldots + f_p Y_{t-p} + e_t \quad \text{Ec. 6}$$

Donde:

$f_0, f_1, \ldots, f_p$: Coeficientes de regresión.

$Y_{t-1}, \ldots, Y_{t-p}$: Variables independientes (Variable dependiente desfasada un número específico de valores en el tiempo)

e_t : Residuo que representa sucesos aleatorios no explicados por el modelo

2.2.1.2 Modelo de promedio móvil. Al igual que un promedio simple de los datos para cada hora del día, durante todos los días del año, o el suavizado exponencial que se verá mas adelante, el promedio móvil puede incluirse dentro de las técnicas de atenuación de las series de tiempo. Esta atenuación, puede decirse que elimina la aleatoriedad o el ruido de la serie de tiempo (en otras palabras la Irregularidad). Podría definirse al promedio móvil para un valor de tiempo t, como la media aritmética de las n observaciones más recientes. Es decir, en el promedio móvil, la predicción se obtiene encontrando la media de un conjunto específico de valores, la cual es empleada para predecir el siguiente

---

[5] En adelante se adoptará como $Y(t)$ el valor real u observado de la serie de tiempo y $\hat{Y}(t)$ el valor predicho de la serie de tiempo.

[6] El orden p del polinomio formado se obtiene del análisis de la función de autocorrelación parcial, e indica que cantidad de datos previos son representativos para obtener el valor de la variable en el tiempo t.

intervalo de tiempo.

$$\hat{Y}_{t+1} = \frac{(Y_t + Y_{t-1} + Y_{t-2} + ... + Y_{t-n+1})}{n} \quad \text{Ec. 7}$$

No obstante, el interés principal del promedio móvil dentro de la metodología Box-Jenkins, es su efecto de suavizado, el cual puede ser expresado en términos de los errores obtenidos entre la serie real Y_t y la serie predicha $\hat{Y}_t$, a medida que se avanza a través de todos los datos de la serie. Un modelo de predicción de promedio móvil de orden q, siendo q el número de datos en los cuales se agrupa la serie para sacar los promedios móviles, tiene la forma:

$$\hat{Y}_t = w_0 + e_t - w_1 e_{t-1} - w_2 e_{t-2} - ... - w_q e_{t-q} \quad \text{Ec. 8}$$

Donde:

$w_0, w_1, ..., w_q$: Pesos específicos

e_t : Residuo o error

$e_{t-1}, ..., e_{t-q}$: Valores previos de los residuos

La ecuación anterior es similar a la a ecuación del modelo autorregresivo (Ec.5), excepto que la variable dependiente Y depende de los valores previos de los residuos, en vez de la misma variable. Los modelos de promedio móvil (MA) proporcionan pronósticos de Y_t, con base en una combinación lineal de errores anteriores, mientras que los modelos autorregresivos (AR) expresan Y como una función lineal de cierto número de valores anteriores reales de Y_t. Es una costumbre presentar los pesos específicos con coeficientes negativos, aun cuando los pesos pueden ser positivos o negativos. La suma de $w_1 + w_2 + ... + w_q$ no necesariamente son iguales a 1, y los valores de w_i no se "mueven ' con las nuevas observaciones, como sucede con los cálculos de promedio móvil que se mostraron inicialmente.

2.2.1.3 Modelo autorregresivo de promedio móvil. Al juntar un modelo autorregresivo de orden p con uno de promedio móvil de orden q, se puede crear un tercer modelo de característica más completas (ARMA (p,q)). Sin embargo, este modelo calcula por separado cada coeficiente y el orden del polinomio resultante. Su forma es simplemente la suma de las dos ecuaciones anteriormente vistas.

$$\hat{Y}_t = f_0 + f_1 Y_{t-1} + f_2 Y_{t-2} + \ldots + f_p Y_{t-p} + e_t - w_1 e_{t-1} - w_2 e_{t-2} - \ldots - w_q e_{t-q} \quad \text{Ec. 9}$$

Una mejor implementación de este tipo de modelos se consigue a través de la metodología Box-Jenkins, la cual da como resultado un modelo autorregresivo de promedio móvil integrado (ARIMA – AutoRegressive Integrated Moving Average) más acorde con el comportamiento de la serie en estudio.

Un modelo ARIMA(p,q) utiliza combinaciones de errores anteriores y valores anteriores, ofreciendo un potencial para ajustar modelos que no pudieron ser ajustados en forma adecuada mediante los modelos AR y MA por sí solos.

Una diferencia importante entre la metodología Box-Jenkins y los métodos AR, MA y ARMA consiste en que Box-Jenkins no hace suposiciones acerca del número de términos o de los pesos específicos a asignar a los términos. El analista selecciona el modelo apropiado, incluyendo el número de términos; después el programa calcula los coeficientes, utilizando un método no lineal de mínimos cuadrados. Se puede, además, hacer pronósticos de periodos futuros y construir intervalos de confianza para estas estimaciones. El enfoque de Box-Jenkins comprende tres etapas separadas, las cuales marcan el procedimiento a seguir en el planteamiento de cualquier algoritmo para el empleo de esta metodología; estas son: Identificación del modelo, estimación y prueba del modelo, y aplicación del modelo.

En la etapa de identificación del modelo se determina si la serie es estacionaria,

es decir, si el valor de la media varía a través del tiempo (series con tendencia). Si la serie no es estacionaria, en general se puede convertir a una serie estacionaria mediante el método de diferenciación (eliminar la tendencia de la serie). El grado de diferenciación es especificado por el analista y el algoritmo de la metodología Box-Jenkins convierte los datos en una serie estacionaria y realiza los cálculos subsecuentes utilizando los datos convertidos. Una vez obtenida una serie estacionaria, el analista debe identificar la forma del modelo a utilizar; este paso se logra mediante la comparación de los coeficientes de autocorrelación y de autocorrelación parcial de los datos a ajustar con las correspondientes distribuciones de los diversos modelos ARIMA. Para ayudar en la selección de un modelo apropiado existen distribuciones teóricas para los modelos ARIMA más comunes, ya que cada modelo tiene un conjunto único de autocorrelaciones y de autocorrelaciones parciales.

En la segunda etapa (estimación del modelo y prueba de su adecuación), una vez se ha seleccionado un modelo tentativo, se deben estimar los parámetros para ese modelo. A diferencia de un modelo ARMA, el cálculo de parámetros como coeficientes de regresión (f) y pesos específicos (w) no se realiza por separado, sino seleccionando los valores óptimos de estos parámetros, a través del error medio cuadrado. Antes de emplear el modelo en la predicción, el analista deberá probar si el modelo seleccionado es adecuado; para esto se revisan los errores $e_t = Y_t - \hat{Y}_t$ para asegurarse de que son aleatorios y no poseen algún patrón determinado. Esta verificación puede hacerse revisando las autocorrelaciones de los términos de error, asegurándose de que no sean diferentes de cero en forma significativa. Si algunos retrasos de orden menor o estacionales, son diferentes de cero de manera significativa, entonces el modelo resulta inadecuado y se deberá regresar a la etapa 1 para seleccionar un modelo alternativo. Otra manera de revisar si el modelo planteado es adecuado, consiste en emplear la prueba ji cuadrada (c^2), también conocida como estadística Q de Box-Pierce, la cual trabaja sobre las autocorrelaciones de los residuos. La expresión de la prueba es la

siguiente:

$$Q = (N - d) \sum_{k=1}^{m} r_k^2 \quad \text{Ec. 10}$$

Donde:

N : Longitud de la serie histórica

k : Primeras k autocorrelaciones que se verifican

m : Número máximo de retrasos verificados

r_k : Función de autocorrelación de la muestra de k-ésimo término de residuo

d : Grado de diferenciación para obtener una serie estacionaria

Esta estadística está distribuida aproximadamente como una distribución de ji cuadrada con k-p-q grados de libertad. En la ecuación anterior, Si el valor calculado de Q es mayor que la c^2 para k-p-q grados de libertad, entonces se debe considerar que el modelo es inadecuado y regresar a la etapa inicial.

Una vez que se ha encontrado el modelo adecuado se puede ingresar en la tercera etapa, en la cual se realiza la predicción en sí. Estas predicciones pueden ser hechas para uno o varios periodos en el futuro. Con suficientes datos, se puede emplear el mismo modelo para otro valor en el tiempo como origen y así validar el modelo encontrado. En caso de que se tenga una serie cambiante en el tiempo, es necesario recalcular los parámetros del modelo seleccionado, para mantener los errores de predicción dentro de valores adecuados.

2.2.2 Suavizado exponencial. Es un método basado en el promediado de los valores anteriores de una serie, haciendo esto de forma decreciente (exponencial). Las observaciones se ponderan, asignando un mayor peso a las más recientes. Estas ponderaciones se designan como a para la observación mas reciente, $a(1-a)$ para la siguiente más reciente, $a^2(1-a)$ para la siguiente y así sucesivamente.

Podría decirse que una predicción mediante este método (para el valor de tiempo $t+1$) puede ser expresada como el promedio ponderado de la nueva observación (en el valor de tiempo t) y el promedio ponderado anterior de la predicción anterior (en el valor de tiempo t). En otras palabras:

$$Nueva\ predicción = a(Nueva\ observación) - (1-a)(Predicción\ anterior) + \ldots \quad \text{Ec. 11}$$

En el caso de una predicción que tenga en cuenta tres retardos en el tiempo, la expresión quedará.

$$\hat{Y}_{t+1} = aY_t + (1-a)\hat{Y}_t + a^2 Y_{t-1} + a(1-a)\hat{Y}_{t-1} + a^3 Y_{t-2} + a^2(1-a)\hat{Y}_{t-2} \quad \text{Ec. 12}$$

Reorganizando la anterior ecuación para interpretar mejor la ponderación a se tiene:

$$\begin{aligned} \hat{Y}_{t+1} &= \hat{Y}_t + a(Y_t - \hat{Y}_t) + a\hat{Y}_{t-1} + a^2(Y_{t-1} - \hat{Y}_{t-1}) + a^2\hat{Y}_{t-2} + a^3(Y_{t-2} - \hat{Y}_{t-2}) \\ &= \left[\hat{Y}_t + a\hat{Y}_{t-1} + a^2\hat{Y}_{t-2}\right] + \left[a(Y_t - \hat{Y}_t) + a^2(Y_{t-1} - \hat{Y}_{t-1}) + a^3(Y_{t-2} - \hat{Y}_{t-2})\right] \end{aligned} \quad \text{Ec. 13}$$

De la anterior se ve que el suavizado exponencial, es simplemente la predicción anterior $\hat{Y}_t$ mas a veces el error $e_t = Y_t - \hat{Y}_t$, el resto de términos es la influencia en el valor predicho de los valores anteriores, afectados de una manera exponencial decreciente por la ponderación a (la cual está entre 0 y 1). El valor de la ponderación determina qué tanto una observación reciente puede influir en el valor del pronóstico. Cuando a se acerca a 1, la predicción incluirá un ajuste sustancial

de cualquier error ocurrido en la predicción anterior. De manera contraria, si a es cercano a 0, la predicción es similar a la anterior.

La anterior explicación es la base del suavizado exponencial; sin embargo, mejoramientos a esta técnica como la atenuación exponencial doble, los modelos de Hold, de Winter y de Brown (empleado este último en la metodología propuesta), presentan variantes que perfeccionan el proceso anterior mejorando las expectativas de error.

2.2.3 Modelo de espacio de estados.

El concepto de espacio de estados aplicado al estudio de señales y sistemas, puede ser empleado en el caso de la predicción de demanda de energía, aprovechando el comportamiento discreto de la curva de demanda diaria.

Tradicionalmente los sistemas pueden ser caracterizados relacionando sus entradas y salidas por medio de ecuaciones polinómicas y diferenciales. Al considerar la técnica de espacio de estados, el sistema se representa a través de la forma convencional de entrada y salida, más un conjunto adicional de variables conocidas como variables de estado. Bajo este enfoque, las expresiones matemáticas que describen un sistema cualquiera estarán divididas en dos conjuntos: Un grupo de ecuaciones encargadas de relacionar las variables de estado con la señal de entrada, y otro grupo encargado de relacionar las variables de estado con la señal de salida.

Estas variables de estado proporcionan información sobre todas las señales internas seleccionadas del sistema, dando como resultado una caracterización mas detallada del mismo, que la descripción convencional entrada-salida. Para la determinación de la salida de un sistema, además de su entrada y salida, se requiere conocer un conjunto de condiciones iniciales en el momento en que se

aplica la entrada. Si un sistema no está en reposo inicial, es decir, el estado n_0, entonces el conocimiento de la señal de entrada $\boldsymbol{X(n)}$ para $\boldsymbol{n>n_0}$ no es suficiente para determinar unívocamente la salida $\boldsymbol{Y(n)}$ para $\boldsymbol{n>n_0}$. Se deben entonces conocer y tener en cuenta las condiciones iniciales del sistema en $\boldsymbol{n=n_0}$. Este conjunto de condiciones se denomina el estado inicial del sistema (en $\boldsymbol{n=n_0}$) y los valores que produce el avance de estas variables en el tiempo se denominan estados del sistema.

Un esquema de solución a través de espacio de estados para señales discretas, presenta la siguiente forma general:

1. $X_{k+1} = AX_k + BU_k$ Ec. 14

2. $Y_k = CX_k + DU_k$ Ec. 15

Donde:

$\boldsymbol{X_{k+1}}$: Vector de variables de estado actualizadas

$\boldsymbol{A}$: Matriz de coeficientes del sistema

$\boldsymbol{X_k}$: Vector de variables de estado

$\boldsymbol{B}$: Matriz de acoplamiento de la entrada

$\boldsymbol{Y_k}$: Vector de salidas del sistema

$\boldsymbol{U_k}$: Entrada del sistema

$\boldsymbol{C}$: Matriz que relaciona la salida con las variables de estado

La aplicación de la técnica de espacio de estados a la predicción de demanda, puede incluir otros métodos de predicción como los anteriormente mencionados,

ya que es necesaria la formulación de un modelo matemático para su implementación. En el problema de predicción de la demanda, la carga es modelada como una variable de estado y las dos ecuaciones anteriores toman la siguiente forma:

Ecuación de estados: $X_{k+1} = f_{k+1} X_k + W_k$ **Ec. 16**

Ecuación de observaciones: $Y_k = H_k X_k + V_k$ **Ec. 17**

Donde

f_k : Matriz ($n \times n$) de transición de estados que relaciona X_k con X_{k+1} cuando no existe función forzada

X_k : Vector ($n \times 1$) de estados del proceso en el instante k

W_k : Vector ($n \times 1$) de Ruido blanco[7] con covarianza conocida Q_k

Y_k : Vector ($m \times 1$) de carga observada en el instante t_k

H_k : Matriz ($m \times n$) que relaciona X_k con Y_k sin considerar el ruido

V_k : Vector ($m \times 1$) del error en los valores de la carga, el cual es representado como ruido blanco con covarianza conocida R_k

Las matrices de covarianza para los vectores W_k y V_k están dadas por:

$$E(W_k, W_i^T) = \begin{cases} Q_k & i = k \\ 0 & i \neq k \end{cases} \qquad E(V_k, V_i^T) = \begin{cases} R_k & i = k \\ 0 & i \neq k \end{cases} \qquad \text{Ec. 18}$$

[7] El término “Ruido blanco” se asocia a una señal de ruido que tiene una distribución uniforme de frecuencias, como ocurre en el caso de la luz blanca.

El ruido del proceso de actualización del estado W_k y el ruido de los valores observados es asumido como no correlacionado de acuerdo con:

$$E(W_k, V_i^T) = 0 \quad para\ todo\ k\ e\ i \qquad \text{Ec. 19}$$

Para cualquier instante t_k habrá una estimación para el proceso basado en el conocimiento del estado para t_{k-1}. A este valor se le llama estimación a priori y se designa como $X_{k|k-1}$. Entonces el error asociado a la diferencia entre la actual y la estimación previa del proceso, está dado por:

$$e_{k|k-1} = X_k - X_{k|k-1} \qquad \text{Ec. 20}$$

Este vector de error, tiene una matriz de covarianza expresada por:

$$E(e_{k|k-1}, e_{k|k-1}^T) = P_{k|k-1} \qquad \text{Ec. 21}$$

La estimación a posteriori es obtenida como una combinación lineal de la estimación a priori y el ruido de las observaciones, de la siguiente forma:

$$X_{k|k} = X_{k|k-1} + K_k \lfloor Y_k - H_k X_{k|k-1} \rfloor \qquad \text{Ec. 22}$$

Donde:

$X_{k|k}$: Estimación actualizada

K_k : Factor de combinación

El error asociado a la diferencia entre la actual y la estimación a posteriori del proceso es:

$$E(e_{k|k}, e_{k|k}^T) = P_{k|k} \qquad \text{Ec. 23}$$

El factor de combinación K_k óptimo es obtenido a través del criterio del mínimo error medio cuadrado (MSE-mean-square error). A este factor se le conoce como

ganancia Kalman y el procedimiento para la implementación del filtro Kalman en el proceso de predicción, permite aprovechar la propiedad recursiva de tal algoritmo, lo cual lo hace muy atractivo para implementaciones para predicción on-line. La predicción óptima estará basada en el modelo supuesto de comportamiento de la demanda (puede ser por ejemplo ARIMA), y debe ser conocido antes de la implementación del filtro Kalman. La dificultad principal en la aplicación de la técnica de espacio de estados, radica en el problema de hallar las matrices de covarianza de ruido Q_k y R_k, las cuales no pueden ser fácilmente determinadas.

2.2.4 Análisis de regresión lineal múltiple. La existencia de más de una variable independiente explicativa del comportamiento de una serie de tiempo y la posibilidad de proyectar este comportamiento en el futuro, en términos de estas variables, permite en previsiones de demanda (energía o potencia) expresar el consumo de potencia horaria en términos de variables de tipo climáticas, temporales o económicas que puedan afectar el comportamiento de la demanda eléctrica. En el caso de esta técnica de predicción, la demanda se expresa como una función lineal de variables independientes afectadas por los respectivos coeficientes de regresión lineal. El coeficiente de regresión se calcula por medio de mínimos cuadrados y se evalúa su eficiencia mediante pruebas estadísticas como la prueba-F y la Prueba-T. El modelo de predicción poseerá la forma:

$$\hat{Y}(t) = a_0 + a_1 X_1(t) + \ldots + a_n X_n(t) + A(t) \quad \text{Ec. 24}$$

Donde:

$\hat{Y}(t)$: Serie de tiempo a predecir

a_0 : Valor constante

$a_1,..., a_n$: Coeficientes de regresión lineal neta

$X_1(t), X_n(t)$: Variables independientes correlacionadas con $Y(t)$

$A(t)$: Variable aleatoria con media cero y varianza constante

En cuanto a las variables independientes, estas son identificadas de acuerdo al análisis de correlación entre cada variable explicativa (independiente) y la carga en watts del sistema (variable dependiente). Un aspecto fundamental en esta técnica, consiste en determinar mediante un juicio a priori dado por la experiencia, cuales de estas variables independientes pueden ser influyentes dentro del modelo de regresión múltiple. Una vez se han establecido las posibles variables explicativas, se seleccionan cuales son las más influyentes por sus coeficientes de correlación y determinación. Para esto, un buen análisis de regresión múltiple requiere que el coeficiente de correlación se calcule entre pares de variables formando una matriz de coeficientes de correlación, la cual muestra la influencia de una variable con respecto a la otra (no solamente de la variable independiente, con respecto a la variable dependiente, sino también entre las independientes). Como en esta matriz se ven coeficientes de correlaciones entre variables independientes, se puede apreciar efectos como la colinealidad, la cual se presenta cuando las variables independientes están altamente intercorrelacionadas, presentándose dependencia entre ellas. En otras palabras, los atributos de una buena variable de predicción son: Estar relacionada con la variable dependiente a predecir y no estar altamente relacionada con otra variable independiente.

Además del coeficiente de correlación, en regresión lineal múltiple existe otro coeficiente que caracteriza una buena coordinación entre las variables explicativas y la variable dependiente; este es el coeficiente de determinación, y su importancia radica en que mide el porcentaje de variabilidad en Y que puede explicarse a través del conocimiento de las variables independientes X seleccionadas en el modelo de predicción. Una forma de representar este coeficiente es la siguiente:

$$CD = 1 - \frac{Desviación\ \ NoExplicada}{Desviación\ \ Total} = 1 - \frac{\sum (Y - \hat{Y})}{\sum (Y - \bar{Y})} \quad \text{Ec. 25}$$

Donde:

Y : Valor real de la serie de tiempo

$\bar{Y}$: Valor medio de los datos

$\hat{Y}$: Valor predicho de la serie

2.2.5 Técnicas basadas en inteligencia artificial. Con base en una definición de [Haykin-1999], se puede decir que la inteligencia artificial tiene como fin el desarrollo de paradigmas o algoritmos que emplean las máquinas para realizar tareas que aparentemente requieren cognición[8] cuando son ejecutadas por humanos. Un sistema basado en inteligencia artificial deberá, entonces, ser capaz de realizar tres cosas: Almacenar conocimiento, aplicar el conocimiento almacenado para resolver problemas y adquirir conocimiento a través de nuevas experiencias.

Entre los principales paradigmas en inteligencia artificial se pueden nombrar los siguientes:

- Redes Neuronales
- Sistemas Difusos
- Temple Simulado

[8] Se ha usado el término "cognición" (proceso mediante el cual se adquiere conocimiento) en vez de "inteligencia" para incluir dentro de las tareas aspectos tales como percepción, lenguaje, solución de problemas.

- Sistemas Expertos

- Algoritmos Genéticos

Además de las tres actividades mencionadas anteriormente, para que un sistema sea considerado como “inteligente” deberá estar compuesto de estructuras capaces de representar, razonar y aprender. A continuación serán descritas estas características en el contexto de la inteligencia artificial.

2.2.5.1 Capacidad de Representar. Uno de los rasgos más distintivos de los sistemas inteligentes es probablemente el uso de un lenguaje de estructuras simbólicas que representa tanto el conocimiento general acerca del dominio de interés del problema, como el conocimiento específico acerca de la solución del problema. El término “conocimiento" usado en el campo de inteligencia artificial, puede ser atribuido simplemente a la representación simbólica de datos, la cual puede ser del tipo declaratorio o procesal. En una representación declaratoria, el conocimiento se representa como una colección estática de “hechos”, más un conjunto pequeño de procedimientos generales que manipulan estos “hechos”. En una representación procesal, el conocimiento está incluido dentro de un código ejecutable que representa el significado del conocimiento. Ambos tipos de conocimiento, declaratorio y procesal, se necesitan en la mayoría los dominios del problema de interés.

2.2.5.2 Capacidad de Razonar. En su forma más básica, razonar es la habilidad de resolver problemas. Para que un sistema clasifique como un sistema de razonamiento, debe satisfacer las siguientes condiciones:

- Debe poder expresar y resolver un amplio rango de problemas y problemas tipo.

- Debe ser capaz de hacer explícita cualquier información implícita conocida por él.

- Debe tener un mecanismo de control que determine qué operaciones aplicar a un problema en particular, cuando una solución al problema se ha obtenido, o cuando el trabajo extenso en el problema debe terminarse.

2.2.5.3 Capacidad de Aprender. Una máquina que aprende, puede involucrar dos tipos diferentes de procesamiento de la información: el procesamiento inductivo y el deductivo. En el procesamiento de información inductivo, los patrones y/o reglas generales, son determinados a partir de datos y de la experiencia. Por otro lado, en el procesamiento de información deductivo, se usan reglas generales para determinar hechos específicos. La importancia de bases de conocimiento y las dificultades experimentadas en el proceso de aprendizaje, han llevado al desarrollo de varios métodos apoyados en bases de conocimiento. Específicamente, si hay expertos en un campo dado, es normalmente más fácil obtener la experiencia compilada de los expertos, que intentar reproducir y dirigir la experiencia que llevó al experto a su grado de conocimiento en el campo dado. Esta idea permite el surgimiento de los sistemas expertos.

Uno de los paradigmas mas desarrollados de la inteligencia artificial en la actualidad son las Redes Neuronales Artificiales (RNA), las cuales, al igual que en el sistema nervioso humano, se constitutyen como procesadores distribuidos conectados masivamente en paralelo, con una natural tendencia a adquirir conocimiento de la experiencia y hacerlo disponible para el uso. Su parecido con el cerebro biológico se centra principalmente en dos aspectos:

- El conocimiento es adquirido por la red a través de un proceso de aprendizaje.

- Se usan conexiones entre neuronas, conocidas como pesos sinápticos, para guardar el conocimiento adquirido.

Como en la naturaleza, el funcionamiento de la red está ampliamente determinado por las conexiones entre los elementos de procesamiento o somas. A una red neuronal artificial se le puede entrenar para realizar una función particular

ajustando los valores de las conexiones (pesos sinápticos) entre los elementos, a través de un proceso de aprendizaje. Al orden de acciones realizado dentro del proceso de aprendizaje se llama algoritmo de entrenamiento, y su función consiste, principalmente, en modificar los pesos sinápticos de la red de una manera ordenada, con el fin de lograr a la salida un objetivo deseado, partiendo de determinadas entradas a la red[9].

Se puede decir entonces, que la red neuronal artificial obtiene su fortaleza computacional, en primer lugar, a partir de su estructura distribuida conectada masivamente en paralelo, en segundo, de su habilidad para aprender a partir de ejemplos mostrados y su capacidad de generalización[10] y, en tercer lugar, su característica no-lineal. Estos tres aspectos anteriores hacen posible para las redes neuronales artificiales el resolver problemas que matemáticamente y computacionalmente pueden resultar muy complicados.

Dentro de las bondades ofrecidas por las redes neuronales para la solución de problemas complejos en diferentes áreas del conocimiento se tienen las siguientes:

- **No-linealidad:** Una neurona es básicamente un elemento no-lineal. Por consiguiente, una red neuronal compuesta por una interconexión de neuronas, es también no-lineal. Adicionalmente, la no-linealidad que se presenta es de tipo especial por ser distribuida a través de toda la red.

- **Correspondencia Entrada-Salida:** Un paradigma popular para el aprendizaje de la red, llamado entrenamiento supervisado, involucra la modificación de los pesos de la red neuronal a medida que se le muestra un conjunto de tareas o ejemplos. Cada ejemplo consiste en una única señal de entrada y su

[9] La fundamentación teórica sobre redes neuronales artificiales presentada aquí, está basada en las siguientes referencias bibliográficas [Haykin-96],[Demuth-98], [Freeman-93] y [Lippman-87]
[10] Generalización se refiere a la capacidad de la red para producir salidas razonables para entradas que durante el proceso de entrenamiento (aprendizaje) no le fueron mostradas.

correspondiente respuesta a la salida. El entrenamiento consiste en presentar a la red un ejemplo escogido al azar del conjunto de ejemplos, y modificar los pesos (conexiones entre elementos de procesamiento) para minimizar la diferencia entre la respuesta deseada y la respuesta real, de acuerdo con un criterio estadístico seleccionado. Este proceso se repite para muchos ejemplos, hasta que la red alcance un "estado estable", en el cual no existan cambios significativos en los valores de los pesos al presentar un nuevo ejemplo. El resultado final después del entrenamiento, es un sistema de procesamiento capaz de responder con una salida adecuada, a cualquier señal de entrada que se le presente, sin importar que esta señal de entrada se encuentre fuera de rango, o que nunca se le haya mostrado en el proceso de entrenamiento.

- **Adaptatividad:** Las redes neuronales tienen la capacidad para adaptar sus pesos a los cambios en el comportamiento de los datos; es decir, si una red se entrenó para operar en un ambiente específico y este ambiente cambia (no drásticamente), la red puede re-entrenarse para trabajar con los cambios menores que se presenten en el transcurso del tiempo, de una manera más fácil.

- **Respuesta evidenciada:** En el contexto de la clasificación de patrones, una red neuronal puede ser diseñada no sólo para proporcionar información sobre el modelo particular que se seleccionará, sino también sobre la confianza en la decisión hecha. Esta última información puede ser usada para explicar la causa del rechazo de los patrones ambiguos, que puedan surgir, y por ende, mejorar el desempeño de la red en este tipo de tareas.

- **Distribución de la información contextual:** Como ocurre en el trabajo con información numérica, el conocimiento, a nivel general, es representado por una estructura y estado de activación de la red neuronal. Cada neurona en la red es potencialmente afectada por la actividad global de todas las otras

neuronas en la red. Por consiguiente, la información contextual también se reparte naturalmente por la red neuronal.

- **Implementabilidad en hardware:** La naturaleza de procesamiento paralelo masivamente conectado, hace que una red neuronal sea potencialmente rápida para labores computacionales de ciertas áreas. El desarrollo de la electrónica ha permitido implementar redes neuronales en chips de computador, capaces de realizar labores equivalentes y con mayor agilidad o desempeño que las implementaciones comunes en software. A este tipo de implementaciones se les conoce como integración a gran escala y se designan con la sigla VLSI (Very Large Scale Integrated). La virtud particular de la tecnología VLSI es que proporciona medios para capturar verdaderamente la complejidad de un comportamiento en una manera muy jerárquica, lo cual hace posible usar estas implementaciones hardware como herramientas en aplicaciones de tiempo real que involucran reconocimiento de patrones, procesamiento de señales y control.

- **Tolerancia a fallas:** Debido a la alta interconectividad, cuando una red neuronal implementada en hardware posee una neurona que falla o alguna conexión entre ellas se deteriora, el sistema en conjunto puede salir airoso del impase, ya que la información está distribuida a través de toda la red.

- **Uniformidad en el análisis y diseño:** A pesar de la amplia variedad de aplicaciones posibles, en su estructura básica las redes neuronales disfrutan de universalidad como procesadores de información, ya que en primer lugar, todas las redes neuronales comparten características comunes tales como: tipos de arquitectura de red, funciones de activación o algoritmos de entrenamiento; en segundo lugar, pueden construirse redes en forma modular que permitan integrar redes que desarrollen tareas complementarias.

- **Analogía neurobiológica:** El diseño de una red neuronal es motivado por

analogía con el cerebro, el cual es una prueba viviente de que el procesamiento paralelo tolerante a fallas no es sólo físicamente posible, sino también rápido y poderoso. Algunos investigadores en neurobiología han utilizado a las redes neuronales artificiales como herramienta de investigación para la interpretación de fenómenos biológicos.

Por su importancia dentro de este trabajo de investigación, se ampliará con mayor detalle lo referente a la técnica de predicción mediante redes neuronales, en el siguiente capítulo.

2.3. EL PROCESO DE PREDICCIÓN Y LAS FUENTES DE ERROR

Todo proceso formal de predicción entendido en su contexto general, como la extensión de las experiencias del pasado al futuro incierto, tendría los siguientes pasos en común, como se plantea en [Hanke-1996]:

1. Recopilación de datos.
2. Reducción o condensación de datos.
3. Construcción del modelo.
4. Extrapolación del modelo (predicción en sí).

El primer paso sugiere la importancia de obtener los datos adecuados y asegurarse que son correctos. Con frecuencia este paso es el mayor reto de todo el proceso de predicción y el más difícil de controlar, ya que los pasos siguientes se efectúan sobre los datos, sean o no relevantes para el problema en cuestión.

El segundo paso, plantea la necesidad de establecer la relevancia de los datos

dentro del modelo propuesto; por lo general, la reducción de datos es necesaria ya que en el proceso de predicción es posible tener muchos o muy pocos datos. En este aspecto es importante diferenciar dos problemas; uno relacionado con la cantidad de datos y otro relacionado con la determinación de variables influyentes dentro del modelo de predicción.

El paso 3, la construcción del modelo, implica el ajustar los datos reunidos en un modelo de predicción que sea el adecuado para minimizar el error en el pronóstico. Entre más sencillo sea el modelo, será mejor para lograr la aceptación del proceso por parte de los administradores que toman las decisiones en la empresa. Aspectos como la manera de cargar los datos a modelo, tiempo de ejecución de la predicción, facilidad de manejo de la herramienta computacional que lo aplique, además de un bajo margen de error en la predicción, son aspectos influyentes a la hora de establecer el modelo e implementar lo dentro de una herramienta computacional. Es común dentro de este paso establecer dos conjuntos de datos: unos datos empleados para el establecimiento del modelo de predicción (que en el caso de modelos basados en redes neuronales se conocen como "datos de entrenamiento") y un conjunto menor de datos conocidos como "datos de validación" empleados para probar, una vez el modelo se ha formulado, si las predicciones son confiables dentro de unos límites de error establecidos, permitiendo que quien realizó el pronóstico revise la precisión del proceso mediante el pronóstico de periodos recientes de los que se conocen los valores históricos reales. Es entonces cuando se observan los errores de pronóstico y se resumen de algún modo en indicadores de error en la predicción como Desviación Estándar, Desviación Absoluta de la Media, Error Medio Cuadrado, Porcentaje de Error Medio Absoluto, y Porcentaje Medio de Error, los cuales serán tratados más adelante.

El último paso consiste en la extrapolación en sí del modelo de predicción, lo cual ocurre una vez que se recolectaron y tal vez redujeron, los datos adecuados y que se seleccionó un modelo de predicción apropiado. Es claro que una vez se

establece el modelo de predicción, las variables que intervienen en él quedan fijas, siendo los datos de la serie de tiempo histórica, los únicos valores que cambiarían. Esto implica que durante la etapa de "producción" del modelo se deben hacer continuos monitoreos del error en la predicción, a fin de establecer cuando es necesario reformular el modelo para que se adapte a los cambios en el comportamiento de la serie de tiempo.

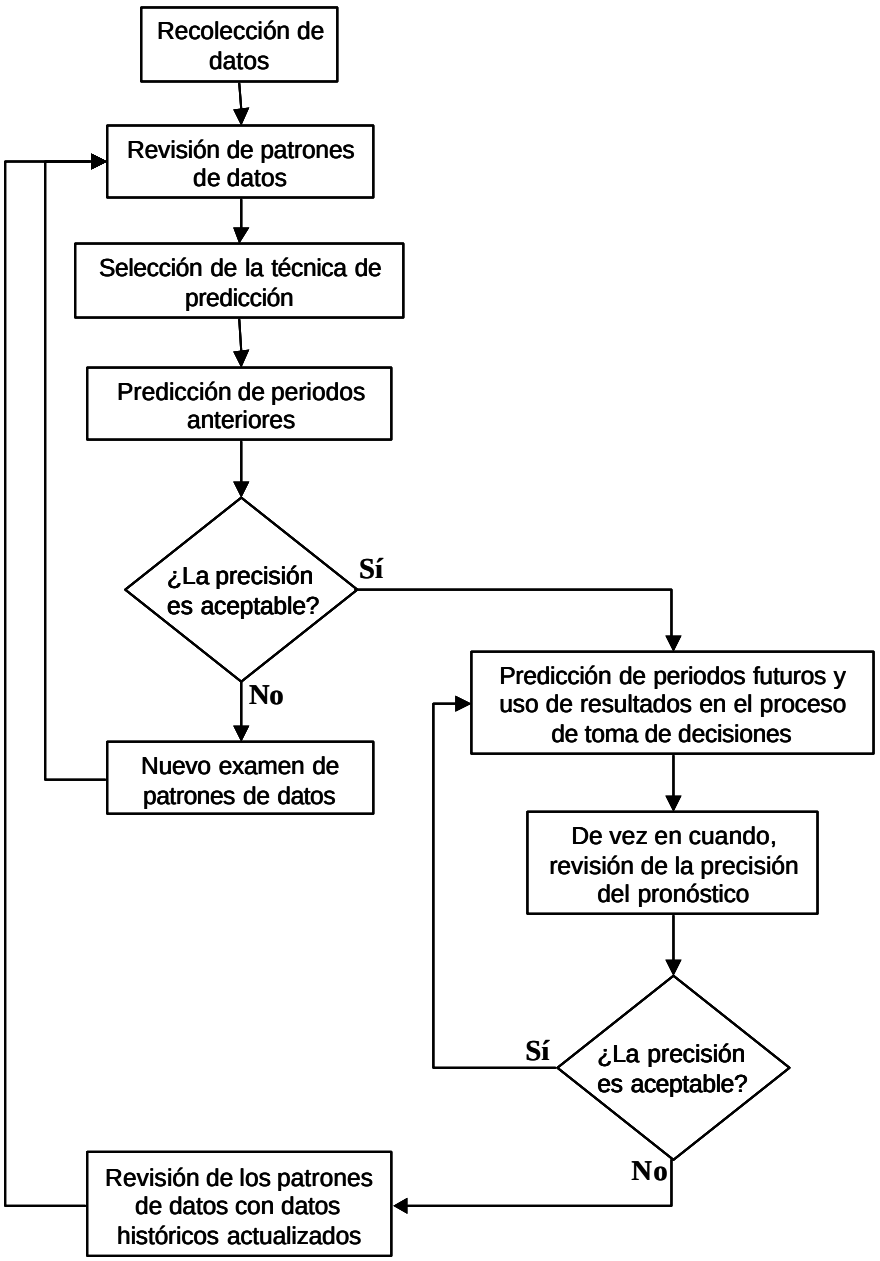
En la figura 10 se muestra un diagrama de flujo que plantea el proceso de generación de predicciones de series temporales a nivel general (sin importar la técnica de predicción empleada o el horizonte del pronóstico). Se observa en ella que la generación de una predicción útil implica dos consideraciones básicas antes de la construcción y extrapolación del modelo[11]. En Primera instancia es necesario reunir y analizar los datos que sean aplicables para la tarea de predicción, siendo estos continuos, pertinentes y coherentes.

Una vez se ha cumplido este requerimiento, se procede en segunda instancia, a seleccionar la técnica de predicción mas adecuada según aspectos tales como el horizonte de predicción, patrones de datos, necesidades de predicción y márgenes de errores esperados versus complejidad de los algoritmos empleados. No obstante, estos son los primeros pasos en un proceso efectivo y dinámico de predicción.

Como sugiere la figura 10 el siguiente paso consiste por lo regular en pronosticar periodos históricos anteriores cuyos valores reales se conocen. Los errores resultantes se pueden resumir en varias formas, como se expuso anteriormente y, este proceso se continúa hasta encontrar una técnica o un modelo con una proporción aceptable de precisión.

[11] En este documento, en cuanto al tema de predicción de demanda convencional y las técnicas o procedimientos estadísticos empleados, se toman como base las siguientes referencias: [Chaw-96], [Gross-87], [Hanke-96], [Leon-94], [Moghram-89], y [Portus-85].

Figura 10. Procedimiento general de predicción de series temporales



Una vez seleccionado el modelo mediante pronósticos de períodos anteriores donde se conocen los resultados, a fin de establecer la precisión del mismo, se inicia el proceso de predicción de períodos a futuro y los resultados se incorporan al proceso de toma de decisiones de la empresa.

De vez en cuando es necesario hacer una pausa en el proceso de pronóstico y reconsiderar los procedimientos que se están utilizando. Los pasos usuales son los siguientes:

1. Se descartan los valores históricos más antiguos que se están empleando en el proceso de pronóstico y se agregan al banco de datos los valores reales más recientes.

2. Después de esta actualización de datos, se recalculan los parámetros utilizados en el modelo de predicción (por ejemplo constantes de atenuación en el suavizado exponencial, coeficientes de regresión o pesos de una red neuronal), ya que estos pueden cambiar al agregar valores de datos más recientes.

3. Se examina el modelo de predicción con nuevos parámetros para una adecuada precisión. Si se juzga que esta precisión es suficiente, el modelo se usa como antes hasta el siguiente período de actualización. Si la precisión del pronóstico se juzga inadecuada o marginal, se pueden revisar los patrones en los nuevos datos con la posibilidad de elegir un nuevo modelo de predicción. Este proceso continúa hasta que se juzga adecuada la precisión del modelo elegido, de acuerdo con la precisión de pronóstico de períodos históricos anteriores.

El proceso presentado anteriormente y resumido en el diagrama de flujo de la figura 10, constituye el tipo de ciclo de retroalimentación que se encuentra comúnmente en diseños de todo tipo de sistemas empleados para la predicción de series temporales.

Es común encontrar en los sistemas de predicción de series temporales, esquemas que monitorean los pronósticos de manera constante mediante una señal de rastreo que represente el error en la predicción. La idea consiste en establecer límites dentro de los cuales se espera que se ubiquen los errores generados por el pronóstico, si el proceso de pronóstico es adecuado. Mientras que los errores se ubiquen dentro de estos límites aceptables, el proceso de pronóstico continúa. Tan pronto como un error cae fuera del margen aceptable, la atención de la administración se enfoca en el proceso de pronóstico y se retoman los pasos de actualización y revisión antes señalados.

En cada uno de los pasos mencionados anteriormente existen fuentes de error que al acumularse pueden llegar a tener magnitudes que descartan un modelo de predicción, que en sí mismo pudo ser adecuado. Una de las principales fuentes de error está relacionada con los datos; ya se mencionó anteriormente que su selección y relevancia como variable explicativa del modelo de predicción son importantes, no solo en su establecimiento sino en la producción de predicciones. Estos problemas son fácilmente solucionados mediante análisis de correlación entre las "variables explicativas del modelo" y la serie de tiempo a predecir.

No obstante, el problema de mayor frecuencia radica en la obtención de datos continuos (presentes durante toda la serie de tiempo) y la coherencia de dichos datos. En el caso específico de las series de tiempo relacionadas con la demanda de potencia horaria, son frecuentes errores en los datos como en los ejemplos a continuación:

- Datos con decimales fuera de posición como por ejemplo: 25,35 MW cuando debería ser 253,5 MW.
- Datos faltantes en algunas hora del día, ya sea por fallas eléctricas o simplemente que al operario de la subestación encargado de tomar el dato se le olvidó.

- Datos no reales cuando en el caso de que el operario se le olvide tomar el dato, escriba en la planilla cualquier dato cercano con el fin de llenarla completamente.

- Datos anormales debidos a condiciones especiales de operación del sistema, ante alguna contingencia. Es decir, el dato de la planilla es real, pero para efectos de predicción debe ser descartado por tener condición anormal.

A fin de solucionar problemas en los datos como los anteriores, se puede realizar una análisis estadístico sencillo basado en medidas de tendencia central. Al estimar la media de los datos en cada hora del día y calcular su desviación estándar, se puede establecer cuales datos se encuentran fuera de su valor normal según una o dos desviaciones estándar. Se puede plantear para este caso una algoritmo que recorra los datos y evalué cuales de ellos están errados y los reemplace por la media de los datos a la misma hora. Otra alternativa planteada en [Portela-2000] utiliza una red neuronal para establecer tendencias de datos y reemplazar aquellos fuera de rango. Sin embargo, para fines del estudio realizado, se empleará el procesamiento estadístico por considerarse menos complejo.

En cuanto a la manera de evaluar un modelo de predicción, es claro que la única forma de saber si el modelo de predicción es adecuado, consistiría en la comparación de los datos reales y los datos predichos. Se tendría de esta manera para cada hora de la serie un residuo resultante de la diferencia entre la serie real $\boldsymbol{Y_t}$ y la serie predicha $\widehat{\boldsymbol{Y_t}}$, conocido como error residual $\boldsymbol{e_t}$.

$$e_t = Y_t - \widehat{\boldsymbol{Y_t}} \quad \text{Ec. 26}$$

Dado que se tendría una serie de tiempo de errores (esto es, para cada dato de tiempo existiría un error), se han ideado diversas formas de medir el error "total" generado en la predicción. Ciertos procedimientos de predicción suman los

valores absolutos de los errores y pueden reportar esta suma, o dividirla entre el número de intentos de pronóstico para obtener el error de predicción promedio. Otros procedimientos obtienen la suma de cuadrados de los errores, que se compara luego con cifras similares de métodos de predicción alternativos. Algunos procedimientos también rastrean y reportan la magnitud de los términos de error sobre el periodo de predicción. El examen de los patrones de error conduce con frecuencia al analista a la modificación del procedimiento de pronóstico, el cual genera después pronósticos más precisos. Los métodos específicos de medición de error en la predicción se presentan a continuación.

Un método para evaluar una técnica de predicción consiste en obtener la suma de los errores absolutos. La Desviación Absoluta de la Media (***MAD*** por sus iniciales en inglés) mide la precisión de un pronóstico mediante el promedio de la magnitud de los errores de predicción (valores absolutos de cada error). La desviación absoluta de la media resulta de gran utilidad cuando el analista desea medir el error de la predicción en las mismas unidades de la serie original. En la ecuación 27 se muestra la manera de calcularla.

$$MAD = \frac{\sum_{t=1}^{n} \left| Y_t - \hat{Y}_t \right|}{n} \qquad \text{Ec. 27}$$

Otro método para evaluar una técnica de predicción es el Error Medio Cuadrado (***MSE*** por sus iniciales en inglés). Cada error o residual se eleva al cuadrado, luego, estos valores se suman y se divide entre el número de observaciones. Este enfoque penaliza los errores mayores de pronóstico ya que eleva cada uno al cuadrado, lo cual es importante en ocasiones donde es preferible escoger un modelo que produzca errores moderados en vez de un modelo que produzca errores bajos pero que ocasionalmente arroje algunos en extremo grandes, como es el caso del problema de predicción de demanda, en donde se pueden tener errores bajos en casi la totalidad de la curva, pero en horas punta los errores

aumentan considerablemente. El error medio cuadrado puede calcularse como se muestra en la ecuación 28.

$$MSE = \frac{\sum_{t=1}^{n} \left(Y_t - \hat{Y}_t\right)^2}{n} \quad \text{Ec. 28}$$

No obstante, el ***MSE*** quedaría en unidades al cuadrado (p.e. kW^2) lo cual hace difícil su comparación. Por tal motivo se emplea comúnmente la raíz del error medio cuadrado (***RMSE*** por sus iniciales en inglés) como se muestra en la siguiente ecuación, a fin de tener un valor de error en las unidades de los datos y complementar la información arrojada por el ***MAD***.

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n} \left(Y_t - \hat{Y}_t\right)^2}{n}} \quad \text{Ec. 29}$$

En ocasiones, resulta mas útil calcular los errores de predicción en términos de porcentaje y no en cantidades con las mismas unidades de los datos. El Porcentaje de Error Medio Absoluto (***MAPE*** por sus iniciales en inglés) se calcula encontrando el error absoluto en cada periodo, dividiendo éste entre el valor real observado para ese periodo y después promediando estos errores absolutos entre el número de observaciones. Este enfoque es útil cuando el tamaño o magnitud de la variable de predicción es importante en la evaluación de la precisión de la predicción. Esta forma de error proporciona una indicación de que tan grandes son los errores de pronóstico comparados con los valores reales de la serie. También se puede utilizar el ***MAPE*** para comparar la precisión de la misma u otra técnica sobre dos series completamente diferentes. La ecuación 30 muestra el cálculo del ***MAPE***.

$$MAPE = \frac{\sum_{t=1}^{n} \frac{\left|Y_t - \hat{Y}_t\right|}{Y_t}}{n} \quad \text{Ec. 30}$$

A veces resulta necesario determinar si un método de pronóstico esta sesgado (pronóstico consistentemente alto o bajo). En estos casos, se emplea el Porcentaje Medio de Error (***MPE*** por sus iniciales en inglés), el cual se calcula encontrando el error en cada periodo, dividiendo esto entre el valor real de ese periodo y promediando después estos porcentajes de error. Si un enfoque de pronóstico no esta sesgado, la ecuación 31 producirá un porcentaje cercano a cero. Si el resultado es un porcentaje negativo grande, el método de predicción está sobrestimando de manera consistente. Si el resultado es un porcentaje positivo grande, el método de pronóstico esta subestimando en forma consistente.

$$MPE = \frac{\sum_{t=1}^{n} \frac{\left(Y_t - \hat{Y}_t\right)}{Y_t}}{n} \quad \text{Ec. 31}$$

Una parte de la decisión para utilizar una técnica de predicción o un modelo en particular, es la determinación de si la técnica producirá errores de predicción que se juzguen como suficientemente pequeños. Es en efecto realista esperar que una técnica produzca errores de predicción relativamente bajos sobre una base consistente.

Las cuatro mediciones de precisión de la predicción pueden ser empleadas para la comparación de la precisión de dos modelos diferentes, la medición de la utilidad o confiabilidad de un modelo y la búsqueda de un modelo óptimo.

3. CARACTERÍSTICAS DE LA CURVA DE DEMANDA

Con el objeto de probar las diferentes técnicas de predicción y establecer sus ventajas y desventajas, se considera importante dedicar un espacio a las series de tiempo empleadas para la validación de los modelos propuestos.

A pesar de que la teoría presentada hasta el momento es general para cualquier tipo de serie de tiempo en el corto plazo, uno de los objetivos del trabajo de investigación plantea que los modelos sean probados con datos de demanda de potencia horaria de la Electrificadora de Santander S.A., permitiendo de esta manera comparar con resultados de proyectos de grado anteriores en el tema y manejar series de tiempo "conocidas" a fin de establecer explicaciones en los resultados obtenidos por los modelos.

3.1 COMPORTAMIENTO DE LOS DATOS UTILIZADOS

En este nivel de agregación (demanda atendida por un agente comercializador, es decir carácter regional), la curva de demanda se ve influenciada por el efecto del consumidor individual que el imprime una componente aleatoria ya importante en el modelo. En el caso específico de la curva de demanda de los clientes atendidos por la ESSA (patrones empleados como base para el planteamiento del esquema de predicción de este trabajo de investigación), no se observan efectos estacionales, ni la influencia de variables climáticas como temperatura diaria, humedad o brillo solar. En [Rodriguez-2000] se demostró mediante un análisis estadístico de componentes principales (PCA), que estas variables no tienen una relación directa entre valores de la carga horaria. Para una mayor amplitud sobre el método de componentes principales se puede consultar [Jollife-1986].

Desde el punto de vista de la aplicación de la previsión de demanda en la operación de la red de distribución, el comportamiento del consumidor tiene una incidencia aún mayor que en la anterior ya que el grado de agregación esta en su mínima expresión, por considerar alimentadores individuales. No obstante, este trabajo de investigación tiene por objeto predecir la demanda agregada para fines de transacciones comerciales de potencia y energía, por lo tanto el modelo de curva de demanda del párrafo anterior es más adecuado.

A continuación se presentarán algunas características de la curva de demanda que se trabajará dentro del esquema de predicción planteado en este trabajo de investigación. Es de aclarar que el modelo planteado, así como el prototipo, están diseñados para ser aplicados en cualquier sistema de distribución, e incluso podría emplearse mediante pequeñas modificaciones al prototipo para predecir otros tipos de series de tiempo de características similares, como por ejemplo el despacho de gas natural, o el consumo de agua de un municipio.

Para empezar, los datos de demanda empleados para probar el modelo de predicción están compuestos por usuarios regulados[12] y no regulados, así como algunas cargas de Centrales Eléctricas de Norte de Santander y Electroboyacá, atendidos eventualmente por la ESSA. No obstante, dada la clara diferenciación entre demanda regulada y no regulada, así como la totalidad de los datos para estas cargas, el trabajo se realizará sobre pruebas en usuarios regulados y no regulados de la ESSA.

Para probar los modelos se emplearon datos de demanda regulada y no regulada desde Junio del 2000, hasta Septiembre del 2003. No obstante, para efectos de comparaciones con trabajos anteriores realizados en la Escuela de Ingeniería Eléctrica de la UIS, se emplearán en una prueba de comparación inicial de los modelos de predicción datos de demanda de 1997 y 1998.

---

[12] Usuario regulado: Consume menos de 0,1 MW mensuales y no puede comprar directamente su energía.

En la figura 11 se puede observar una curva de demanda promedio total atendida por la ESSA. Es notable el predominio de la carga de tipo residencial, la cual se evidencia con una pequeña punta de carga alrededor de las 6 a.m. cuando las personas se preparan para ir a sus trabajos y una gran punta de carga alrededor de las 7 p.m. cuando la mayoría de luces de la casa están encendidas y la mayoría de personas están en casa viendo televisión o empleando otros electrodomésticos. La demanda máxima promedio de los datos empleados para la prueba del modelo es del orden de los 250 MW, y la mínima se encuentra alrededor de los 125 MW. La energía total consumida es del orden de 4010 MWh por día.

La demanda no-regulada, es decir la debida principalmente a usuarios industriales con demandas máximas superiores a 0,1 MW, con la capacidad para comprar energía directamente o a través de un comercializador como es el caso de la ESSA, posee una demanda máxima alrededor de los 15 MW entre las 10 y las 11 a.m y su mínima alrededor de 8 MW, en horas de la madrugada y la Energía total consumida es del orden de 280 MWh por día. Se observa además en la figura 12, el comportamiento del trabajo en las empresas que se constituyen como usuarios no regulados, principalmente ubicadas en la zona industrial del Área Metropolitana del Municipio de Bucaramanga. Ingresan a laborar alrededor de las 7 a.m., descansando al medio día y terminado labores alrededor de las 6 p.m. Adicionalmente se puede observar en las figuras 12 y 13 el comportamiento desagregado de los usuarios no-regulados y regulados, quienes al ser sumados con las pérdidas de energía (figuras 14 y 15) y otras transacciones e energía de la ESSA hacia otras electrificadoras, conforman la demanda total.

Figura 11. Demanda promedio de potencia horaria total ESSA

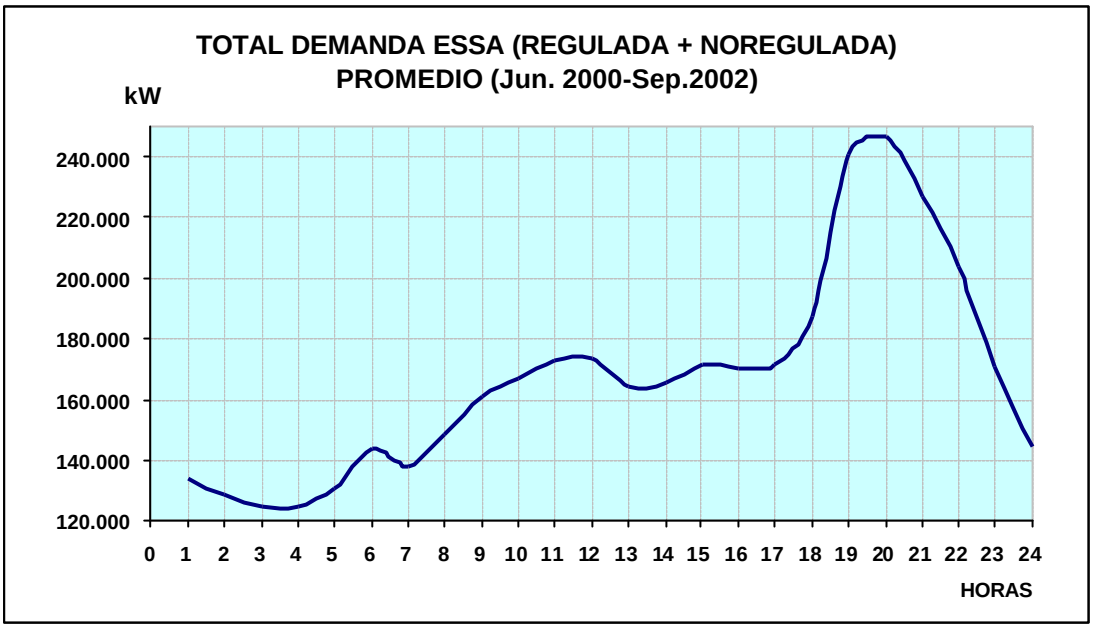
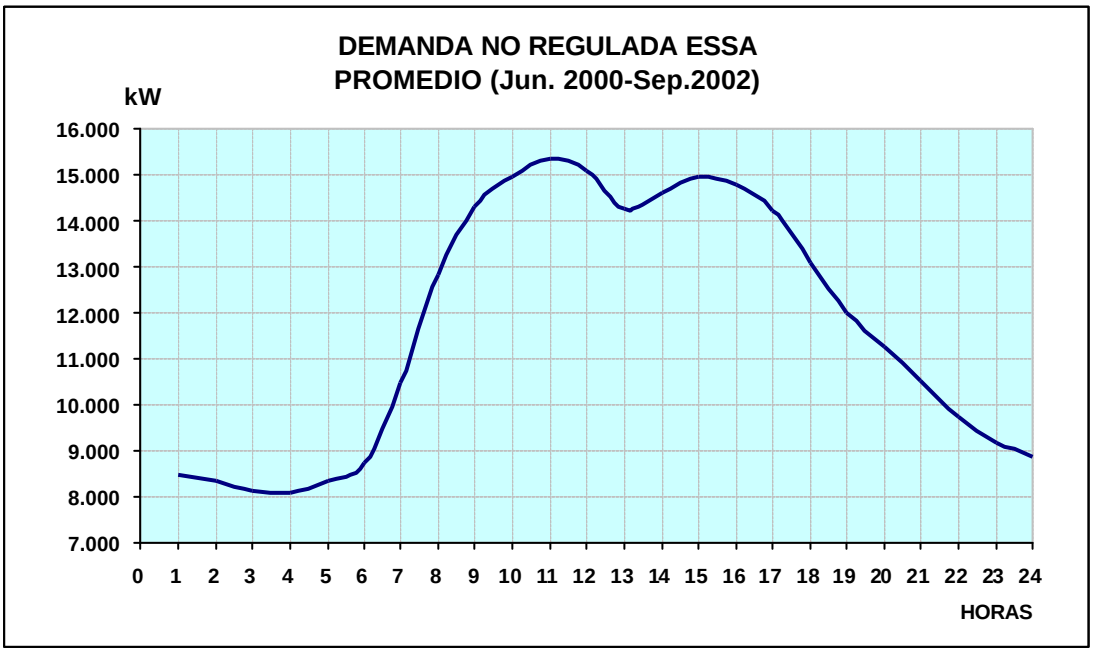


Figura 12. Demanda no regulada promedio ESSA



Al comparar la curva no-regulada con la total, se observa que la demanda industrial de la ESSA es muy baja (7% de la demanda total), lo cual justifica la notoria la similitud entre la demanda total y la demanda regulada, mostrada en la figura 13.

Figura 13. Demanda regulada promedio ESSA

En cuanto a los usuarios regulados, su composición es predominantemente residencial, son clientes "cautivos" de la comercializadora y está sujetos a tarifas fijas reguladas por la CREG. En la figura 13 se observa la curva de demanda promedio de los usuarios regulados, de los datos de estudio. Su demanda máxima está alrededor de los 230MW, ocurriendo entre las 7 p.m. y las 8 p.m. La demanda mínima es alrededor de los 115 MW y la energía total consumida es del orden de 3670 MWh por día (91% de la demanda total).

El modelo planteado dentro de este trabajo de investigación permite la predicción de cualquiera de las diferentes curvas de demanda mencionadas en los párrafos anteriores e inclusive la predicción de pérdidas, que complementan el balance

energético. Estas pueden ser debidas a los usuarios regulados y no-regulados que suman alrededor de los 56 MWh y 4 MWh por día respectivamente. En las figuras 14 y 15 se muestra el comportamiento de las perdidas.

Figura 14. Pérdidas promedio de energía para usuarios no-regulados

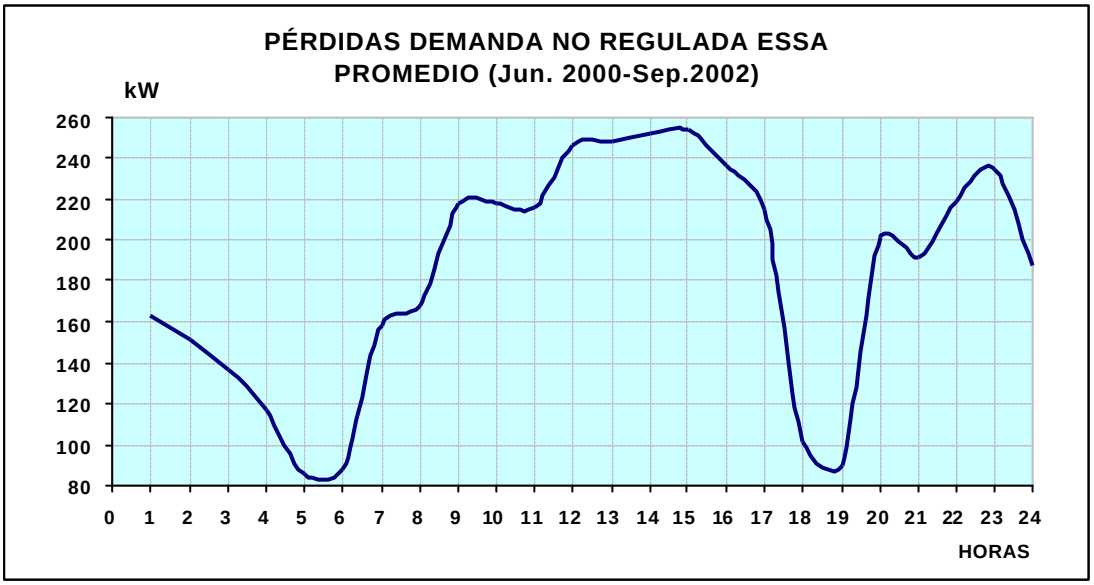
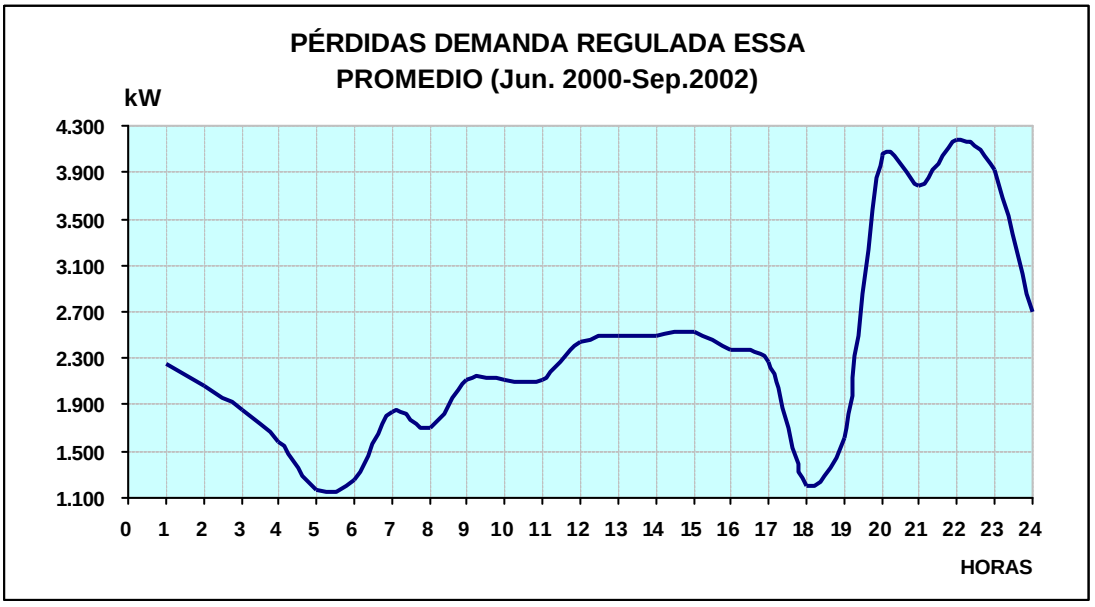


Figura 15. Pérdidas promedio de energía para usuarios regulados



Es de anotar que los datos son de comercialización y por tanto, la información de demanda regulada y no-regulada es la facturada (medida en el contador del cliente) por la comercializadora, siendo entonces las pérdidas una diferencia entre lo facturado y lo medido en los alimentadores, empleándose como ajuste (las cuales estarían alrededor de un 1%). En ningún caso corresponden a pérdidas técnicas en el sistema de distribución, ya que estas estarían por encima de un 10% de la demanda total.

3.2 DIFERENCIACIÓN DE LOS DATOS EMPLEADOS

Al observar los datos a emplear en el modelo, se observa tanto en los datos de demanda regulada como en los de no-regulada, una clara diferenciación entre los días Martes, Miércoles, Jueves y Viernes, con los Sábados, Domingos y Lunes. En las figuras 16 y 17 puede ser observada.

Figura 16. Diferenciación diaria de los datos de demanda regulada

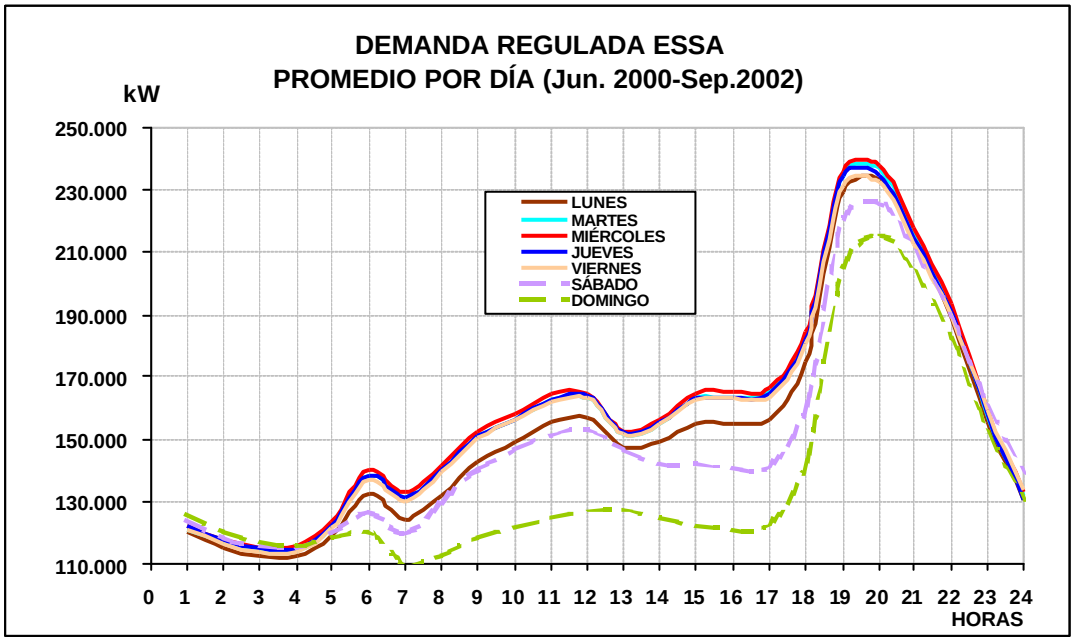
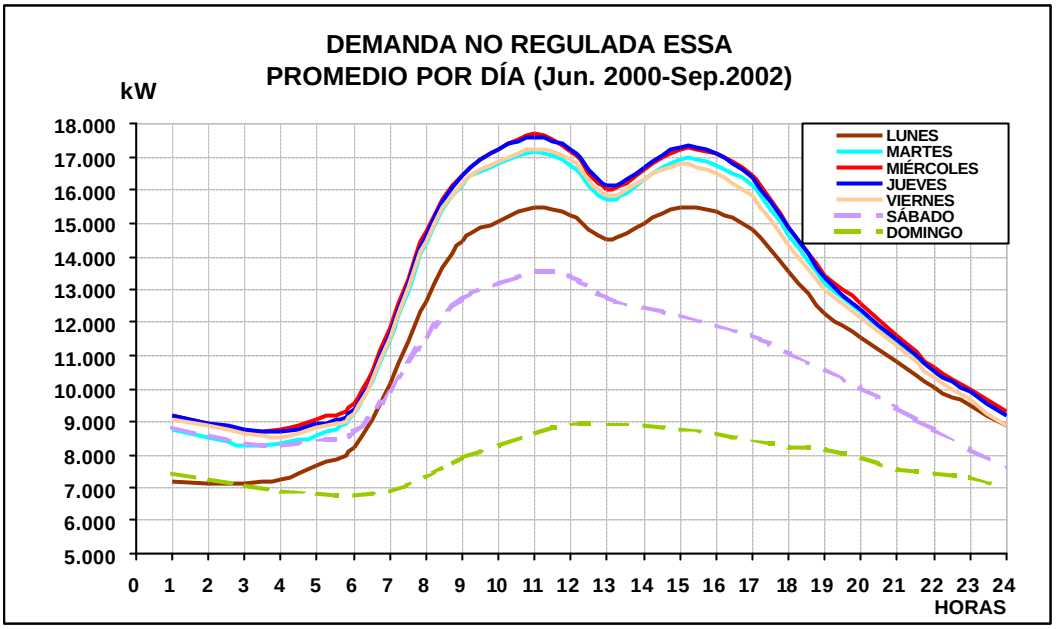


Figura 17. Diferenciación diaria de los datos de demanda no-regulada



Como se observa en las anteriores gráficas, existen 4 tipos de comportamiento:

- Días normales de trabajo: Lunes, Martes, Miércoles, Jueves y Viernes.
- Sábados
- Domingos
- Festivos (incluye Lunes festivos y otros días festivos entre semana)

Se separa el día Lunes de los días normales de trabajo, debido al efecto de los Lunes festivos en el calendario laboral colombiano. En la figuras 16 y 17 el día Lunes se ve cercano a los días hábiles debido a que es un promedio de toda la serie de datos (compuesta por Lunes hábiles en mayor cantidad y Lunes festivos), si se graficara solo los lunes festivos se observaría una curva mas parecida al Domingo. Esto quiere decir que el modelo de predicción planteado deberá disponer de un sistema que permita calcular las fechas de los datos y sus

festividades, de tal manera que en el modelo se tenga en cuenta la diferenciación de los cuatro tipos de días.

3.3 PATRONES DE LOS DATOS EMPLEADOS

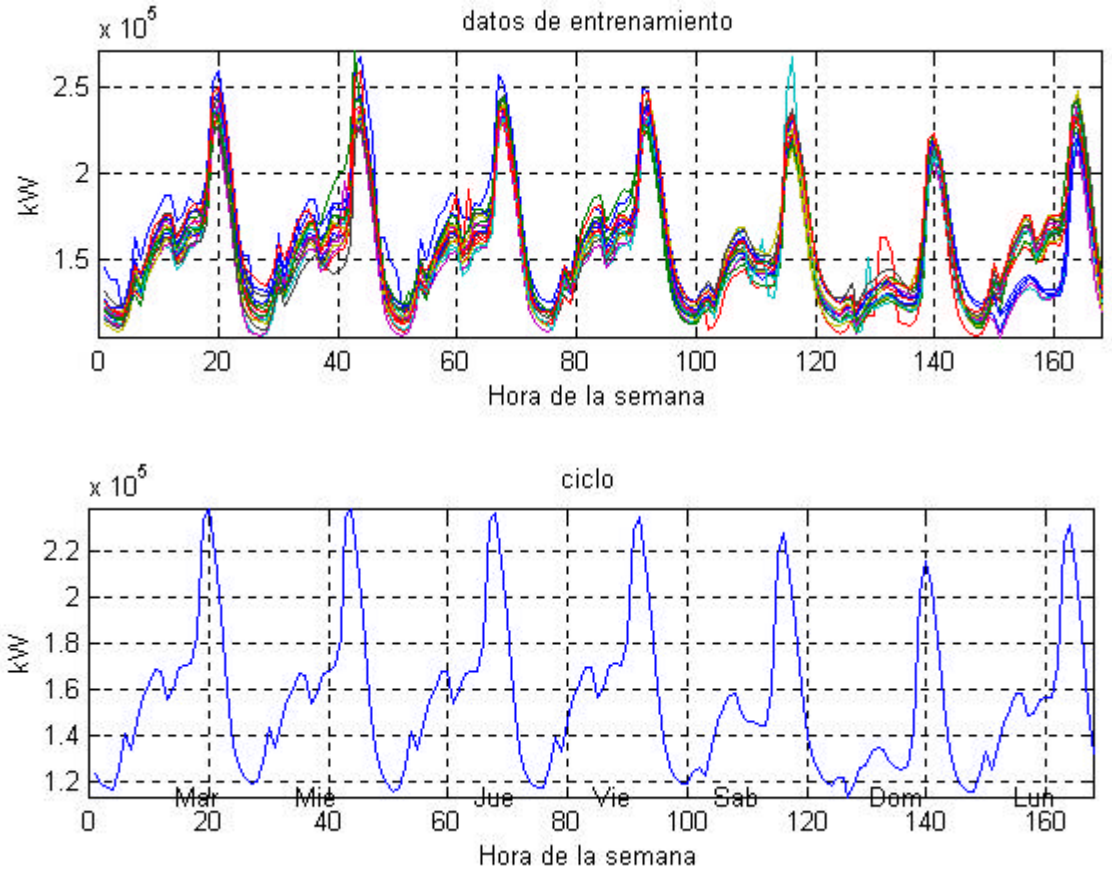
Con el objeto de continuar con el empleo del paquete estadístico SPSS, para realizar el análisis previo de las series de tiempo estudiada (Demanda horaria de potencia en usuarios regulados y no-regulados de la ESSA, 24 horas al día durante dos años), fue necesario emplear correlación cruzada entre la demanda de días seleccionados aleatoriamente, ya que el paquete estadístico, no es muy flexible para el análisis de series de tiempo, más aún con la gran cantidad de datos por analizar.

En el caso de las series de tiempo que se trabajaron en la investigación propuesta, corresponden a registros de la demanda de la Electrificadora de Santander (ESSA) hora a hora de 1.997, 1.998 y parte de 1.999 disponible (para realizar comparaciones iniciales de los modelos con trabajos anteriores); además de las demandas de la ESSA, desde Junio de 2000 hasta Septiembre de 2002 (pruebas de producción del modelo).

A continuación se presenta un ejemplo de la descomposición de la serie tiempo correspondiente a la curva de demanda de potencia horaria, de los usuarios regulados de la Electrificadora de Santander S.A, durante 4 meses (mayo 2002 a sept2002)

En la figura 18 se observa en la parte superior, los datos de la demanda horaria de cada día superpuestos organizados por semana y en la parte inferior el ciclo de la serie por semana. La organización de los datos por semana obedece a las necesidades de predicción planteadas para el Sector Eléctrico Colombiano.

Figura 18. Serie de tiempo ejemplo y ciclo de la serie por semanas



En las figuras 19 y 20 se observan la tendencia de la serie de datos obtenida mediante un análisis de regresión logarítmica y la irregularidad respectivamente. Es de anotar que el tipo de análisis de regresión empleado, depende de aquel que minimice los residuales entre la curva de valores reales y su curva de ajuste; en este caso, la tendencia es muy cercana a 1 y poco creciente debido a que no existiría un crecimiento notable de la serie de tiempo durante solo 4 meses. También es importante aclarar que al emplear el modelo multiplicativo, las unidades y magnitudes del modelo (en este caso kW) quedan en la componente ciclica, por tanto la tendencia e irregularidad son adimensionales.

Figura 19. Tendencia de la serie ejemplo

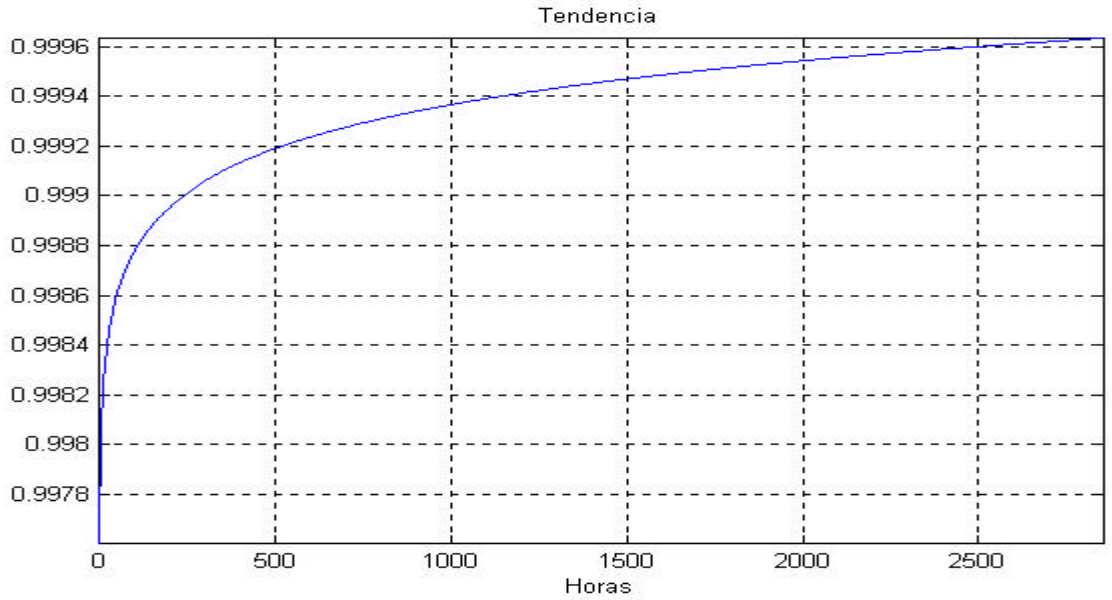
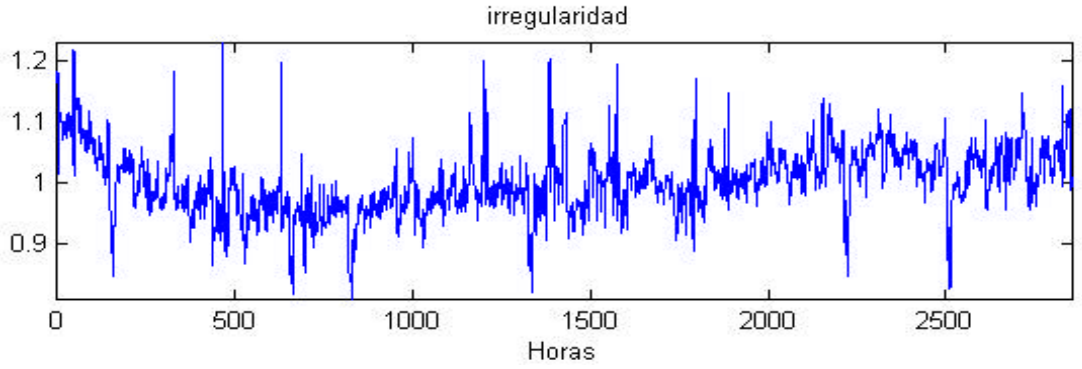


Figura 20. Irregularidad de la serie ejemplo



4. EVALUACIÓN DE LAS PRINCIPALES TÉCNICAS DE PREDICCIÓN

Una ampliación de la heurística en el tema de predicción de demanda con técnicas de predicción convencional, permite determinar ventajas del desempeño individual de estas técnicas frente a las redes neuronales artificiales, de manera tal que puedan ser aprovechadas en el planteamiento de modelos de predicción.

Debido a la amplitud de posibilidades, las técnicas de predicción convencional que se tratarán en el trabajo de investigación, han sido seleccionadas con base en su facilidad de implementación en herramientas computacionales disponibles para el trabajo como MATLAB [Nakamura-1997], y el paquete estadístico SPSS, además de su aplicabilidad como modelos de predicción en el horizonte de corto plazo.

Una revisión de la bibliografía en el tema, permite proponer el siguiente grupo de técnicas de predicción de demanda en el corto plazo, como las más comúnmente empleadas y de mejores resultados obtenidos [Gross-1987], [Moghram-1989]: Modelo autorregresivo mejorado (ARX), Modelo Autorregresivo de Promedio Movil (ARMA) que evoluciona en la metodología Box-Jenkins, y espacio de estados.

El análisis de regresión múltiple y el suavizado exponencial, mas que ser empleados como técnica de predicción permiten realizar el tratamiento previo de los datos de entrada al modelo. En el caso de la regresión múltiple su aplicabilidad se centra principalmente en el establecimiento de las variables explicativas de la serie de tiempo apoyados en un análisis de correlación. En el caso del suavizado exponencial, este permite eliminar el ruido de la serie. No obstante, la realización de un amplio número de pruebas se sale de los objetivos de este trabajo de investigación, ya que su finalidad principal, es la aplicación de

redes neuronales al problema de predicción de demanda de corto plazo.

Las pruebas que permitan reconocer ventajas o desventajas de cada una de estas técnicas con respecto a las redes neuronales, debe hacerse de manera tal que la comparación sea lo más objetiva posible. A fin de establecer un análisis comparativo de las técnicas de predicción mencionadas anteriormente con relación a las redes neuronales, se ejecutaron pruebas de predicción con pocos datos y condiciones semejantes de trabajo en cada uno de los modelos presentados en el numeral 2.2. Entre los objetivos de las pruebas realizadas se tienen lo siguientes:

- Probar la hipótesis de descomposición de la serie de tiempo, es decir, que se obtienen mejores resultados al predecir solo la irregularidad para luego volver a multiplicarla con la tendencia y ciclo (series fácilmente extrapolables, una vez se ha hecho un análisis estadístico previo de los datos).

- Examinar el efecto en los errores de predicción cuando los datos de entrada son suavizados exponencialmente.

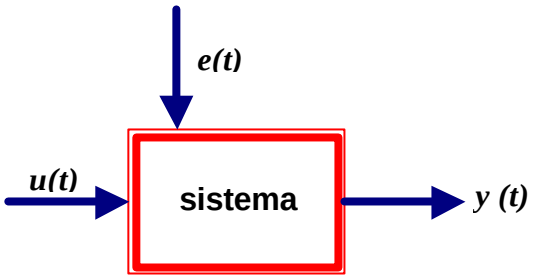
- Establecer el efecto de los días festivos en los errores de predicción, y mostrar la conveniencia de emplear un modelo que involucre variables de valor discreto para los tipos de días de la semana.

Para la realización de las pruebas se desarrollaron archivos sencillos en MATLAB, aprovechando la Toolbox de identificación de sistemas referenciada en [Ljung-1997] y la Toolbox de Redes Neuronales referenciada en [Demuth-1998] (ver anexo D). Los resultados obtenidos se clasificaron de acuerdo a los objetivos de la prueba mencionados anteriormente. Debido a que la herramienta computacional empleada para realizar las pruebas es MATLAB, se mostrará a continuación algunos aspectos importantes por destacar de los modelos que este trae en la toolbox de identificación de sistemas, en cuanto a la forma de los modelos empleados.

4.1 MODELOS DE IDENTIFICACIÓN DE SISTEMAS EN MATLAB

El problema de identificación de sistemas consiste en determinar una función que relacione las variables de entrada con las variables de salida, esta identificación tradicionalmente se hace a través de técnicas de regresión y con un conocimiento previo de la naturaleza del sistema. Es común representar el problema de identificación como un sistema dinámico en donde una señal entrada $u(t)$ afectada por una señal de ruido $e(t)$ produce una señal de salida $y(t)$, como se muestra en la figura 21, siendo el objeto del análisis la forma como estos pueden ser relacionados.

Figura 21. Esquema de identificación de sistemas



Esta relación tiene la forma paramétrica general:

$$y(t) = G(q)\,u(t) + H(q)\,e(t) \qquad \text{Ec. 32}$$

Donde $G(q)$ es una función de transferencia asociada al comportamiento entrada-salida del sistema dados q retardos en el tiempo y $H(q)$ es la transferencia asociada al ruido que ingresa al sistema.

En el caso del modelo autorregresivo mejorado (ARX en MATLAB) las relaciones paramétricas correspondientes son:

$$G(q) = q^{-nk}\frac{B(q)}{A(q)}; \qquad H(q) = \frac{1}{A(q)} \qquad \text{Ec. 33}$$

Donde ***B*** y ***A*** son polinomios para operador de retado en el tiempo q^{-1} dados por:

$$\begin{aligned} A(q) &= 1 + a_1 q^{-1} + ... + a_{na} q^{-na} \\ B(q) &= b_1 + b_2 q^{-1} + ... + b_{nb} q^{-nb+1} \end{aligned} \qquad \text{Ec. 34}$$

Para el trabajo de identificación de sistemas en MATLAB en el caso del modelo ARX, el problema se resume en encontrar los números ***na***, ***nb*** los cuales representan los ordenes de los polinomios y el número ***nk*** que representa el retardo entre la entrada y la salida. Esto plantea una expresión general del modelo muy similar a la mostrada en el numeral 2.2 como se muestra a continuación:

$$\boldsymbol{A(q)\, y(t) = B(q)\, u(t\text{-}nk) + e(t)} \qquad \text{Ec. 35}$$

O de una manera más explicita como se muestra en la ecuación 36:

$$\begin{aligned} &y(t) + a_1 y(t-1) + ... + a_{na} y(t-na) = \\ &b_1 u(t-nk) + b_2 u(t-nk-1) + ... + b_{nb} u(t-nk-nb+1) + e(t) \end{aligned} \qquad \text{Ec. 36}$$

Una estructura similar aplicada al modelo autorregresivo de promedio móvil (ARMAX en MATLAB) se puede expresar mediante la ecuación 37:

$$\boldsymbol{A(q)\, y(t) = B(q)\, u(t\text{-}nk) + C(q)\, e(t)} \qquad \text{Ec. 37}$$

Donde ***A(q)*** y ***B(q)*** son los polinomios asociados a la entrada y salida del modelo, tal como se expresaron en las ecuaciones 34 y 35, y ***C(q)*** es el polinomio asociado al ruido, dado por:

$$C(q) = 1 + c_1 q^{-1} + ... + c_{nc} q^{-nc} \qquad \text{Ec. 38}$$

Entonces el problema de identificación con el modelo ARMAX en MATLAB

consiste en establecer los números ***na***,***nb***,***nc*** y ***nk.***

En el caso del modelo de Box-Jenkins, su estructura esta dada por la siguiente ecuación, en donde aparecen dos nuevos polinomios ***F(q)*** y ***D(q)***:

$$y(t) = \frac{B(q)}{F(q)} u(t - nk) + \frac{C(q)}{D(q)} e(t) \quad \text{Ec. 39}$$

Donde:

$$F(q) = 1 + f_1 q^{-1} + ... + f_{nf} q^{-nf}$$
$$D(q) = 1 + d_1 q^{-1} + ... + d_{nd} q^{-nd} \quad \text{Ec. 40}$$

Lo anterior implica que en el caso de Box-Jenkins es necesario determinar los números ***nb***,***nc***,***nd***,***nf*** y ***nk*** que representan los ordenes de los respectivos polinomios.

Para el caso de los modelos de espacio de estados la representación del problema mediante las funciones de transferencia ***G(q)*** y ***H(q)*** como se presenta en la ecuación 32, están relacionadas con la forma del modelo en sus variables de estado:

$$x(t+1) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t) + Ke(t) \quad \text{Ec. 41}$$

La manera como se parametriza el modelo a fin de resolver en MATLAB es utilizando ***G(q)*** y ***H(q)*** como se muestran en la ecuación 42.

$$G(q) = C(qI_{nx} - A)^{-1}B + D$$
$$H(q) = C(qI_{nx} - A)^{-1}K + I_{ny} \quad \text{Ec. 42}$$

Lo anterior implica que en el caso de espacio de estados hay que seleccionar un número ***n*** que representa el orden del modelo, siendo las matrices identidad $\boldsymbol{I_{nx}}$ e $\boldsymbol{I_{ny}}$ de orden ***n*** x ***x*** o ***n*** x ***y,*** siendo ***x*** e ***y*** las dimensiones de los vectores de estado

x(t) y de salida ***y(t)*** respectivamente.

El proceso de creación de un modelo y de predicción empleando la Toolbox de Identificación de Sistemas en MATLAB, es común a todos los modelos. Una vez se han procesado los datos de entrada y salida, se procede a establecer la estructura de los polinomios que conforman los modelos, en primera instancia, seleccionando los ordenes de los polinomios (***na***,***nb***,***nc***,***nd***,***nf*** y ***nk***) según el tipo de modelo; para esto se puede emplear el comando "selstruc". Posteriormente se establecen los polinomios del modelo A,B,C,D,F los cuales se presentan en el formato "Theta" y son calculados mediante el comando "th".

Una vez se hab establecido los polinomios del modelo se procede a realizar las comprobaciones pertinentes con los comandos "sim" y "predict", los cuales varían según el tipo de modelo y de la estructura del formato "th".

4.2 PRUEBAS EFECTUADAS A LOS MODELOS

Como se mencionó al comienzo de este capítulo, uno de los objetivos principales de estas pruebas son la comparación de los esquemas tradicionales de predicción con respecto a las redes neuronales; pero sin duda, el mayor aporte que ofrecen estas pruebas de las técnicas convencionales, radica en la posibilidad de comprobar las hipótesis que sustentan el trabajo de investigación, es decir, la descomposición de la serie permite reducir los errores de predicción y el suavizado exponencial facilita la operación del esquema de predicción.

La primera prueba realizada se efectuó con pocos datos sin tener en cuenta el efecto de los festivos (para lo cual solo se ingresó semanas de Lunes a Viernes). Se intentó identificar el sistema simulando el día 14 de Marzo de 1.997 a partir de

siete semanas anteriores disponibles para armar los modelos[13].

La metodología dispuesta para tal fin fué:

- Se empleó para la comparación el porcentaje medio de error (MPE), error medio cuadrado (MSE), máximo de error, mínimo de error, y desviación estándar del error.

- Se trabajó la predicción a partir de seis semanas anteriores y omitiendo datos de sábados domingos y festivos (estos últimos tienen problema por su tratamiento mediante variables ficticias, en algunas técnicas).

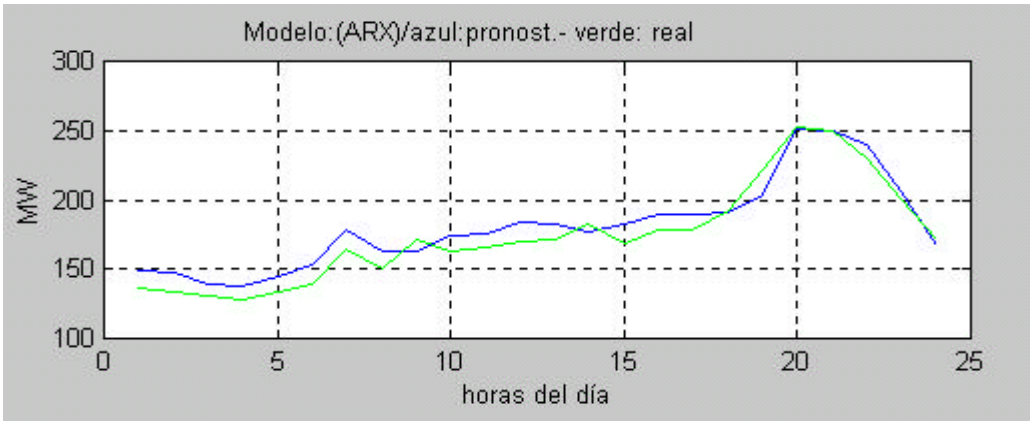
- Los datos de prueba empleados en entrada fueron ingresados tal cual se presentó la serie en los días: 20-21-22-23-24 Ene97, 27-28-29-30-31 Ene97, 3-4-5-6-7 Feb97, 10-11-12-13-14 Feb97, 17-18-19-20-21 Feb97, 24-25-26-27-28 Feb97, 3-4-5-6-7 Mar97

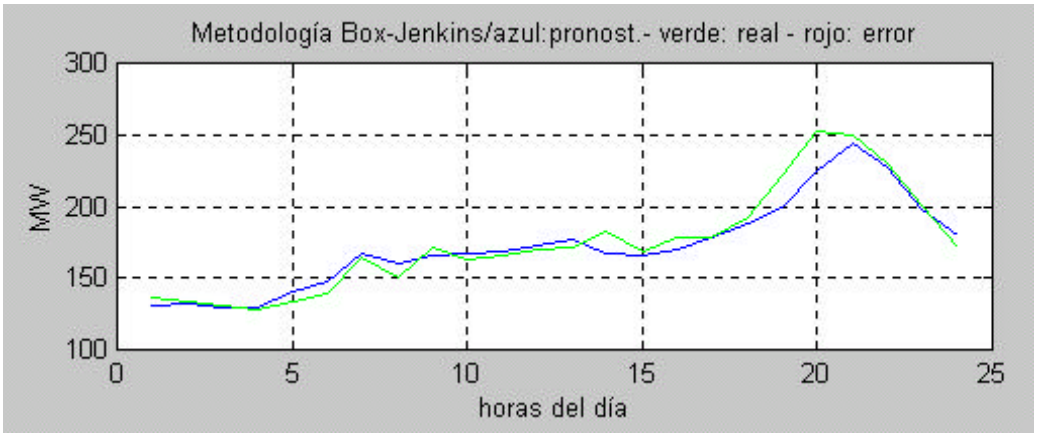
- Los datos a la salida fueron : 10-11-12-13-14 Mar97

En las siguientes figuras se muestran los resultados obtenidos para los modelos ARX, Box-Jenkins y Espacio de Estados. Posteriormente en la tabla 1, se muestran los comparativos de los resultados obtenidos en las tres primeras pruebas.

[13] Es conveniente aclarar que en esta prueba y las dos siguientes, no se realizó predicción, sino simplemente una simulación de datos, es decir se buscó que el modelo relacionara entradas con salidas.

Figura 22. Resultados Prueba 1 - Ingresando serie sin descomponer





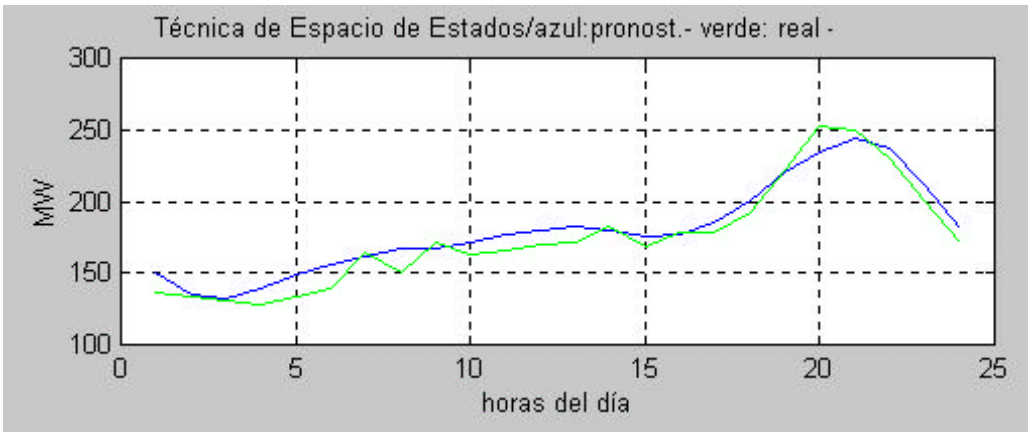


Figura 23. Resultados Prueba 1 - Ingresando irregularidad de la serie

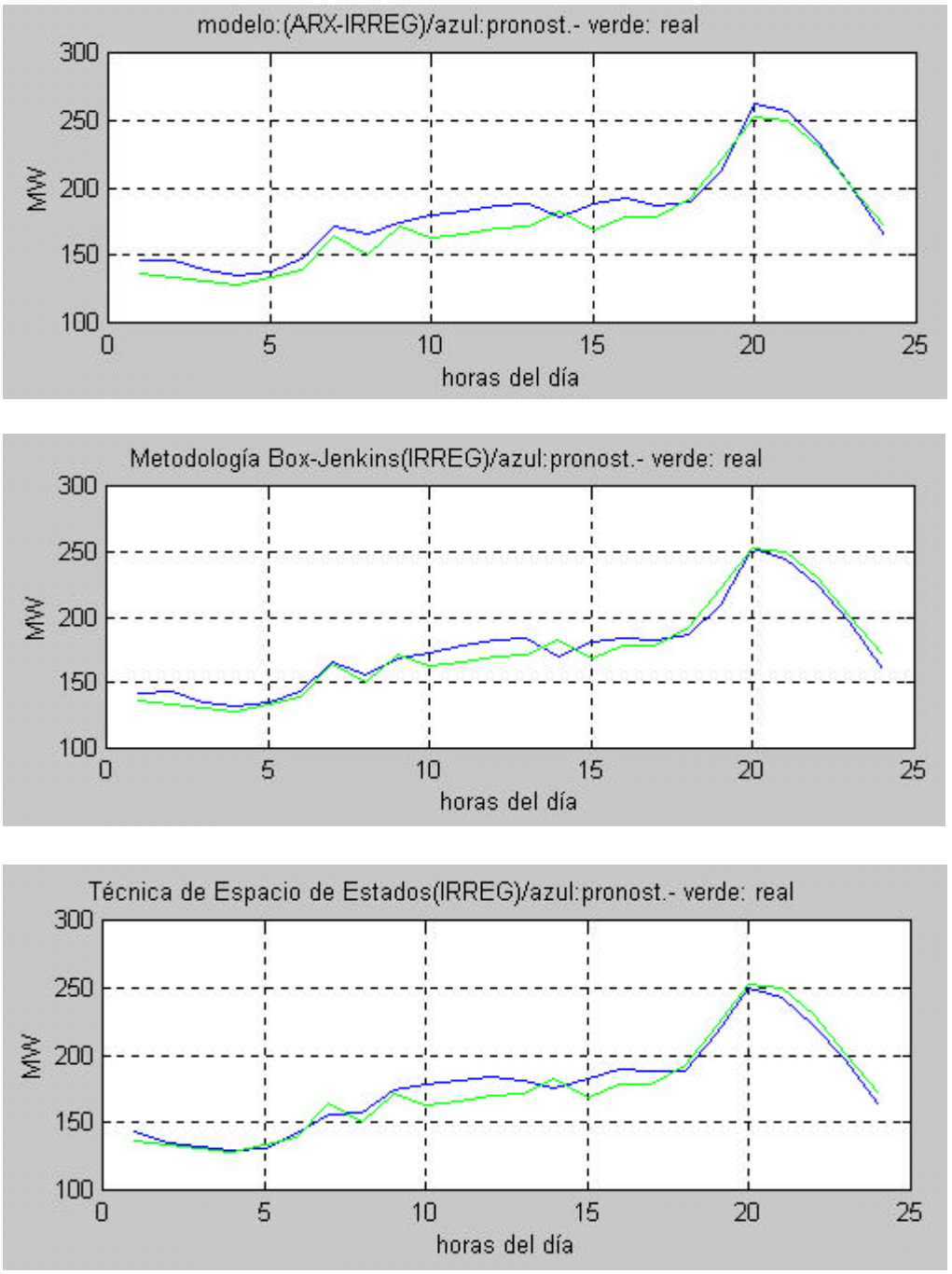
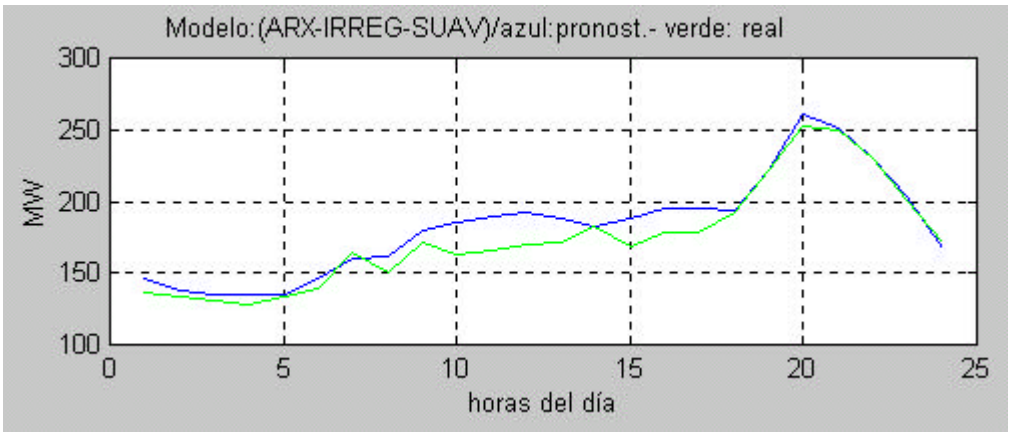
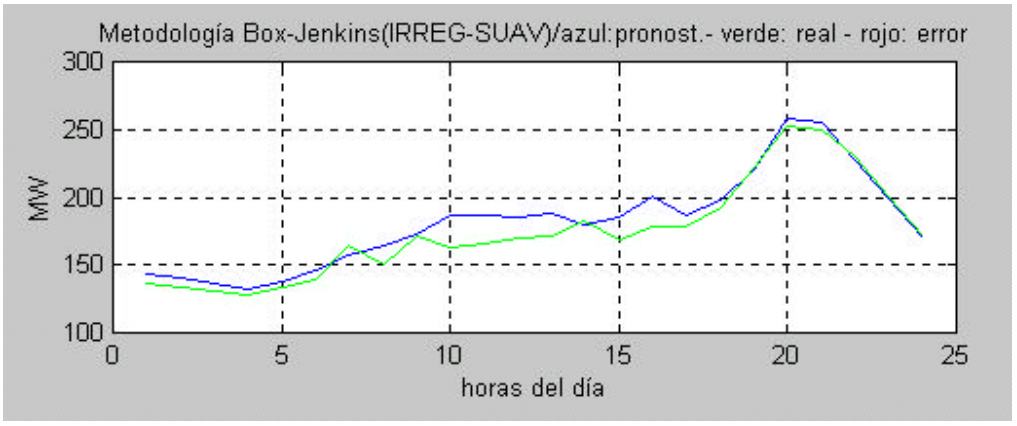
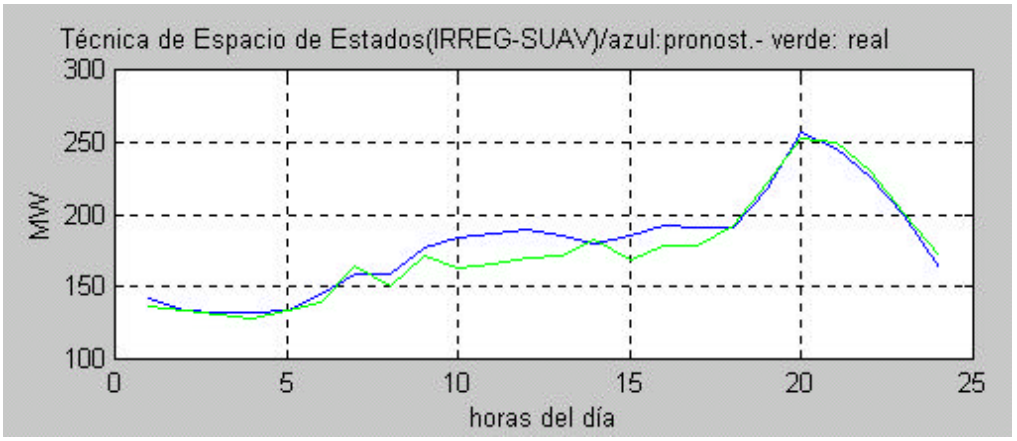


Figura 24. Resultados Prueba 1 - Ingresando irregularidad suavizada







De la primera prueba se puede observar en la tabla 1 el mejoramiento sostenido en los tres modelos (ARX, Box-Jenkins y Espacio de Estados) cuando en vez de ingresar la serie de tiempo completa, se ingresa la irregularidad de la serie. No obstante, se observa desmejoramiento en los errores al ingresar la irregularidad suavizada al mismo modelo.

Tabla 1. Resultados consolidados de la prueba 1

Modelo	ARX		
Datos de Entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MPE (%)	-6,4568	-7,4253	-8,242
MSE (kW^2)	114,5857	112,9355	134,6616
error máximo	17,0802	7,8219	4,386
error mínimo	-14,5058	-18,6214	-23,3219
Desv. Estándar del error	8,7215	7,7661	8,3366

Modelo	BOX-JENKINS		
Datos de Entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MPE (%)	2,3699	-1,8986	-7,3199
MSE (kW^2)	90,0033	65,0849	121,4104
error máximo	28,2127	12,5156	7,0797
error mínimo	-10,0112	-13,004	-23,8208
Desv. Estándar del error	9,3838	8,0096	8,413

Modelo	ESPACIO DE ESTADOS		
Datos de Entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MPE (%)	-5,3536	-2,139	-4,8662
MSE (kW^2)	99,5268	65,2106	98,5055
error máximo	18,992	8,4661	8,7192
error mínimo	-17,3018	-14,8294	-20,5098
Desv. Estándar del error	8,5996	7,9542	8,8362

Al revisar trabajos anteriores realizados en la UIS que emplean las mismas series de tiempo (datos de demanda ESSA 1997-1998), los presentados hasta el momento ([Acevedo-2000] [Portela-2000] [Rodriguez-2000]) se acercan a los obtenidos mediante redes neuronales artificiales, lo cual vislumbra posibilidades de mejoramiento en los porcentajes de error cuando esta metodología sea

aplicada a la red neuronal, y sobre todo, se puede tener un mejor acercamiento a los valores reales máximos y mínimos de la serie (principal problema encontrado en la predicción), el cual se puede observar con la disminución del error medio cuadratico (MSE).

La segunda prueba se realizó incluyendo el modelo ARMAX y una red neuronal Feed-forward de tres capas, cada una de ellas de 14 neuronas, con el fin de establecer comparativos de rendimiento respecto de una red neuronal sencilla. En este caso se atendió el requerimiento regulatorio de predicción, en el cual se debe basar el modelo en las 14 semanas anteriores.

El objetivo de esta prueba es predecir la semana del Domingo 1 de septiembre al Sábado 7 de Septiembre de 2002 a partir de las 15 anteriores (atendiendo a las disposiciones de la CREG), es decir, desde el Domingo 19 de Mayo de 2002 hasta el Sábado 31 de Agosto de 2002, incluyéndole al modelo todos los días (laborales y festivos). Para tal fin se dispuso un conjunto de datos para creación y validación de los modelos desde el Domingo 12 de Mayo de 2002 (1 semana antes de las necesarias para la predicción) a fin de establecer su comportamiento en validación. Los datos empleados corresponden en este caso a la demanda regulada de la ESSA y la metodología dispuesta fue:

- Se siguió el mismo orden de la prueba 1, es decir, se crearon los modelos con la serie completa, luego la irregularidad de la serie y posteriormente la irregularidad suavizada.

- Se empleó para la comparación el porcentaje medio de error absoluto (MAPE), la raíz del error medio cuadrado (RMSE), máximo de error, mínimo de error, y la diferencia entre energía real-predicha.

- Se realizó creación (en el caso de la red neuronal "entrenamiento"), validación y predicción de la serie corriendo los datos una semana.

En las siguientes figuras se muestran los resultados obtenidos para los modelos ARX, ARMAX, Box-Jenkins, Espacio de Estados y Red Neuronal. Posteriormente en la tabla 2, se muestran los comparativos de los resultados obtenidos en la prueba.

Figura 25. Resultados prueba 2 - Ingresando serie sin descomponer (ARX-ARMAX)

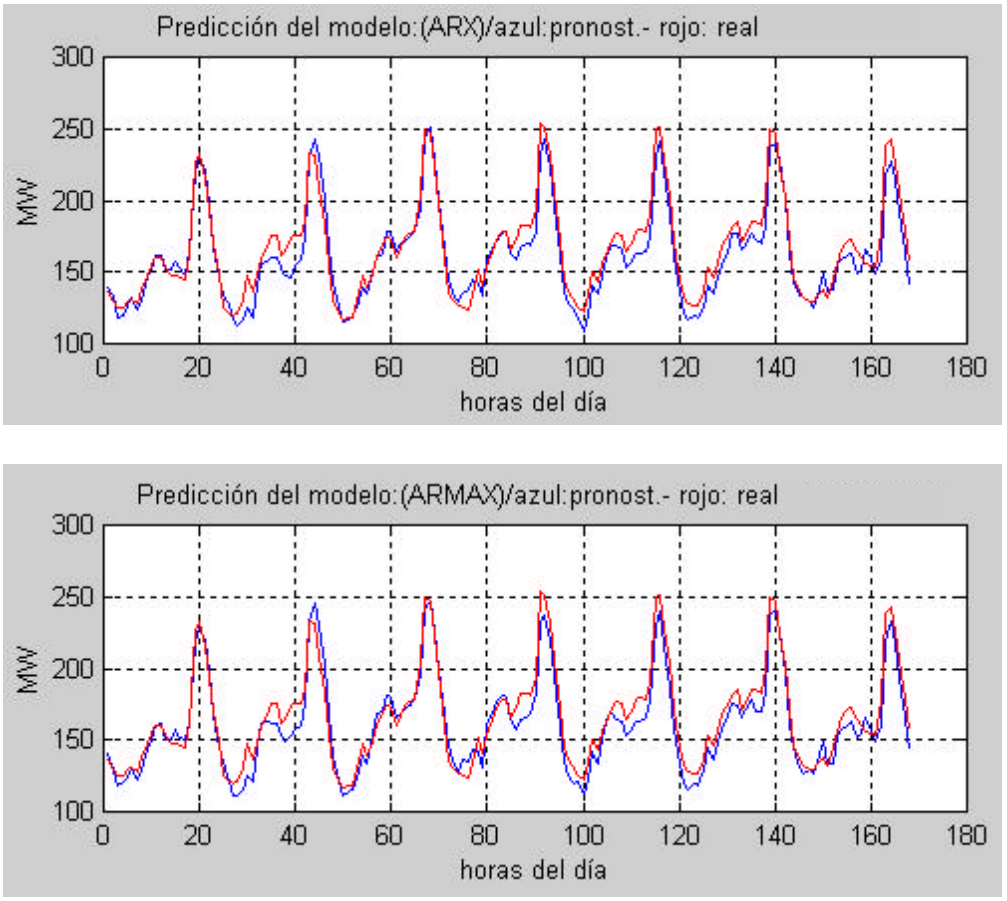


Figura 26. Resultados prueba 2 - Ingresando serie sin descomponer (Box-Jenkins - Espacio de Estados - Red Neuronal)

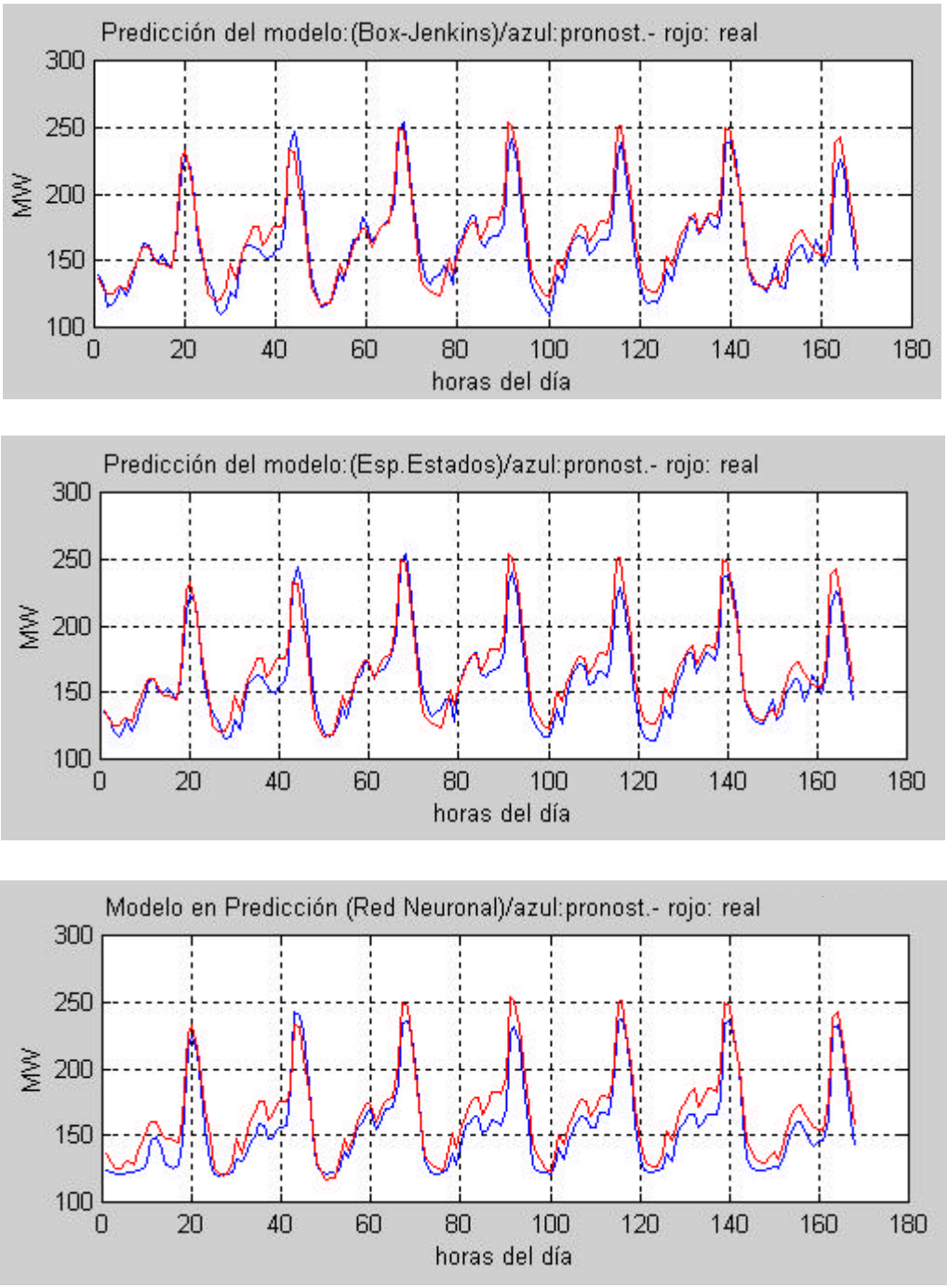


Figura 27. Resultados Prueba2 - Ingresando la irregularidad de la serie (ARX - ARMAX - Box-Jenkins)

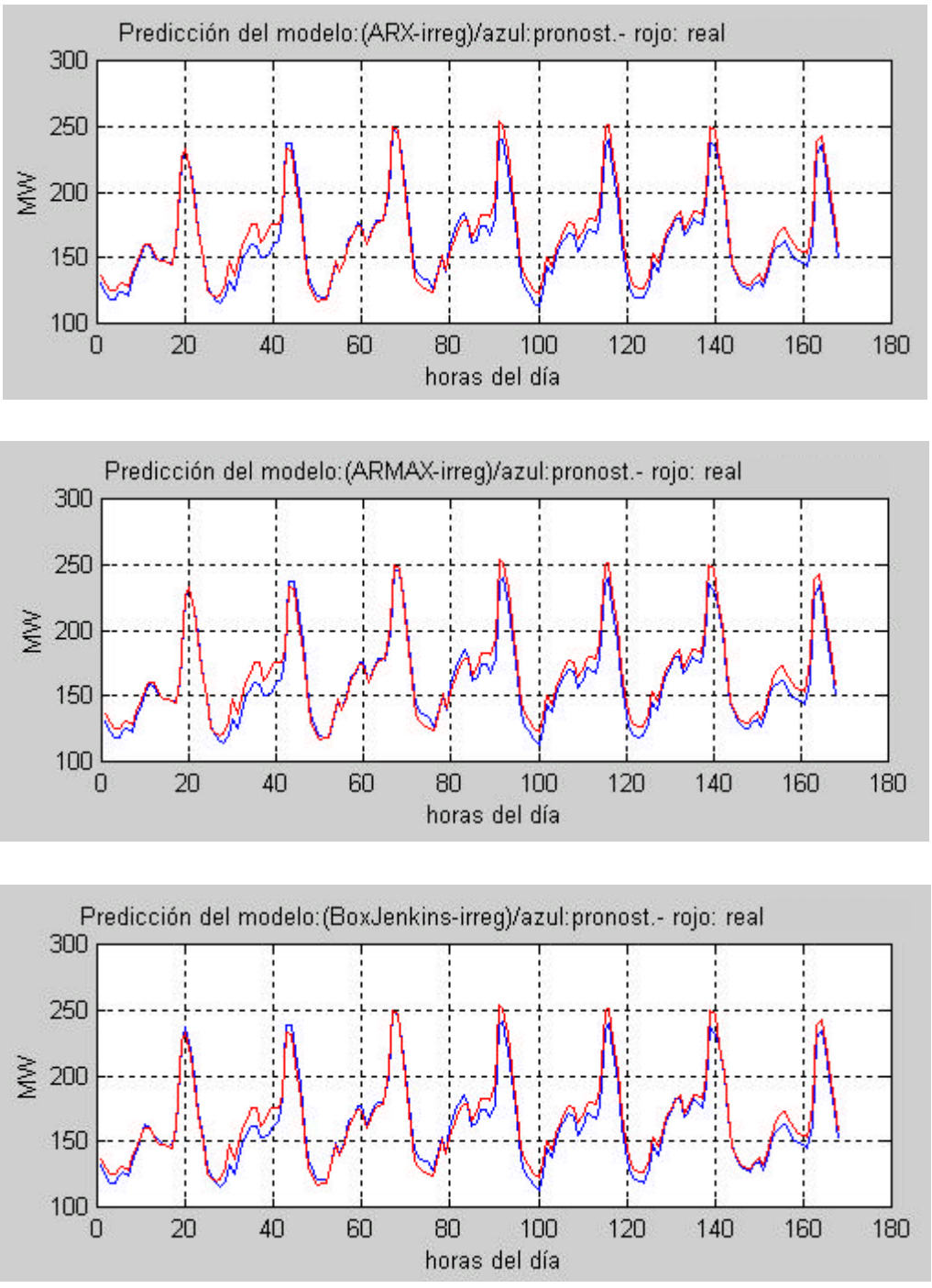
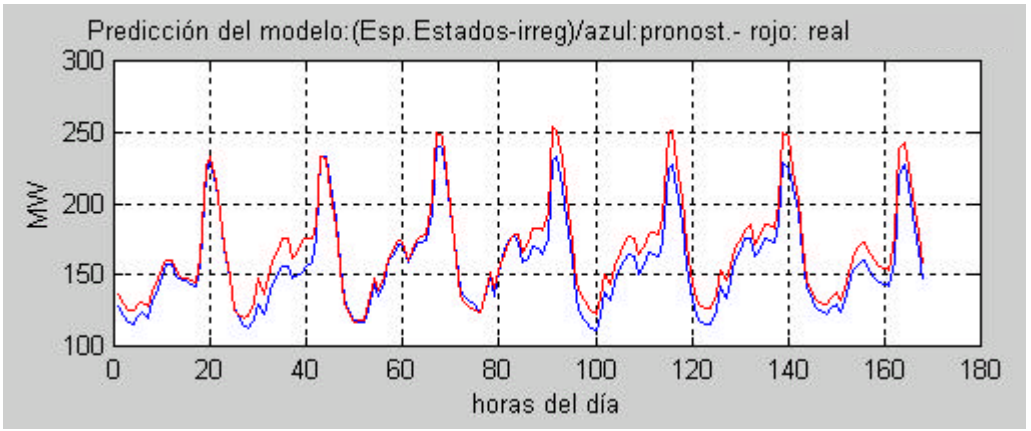


Figura 28. Resultados prueba 2 - Ingresando la irreguklaridad de la serie (Espacio de estados - Red Neuronal)



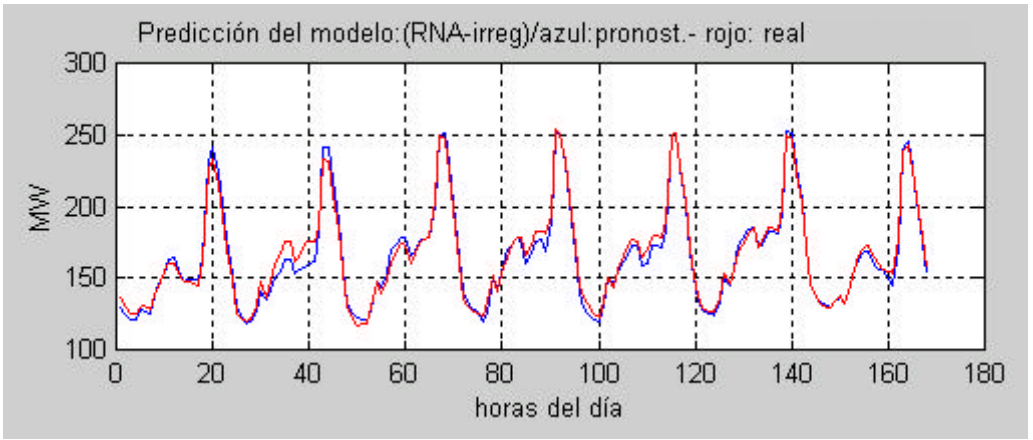


Figura 29. Resultados prueba 2 - Ingresando la irregularidad suavizada (ARX - ARMAX - Box-Jenkins)

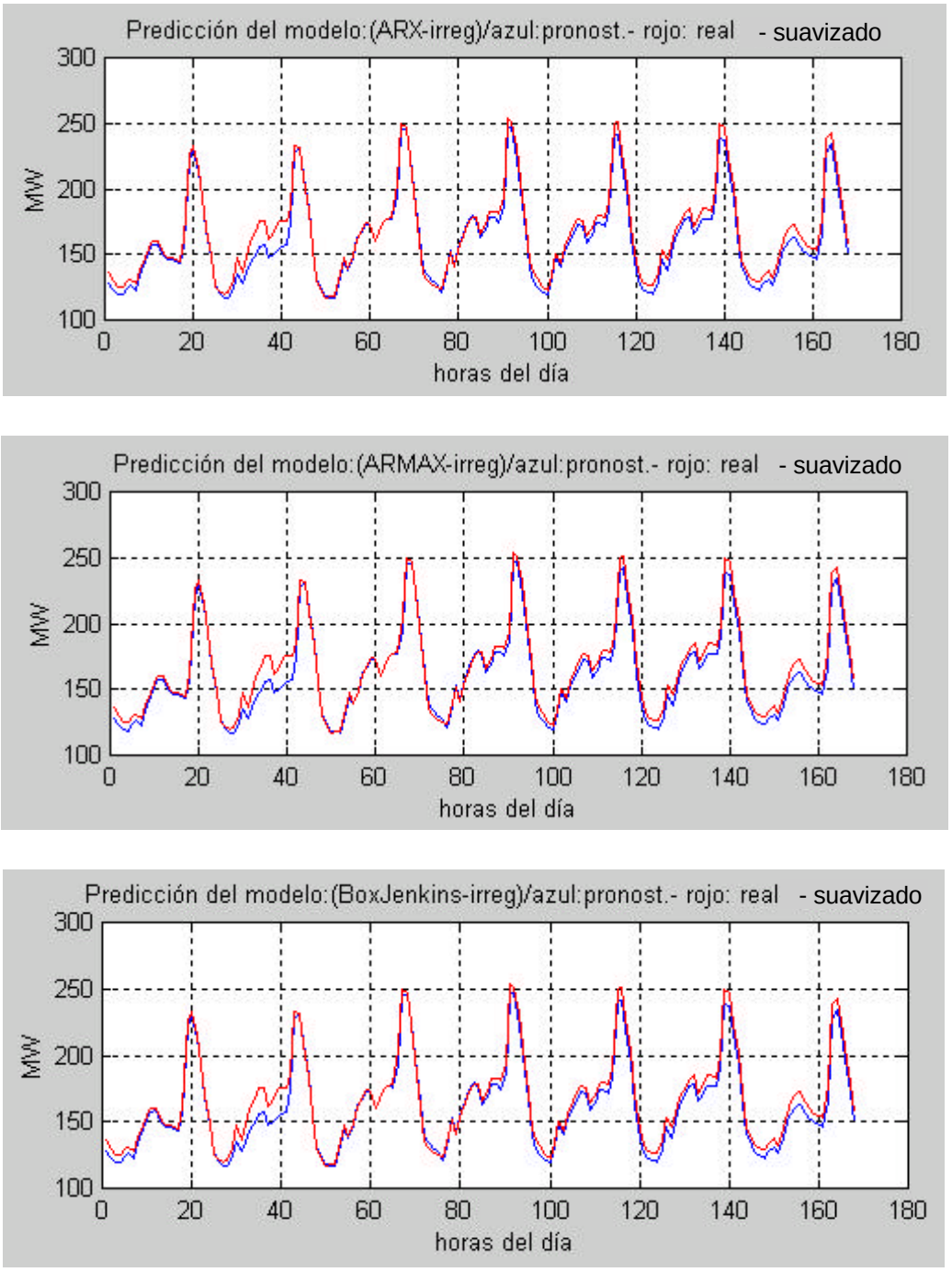
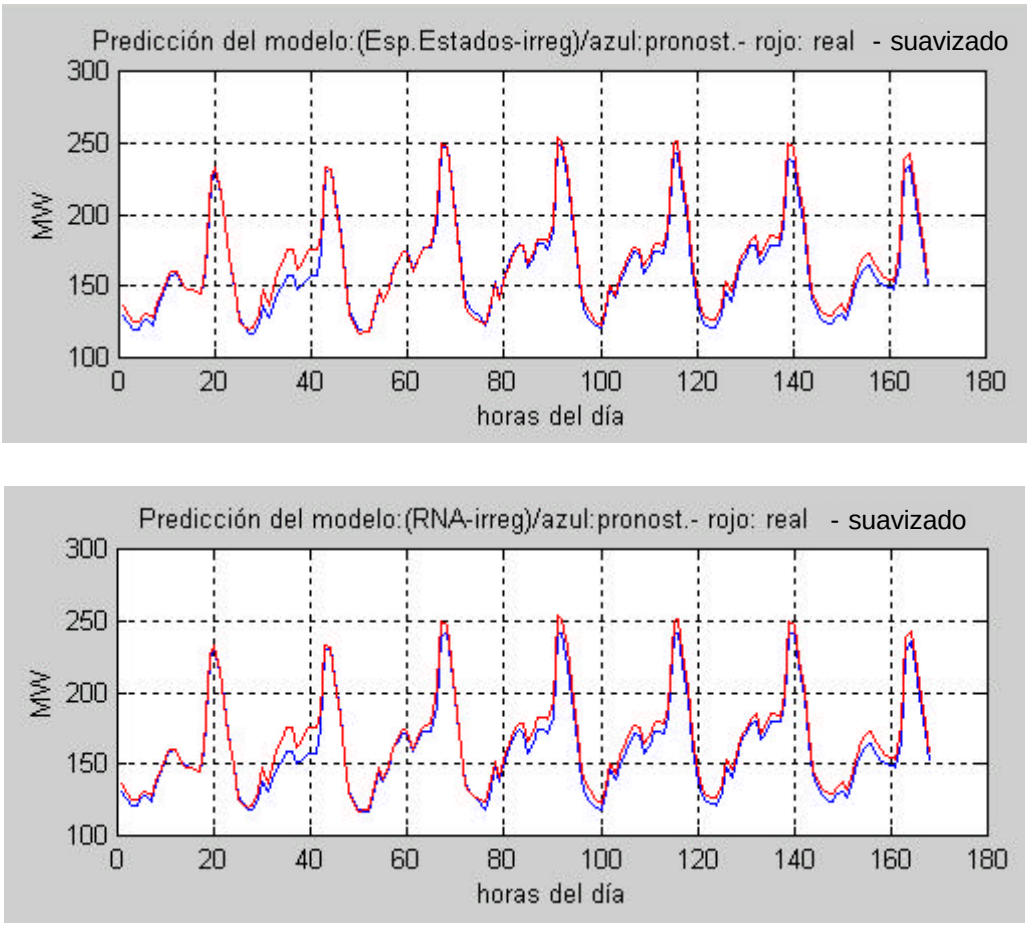


Figura 30. Resultados Prueba 2 - Ingresando la irregularidad suavizada (Espacio estados - Red Neuronal)



De la segunda prueba se reafirma el mejoramiento sostenido en todos los modelos cuando en vez de ingresar la serie de tiempo completa, se ingresa la irregularidad de la serie. Adicionalmente, aunque no se mejoran ostensiblemente los errores de predicción cuando se utiliza la irregularidad suavizada, si se construyen modelos menos complicados en cuando a ordenes menores de los polinomios empleados y en el caso de la red neuronal se requieren menores épocas de entrenamiento. En la tabla 2 se puede observar este análisis de manera más cualitativa. En capítulos posteriores se dedicará mayor interés al funcionamiento y pruebas en RNA's.

Tabla 2. Resultados consolidados de la prueba 2

Modelo	ARX		
Datos de entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MAPE validación (%)	2,123	1,187	2,735
MAPE predicción (%)	4,963	3,903	3,321
RMSE validación (MW)	4,506	2,784	5,253
RMSE predicción (MW)	9,923	7,690	6,870
Error Máximo (MW)	27,211	20,653	20,379
Error Mínimo (MW)	-21,865	-9,845	-4,403
Dif. Energía (real-predicha) (MWh)	839,353	785,040	842,689

Modelo	ARMAX		
Datos de entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MAPE validación (%)	2,100	1,205	2,733
MAPE predicción (%)	5,043	3,928	3,317
RMSE validación (MW)	4,354	2,846	5,250
RMSE predicción (MW)	9,856	7,780	6,860
Error Máximo (MW)	21,821	20,443	20,378
Error Mínimo (MW)	-20,888	-9,535	-4,424
Dif. Energía (real-predicha) (MWh)	863,401	805,026	840,965

Modelo	BOX-JENKINS		
Datos de entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MAPE validación (%)	1,980	1,304	2,733
MAPE predicción (%)	5,002	3,690	3,331
RMSE validación (MW)	4,314	2,887	5,249
RMSE predicción (MW)	10,069	7,251	6,891
Error Máximo (MW)	28,282	18,391	20,380
Error Mínimo (MW)	-22,762	-10,239	-4,405
Dif. Energía (real-predicha) (MWh)	729,320	603,596	846,788

Modelo	ESPACIO DE ESTADOS		
Datos de entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MAPE validación (%)	2,317	1,772	2,739
MAPE predicción (%)	5,272	5,385	3,037
RMSE validación (MW)	4,951	3,976	5,265
RMSE predicción (MW)	10,509	10,781	6,385
Error Máximo (MW)	29,178	25,600	19,753
Error Mínimo (MW)	-23,895	-4,800	-5,212
Dif. Energía (real-predicha) (MWh)	947,239	1438,000	720,339

Modelo	RED NEURONAL		
Datos de entrada	Serie Completa	Irregularidad	Irregularidad Suavizada
MAPE validación (%)	5,277	4,856	2,928
MAPE predicción (%)	7,008	2,347	3,223
RMSE validación (MW)	10,333	8,962	5,687
RMSE predicción (MW)	13,457	4,953	6,651
Error Máximo (MW)	30,300	15,940	18,705
Error Mínimo (MW)	-11,010	-10,055	-3,634
Dif. Energía (real-predicha) (MWh)	1708,400	161,785	876,341
Épocas	1800	2000	744
Capas	3	3	3
Neuronas por capa	14	14	14

5. REDES NEURONALES EN LA PREDICCIÓN DE LA DEMANDA ELÉCTRICA

Las redes neuronales artificiales (RNA), o simplemente redes neuronales como se les llama a menudo, se refieren a una clase de modelos inspirada por los sistemas nerviosos biológicos. Estos modelos están compuestos de muchos elementos de información, denotados normalmente como neuronas, trabajando en paralelo y conectadas a través pesos sinápticos que se permiten adaptar a través de un proceso de aprendizaje. Se puede interpretar que una red neuronal artificial es una máquina adaptable que puede guardar el conocimiento a través del proceso de aprendizaje.

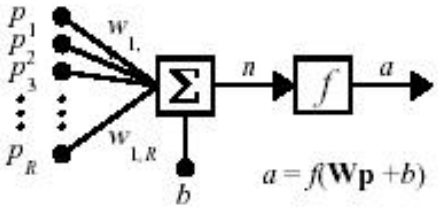
La investigación en el campo de las redes neuronales tiene ya una historia de algunas décadas, iniciando en 1943 con Warren McCulloch (neurofisiólogo) y Walter Pitts (matemático), quienes lanzaron una teoría acerca de la forma de trabajar de las neuronas biológicas, modelando el comportamiento de una neurona mediante circuitos eléctricos. No obstante, el auge investigativo que se dio posteriormente al tema, fue frenado en 1969, cuando Marvin Minsky y Seymour Papert del MIT, publicaron el documento "Perceptrons", en donde se afirmaba la "esterilidad" del perceptron[14] para resolver algunos problemas interesantes (como el caso de la función XOR), lo cual orientó las investigaciones hacia otras formas de inteligencia artificial. A pesar de esto, algunos investigadores continuaron el estudio consiguiendo que en para 1980, el interés creciera posteriormente en forma acelerada. Hoy, las redes neuronales tienen muchas aplicaciones, de las cuales se destacan el reconocimiento de patrones, el control de procesos y la predicción de serie temporales.

[14] Es de anotar que en aquella epoca el perceptron original solo poseia una capa, hoy en día es multicapa.

5.1 CONCEPTOS BÁSICOS DE LAS RNA

Podría definirse a una red neuronal artificial como una estructura de proceso de información, paralela y distribuida, formada por elementos de procesamiento (EP) o neuronas, interconectados a través de canales unidireccionales llamados conexiones sinapticas. Cada EP tiene una salida única que se distribuye sobre un número de conexiones bilaterales. El proceso de información interno en cada EP puede definirse de forma arbitraria, pero ha de ser totalmente local, es decir, depende solamente de los valores almacenados en su memoria local. En la figura 32 se muestra una representación esquemática del modelo matemático de una neurona[15].

Figura 31. Modelo esquemático de una neurona artificial



Cada neurona puede poseer muchas entradas provenientes de otras unidades o de las entradas del sistema, pero una única salida, la cual se puede conectar a otras neuronas o ser una salida del sistema.

Del modelo anterior hay que tener en cuenta dos aspectos claves. En primer lugar los arreglos matriciales que se dan en la unidad de procesamiento y en segundo lugar la naturaleza no-lineal de la salida de la neurona, al ser afectada por la función de activación f .

[15] Este modelo matemático, así como la denotación de sus variables, es tomada del manual de la TOOLBOX de redes neuronales de MATLAB, por ser esta herramienta la que se usará en el trabajo de investigación.

Estas unidades de procesamiento poseen las siguientes partes:

- Un conjunto de conexiones sinápticas, las cuales están caracterizadas por un peso o intensidad de conexión $\boldsymbol{w}$ que multiplica los valores de las $\boldsymbol{R}$ entradas $\boldsymbol{p}$. Estos pesos, son las únicas variables que se modifican durante el proceso de entrenamiento de la red, una vez ha sido definida su arquitectura[16].

- Un soma (o sumador), el cual se encarga de sumar todas las entradas pesadas en sus respectivas sinapsis. Esta operación es una simple combinación lineal de las variables de entrada a la neurona.

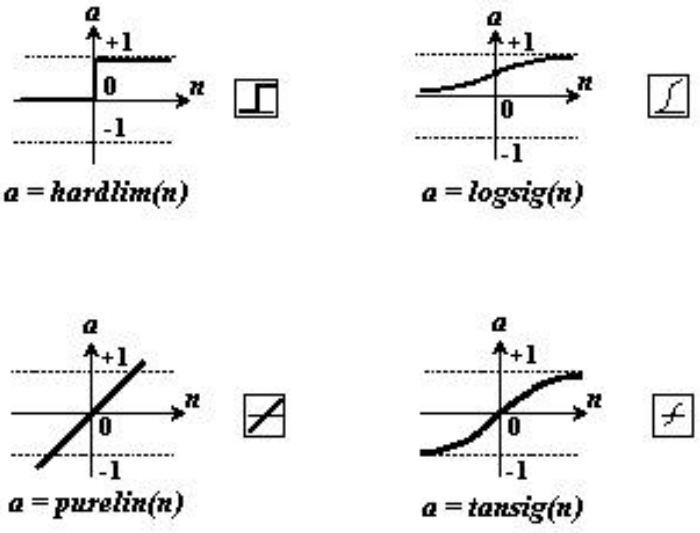
- Una función de activación para limitar la amplitud de la salida de la neurona. Esta función de activación es un parámetro que se escoge antes del entrenamiento de la red. Existen diferentes tipos de función de activación, incluso el usuario puede inventarse una función de activación acorde con sus requerimientos. En la figura 33 se muestran algunas de las comúnmente usadas.

- Un umbral exterior $\boldsymbol{b}$ (bias) que determina el valor por encima del cual la neurona se activa.

A un grupo de neuronas idénticas que están al mismo nivel de procesamiento de la información, se le conoce como "capa" de neuronas. Por lo general, una red neuronal posee una capa de entrada, una capa de salida, y $\boldsymbol{n}$ capas "ocultas" intermedias. Aunque las funciones de activación de cada neurona de la red neuronal pueden ser diferentes, se acostumbra a mantener la misma función de activación en las neuronas de una misma capa y en la mayoría de los casos emplear funciones de activación diferentes entre capas.

[16] Existe sin embargo un proceso de creación de arquitectura y entrenamiento de la red neuronal conocido como algoritmo de correlación en cascada, en el cual no sólo se modifican los pesos, sino también la topología de la red cuando se entrena.

Figura 32. Algunas funciones de activación disponibles en MATLAB



La formulación matemática presente por capa puede ser extraída del esquema de red presentado de la figura 34. En esa red, cada elemento del vector de entrada $\boldsymbol{p}$ se conecta a cada neurona a través de la matriz de pesos $\boldsymbol{W}$. La i-ésima neurona tiene un soma que recoge sus entradas pesadas y el factor de sesgo $\boldsymbol{b}$ (bias), para formar su propia salida escalar $\boldsymbol{n(i)}$. Varios $\boldsymbol{n(i)}$ forman el vector $\boldsymbol{n}$ de $\boldsymbol{S}$ elementos. Finalmente, las salidas de cada neurona forman un vector columna $\boldsymbol{a}$, el cual se calcula según la expresión mostrada en la parte inferior de la figura 34.

En cuanto a la notación empleada, es conveniente resaltar que el número de entradas a una capa $\boldsymbol{R}$, puede ser diferente del número de neuronas por capa $\boldsymbol{S}$, de tal forma que no se restringe el número de neuronas de una capa al número de entradas de la misma.

Figura 33. Representación esquemática y matricial de una capa de neuronas

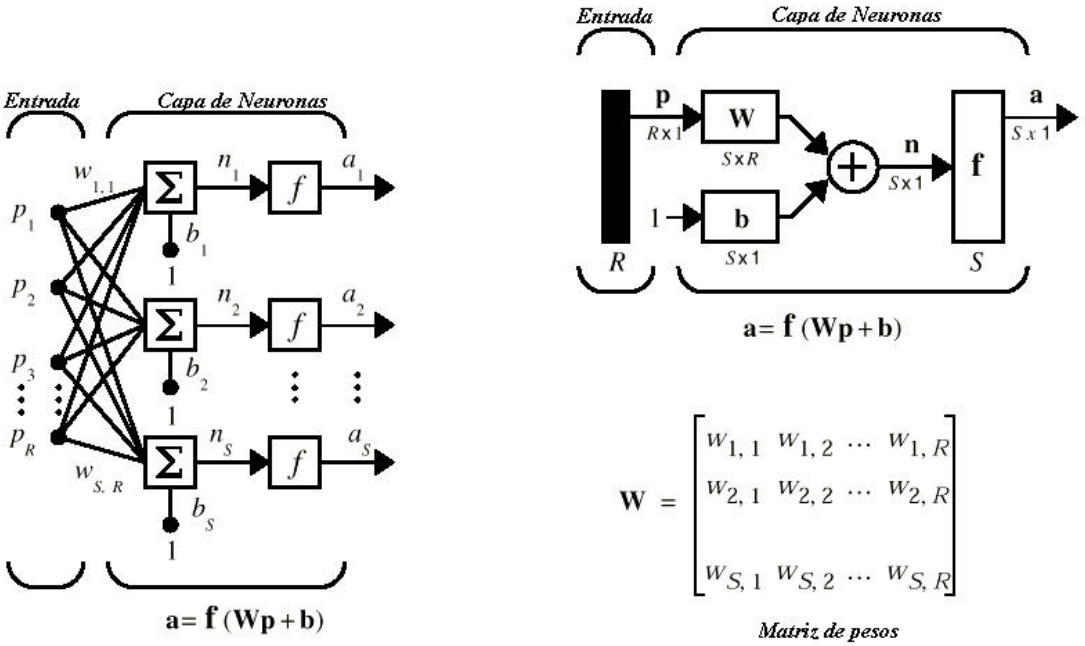
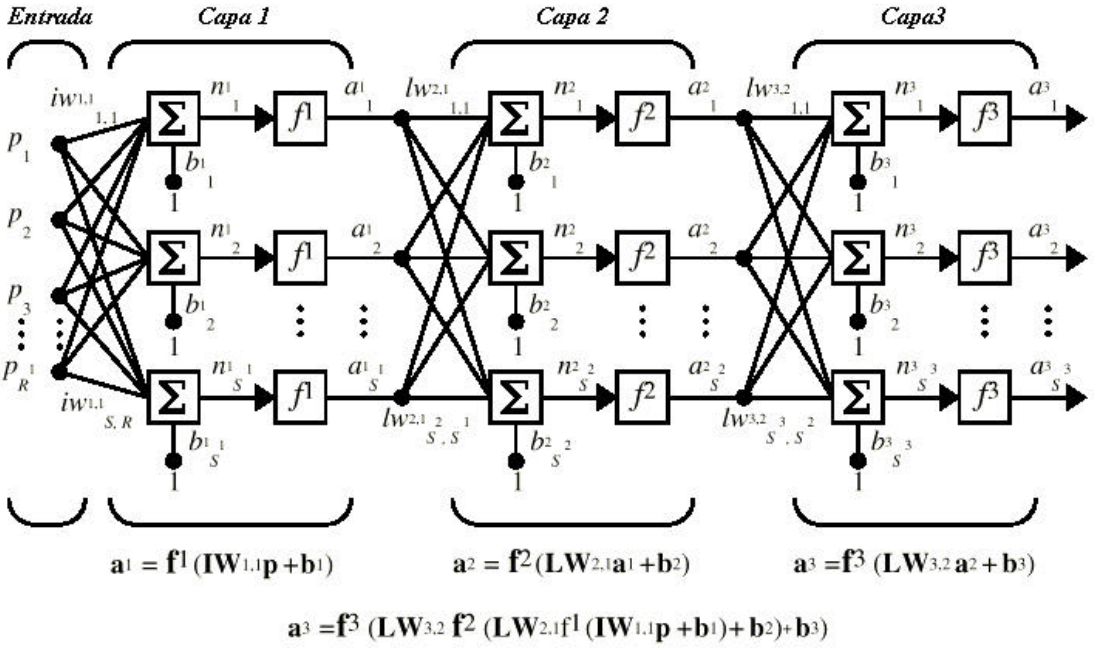


Figura 34. Ejemplo de una red neuronal de 3 capas

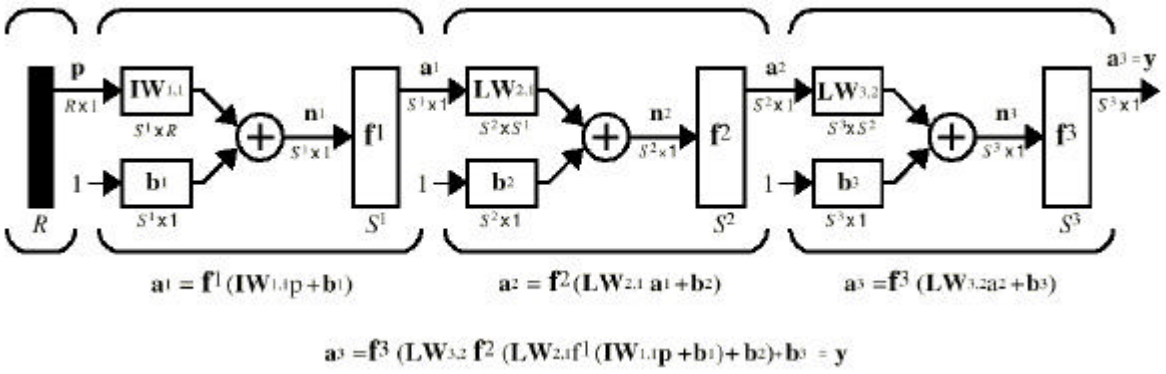


Dentro de una red neuronal, los elementos de procesamiento se encuentran agrupados por capas; una capa es una colección de neuronas, que de acuerdo a su ubicación recibe diferentes nombres: Capa de entrada a aquella que recibe las señales de entrada (aunque algunos autores la consideran simplemente como un vector de entrada, ya que en ella no se lleva a cavo ningún proceso). Capas Ocultas a aquellas que no tienen contacto con los datos exteriores, sus elementos pueden poseer diferentes formas conexión, determinando de esta manera la topología de la red; y Capa de Salida, a aquella que recibe la información de las capas ocultas y la transmite al medio exterior.

Cuando se tienen varias capas en una red neuronal, cada capa tiene una matriz de pesos $\boldsymbol{W}$, un vector de factores de sesgo $\boldsymbol{b}$ y un vector de salida $\boldsymbol{a}$. En la figura 35, se muestra el esquema de una red de tres capas (Una capa de entrada, una capa de salida y una capa oculta) y en la parte inferior la ecuación resultante para el vector de salida.

Para distinguir entre las matrices de pesos, las salidas, las funciones de activación etc., de cada capa, se coloca un superíndice en la variable. En la figura 36 se muestra la red de tres capas de la figura 35, como una representación matricial.

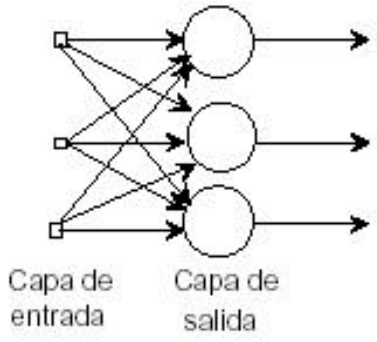
Figura 35. Representación matricial de la red ejemplo de 3 capas



5.1.1 Taxonomía de las RNA's. Los elementos básicos comentados anteriormente se pueden conectar entre sí para dar lugar a las estructuras neuronales o modelos conexionistas que podrían ser clasificados de diferentes formas según el criterio usado. De esta manera se podrían tener redes neuronales de los siguientes tipos:

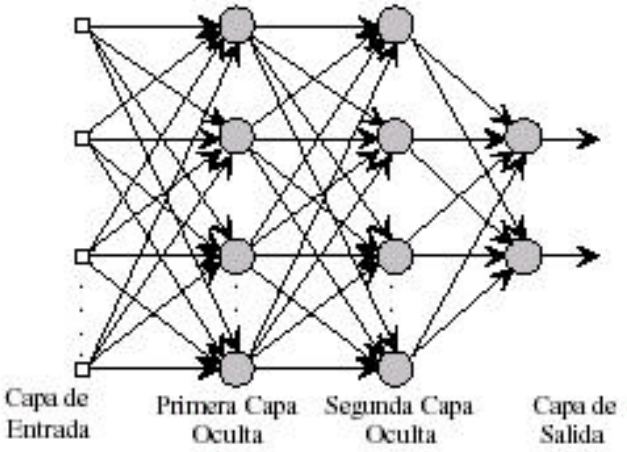
5.1.1.1 Según el número de capas. Pueden ser sub-clasificadas como Rede Neuronales Monocapa que corresponde a la red neuronal más sencilla, en donde una capa de neuronas proyecta las entradas a una capa de salida donde se realizan diferentes cálculos; por no realizarse ningún cálculo en la capa de entrada, se consideran estas redes como una sola capa. Una aplicación típica de este tipo de redes es como memorias asociativas.

Figura 36. Red monocapa



La otra clasificación la conforman las Redes Neuronales Multicapa, que serían una generalización de la anterior existiendo un conjunto de capas intermedias entre la entrada y la salida (capas ocultas). Este tipo de red puede estar total o parcialmente conectada.

Figura 37. Red multicapa

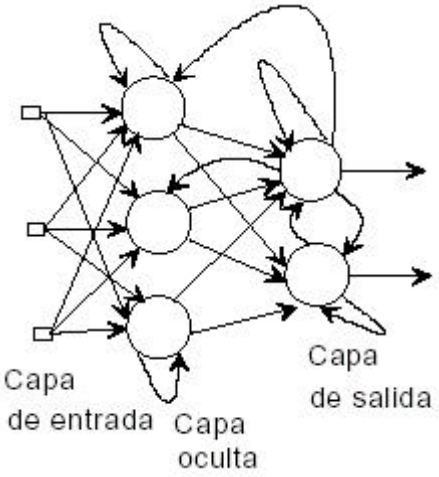


5.1.1.2 Según la forma de conexión. Pueden ser sub-clasificadas según su conexión como: Redes Neuronales Unidireccionales (feed-forward), en donde la propagación de las señales se produce en un sentido solamente, no existiendo la posibilidad de realimentaciones. Como ejemplo puede observarse la figura 37.

O como Redes Neuronales Recurrentes, las cuales se caracterizan por la existencia de lazos de realimentación. Estos lazos pueden ser entre neuronas de diferentes capas, neuronas de la misma capa o, más sencillamente, entre una misma neurona. Esta estructura recurrente la hace especialmente adecuada para estudiar la dinámica de sistemas no lineales. En la figura 38 se representa el esquema de una red recurrente.

5.1.1.3 Según la forma de asociación. Pueden ser sub-clasificadas como Redes Heteroasociativas, cuando asocian un patrón de entrada A con otro diferente B a la salida; y Redes Autoasociativas, cuando asocian un patrón de entrada A (posiblemente perturbado por ruido) consigo mismo a la salida.

Figura 38. Red neuronal recurrente



5.1.1.4 Según la dinámica de actualización. Puden ser sub-clasificadas como Redes Síncronas, cuando los estados se actualizan en función de un cierto reloj común a todas las neuronas, las cuales se van actualizando por capas, empezando por las de entrada y prosiguiendo hacia la salida.

El caso contrario a las anteriores son las Redes Asíncronas, es decir, no hay reloj, y entonces las neuronas actualizan su estado independientemente de las demás.

Por otro lado, las Redes Estocásticas generan una componente aleatoria tanto en el instante de actualización, como en el valor del estado.

5.1.2 Definición de parámetros en la red neuronal.

Además de la selección de la función de activación en cada una de las neuronas, el usuario deberá definir otros parámetros de la red neuronal en conjunto, antes de iniciar el proceso de entrenamiento. Estos parámetros son: arquitectura de red, proceso de aprendizaje, criterio de normalización y criterio de parada. A continuación se presenta una descripción de cada uno de ellos.

5.1.2.1 Arquitectura de red. Es la forma como se conectan las neuronas que conforman la red; la arquitectura o topología de red está definida por el número de capas, número de neuronas por capa y forma de conexión. En cuanto a esta última característica, ya se habló en el numeral anterior y se dice que la red es alimentada hacia adelante (feed-forward) cuando las neuronas se conectan sólo hacia capas posteriores[17], o recurrente cuando una o varias salidas de las neuronas de una capa se emplean como realimentación de si mismas o de otras neuronas de una capa anterior.

5.1.2.2 Proceso de Aprendizaje. Es el orden de sucesos a realizar para modificar los valores de los pesos a través de toda la red neuronal, a fin de obtener una salida deseada, partiendo de sus correspondientes entradas a la red.

Los pesos de una red neuronal son los responsables de que dos redes con arquitecturas idénticas resuelvan problemas totalmente diferentes, y por lo tanto, estos representan la experiencia de la red sobre un problema particular. Es por ello que adicionalmente a la estructura, la red neuronal necesita utilizar un mecanismo que le permita establecer las intensidades de cada conexión sináptica o pesos de manera eficiente. Son muchas las variedades en cuanto a procesos para conseguir el aprendizaje de la red. Sin embargo, se pueden diferenciar dentro del proceso unas reglas o algoritmos de aprendizaje[18] y unos paradigmas de aprendizaje.

Los algoritmos de aprendizaje, son simplemente metodologías planteadas por investigadores en el campo, para lograr el aprendizaje de la red mediante alguna

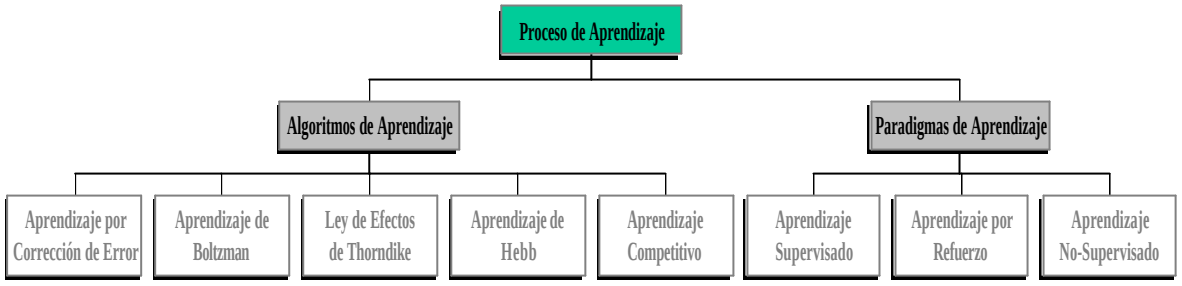
---

[17] Aquí también se distingue el término totalmente conectada, cuando una neurona de una capa anterior se conecta con todas las neuronas de la capa siguiente, o parcialmente conectada cuando una neurona de la capa anterior se conecta sólo con algunas de la capa siguiente.

[18] Es común emplear el término “algoritmo de entrenamiento”, en vez de algoritmo de aprendizaje, dada la naturaleza del proceso, más similar a un entrenamiento, por el número de veces que se le muestran a la red los ejemplos, con respecto a los cuales, tiene que adaptar sus pesos

técnica matemática específica. Los paradigmas de aprendizaje, se refieren a la manera como la red se relaciona con su ambiente de aprendizaje. En la figura 39 se muestra un esquema de los principales algoritmos y paradigmas de aprendizaje según [Haykin-1999].

Figura 39. Clasificación de los modelos de aprendizaje



Entre los algoritmos de aprendizaje, el de corrección de error es el más comúnmente usado en todas sus variantes (regla delta, tasa de aprendizaje adaptativa, técnica del gradiente descendente, algoritmo quasi-Newton, Levenberg-Maquardt, entre otros), se fundamenta simplemente en minimizar una función basada en el error entre la respuesta objetivo y la respuesta de la red en la simulación. La función de error se plantea a partir de un criterio estadístico, el más comúnmente usado es el error MSE (mean-square-error) o error medio cuadrado. Las Técnicas por corrección de error son estrictamente un problema de optimización en el cual cualquier algoritmo para tal fin puede ser planteado.

Los algoritmos de aprendizaje de Hebb y el competitivo están inspirados en el comportamiento neurobiológico de las neuronas naturales. En el caso del primero, éste se basa en el postulado del Neurobiólogo Donald O. Hebb el cual dice: ”Cuando un axón de la célula A esta suficientemente próximo para excitar una célula B y repetidamente o persistentemente toma parte de su activación, tiene lugar algún proceso de crecimiento o algún cambio metabólico en una o ambas células, de tal modo que la eficiencia de A, como una de las células que

desencadena la activación de B, se ve incrementada". Este postulado aplicado a las redes neuronales artificiales, establece un refuerzo a las neuronas que toman mayor parte en el proceso de aprendizaje, incrementado el valor de los pesos entre las neuronas participantes. En el aprendizaje competitivo, como su nombre lo indica, las neuronas de salida de la red compiten entre ellas por ser activadas. A diferencia del algoritmo de Hebb, en donde dos neuronas pueden ser simultáneamente activas, mientras que en el caso del aprendizaje competitivo sólo una sola neurona está activa.

El aprendizaje de Boltzman es un algoritmo estocástico derivado de la teoría termodinámica y la mecánica estadística. En este caso la salida de las unidades de procesamiento son una función estocástica de las entradas, en vez de ser una función determinista como en el algoritmo de corrección de error. Adicionalmente, en las neuronas de la red, sus salidas se calculan usando probabilidades en vez de una función umbral o una función de salida sigmoidea. La función a minimizar en este caso es la energía y se emplea la distribución de Boltzman para encontrar las probabilidades de los estados energéticos. Este mismo planteamiento es empleado en otra técnica de optimización conocida como simulated annealing o temple simulado.

En cuanto a los paradigmas de aprendizaje, éstos se clasifican como aprendizaje supervisado, aprendizaje por refuerzo y aprendizaje no-supervisado. Los supervisados requieren que cada entrada que ha de ser utilizada en el entrenamiento, esté acompañada de la respuesta o salida que la red debe producir ante ella. En este aprendizaje, también conocido como aprendizaje por corrección de error, se conoce la magnitud del error y ésta determina la magnitud en el cambio de los pesos. Es la forma más común de aprendizaje empleada.

En el aprendizaje por refuerzo sólo se conoce si la salida de la red corresponde o no con la señal deseada, es decir, la información es de tipo booleana (verdadero o falso), y se basan en la ley de efectos de Thorndike, la cual es a su vez, un

algoritmo de entrenamiento. Este algoritmo se basa en que si un sistema responde satisfactoriamente a una acción, este proceso es reforzado.

Los paradigmas de aprendizaje no-supervisados, generalmente utilizados en clasificación, requieren a cambio de la respuesta deseada, asignar valores a un conjunto de parámetros que regulan el aprendizaje.

5.1.2.3 Criterios de Normalización y Parada. La selección (incluso creación) de estos dos aspectos, hacen particular dos procesos de entrenamiento, ya que se manipula de cierta forma, por parte del usuario, la información de entrada y las respuestas a la salida de la red. La normalización de la información (convertir datos a valores en el rango de 0 a 1) de entrada, es necesaria si se considera el uso de funciones de activación tales como las sigmoides (tansig y logsig), cuyos valores están en rangos de 0 a 1. Una normalización común de un dato a ser ingresado a la red, puede ser realizada empleando los valores máximos y mínimos de los datos disponibles, por ejemplo:

$$Dato\ Normalizado = (Dato - Valor\ Mín)/(Valor\ Máx - Valor\ Mín) \quad \text{Ec. 43}$$

El criterio de normalización depende de los requerimientos del usuario en las funciones de activación que empleará en la red.

En cuanto al criterio de parada, un sistema que se considere inteligente deberá prever mecanismos para detener su proceso de aprendizaje cuando las metas en cuanto al error (diferencia entre la respuesta real del sistema y la obtenida por la red) han sido alcanzadas o para dar por terminado el proceso cuando parezca que no se van a poder alcanzar tales metas, evitando gastar tiempo computacional inútil. Existen diferentes formas de contabilizar el error entre una respuesta de la red y la respuesta real objetivo. Estos errores provienen de la teoría estadística, y entre ellos los más comúnmente usados son: error residual, desviación media

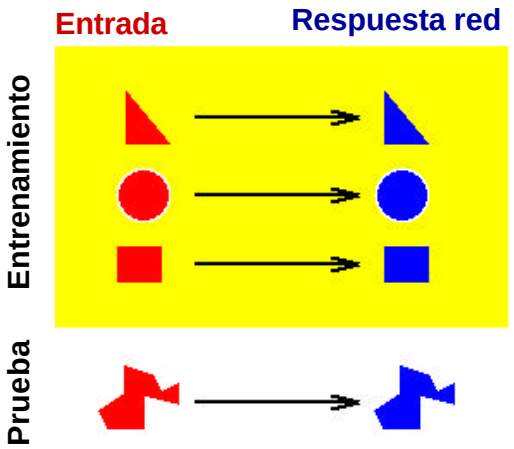
absoluta, error medio cuadrado, porcentaje de error medio absoluto y porcentaje medio de error.

El criterio de parada es de vital importancia en todo algoritmo de optimización a nivel general y en este caso concreto en todo algoritmo de entrenamiento de un sistema de aprendizaje basado en ejemplos. Sus funciones principales son:

1. Reducir el tiempo de aprendizaje: Es frecuente al utilizar ciertos métodos de optimización en los que la llegada al mínimo sea muy lenta, por lo que es aconsejable parar antes de llegar a éste, obteniendo aún buenos resultados.

2. Asegurar que el sistema de aprendizaje tenga una buena generalización: Se entiende por generalización, cuando la red es capaz de dar una respuesta correcta, cuando se le muestran entradas diferentes a aquella con las cuales ha sido entrenada. En la figura 41 se muestra un esquema idealizado del concepto de generalización.

Figura 40. Ejemplo del concepto de generalización



La topología de una red neuronal, es decir, el número de nodos y la ubicación y el número de conexiones entre ellos, tiene un impacto significativo en el desempeño de la red y su habilidad para generalizar. La densidad de conexiones en una red

neuronal determina su habilidad para almacenar información. Si una red no tiene suficientes conexiones entre nodos, el algoritmo de entrenamiento puede no converger nunca; la red neuronal no es capaz de aproximar la función. Por otro lado, en una red densamente conectada, puede ocurrir el sobreajuste (overfitting), el cual es un problema de los modelos estadísticos donde se presentan demasiados parámetros. Esto es una mala situación porque en lugar de aprender a aproximar la función presente en los datos, la red simplemente puede memorizar cada ejemplo de entrenamiento. El ruido en los datos de entrenamiento se aprende entonces como parte de la función, a menudo destruyendo la habilidad de la red para generalizar.

El criterio de parada no implica en sí la ecuación o el tipo de error empleado, sino el juicio utilizado por el algoritmo de entrenamiento de la red para detener el proceso. Este criterio puede estar basado en las metas de error de entrenamiento y/o de validación, sobre los cuales se hará referencia posteriormente.

En algunas formas de entrenamiento como el backpropagation, la esencia del aprendizaje es codificar un mapeo entrada-salida (representado por un conjunto de ejemplos etiquetados) en los pesos y umbrales de un Perceptrón multicapa. Lo que se desea es que la red se entrene bien de forma tal que aprenda lo suficiente acerca del pasado para generalizar en el futuro. Desde esta perspectiva el proceso de aprendizaje debe elegir la parametrización de la red más acorde a este conjunto de ejemplos.

Más específicamente, se puede considerar el problema de selección de la red como una elección, dentro del conjunto de modelos de estructuras candidatas (parametrizaciones), de la mejor estructura de acuerdo a un cierto criterio. En este contexto, una herramienta estándar en estadística conocida como validación cruzada provee una guía interesante [Haykin-1999]. En ella set de entrenamiento se particiona aleatoriamente en dos subsets disyuntos:

- El subset de estimación, usado para seleccionar el modelo (también llamado set

de entrenamiento).

- El subset de validación, usado para evaluar o validar el modelo.

La motivación aquí es validar el modelo sobre un set de datos diferente de aquel utilizado para la estimación de los parámetros. De este modo se puede usar el set de entrenamiento para evaluar la desempeño de varios modelos candidatos, y así elegir el mejor.

El uso de la validación cruzada resulta particularmente interesante cuando se debe diseñar una red neuronal grande que tenga como objetivo a la generalización.

Comúnmente, un perceptrón multicapa entrenado con el algoritmo de backpropagation aprende en etapas, moviéndose desde la realización de funciones de mapeo bastante simples a más complejas a medida que progresa la sesión de entrenamiento. Esto se ejemplifica por el hecho de que en una situación típica el error cuadrático medio disminuye con el incremento del número de repeticiones durante el entrenamiento: comienza con un valor grande, decrece rápidamente, y luego continúa disminuyendo lentamente mientras la red hace su camino hacia un mínimo local sobre la superficie de error. Teniendo como meta a una buena generalización, es muy difícil darse cuenta cuándo es el mejor momento de detener el entrenamiento si solamente se está mirando a la curva de aprendizaje para el entrenamiento.

En particular, como se mencionó anteriormente, es posible que la red termine sobreajustándose a los datos de entrenamiento si la sesión de entrenamiento no se detiene en el momento correcto. Sin embargo, se puede identificar el comienzo del sobreajuste a través del uso de la validación cruzada, para lo cual los ejemplos de entrenamiento se separan en un subset de estimación y un subset de validación. El subset de estimación se utiliza para entrenar a la red en el modo usual, excepto por una modificación menor: la sesión de entrenamiento se detiene

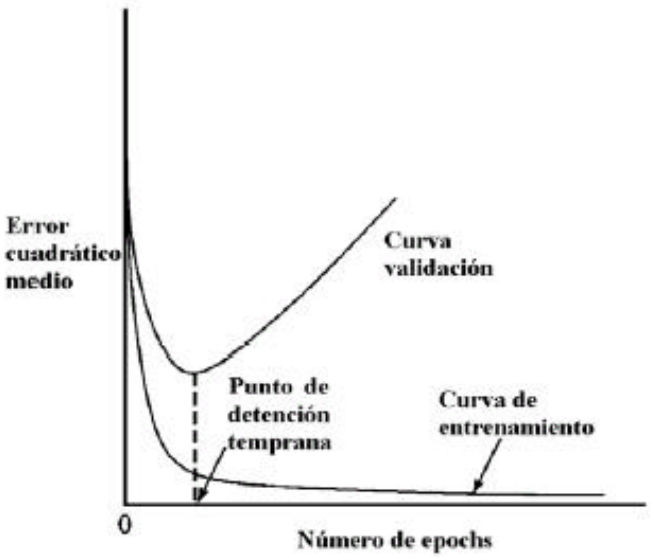
periódicamente (cada tantas repeticiones), y se evalúa la red con el set de validación después de cada período de entrenamiento. Más específicamente, el proceso periódico de estimación seguida de validación procede de la siguiente manera:

1. Después del período de estimación (entrenamiento), se fijan todos los pesos y los umbrales del Perceptrón multicapa, y la red opera en su modo hacia delante. El error de validación se mide así para cada ejemplo en el set de validación.

2. Cuando la fase de validación se completa, la estimación (entrenamiento) se reanuda para otro período y el proceso se repite.

Este procedimiento se denomina método de entrenamiento con detención temprana o "early stopping". En la figura 42 se muestran las formas conceptualizadas de dos curvas de aprendizaje, una perteneciente a las medidas sobre el subset de estimación y la otra sobre el subset de validación.

Figura 41. Regla de detención temprana basada en validación cruzada



Normalmente, el modelo no trabaja tan bien sobre el subset de validación como lo hace sobre el set de estimación, en el cual se basó su diseño. La curva de aprendizaje de estimación decrece monótonamente para un número creciente de repeticiones en la forma acostumbrada.

En contraste, curva de aprendizaje de validación decrece monótonamente hasta un mínimo, entonces empieza a incrementarse mientras continúe el entrenamiento. Al observar la curva de aprendizaje independientemente se podría pensar que después del punto señalado como detección, se podría mejorar aún más; en realidad, lo que la red está aprendiendo más allá de ese punto es esencialmente ruido contenido en el set de entrenamiento, lo cual también produce efectos no deseados en la respuesta de la red. A este problema se le conoce como sobreentrenamiento (overtraining), sugiriendo la heurística que el punto mínimo sobre la curva de aprendizaje de validación sea utilizado como un criterio para detener la sesión de entrenamiento.

5.1.3 El perceptrón multicapa.

El perceptrón multicapa es la red neuronal artificial más conocida y de mayor número de aplicaciones dentro del estado del arte de las redes neuronales artificiales; por su importancia dentro de este trabajo de investigación, se profundizará con mayor detalle este tipo de red neuronal.

Su historia comienza en 1958 cuando Rosenblatt publica los primeros trabajos sobre un modelo neuronal y su algoritmo de aprendizaje, al cual llamó perceptrón, en “The Perceptrón: A Probabilistic Model for Information Storage and Organization in the Brain”.

El perceptrón está formado por una única neurona (ver figura 33), por lo que su utilización está limitada a la clasificación de patrones en dos clases. Si se expande esta capa de salida con más de una neurona se ampliará el número de

clases posibles aunque con la limitación demostrada por Minsky y Papert en 1969, consistente en que estas clases deben ser linealmente separables, de lo cual se habló en el inicio de este capítulo.

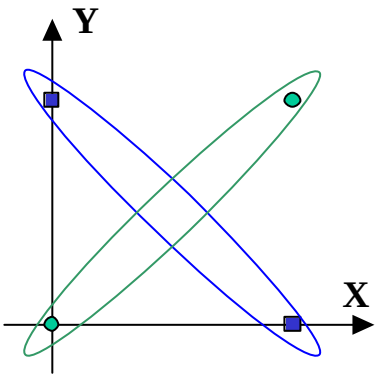
Esta limitación imposibilita al perceptrón para resolver problemas tan sencillos como el de la función lógica XOR de 2 entradas mostrada en la tabla 3, lo cual produjo un estancamiento en el desarrollo investigativo en torno a las RNA.

Tabla 3. Función XOR

X	Y	XOR
0	0	0
0	1	1
1	0	1
1	1	0

La siguiente figura muestra que las dos clases son no linealmente separables representándose los ceros por círculos y los unos por cuadrados.

Figura 42. Representación de la función XOR de dos entradas



La separación de clases no linealmente separables se consigue introduciendo al menos una capa de neuronas entre la salida y la entrada como se muestra en la figura 34. A este tipo de red se le denominó como "perceptrón multicapa" y nació

básicamente con la intención de dar solución a las limitaciones del perceptrón clásico o monocapa, y supuso el resurgimiento del movimiento conexionista a inicios de los 80's.

El problema de este tipo de perceptrón está en su entrenamiento, ya que es difícil modificar correctamente los pesos de la capa oculta. Aparece entonces el problema de asignar un “error” durante el proceso de aprendizaje a estas neuronas; este problema es conocido en la teoría de los sistemas conexionistas como el problema de la "asignación de crédito" (assignment credit). El cual fue resuelto por una serie de investigadores de forma independiente pero que alcanzó su difusión en el trabajo de Rumelhart, Hinton y Williams denominado "Learning Internal Representations by Error Propagation", lo cual dio luces para lo que posteriormente se conoció como la regla de aprendizaje "Backpropagation", que tal como su nombre indica tiene la función de ir propagando los errores producidos en la capa de salida hacia atrás.

5.1.4 La regla de aprendizaje Backpropagation. El Algoritmo backpropagation para redes multicapa realiza su labor para actualización de pesos y ganancias mediante el error medio cuadrático. La red con backpropagation trabaja bajo aprendizaje supervisado y por tanto necesita un set de entrenamiento que le describa cada salida y su valor de salida esperado de la forma $\{p_1, t_1\},\{p_2, t_2\},....,\{p_Q, t_Q\}$. Donde p_Q es una entrada de la red y t_Q es la correspondiente salida deseada para el patron q-ésimo.

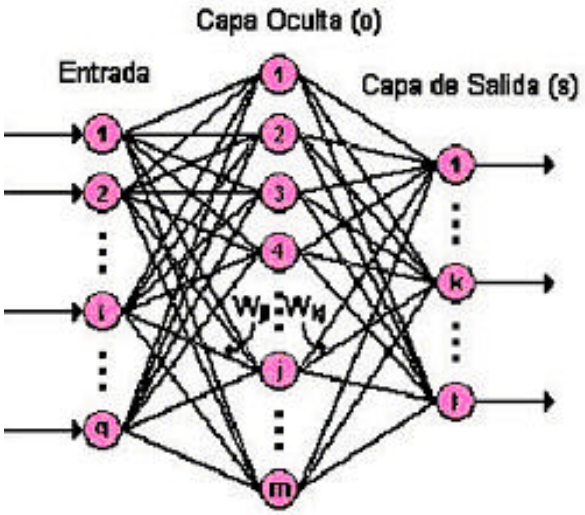
El entrenamiento de una red neuronal multicapa se realiza mediante un proceso de aprendizaje, en el cual se debe inicialmente tener definida la topología de la red esto es: número de neuronas en la capa de entrada el cual depende del número de elementos del vector de entrada, cantidad de capas ocultas y número de neuronas de cada una de ellas, número de neuronas en la capa de salida el cual

depende del número de componentes del vector de salida o patrones objetivo y funciones de transferencia requeridas en cada capa; con base en la topología escogida se asignan valores iniciales a cada uno de los parámetros que conforma la red.

Cada patrón de entrenamiento se propaga a través de la red y sus parámetros para producir una respuesta en la capa de salida, la cual se compara con los patrones objetivo o salidas deseadas para calcular el error en el aprendizaje, este error marca el camino mas adecuado para la actualización de los pesos y ganancias que al final del entrenamiento producirán una respuesta satisfactoria a todos los patrones de entrenamiento, esto se logra minimizando el error medio cuadrático en cada iteración del proceso de aprendizaje.

A continuación se presenta la deducción matemática de este procedimiento para una red con una capa de entrada, una capa oculta y una capa de salida como se aprecia en la figura 43, donde $\boldsymbol{q}$ es el número de componentes del vector de entrada, $\boldsymbol{m}$ el número de neuronas de la capa oculta, y $\boldsymbol{l}$ el número de neuronas de la capa de salida.

Figura 43. Disposición de una red ejemplo de 3 capas



Para iniciar el entrenamiento se presenta a la red un patrón de entrenamiento $\boldsymbol{P}$, el cual tiene $\boldsymbol{q}$ componentes (ver ecuación 44). Cuando se le presenta a la red un patrón de entrenamiento, este se propaga a través de las conexiones existentes produciendo una entrada neta $\boldsymbol{n}$ en cada una las neuronas de la siguiente capa, la entrada neta a la neurona $\boldsymbol{j}$ de la siguiente capa debido a la presencia de un patrón de entrenamiento en la entrada esta dada por la ecuación 44, nótese que la entrada neta es el valor justo antes de pasar por la función de transferencia.

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{p_1} \\ \boldsymbol{p_2} \\ \vdots \\ \boldsymbol{p_q} \end{bmatrix} \text{ (patrón de entrenamiento)} \qquad \boldsymbol{n_j^0} = \sum_{i=1}^{q} \boldsymbol{W_{ij}^0 p_i} + \boldsymbol{b_j^0} \quad \textbf{Ec. 44}$$

Donde:

$\boldsymbol{W_{ij}^0}$: Peso que une la componente $\boldsymbol{i}$ de la entrada con la neurona $\boldsymbol{j}$ de la capa oculta.

$\boldsymbol{p_i}$: Componente $\boldsymbol{i}$ del vector $\boldsymbol{p}$ que contiene el patrón de entrenamiento de *q* componentes.

$\boldsymbol{b^0_j}$: Ganancia de la neurona $\boldsymbol{j}$ de la capa oculta.

Donde el superíndice (^{0}) representa la capa a la que pertenece cada parámetro, es este caso la capa oculta.

Cada una de las neuronas de la capa oculta tiene como salida $\boldsymbol{a^0_j}$ que esta dada por la ecuación 45.

$$\boldsymbol{a_0^j} = \boldsymbol{f}^0 \left[\sum_{i=1}^{q} \boldsymbol{W_{ji}^0 p_i} + \boldsymbol{b_j^0} \right] \quad \textbf{Ec. 45}$$

Donde:

f^o : Función de transferencia de las neuronas de la capa oculta

Las salidas a^o_j de las neuronas de la capa oculta (de l componentes) son las entradas a los pesos de conexión de la capa de salida, este comportamiento esta descrito por la ecuación 46.

$$n^s_k = \sum_{j=1}^{m} W^s_{kj} a^0_j + b^s_k \quad \textbf{Ec. 46}$$

Donde:

W^s_{kj} : Peso que une la neurona j de la capa oculta con la neurona k de la capa de salida, la cual cuenta con s neuronas

a^o_j : Salida de la neurona j de la capa oculta, la cual cuenta con m neuronas.

b^s_k : Ganancia de la neurona k de la capa de salida.

n^s_k : Entrada neta a la neurona k de la capa de salida

La red produce una salida final descrita por la ecuación 47.

$$a^s_k = f^s\left(n^s_k\right) \quad \textbf{Ec. 47}$$

Donde:

f^s : Función de transferencia de las neuronas de la capa de salida

Reemplazando la ecuación 46 en la 47 se obtiene la salida de la red en función de la entrada neta y de los pesos de conexión con la ultima capa oculta.

$$a_k^s = \left[\sum_{j=1}^{m} W_{kj}^s a_j^s + b_k^s \right] \quad \textbf{Ec. 48}$$

La salida de la red de cada neurona a^s_k se compara con la salida deseada t_k para calcular el error en cada unidad de salida.

$$d_k = \left(t_k - a_k^s \right) \quad \textbf{Ec. 49}$$

El error debido a cada patrón p propagado esta dado por:

$$e_p^{\,2} = \frac{1}{2} \sum_{k=1}^{s} (d_k)^2 \quad \textbf{Ec. 50}$$

Donde:

$e_p^{\,2}$: Error medio cuadrático para cada patrón de entrada p

d_k : Error en la neurona k de la capa de salida con l neuronas

Este proceso se repite para el número total de patrones de entrenamiento (r), para un proceso de aprendizaje exitoso el objetivo del algoritmo es actualizar todos los pesos y ganancias de la red minimizando el error medio cuadrático total descrito en la siguiente expresión:

$$e^2 = \sum_{p=1}^{r} e_p^2 \quad \textbf{Ec. 51}$$

Donde:

e^2 : Error total en el proceso de aprendizaje en una iteración luego de haber presentado a la red los r patrones de entrenamiento

El error que genera una red neuronal en función de sus pesos, genera un espacio de n dimensiones, donde n es el número de pesos de conexión de la red, al evaluar el gradiente del error en un punto de esta superficie se obtendrá la

dirección en la cual la función del error tendrá un mayor crecimiento, como el objetivo del proceso de aprendizaje es minimizar el error debe tomarse la dirección negativa del gradiente para obtener el mayor decremento del error y de esta forma su minimización, condición requerida para realizar la actualización de la matriz de pesos en el algoritmo backpropagation.

$$W_{k+1} = W_k - a\tilde{N}e_p^2 \quad \textbf{Ec. 52}$$

El gradiente negativo de e_p^2 se denotara como $-\tilde{N}e_p^2$ y se calcula como la derivada del error respecto a todos los pesos de la red.

En la capa de salida el gradiente negativo del error con respecto a los pesos es:

$$-\frac{\P e_p^2}{\P W_{kj}^s} = -\frac{\P}{\P W_{kj}^s}\left[\frac{1}{2}\sum_{k=1}^{l}(t_k - a_k^s)^2\right] = (t_k - a_k^s)' \frac{\P a_k^s}{\P W_{kj}^s} \quad \textbf{Ec. 53}$$

Donde:

$-\frac{\P e_p^2}{\P W_{kj}^s}$: Componente del gradiente $-\tilde{N}e_p^2$ respecto al peso de la conexión de la neurona de la capa de salida y la neurona j de la capa oculta, respecto al peso W^s_{kj}

$\frac{\P a_k^s}{\P W_{kj}^s}$: Derivada de la salida de la neurona k de la capa de salida respecto al peso W^s_{kj}

Para calcular $\frac{\P a_k^s}{\P W_{kj}^s}$ se debe utilizar la regla de la cadena, pues el error no es una función explícita de los pesos de la red, de la ecuación 47 puede verse que la salida de la red a^s_k esta explícitamente en función de n^s_k y de la ecuación 46 puede verse que n^s_k esta explícitamente en función de W^s_{kj}, considerando esto se genera

la siguiente ecuación:

$$\frac{\partial a_k^s}{\partial W_{kj}^s} = \frac{\partial a_k^s}{\partial n_k^s} \cdot \frac{\partial n_k^s}{\partial W_{kj}^s} \quad \textbf{Ec. 54}$$

Tomando la ecuación 54 y reemplazándola en la ecuación 53 se obtiene:

$$-\frac{\partial e_p^2}{\partial W_{kj}^s} = (t_k - a_k^s) \cdot \frac{\partial a_k^s}{\partial n_{kj}^s} \cdot \frac{\partial n_k^s}{\partial W_{kj}^s} \quad \text{Ec. 55}$$

Donde:

$\frac{\partial n_k^s}{\partial W_{kj}^s}$: Derivada de la entrada neta a la neurona k de la capa de salida respecto a los pesos de la conexión entre las neuronas de la capa oculta y la capa de salida.

$\frac{\partial a_k^s}{\partial n_k^s}$: Derivada de la salida de la neurona k de la capa de salida respecto a su entrada neta.

Reemplazando en la ecuación 55 las derivadas de las ecuaciones 46 y 47 se obtiene:

$$-\frac{\partial e_p^2}{\partial W_{kj}^s} = (t_k - a_k^s) \cdot f'^s(n_k^5) \cdot a_j^0 \quad \text{Ec. 56}$$

Como se observa en la ecuación 56 las funciones de transferencia utilizadas en este tipo de red deben ser continuas para que su derivada exista en todo el intervalo, ya que el término $f'^s(n^s_k)$ es requerido para el cálculo del error.

Las dos funciones de transferencia f más utilizadas y sus respectivas derivadas

son las siguientes:

- Logsig:

$$f(n) = \frac{1}{1 + e^{-n}} \qquad f'(n) = f(n)(1 - f(n)) \qquad f'(n) = a(1 - a) \quad \text{Ec. 57}$$

- Tansig:

$$f(n) = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}} \qquad f'(n) = 1 - (f(n))^{2} \qquad f'(n) = (1 - a^{2}) \quad \textbf{Ec. 58}$$

De la ecuación 56, los términos del error para las neuronas de la capa de salida están dados por la ecuación 59, la cual se le denomina comúnmente sensitividad de la capa de salida.

$$d_k^s = (t_k - a_k^s) \cdot f'^s(n_k^5) \quad \textbf{Ec. 59}$$

La denominación de este algoritmo como Backpropagation o de propagación inversa, se debe a que el error se propaga de manera inversa al funcionamiento normal de la red, de esta forma, el algoritmo encuentra el error en el proceso de aprendizaje desde las capas más internas hasta llegar a la entrada; con base en el cálculo de este error se actualizan los pesos y ganancias de cada capa.

Después de conocer la ecuación 59 se procede a encontrar el error en la capa oculta, el cual está dado por la ecuación 60.

$$-\frac{\partial e_p^2}{\partial W_{ji}^0} = -\frac{\partial}{\partial W_{ji}^0}\left[\frac{1}{2}\sum_{k=1}^{l}(t_k - a_k^s)^2\right] = \sum_{k=1}^{l}(t_k - a_k^s)\cdot\frac{\partial a_k^s}{\partial W_{ji}^0} \quad \text{Ec. 60}$$

Para calcular el último término de la ecuación 60 se debe aplicar la regla de la cadena en varias ocasiones como se observa en la ecuación 61 puesto que la salida de la red no es una función explícita de los pesos de la conexión entre la capa de entrada y la capa oculta.

$$\frac{\partial a_k^s}{\partial W_{ji}^0} = \frac{\partial a_k^s}{\partial n_k^s} \cdot \frac{\partial n_k^s}{\partial a_k^0} \cdot \frac{\partial a_k^o}{\partial n_j^0} \cdot \frac{\partial n_j^0}{\partial W_{ji}^0} \quad \textbf{Ec. 61}$$

Si todos los términos de la ecuación 61 son derivados respecto a variables de las que dependan explícitamente, reemplazando 61 en 60 se tiene:

$$-\frac{\partial e_p^2}{\partial W_{ji}^0} = \sum_{k=1}^{l} (t_k - a_k^s) \cdot \frac{\partial a_k^s}{\partial n_k^s} \cdot \frac{\partial n_k^s}{\partial a_k^0} \cdot \frac{\partial a_k^o}{\partial n_j^0} \cdot \frac{\partial n_j^0}{\partial W_{ji}^0} \quad \textbf{Ec. 62}$$

Tomando las derivas de las ecuaciones 44, 45, 46 y 47 y reemplazándolas en la ecuación 62 se obtiene la expresión del gradiente del error en la capa oculta.

$$-\frac{\partial e_p^2}{\partial W_{ji}^0} = \sum_{k=1}^{l} (t_k - a_k^s) \cdot f'^s(n_k^s) \cdot W_{kj}^s \cdot f'^0(n_j^0) \cdot p_i \quad \textbf{Ec. 63}$$

Reemplazando la ecuación 59 en la ecuación 63 se tiene:

$$-\frac{\partial e_p^2}{\partial W_{ji}^0} = \sum_{k=1}^{l} d_k^s \cdot W_{kj}^s \cdot f'^0(n_j^0) \cdot p_i \quad \textbf{Ec. 64}$$

Los términos del error para cada neurona de la capa oculta esta dado por la ecuación 65, este término también se denomina sensitividad de la capa oculta.

$$d_j^0 = f'^0(n_j^0) \cdot \sum_{k=1}^{l} d_k^s \cdot W_{kj}^s \quad \textbf{Ec. 65}$$

Luego de encontrar el valor del gradiente del error se procede a actualizar los pesos de todas las capas empezando por la de salida, para la capa de salida la actualización de pesos y ganancias esta dada por las ecuaciones 66 y 67.

$$W_{kj}(t+1) = W_{kj}(t) - 2\alpha d_k^s \quad \textbf{Ec. 66}$$

$$b_k(t+1) = b_k(t) - 2\alpha d_k^s \quad \textbf{Ec. 67}$$

Donde:

a: Tasa de aprendizaje que varía entre 0 y 1 dependiendo de las características del problema a solucionar.

Luego de actualizar los pesos y ganancias de la capa de salida se procede a actualizar los pesos y ganancias de la capa oculta mediante las ecuaciones 68 y 69.

$$W_{ji}(t+1) = W_{ji}(t) - 2a\, d_j^0\, p_i \quad \textbf{Ec. 68}$$

$$b_j(t+1) = b_j(t) - 2a\, d_j^0 \quad \textbf{Ec. 69}$$

Esta deducción fue realizada para una red de tres capas, si se requiere realizar el análisis para una red con dos o más capas ocultas, las expresiones pueden derivarse de la ecuación 65 donde los términos que se encuentran dentro de la sumatoria pertenecen a la capa inmediatamente superior. Este algoritmo es conocido como la regla Delta Generalizada desarrollada por Rumelhart D, la cual es una extensión de la regla delta desarrollada por Widrow en 1930.

Para algunos autores las sensitividades de las capas están detonadas por la letra S; reescribiendo las ecuaciones 59 y 65 con esta notación se obtienen las ecuaciones 70 y 71.

$$S^M = -2\, f^M(n^M)(t-a) \quad \textbf{Ec. 70}$$

$$S^m = f^m(n^m)(W^{m+1})^T S^{m+1}, \text{ para } \quad m = M-1,\ldots,2,1 \quad \textbf{Ec. 71}$$

En la ecuación 70, M representa la última capa y S^M la sensitividad para esta capa, la ecuación 71 expresa el cálculo de la sensitividad capa por capa comenzando desde la última capa oculta, cada uno de estos términos involucra que el término para la sensitividad de la capa siguiente ya este calculado.

Como se ve el algoritmo Backpropagation utiliza la técnica de aproximación en pasos descendientes, la única complicación está en el cálculo del gradiente, el cual es un término indispensable para realizar la propagación de la sensitividad.

En las técnicas de gradiente descendiente es conveniente avanzar por la superficie de error con incrementos pequeños de los pesos; esto se debe a que se tiene una información local de la superficie y no se sabe lo lejos o lo cerca que se está del punto mínimo, con incrementos grandes, se corre el riesgo de pasar por encima del punto mínimo, con incrementos pequeños, aunque se tarde más en llegar, se evita que esto ocurra.

El elegir un incremento adecuado influye en la velocidad de convergencia del algoritmo, esta velocidad se controla a través de la rata de aprendizaje a, la que por lo general se escoge como un número pequeño, para asegurar que la red encuentre una solución. Un valor pequeño de a significa que la red tendrá que hacer un gran número de iteraciones, si se toma un valor muy grande, los cambios en los pesos serán muy grandes, avanzando muy rápidamente por la superficie de error, con el riesgo de saltar el valor mínimo del error y estar oscilando alrededor de él, pero sin poder alcanzarlo.

Figura 44. Ejemplo del comportamiento de la superficie de error.

Es recomendable aumentar el valor de a a medida que disminuye el error de la red

durante la fase de entrenamiento, para garantizar así una rápida convergencia, teniendo la precaución de no tomar valores demasiado grandes que hagan que la red oscile alejándose demasiado del valor mínimo. Algo importante que debe tenerse en cuenta, es la posibilidad de convergencia hacia alguno de los mínimos locales que pueden existir en la superficie del error del espacio de pesos como se ve en la figura 44.

En el desarrollo matemático que se ha realizado para llegar al algoritmo backpropagation, no se asegura en ningún momento que el mínimo que se encuentre sea global, una vez la red se asiente en un mínimo sea local o global cesa el aprendizaje, aunque el error siga siendo alto. En todo caso, si la solución es admisible desde el punto de vista del error, no importa si el mínimo es local o global o si se ha detenido en algún momento previo a alcanzar un verdadero mínimo.

En 1989 Funahashi demostró matemáticamente que una red neuronal multicapa puede aproximar cualquier función no lineal o mapa lineal multivariable, $f(x) = R^n \rightarrow R$. Este teorema es de existencia, pues prueba que la red existe pero no indica como construirla y tampoco garantiza que la red aprenderá función [Funahashi-1990].

El algoritmo Backpropagation es fácil de implementar, y tiene la flexibilidad de adaptarse para aproximar cualquier función, siendo una de las redes multicapa más potentes; esta característica ha convertido a esta red en una de las más ampliamente utilizadas y ha llevado al desarrollo de nuevas técnicas que permitan su mejoramiento.

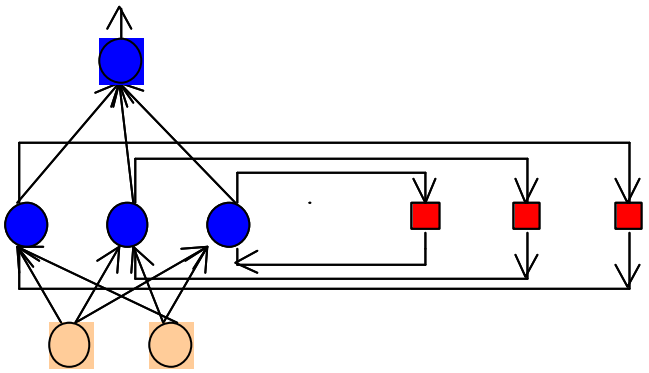
5.1.5 La red de Elman. A pesar de que las redes neuronales conectadas hacia adelante (feedforward) con el algoritmo backpropagation son las

más comúnmente usadas para la resolución de problemas relacionados con aproximación de funciones desconocidas y predicción de las mismas dentro del estado del arte estudiado, se encontró dentro de las redes recurrentes una red que también puede ser aplicada a la resolución del tipo de problemas requerido en este trabajo, a esta red se le conoce como red de Elman, en honor a su creador Jeff L. Elman de la Universidad de California [Elman-1990].

La red de Elman posee típicamente la estructura de capa de entrada, capas ocultas y capa de salida convencionales de una red tipo feedforward-backpropagation, con la adición de una conexión de realimentación que va desde la salida de la capa oculta hasta la entrada de la misma capa, lo cual le permite aprender a reconocer y generar patrones variantes con le tiempo, lo cual es el tipo de tareas a resolver en este trabajo de investigación.

En la figura 45 se muestra un ejemplo de la arquitectura de este tipo de red, donde se puede observar la capa adicional de realimentación.

Figura 45. Esquema de conexión de una red tipo Elman



La red de Elman posee por lo general neuronas de tipo sigmoide (tansig y logsig) en sus capas ocultas y de tipo lineal (purelin) en su capa de salida. La ventaja de este tipo de conexiones y funciones de transferencia empleados en la red de Elman, es que puede aproximar prácticamente cualquier función con la precisión

deseada, mientras ésta posea un número finito de discontinuidades, para lo cual se requiere un número adecuado de neuronas en la capa oculta [Funahashi-1990].

Para la red de Elman, la capa oculta es la capa recurrente y el retardo en las conexiones de realimentación almacena los valores de la iteración previa, los cuales serán usados en la siguiente iteración, permitiendo de esta manera que la red tenga memoria de la iteración anterior en el ajuste de sus pesos. Debido a los estados producidos en la realimentación, dos redes de Elman idénticas, con los mismos parámetros y datos de entrenamiento, podrían producir salidas diferentes en la misma iteración.

En cuanto al aprendizaje, la red de Elman pude ser entrenada con cualquier tipo de algoritmo tipo backpropagation como gradiente conjugado o Levemberg Marquard, dada su similitud con las redes tipo feedforward. El entrenamiento de la red puede resumirse en los siguientes pasos:

1. Presentar a la red los datos de entrenamiento y calcular la salida de la red con los pasos iniciales, comparar la salida de la red con los datos objetivo y generar la secuencia de error.

2. Propagar inversamente el error para encontrar el gradiente del error para cada conjunto de pesos y ganancias.

3. Actualizar los pesos y ganancias con el gradiente encontrado según el algoritmo backpropagation

La red de Elman no es tan confiable como otros tipos de redes porque el gradiente se calcula por aproximación del error. Para solucionar un problema con este tipo de red se requerirán mas neuronas en la capa oculta que con la red feedforward,

5.2 PREDICCIÓN DE DEMANDA MEDIANTE RNA'S

La idea detrás del uso de modelos de basados en redes neuronales para la predicción de parámetros relacionados con el consumo de energía es simple: No se requiere conocer una ecuación o función matemática que represente el comportamiento del consumo de energía, simplemente se requiere un sistema que a partir de un conjunto de datos históricos, determine el comportamiento futuro del consumo de energía. En otras palabras, se requiere de un modelo tipo "caja negra", en el cual no es de interés la ecuación matemática que modela el comportamiento, sino que respuesta da el sistema ante una entrada en particular.

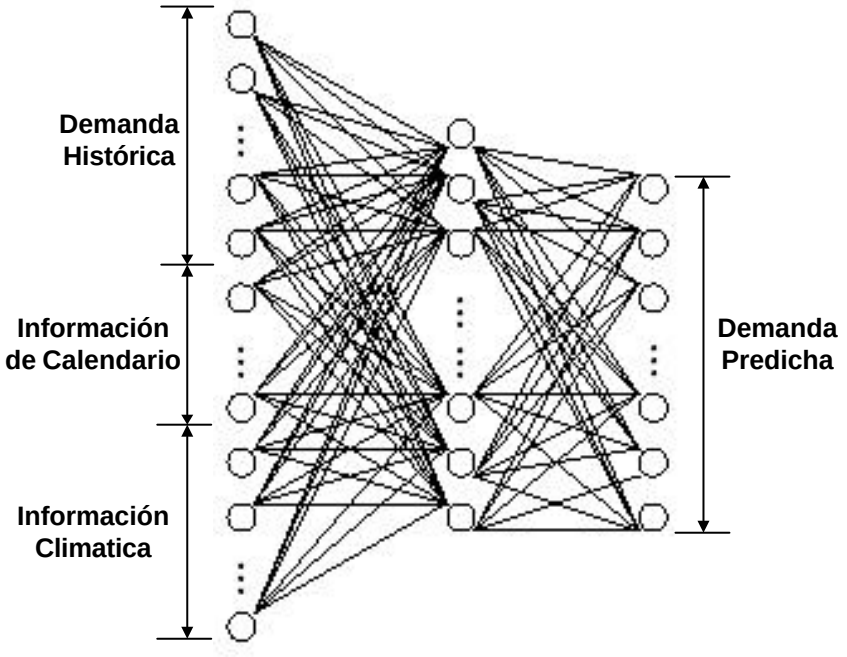
Otro de los aspectos importantes dentro de los esquemas de predicción, es la suposición de que la carga futura es dependiente en la carga del pasado y algunos factores externos como variables económicas, climáticas y culturales, que tuvieron también relación con ella, empleándose la RNA para aproximar esta dependencia.

Los factores económicos de influencia en la demanda tienen una mayor cabida en el pronóstico de largo plazo. Se deben principalmente a cambios en factores productivos como crecimiento en la industria, recesión económica, penetración en el mercado de energía eléctrica de algún sustituto como el gas, programas de gestión de la demanda, entre otros.

Los factores climáticos están mas relacionados con el pronóstico de la demanda de corto plazo. Algunos esquemas de predicción consideran efectos en la demanda tales como: Temperatura, humedad, velocidad del viento, duración del brillo solar y efectos estacionales (épocas de tormentas, nieve, verano). Infortunadamente, las condiciones climáticas son bastante específicas de una región y se requiere información actualizada específica de la zona de demanda, lo que imposibilita la aplicación de estas variables explicativas en modelo de predicción de la demanda con mayor grado de agregación, sin contar el problema de la disponibilidad de los datos.

Los factores culturales, están asociados principalmente al calendario laboral (días laborales, fines de semana y festivos). No obstante, algunos esquemas de predicción consideran efectos no comunes, como eventos deportivos, periodos vacacionales de los estudiantes, cambio de horario por estaciones, programas de televisión de gran impacto, entre otros. Una posible arquitectura que solucione el problema, se muestra a continuación.

Figura 46. Estructura típica de una RNA para predicción de demanda



En la figura 46 se presenta una arquitectura de red común a la mayoría de esquemas de predicción, encontrados en el estado del arte. Sería muy difícil ir en un mayor grado de detalle en esta estructura, ya que el problema de predicción varía ampliamente en el estado del arte. En este esquema se distinguen tres tipos de datos en los que se subdivide la capa de entrada: Datos de demanda históricos, Información del calendario e Información climática.

Por las características del problema, se considera la construcción de un modelo de

predicción de demanda como un problema de identificación de sistemas no lineales. No obstante, a diferencia de las metodologías clásicas de identificación de sistemas como ARIMA, o espacio de estados, el trabajo con redes neuronales adolece de procedimientos generales para construir a los modelos, siendo por lo general seleccionados de manera heurística.

Existen dos fases en toda aplicación de las redes neuronales: la *fase de aprendizaje o entrenamiento* y la *fase de prueba*. En la fase de entrenamiento, se usa un conjunto de datos o patrones de entrenamiento para determinar los pesos (parámetros de diseño) que definen el modelo neuronal. Por lo general el conjunto de datos empleado en la fase de entrenamiento se divide en un conjunto para el entrenamiento propiamente dicho y otro para efectos de validar el modelo durante el entrenamiento, incluidos dentro del algoritmo de aprendizaje.

Una vez entrenado este modelo, se usará en la llamada fase de prueba o funcionamiento directo, en la que se procesan los patrones de prueba que constituyen la entrada habitual de la red, analizándose de esta manera las prestaciones definitivas de la red.

En cuanto a la aplicación de las redes neuronales en el problema de la predicción del consumo eléctrico en el corto plazo, se encuentran en el estado del arte diferentes tipos de red, arquitecturas y algoritmos de aprendizaje empleados, de acuerdo a las características propias del problema, el cual puede consistir en la predicción de la demanda hora a hora de la potencia consumida durante el día, la energía total consumida en el día, la demanda máxima y mínima.

Una revisión del estado del arte muestra que la mayoría de trabajos en torno al tema han usado un perceptron multicapa feedforward como un aproximador de una relación del no lineal desconocida, pero con aspectos diferenciadores en el uso de los datos de tiempo, las otras variables de entrada, la arquitectura de la red, el algoritmo de entrenamiento, la selección y tratamiento de los datos de entrenamiento.

No obstante, dentro del estado del arte resumido en el numeral 1.2, se observan muchos tipos del esquemas para la solución del problema, como aquellos que aplican modelos híbridos con combinaciones de técnicas que emplean inteligencia artificial y/o técnicas convencionales de predicción, y dentro de aquellas que usan redes neuronales, es también común encontrar modelos que usan aprendizaje y redes recurrentes, por mencionar algunos ejemplos.

Un acercamiento común entre los esquemas de predicción encontrados en el estado del arte, es construir un sistema modular dónde se concentran las tareas específicas del proceso de predicción, como por ejemplo el tratamiento de los datos de entrada o la realimentación de errores en un sistema dinámico de predicción.

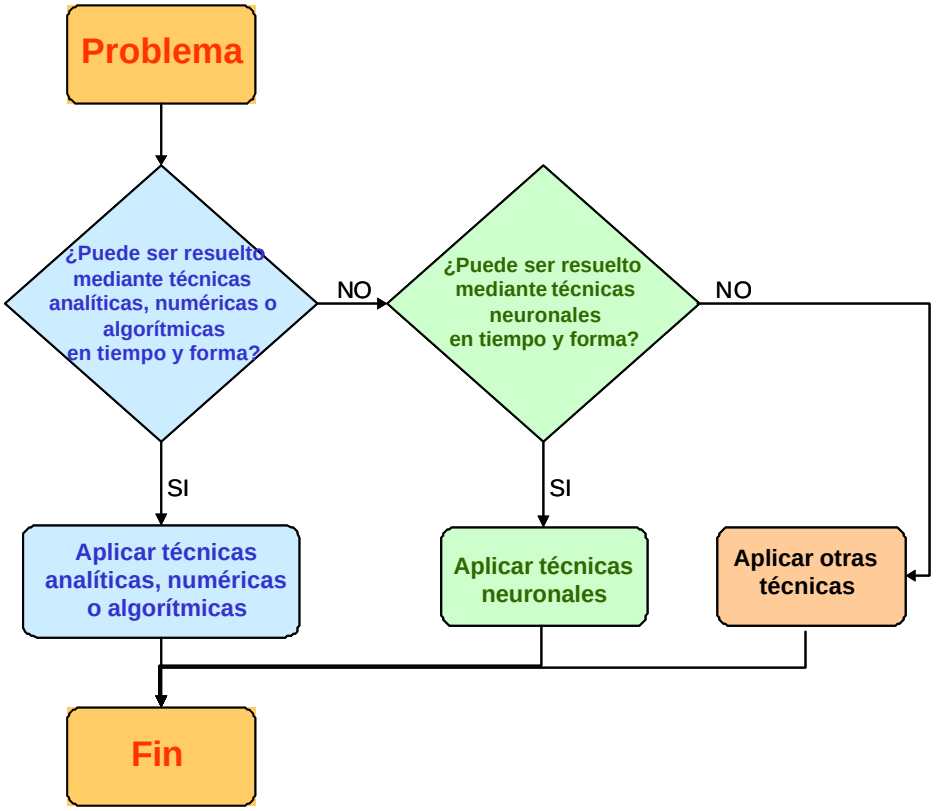
6. DESARROLLO METODOLÓGICO DEL SISTEMA DE PREDICCIÓN

Dentro de los objetivos planteados en este trabajo de investigación se proponen dos aspectos, considerados como los de mayor importancia en cuanto al aporte que podrían tener al desarrollo del estado del arte en el tema de la predicción de demanda en el corto plazo. Por un lado el estudio comparativo de las técnicas de predicción más comúnmente utilizadas en el horizonte del corto plazo, y por otro, una propuesta metodológica para la predicción de demanda mediante un esquema que emplea redes neuronales artificiales.

Estos dos aspectos se conjugan en la hipótesis de la propuesta de investigación, en el cual se plantea que no es necesario utilizar la red neuronal para predecir variables que mediante otras técnicas matemáticas y estadísticas más sencillas, pueden perfectamente ser extrapoladas. En otras palabras, la hipótesis principal del trabajo de investigación es que la red debe ser aprovechada para predecir la variable irregular (en donde las RNA's son más aventajadas), mientras que el ciclo y la tendencia, pueden ser extrapolados por aparte empleando técnicas de identificación de sistemas o algoritmos estadísticos. En la figura 47 se observa el planteamiento de solución al problema de predicción en el corto plazo.

Bajo este enfoque, el primer aspecto que se estudió en el estado del arte de la predicción de demanda de corto plazo, fueron las técnicas más comúnmente utilizadas (empleando modelos de identificación de sistemas como ARX, ARMAX, Box-Jenkins y Espacio de estados). Al verificar sus posibilidades de predicción se encuentran estimaciones de errores aceptables dentro del estado del arte, pero desventajas notables en los modelos relacionadas con el manejo de la información de entrada y salida, ya que estas matrices deben ser del mismo orden, además de las dificultades para considerar información de festivos dentro del modelo.

Figura 47. Enfoque de solución al problema de predicción



El segundo aspecto se centra en el desarrollo de un esquema de predicción basado en RNA's el cual, además de tener en cuenta las consideraciones generales anteriormente mencionadas en el capítulo 1, buscan corregir algunas deficiencias encontradas en los reportes de la bibliografía revisada sobre predicción de demanda mediante redes neuronales; estas deficiencias podrían resumirse en:

- Errores altos en los picos y valles de la curva de demanda: Entre los principales inconvenientes observados y no solucionados específicamente en las publicaciones internacionales revisadas, en lo referente a la predicción hora a hora de la demanda diaria de potencia, es el error relativamente alto en aquellas horas donde ocurre el pico y/o el valle de la curva de demanda,

imposibilitando además, el tener un resultado exacto para el dato de energía consumida durante el día.

- Distracción de la red en predecir información inoficiosa: Una tendencia generalizada en el empleo de técnicas no convencionales, es utilizarlas para simular comportamientos que pueden ser establecidos fácil y exactamente mediante un modelo matemático. Las redes neuronales presentan buenos resultados como aproximadoras de funciones matemáticamente desconocidas o predictoras de variables aleatorias, y es allí, donde se deben aprovechar las ventajas ofrecidas por ellas en los esquemas de predicción planteados.

- Inclusión de variables externas de poca influencia sobre la predicción: La inclusión de variables externas implica aumento en la complejidad del modelo de predicción. Además del tratamiento inicial de la información, es necesario establecer la influencia de las variables externas sobre la curva de demanda diaria, a fin de tomar una determinación de incluirlas o no, en el esquema de predicción, ya que a pesar de afectar en algún modo el comportamiento de la demanda, su efecto sobre el valor predicho puede ser despreciable, siendo mejor, asumirlo como parte de la irregularidad de la serie. Esta influencia de variables requiere especial atención en algunas variables de tipo climáticas, en las cuales los datos vienen dados en intervalos diarios, mientras que la demanda a predecir está dada por intervalos horarios. Otro aspecto relacionado con las variables externas, esta relacionado con la necesidad de predecir estas variables, dadas las condiciones de consecución tardía o incompleta de esta información.

- Vida útil corta del predictor en su etapa de producción: Una vez la red neuronal ha sido entrenada adecuadamente, y es puesta en producción (realización de predicciones), ella continua emulando valores con base en su entrenamiento, con lo cual la vida útil de la red estaría dada por el límite de error establecido. Considerando que las diferencias entre el valor de demanda

real y el predicho, están condicionadas a las variaciones en los hábitos de consumo y al crecimiento natural de la demanda, a medida que se avanza en el tiempo, los errores de predicción llegan a un punto donde empiezan a ser inaceptables, siendo entonces necesario un nuevo proceso de re-entrenamiento en el cual se tengan en cuenta los nuevos cambios en los patrones de consumo.

Debido a la dificultad para atacar matemáticamente el problema de encontrar una estructura de red neuronal óptima (tipo de red, arquitectura, algoritmo de entrenamiento, tratamiento de datos, etc.) para el caso en estudio, dada la complejidad y extensión de las relaciones matemáticas resultantes, el procedimiento que será llevado a cabo en este trabajo de investigación será de gran carácter heurístico. No obstante, el camino andado hasta el momento en el estado del arte, presenta algunas directrices a seguir en orden a establecer el procedimiento adecuado de experimentación y selección de la estructura adecuada para el problema como se plantea en [Haykin-1999] o [Becerra-1998], ya que el espectro de experimentación sería demasiado amplio como para ser abarcado en su totalidad.

La metodología propuesta en el trabajo de investigación busca desarrollar un esquema de predicción basado en RNAs, que cumpla con las consideraciones generales anteriormente mencionadas y solucione las deficiencias comunes encontradas en la revisión bibliográfica. Esta propuesta metodológica está basada en la técnica de descomposición de series de tiempo; en la cual, mediante la separación de las “fuerzas” que rigen el comportamiento de la serie de tiempo, se podrá aprovechar al máximo la capacidad de la red neuronal en la predicción de la componente irregular (aleatoria) de la curva de demanda, dejando las otras componentes para ser predichas por separado mediante la simple extrapolación de una función matemática cuya determinación es más sencilla. El enfoque de descomposición de la serie de tiempo propuesto en este trabajo, no ha sido aplicado en predicción de demanda mediante redes neuronales hasta el momento,

en lo que respecta a la bibliografía revisada. Sin embargo, a nivel hipotético, se considera como una buena opción para evitar que la red se “distraiga” prediciendo el ciclo (el cual es un patrón fácilmente representable y extrapolable) y mejor encamine su esfuerzo en predecir la irregularidad, que podría decirse al final de cuentas, es lo que diferencia la serie de tiempo entre un día y otro.

6.1 PROPUESTA METODOLÓGICA

Además de los planteamientos anteriores, la metodología debe considerar problemas implícitos dentro del proceso de predicción. A continuación se presentan algunas de las consideraciones básicas.

El empleo de un criterio para la selección de los datos base es importante si se considera que algunos datos pueden estar fuera de rango, ya sea por situaciones inesperadas en el sistema, como por ejemplo, fallos en líneas, cambios por reconfiguración de circuitos, desconexiones de carga, etc.; o por errores en la toma de datos o ausencia de los mismos (no se tomaron en determinada hora del día). La ausencia de datos o la presencia de datos atípicos debe ser mirada con cuidado, ya que de esto depende la exactitud y la capacidad de generalización del esquema de predicción.

Debido a que la demanda es altamente dependiente del tipo de día, es necesario una vez se tienen los datos depurados, establecer a que tipo de día pertenecen (sábado, domingo, festivo, día hábil). En este punto es importante destacar que el modelo a emplear debe resolver el problema de establecer cuales días son festivos (Fiestas religiosas, Fiestas patrias o lunes festivos por "Ley Emiliani"), siendo interesante el establecimiento de fiestas religiosas como Semana Santa, Corpus Cristi, ya que no son días fijos en el calendario. Para una mayor ilustración en el tema, se puede consultar el Anexo C.

Con los datos adecuados, se puede proceder a explorar los patrones de datos y a generar las series de tendencia, ciclo e irregularidad. Este paso comprende la observación previa de los datos, la comprensión que los datos sugieren y el uso de varios métodos gráficos para obtener una mejor visión de las causas que generaron los datos. Los procedimientos de análisis (descomposición) de la serie de tiempo se presentaron en el numeral 2.2.

La utilización de variables externas al consumo de energía, permite proponer modelos de predicción más exactos. El problema de identificar y encontrar relaciones funcionales que ayuden a inferir comportamientos de la demanda a partir de variables ambientales o temporales, consiste en primera instancia, en apropiarse de gran cantidad de información acerca de las posibles variables a considerar, en segunda, establecer cuales variables son las más representativas dentro del modelo formulado y finalmente incluirlas dentro del modelo de predicción de una forma adecuada.

La información considerada como variables ambientales, se refiere a datos climáticos tales como temperatura, humedad, brillo solar, etc. En cuanto a la información considerada como variables temporales, se pueden identificar tipos de días, festivos u otros acontecimientos como situaciones de contingencia, los cuales se pueden tratar a través de variables ficticias con valores 0 o 1 que indican la presencia o no del acontecimiento, variables indicadoras que de acuerdo a un valor (puede ser también entre 0 y 1) definen un estado entre múltiples posibles.

En el caso del metodología planteada, el modelo se plantea como autorregresivo, es decir, las predicciones toman como datos de entrada la historia de la serie de tiempo y no las variables explicativas. No obstante, se adiciona a la información de entrada el tipo de día según el calendario. Las razones por las cuales no se toman las variables climáticas ya fueron establecidas en el numeral 6.1.

Partiendo de la premisa de que las redes neuronales poseen ventajas favorables para el desarrollo de problemas de identificación, modelado y predicción de

señales con alto contenido aleatorio, se ha planteado el predictor como una red neuronal dentro de la cual deben existir los procesos de entrenamiento, validación y predicción en sí. Esto implica además un sistema de pre-procesamiento de la información que permita que los datos de irregularidad estén suavizados, normalizados, y en la forma matricial adecua para que se incluya la información del tipo de día según el calendario.

Una vez se ha establecido el esquema del predictor se deberá considerar la estructura y características de la red neuronal a emplear. Como se presentó en el numeral 5.2, las opciones en cuanto a arquitectura de red a emplear y algoritmos de entrenamiento son diversas. No obstante, las opciones de estudio en cuanto a arquitecturas más empleadas son el Perceptrón Multicapa y las Redes de Elman [Elman-1990].

Con respecto al algoritmo de entrenamiento, se empleará el Back propagation teniendo como alternativas todas sus variantes tales como el método del momentum, tasa de aprendizaje adaptativa, método del gradiente, regla delta, algoritmo de correlación en cascada, etc. Como elemento adicional, también será necesario definir número de capas internas, número de neuronas por capa, tipo de función de activación, y conectividad entre neuronas, ya sea recurrente [Vermaak-1998] o hacia adelante (feed-forward) [Tang-1991].

Para definir los parámetros anteriores se empleará el método heurístico, ya que una de las principales desventajas de estos modelos, es precisamente su dificultad para determinar de una manera matemática estos parámetros.

Una vez se ha escogido la arquitectura y el algoritmo de entrenamiento adecuado, surgen dos interrogantes: ¿Cuándo es adecuado detener el proceso de entrenamiento?, y ¿Cuáles resultados se pueden considerar como aceptables?. Para solucionarlos se deberán establecer los tipos de error y límites de confiabilidad de los mismos. Por lo general, la red neuronal puede obtener errores muy bajos en el entrenamiento, pero sacrifica su capacidad de generalización ante

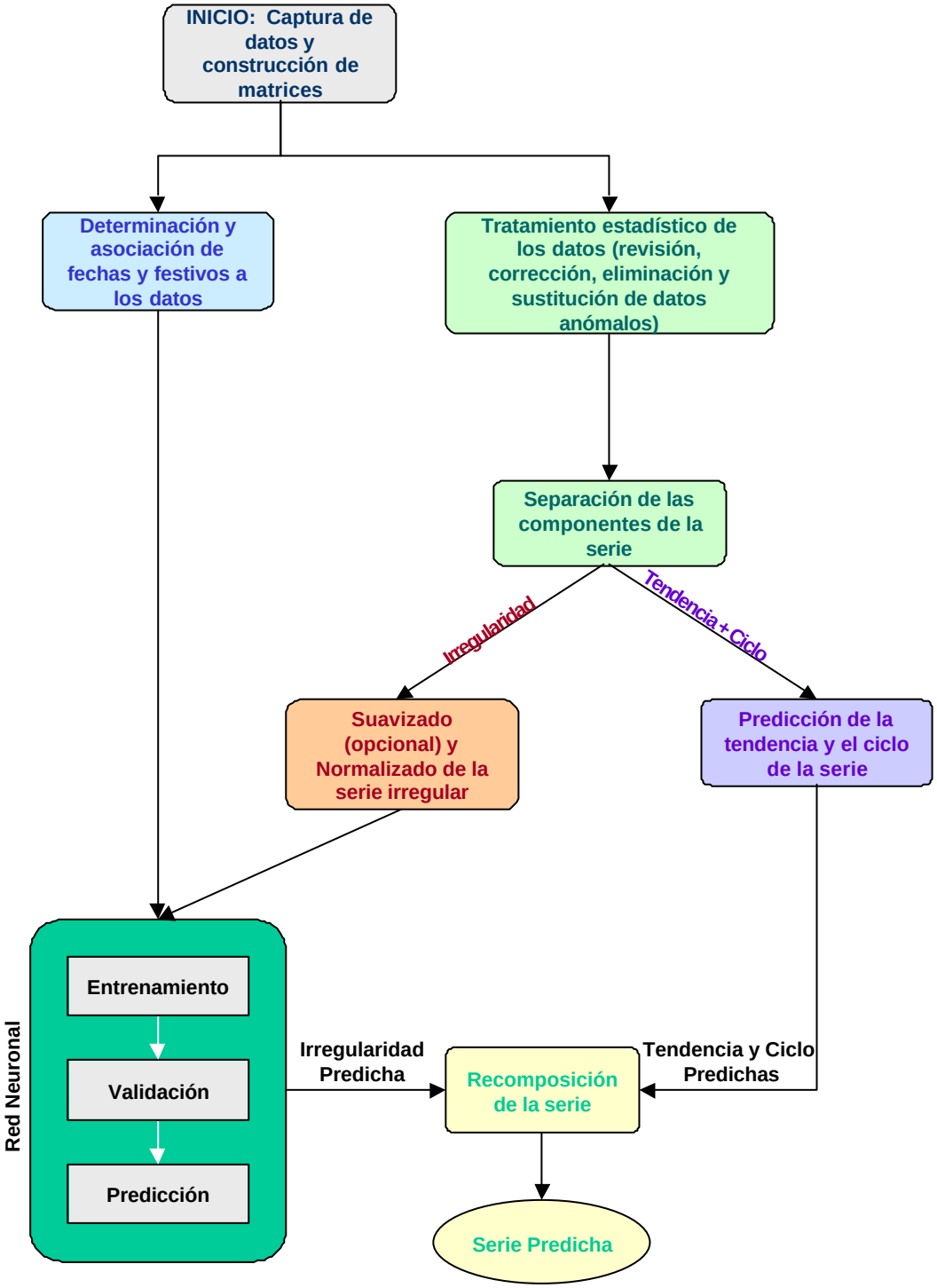
nuevas situaciones al sobre-entrenarse para los casos mostrados como ejemplo en el aprendizaje; cuando ello sucede, se obtienen errores altos en la validación de resultados. Por lo tanto, es necesario encontrar el punto óptimo en el cual se presenten, tanto mínimos errores posibles de entrenamiento, como de validación. Planteamientos como los propuestos en [Haykin-1999] pueden ser de gran utilidad en la solución de este problema.

Debido a que la red neuronal sólo predice la irregularidad normalizada de la serie de tiempo, es necesario "desnormalizar" el resultado, es decir volver la irregularidad a sus valores base, y por otro camino, predecir la tendencia y el ciclo, a fin de sintetizar (recomponer) la serie de tiempo y presentar el resultado, el cual será evaluado tomando como base las perspectivas de error dentro de un proceso de validación. En la página siguiente se puede observar el planteamiento metodológico para el problema de predicción de demanda de corto plazo.

6.2 APLICACIÓN DE LA METODOLOGÍA EN UNA HERRAMIENTA SOFTWARE

Con el objeto de probar la metodología propuesta se diseñó la herramienta software "PREDICCIÓN", la cual se desarrolló para trabajar en el entorno del mercado eléctrico colombiano (considera festivos, predicción de la semana y elimina la estacionalidad de la serie de tiempo). No obstante, con ligeras modificaciones de forma, en cuanto a la presentación de los resultados, "PREDICCION" permite gracias a su planteamiento generalizado, realizar pronósticos de cualquier serie de tiempo relacionada con demanda horaria (Consumo de agua, Gas Natural, y otros servicios).

Figura 48. Esquema de la metodología de predicción propuesta



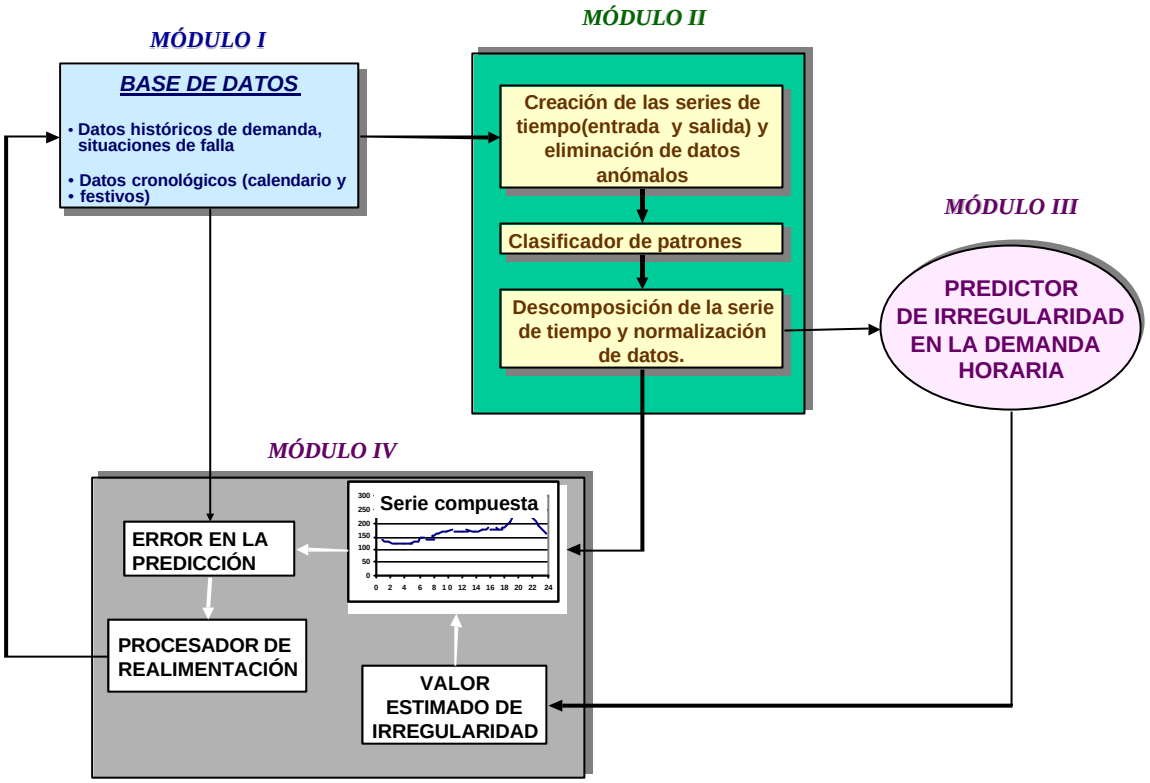
Dentro de las consideraciones de diseño de la herramienta software, se tuvieron en cuenta además de la metodología propuesta, aspectos de comodidad para el usuario de la misma, como por ejemplo la interfaz gráfica, la posibilidad de capturar los datos desde una hoja de calculo tipo Excel y de obtener resultados en el mismo formato, el planteamiento de la herramienta acorde con los lineamientos de la CREG en torno a la información semanal que deben suministrar los comercializadores para efectos del despacho al CND, y las posibilidades de manejo sencillo y graficación de resultados, para los procesos captura de datos, entrenamiento y predicción.

También se consideró en el diseño de la herramienta, la flexibilidad de la misma a fin de encontrar aquel modelo que mejor se adapte a las características de la demanda a predecir. Dentro de estos aspectos de flexibilidad, se puede adaptar el tipo de tendencia (o dejar que la herramienta escoja la mejor mediante un análisis de correlación), seleccionar todos los parámetros de entrenamiento de la red neuronal como por ejemplo el tipo de red, algoritmos de entrenamiento, número máximo de épocas, tasas de aprendizaje, número de capas, neuronas por capa entre otros; o también, dejarla que entrene en forma automática, o realizar pruebas con todas las posibilidades a fin de seleccionar la mejor.

Para una mejor orientación sobre el uso, instrucciones de operación, secuencia de procesos, nombres de los archivos empleados y posibilidades de trabajo, se presenta en el anexo A el manual del usuario de la herramienta software "PREDICCIÓN".

En la figura se plantea un diagrama modular de la herramienta, en la cual se aprecia el enfoque metodológico anteriormente expuesto. Este esquema se basa en cuatro partes principales: Una base de datos, un módulo de pre-procesamiento de información, un módulo de predicción y un módulo de salida.

Figura 49. Esquema modular de la herramienta software



En el módulo I o base de datos, se encuentra toda la información necesaria para el entrenamiento de la red y los procesos necesarios para la captura de información e interfaz con el usuario. Esta información la conforman los datos históricos existentes demanda y calendario, y es actualizada con la información real una vez ha sido obtenida, o si es el caso, con información predicha en la circunstancia de que no se tenga información y sea necesario re-entrenar la red por ineficientes errores obtenidos.

El módulo II o módulo de pre-procesamiento de información, es quizás el aspecto más importante del esquema de predicción planteado; este módulo maneja la información que se va a ingresar a la red en el proceso de entrenamiento. Como la información que se encuentra en la base de datos no es una información depurada y pertinente, debe realizarse un proceso de análisis estadístico, a fin de

encontrar valores fuera de rango o discontinuidades en la serie de tiempo.

Adicionalmente es necesario descomponer las series y normalizar los datos, evitando que la red neuronal encargada de la predicción sea de un tamaño considerable, al manejar excesivo número de variables y, a través de él, se pretende encontrar relaciones funcionales que eviten la distracción de la red neuronal. Cuando ha sido realizada la predicción, es necesario realizar los procesos inversos, es decir recomponer serie y encontrar magnitudes base ("desnormalizar").

El tercer módulo (predictor), está encargado de realizar la predicción en sí. Se desea que este módulo sea de antemano una red neuronal sencilla, que esté encargada de predecir la irregularidad de la serie de tiempo (y no la serie de tiempo completa normalizada (***Y(t)/k***), la cual es en realidad lo que no se puede modelar matemáticamente de una forma sencilla por su alta componente aleatoria. En el planteamiento de esta red neuronal se dedicarán esfuerzos para determinar la arquitectura de red adecuada para la cantidad y tipo de información suministrada. Para esto será importante además de las experiencias internacionales los resultados obtenidos en los trabajos realizados en la universidad como los de [Becerrra-1998], [Alvarado-1998] y [Portela-2000] en donde se hacen primeros avances en nuestro contexto acerca de una arquitectura de red adecuada para este problema.

Para finalizar, el módulo de salida será el encargado de sintetizar (volver a reunir ciclo e irregularidad) la serie de tiempo, calcular y mostrar resultados de demanda predicha en MW, errores de predicción y servir de puente para reentrenar la red en caso de no ser aceptables los errores encontrados.

A continuación se describe el proceso que sigue la herramienta para realizar la predicción.

1. Mediante un procedimiento soportado en Delphi se inicia MATLAB y se ejecutan los procesos de captura de datos (buscar y abrir la hoja de cálculo en Ecxel donde se encuentre la información), permitiendo de esta manera copiar los datos e ingresarlos a MATLAB.

2. Luego de copiados los datos provenientes de Microsoft Excel (u otro programa de características similares) en el portapapeles, son revisados estos en busca de separadores de caracteres (indicadores de punto decimal y miles), siendo removidos.

3. Posteriormente los datos son llevados a los archivos datosentre o datosentrena de acuerdo al proceso realizado. Adicionalmente a los archivos correspondientes a los datos, la interfase crea otros ficheros de texto con información acerca de la fecha de inicio de los datos, sus características, etc. cuyos nombres y finalidad pueden ser consultados en el anexo A.

4. Los archivos creados por la interfase son accedidos desde los scripts de MATLAB. Entre la información obtenida se encuentra la existencia o no de las fechas para los datos; si estas están en los datos se remueve la columna correspondiente a ellas, en caso contrario son determinadas mediante la utilización del inicio de los datos y la cantidad de días asociados a los datos.

5. Conocida la fecha inicial de ella se extrae el año dato clave para determinar la ubicación de la pascua, la metodología utilizada se presenta en le anexo C. Ubicada la pascua pasa a determinar el resto de la Semana Santa y las demás fiestas litúrgicas de acuerdo al calendario litúrgico (dichas fiestas tienen una separación definida dentro de dicho calendario).

6. Determinadas las fiestas móviles del calendario litúrgico se procede a ubicarlas en el calendario colombiano, verificando su ubicación en la semana a fin de colocar las fiestas de acuerdo a la Ley Emiliani (Lunes festivos).

7. Determinada la posición “real” de las fiestas móviles se anexan las fiestas fijas, por ejemplo, el 20 de Julio, armando una base de datos de las fiestas nacionales pasando a comparar dicha información con la información obtenida para las fechas de los datos.

8. Si se encuentran coincidencias de las fechas festivas con los Lunes se coloca un peso igual a 0,25 en caso contrario corresponde a 1 (día hábil), posteriormente se verifica cuales son sábados y domingos para ubicar los pesos correspondientes (0,75 y 0,5 respectivamente).

9. A continuación se normalizan los datos, de acuerdo a sus unidades (en caso de agregar datos con diferentes unidades kW, MW o GW); se determina la media por hora y unas cotas (máximas y mínimas, correspondientes a 2,5 desviaciones estándar). Si algún dato se ubica fuera de las cotas es reemplazado por la media de la hora.

10. Seguidamente se dividen dichos datos en dos partes: 80% para entrenamiento y 20 % para validación. A fin de asegurar que los datos a llevar a la RNA correspondan a semanas completas a partir de la última fila de los datos se toman grupos de datos correspondientes a semanas determinando la fecha inicial, de acuerdo a dicha fecha se establece el vector de días de la semana correspondiente a los datos usados, tanto los datos como los pesos de cada grupo (entrenamiento y validación).

11. Se inicia entonces la descomposición de las series; para ello el ciclo se obtiene mediante un filtro de primer orden, retirándose dicha componente mediante la razón serie/ciclo. Para obtener la tendencia los datos obtenidos de la anterior relación se ajustan a tres modelos; lineal, cuadrático y exponencial usando la norma de dichos ajustes para determinar cual modelo se aproxima más a los datos, dicha componente se extrae de los datos mediante la razón TI/Tendencia quedando la irregularidad (la estacionalidad no se considera pues dicha componente no presenta mayor incidencia).

12. Para la irregularidad se han considerado dos opciones suavizarla o no, para el suavizado se utiliza una modificación del algoritmo de suavizado exponencial desarrollada por Brown (la modificación consiste en un aumento en 48 unidades de la constante alfa; originalmente correspondía a 2), adicionalmente permite la revisión de las graficas resultado de la descomposición, salvar los datos hasta el momento e introducir la información para la creación de la RNA. Dicha irregularidad será la variable de entrada llevada a la RNA;

13. A fin de asegurar que la irregularidad se halle en el rango [-1 1] antes de ser ingresada al entrenamiento de la RNA, dicha irregularidad se normaliza, mediante un procedimiento establecido en MATLAB para tal fin.

14. La estructura de la capa oculta de la RNA es determinada por el usuario siendo esta 1-N-1, donde N es el número de capas internas. Para la inicialización de las conexiones se utilizó el algoritmo de Nguyen Widrow a fin de maximizar la utilización de las neuronas pertenecientes a la capa oculta definidas por el usuario.

15. Durante la fase de entrenamiento la RNA recibe como entrada el 80% de los datos de entrenamiento y como objetivo el 20% de los mismos; dichas entradas incorporan como columna adicional la información correspondiente al tipo de días existentes en la semana.

16. A fin de verificar el comportamiento de la RNA se compara la composición de la serie utilizando como irregularidad la salida de la RNA (posteriormente a su "desnormalización"), la proyección de la tendencia y el ciclo correspondiente estos a los datos de entrenamiento frente a la última semana de los datos utilizados como objetivos.

17. Para la comparación se realizan mediciones de error (Raíz del error medio cuadrado, Desviación de la media absoluta y el porcentaje del error medio absoluto), mostrándolas al usuario mediante la interfaz.

18. Para la prueba de la RNA se agrega a los datos de irregularidad (80% entrenamiento) la porción separada al grupo inicial de datos (20% validación), se ajusta el tamaño de la ventana de datos a introducir a la RNA, se simula el comportamiento y se verifica respecto al dato de entrada de la última semana de los datos de validación. Realiza medidas del error y las presenta al usuario mediante la interfaz.

19. La herramienta permite reentrenar presentando dos opciones: modificar la estructura de la RNA o agregar nuevos datos; si se escoge la última los datos reciben un tratamiento similar al explicado anteriormente.

20. Antes de iniciar la predicción presenta los errores obtenidos con la RNA actual. Como opciones en el proceso se puede predecir la semana siguiente para lo cual toma el conjunto de datos utilizados en la prueba incluyendo la última semana predicha en dicho proceso y genera la salida; dicha salida se compone con el ciclo y la proyección de la tendencia de los datos de validación. En caso de elegir "otro" preguntará el avance en semanas prediciendo el modelo una semana, anexa la predicción a los datos y remueve los datos de la primera fila hasta completar el avance indicado por el usuario.

21. Terminada la predicción pasa a la proyección de la tendencia; para ella se han determinado los coeficientes que más se ajustan al comportamiento de los datos y conocido el avance en semanas, se determina el comportamiento de la tendencia y se compone la serie a presentar al usuario.

6.3 RESULTADOS OBTENIDOS CON LA HERRAMIENTA SOFTWARE

Con el objeto de comprobar la metodología planteada y establecer parámetros para el entrenamiento y predicción de la red neuronal, se realizaron pruebas con la demanda agregada de la ESSA de usuarios regulados.

Inicialmente se realizaron pruebas con gran cantidad de datos de la demanda regulada de la ESSA desde 1 de Junio de 2000 hasta 31 de Agosto de 2002, es decir, 822 días x 24 horas = 19728 datos. Se dispuso realizar pruebas con todas las combinaciones de tipo de red que ofrece el programa, pero restringiendose a máximo tres capas de 15 neuronas cada una :

Tipo de red: Feed Forward, Elman y Correlación en Cascada.

Algoritmo de entrenamiento: Gradiente descendente, Levenberg Marquardt, Gradiente descendente con momento y back resilente.

Reglas de aprendizaje: Back propagation y Back porpagation con momento.

Dado el gran número de combinaciones posibles y la cantidad de datos, las pruebas duraron alrededor de 1 semana de trabajo continuo del computador.

En las siguientes tablas se presentan los mejores resultados obtenidos en la prueba clasificados por número de capas (3,2 y 1 capa de máximo 15 neuronas). A fin de establecer una estructura óptima se calculó la desviación estándar del número de neuronas por capa su mediana (aproximándose a una entero).

A nivel general se observa que los errores se mejoran considerablemente cuando la serie se suaviza exponencialmente, tanto así que ninguna serie sin suavizar quedó en las mejores 40 redes.

Tabla 4. Selección de 40 mejores redes de 1 capa

Entrenada el	Tipo de red	Función de entrenamiento	Función de aprendizaje	Numero de capas	Neuronas por capa 1	MAD en validación (kW)	RMSE en validación (kW)	MAPE en validación (%)	Cantidad de épocas empleadas
15/12/2003 0:36	newff	trainlm	learngd	1	4	4609,2559	6192,1989	2,8461	21
15/12/2003 0:36	newff	trainlm	learngd	1	4	4609,2559	6192,1989	2,8461	21
19/12/2003 10:25	newelm	traingd	learngd	1	4	6071,1323	8070,9936	3,8766	3000
22/12/2003 19:13	newelm	traingd	learngdm	1	3	6085,3424	8125,6567	3,8867	3000
26/12/2003 4:04	newelm	trainlm	learngd	1	3	6093,4726	8017,7809	3,899	8
26/12/2003 4:03	newelm	trainlm	learngd	1	2	6062,48	7733,0158	3,9097	12
22/12/2003 19:14	newelm	traingd	learngdm	1	4	6140,5669	8097,6334	3,9129	3000
26/12/2003 4:04	newelm	trainlm	learngd	1	4	6119,0791	7974,7245	3,9168	8
26/12/2003 4:19	newelm	trainlm	learngd	1	15	6152,2158	8140,5459	3,9209	16
26/12/2003 4:04	newelm	trainlm	learngd	1	6	6188,5298	8117,1486	3,9513	8
19/12/2003 10:27	newelm	traingd	learngd	1	7	6219,7516	8132,3277	3,9605	3000
26/12/2003 4:03	newelm	trainlm	learngd	1	1	6275,1741	8370,3607	3,9785	23
26/12/2003 4:04	newelm	trainlm	learngd	1	5	6248,5737	8023,6609	3,9864	7
19/12/2003 10:32	newelm	traingd	learngd	1	14	6279,9994	8240,7204	3,9871	3000
26/12/2003 4:08	newelm	trainlm	learngd	1	11	6300,3543	8421,3941	3,9965	9
22/12/2003 19:17	newelm	traingd	learngdm	1	9	6315,2979	8320,2822	4,0055	3000
19/12/2003 10:24	newelm	traingd	learngd	1	3	6324,47	8260,8001	4,0151	3000
22/12/2003 19:12	newelm	traingd	learngdm	1	1	6370,4189	8395,7755	4,0231	3000
26/12/2003 4:04	newelm	trainlm	learngd	1	7	6365,689	8496,3269	4,0326	36
22/12/2003 19:14	newelm	traingd	learngdm	1	5	6384,0845	8569,4938	4,0371	3000
22/12/2003 19:16	newelm	traingd	learngdm	1	7	6367,4007	8221,2726	4,0475	3000
22/12/2003 19:19	newelm	traingd	learngdm	1	12	6399,1955	8387,1701	4,0521	3000
19/12/2003 10:30	newelm	traingd	learngd	1	11	6444,0306	8560,0308	4,0711	3000
22/12/2003 19:16	newelm	traingd	learngdm	1	8	6428,8408	8544,4768	4,0719	3000
26/12/2003 4:07	newelm	trainlm	learngd	1	10	6495,2233	8769,423	4,0839	12
26/12/2003 4:12	newelm	trainlm	learngd	1	13	6476,2162	8606,4276	4,0857	15
22/12/2003 19:19	newelm	traingd	learngdm	1	11	6473,5457	8515,5024	4,0897	3000
19/12/2003 10:28	newelm	traingd	learngd	1	9	6452,2532	8444,8093	4,0918	3000
22/12/2003 19:20	newelm	traingd	learngdm	1	13	6465,2253	8597,5242	4,0924	3000
19/12/2003 10:30	newelm	traingd	learngd	1	12	6447,5047	8386,2531	4,0933	3000
22/12/2003 19:21	newelm	traingd	learngdm	1	14	6468,74	8508,6212	4,0996	3000
19/12/2003 10:33	newelm	traingd	learngd	1	15	6496,9686	8454,2916	4,1121	3000
22/12/2003 19:18	newelm	traingd	learngdm	1	10	6548,2351	8653,7991	4,1338	3000
22/12/2003 19:15	newelm	traingd	learngdm	1	6	6563,2421	8908,7745	4,1359	3000
26/12/2003 4:11	newelm	trainlm	learngd	1	12	6630,7377	8848,2478	4,1759	10
26/12/2003 4:06	newelm	trainlm	learngd	1	9	6660,0622	8856,7971	4,1886	9
22/12/2003 19:22	newelm	traingd	learngdm	1	15	6661,8751	8845,3959	4,1991	3000
19/12/2003 10:31	newelm	traingd	learngd	1	13	6645,3963	8513,7658	4,2158	3000
19/12/2003 10:29	newelm	traingd	learngd	1	10	6714,1248	8700,6658	4,2579	3000
26/12/2003 4:05	newelm	trainlm	learngd	1	8	6759,3263	8947,8317	4,2655	9

desviación estándar: 4,235 (capa1)

mediana: 9

promedio: 8,25

Tabla 5. Selección de 40 mejores redes de 2 capas

Entrenada el	Tipo de red	Función de entrenamiento	Función de aprendizaje	Numero de capas	Neuronas por capa 1	Neuronas por capa 2	MAD en validación (kW)	RMSE en validación (kW)	MAPE en validación (%)	Cantidad de épocas empleadas
15/12/2003 9:02	newff	trainlm	learngd	2	8	14	4508,228	6151,3408	2,7759	42
09/12/2003 12:04	newelm	trainlm	learngd	2	9	8	4552,8152	6320,8256	2,7856	26
15/12/2003 22:17	newff	trainlm	learngd	2	15	4	4562,3627	6226,3182	2,8069	22
09/12/2003 12:27	newelm	trainlm	learngd	2	9	10	4590,3946	6289,926	2,8194	13
15/12/2003 16:57	newff	trainlm	learngd	2	12	11	4607,9252	6300,7153	2,8338	29
15/12/2003 7:09	newff	trainlm	learngd	2	6	8	4602,655	6249,8911	2,8346	18
15/12/2003 7:09	newff	trainlm	learngd	2	6	8	4602,655	6249,8911	2,8346	18
09/12/2003 6:50	newelm	trainlm	learngd	2	6	10	4625,1463	6338,9961	2,836	21
09/12/2003 6:46	newelm	trainlm	learngd	2	6	9	4608,3201	6230,0997	2,8416	31
10/12/2003 8:17	newelm	trainlm	learngd	2	14	6	4608,7863	6195,5698	2,8473	22
09/12/2003 18:31	newelm	trainlm	learngd	2	10	15	4680,2732	6566,1506	2,8576	17
15/12/2003 8:13	newff	trainlm	learngd	2	8	2	4648,5485	6277,8004	2,8644	35
15/12/2003 8:13	newff	trainlm	learngd	2	8	2	4648,5485	6277,8004	2,8644	35
15/12/2003 1:51	newff	trainlm	learngd	2	1	2	4708,6255	6339,0069	2,8988	39
09/12/2003 14:17	newelm	trainlm	learngd	2	10	1	4726,8319	6478,5975	2,9	73
09/12/2003 23:19	newelm	trainlm	learngd	2	12	8	4700,8969	6271,6027	2,9058	19
26/12/2003 7:37	newelm	trainlm	learngd	2	11	2	5926,7296	7743,8257	3,8202	27
19/12/2003 14:04	newelm	traingd	learngd	2	15	4	6000,6298	7933,0093	3,8422	3000
26/12/2003 5:22	newelm	trainlm	learngd	2	6	10	6011,851	7890,5268	3,8486	26
26/12/2003 4:53	newelm	trainlm	learngd	2	4	13	6045,6426	8038,4223	3,8545	13
19/12/2003 13:45	newelm	traingd	learngd	2	14	3	6060,4456	8042,7282	3,8588	3000
26/12/2003 5:44	newelm	trainlm	learngd	2	7	9	6043,556	7930,9006	3,8632	16
26/12/2003 5:29	newelm	trainlm	learngd	2	6	14	6044,8812	8026,5782	3,864	16
19/12/2003 11:58	newelm	traingd	learngd	2	7	6	6018,5437	7850,0149	3,868	3000
22/12/2003 20:42	newelm	traingd	learngdm	2	7	2	6010,8686	7626,3787	3,8735	3000
22/12/2003 22:43	newelm	traingd	learngdm	2	14	8	6063,9395	7951,9397	3,8738	3000
26/12/2003 4:48	newelm	trainlm	learngd	2	4	8	6066,8017	8173,5815	3,875	17
26/12/2003 4:49	newelm	trainlm	learngd	2	4	10	6085,0162	8030,3861	3,8838	14
26/12/2003 5:41	newelm	trainlm	learngd	2	7	7	6099,9764	8273,6066	3,8839	21
26/12/2003 4:29	newelm	trainlm	learngd	2	2	2	6041,1495	7739,1062	3,8841	12
19/12/2003 11:12	newelm	traingd	learngd	2	4	1	6009,7987	7657,2896	3,8841	3000
22/12/2003 22:35	newelm	traingd	learngdm	2	14	1	6116,207	8266,5921	3,8902	3000
22/12/2003 20:17	newelm	traingd	learngdm	2	5	6	6092,729	8045,1235	3,8946	3000
26/12/2003 4:59	newelm	trainlm	learngd	2	5	7	6090,0143	7963,497	3,8951	16
19/12/2003 11:59	newelm	traingd	learngd	2	7	7	6129,4164	8124,6319	3,8953	3000
26/12/2003 11:35	newelm	trainlm	learngd	2	15	14	6077,9982	7931,9564	3,8958	23
22/12/2003 19:51	newelm	traingd	learngdm	2	3	6	6050,6463	7665,7064	3,8965	3000
22/12/2003 20:19	newelm	traingd	learngdm	2	5	8	6127,3207	8094,3067	3,8987	3000
19/12/2003 11:34	newelm	traingd	learngd	2	5	11	6110,4413	8071,5118	3,9003	3000
22/12/2003 22:36	newelm	traingd	learngdm	2	14	2	6114,7976	8098,7749	3,9005	3000

desviación estándar:	3,975 (capa1)	4,085 (capa2)
mediana	7	8
promedio	8,125	6,975

Tabla 6. Selección de 40 mejores redes de 3 capas

Entrenada el	Tipo de red	Función de entrenamiento	Función de aprendizaje	Numero de capas	Neuronas por capa 1	Neuronas por capa 2	Neuronas por capa 3	MAD en validación (kW)	RMSE en validación (kW)	MAPE en validación (%)	Cantidad de épocas empleadas
17/12/2003 3:34	newff	trainlm	learngd	3	3	12	13	4533,244	6275,261	2,7739	24
16/12/2003 22:55	newff	trainlm	learngd	3	3	3	5	4527,1558	6205,4205	2,7816	28
16/12/2003 7:59	newff	trainlm	learngd	3	1	14	12	4539,1955	6196,2538	2,7858	18
16/12/2003 22:36	newff	trainlm	learngd	3	3	2	12	4515,198	6142,35	2,7872	30
17/12/2003 9:52	newff	trainlm	learngd	3	4	5	8	4554,0077	6284,4569	2,7893	86
16/12/2003 20:18	newff	trainlm	learngd	3	2	14	7	4557,4457	6222,4853	2,8015	42
16/12/2003 19:06	newff	trainlm	learngd	3	2	12	7	4549,4944	6157,6985	2,8024	37
16/12/2003 5:18	newff	trainlm	learngd	3	1	12	7	4573,8354	6223,7465	2,8059	69
16/12/2003 1:05	newff	trainlm	learngd	3	1	2	3	4590,7978	6325,6726	2,8131	25
17/12/2003 5:42	newff	trainlm	learngd	3	3	15	4	4571,9593	6201,4249	2,8203	15
16/12/2003 3:54	newff	trainlm	learngd	3	1	10	5	4555,5882	6129,2693	2,8208	104
16/12/2003 1:13	newff	trainlm	learngd	3	1	2	14	4621,9424	6242,2404	2,8429	54
16/12/2003 13:57	newff	trainlm	learngd	3	2	3	14	4608,5694	6208,7024	2,8469	15
16/12/2003 6:53	newff	trainlm	learngd	3	1	13	15	4625,5125	6233,3498	2,8471	17
17/12/2003 2:12	newff	trainlm	learngd	3	3	10	11	4629,1629	6268,2906	2,8481	18
16/12/2003 5:14	newff	trainlm	learngd	3	1	12	5	4612,4029	6151,4854	2,8551	15
17/12/2003 6:47	newff	trainlm	learngd	3	4	1	5	4625,7348	6204,5398	2,8577	34
17/12/2003 5:09	newff	trainlm	learngd	3	3	14	11	4636,4758	6250,1871	2,8603	19
27/12/2003 17:09	newelm	trainlm	learngd	3	5	9	4	5881,7722	7878,1475	3,7679	16
28/12/2003 9:39	newelm	trainlm	learngd	3	7	10	15	5866,3514	7523,4575	3,7907	21
27/12/2003 21:41	newelm	trainlm	learngd	3	5	15	15	5967,2676	8141,5361	3,804	43
25/12/2003 16:53	newelm	traingd	learngdm	3	14	3	3	5955,8199	7882,4913	3,8047	3000
27/12/2003 23:15	newelm	trainlm	learngd	3	6	6	11	5976,5118	8042,1777	3,8083	22
27/12/2003 0:58	newelm	trainlm	learngd	3	3	1	4	5957,824	7943,6745	3,8131	20
26/12/2003 20:15	newelm	trainlm	learngd	3	2	5	10	5931,0879	7701,0899	3,8132	18
27/12/2003 2:57	newelm	trainlm	learngd	3	3	10	10	5958,4372	8051,6438	3,814	16
27/12/2003 1:19	newelm	trainlm	learngd	3	3	3	13	5955,67	8028,9771	3,8164	32
21/12/2003 17:23	newelm	traingd	learngd	3	11	9	4	5959,3115	7850,9225	3,8181	3000
26/12/2003 13:55	newelm	trainlm	learngd	3	1	8	10	5909,3312	7524,1805	3,8191	37
26/12/2003 7:37	newelm	trainlm	learngd	2	11	2	0	5926,7296	7743,8257	3,8202	27
27/12/2003 19:18	newelm	trainlm	learngd	3	5	13	7	5951,2871	7671,7151	3,8255	28
21/12/2003 3:48	newelm	traingd	learngd	3	8	15	7	5983,3754	7941,567	3,8268	3000
21/12/2003 11:57	newelm	traingd	learngd	3	10	9	6	5995,2272	8092,0024	3,8271	3000
19/12/2003 15:31	newelm	traingd	learngd	3	1	5	13	6010,0431	8151,0652	3,8275	3000
23/12/2003 15:02	newelm	traingd	learngdm	3	4	10	5	5976,3708	7880,0595	3,8291	3000
22/12/2003 7:48	newelm	traingd	learngd	3	14	2	2	5946,9573	7538,0486	3,8317	3000
22/12/2003 23:49	newelm	traingd	learngdm	3	1	3	12	6003,8746	7991,0159	3,8318	3000
19/12/2003 15:24	newelm	traingd	learngd	3	1	5	7	5986,2377	7890,2599	3,8327	3000
21/12/2003 13:47	newelm	traingd	learngd	3	10	14	2	6001,4641	7987,9906	3,8372	3000
28/12/2003 19:55	newelm	trainlm	learngd	3	8	7	1	5962,3936	7628,967	3,8375	56

desviación estándar:	3,715 (capa1)	4,728 (capa2)	4,341 (capa3)
mediana:	3	9	7
promedio:	4,301	8,100	7,975

Adicionalmente se observa que la red tipo Elman (newelm) da mejores resultados con pocas capas mientras que la feed-forward da los mejores resultados en tres capas, aunque es muy similar el resultado de ambos tipos.

También se puede observar que en cuanto a los algoritmos de entrenamiento el Gradiente descendente y Levenberg Marquardt son los de mejores resultados, ya que los otros algoritmos no alcanzan a dar los resultados para clasificar dentro de los mejores 40.

En cuanto al número de neuronas por capa, no es necesario tener gran número de ellas, los mejores resultados observados no pasan de 10 neuronas, ni siquiera en la arquitectura de red de una capa.

Solo en el caso de la red de 1 capa las mejores redes pararon su entrenamiento por llegar al máximo número permitido de épocas (en este caso se restringió a 3000 y el error de entrenamiento a 0,01), en los otros casos (2 y 3 capas de neuronas), la mayoría de las mejores redes detuvieron su entrenamiento en números muy inferiores a 3000, incluso menores que 100 épocas.

No obstante, estos resultados no permiten dar un esquema general de arquitectura para el problema de predicción y aunque la herramienta software pueda trabajar en modo automático con una red feed-forward, de 5 capas cada una de 10 neuronas, lo más conveniente es desarrollar inicialmente un proceso de prueba con todas las posibilidades que ofrece la herramienta a fin de establecer la red de mejores prestaciones, acorde con la curva de demanda específica que se desee predecir.

CONCLUSIONES

El trabajo de investigación plantea y demuestra la hipótesis de la descomposición de la serie de tiempo como una solución al problema de predicción de la demanda de potencia eléctrica horaria, aportando este procedimiento metodológico al estado del arte en el tema, ya que la revisión bibliográfica del mismo no plantea este esquema como partes de las soluciones encontradas.

Como resultado del planteamiento metodológico se diseña la herramienta software "PREDICCIÓN", la cual está basada en MATLAB dadas sus ventajas computacionales y el aprovechamiento de las toolboxes de identificación de sistemas, redes neuronales y estadística, permitiendo ahorrar esfuerzos de programación de los modelos, funciones o algoritmos matemáticos implícitos en los procesos de predicción.

En cuanto a los sistemas convencionales de predicción basados en identificación de sistemas, se puede decir que ha existido un avance también notable, en el desarrollo de algoritmos que pueden perfectamente competir con las redes neuronales obteniéndose resultados satisfactorios en la predicción y muy cercanos al de la red neuronal.

La propuesta metodológica permitirá aprovechar al máximo la capacidad de la red neuronal en la predicción de la componente irregular de la curva de demanda, dejando las otras componentes para ser predichas por separado mediante la simple extrapolación de una función matemática cuya determinación es más sencilla. Esto permite plantear esquemas de predicción basados en redes neuronales que manejen gran cantidad de datos históricos (lo cual permite incluir efectos estacionales no estimados) y necesiten menor esfuerzo de entrenamiento así como, menor tamaño en su arquitectura.

El trabajo de investigación presenta un estudio de las diferentes formas de modelado de las series de tiempo, técnicas y procesos de predicción, aplicados en forma específica al análisis de las características de la demanda de potencia eléctrica horaria de la ESSA. No obstante la metodología planteada muy general, podría ser aplicada en cualquier tipo de serie de tiempo en el corto plazo, aprovechando la herramienta software con ligeras modificaciones de forma.

En cuanto a la definición de una red óptima para el problema de predicción, se encontró que no se puede generalizar en lo que respecta a arquitecturas de red, tipos de conexión algoritmos de entrenamiento, etc. Infortunadamente la heurística es el mejor camino para bordar el problema, llegando a una solución basada en el ensayo y no en la deducción matemática. No obstante, esta heurística permite en el caso de la demanda estudiada, establecer caminos por donde los cuales es más fácil el transito hacia el encuentro de la red óptima, por ejemplo:

Las mejores tipos de red encontrados Elman (recurrente) y feed-forward.

Los mejores algoritmos de entrenamiento encontrados son el Gradiente descendente y Levenberg Marquardt.

El mejor número de neuronas por capa, está alrededor de 10.

Si la red es la adecuada, se llega rápido a la meta de error (en este caso en menores que 100 épocas), cuando hay números grandes de épocas en entrenamiento (mayores de 1000) se corre el riesgo de que la red pierda su capacidad de generalizar (se sobre-entrene).

BIBLIOGRAFÍA

[Acevedo-2000] ACEVEDO, Alexander; GONZALEZ, Leonardo. "Predicción de los Valores Máximo y Mínimo de la Demanda de potencia Diaria Utilizando Redes Neuronales". Proyecto de Grado, Universidad Industrial de Santander. Bucaramanga, 2000.

[AlFuhaid-1997] ALFUHAID, A; ELSAYED, M; MAHMOUD, M. "Cascaded Artificial Neural Network for Short Term Load Forecasting". IEEE Transactions on Power Systems, Vol 12, Nº4, November 1997.

[Alvarado-1998] ALVARADO, Eber; WILCHES, Andrés. "Predicción de Caudales en el Mediano Plazo Mediante Redes Neuronales". Proyecto de Grado, Universidad Industrial de Santander. Bucaramanga, 1998

[Baumann-1993] BAUMANN, Thomas; GERMOND, Alain. "Application of the Kohonen Network to Short Term Load Forecasting". IEEE. 1993.

[Becerra-1998] BECERRA, Alexander; GUALDRÓN, Oscar. "Predicción de demanda Mediante Redes Neuronales". Proyecto de Grado, Universidad Industrial de Santander. Bucaramanga, 1.998.

[Box-1970] BOX, G.E. and JENKINS, G.M. "Time Series Analysis", Holden-Day, San Francisco. 1970.

[Choueiki-1997] CHOUEIKI, H; MOUNT-CAMPBELL, C; AHALT, Stanley. "Building a 'Quasi Optimal' Neural Networks to Solve the Short Term Load Forecasting Problem". IEEE Transactions on Power Systems, Vol 12, Nº4, November 1997.

[Collopy-1992] COLLOPY, F; ARMSTRONG, J.S. "Expert opinions

about extrapolation and the mystery of the overlooked discontinuities". International Journal of Forecasting, 8, pp. 575-582. 1992

[Demuth-1998] DEMUTH, Howard; BEALE, Mark. "Neural Networks Toolbox: User's Guide". The Math Works,Inc. 1998.

[Elman-1990] ELMAN, Jeffrey. "Finding Structure in Time". Cognitive Science. Nº14, 1990.

[ElSharkawi-1996] EL-SHARKAWI, M; NIEBUR, D. "A Tutorial Course on Artificial Neural Networks whit Applications. Chapter 6: State of the Art, Overview on Artificila Neural Networks in Power Systems". IEEE Power Engineering Society. 96 TP 112-0.

[Freeman-1993] FREEMAN, James; SKAPURA, David. "Redes Neuronales Artificiales: Algoritmos, aplicaciones y Técnicas de Programación". Addison-Wesley/Diaz de Santos, 1993.

[Funahashi-1990] FUNAHASHI,K;. "Approximate realization of identity mappings by three layer neural networks". Electronics and Computation in Japan. part 3, #66, 1990.

[Gómez-1999] GÓMEZ, Tomás. "La Regulación del Sector Eléctrico. Nuevas Tendencias". Curso sobre distribución, comercialización y gestión de la demanda. Programa ALURE, Proyecto CREG. Cartagena, Diciembre de 1999.

[Granger-1989] GRANGER, C. "Forecasting in Business and Economics". Academic Press, Inc. 1997

[Gross-1987] GROSS, George; GALIANA, Francisco. Short-term Load Forecasting. Procedings of the IEEE, Vol 75, Nº12, December 1987.

[Hammerstorom-1993] HAMMERSTROM, Dan. "Neural Networks at Work". IEEE Spectrum. June 1993.

[Hanke-1996] HANKE, John; REITSCH, Arthur. “Pronósticos en los Negocios”. Prentice Hall Hispanoamericana. Mexico,1996.

[Haykin-1999] HAYKIN, Simon. “Neural Networks: A Comprensive Foundation”. Macmillan college Publishing Company. 1999.

[Heny-1998] HENY, E; SRINIVASAN, D; LIEW, A. “Short Term Load Forecasting Using Genetic algorithm and Neural Networks”. International Conference on Energy Management and Power Delivery – EMPD’98. Vol 2, 1998.

[Ho-1990] HO, Kung-Long; HSU, Yuan-Yih. “Short Term Load Forecasting of Taiwan Power System Using a Knowledge-based Expert System”. IEEE Transaction on Power Aparatus and Systems, Vol 5, Nº 4, November, 1990.

[Hoproff-1993] HOPTROFF, R.G; “The principles and practice of time series forecasting and business modeling using neural nets”. Neural Computing and Applications, 1, pp. 59-66. 1993

[Hsu-1991] HSU, Yuan-Yih; YANG, Chien-Chuen. “Desing of Artificial Neural Networks for Short Term Load Forecasting, Part I & II”. Procedings of the IEEE, Vol 138, Nº5, September 1991.

[Jollife-1986] JOLLIFE. Principal Component Analysis, Springer-Verlag, 1986.

[Juberias-1999] JUBERIAS, G; YUNTA, R; MENDIVIL, C. “A New ARIMA Model for Hourly Load Forecasting”. IEEE Transmission and Distribution Conference. Vol 1, 1999.

[Kandil-1999] KANDIL, N; SOOD, V; SAAD, M. “Use of ANNs for Short Term Load Forecasting”. IEEE Canadian Conference on Electrical and Computer Enineering. Vol 2, 1999.

[Kohonen-1990] KOHONEN, Teuvo. “The Self-organizing Map”. Procedings of the IEEE, Vol 78, Nº9, September 1990.

[Lee-1991] LEE, K; CHA,Y; KU,C. “ A Study on Neural Networks for Short Term Load Forecasting”. IEEE. 1991.

[León-1994] LEÓN, Miguel; RODRIGUEZ, Nelson. “Técnicas de Modelado y Métodos de Previsión de Demanda”. Proyecto de Grado, Universidad Industrial de Santander. Bucaramanga, 1994.

[Lippmann-1987] LIPPMANN, Richard. “An Introduction to Computing with Neural Nets”. IEEE ASSP Magazine. April 1987.

[Ljung-1994] LJUNG, Lennart. “From Data to Model: A Guided Tour”. Control’94, Conference Publication, IEE. March, 1994.

[Ljung-1997] LJUNG, Lennart. “System Identification Toolbox: User’s Guide”. The Math Works,Inc. 1997.

[Moghram-1989] MOGHRAM, Ibrahim; RAHMAN, Saifur. “Análisis and Evaluation of Five Short-term Load Forecasting Techniques”. IEEE Transaction on Power Aparatus and Systems. Vol 4, Nº 4, October 1989.

[Morioka-1993] MORIOKA; Y et. al. “Next Day Peak Load Forecasting Using Multilayer Neural Network With an Aditional Learning”. IEEE. 1993.

[Nakamura-1997] NAKAMURA, Shoichiro. “Análisis Numérico y Visualización Gráfica con MATLAB”. Prentice Hall Hispanoamericana. Mexico, 1997.

[Park-1993] PARK, D et. al.. “Load Curve Shaping Using Neural Networks”. IEEE, 1993.

[Perry-1999] PERRY, C. “Short Term Load Forecasting using

Multiple regression Analysis. IEEE Rural Electric Power Conference, 1999.

[Portela-2000] PORTELA, Alvaro; ROJAS, Hernán. “Predicción de Demanda Usando Redes Neuronales Entrenadas con el Algoritmo de Correlación en Cascada”. Proyecto de Grado, Universidad Industrial de Santander. Bucaramanga, 2000.

[Priestley-1998] PRIESTLEY, M.B. "Non-Linear and Non-Stationary Time Series Analysis". Academic Press, London. 1998.

[Rewagad-1998] REWAGAD, A; SOANAWANE, V. “Artificial Neural Network Based Short Term Load Forecasting”. IEEE Region 10 International Conference on Global Conectitivity in Energy Computer Comunications and Control – TENCON’98. Vol 2, 1998.

[Rodriguez-2000] RODRIGUEZ, Hector; URREGO, Liz. "Utilización de las Redes Neuronales para la Predicción de las Variaciones del Perfil de Demanda Diaria a Corto Plazo". Proyecto de Grado, Universidad Industrial de Santander. Bucaramanga, 2000.

[Rumelhart-1986] RUMELHART, D.E; McCLELLAND, J. "Parallel Distributed Processing". MIT Press, Cambridge. 1986.

[Shan-1997] SHAN, Shao; YAMING, Sun. “Short Term Load Forecasting Using Fuzzy Neural Networks”. Fourth International Conference on Advances in Power System Control, Operation and Management – ASPSCOM’97. Vol 1, 1997.

[Sharaf-1993] SHARAF, A; LIE, T; GOOI, H. “A Neural Network Based Short Term Load Forecasting Model” IEEE, CCECE/CCGEI’93. 1993.

[Srinivisan-1996] SRINIVISAN, Dipti; CHANG, C; TAN, Swee-Sien.

"One-day Ahead Electric Load Forecasting With Hybrid Fuzzy-neural Networks". IEEE,1996.

[Srivastava-1997] SRIVASTAVA, S; VEANKATARAMAN, D. "Short Term Load Forecasting Using Recurrent Neural Networks". Fourth International Conference on Advances in Power System Control, Operation and Management – ASPSCOM'97. Vol 1, 1997.

[Tang-1991] TANG, Zaiyong, et al. "Feed Forward Neural Nets as Model for Time Series Forecasting". Computer Science TR91-008. 1.991.

[Tong-1990] TONG, H. (1990): Non-Linear Time Series: A Dynamic System Approach. Oxford University Press, Oxford. 1999

[Vermaak-1998] VERMAAK, J; BOTHA, E. "Recurrent Neural Networks for Short Term Load Forecasting". IEEE Transactions on Power Systems. Vol 13, 1998.

[Wang-1998] WANG, Feng; et al. "Short Term Load Forecasting Based on Weather Information". IEEE Proceedings of the International Conference on Power Systems Technology – POWERCON'98. Vol 1, 1998.

[Xiao-1995] XIAO, B; MCLAREN, P. "An Artificial Neural Networks for Short Term Load Forecasting". IEEE Proceedings of WESCANEX'95. 1995.

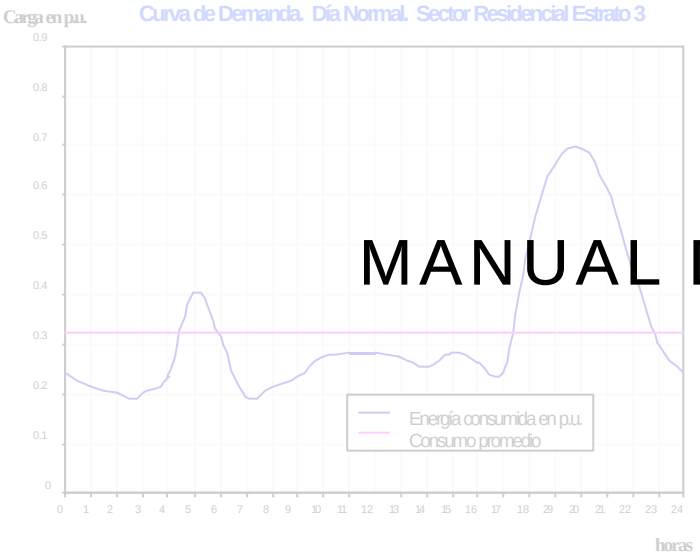
[Yule-1927] YULE, G. "On a Method of Investigating Periodicity in Disturbed Series with special reference to Wolfer's Sunspot Numbers". Phil. Trans. Royal Society London, 226, pp. 267-298. 1927.

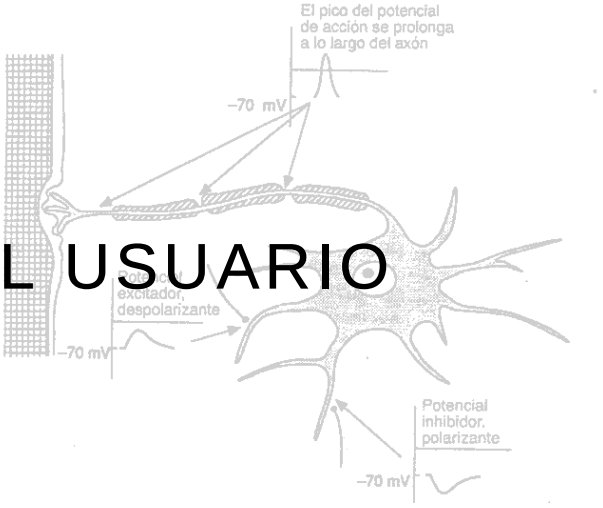
ANEXO A. MANUAL DEL USUARIO DE LA HERRAMIENTA SOFTWARE

MAESTRIA EN POTENCIA ELECTRICA

PREDICCIÓN

PROTOTIPO PARA LA ESTIMACIÓN DE DEMANDA DE CORTO PLAZO





MANUAL DEL USUARIO

Prototipo desarrollado como parte del trabajo de investigación: "Predicción de demanda de corto plazo empleando redes neuronales", para optar al título de Magister en Potencia Eléctrica, por parte del Ingeniero Electricista: César Yobany Acevedo Arenas.

UNIVERSIDAD INDUSTRIAL DE SANTANDER
ESCUELA DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
MAESTRÍA EN POTENCIA ELECTRICA
BUCARAMANGA - 2004

PREDICCIÓN - Prototipo Para La Estimación De La Demanda De Corto Plazo (Manual De Usuarioa)

1. INTRODUCCIÓN

"Predicción" es un una herramienta computacional prototipo desarrollada como parte del trabajo de investigación "Predicción de demanda de corto plazo mediante redes neuronales", el cual es requisito para optar al título de Magíster en Potencia Eléctrica de la Universidad Industrial de Santander, por parte del Ingeniero Electricista César Yobany Acevedo Arenas.

Su finalidad es la de aplicar la metodología propuesta dentro del trabajo de investigación, para la realización predicciones de la demanda horaria de potencia eléctrica independiente del grado de agregación de la carga, aunque puede ser también empleado en otro tipo de predicciones de demanda de corto plazo como por ejemplo la demanda de gas natural o el consumo de agua potable.

Por ser un modelo básicamente autorregresivo, es decir, las predicciones futuras se basan en el comportamiento de la demanda histórica, su aplicabilidad en la predicción de demanda de corto plazo tanto en sector eléctrico como del gas natural, acueducto y otros servicio públicos es amplia, debido a que se supera el inconveniente de contar con variables climáticas explicativas del modelo.

Su desempeño está basado en el concepto de descomposición de series de tiempo y la aplicación de redes neuronales artificiales como aproximadoras de funciones matemáticamente desconocidas. Este planteamiento permite un esquema en el cual una vez se han determinado las componentes, se proyecta por separado la tendencia y ciclo, dejando la irregularidad y estacionalidad para que la red neuronal se encargue de predecirlas.

2. REQUERIMIENTOS

"Predicción" puede funcionar con requerimientos mínimos de hardware y software en su etapa de operación, es decir, una vez se ha encontrado una arquitectura de red óptima para el problema en específico y se pone en funcionamiento el predictor. Sin embargo, se recomienda un computador de buenas prestaciones en cuanto a velocidad de procesamiento en la etapa previa a la de operación, donde se establecen los parámetros del modelo a emplear, ya que se requiere ejecutar pruebas con varias arquitecturas de red a fin de encontrar la red neuronal óptima para el problema de predicción en específico. Los requerimientos mínimos de operación son:

HARDWARE: Computador con procesador superior a Pentium 233 Mhz – 32 Mb de memoria RAM, 15 Mb de espacio en disco duro.

SOFTWARE: Plataforma Windows, Paquete de análisis matemático MATLAB 5.3 (o superior) incluyendo la toolbox de neural networks

Como sugerencia seria ideal utilizar el prototipo como única aplicación activa en su PC.

3. MANEJO DEL PROGRAMA

3.1. INICIO DEL PROGRAMA

Diríjase al botón *Inicio* de Windows, vaya a *programas* busque *prediccion* y haga clic en *Arranque Programa*

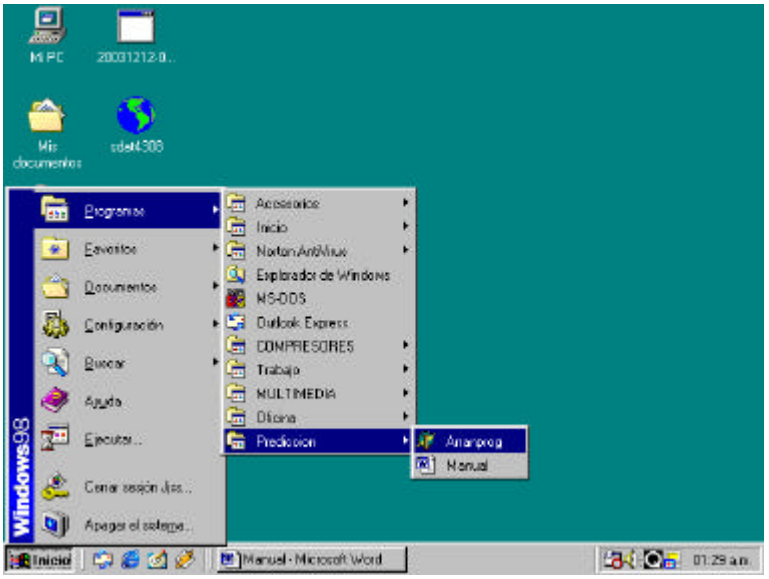


Figura 1. Inicio del programa

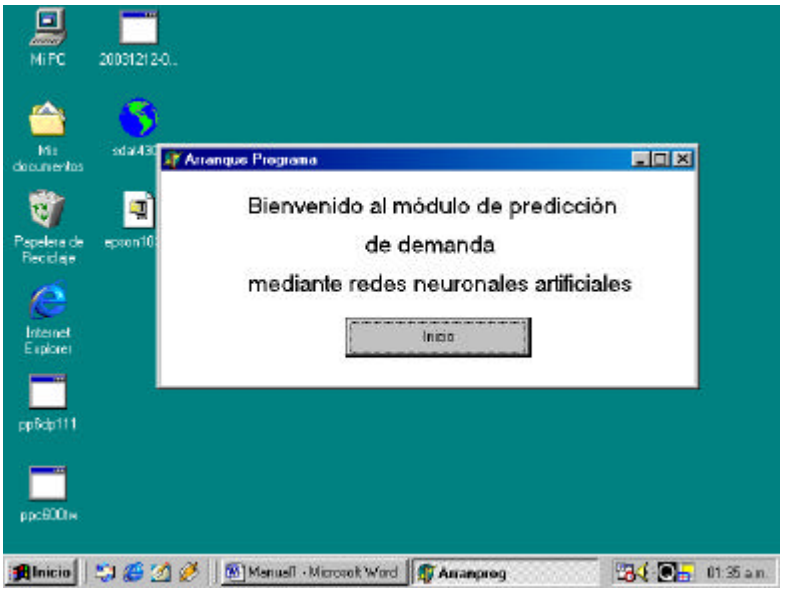


Figura 2. Confirmación del paso a realizar

Al seguir dichos procedimientos se iniciara un script de MATLAB; el cual lo lleva a

la siguiente ventana de opciones

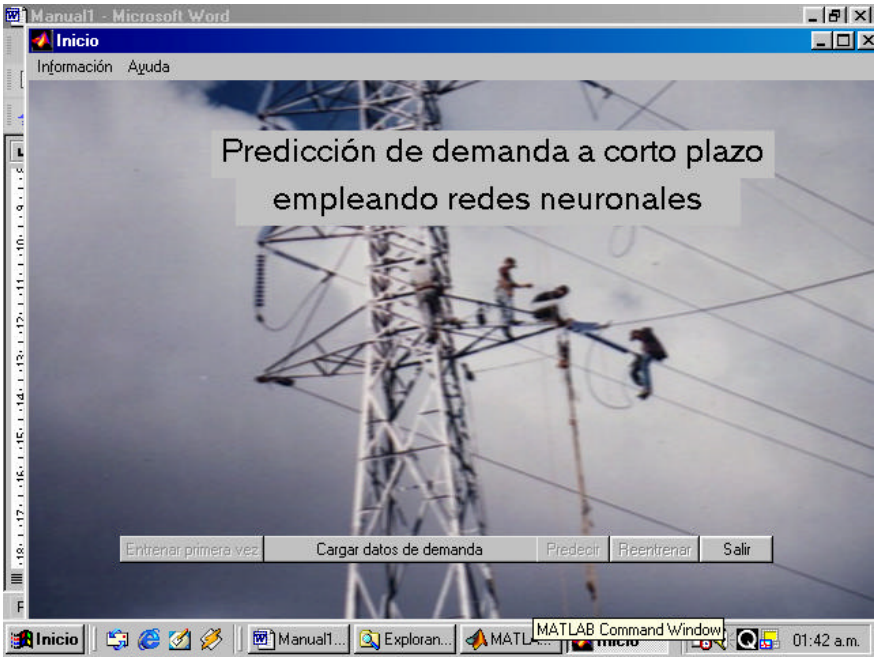


Figura 3. Menú inicial de opciones

Como se observa solo están activas dos de las opciones (*Cargar datos de demanda* y *salir*) pues las otras tres, solo se activan cuando ya se haya realizado el proceso alguna vez (ya sea que se hayan cargado datos y realizado su análisis estadístico, o que se haya realizado el entrenamiento una vez se tengan datos, o que se haya realizado la predicción una vez se haya entrenado la red neuronal). En los siguientes pasos se explicará el proceso por primera vez, en cuyo caso, la carga de datos de demanda es el primer paso obligatorio y se explica a continuación. as otras opciones serán explicadas posteriormente.

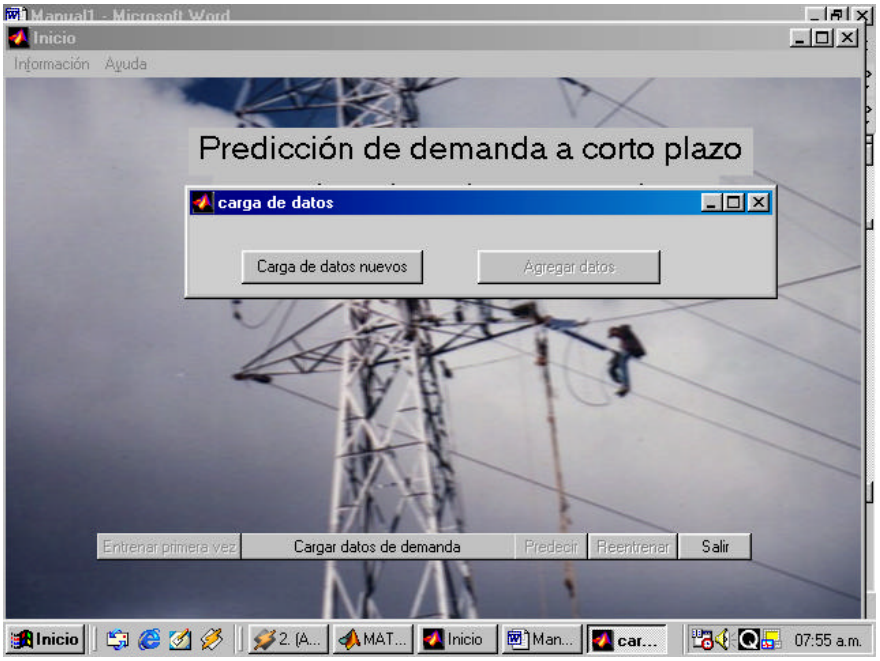


Figura 4. Inicio carga de datos

Al dar click en *cargar datos de demanda* aparece la el cuadro de dialogo *carga de datos,* el cual se muestra en la figura 4. Como se observa existen dos opciones, *Carga de datos nuevos* y *Agregar datos.* Por ser el proceso por primera vez, el botón de la opción *Agregar datos* está inactivado, ya que su función es adicionar datos de demanda a un archivo ya existente. Por lo tanto la única opción es el botón *Carga de datos nuevos*, el cual inicia el proceso de adquisición de la información de demanda y borra los datos que existan anteriormente de otro proceso de predicción; cuidado que debe ser tenido en cuenta, cuando se desea realizar las predicciones de demanda con datos ya existentes, ya que al dar click en este botón se perderá toda la información antes ingresada. A continuación se presenta el proceso de carga de datos.

3.2. CARGA Y PROCESAMIENTO DE DATOS

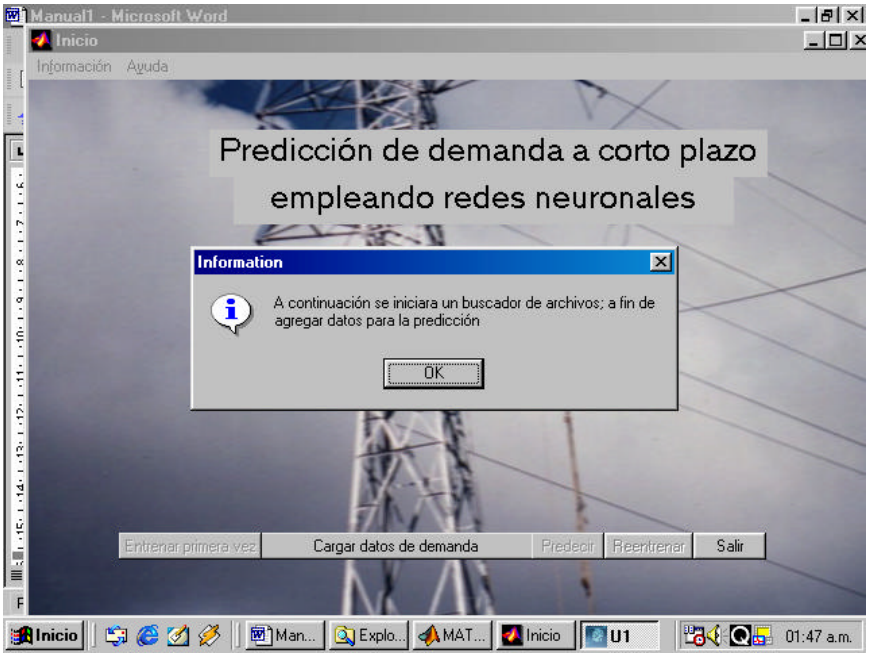


Figura 6. inicio buscador de archivos

Al dar click en cualquiera de los botones *Carga de datos nuevos* y *Agregar datos,* se iniciara un proceso que permite explorar las unidades de disco presentes en el computador, como se muestra en la figura 6. Estos archivos deben tener la extensión ".xls", es decir deben estar en MS-EXCEL y contener la información de demanda de manera que en sus columnas estén las horas del día (24 columnas) y en sus filas cada uno de los días del historial de datos disponibles. Al dar clic en *OK,* se abre un cuadro de dialogo en el cual se puede establecer la ruta para la ubicación del documento en MS-EXCEL que contiene los datos de demanda y abrir el archivo seleccionado como se muestra en la figura 7.

Al dar click en *Abrir archivo seleccionado*, la interfase abre la hoja de cálculo deseada y muestra el cuadro de dialogo de la figura 8, el cual permite la captura de los datos desde MS-EXCEL.

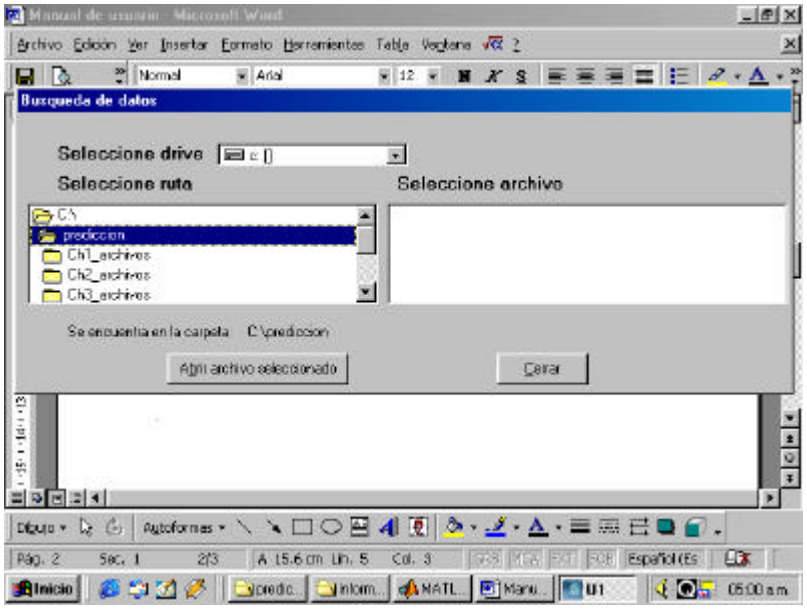


Figura 7. Búsqueda de archivos

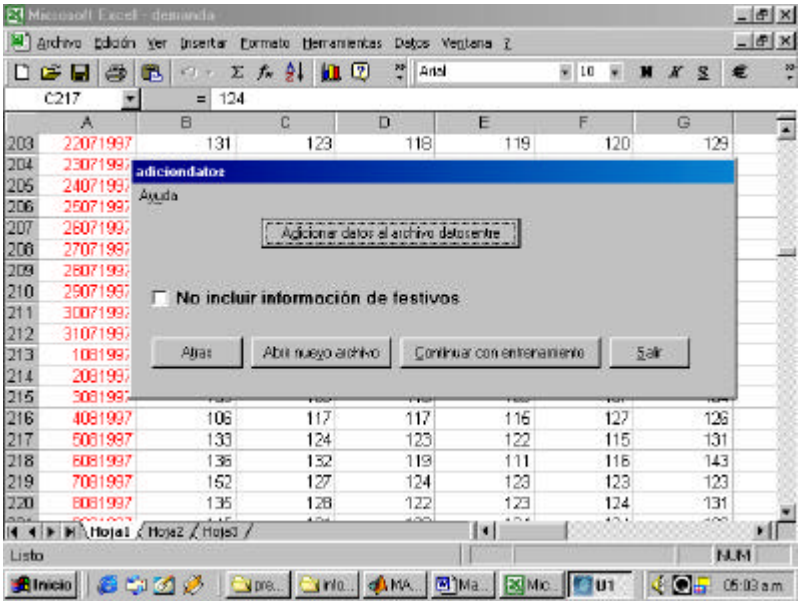


Figura 8. Inicio captura de datos

El botón *Adicionar datos al archivo datosentre* es que finalmente realiza la captura de los datos. No obstante, antes de realizar la captura es necesario ingresar cierta información sobre los datos, las cuales se ingresan en los cuadros de dialogo mostrados en las figuras 9, 10 y 11, y se detallan en la ayuda incorporada a la interfase.

En el cuadro de dialogo de la figura 9, se selecciona la forma como vienen presentados los datos en la hoja de cálculo en MS-EXCEL. Si únicamente se ingresan datos de demanda (kW, MW o GW) se escoge la opción *sus datos son solo de demanda?*, si además de los datos de demanda se capturan datos de fechas correspondientes a cada día en el formato dd-mm-aaaa (por ejemplo: 5 de Enero de 2003 = 05-01-2003), se selecciona la opción *incluyen sus datos fecha(ddmmaaaa) y demanda?*, con la cual el programa no necesitaría información adicional de fechas. Al dar clic en *Continuar* aparece el cuadro de dialogo de la figura 10.

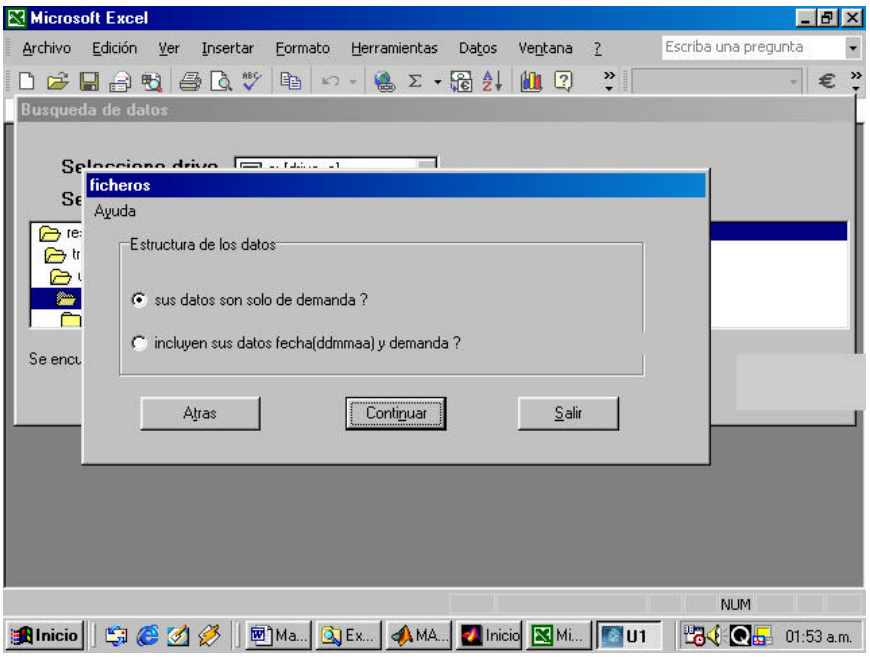


Figura 9. Preguntas adicionales (Estructura de los datos)

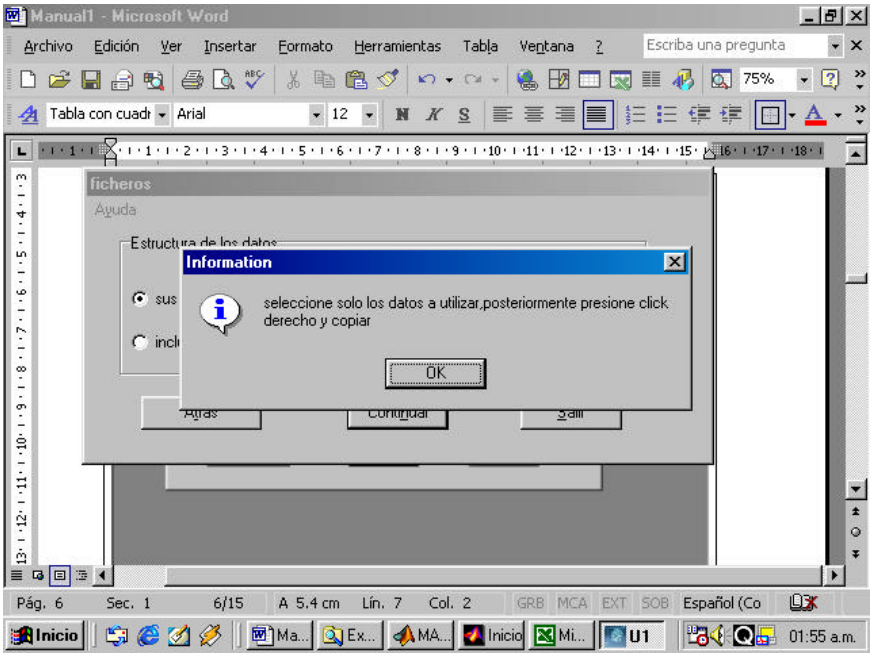


Figura 10. Ventana preliminar de captura

Como el programa permite incorporar datos de demanda a un archivo previamente establecido sobre el cual ya se hayan realizado entrenamientos y/o predicciones, se muestra al operador un cuadro de dialogo en el cual aparece la última fecha de utilización de los archivos y el rango de fechas introducidas al prototipo en dicha ocasión. Cuando se cargan datos por primera vez, aparecerá un número 1, como se muestra en a figura 11.

Sí en el cuadro de dialogo de la estructura de datos mostrado en la figura 9, se seleccionó la opción *sus datos son solo de demanda?*, es decir la captura de datos no incluye fechas, aparecerá el cuadro de dialogo de la figura 12, en el cual se ingresan las fechas inicial y final de los datos, así como sus unidades (kW, MW o GW), a fin de que el programa las tenga en cuenta para el procesamiento de la información de entrada.

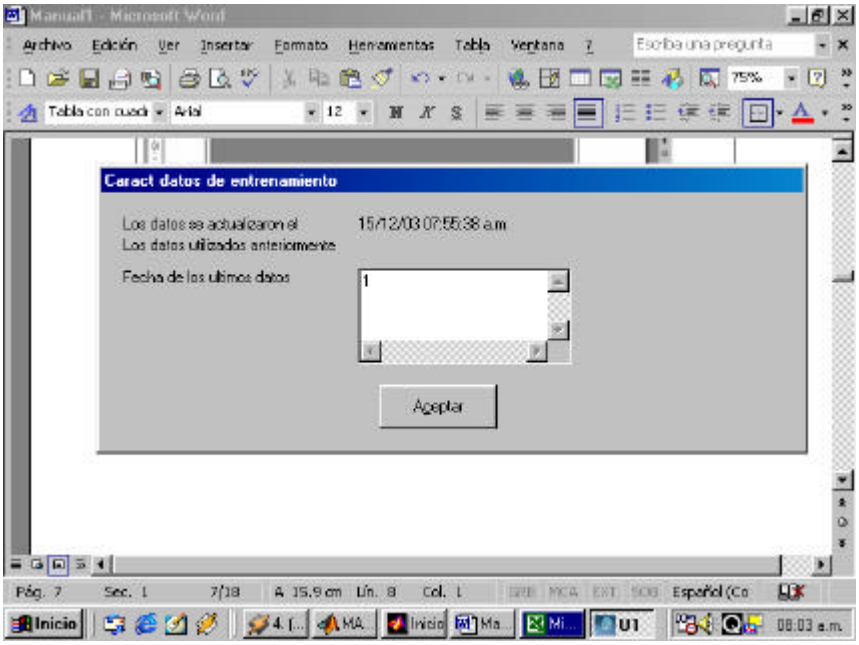


Figura 11. Ventana con información acerca de datos anteriores

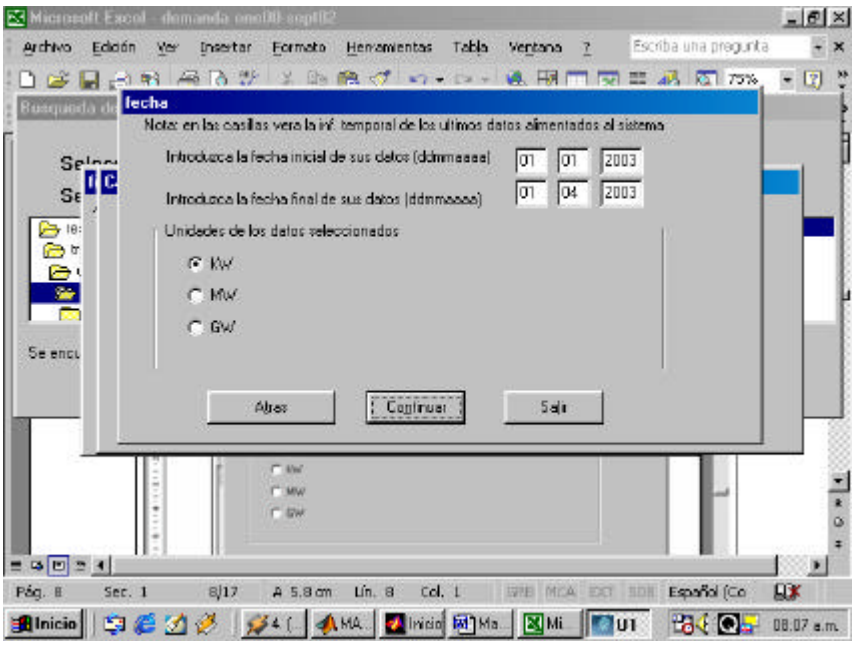


Figura 12. Entrada del rango de las fechas y unidades

Una vez se ha ingresado la información anterior se procede a capturar los datos desde la hoja de calculo, marcado las celdas correspondientes. Si los datos de que se dispone tienen una estructura como en la figura 13, es decir no tiene un formato dd-mm-aaaa en la primera fila, debe seleccionar en el cuadro de dialogo de la figura 9 la opción *sus datos son solo de demanda?*, ya que el programa asume que la primera fila corresponde a la primera hora del día. En este caso, para cargar los datos debe seleccionar solamente los datos de demanda de potencia con el ratón y posteriormente dar click derecho del ratón, para que aparezca la opción copiar seleccionándola. Posteriormente se debe ir al cuadro de dialogo de la figura 8 y dar click en el botón *Llevar datos al archivo datosentre* a fin de guardar la información.

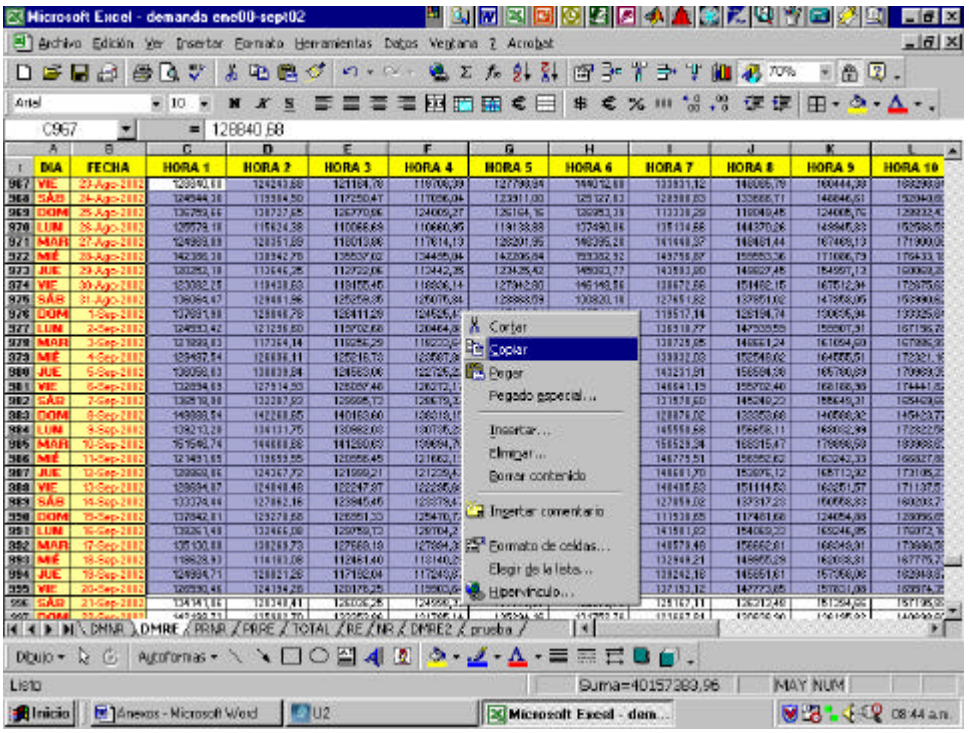


Figura 13. Carga de datos solo de demanda

Si los datos tienen una estructura como en la figura 14, es decir tiene un formato

de fechas dd-mm-aaaa en la primera fila, debe seleccionar en el cuadro de dialogo de la figura 9 la opción *incluyen sus datos fecha(ddmmaaaa) y demanda?*, ya que el programa asume que la primera fila corresponde a la fecha de los datos de demanda. En este caso, para cargar los datos se deben seleccionar tanto los datos de fechas como los de demanda y seguir el mismo pocedimiento de copiado y guardado del caso anterior.

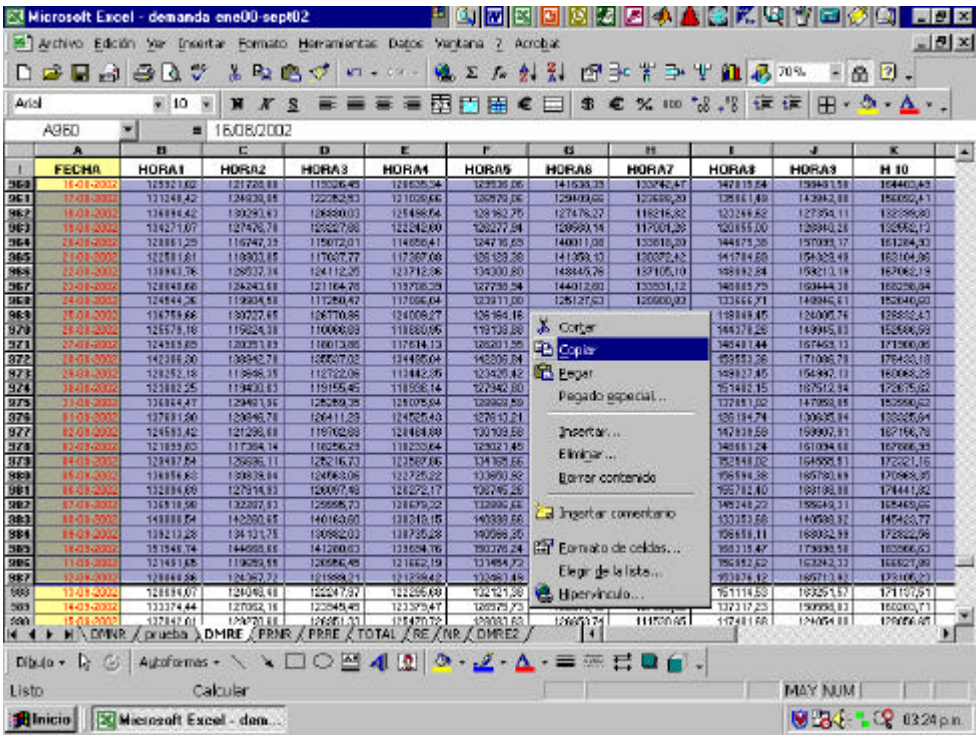


Figura 14. Carga de datos incluyendo fechas

Posterior a la captura de los datos se inicia el procesado estadístico de los datos realizado por MATLAB, para esto se debe dar click en la opción *Continuar con entrenamiento* del cuadro de dialogo mostrado en la figura 8. El avance de dicho análisis se observa mediante una interfase web que se despliega a medida que avanza el proceso, y se muestra a continuación.

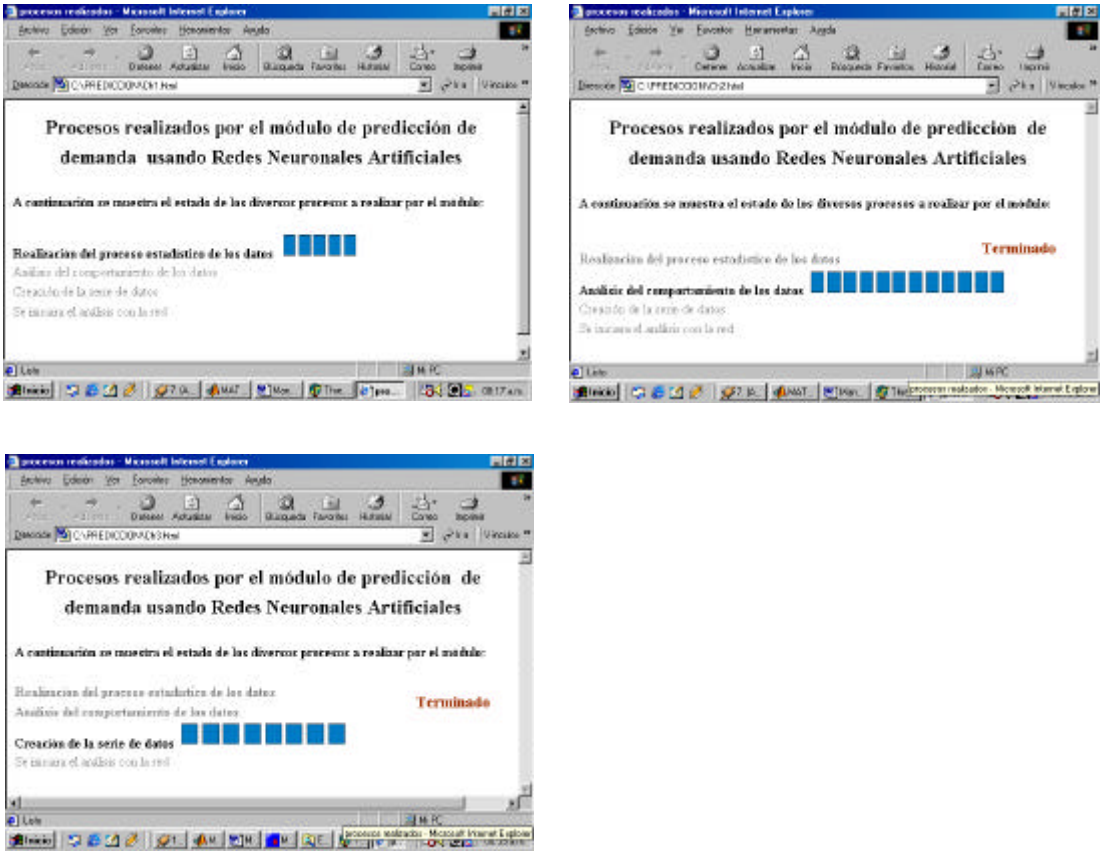


Figura 15. Ventanas mostrando estado del proceso estadístico

Una vez se ha realizado el proceso estadístico de los datos aparecerá el cuadro de dialogo de la figura 16. En él se solicita confirmación para realizar suavizado exponencial a los datos de la componente irregular de la serie obtenidos del procesamiento anterior. Una vez se ha dado click en cualquiera de los botones de confirmación aparece el cuadro de la figura 17.

Si se estan realizando tareas simultaneas al prototipo probablemente luego de las tres ventanas anteriores notara que el prototipo toma mucho tiempo en ofrecer alguna respuesta, en este caso presione en el teclado control+alt para ver las tareas ocultas, a fin de obtener una visión como la siguiente.

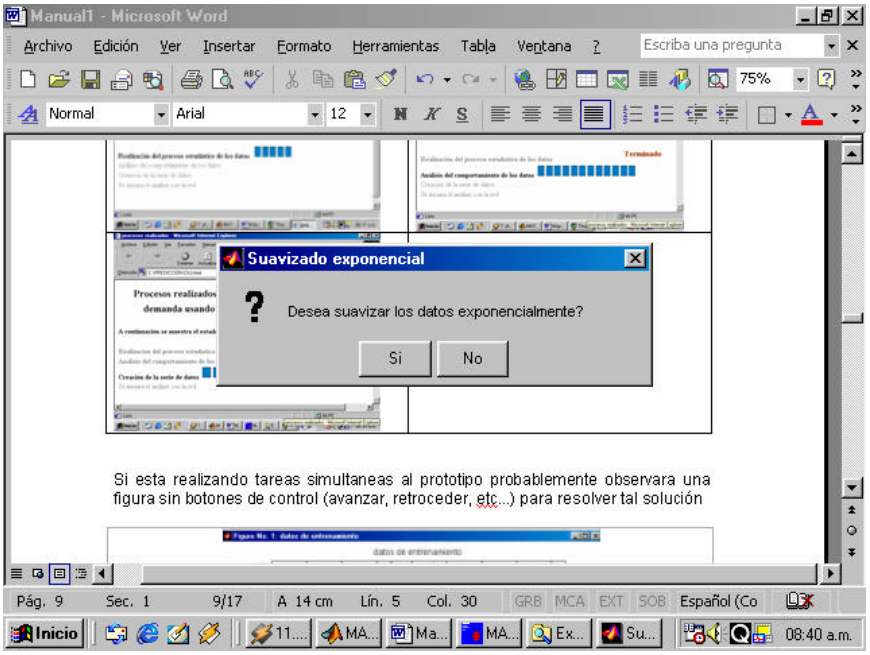


Figura 16. Suavizado de los datos de irregularidad

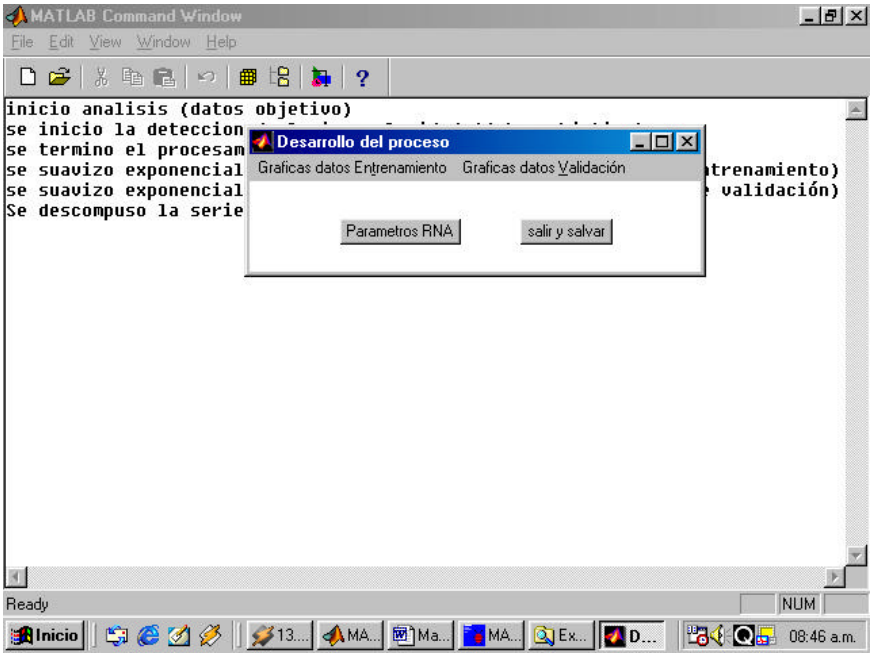


Figura 17. Finalización del proceso estadístico

En el cuadro de dialogo mostrado en la figura 17, aparecen las opciones de continuar con el entrenamiento en el botón *Parámetros RNA* o postergar el proceso de entrenamiento de la red neuronal para otra ocasión previo guardado de los datos, mediante el botón *salir y salvar*, en cuyo caso se regresará a la ventana principal. Adicionalmente, el cuadro de dialogo permite observar las componentes resultantes del procesamiento estadístico de los datos y el análisis de la serie de tiempo (tanto para entrenamiento como para validación) mediante un menú de opciones en la parte superior.

3.3. ENTRENAMIENTO DE LA RED NEURONAL

Mediante el botón *Parámetros RNA* del cuadro de dialogo mostrado en la figura 17, se ingresa a la ventana de selección de casos de entrenamiento, en la cual se establece la arquitectura de la red a entrenar o se modifican sus características en cuanto a los parámetros de la red a utilizar durante el entrenamiento, lo cual se puede observar en la figura 18.

En la parte superior de la ventana de selección de casos se encuentran dos opciones: *Selección de los parámetros de entrenamiento* y *Ajustes avanzados*.

En la primera se establece la manera como se seleccionarán los parámetros de entrenamiento ya sea por selección propia del usuario, de manera automática o realizando todas las combinaciones de red neuronal posibles permitidas por el programa. En la opción de ajustes avanzados se establecen parámetros comunes al entrenamiento, como cantidad de épocas, error permisible de la red, tasa de aprendizaje, número de capas máximo y número neuronas por capa máximo, como se observa en la figura 19.

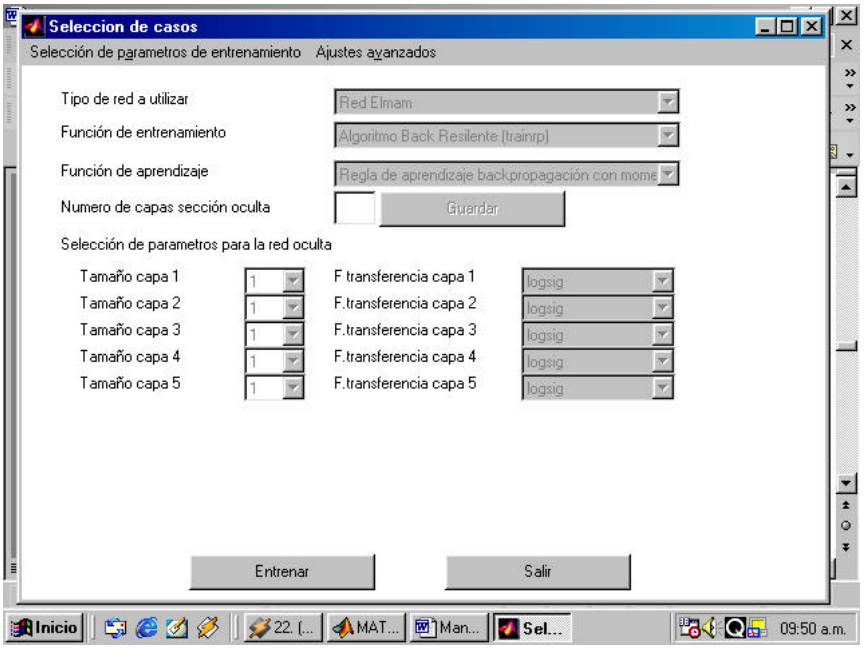


Figura 18. Selección de parámetros y arquitectura de red

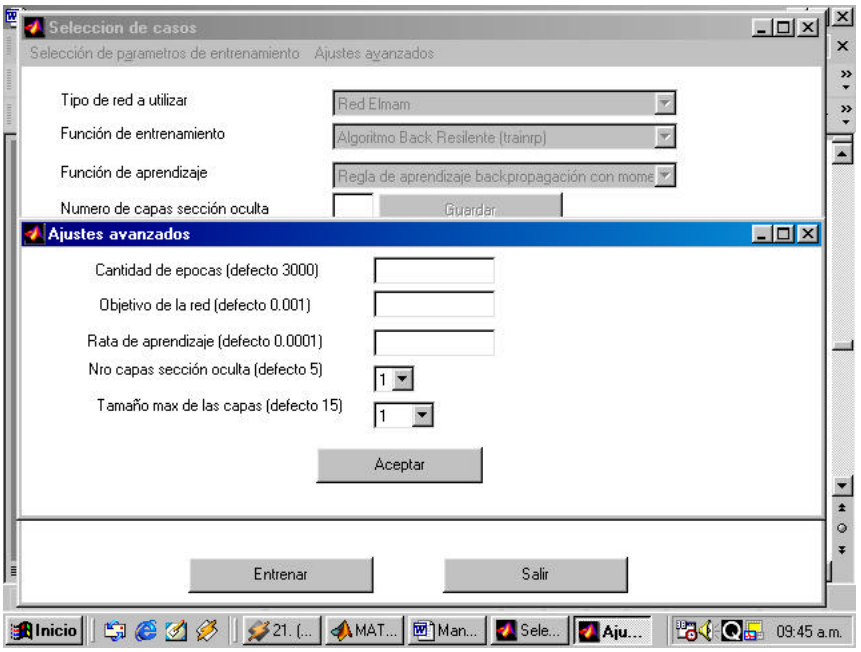


Figura 19. Opción de ajustes avanzados

Cualquiera de los parámetros anteriores pueden ser modificados de manera independiente, si no se realiza ningún cambio, el programa asume los siguiente valores por defecto en los parámetros de entrenamento de la red neuronal: Cantidad de épocas (3000), Error permisible (0.001), Tasa de aprendizaje (0.0001), Número máximo de capas de la sección oculta (5), Tamaño máximo para las capas de la sección oculta (15 neuronas por capa).

Una vez se han realizado los ajustes avanzados si son requeridos, se selecciona la manera como el usuario configurará la arquitectura de red. Para esto se da click en el botón *Selección de parámetros de entrenamiento*, apareciendo entonces el cuadro de dialogo de la figura 20.

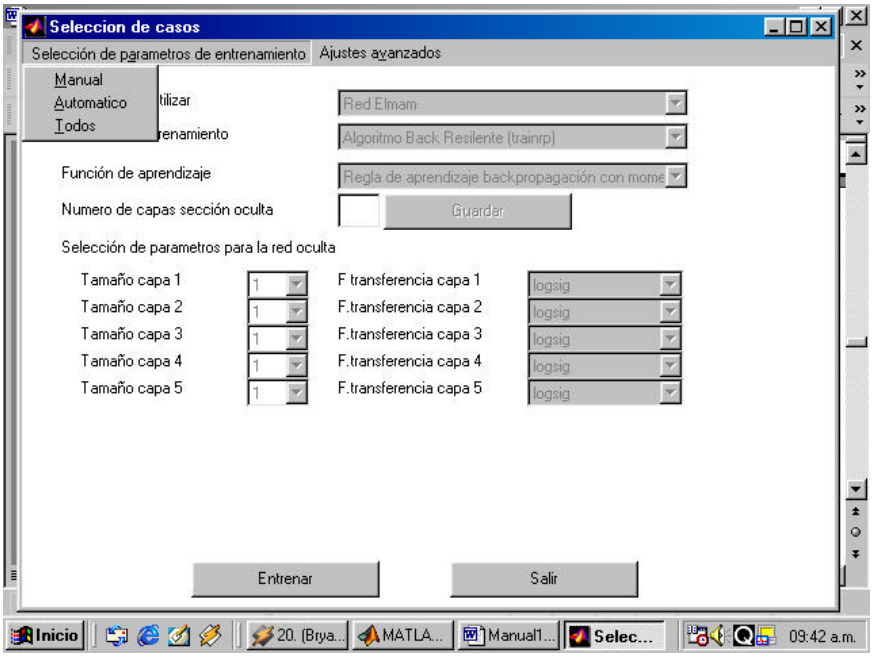


Figura 20. Formas de selección de parámetros y arquitectura de red

La selección de una de estas opciones permite crear o modificar la red neuronal, en cuanto a sus parámetros de entrenamiento y arquitectura. A continuación se presentan las posibilidades del programa en este aspecto:

- *Manual*: permite seleccionar los valores más aceptables en concepto del usuario, tales como: a opción manual activara las diversas casillas de entrada y selección de datos permitiendo seleccionar: Tipo de red (Elman, Feed Forward y Feed Forward en cascda), Función de entrenamiento (Gradiente descendente, Levenberg-Marquardt, Back resilente y gradiente descendente con momento), Regla de aprendizaje (Back propagation y Backpropagation con momento) y Número de capas de la sección oculta.

- *Automático*: toma parámetros predefinidos durante la elaboración y prueba del prototipo (Red de Elman con 5 caas ocultas de a 10 neuronas por capa)

Todos: realiza combinaciones de todas las arquitecturas de red posibles (15 neuronas por capa, 5 capas ocultas) y ajusta sus variables de acuerdo a las modificaciones realizadas por el usuario en los ajustes avanzados. Esta opción requiere de un computador con buenas características de velocidad, ya que los cálculos pueden tardar incluso días. Al terminar todas las combinaciones posibles presentara la opción de retomar la mejor a juicio del operador del prototipo para ser salvada, reentrenada y/o predecir con ella.

Sin importar cual sea el mecanismo de entrada de los parámetros a la red; esta será simulada. Presentando al terminar el proceso las graficas correspondientes a la respuesta de la red, la Desviación Absoluta de la Media (MAD), Raíz Cuadrada del Error Medio Cuadrado (RMSE) y el Porcentaje del Error Medio Absoluto (MAPE) tanto durante el entrenamiento como durante la prueba de la red. En la ventana que muestra el resultado de la prueba tienen cuatro opciones: *Salvar RNA actual*, *Predecir con RNA actual*, *Reentrenar RNA actual* y *Abandonar*, las cuales conectan con el siguiente paso del proceso que es la predicción. En las figuras 21 y 22 se observan ejemplos de lo anteriormente mencionado.

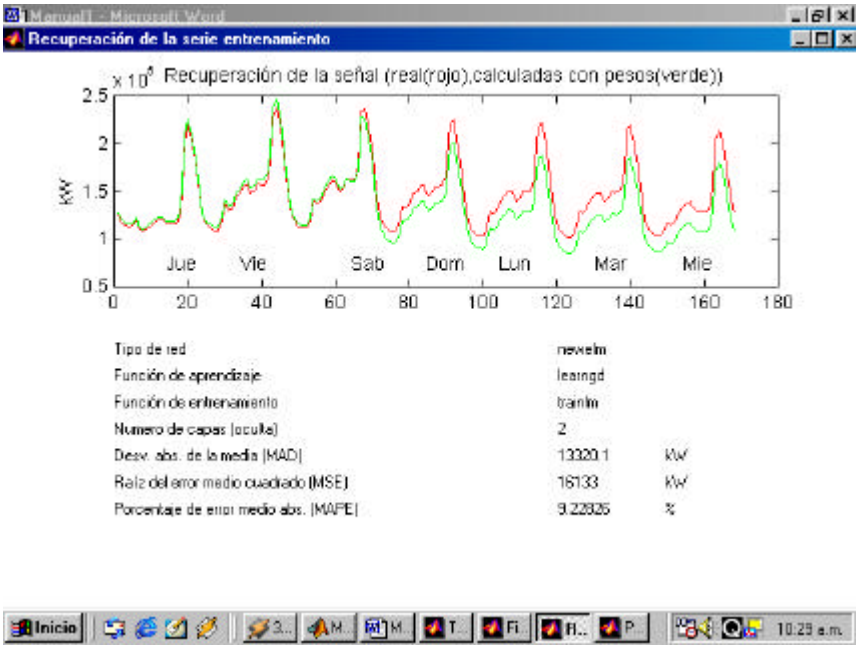


Figura 21. Resultado del entrenamiento

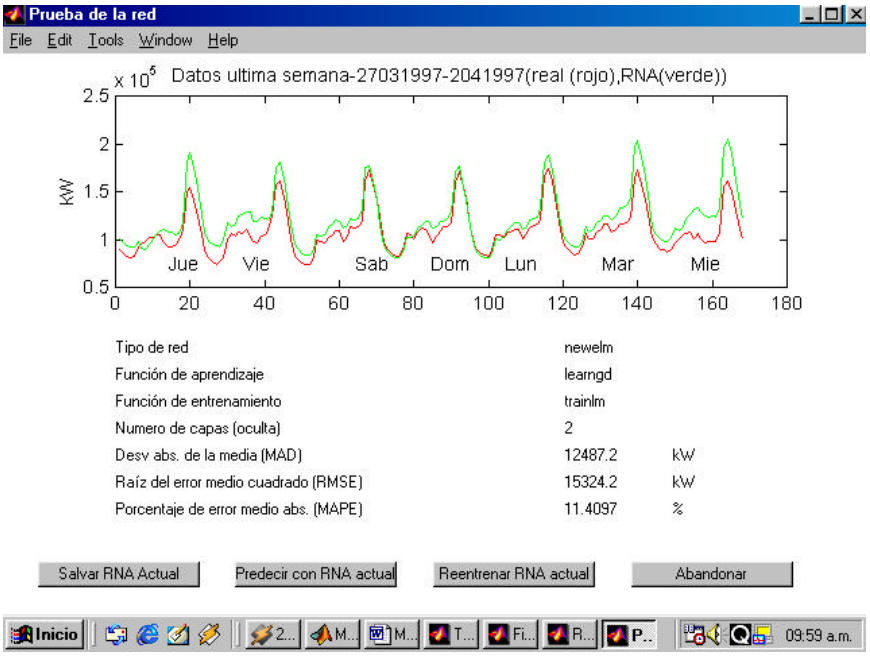


Figura 22. Resultado de la prueba

3.4. PREDICCIÓN

En la ventana que se muestra en la figura 22, se continua con el proceso de predicción, ya sea guardando la red neuronal obtenida del entrenamiento mediante la opción *Salvar RNA actual*, o realizar directamente la predicción de la demanda mediante *Predecir con RNA actual*. Si los resultados obtenidos del entrenamiento no cumplen con las expectativas del operador, se puede regresar al proceso de entrenamiento y modificar los paráméros de la red neuronal mediante la opción *Reentrenar RNA actual*.

En la figura 23 se muestra el cuadro de dialogo que aparece una vez de da click en el botón *Salvar RNA actual.* En éste, se le debe dar un nombre a la red, para que posteriormente sea recuperada, ya que se puede continuar con el proceso de predicción o abandonar el programa para ser retomado posteriormente.

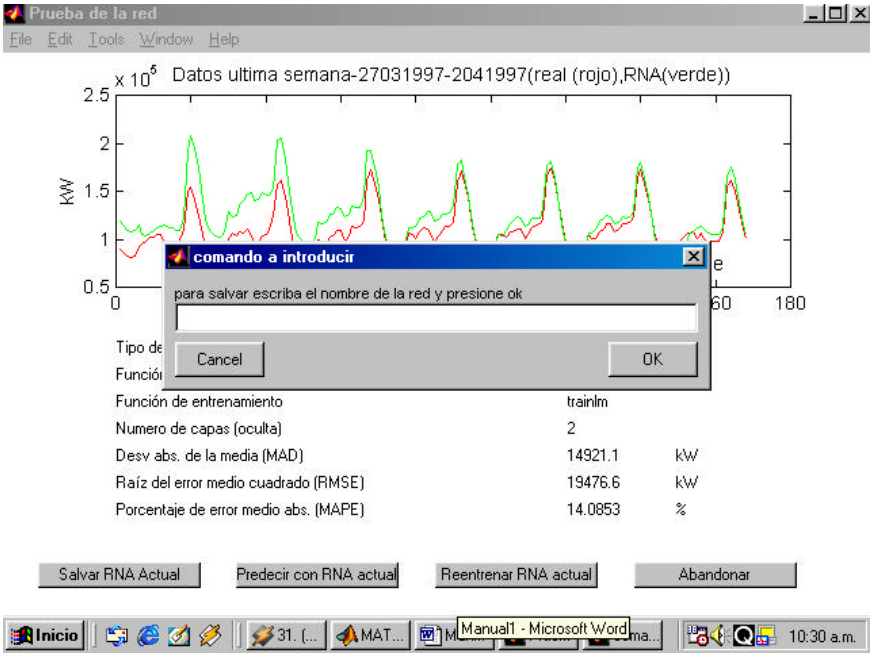


Figura 23. Opción Salvar RNA actual

En la figura 24 se muestra el cuadro de dialogo que aparece una vez se da click

en la opción *Predecir con RNA actual*. La predicción permite proyectar el comportamiento de los datos en el futuro, preguntándole la cantidad de semanas a avanzar en el tiempo.

En la opción *Semana Siguiente* se predice la siguiente semana a los datos introducidos a la red, mientras que en la opción *Otro* le preguntara la cantidad de semanas a avanzar en el tiempo.

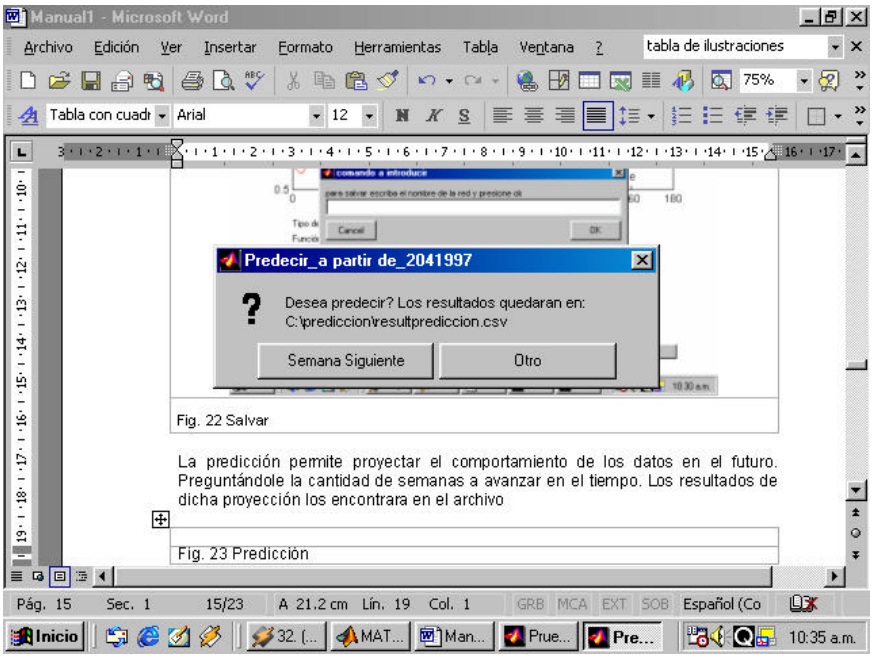


Figura 24. Selección de la semana a predecir

En la figura 25, se puede observar el cuadro de dialogo que aparece una vez se ha dado click en la opción *Otro*. En este cuadro de dialogo se debe introducir la cantidad de semanas posteriores a los datos, en la cual se encuentra la semana a predecir.

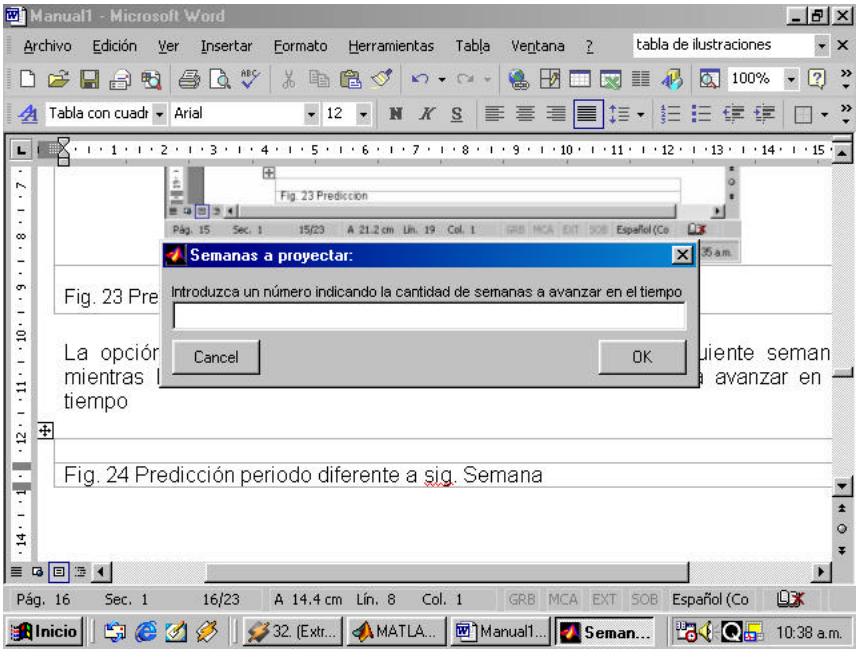


Figura 25. Predicción de periodo diferente a la semana siguiente

Como resultado de cualquiera de las opciones (*Semana Siguiente* u *Otro*) se generara el comportamiento proyectado por la RNA para la semana correspondiente. En la figura 26 se muestra la ventana que despliega el gráfico de la demanda proyectada para la semana requerida. Los resultados numéricos de dicha predicción se encontrarán en el archivo c:\prediccion\resultprediccion.csv

A partir de este punto puede: Cambiar los de trabajo de la red neuronal retornándolo a la ventana de inicio mostrada en la figura 3, mediante el botón *Reiniciar*. Volver a modificar el entrenamiento de la red, regresando a la ventana mostrada en la figura 22, mediante el botón *Regresar*.

Mediante el botón *Abandonar* se sale del proceso, en cuyo caso debe haberse guardado la red neuronal con la opción *Salvar RNA actual*, mostrada en la figura 23, ya que de no hacerlo se pierde la configuración de la red al salir del proceso.

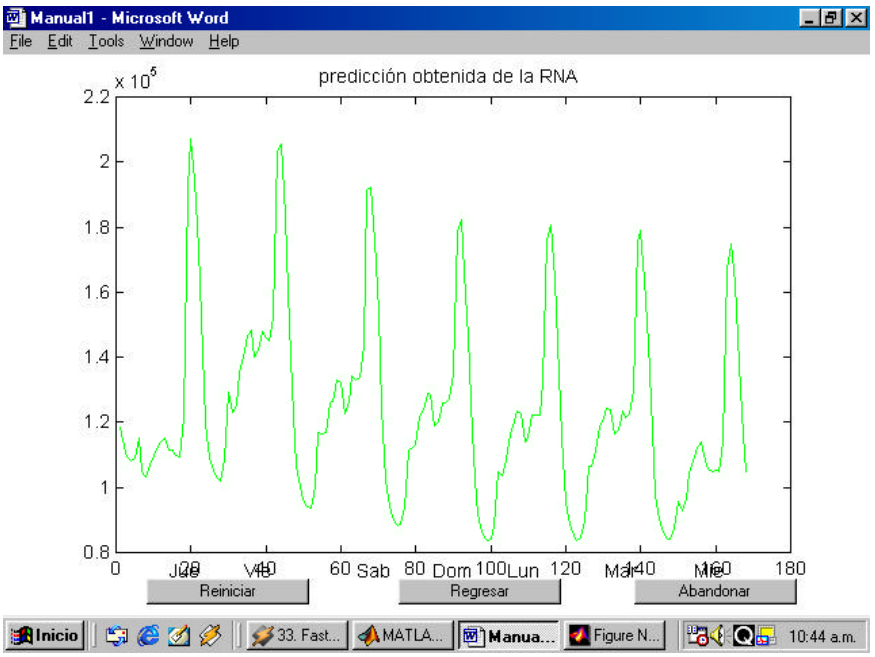


Figura 26. Predicción obtenida de la red neuronal

Al dar click en el botón *Regresar* a fin de reentrenar la red neuronal, se llega a la ventana mostrada en la figura 22. Cuando en ella se selecciona el botón *Reentrenar RNA actual* aparee el cuadro de dialogo mostrado en la figura 27, el cual ofrece dos posibilidades: Cambiar los parámetros de la red neuronal mediante la opción *modificar parametros*, retornando a la ventana de selección de casos mostrada en la figura 18, continuando el proceso como fue descrito allí; o, Iniciar una interfase similar a la utilizada durante la carga de los datos de demanda mostrada en la figura 28 a fin de cargar los nuevos datos a utilizar en el proceso de predicción mediante la opción *Adicionar datos*, repitiendo los procesos de carga de datos ya ilustrados en las figuras 7 a 15.

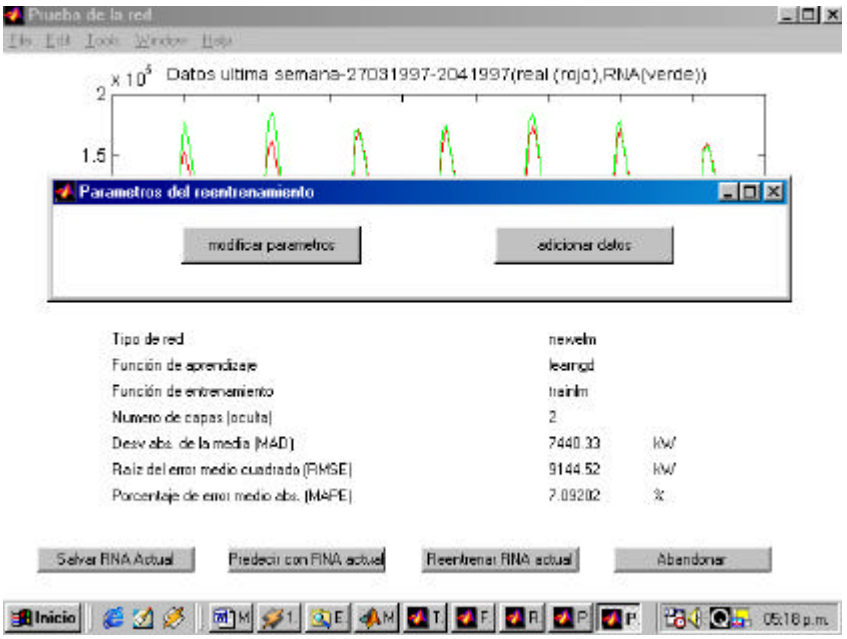


Figura 27. Opción de reentrenamiento de la red neuronal

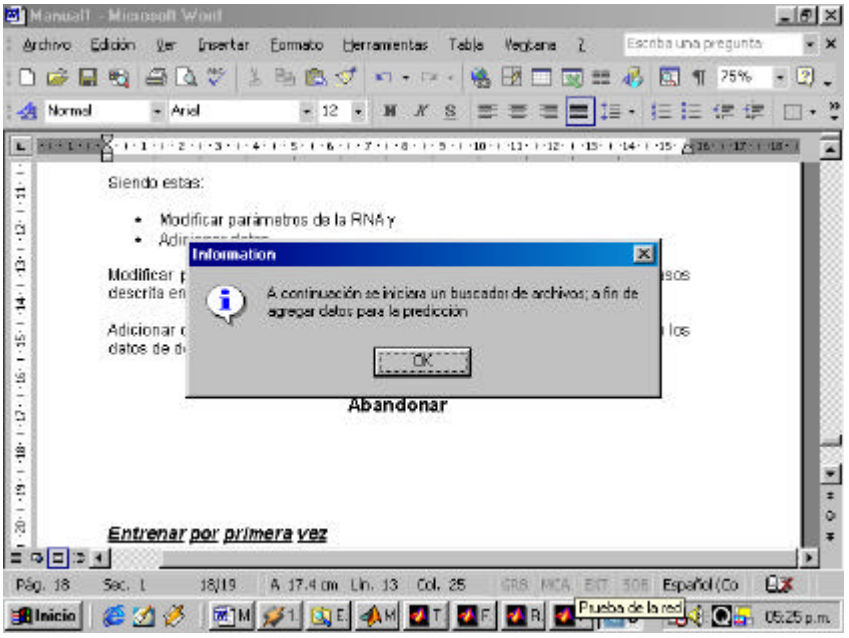


Figura 28. Inicio del proceso de adición de datos

Los pasos descritos anteriormente son los realizados por primera vez frente al programa; cuando se ha realizado la primera carga de datos y su análisis estadístico, la información queda en la memoria para ser retomada en caso de abandonar el proceso.

Cuando ya se ha realizado entrenamientos y se reinicia el programa, el botón *Entrenar por primera vez* de la ventana inicial queda habilitado, permitiendo iniciar el proceso ingresando parámetros de la red neuronal para su entrenamiento o agregar datos a la serie existente o cargando datos nuevos (con el botón *Cargar datos de demanda*), permitiendo esta última opción iniciar un proceso nuevo de predición con otros datos. La ventana puede ser apreciada en la figura 29. En resumen, al salir del proceso intencionalmente o por accidente y reiniciar el programa se activara la opción *Entrenar por primera* vez, siendo posible recuperar las componentes (TCI) de la serie de tiempo saltando inmediatamente hasta la ventana selección de casos.

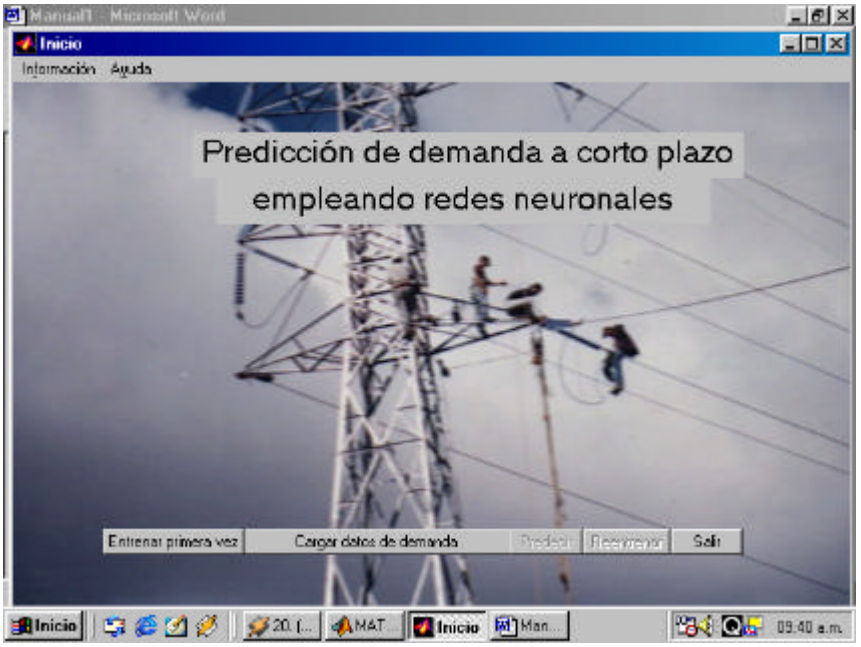


Figura 29. Activación de la opción Entrenar por primera vez

A partir de allí el proceso sigue igual a lo explicado anteriormente, cuando se realizó por primera vez.

Una vez se han realizado predicciones, se activan los botones *Predecir* y *Reentrenar*, de la ventana inicial mostrada en la figura 29. Al dar click en el botón *Predecir* se recupera la última red neuronal guardada en la predicción anterior, a fin de reentrenar con nuevos datos o parámetros (botón *Reentrenar*), realizar predicciones de la semana siguiente o posteriores a los datos (botón *Predecir*) o simplemente regresar a la ventana anterior (botón *Abandonar*). En la figura 30 se ilustra el resultado de una prueba de predicción en la cual se salvó la red neuronal como *prueba4.mat*, indicación que aparece en la parte superior de la ventana.

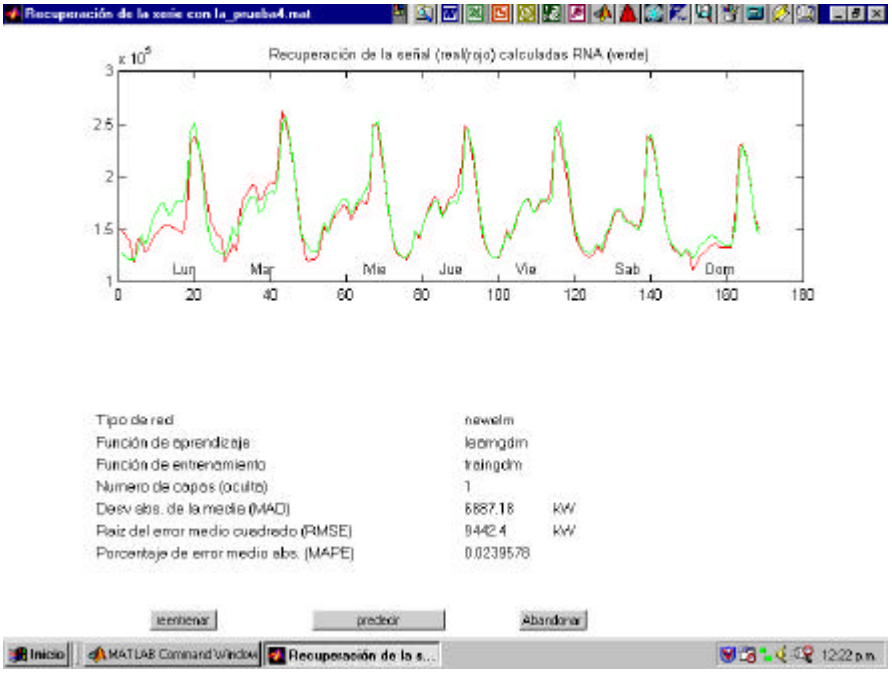


Figura 30. Inicio predicción y reentrenamiento de una red salvada

Al iniciar la predicción, el paso siguiente seria determinar los datos a utilizar durante esta (usados en el entrenamiento o nuevos datos)

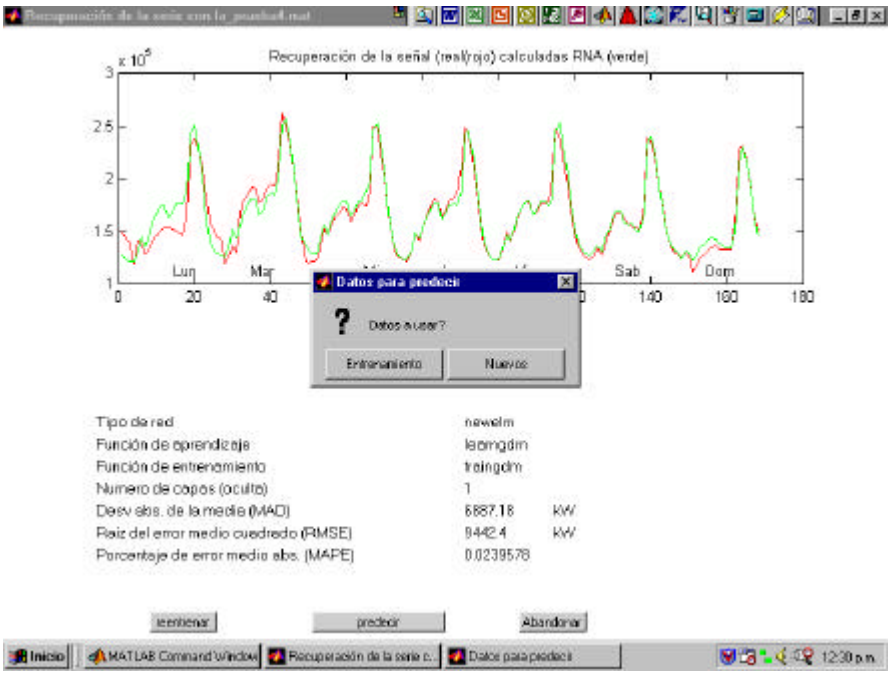
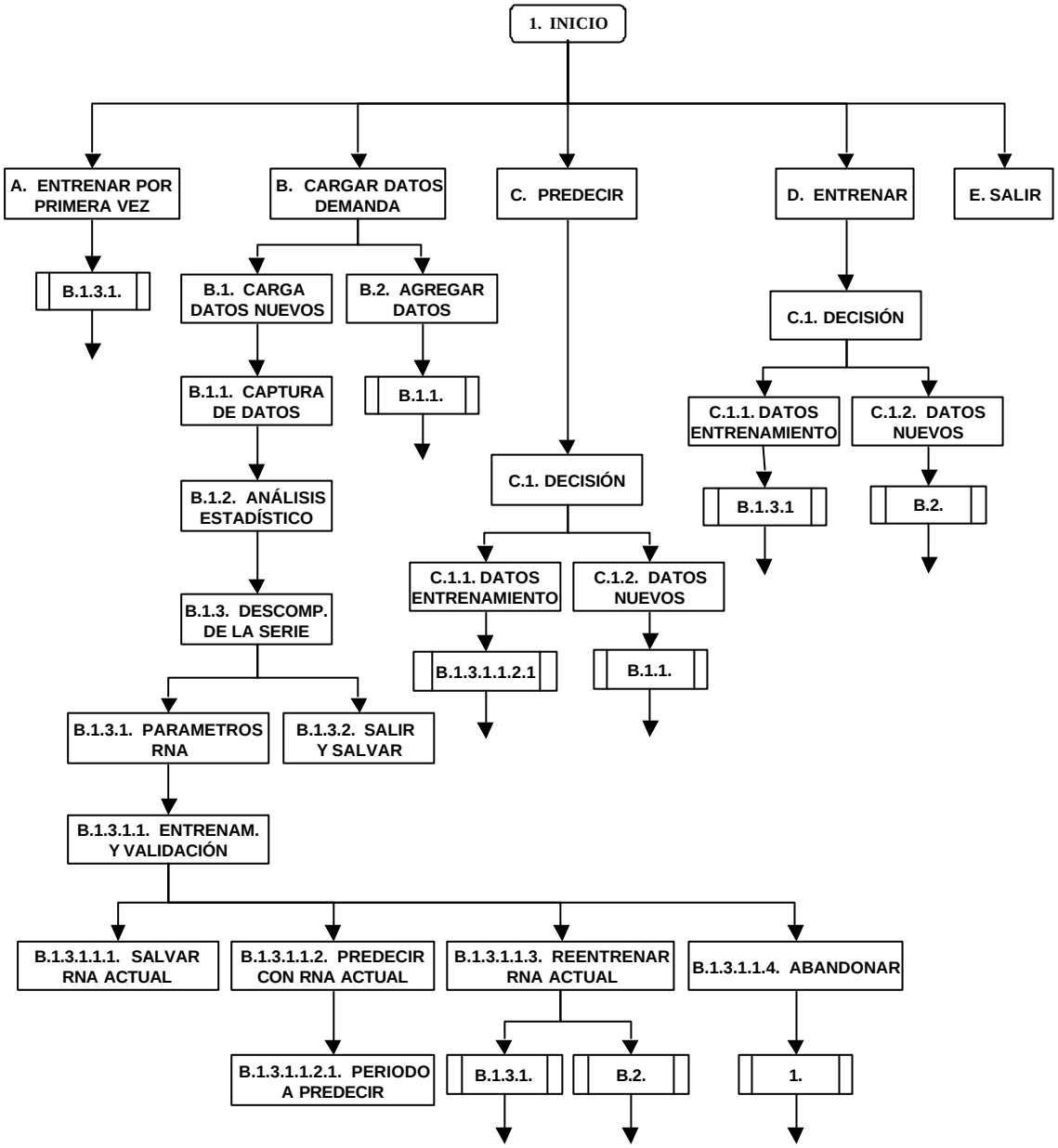


Figura 31. Selección de datos para iniciar predicción

Al señalar como datos de trabajo los datos de entrenamiento en la opción *Entrenamiento* el prototipo le preguntara por el periodo a predecir como se mostró en las figuras 24 y 25; Al seleccionar la opción *Nuevos* abrirá la interfase de captura de datos y se realizaran los pasos explicados anteriormente correspondientes a las figura 7 a 16.

A continuación en el numeral 3 se presenta un flujograma del proceso de predicción a realizar en el prototipo y en el numeral 4 una breve descripción de las rutinas que se ejecutan dentro del mismo.

4. FLUJOGRAMA DEL PROCESO DE PREDICCIÓN REALIZADO POR EL PROTOTIPO



5. DESCRIPCIÓN DE LOS PROCESOS EN LOS SCRIPTS EL PROTOTIPO "PREDICCIÓN"

"Predicción" es un prototipo desarrollado en MATLAB, apoyado en Borland DELPHI, para mejorar la presentación en cuanto a interfaz del usuario. El diagrama de bloques mostrado al final del documento permite representar las tareas desarrolladas durante el proceso de predicción. Para una mejor comprensión del funcionamiento del prototipo, se presenta a continuación una descripción de los scripts desarrollados en MATLAB, en orden alfabetico.

Acerca: Incorpora la interfase de ayuda explicando el objeto del proyecto, la metodología empleada.

Ajust: Permite modificar la información referente a los parámetros: número de capas, cantidad de épocas, Objetivo de la red y tasa de aprendizaje

Borrado: Dicho script verifica la existencia de los archivos: *estructura.txt* (tipo de datos), *ciclo.txt*, *fechas.txt*, *actual.mat* (la red de trabajo), *irreg.txt*, *irreg1.txt*, *irregula1.txt*, *irrtot.txt*, *resultados.txt*, *tendencia.txt*, *pesos.txt*, *inicioproc.txt*, *resultentrena.txt*, *resultpruebagca.txt*, *resultpruebagca.csv* y realiza su eliminación de la carpeta de trabajo.

Bucle: Permite la entrada de datos nuevos y/o agregar datos. Si se entran datos nuevos se eliminan los datos existentes en los archivos *datosentre.txt*, *findatos.txt*, *fechasfin.txt*, *selecunid.txt*, *unidades.txt*; en caso de existir estos. Adicionalmente ejecuta *borrado*, llama la interfase de carga de datos (*u1.exe*), elimina *findatos.txt* y finalmente llama el script *estad*. Si se agregan datos se ejecuta *borrado*, la interfase de carga (*u2.exe*) y finalmente el script *estad*.

Creaserie24: Determina la fecha inicial de acuerdo al número de semanas completas a partir de la ultima fecha, de acuerdo a dicha fecha se establece el vector de días de la semana correspondiente a los datos usados, organiza los

datos y pesos por semanas, determina el ciclo mediante un filtro de primer orden, la tendencia se ajusta a tres modelos; lineal, cuadrático y exponencial usando la norma de dichos ajustes para determinar cual modelo se aproxima más a los datos, extraída la tendencia y el ciclo pasamos a manejar la irregularidad.

Dicha irregularidad se ajusta mediante una modificación del algoritmo de suavizado exponencial desarrollada por Brown (la modificación consiste en un aumento en 48 unidades de la constante alfa), adicionalmente permite la revisión de las graficas resultado de la descomposición (*grafent1*, *grafent2*, *grafent3*, *grafent4*, *grafval*, *grafval2*, *grafval3*), permite salvar los datos hasta el momento y continuar con la ejecución de *paramet* o cerrar todo y retornar a la ventana principal de control (*princip*).

Decis: Recupera el comportamiento de las redes neuronales salvadas; graficas de recuperación y tasas de error. Permitiendo al usuario decidir que hacer con la red salvada: Reentrenar (*reent*), predecir(*predice1*), cerrar (limpia variables).

Entre: Carga a partir de *paramet* las variables a llevar a la RNA, recupera la componente irregularidad, dividiéndola en dos grupos, normaliza la entrada a la red, determina las características de la RNA a usar, inicializa esta mediante el algoritmo de Nguyen-widrow, entrena la RNA, verifica el comportamiento de la RNA y ejecuta *presresul.*

Estad: Verifica la existencia de RNA salvadas en caso de encontrarlas las elimina, carga las características de los datos (*estructura.txt*, *selecunid.txt*, *festivos.txt*, *findatos.txt*), determina las fechas involucradas en el archivo de datos; si no se incluyen las calcula a partir del día inicial y la cantidad de filas de los datos, en caso de hallarse incluidas en los datos simplemente toma dicha columna, siendo salvadas (*fechas.txt*), luego pasa a determinar la ubicación de las fechas obligatorias.

Posteriormente determina la ubicación de la pascua para ello se toma el año de

los datos a utilizar se extrae el residuo entero respecto a 19 al cual se le aplica una constante igual a 11; dicha operación se sustrae de 225, dicha operación será mayor a 50 por ello se introduce a un ciclo el cual disminuye la variable en partes múltiplos de 30, con dichos cálculos verificamos el valor respecto a 31 si es mayor la fecha de la pascua corresponde a Abril en caso contrario pertenece a Marzo; dicho algoritmo se obtuvo de *http://www.rog.nmm.ac.uk/leaflets/easter/easter.html*. Ubicada la pascua pasa a determinar el resto de la Semana Santa y las demás fiestas litúrgicas de acuerdo al calendario litúrgico.

Determinadas las fiestas del calendario colombiano verifica su ubicación en la semana a fin de ubicarse de acuerdo a la Ley Emiliani. Ubicadas las fiestas en su días correctos se compara la base de datos de las fechas de los datos con la correspondiente a los festivos si se encuentran correspondencias (fechas festivas) se coloca un peso igual a 0.25 en caso contrario corresponde a 1, posteriormente se verifica cuales son sábados y domingos para ubicar los pesos correspondientes (0.75 y 0.5 respectivamente).

A continuación se normalizan los datos, de acuerdo a sus unidades y a 2.5 desviaciones estándar, siendo divididos posteriormente en 80 y 20 % para entrenamiento y objetivo respectivamente, finalmente ejecuta creaserie24

Frame: De acuerdo a la cantidad de neuronas de la capa oculta introducidas en *paramet* activa tantas casillas de entrada de datos para las características de la RNA.

Grafent1: Determina el máximo y mínimo de los datos graficándolos de igual manera funciona el ciclo.

Grafent2: Toma el comportamiento determinado para la tendencia por Creaserie24 y lo grafica

Grafent3: Recupera el comportamiento de la irregularidad de Creaserie24 y

grafica.

Grafent4: Compone nuevamente la serie y la grafica junto a los datos originales.

Grafval: Grafica los datos, ciclo y tendencia de los datos objetivo.

Grafval2: Grafica la tendencia de los datos objetivo

Grafval3: Grafica la irregularidad de los datos objetivo

Habilita: En *paramet* en caso de seleccionar manual permite la entrada de datos para las características de la RNA.

Paramet: Permite la entrada de características a la RNA, presentando la opción manual (*habilita*), automática (toma unas características predeterminadas) y todas (*pruebagca.m*) adicionalmente permite realizar ajustes avanzados (*ajust*).

Predice: Carga las variables correspondientes a tendencia y ciclo, interrogando al usuario acerca del periodo a predecir, simula el periodo definido y envía los resultados al archivo *resultprediccion.csv*. Posteriormente presenta las opciones: Reiniciar proceso (retorna a *princip*), Abandonar (Elimina variables y retorna a *princip*), Regresar (retorna a *presresul*).

Predice1: Inicia predicción interrogando acerca del grupo de datos a utilizar: nuevos o entrenamiento. Si es entrenamiento: carga *predice.* Si son nuevos: elimina los archivos de manera similar a *borrado*, carga la interfase de captura de datos (*u3.exe*), realiza los procesos de los script *estad*, *creaserie24*, carga las variables para la red neuronal y ejecuta el entrenamiento.

Presresul: Prueba la red, determina su error y grafica su comportamiento, permite realizar con la RNA de trabajo las siguientes opciones: Salvar la RNA actual (*salvar*), Predecir con RNA actual (*predice*), Reentrenar RNA actual (*reent*), Abandonar (cierra todo y regresa a *princip*).

Princip: Es la interfase de bienvenida de MATLAB, en el menú de opciones proporciona información de los desarrolladores y procesos de la red neuronal. Sus opciones son: Cargar datos de demanda (*bucle*), entrenar por primera vez (carga ultima RNA y sigue a *paramet*; solo se activa si se descompone la serie), Salir (Cierra Matlab), Predecir (*decis*) y Reentrenar (*decis*). Las dos últimas opciones necesitan una red neuronal ya salvada.

Pruebagca: Crea los archivos para guardar el comportamiento de cada modelo, Simula cada modelo y guarda su comportamiento mediante *savegca*. Luego de la simulación de todos los modelos inicia una interfase de captura (*u4.exe*) de los parámetros del modelo que a juicio del operador es el más acertado; con dichos parámetros determina la red neuronal a utilizar en *entre.*

Reent: Modifica el comportamiento de la red neuronal para ello le permite al operador: Adicionar datos ejecutando *borrado*, la interfase de captura de datos (*u2.exe*) y *estad*. Modificar parámetros ejecutando *paramet.*

Salvar: Interroga al operador por el nombre con el cual almacenar la red neuronal, crea los formatos para la escritura de los resultados del entrenamiento con la opción todos, escribe la información en los respectivos formatos.

Savegca: Escribe la información en los formatos creados previamente.

ANEXO B. DESCRIPCIÓN DE LAS FUNCIONES UTILIZADAS EN MATLAB PARA RNA'S

DESCRIPCIÓN DE LAS FUNCIONES UTILIZADAS EN MATLAB PARA RNA'S

Red tipo Backpropagation: La red neuronal Backpropagation presenta una gran variedad de opciones de configuración, dependiendo de la necesidad de aprendizaje y de la aplicación que se este desarrollando.

newff: Crea una red tipo Backpropagation, requiere que le sean especificados los siguientes parámetros

newff: (PR,[S1 S2...SNl],{TF1 TF2...TFNl},BTF,BLF,PF)

PR : Rx2 Matriz de valores máximos y mínimos de cada uno de las R neuronas de entrada.

Si : Número de neuronas para cada una de las capas.

TFi : Función de transferencia a utilizar en cada una de las capas, por defecto utiliza tansig

BTF : Algoritmo de entrenamiento a utilizar, por defecto utiliza trainlm

BLF : Función de actualización de los pesos, por defecto utiliza learngdm.

PF : Función para evaluar el desempeño de la red, por defecto utiliza mse.

Los siguientes fueron los algoritmos de entrenamiento que se utilizaron en el ejemplo de control de voltaje por inyección de reactivos en una barra remota y en la aplicación de predicción de consumo de carga durante sus respectivos procesos de aprendizaje hasta que se encontró uno que brindara un aprendizaje óptimo, para cada uno de ellos utilizando la red Backpropagation:

1. Traingd: Algoritmo de pasos descendientes, que actualiza pesos y ganancias variándolos en la dirección negativa del gradiente de la función del error. Es un algoritmo de aprendizaje muy lento, que requiere de la siguiente sintaxis:

net.trainParam.epochs: Máximo número de iteraciones para obtener convergencia

net.trainParam.goal: Error máximo permitido

net.trainParam.lr: Rata de aprendizaje

net.trainParam.max_fail: Máximo número de fallas

net.trainParam.min_grad: Mínimo rendimiento del gradiente

net.trainParam.show: Intervalo de visualización de los resultados

net.trainParam.time: Máximo tiempo de entrenamiento en segundos

Con este algoritmo el aprendizaje de la red se detendrá si el número de iteraciones excede el comando net.trainParam.epochs, si se alcanzó el valor del error propuesto como meta, si la magnitud del gradiente es menor que net.trainParam.min_grad, o si el tiempo d entrenamiento supera el valor de net.trainParam.time.

2. Traingdm: Equivale al algoritmo tradicional, más un nuevo coeficiente de momentum, que interviene en el proceso de actualización de los pesos. Si el error de la red en una iteración dada, excede el valor del error en la iteración anterior, en un valor mayor al definido por un radio de cobertura dado el que puede determinarse por medio de la función max_perf_inc y que está típicamente alrededor de 1.04, los nuevos pesos y ganancias son descartados y el coeficiente de momentum mc es fijado en cero.

La sintaxis de este algoritmo es igual a la utilizada para el algoritmo traingd, más

un nuevo comando que permite modificar el coeficiente de momentum

net.trainParam.mc: Valor fijado para el coeficiente de momentum

3. Traingda: Algoritmo de Gradiente Descendiente, que emplea una rata de aprendizaje adaptiva durante el proceso de entrenamiento. La rata de aprendizaje varía entre 0.01 y 1, una rata de aprendizaje muy pequeña torna lento el aprendizaje, pero si se incrementa demasiado el aprendizaje puede tornarse inestable y crear divergencia, por esto la función traingda varía la rata de aprendizaje tratando de sacar provecho de la inclinación del gradiente en cada momento; su gran desventaja es que los pesos iniciales varían muy poco así se encuentren distantes de los valores de convergencia. La sintaxis de este el algoritmo es la siguiente:

net.trainParam.epochs: Máximo número de iteraciones para obtener convergencia

net.trainParam.goal: Error máximo permitido

net.trainParam.lr: Rata de aprendizaje inicial

net.trainParam.lr_inc: Porcentaje que incrementa la rata de aprendizaje cuando el error disminuye

net.trainParam.lr_dec: Porcentaje en que es decrementada la rata de aprendizaje cuando el error aumenta

net.trainParam.max_fail: Máximo número de fallas

net.trainParam.max_perf_inc: Máximo incremento del rendimiento

net.trainParam.min_grad: Mínimo rendimiento del gradiente

net.trainParam.show: Los resultados son visualizados siempre que transcurre este

número de iteraciones.

net.trainParam.time: Máximo tiempo de entrenamiento en segundos

4. Trainrp: Las redes multicapa, utilizan típicamente una función de transferencia sigmoidal (ver capítulo 1) en las capas ocultas, estas funciones comprimen un infinito rango de entradas, dentro de un finito rango de salidas, además se caracterizan porque su pendiente tendera cada vez más a cero, mientras más grande sea la entrada que se le presenta a la red, esto ocasiona problemas cuando se usa un algoritmo de entrenamiento de pasos descendientes, porque el gradiente empieza a tomar valores muy pequeños y por lo tanto no habrán cambios representativos en los pesos y las ganancias, así se encuentren bastante lejos de sus valores óptimos. El propósito del algoritmo Backpropagation Resileint (RPROP) es eliminar este efecto en la magnitud de las derivadas parciales. En este algoritmo solamente el signo de la derivada es utilizado para determinar la dirección de actualización de los parámetros, la magnitud de las derivadas no tiene efecto en la actualización. La magnitud en el cambio de cada peso es determinada por separado; el valor del incremento de pesos y ganancias es determinado por el factor delt_inc, así la derivada parcial del error con respecto a los pesos tenga el mismo signo durante dos iteraciones sucesivas; el valor de decremento está determinado por el factor delt_dec así la derivada del error con respecto a los pesos haya cambiado de signo con respecto a la anterior iteración; si la derivada es cero, entonces el valor actualizado se conserva; si los pesos continúan cambiando en la misma dirección durante varias iteraciones, la magnitud de cambios de los pesos se decrementa.

La sintaxis de este algoritmo se resume a continuación:

net.trainParam.epochs: Máximo número de iteraciones del entrenamiento

net.trainParam.show: Intervalo de visualización de los resultados

net.trainParam.goal: Error deseado

net.trainParam.time=inf: Máximo tiempo de entrenamiento en segundos

net.trainParam.min_grad: Mínimo rendimiento del gradiente

net.trainParam.max_fail: Máximo número de fallas

net.trainParam.lr: Rata de aprendizaje

net.trainParam.delt_inc: Incremento en la actualización de pesos

net.trainParam.delt_dec: Decremento en la actualización de pesos

net.trainParam.delta0: Incremento inicial en la actualización de pesos

net.trainParam.deltamax: Máximo cambio en los pesos

5. Trainbfg: Algoritmo alternativo que emplea la técnica del gradiente conjugado, su expresión matemática se deriva del método de Newton, con la ventaja de que no es necesario computar las segundas derivadas; este algoritmo requiere mas capacidad de almacenamiento que el algoritmo tradional, pero generalmente converge en menos iteraciones. Requiere de un cálculo aproximado de la matriz Hessiana, la cual es de dimensiones n2 x n2, donde n la cantidad de pesos y ganancias de la red; para redes que involucren una gran cantidad de parámetros es preferible emplear el algoritmo trainrp.

net.trainParam.epochs: Máximo número de iteraciones del entrenamiento

net.trainParam.show: Número de iteraciones entre las cuales se muestran resultados

net.trainParam.goal: Error deseado

net.trainParam.time=inf: Máximo tiempo de entrenamiento en segundos

net.trainParam.min_grad: Mínimo rendimiento del gradiente

net.trainParam.max_fail=5: Máximo número de fallas

net.trainParam.searchFcn 'srchcha' Nombre de la rutina de búsqueda lineal a utilizar.

net.trainParam.scal_tol: Se divide entre el valor de Delta para determinar la tolerancia para la búsqueda lineal.

net.trainParam.alpha: Factor de escala que determina una reducción suficiente en el desempeño.

net.trainParam.beta: Factor de escala que determina un tamaño de paso suficientemente grande.

net.trainParam.delta: Tamaño de paso inicial en el intervalo de localización de paso.

net.trainParam.gama: Parámetro para evitar pequeñas reducciones en el desempeño.

net.trainParam.low_lim: Límite inferior en el cambio del tamaño del paso.

net.trainParam.up_lim: Límite superior en el cambio del tamaño del paso.

net.trainParam.maxstep: Máximo longitud de paso.

net.trainParam.minstep: Mínima longitud de paso; por defecto es 1.0e-6

net.trainParam.bmax: Máximo tamaño de paso.

6. Trainlm: Algoritmo que actualiza los pesos y las ganancias de acuerdo a la optimización de Levenberg-Marquardt. Es el algoritmo más rápido para redes Backpropagation; tiene la desventaja de requerir de un set de entrenamiento lo más estándar posible, pues de otra forma solo aproximará correctamente valores que se encuentren dentro de los patrones de aprendizaje. Si el set de entrenamiento es muy extenso, se recomienda reducir el Jacobiano.

La sintaxis de este algoritmo es la siguiente:

net.trainParam.epochs: Máximo número de iteraciones del entrenamiento

net.trainParam.goal: Error deseado

net.trainParam.lr: Rata de aprendizaje

net.trainParam.max_fail: Máximo número de veces que falla el valor de Mu

net.trainParam.mem_reducFactor de fraccionamiento de Jacobiano para ahorrar memoria

net.trainParam.min_grad: Mínimo rendimiento del gradiente

net.trainParam.show:Intervalo de visualización de los resultados.

net.trainParam.time: Máximo tiempo de entrenamiento en segundos

tr.mu: Valor del Mu adaptivo

ANEXO C. DETERMINACIÓN DE LAS FIESTAS LITÚRGICAS

DETERMINACIÓN DE LAS FIESTAS LITÚRGICAS

La base para la determinación de las fiestas litúrgicas movibles como Semana Santa, es la el Domingo de Pascua el cual se fija cada año según la luna llena del equinoccio de primavera, el cual puede oscilar del 22 de marzo al 25 de abril.

Después de este Domingo, a los 50 días se celebra el Domingo de Pentecostés que concluye la cincuentena pascual. Su preparación, la Cuaresma, da comienzo el Miércoles de Ceniza, 44 días antes del Triduo Pascual que es, por el Misterio de la Resurrección, la fuente de luz que transfigura el año entero en "Año de gracia del Señor" (Lc 4,19).

Por esta razón la Semana Santa tiene fecha variable; desde el Concilio de Nicea (año 325) en donde se decidió que la Pascua de Resurrección se celebrase el Domingo después de la primera luna llena que siguiera al equinoccio de primavera, a fin de que en cualquier Semana Santa existiera luna llena.

Desde entonces eminentes matemáticos han obtenido diferentes algoritmos para calcular la fecha de la Pascua, el que se aplica en el programa y se explica aquí apareció publicado en 1876 en el Butcher's Eclesiastical Calendar y es válido sin excepciones para todo el calendario gregoriano.

El método del Butcher's consiste en hacer divisiones enteras entre números tomando el cociente y residuo como se aplica en la siguiente tabla ejemplo para el año 2000.

Otro método famoso se debe a **Gauss** y es válido entre 1900 y 2099.

1. Sea Y el año. (aplicándolo al 2000) N=Y-1900=100
2. Se divide N entre 19, sea A el resto división: A=5
3. Se divide (7A+1) entre 19. Sea B el cociente entero: B=1

4. Se divide (11A+4-B) entre 29. Sea M el resto: M=0

5. Se divide N entre 4. Sea Q el cociente: Q=25

6. Se divide (N+Q+31-M) entre 7. Sea W el resto: W=2

7. La fecha de la Pascua es R=25-M-W. Ej:23.

- Si el resultado R es positivo, caerá en Abril.

- Si el resultado es negativo, caerá en Marzo, interpretando 0 como 31 de Marzo, -1 como 30 de Marzo y así hasta -9 que es el 22 de Marzo es decir P=31+R . Entonces la Pascua para el año 2000 cae el 23 de Abril.

Tabla 7. Determinación de la pascua del 2000 por el método del Butcher's

OPERACIÓN	VALOR	COCIENTE	RESIDUO
1.Divide el año por 19	105,2631	105	a=5
2.Divide el año por 100	20	b=20	c=0
3.Divide b por 4	5	d= 5	e=0
4.Divide (b+8) por 25	28	f= 1	----
5.Divide (b-f+1) por 3	20	g=6	----
6.Divide (19a+b-d-g+15) por 30	119	-----	h=29
7.Divide c por 4	0	i= 0	k=0
8.Divide 32+2e+2i-h-k por 7	3	----	l=3
9.Divide a+11h+22l por 451	390	m=0	----
10.Divide h+l-7m+114 por 31	146	n=4	p=22
11.Dia del mes p+1=23	mes=4 (ABRIL)	----	

ANEXO D. ARCHIVOS EJEMPLO EMPLEADOS EN LAS PRUEBAS 1 Y 2

PRUEBA CON MODELO AUTOREGRESIVO

```
clc
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&')
disp('Prueba de predicción con modelo Autorregresivo mejorado (ARX)')
disp('Se predecirá la primera semana de septiembre de 2002 a partir de las 15 semanas')
disp('anteriores empleando la irregularidad')
disp('Los datos para el modelado estan en datosprueba.m')
disp('        ')
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&')
%datosprueba2
u1=[sem1;sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14];
y1=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15];
u2=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15];
y2=[sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15;sem16];
zm=[y1 u1];
zv=[y2 u1];

% la mayoria de modelos presentan la siguiente estructura
% y(t) + A1*y(t-1) + ... + Ana*y(t-na) = B1*u(t-nk) + ... + Bnb*u(t-nb-nk+1)
% primero determinamos el retardo (nk), probando con valores entre 1 a 100.
% dejando fijo el orden del polinomio (na=nb=2), en este caso es de orden 2,
% con dos polos y dos ceros

n=input('orden máximo de prueba del polinomio (max=100):');
V=arxstruc(zm,zv,struc(2,2,1:n));

% Con lo anterior se selecciona el retardo mas aconsejable
[nn,Vm]=selstruc(V,0);
nk=nn(3);

% Ahora probamos los ordenes del polinomio que mas se ajustan, haciendolos variar
% en este caso según el valor de n

V=arxstruc(zm,zv,struc(1:n,1:n,nk));
disp('    ');
disp('el modelo tiene la forma general:')
disp('y(t) + A1*y(t-1) + ... + Ana*y(t-na) = B1*u(t-nk) + ... + Bnb*u(t-nb-nk+1)')
disp('      ')
disp('Las mejores opciones de na,nb y nk son entonces:')
nn=selstruc(V,0)
```

```
th=arx(zm,nn);
disp('Los coeficientes del modelo (A y B) son:')
present(th)

disp('presione enter')
pause
clf
disp('Graficando Validación del modelo (semana del 25 al 31 de Agosto de
2002)')
disp('    ');
yv=predict(zv,th,1);
yv1=yv.*cicloprueba(337:2688).*tendenciaprueba(337:2688);
subplot(2,1,1)
plot(yv1(2185:2352)*0.001,'b')
title('Modelo en validación:(ARX-irreg)/azul:pronost.- rojo: real -
verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2521:2688))*0.001,'r')
grid on
ev=totalprueba(2521:2688)-yv1(2185:2352);
subplot(2,1,2)
plot(ev*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
MAPEv=((sum(abs(ev)./totalprueba(2521:2688)))/168)*100;
RMSEv=sqrt(sum((ev).^2)/168)*0.001;

disp('presione enter')
pause
disp('Graficando Predicció del modelo (semana del 1 al 7 de Septiembre de
2002)')
disp('   ');
yp=predict(zv,th,2);
yp1=yp.*cicloprueba(505:2856).*tendenciaprueba(505:2856);

clf
subplot(2,1,1)
plot(yp1(2185:2352)*0.001,'b')
title('Predicción del modelo:(ARX-irreg)/azul:pronost.- rojo: real -
verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2689:2856))*0.001,'r')
grid on
ep=totalprueba(2689:2856)-yp1(2185:2352);
subplot(2,1,2)
plot(ep*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
```

```
MAPEp=((sum(abs(ep)./totalprueba(2689:2856)))/168)*100;
RMSEp=sqrt(sum((ep).^2)/168)*0.001;
MWerrormax=max(ep)*0.001;
MWerrormin=min(ep)*0.001;
MWherror=(sum(totalprueba(2689:2856))-sum(yp1(2185:2352)))*0.001;
clc
disp('MAPEvalidacion MAPEpredicción  (en %)')
[MAPEv MAPEp]
disp('RMSEvalidacion RMSEpredicción (en MW)')
[RMSEv RMSEp]
disp('ErrorMáximo ErrorMínimo DifEnergía(Real-Pred)  (en MW)')
[MWerrormax MWerrormin MWherror]
```

PRUEBA CON MODELO AUTOREGRESIVO INTEGRADO CON PROMEDIO MOVIL

```
clc
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&')
disp('Prueba de predicción con modelo Autorregresivo mejorado (ARX)')
disp('Se predecirá la primera semana de septiembre de 2002 a partir de las 15 semanas')
disp('anteriores empleando la irregularidad')
disp('Los datos para el modelado estan en datosprueba.m')
disp('        ')
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&')
%datosprueba2
u1=[sem1;sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14];
y1=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15];
u2=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15];
y2=[sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15;sem16];
zm=[y1 u1];
zv=[y2 u1];

% la mayoria de modelos presentan la siguiente estructura
% y(t) + A1*y(t-1) + ... + Ana*y(t-na) = B1*u(t-nk) + ... + Bnb*u(t-nb-nk+1)
% primero determinamos el retardo (nk), probando con valores entre 1 a 100.
% dejando fijo el orden del polinomio (na=nb=2), en este caso es de orden 2,
% con dos polos y dos ceros

n=input('orden máximo de prueba del polinomio (max=100):');
V=arxstruc(zm,zv,struc(2,2,1:n));

% Con lo anterior se selecciona el retardo mas aconsejable
[nn,Vm]=selstruc(V,0);
nk=nn(3);

% Ahora probamos los ordenes del polinomio que mas se ajustan, haciendolos variar
% en este caso según el valor de n

V=arxstruc(zm,zv,struc(1:n,1:n,nk));
disp('    ');
disp('el modelo tiene la forma general:')
disp('y(t) + A1*y(t-1) + ... + Ana*y(t-na) = B1*u(t-nk) + ... + Bnb*u(t-nb-nk+1)')
```

```
disp('      ')
disp('Las mejores opciones de na,nb y nk son entonces:')
nn=selstruc(V,0)
th=arx(zm,nn);
disp('Los coeficientes del modelo (A y B) son:')
present(th)

disp('presione enter')
pause
clf
disp('Graficando Validación del modelo (semana del 25 al 31 de Agosto de
2002)')
disp('    ');
yv=predict(zv,th,1);
yv1=yv.*cicloprueba(337:2688).*tendenciaprueba(337:2688);
subplot(2,1,1)
plot(yv1(2185:2352)*0.001,'b')
title('Modelo en validación:(ARX-irreg)/azul:pronost.- rojo: real -
verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2521:2688))*0.001,'r')
grid on
ev=totalprueba(2521:2688)-yv1(2185:2352);
subplot(2,1,2)
plot(ev*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
MAPEv=((sum(abs(ev)./totalprueba(2521:2688)))/168)*100;
RMSEv=sqrt(sum((ev).^2)/168)*0.001;

disp('presione enter')
pause
disp('Graficando Predicció del modelo (semana del 1 al 7 de Septiembre de
2002)')
disp('   ');
yp=predict(zv,th,2);
yp1=yp.*cicloprueba(505:2856).*tendenciaprueba(505:2856);

clf
subplot(2,1,1)
plot(yp1(2185:2352)*0.001,'b')
title('Predicción del modelo:(ARX-irreg)/azul:pronost.- rojo: real -
verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2689:2856))*0.001,'r')
grid on
ep=totalprueba(2689:2856)-yp1(2185:2352);
subplot(2,1,2)
plot(ep*0.001,'g')
```

```
xlabel('horas del día')
ylabel('MW')
grid on
MAPEp=((sum(abs(ep)./totalprueba(2689:2856)))/168)*100;
RMSEp=sqrt(sum((ep).^2)/168)*0.001;
MWerrormax=max(ep)*0.001;
MWerrormin=min(ep)*0.001;
MWherror=(sum(totalprueba(2689:2856))-sum(yp1(2185:2352)))*0.001;
clc
disp('MAPEvalidacion MAPEpredicción  (en %)')
[MAPEv MAPEp]
disp('RMSEvalidacion RMSEpredicción (en MW)')
[RMSEv RMSEp]
disp('ErrorMáximo ErrorMínimo DifEnergía(Real-Pred)  (en MW)')
[MWerrormax MWerrormin MWherror]
```

PRUEBA CON METODOLOGÍA BOX-JENKINS

```
clc
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
&&&&&&&&&&&&&&&')
disp('Prueba de predicción con la metodología de Box-Jenkins')
disp('Se predecirá la primera semana de septiembre de 2002 a partir de
las 15 semanas')
disp('anteriores empleando la irregularidad')
disp('Los datos para el modelado estan en datosprueba.m')
disp('        ')
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
&&&&&&&&&&&&&&&')
%datosprueba2
u1=[sem1;sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;
sem14];
y1=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14
;sem15];
u2=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14
;sem15];
y2=[sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem1
5;sem16];
zm=[y1 u1];
zv=[y2 u1];

%el modelo BJ emplea nn=[nb nc nd nf nk], de la prueba con arx tenemos
nb=12 y nk=24
%con los otros valores probamos hasta encontrar el mínimo emc,
%mediante un valor inicial de thi, basado en nd=12 y nk=24

nb=48;
nc=24;
nk=24;
thi=bj(zm,[nb nc 24 48 nk],0);
th=bj(zm,thi);

disp('Los coeficientes del modelo son:')
present(th)

disp('presione enter')
pause
clf
disp('Graficando Validación del modelo (semana del 25 al 31 de Agosto de
2002)')
disp('    ');
yv=predict(zv,th,1);
yv1=yv.*cicloprueba(337:2688).*tendenciaprueba(337:2688);
subplot(2,1,1)
plot(yv1(2185:2352)*0.001,'b')
title('Modelo en validación:(BoxJenkins-irreg)/azul:pronost.- rojo: real
- verde: error')
xlabel('horas del día')
ylabel('MW')
```

```
hold on
plot((totalprueba(2521:2688))*0.001,'r')
grid on
ev=totalprueba(2521:2688)-yv1(2185:2352);
subplot(2,1,2)
plot(ev*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
MAPEv=((sum(abs(ev)./totalprueba(2521:2688)))/168)*100;
RMSEv=sqrt(sum((ev).^2)/168)*0.001;

disp('presione enter')
pause
disp('Graficando Predicció del modelo (semana del 1 al 7 de Septiembre de
2002)')
disp('   ');
yp=predict(zv,th,2);
yp1=yp.*cicloprueba(505:2856).*tendenciaprueba(505:2856);

clf
subplot(2,1,1)
plot(yp1(2185:2352)*0.001,'b')
title('Predicción del modelo:(BoxJenkins-irreg)/azul:pronost.- rojo: real
- verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2689:2856))*0.001,'r')
grid on
ep=totalprueba(2689:2856)-yp1(2185:2352);
subplot(2,1,2)
plot(ep*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
MAPEp=((sum(abs(ep)./totalprueba(2689:2856)))/168)*100;
RMSEp=sqrt(sum((ep).^2)/168)*0.001;
MWerrormax=max(ep)*0.001;
MWerrormin=min(ep)*0.001;
MWherror=(sum(totalprueba(2689:2856))-sum(yp1(2185:2352)))*0.001;
clc
disp('MAPEvalidacion MAPEpredicción  (en %)')
[MAPEv MAPEp]
disp('RMSEvalidacion RMSEpredicción (en MW)')
[RMSEv RMSEp]
disp('ErrorMáximo ErrorMínimo DifEnergía(Real-Pred)  (en MW)')
[MWerrormax MWerrormin MWherror]
```

PRUEBA CON MODELO DE ESPACIO DE ESTADOS

```
clc
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&')
disp('Prueba de predicción con la metodología de Espacio de Estados')
disp('Se predecirá la primera semana de septiembre de 2002 a partir de las 15 semanas')
disp('anteriores empleando la irregularidad')
disp('Los datos para el modelado estan en datosprueba.m')
disp('        ')
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&')
%datosprueba1
u1=[sem1;sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14];
y1=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15];
u2=[sem2;sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15];
y2=[sem3;sem4;sem5;sem6;sem7;sem8;sem9;sem10;sem11;sem12;sem13;sem14;sem15;sem16];
zm=[y1 u1];
zv=[y2 u1];

%calculamos el modelo
%n=input('max orden');
n=78;
th=n4sid(zm,1:n);
clc
disp('Los coeficientes del modelo son:')
present(th)
disp('presione enter')
pause

clf
disp('Graficando Validación del modelo (semana del 25 al 31 de Agosto de 2002)')
disp('    ');
yv=predict(zv,th,1);
yv1=yv.*cicloprueba(337:2688).*tendenciaprueba(337:2688);
subplot(2,1,1)
plot(yv1(2185:2352)*0.001,'b')
title('Modelo en validación:(Esp.Estados-irreg)/azul:pronost.- rojo: real - verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2521:2688))*0.001,'r')
grid on
ev=totalprueba(2521:2688)-yv1(2185:2352);
subplot(2,1,2)
plot(ev*0.001,'g')
```

```
xlabel('horas del día')
ylabel('MW')
grid on
MAPEv=((sum(abs(ev)./totalprueba(2521:2688)))/168)*100;
RMSEv=sqrt(sum((ev).^2)/168)*0.001;

disp('presione enter')
pause
disp('Graficando Predicció del modelo (semana del 1 al 7 de Septiembre de
2002)')
disp('   ');
yp=predict(zv,th,2);
yp1=yp.*cicloprueba(505:2856).*tendenciaprueba(505:2856);

clf
subplot(2,1,1)
plot(yp1(2185:2352)*0.001,'b')
title('Predicción del modelo:(Esp.Estados-irreg)/azul:pronost.- rojo:
real - verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2689:2856))*0.001,'r')
grid on
ep=totalprueba(2689:2856)-yp1(2185:2352);
subplot(2,1,2)
plot(ep*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
MAPEp=((sum(abs(ep)./totalprueba(2689:2856)))/168)*100;
RMSEp=sqrt(sum((ep).^2)/168)*0.001;
MWerrormax=max(ep)*0.001;
MWerrormin=min(ep)*0.001;
MWherror=(sum(totalprueba(2689:2856))-sum(yp1(2185:2352)))*0.001;
clc
disp('MAPEvalidacion MAPEpredicción  (en %)')
[MAPEv MAPEp]
disp('RMSEvalidacion RMSEpredicción (en MW)')
[RMSEv RMSEp]
disp('ErrorMáximo ErrorMínimo DifEnergía(Real-Pred)  (en MW)')
[MWerrormax MWerrormin MWherror]
```

PRUEBA CON RED NEURONAL

```
clc
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
&&&&&&&&&&&&&&&')
disp('Prueba de predicción empleando RedesNeuronales')
disp('Se utilizará una red feed-forward de 3 capas internas de 14,7,21
Neuronas')
disp('Se predecirá la primera semana de septiembre de 2002 a partir de
las 15 semanas')
disp('anteriores empleando la irregularidad')
disp('Los datos para el modelado estan en datosprueba.m')
disp('        ')
disp('&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
&&&&&&&&&&&&&&&')
%datosprueba2
u1=[sem1 sem2 sem3 sem4 sem5 sem6 sem7 sem8 sem9 sem10 sem11 sem12 sem13
sem14];
y1=[sem2 sem3 sem4 sem5 sem6 sem7 sem8 sem9 sem10 sem11 sem12 sem13 sem14
sem15];
u2=[sem2 sem3 sem4 sem5 sem6 sem7 sem8 sem9 sem10 sem11 sem12 sem13 sem14
sem15];
y2=[sem3 sem4 sem5 sem6 sem7 sem8 sem9 sem10 sem11 sem12 sem13 sem14
sem15 sem16];
yprueba=[sem4 sem5 sem6 sem7 sem8 sem9 sem10 sem11 sem12 sem13 sem14
sem15 sem16 sem17];

pentre=transp(u1);
tentre=transp(y1);
pvalid=transp(u2);
tvalid=transp(y2);
yprueba=transp(yprueba);
%v.P=pentre;
%v.T=tvalid;

% normalizacion de la entrada a la RNA
[npentre,minpentre,maxpentre,ntentre,mintenre,maxtentre]=premnmx(pentre,t
entre);
[npvalid,minpvalid,maxpvalid,ntvalid,mintvalid,maxtvalid]=premnmx(pvalid,
tvalid);

%ta=input('tasa de aprendizaje=');
n1=40;
n2=40;
n3=40;
ta=0.25;

red=newff(minmax(npentre),[n1 n2 n3
14],{'logsig','tansig','logsig','purelin'},'traingd');
red.trainParam.epochs=2000;
red.trainParam.lr=ta;
red.trainParam.show=50;
red.trainParam.goal=1e-2;
```

```
red=init(red);
%[red,tr]=train(red,npentre,ntentre,[],[],v);
[red,tr]=train(red,npentre,ntentre);

disp('Graficando Validación del modelo (semana del 25 al 31 de Agosto de
2002)')
disp('    ');
nyv=sim(red,npvalid);
yv=postmnmx(nyv,mintvalid,maxtvalid);
yv=transpose(yv);
tvalid=transpose(tvalid);

figure(1)
clf
yv1=transp(yv(2185:2352)).*cicloprueba(2521:2688).*tendenciaprueba(2521:2
688);
subplot(2,1,1)
plot(yv1*0.001,'b')
title('Modelo en validación:(RNA-irreg)/azul:pronost.- rojo: real -
verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2521:2688))*0.001,'r')
grid on
ev=totalprueba(2521:2688)-yv1;
subplot(2,1,2)
plot(ev*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
MAPEv=((sum(abs(ev)./totalprueba(2521:2688)))/168)*100;
RMSEv=sqrt(sum((ev).^2)/168)*0.001;


figure(2)
clf
disp('presione enter')
pause
disp('Graficando Predicció del modelo (semana del 1 al 7 de Septiembre de
2002)')
disp('   ');
nyp=sim(red,ntvalid);
yp=postmnmx(nyp,mintvalid,maxtvalid);
yp=transpose(yp);
yprueba=transpose(yprueba);
subplot(2,1,1)

yp1=transp(yp(2185:2352)).*cicloprueba(2689:2856).*tendenciaprueba(2689:2
856);

clf
subplot(2,1,1)
plot(yp1*0.001,'b')
```

```
title('Predicción del modelo:(RNA-irreg)/azul:pronost.- rojo: real -
verde: error')
xlabel('horas del día')
ylabel('MW')
hold on
plot((totalprueba(2689:2856))*0.001,'r')
grid on
ep=totalprueba(2689:2856)-yp1;
subplot(2,1,2)
plot(ep*0.001,'g')
xlabel('horas del día')
ylabel('MW')
grid on
MAPEp=((sum(abs(ep)./totalprueba(2689:2856)))/168)*100;
RMSEp=sqrt(sum((ep).^2)/168)*0.001;
MWerrormax=max(ep)*0.001;
MWerrormin=min(ep)*0.001;
MWherror=(sum(totalprueba(2689:2856))-sum(yp1))*0.001;
clc
disp('MAPEvalidacion MAPEpredicción  (en %)')
[MAPEv MAPEp]
disp('RMSEvalidacion RMSEpredicción (en MW)')
[RMSEv RMSEp]
disp('ErrorMáximo ErrorMínimo DifEnergía(Real-Pred)  (en MW)')
[MWerrormax MWerrormin MWherror]
```