UNIVERSIDAD DE GRANADA

SISTEMAS INTELIGENTES ADAPTATIVOS PARA APROXIMACIÓN Y PREDICCIÓN UTILIZANDO ARQUITECTURAS AVANZADAS

TESIS DOCTORAL

Luis Javier Herrera Maldonado

2007

Departamento de Arquitectura y Tecnología de Computadores

Editor: Editorial de la Universidad de Granada
Autor: Luis Javier Herrera Maldonado
D.L.: Gr. 1554 - 2007
ISBN: 978-84-338-4403-3

UNIVERSIDAD DE GRANADA

SISTEMAS INTELIGENTES ADAPTATIVOS PARA APROXIMACIÓN Y PREDICCIÓN UTILIZANDO ARQUITECTURAS AVANZADAS

Memoria presentada por

Luis Javier Herrera Maldonado

Para optar al grado de

DOCTOR EN INFORMÁTICA

Fdo. Luis Javier Herrera Maldonado

D. Ignacio Rojas Ruiz y **D. Héctor Pomares Cintas**, Profesores Titulares de Universidad del Departamento de Arquitectura y Tecnología de Computadores

CERTIFICAN

Que la memoria titulada: **"Sistemas Inteligentes Adaptativos para Aproximación y Predicción Utilizando Arquitecturas Avanzadas"** ha sido realizada por **D. Luis Javier Herrera Maldonado** bajo nuestra dirección en el Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada para optar al grado de Doctor en Informática.

Granada, a 5 de Junio de 2007

Fdo. Ignacio Rojas Ruiz
Director de la Tesis

Fdo. Héctor Pomares Cintas
Director de la Tesis

A mis padres

Tesis

Luis Javier Herrera Maldonado

13 de junio de 2007

Índice general

Capítulo 1
Introducción

El modelado en general consiste en una representación conceptual o física a escala de un proceso o sistema (fenómeno), con el fin de analizar su naturaleza, desarrollar o comprobar hipótesis o supuestos y permitir una mejor comprensión del fenómeno real al cual el modelo representa. Así, gran parte de la investigación de áreas como la ingeniería, física, estadística, ciencias económicas, etc. se centran en el estudio de técnicas y métodos para la definición y explicación de fenómenos naturales, económicos, físicos, etc. El objetivo en todo caso es encontrar un modelo como representación de la realidad, que simplificamos de alguna manera para poder estudiarlo y manipularlo.

Una tipología concreta del problema de modelado es aquella en la que se dispone de un conjunto de muestras de entrada y salida del tipo $\{(\vec{x}_i, y_i); i = 1 \dots m / F(\vec{x}_i) = y_i\}$[1], a partir del cual se desea entrenar un modelo que identifique el comportamiento del fenómeno F que provoca esas salidas a partir de las entradas. Una vez que dicha relación ha sido identificada, puede ser usada para la predicción del comportamiento del fenómeno, esto es, para la obtención de nuevas salidas dadas otras nuevas entradas. Las entradas y salidas proporcionadas para un problema dado pueden ser, en general, variables continuas y/o discretas. En el caso en el que las variables tratadas son continuas, nos encontramos ante un problema de regresión o aproximación de funciones, a diferencia de los problemas de clasificación donde la variable de salida es categórica.

La aproximación de una función desconocida a partir de un conjunto de muestras experimentales ha sido y sigue siendo, una parte fundamental en un gran número de disciplinas científicas e industriales. Esta tarea se ha abordado de diversas formas a lo largo de la historia, pasando por los modelos estadísticos, y en las últimas décadas, han tomado un gran auge los modelos de soft-computing, como los modelos basados en lógica difusa y los modelos basados en redes neuronales, así como los métodos de kernel en gran auge actualmente. Un problema típico de modelado de funciones de gran importancia en diversas áreas, es la llamada "predicción de series temporales". En este caso se dispone de una secuencia de muestras (normalmente continuas) y se pretende predecir el comportamiento de dicha serie de muestras a corto o largo plazo. Este problema se suele tratar como un problema de modelado donde se pretenden obtener muestras futuras (salida) en función de muestras anteriores (entradas).

En general los problemas de clasificación reciben una mayor atención por parte de

[1] Pueden encontrarse un gran número de problemas con diversas variables de salida igualmente. En este escrito se consideran tan solo problemas con una sola variable de salida, sin pérdida de generalidad, al poderse tratar los problemas multi-salida como múltiples problemas distintos de una salida.

los investigadores, habiendo en muchos casos un mayor número de trabajos centrados en este segundo tipo de problemas. Sin embargo, el estudio de problemas con variables continuas, pese a poder requerir un mayor esfuerzo en su tratamiento, puede verse como una problemática más general de modelado, ya que en principio cualquier técnica de variables continuas puede ser aplicada de manera casi directa a un problema de variables discretas.

Como se ha comentado, existen un gran número de técnicas y métodos para el modelado de una función a partir de un conjunto de valores de entrada y salida: modelos estadísticos, modelos basados en lógica difusa y los modelos basados en redes neuronales, métodos de kernel, etc. En cualquiera de los casos, y dentro del marco general del aprendizaje automático, el mayor problema que se presenta en general a la hora de tratar un problema de este tipo es la "maldición de la dimensionalidad" [Bellman, 1961]. Este término se refiere al problema causado por el crecimiento exponencial en volumen asociado con añadir dimensiones a un espacio (matemático). Esto tiene repercusiones a nivel de la dificultad en el entrenamiento de modelos, tanto a nivel de efectividad como a veces a nivel de coste computacional.

Partiendo de esto, en primer lugar, dado cualquier problema de modelado y cualquier paradigma de aprendizaje, en general es conveniente el realizar una eliminación de las variables de entrada que son innecesarias para poder resolver el problema de forma más efectiva. Este proceso se denomina "selección de las variables", y es un problema muy complejo y de difícil solución; especialmente en problemas con variables continuas en las que es más complicado identificar las interrelaciones existentes entre las variables de entrada y entre éstas y la variable de salida.

En un problema de modelado concreto (bien sea de aproximación funcional, de predicción de series temporales, etc.), en principio no es sencillo establecer qué paradigma es más conveniente utilizar ni qué criterios pueden ayudar a resolver ese problema. Además, las fronteras y limitaciones de cada paradigma o metodología y la repercusión de la maldición de la dimensionalidad en su eficacia y eficiencia normalmente son impredecibles. Este escrito, lejos de intentar establecer criterios o comparar diferentes técnicas para el problema planteado, presenta un conjunto de trabajos que pretenden dar solución a distintos problemas específicos, dentro de la amplísima problemática del modelado de funciones, utilizando varias técnicas de modelado continuo.

Primeramente, el grueso de este escrito trata el problema de la maldición de la dimensionalidad en los sistemas neuro-difusos, así como el problema de la pérdida de la interpretabilidad en estos sistemas, que ocurre durante el entrenamiento para problemas de modelado a partir de datos de entrada y salida. Se presentan en sucesivos capítulos varias alternativas a los modelos neuro-difusos tradicionales de Takagi-Sugeno-Kang y de redes de Funciones de Base Radial para resolver estas problemáticas.

Posteriormente se trata el problema de predicción de series temporales utilizando modelos recursivos de diferentes paradigmas y se presenta una técnica general para mejorar este tipo de predicción, comparándose también el rendimiento entre ejemplos de métodos de kernel y modelos difusos.

Finalmente se trata el problema de la selección de variables, ofreciendo una metodología robusta basada en la medida de la información mutua de la Teoría de la Información de Shannon para problemas reales de aproximación funcional, como predicción de series temporales y problemas espectrométricos, donde en general existe una fuerte interrelación entre las variables de entrada, y es muy difícil establecer qué variables son más convenientes conservar y cuáles y cuántas variables eliminar.

Antes de pasar a describir el trabajo de investigación desarrollado, a continuación se hace una descripción más detallada de los capítulos de los que consta la presente memoria:

- **Capítulo 2**: Este capítulo realiza una revisión sobre los conceptos más importantes, necesarios para una compresión adecuada del trabajo expuesto en los siguientes capítulos. En las primeras secciones se introducen los sistemas difusos y el sistema de inferencia difusa, centrándonos sobre todo en los sistemas tipo Takagi-Sugeno-Kang (TSK). Posteriormente se revisan las redes neuronales de Funciones de Base Radial (redes RBF) y la equivalencia de éstas con los sistemas difusos TSK cuando se cumplen un conjunto de condiciones. Finalmente se introducen las Máquinas de Vectores Soporte de Mínimos Cuadrados (Least Squares SVMs, LS-SVMs) que son una herramienta con resultados óptimos para problemas de aproximación funcional y que evitan ciertos problemas en el entrenamiento de las Máquinas de Vectores Soporte (SVMs) tradicionales en este tipo de problemas.

- **Capítulo 3**: Este capítulo sienta las motivaciones de los modelos difusos TSK presentados en los siguientes tres capítulos. El aprendizaje automático de sistemas difusos para problemas de aproximación funcional viene siendo un área importante de investigación en los últimos años. Existen un número de escollos importantes que surgen a la hora de tratar este tipo de problemas de manera automática. Vistas las bases teóricas de los sistemas difusos y de las redes RBF en el tema anterior, este tema introduce brevemente el aprendizaje de los sistemas neuro-difusos TSK, junto con algunas de las opciones existentes en su diseño y los problemas que se presentan en dicho aprendizaje. También se resume el aprendizaje de las redes RBF, que presenta grande similitudes al aprendizaje de los sistemas TSK. Finalmente se introducen los tres modelos propuestos y la motivación en su investigación, con las ventajas que presentan respecto de los modelos neuro-difusos TSK y redes RBF tradicionales.

- **Capítulo 4**: Este capítulo presenta una nueva propuesta de modelo TSK, que denominamos modelo TaSe, que obtiene un sistema eficaz e interpretable, que hace uso del la expansión en serie de Taylor (Taylor Series) de una función en torno a un punto, para aproximar la función objetivo usando un número reducido de reglas. Asimismo, se proporciona una metodología completa de aprendizaje para la obtención de la estructura óptima del modelo TaSe, así como sus parámetros óptimos en los antecedentes y consecuentes. Además este capítulo se complementa con un análisis estadístico sobre efecto en el rendimiento de los parámetros más importantes del diseño de un sistema difuso tipo TSK. Se da un relevancia especial al orden de las reglas del modelo (orden polinómico de los consecuentes: cero, uno y dos). Como se muestra en los resultados del análisis ANOVA, el orden de las reglas usadas es estadísticamente significativo en el rendimiento del modelo. Los resultados muestran la conveniencia de utilizar el modelo TaSe para una amplia gama de problemas de modelado, puesto que puede usar reglas de orden alto para obtener sistemas más sencillos, con reglas interpretables y sin afectar negativamente a la complejidad computacional. Este capítulo engloba los siguientes trabajos publicados:

  - [Herrera et al., 2004b]: Trabajo preliminar sobre el modelo TaSe.
  - [Herrera et al., 2005f]: Trabajo completo sobre el modelo TaSe propuesto, incluyendo una metodología de aprendiza completa.
  - [Herrera et al., 2005c]: Trabajo preliminar sobre el análisis estadístico sobre el efecto en el rendimiento del orden del consecuente de las reglas en los sistemas TSK y el modelo TaSe.

- **Capítulo 5**: En el capítulo anterior (capítulo 4) se estudia cómo se puede proporcionar interpretabilidad a los modelos locales en sistemas difusos basados en grid.

Sin embargo, en sistemas basados en clustering, y en redes de funciones de base radial y modelos neuro-difusos en general, esta tarea es más compleja. Este capítulo presenta un modelo neuro-difuso basado en clustering que mantiene las propiedades de optimización de los modelos locales a la vez que entrena la red optimizándola globalmente. Además, debido a los problemas de falta de transparencia en los modelos basados en clustering, el trabajo propuesto implementa un enfoque de compartición de funciones de pertenencia, con el objetivo de mejorar la trasparencia e interpretabilidad del conjunto de reglas difusas extraídas del modelo. Para la mejora del rendimiento, se implementa también un enfoque apropiado para la inicialización de centros, que supera en rendimiento a los enfoques presentes en la literatura con este objetivo. Este capítulo engloba los siguientes trabajos publicados:

- [Herrera et al., 2005d]: Trabajo preliminar sobre optimización de modelos locales en sistemas TSK basados en clustering.
- [Herrera et al., 2005a]: Trabajo sobre la inicialización óptima de centros en sistemas TSK basados en clustering utilizando el método de equidistribución de errores.
- [Herrera et al., 2005b]: Trabajo preliminar sobre extracción de reglas óptimas e interpretables en sistemas TSK basados en grid y en sistemas TSK basados en clustering.

- **Capítulo 6**: Este capítulo presenta los sistemas difusos Multigrid, como metodología alternativa para evitar el problema de la maldición de la dimensionalidad en la complejidad, de los sistemas difusos basados en grid para problemas complejos. La arquitectura de los sistemas Multigrid permite reducir exponencialmente el número de reglas, mediante el descarte de relaciones irrelevantes entre las variables. Este trabajo presenta un algoritmo de selección de arquitectura para sistemas difusos Multigrid a partir de un conjunto de datos de entrada/salida, que incorpora un proceso de selección de variables. Este algoritmo de selección de arquitectura se complementa con un procedimiento de identificación de la estructura, que obtiene el particionamiento óptimo del espacio en los diferentes grids que forman el modelo Multigrid y sus parámetros (número y posición de las funciones de pertenencia y consecuentes de las reglas). Las ventajas que proporciona esta metodología se verán en la sección de simulaciones, donde cabe resaltar la aplicación al problema de predicción de series temporales CATS, propuesto inicialmente como competición en la conferencia IJCNN'2004 (International Joint Conference on Neural Networks) [Lendasse et al., 2004]. Este capítulo engloba los siguientes trabajos publicados:

  - [Herrera et al., 2004c]: Trabajo preliminar sobre sistemas multigrid para problemas de aproximación funcional básicos.
  - [Herrera et al., 2004a]: Trabajo sobre la aplicación de sistemas multigrid al problema de predicción de series temporales CATS en la conferencia IJCNN'2004.
  - [Herrera et al., 2007a]: Trabajo completo sobre aprendizaje en modelos difusos multigrid y aplicación al problema completo de predicción de series temporales CATS [CAT,].

- **Capítulo 7**: El trabajo presentado en este capítulo muestra la aptitud de dos metodologías diferentes, el modelo TaSe y las Máquinas de Vectores Soporte de Mínimos Cuadrados, para resolver el problema de predicción de series temporales a largo plazo utilizando predicción recursiva. Este capítulo introduce además algunas técnicas

que incrementan el rendimiento de este tipo de modelos avanzados de predicción a un paso (y en general de cualquier modelo de predicción a un paso), cuando se utilizan recursivamente para predicción de series temporales a largo plazo. Este capítulo engloba el siguiente trabajo publicado:

- [Herrera et al., 2007b]: Trabajo completo sobre predicción recursiva a largo plazo utilizando modelos directos a un paso, de las tipologías TaSe y LS-SVMs, incluyendo una técnica de mejora de modelos a un paso cuando se utilizan recursivamente para predicción a largo plazo.

- **Capítulo 8**: Este capítulo propone una metodología de selección de variables hacia atrás para problemas de aproximación funcional, basada en la medida de la información mutua de la teoría de la información de Shannon. Entre las ventajas que presenta la metodología propuesta, destaca la posibilidad de evitar el problema de la maldición de la dimensionalidad que presentan los estimadores de la información mutua, haciendo que el método sea factible para problemas con un gran número de variables de entrada, sin perder su robustez. El rendimiento de la metodología propuesta se muestra mediante ejemplos significativos de predicción de series temporales y datos espectrométricos.Este capítulo engloba el siguiente trabajo publicado:

  - [Herrera et al., 2006]: Trabajo preliminar sobre una metodología de selección de variables hacia atrás para problemas de aproximación funcional, basada en la medida de la información mutua de la teoría de la información de Shannon.

- **Capítulo 9**: Este capítulo extrae finalmente las conclusiones obtenidas a lo largo de los trabajos desarrollados en esta memoria.

- **Apéndice A**: Este apéndice presenta las configuraciones de funciones de pertenencia utilizadas en los capítulos 4, 5 y 6.

- **Apéndice B**: Este apéndice presenta el desarrollo de las derivadas parciales de la función de error cuadrático $J = \sum_{m \in D} \left(y^m - F\left(\vec{x}^m\right)\right)^2$, con respecto a cada coeficiente de los consecuentes de las reglas en un sistema difuso TSK. Este desarrollo es igualmente válido para el desarrollo de las derivadas parciales para el cálculo de los pesos de una red RBF. Utilizando las expresiones para todas los coeficientes, se obtiene un conjunto de ecuaciones lineales que proporcionan los valores óptimos de dichos coeficientes para un conjunto de datos dado D.

Para facilitar la lectura de esta memoria conviene tener en cuenta las siguientes convenciones:

- Cada capítulo se ha dividido en secciones, las cuales suelen incluir uno o más apartados, y éstas a su vez, pueden incluir subapartados. Cuando se haga referencia a cualquier parte del texto se indicará el número del capítulo seguido del de la sección, el apartado, etc. Ejemplo: sección 4.3, apartado 4.3.3.

- Las figuras y tablas están numeradas por capítulos. Ejemplo: figura 6.1, tabla 6.1.

- La expresiones matemáticas también siguen una numeración por capítulos y se expresan mediante sus números entre paréntesis. Ejemplo: (7.1).

- Las referencias bibliográficas se indican mediante los apellidos de los autores seguidos por el año de publicación si el trabajo citado tiene menos de tres autores, o bien, con el apellido del autor principal seguido de la abreviatura *et al.* si el trabajo tiene más autores. Ejemplo: [Koller and Sahami, 1996], [Rojas et al., 2002].

Introduction

Modeling in general is about obtaining a scaled conceptual or physical representation of a process or system (phenomenon), with the objective of analyzing its nature, develop or check hypothesis or assumptions and allow a better comprehension of the real phenomenon that the model represents. Thus, most of the research in areas such as engineering, physics, statistics, economic sciences, etc. are centered in the study of techniques and methods for the definition and explanation of natural, economic, physic, etc. phenomena. The objective in any case is to find a model as a representation of reality, which is simplified somehow, in order to properly study and manipulate it.

A specific typology of the modeling problem, is that one for which a set of input/output samples of the type $\{(\vec{x}_i, y_i); i = 1 \dots m / F(\vec{x}_i) = y_i\}$ [2] is available, for which a model is trained in order to identify the behavior of the phenomenon F, that induced those outputs from the respective inputs. Once this relationship has been identified, it can be used for the prediction of the behavior of the phenomenon, that is, for obtaining new outputs given a set of new input data samples. The inputs and outputs provided for a given problem can be, in general, continuous or discrete variables. In the case that the treated variables are continuous, we face a so-called regression or function approximation problem, in opposite to classification problems, which deal with categorical outputs.

Approximating an unknown function from a set of experimental samples, has been and is still, a fundamental part on a large number of scientific and industrial disciplines. This task has been approached in a number of forms along history, passing from statistical modes, and in the last decades, an increasing attention has been paid to soft-computing methods, such as fuzzy logic and neural networks models, as well as to kernel methods, that are at their very peak in the last years. A well-known function modeling typology, that is very important in several areas, is the so-called "time series prediction". In this case, a sequence of samples (normally continuous) is available, and it is pretended to predict the future behavior of this samples series at short or long term. This problem is normally treated as a modeling problem in which future samples (output) are to be predicted as a function of previous samples (inputs).

In general, classification problems receive more attention from the researchers, and there's normally a larger number of works centered in this second type of problems. However, studying problems dealing with continuous variables, although can demand larger efforts in its treatment, can be seen as a more general type of modeling problems, since in principle any technique that is applicable to continuous variables is also directly valid for discrete variables.

As it was mentioned, there is a large number of techniques and methods that are applicable to the problem of modeling a continuous function from a set of input/output

[2] A large number of problems can be found for which several output variables are considered. This thesis, only considers problems with one output variable, without loss of generality, since multiple-output problems can be treated as multiple problems with one output.

samples: statistical methods, neural networks and fuzzy logic models, kernel methods, etc. In any case, and within the general framework of machine learning, the main problem that this type of problems present is the so-called "curse of dimensionality" [Bellman, 1961]. This term refers to the problem caused by the exponential growth in volume associated with adding input dimensions to a (mathematical) space. This has a very important impact on the training of the models, both on the level of the effectiveness as well as on the level of the computational cost.

From this fact, in first place, given any modeling problem and any learning paradigm, in general it is very convenient to perform the elimination of the variables that show to be useless, in order to solve the problem more effectively. Such process is called "variable selection", and is a very complex problem with a difficult solution; specially in continuous problems in which it is more complicated to identify the existent interrelations among the input variables and among those and the output variable.

In a specific modeling problem (function approximation or time series prediction), in principle it is not easy neither to establish which paradigm is more convenient to use nor which criteria are helpful to solve this problem. Moreover, the frontiers and limitations of each paradigm or methodology and the impact of the curse of dimensionality in their effectiveness and efficiency are normally unpredictable. This thesis, far from trying to establish any criterion or to compare different techniques for the raised problem, presents a set of works that try to bring a solution to a set of different specific problems, inside the open-wide problem area of the modeling of functions using different continuous modeling techniques.

The main part of this thesis deals with the problem of the "curse of dimensionality" in neuro-fuzzy models, as well as with the loss of interpretability that might occur during the training process of this type of systems, when dealing with modeling problems from a set of input/output data. Three different chapters present three modified neuro-fuzzy models, which are based on the traditional Takagi-Sugeno-Kang fuzzy systems and Radial Basis Function Networks.

Next, the problem of recursive time series prediction using different paradigms (the advanced neuro-fuzzy TaSe model and LS-SVMs) is treated. It compares the operation of both methodologies and proposes a general technique to improve this type of prediction, that is independent of the specific paradigm used.

Finally, the last work deals with the problem of variable selection for function approximation problems. It proposes a novel robust methodology based on the Mutual Information criterion of Shannon's Information Theory. It is a backward variable selection procedure that utilizes the Markov blanket concept in order to avoid the curse of dimensionality problem that in general mutual information estimators present. The method is very well adapted for problems with a strong interrelation among the input variables, such as spectrometric data and time series prediction problems, in which it is very difficult to establish which variables are to be selected in order to reduce properly the dimensionality of the problem and obtain an optimal performance.

Before describing the research developed in this thesis, a more detailed description of each of the chapters included is presented now:

- **Chapter 2**: This chapter performs a review of the most important concepts needed for an appropriate comprehension of the work presented in the following chapters. In the first section, fuzzy systems and the fuzzy inference system are introduced, centring the presentation in Takagi-Sugeno-Kang (TSK) systems. Later Radial Basis Function Neural Networks are reviewed (RBF networks), as well as their equivalence to TSK fuzzy systems when a set of conditions are hold. Finally Least Squares

Support Vector Machines (LS-SVMs) are introduced, as a powerful tool for function approximation problems bringing optimal results, and that avoid the problems that traditional SVMs present for this type of problems.

- **Chapter 3**: This chapter states the motivations for the TSK neuro-fuzzy models presented in the following three chapters. The automatic learning of fuzzy systems for function approximation problems is a very important research area in the last years. However, there is a number of problems that arise when dealing with such problems in an automatic way. Starting from the theoretical bases of fuzzy systems and RBF networks in the previous chapters, this chapter briefly introduces the learning process of neuro-fuzzy TSK systems, describing some of the existent alternatives in their design, and the problems that this learning process may present. It is also summarized the learning process of RBF networks, that presents a number of similarities with the learning process of TSK systems. Finally, the three models proposed in the following chapters are presented, with the advantages they present with respect to the traditional TSK and RBF neuro-fuzzy models.

- **Chapter 4**: This chapter presents a novel TSK model, that we call TaSe model, which makes use of the Taylor Series Expansion of a function around a point to approximate the objective function using a reduced number of rules. Furthermore, it is also provided an automatic methodology for obtaining the optimum structure of the TaSe fuzzy system as well as its pseudo-optimal rule-parameters (both antecedents and consequents). Moreover, this chapter is complemented by an ANalysis Of VAriance -ANOVA- study of the effect on the performance, of the parameters considered to be the most relevant ones in the design of a TSK fuzzy system. A special relevance is given to the order of the TSK model (zero, one and two). The results obtained in the ANOVA analysis show that the order of the TSK model is statistically significant with respect to the system performance. The study was performed using a number of examples of function approximation and time series prediction problems taken from the literature. These results show the convenience of using the TaSe model for a wide range of applications, since it is able to use higher order rules to obtain simpler systems, with interpretable rules, and without negatively affecting the computational complexity. This chapter includes the following published works:

  - [Herrera et al., 2004b]: Preliminary work on the TaSe model.
  - [Herrera et al., 2005f]: Complete work on the proposed TaSe model, including a complete learning methodology.
  - [Herrera et al., 2005c]: Preliminary work on the statistical analysis of the effect on the performance of the order of the rule consequents in the design of a TSK systems and the TaSe model.

- **Chapter 5**: The previous chapter (chapter 4) studied how local models can be endowed with interpretability in grid based fuzzy systems. But for clustering based TSK fuzzy systems, or similarly to Radial Basis Function Networks, this is a more difficult task. This chapter presents a modified neuro-fuzzy model that keeps the desirable local optimization properties during the global learning process. Furthermore, a simple approach for membership function sharing is implemented in order to improve the transparency and interpretability of the resulting set of rules. For the improvement of the accuracy, a convenient approach for the initialization of the centers is used, that exceeds in performance the most important approaches present in the literature for this purpose. This chapter includes the following published works:

- [Herrera et al., 2005d]: Preliminary work on the local models optimization in clustering-based TSK fuzzy systems.
- [Herrera et al., 2005a]: Work on the optimal initialization of centres in clustering-based TSK fuzzy models using the error equidistribution method.
- [Herrera et al., 2005b]: Preliminary work on the extraction of optimal and interpretable rules in grid-based and clustering-based TSK fuzzy systems.

- **Chapter 6**: This chapter presents MultiGrid-Based Fuzzy System (MGFS), as an alternative methodology in order to avoid the problem of the curse of dimensionality in the system complexity in grid-based fuzzy systems for complex problems. The MGFS model keeps the advantages of the traditional Grid-Based Fuzzy Systems (GBFS), and overcomes the problem inherent to all GBFSs when dealing with high dimensional input data. Thus the MGFS model keeps interpretability and low computational cost. A novel architecture selection algorithm for MGFSs that allows performing input variable selection is proposed. It identifies the sub-optimal architecture, according to a provided data set of input/output data. The architecture selection algorithm is completed with a structure identification procedure, used to obtain the optimal input space partitioning of the different sub-grids of the model. The complete algorithm is used to obtain the MGFS models for the CATS series prediction problem, initially proposed as a competition in the IJCNN'2004 conference (International Joint Conference on Neural Networks), solved using a direct prediction-based approach. This chapter includes the following published works:

  - [Herrera et al., 2004c]: Preliminary work on multigrid systems for basic function approximation problems.
  - [Herrera et al., 2004a]: Work on the application of multigrid systems to the problem of the prediction of the CATS time series in the conference IJCNN'2004.
  - [Herrera et al., 2007a]: Complete work of a learning methodology for multigrid fuzzy systems, and the application of this methodology to the complete CATS time series prediction problem.

- **Chapter 7**: The work presented in this paper presents the utility of two different methodologies, the TaSe Fuzzy TSK model and the Least-Squares SVMs, to solve the problem of long term time series prediction using recursive prediction. This work also introduces some techniques that upgrades the performance of those advanced one-step-ahead models (and in general of any one-step-ahead model), where they are used recursively for long term time series prediction. This chapter includes the following published work:

  - [Herrera et al., 2007b]: Complete work on long term recursive time series prediction using one-step-ahead direct models, using the TaSe and LS-SVM methodologies, including an improving technique for one-step-ahead models when they are used recursively for long term time series prediction.

- **Chapter 8**: This chapter proposes a backward variable selection methodology for function approximation problems that is based on the mutual information measure. Among the advantages that the proposed methodology presents, we stand out the possibility of avoiding of the curse of dimensionality problem that the mutual information estimators can present, making the method feasible for large problems without losing its robustness. The estimation of the mutual information and the selection of

the mutual information estimator parameter, are furthermore improved by using resampling methods as proposed in previous works. The performance of the proposed methodology is shown through significant examples of large function approximation problems, such as spectrometric data and time series prediction problems. This chapter includes the following published work:

- [Herrera et al., 2006]: Preliminary work on a backward variable selection methodology for function approximation problems, that is based in the mutual information measure of the information theory.

- **Chapter 9**: This chapter extracts the conclusions that are drawn from the different works developed in this thesis.

- **Appendix A**: This appendix presents the membership function configuration that are used along the chapters 4, 5 and 6.

- **Appendix B**: This appendix presents the development of the partial derivatives of the quadratic error function, with respect to each of the coefficients of the consequents in the rules of a TSK fuzzy system.

In order to make easier the reading of this thesis, the following conventions were taken:

- Each chapter is divided into sections, each one of which can include one of more subsections. When a reference is done to any part of the text, this reference will include the number of the chapter, followed by the number of the section. For example: section 4.3, subsection 4.3.3.

- Figures and tables are numbered by chapter. Example: figure 6.1, table 6.1.

- Mathematical expressions also follow the same scheme by chapters, and are expressed by their numbers. Example: equation 7.1.

- The bibliographic references are indicated by using the surname of the authors followed by the year of publication if the work has at most 3 authors, and by the surname of the main author followed by the abbreviation *et al.* if the work has more authors. Example: [Koller and Sahami, 1996], [Rojas et al., 2002].

Capítulo 2

Conocimientos Previos: Herramientas para Aproximación Funcional.

En este capítulo se realiza una revisión sobre los conceptos más importantes, necesarios para una compresión adecuada del trabajo expuesto en los siguientes capítulos. Primeramente se presentan tres metodologías muy conocidas y ampliamente utilizadas para resolver problemas de aproximación funcional a partir de datos de E/S. En primera sección se introducen los sistemas difusos y el sistema de inferencia difusa, centrándonos sobre todo en los sistemas tipo Takagi-Sugeno-Kang (TSK), más adaptados a la problemática que tratamos. Posteriormente se revisan las redes neuronales de Funciones de Base Radial (redes RBF), y la equivalencia que existe entre éstas y los sistemas difusos tipo TSK cuando se cumplen ciertas condiciones. Finalmente se introducen las Máquinas de Vectores Soporte de mínimos cuadrados (Least Squares SVM), como herramienta con resultados óptimos para problemas de aproximación funcional, y que evitan ciertos problemas concretos que presentan las Máquinas de Vectores Soporte tradicionales para este tipo de problemas. Finalmente se plantea el problema de la selección de variables en un problema de modelado, así como los tipos de algoritmos que pueden encontrarse, y se diferencia entre selección variables y extracción de características. Se introduce el concepto de información mutua, y se revisa un estimador de la información mutua basado en los k-vecinos más cercanos. Basándonos en el criterio de la información mutua se revisan también las posibles estrategias de selección de variables que se pueden llevar a cabo en un algoritmo de selección de variables cualquiera.

2.1. Conjuntos Difusos, Reglas Difusas, Sistemas Difusos

En la mayoría de los procesos complejos (naturales, físicos,...) es prácticamente imposible obtener una descripción precisa de la dinámica de los mismos, y además, puede ocurrir que ésta sea innecesaria para su control efectivo. Para poder superar la necesidad de una representación precisa del conocimiento, Zadeh introdujo los conjuntos difusos [Zadeh, 1965] que permitían tratar con formas aproximadas de razonamiento en las cuales el concepto de verdadero o falso es una cuestión de grado. En las últimas décadas, la lógica difusa ha sido un área de trabajo muy fructífera para el desarrollo de diferentes disciplinas (como pueden ser los problemas de interpolación, modelado e identificación de sistemas, reconocimiento y clasificación de patrones, etc.). En concreto para el área del modelado continuo/aproximación funcional, son muchos los trabajos que han demostrado su eficacia y conveniencia en la aplicación a dichos problemas. Los modelos difusos presentan la ven-

taja de que son comprensibles por el diseñador, evitando los problemas de caja-negra que presentan otros modelos; son modelos sencillos pero pueden modelar funciones complejas usando un conjunto de reglas sencillas que pueden expresarse en términos de variables lingüísticas (difusos); no necesitan un modelo matemático preciso y presentan una alta capacidad de generalización.

Esta sección presenta las bases sobre las que se asientan los sistemas difusos para problemas de modelado y sirve de punto de partida a los siguientes capítulos de esta tesis. Primeramente se establecen los conceptos básicos de conjuntos difusos. Posteriormente se pasa a introducir los sistemas de reglas difusas, su estructura y su función de inferencia. En su mayoría, los contenidos de la sección se han extraído de la tesis doctoral de Héctor Pomares [Pomares, 2000].

2.1.1. Conjuntos difusos

Un conjunto clásico, como se denominan en este contexto para diferenciarlos de los conjuntos difusos, tiene una barrera o contorno claramente definida. Un elemento pertenece o no a dicho conjunto, sin existir grados intermedios de pertenencia. Por ejemplo se puede definir un conjunto $A = \{x/x < 4\}$, con $x \in \mathbb{Z}$. La función de pertenencia (FP) al conjunto A, denotada como $\mu_A(x)$, de un elemento cualquiera x puede ser expresada como

$$\mu_A(x) = \begin{cases} 1 \text{ si } x \in A \\ 0 \text{ si } x \notin A \end{cases} \tag{2.1}$$

El conjunto A es matemáticamente equivalente a su función de pertenencia, en el sentido en el que conocer $\mu_A(x)$ es lo mismo que conocer el conjunto A. Sin embargo, el conocimiento humano sobre la pertenencia o no de un determinado elemento a un conjunto, no es siempre tan claro y fácil de catalogar como en la teoría clásica. Por ejemplo, si Z representa el conjunto de coches que poseen una velocidad máxima alta, el contorno que define dicho conjunto no está claramente determinado. Una de las aportaciones principales de Zadeh fue la introducción de grados de pertenencia en el intervalo real para conjuntos difusos. Esto admite la posibilidad de asignar diversos grados de certeza sobre la pertenencia o no de un elemento a un conjunto.

Expresado en notación matemática, un conjunto difuso Z está definido por pares de elementos en la forma $(x, \mu_z(x))$, donde x es el elemento propiamente dicho y $\mu_z(x)$ es el grado de pertenencia del elemento al conjunto Z. Ahora, los valores que puede tomar la función $\mu_z(x)$ no están restringidos a $\{0,1\}$, sino que generalmente están normalizados en el intervalo [0,1]. Para un conjunto discreto, la definición de los elementos de dicho conjunto es

$$Z = \sum_{i=1}^{n} x^i/\mu_Z(x^i) = x^1/\mu_z(x^1) + \cdots + x^n/\mu_z(x^n), \tag{2.2}$$

donde se ha utilizado el símbolo de suma '+' para definir la unión de cada uno de los elementos del conjunto. Los elementos se han definido como $x/\mu_z(x)$, donde el símbolo '/' significa que forma un par ordenado. Cuando se trata de un conjunto continuo, la definición sería:

$$Z = \int_x x/\mu_Z(x). \tag{2.3}$$

Definición 1: Conjunto difuso y Función de pertenencia. Si X es una colección de objetos denotados genéricamente por x, un conjunto difuso A en el universo de trabajo

X se define por un conjunto de pares ordenados en la forma: $A = \{(x, \mu_A(x))/x \in X\}$, donde $\mu_A(x)$ es la función de pertenencia de un elemento x al conjunto A. La característica más importante de esta función es que puede tomar cualquier valor dentro del intervalo normalizado [0,1]. Como se puede comprobar, la definición de un conjunto difuso no es más que la extensión de la definición clásica de conjunto, permitiendo que la función de pertenencia tome valores diferentes de 0 y 1. Si la función de pertenencia de un conjunto difuso A está restringida a tomar valores en $\{0, 1\}$, entonces A se reduce a un conjunto clásico.

Definiciones y Conceptos

En la exposición del conocimiento humano es muy frecuente encontrar términos lingüísticos para expresar el estado de una variable. Por ejemplo, si la variable es la temperatura, son habituales expresiones como *temperatura alta*, o *temperatura muy baja*, etc. Una variable lingüística es aquella cuyos valores no son numéricos, como las variables comunes, sino que son palabras o sentencias. La motivación para el uso de palabras o sentencias en lugar de utilizar números es que una caracterización lingüística en general contiene una mayor cantidad de información y es más comprensible para una persona aun siendo menos específica que una numérica.

Una variable lingüística se caracteriza por una quíntupla $(x, T(x), X, G, M)$ en la cual x es el nombre de la variable (refiriéndonos al ejemplo anterior, x sería la temperatura), $T(x)$ son los valores lingüísticos que dicha variable puede tomar ($Baja$, $Media$, $Alta$), cada uno de los cuales viene caracterizado por una función de pertenencia definida en el universo de trabajo X, mientras que G es la regla sintáctica para generar los nombres de los valores de x, y M es la regla para asociar a cada uno de los nombres su significado.

A continuación se van introducir las definiciones más usuales sobre conjuntos difusos, algunas de las cuales serán utilizadas a lo largo de los siguientes capítulos de esta memoria.

Definición 2: Sean A y B dos conjuntos difusos definidos en el mismo universo X. El conjunto A está contenido en el conjunto B, o equivalentemente, el conjunto A es un subconjunto de B, si y sólo si para todo elemento x de A, $\mu_A(x) \leq \mu_B(x)$. En forma matemática:

$$A \subseteq B \iff \mu_A(x) \leq \mu_B(x), x \in X \tag{2.4}$$

Dos conjuntos difusos A y B son iguales si y solo si $\mu_A(x) = \mu_B(x)$, con $x \in X$.

Definición 3: Un conjunto difuso A es convexo si y sólo si satisface la siguiente propiedad:

$$x, y \in X, \lambda \in [0, 1] : \mu_A(\lambda x + (1 - \lambda)y) \geq Min(\mu_A(x), \mu_A(y)) \tag{2.5}$$

Definición 4: El soporte de un conjunto difuso A se define como:

$$S(A) = \{x \in X/\mu_A(x) > 0\} \tag{2.6}$$

Definición 5: La amplitud de un conjunto difuso A convexo, con soporte $S(A)$ se define como:

$$Amplitud(A) = Sup(S(A)) - Inf(S(A)) \tag{2.7}$$

Cuando se trabajan con conjuntos difusos convexos, como es el caso común de las funciones de pertenencia definidas para las variables de un sistema de control, las defini-

ciones de Soporte y Amplitud se utilizan de forma indistinta. El Soporte es un intervalo, y si es cerrado, las operaciones supremo e ínfimo se pueden sustituir por máximo y mínimo.

Definición 6: El núcleo de un conjunto difuso A se obtiene como:

$$Nucleo(A) = \{x \in X / A(x) = 1\} \tag{2.8}$$

Si tan sólo hay un punto donde se verifica dicha condición (como sucede con funciones de pertenencia triangulares o gaussianas), el valor x_p donde $\mu_A(x_p) = 1$ se denomina $pico$ del conjunto A.

Definición 7: La altura de un conjunto difuso A se define como el valor de pertenencia máximo de los elementos del soporte de dicho conjunto o equivalentemente:

$$Altura(A) = Sup\{\mu_A(x) / x \in X\} \tag{2.9}$$

Definición 8: Un conjunto difuso está normalizado, si su altura es 1, o equivalentemente, si su núcleo no es el conjunto vacío.

Definición 9: El concepto de α-nivel de un conjunto difuso A, representa el subconjunto de A en el cual los elementos $x \in X$, tienen un valor de pertenencia $\mu_A(x) \geq \alpha$.

$$\alpha - nivel(A) = \{x \in S / \mu_A(x) \geq \alpha\} \tag{2.10}$$

Si en lugar de utilizar el símbolo $\geq$ se utiliza $>$, se denomina α-nivel fuerte.

2.1.2. Operaciones entre conjuntos difusos

Al igual que se definen operaciones entre conjuntos clásicos, las mismas operaciones se pueden definir entre conjuntos difusos, donde ahora estas operaciones se realizan a través de las funciones de pertenencia. Las operaciones fundamentales entre conjuntos clásicos son la intersección, la unión y el complemento. La extensión de dichas operaciones no se define de forma unívoca en el contexto de la lógica difusa, a diferencia de lo que sucede en lógica clásica. Zadeh [Zadeh, 1973] propuso las siguientes definiciones:

$$\begin{aligned} &\forall x \in X : \mu_{A \wedge B}(x) = Min(\mu_A(x), \mu_B(x)) \\ &\forall x \in X : \mu_{A \vee B}(x) = Max(\mu_A(x), \mu_B(x)) \\ &\forall x \in X : \mu_{A'}(x) = 1 - \mu_A(x) \end{aligned} \tag{2.11}$$

Si se restringen los valores de $\mu_A(x)$ y $\mu_B(x)$ al conjunto $\{0, 1\}$, estas operaciones son las mismas que las operaciones Y, O, y Complemento de los conjuntos clásicos. Sin embargo, existe una gran cantidad de posibilidades para la implementación de la intersección y unión de conjuntos difusos como veremos a continuación.

Operadores de intersección: T-normas

Para la realización de los primeros sistemas difusos inspirados en el diseño propuesto por Mamdani, los operadores que se utilizaban para la intersección y unión eran los operadores Mínimo y Máximo. Sin embargo, ha existido un gran esfuerzo investigador en el estudio de las propiedades de dichos operadores sobre la inferencia difusa [Alsina et al., 1983], [Mizumoto, 1989], [Kovalerchuk and Taliansky, 1992], [Cárdenas et al., 1993] demostrándose que, para ciertas aplicaciones, unos operadores trabajan mejor que otros (ver referencias [Kruse et al., 1994], [Gupta and Qi, 1991]).

Definición 10: Una función continua $t : [0, 1]x[0, 1] \rightarrow [0, 1]$ es una T-norma si satisface los siguientes criterios:

$$\begin{array}{l} t(a,b) = t(b,a) \text{ (propiedad conmutativa)} \\ t(t(a,b),c) = t(a,t(b,c)) \text{ (propiedad asociativa)} \\ \text{si } a \leq c \text{ y } b \leq d \Rightarrow t(a,b) \leq t(c,d) \text{ (función creciente)} \\ t(a,1) = a \text{ (elemento neutro)} \end{array} \tag{2.12}$$

Siendo a, b, c, d números reales en el intervalo [0,1]. De las propiedades elemento neutro y función creciente se deduce que $a \in [0,1] : t(a,0) = 0$. Se puede demostrar que el operador mínimo es la T-norma mayor, es decir, para cualquier conjunto de entradas, cualquier otra T-norma produce un valor menor o igual al mínimo de las entradas. Algunas de las T-normas más utilizadas en la bibliografía son:

$$\begin{array}{cc} t(a,b) = \text{mín}(a,b) & t(a,b) = a \cdot b \\ t(a,b) = \text{máx}(0, a+b-1) & t(a,b) = \begin{cases} a \text{ si } b = 1 \\ b \text{ si } a = 1 \\ 0 \text{ en otro caso} \end{cases} \end{array} \tag{2.13}$$

Operadores de unión: T-conormas

Otra función de gran importancia en la composición de conjuntos difusos, es la operación de unión. Aun existiendo artículos destinados al estudio de sus propiedades matemáticas e influencia empírica en el proceso de inferencia difusa, la investigación sobre este tipo de operador fue menos intensa que en el caso de las T-normas.

Definición 11: Una función continua $t* : [0,1]x[0,1] \rightarrow [0,1]$ es una T-conorma si satisface los siguientes criterios:

$$\begin{array}{l} t*(a,b) = t*(b,a) \text{ (propiedad conmutativa)} \\ t*(t*(a,b),c) = t*(a,t*(b,c)) \text{ (propiedad asociativa)} \\ \text{si } a \leq c \text{ y } b \leq d \Rightarrow t*(a,b) \leq t*(c,d) \text{ (función creciente)} \\ t*(a,0) = a \text{ (elemento neutro)} \end{array} \tag{2.14}$$

Se puede comprobar que las características de elemento neutro y función creciente conllevan a que $a \in [0,1] : t*(a,1) = 1$. Alsina [Alsina et al., 1983] define a una t-conorma como la función S de dos variables que hace que la función T definida como: $T(a,b) = 1 - S(1-a, 1-b)$ sea una t-norma. Cuando la t-norma y t-conorma satisfacen la anterior relación, se dicen que son parejas de operadores conjugados. Ejemplos de ellos pueden ser el mínimo y máximo, el producto y el operador Goguen de unión $(a + b - a\Delta b)$, etc. Se demuestra que el operador máximo es la T-conorma menor. Algunas de las T-conormas más utilizadas en la bibliografía son:

$$\begin{array}{cc} t*(a,b) = Max(a,b) & t*(a,b) = a + b - a \cdot b \\ t*(a,b) = Min(1, a+b) & t*(a,b) = \begin{cases} a \text{ si } b = 0 \\ b \text{ si } a = 0 \\ 1 \text{ en otro caso} \end{cases} \end{array} \tag{2.15}$$

Operadores de negación

El complemento de un conjunto difuso es una operación similar a la realizada en teoría clásica, aunque hay que hacer notar que la operación se realiza sobre los valores de las funciones de pertenencia.

Definición 12: Una función continua $c : [0,1] \rightarrow [0,1]$ es una función complemento difuso, si satisface las siguientes propiedades:

$$\begin{array}{l} c(0) = 1 \\ \text{si } a \leq b \Rightarrow c(a) \geq c(b) \text{ (función decreciente)} \\ c(c(a)) = a \text{ (propiedad de involución)} \end{array} \tag{2.16}$$

Ejemplos de estos tipos de funciones son el complemento ordinario: $c(a) = 1 - a$, el de Yager: $c_w(a) = (1-a^w)^{1/w}$ con $w \in (0, \infty)$, o el de Sugeno: $c_\lambda(a) = (1-a)/(1+\lambda a)$ con $\lambda \in (-1, \infty)$.

Una propiedad importante de estas funciones es que se puede demostrar que toda operación complemento al menos tiene un punto de equilibrio. Se define un punto de equilibrio como un valor 'a' tal que $c(a) = a$, lo que equivale a afirmar que el punto de equilibrio de un conjunto difuso A, es un valor tal que el grado de pertenencia al conjunto o a su complementario es el mismo.

2.1.3. Razonamiento aproximado

Los elementos primitivos de representación del conocimiento humano, en el razonamiento aproximado, lo constituyen las proposiciones difusas. Una proposición difusa es una expresión del tipo: "la temperatura es elevada", donde temperatura es una variable lingüística y elevada es uno de los atributos pertenecientes al conjunto T(temperatura). Estas proposiciones primitivas pueden combinarse mediante conectivas lingüísticas de muy diversas formas, todas ellas derivadas de las cuatro siguientes operaciones fundamentales [Lee, 1990]:

- Conjunción (denotado por $p \wedge q$): donde se forma una nueva proposición basada en la veracidad de ambas proposiciones.

- Disyunción (denotado por $p \vee q$): donde se forma una nueva proposición basada en la veracidad de una de las dos proposiciones.

- Implicación (denotado por $p \rightarrow q$): que usualmente toma la forma de reglas difusas de tipo SI-ENTONCES.

- Negación (denotado por $\neg p$): junto con las proposiciones generadas utilizando conjunción, disyunción o implicación, una nueva proposición puede ser obtenida mediante el uso del prefijo "Es falso que ...", a una proposición existente.

Una regla difusa es, por tanto, una sentencia condicional expresada simbólicamente como

$$\text{SI } proposicion_A \text{ ENTONCES } proposicion_B \tag{2.17}$$

donde tanto $proposicion_A$ como $proposicion_B$ pueden ser proposiciones difusas simples o compuestas. La parte SI se denomina antecedente de la regla mientras que la parte ENTONCES es el consecuente. Por ejemplo:

$$\text{SI } \textit{la temperatura} \text{ es } \textit{alta} \text{ ENTONCES } \textit{la presión} \text{ es } \textit{muy grande} \tag{2.18}$$

donde *alta* y *muy grande* son etiquetas lingüísticas correspondientes a las variables lingüísticas *temperatura* y *presión* respectivamente. Una regla difusa relaciona el conjunto difuso

asociado a la proposición antecedente con el conjunto difuso asociado a la proposición consecuente. Debido a ello, se observa una gran semejanza en la utilización de reglas difusas y el acceso a celdas de memoria asociativas, por lo que a veces a un conjunto de reglas difusas se le denomina "memoria difusa asociativa" FAM (Fuzzy Associative Memory, [Kosko, 1992a], [Sudkamp and Hammell, 1994]).

Una regla difusa del tipo "SI X es P ENTONCES Y es B" se suele abreviar en la forma $P \rightarrow B$. En esencia, esta expresión describe la relación entre dos variables X e Y lo que sugiere que una regla SI-ENTONCES puede ser definida como una relación binaria difusa R en el producto cartesiano XxY. Hay que hacer notar que el producto cartesiano difuso XxY es una extensión del producto cartesiano clásico, puesto que ahora cada elemento $(x, y) \in XxY$ tiene asociado un grado de pertenencia $\mu_R(x, y)$ siendo:

$$R = P \rightarrow B = \int_{XxY} \mu_P(x) \rightarrow \mu_P(y)/(x, y) \tag{2.19}$$

y $\rightarrow$ la función de implicación escogida, que puede ser la de Kleene-Dienes, Lukasiewicz, Zadeh, Gödel, Mamdani, etc. Por tanto, una relación difusa representa el grado de presencia o ausencia de asociación, interacción o interconexión entre elementos de dos o más conjuntos difusos [Mendel, 1995].

Para la obtención de una consecuencia difusa, a partir de unos antecedentes dados, se utiliza lo que se denomina proceso de inferencia difuso [Cao and Kandel, 1992], [Piskunov, 1992], [Reddy and Babu, 1992]. Este proceso consiste fundamentalmente en realizar cálculos entre los conjuntos difusos asociados a las proposiciones que componen las reglas para obtener una conclusión que puede ser difusa o numérica dependiendo de la aplicación (para problemas de aproximación funcional, la conclusión debe ser por lo general numérica). Existen dos mecanismos fundamentales de inferencia: el Modus Ponens Generalizado (GPM) y el Modus Tollens Generalizado (GTM). El esquema básico de ambos sistemas de inferencia es:

premisa 1: X es P'
premisa 2: Si X es P ENTONCES Y es B
———————————————— (GPM)
conclusión: Y es B'

premisa 1: Y es B'
premisa 2: Si X es P ENTONCES Y es B
———————————————— (GTM)
conclusión: X es P'

Donde P, P', B y B' son etiquetas lingüísticas. Se puede observar que en lógica clásica una regla es disparada si y solo si la premisa es exactamente igual al antecedente de la regla. En lógica difusa, por el contrario, una regla es activada siempre y cuando exista una similitud entre la premisa y el antecedente de la regla, permitiendo por tanto distintos grados de activación de la regla. En general, el método de inferencia utilizado es el GPM. Para la obtención del conjunto difuso B' hay que realizar la operación $B' = P'\beta R$, donde el operador β se refiere a la composición de relaciones difusas, y R es la relación difusa generada por la regla que forma la premisa 2. Para poder definir la composición entre P' y R debemos antes introducir las dos operaciones fundamentales entre relaciones difusas: proyección y extensión cilíndrica.

Definición 13: Sea R una relación difusa sobre $U = \prod_{i=1}^{n} U_i = U_1 x U_2 x \dots x U_n$ y sea $(i_1, \dots, i_k)$ un subconjunto de índices de $(1, \dots, n)$ con $k \leq n$, y $(i_1, \dots, il)$ su secuencia complementaria. Sea $V = \prod_{m=1}^{k} U_{i_m}$. La proyección de R sobre V se define como:

$$proy(R)\ sobre\ V = \int_V \sup_{x_{j_1} \dots x_{j_l}} \mu_R(x_1, \dots, x_n)/(x_{i_1}, \dots, x_{i_k}) \tag{2.20}$$

En el caso binario, si definimos R sobre XxY tendríamos:

$$proy(R)\ sobre\ Y = \int_Y \sup_x \mu_R(x, y)/y \tag{2.21}$$

Definición 14: Sea S una relación difusa sobre $V = \prod_{m=1}^{k} U_{i_m}$ (ver definición anterior). La extensión cilíndrica de S en U se define como:

$$ce(S) = \int_U \mu_S(x_{i_1}, \dots, x_{i_k})/(x_1, \dots, x_n) \tag{2.22}$$

En el caso binario, si definimos F como un conjunto difuso sobre Y tendríamos:

$$ce(F) = \int_{XxY} \mu_F(y)/(x, y) \tag{2.23}$$

Ahora ya estamos en condiciones de definir el operador 'β':

Definición 15: Sea A un conjunto difuso definido sobre X y R una relación difusa definida sobre XxY. Se define la composición de A y R ($A\ \beta\ R$) como un conjunto difuso B definido sobre Y, dado por la expresión:

$$B = A \circ R = proy(ce(A) \cap R)\ sobre\ Y \tag{2.24}$$

Si el operador de intersección se realiza con el Min y la proyección con el Max lo que tendremos es la regla de inferencia inicialmente propuesta por Zadeh:

$$\mu_b(y) = \max_x min(\mu_A(x), \mu_R(x, y)) \tag{2.25}$$

Igualmente, si la intersección se realiza con el producto tendríamos:

$$\mu_b(y) = \max_x \mu_A(x) \dot{\mu}_R(x, y) \tag{2.26}$$

En el caso de utilizar reglas con premisas en el antecedente formadas por la conjunción de dos conjuntos difusos el esquema de razonamiento aproximado utilizado (para el GPM) es:

premisa 1: X es P' Y Y es Q'
premisa 2: Si X es P Y Y es Q ENTONCES Z es B
---
conclusión: Z es B'

La premisa 2 es una regla difusa de estructura $PxQ \rightarrow B$. Esta regla puede ser transformada en una relación ternaria difusa R, en la cual los elementos pertenecientes al conjunto Cartesiano $XxYxZ$ tienen un grado de pertenencia definido por $\mu_R(x, y, z) = \mu_{(PxQ) \rightarrow B}(x, y, z)$. El conjunto difuso B' resultante de la inferencia puede ser calculado

utilizando los operadores producto $\cdot$ (para la función de implicación, para la intersección en la composición y para la conjunción de las premisas) y máximo $\vee$ (para la proyección) en la forma:

$$\mu_{B'}(z) = \underbrace{\overbrace{\bigvee_{x,y}}^{\text{proyección}} \left\{ \left[\mu_{P'}(x) \overbrace{\bullet}^{\text{conjunción}} \mu_{Q'}(y) \right] \overbrace{\bullet}^{\text{intersección}} \left[\underbrace{\left(\mu_P(x) \overbrace{\bullet}^{\text{conjunción}} \mu_Q(y) \right)}_{\mu_R(x,y,z)} \overbrace{\bullet}^{\text{implicación}} \mu_B(z) \right] \right\}}_{\text{composición}} \tag{2.27}$$

Desarrollando la expresión anterior tenemos:

$$\mu_{B'}(z) = \bigvee_{x,y} \{ [\mu_{P'}(x) \bullet \mu_{Q'}(y)] \bullet [(\mu_P(x) \bullet \mu_Q(y)) \bullet \mu_B(z)] \} \tag{2.28}$$

$$= \bigvee_{x,y} \{\mu_{P'}(x) \bullet \mu_{Q'}(y) \bullet \mu_P(x) \bullet \mu_Q(y)\} \bullet \mu_B(z) \tag{2.29}$$

$$= \left(\bigvee_x \{\mu_{P'}(x) \bullet \mu_P(x)\} \right) \bullet \left(\bigvee_y \{\mu_{Q'}(y) \bullet \mu_Q(y)\} \right) \bullet \mu_B(z) \tag{2.30}$$

$$= (\alpha_1 \bullet \alpha_2) \bullet \mu_B(z) = \overset{Regla}{\alpha} \bullet \mu_B(z) \tag{2.31}$$

donde se ha simbolizado con α_1 el grado de semejanza entre el valor P del antecedente de la regla y el valor P' medido, α_2 el grado de semejanza entre el valor Q del antecedente de la regla y el valor Q' medido y α^{Regla} la actividad total de la regla. Finalmente, si en lugar de tener una regla tenemos muchas, debemos recurrir a un último operador, llamado operador de agregación (generalmente el Max o la Suma de conjuntos difusos para sistemas Mamdani, y la suma ponderada y la media ponderada para los sistemas Takagi-Sugeno-Kang, ver sección siguiente) de tal forma que podamos, a partir de una entrada difusa dada, obtener la salida inferido a través de un conjunto de reglas difusas.

Existen otras muchas formas de realizar este proceso mediante la utilización de las funciones de implicación difusas. Un análisis más detallado de todo el proceso se puede encontrar en [Driankov et al., 1993], [Jager, 1995].

2.1.4. Estructura de un sistema difuso

Los componentes principales de un sistema difuso se ilustran en la figura 2.1, y vienen dados por:

- Un bloque de conversión de entradas reales en un conjunto difuso, para su posterior utilización utilizando la teoría de conjuntos difusos. Es lo que se denomina proceso de "fuzzificación".

- Una base de conocimiento, consistente en la definición de los valores lingüísticos de cada una de las variables de entrada/salida del sistema y las reglas que lo conforman.

- Un motor de inferencia que actúa según las reglas y funciones de pertenencia descritas por el diseñador del sistema difuso. Es realmente el núcleo del sistema difuso, y

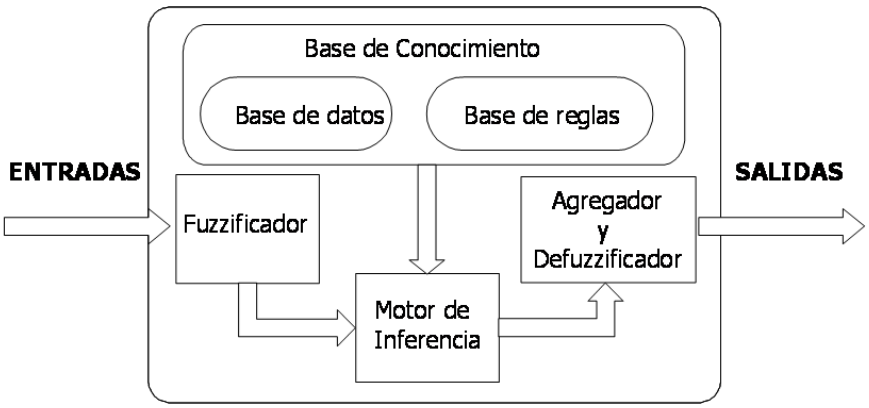


Figura 2.1. *Estructura de un sistema Difuso*

tiene la capacidad de simular el proceso de toma de decisiones de un operador humano.

- Una fase de condensación de las salidas de las reglas en una sola, y conversión del conjunto difuso obtenido como conclusión en un valor numérico. Es lo que se denomina proceso de "agregación" y "defuzzificación".

A continuación se describen con cierto detalle cada una de las funciones realizadas por los bloques más importantes de un sistema difuso.

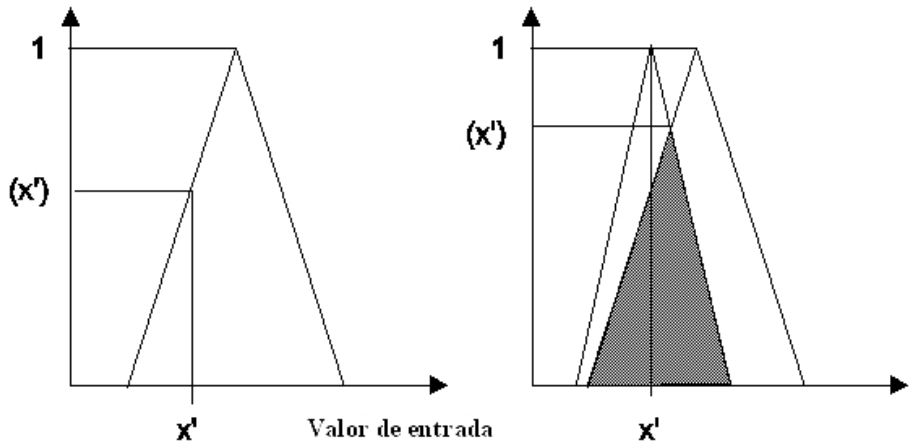
Codificación de las entradas: "fuzzificación"

El primer paso para operar con un sistema difuso es transformar la información de entrada de los datos medidos en términos de valores léxicos usados en las precondiciones de las reglas difusas. La fase de "fuzzificación" requiere dos tareas fundamentales:

- Medición de las variables de entrada al sistema.

- Realización de un ajuste de escala, de forma que se transforme el rango de entradas en valores apropiados para la definición de los dominios de las variables lingüísticas.

Si el valor de entrada es un valor numérico, el proceso de "fuzzificación" requiere comparar el valor numérico con el valor de las diferentes funciones de pertenencia de la variable correspondiente (caso (a) en la figura 2.2). Si el valor de entrada contiene ruido o está deteriorada, una primera aproximación es modelar esta incertidumbre mediante una función triangular centrada en la lectura obtenida, y la base del triángulo proporcional a la desviación estándar de la lectura. En este caso el objetivo del proceso de "fuzzificación" es encontrar la intersección (usando el Min como T-norma) entre los diferentes valores lingüísticos de la variable y la función triangular formada por una determinada medida (caso (b) en la figura 2.2). El método de "fuzzificación" más utilizado es el primero de los mencionados donde, aunque la medida del sensor pueda tener un cierto error, dicha lectura es tomada directamente para determinar la salida del sistema. La razón principal para no implementar el segundo método de "fuzzificación" es la complejidad computacional que requiere.



a) b)

Figura 2.2. *(a) Obtención del valor de pertenencia con la medida directa del sensor (entrada numérica); (b) considerando la entrada del sistema como un conjunto difuso (entrada difusa).*

Base de conocimiento de un sistema difuso

La base de conocimiento de un sistema difuso comprende la información necesaria para definir unívocamente las reglas y funciones de pertenencia de las variables del sistema, de forma que se logren los objetivos de modelado deseados. Se compone de:

La base de datos: encargada de la selección y descripción de los valores lingüísticos que definen las funciones de pertenencia de cada una de las variables de entrada y salida. La selección de la granularidad o densidad de las funciones de pertenencia es muy importante para el proceso de modelado.

La base de reglas: que representa subdivisión del espacio del problema de modelado por medio de reglas difusas. Una vez establecidas las particiones de cada una de las variables de entrada, se tiene que relacionar las diferentes zonas en que queda dividido el subespacio de entrada de un problema con las salidas más adecuadas a cada una de esas areas. Se trata por tanto de determinar el valor de los consecuentes de las reglas que forman el conjunto de reglas del sistema.

Motor de inferencia

Es realmente el núcleo de operación de un sistema difuso. Su misión es calcular la salida difusa a partir de las entradas del sistema y conforme a las reglas y funciones de pertenencia definidas. Dependiendo del diseño de los consecuentes de las reglas difusas, principalmente se pueden distinguir en la bibliografía dos modos fundamentales de realizar dicho proceso:

- Modelo de Mamdani.

- Modelo de Takagi-Sugeno-Kang.

Modelo de Mamdani

En los sistemas difusos tipo Mamdani [Mamdani, 1974] los consecuentes de las reglas se expresan mediante términos lingüísticos o términos numéricos fijos. Este tipo de modelo es el más ampliamente utilizado en la literatura, puesto que se considera más interpretable que el de tipo TSK. Es el sistema difuso por antonomasia, puesto que incluye todas las fases del proceso de inferencia difusa, y utiliza normalmente conjuntos difusos a la salida. Sin embargo puede presentar inconvenientes desde el punto de vista del rendimiento respecto del modelo TSK. Las reglas utilizadas son del tipo anteriormente presentadas

$$\text{Si } X \text{ es } P \text{ Y } X' \text{ es } Q \text{ ENTONCES } Z \text{ es } B. \tag{2.32}$$

Modelo de Takagi-Sugeno-Kang (TSK)

En 1985, Sugeno [Sugeno, 1985a], [Sugeno, 1985b], [Takagi and Sugeno, 1985] realizó un estudio sobre las posibilidades de incluir en el consecuente nuevas formas de actuación. El nuevo modelo de reglas que propuso es de la forma:

$$\text{Si } X \text{ es } P \text{ Y } Y \text{ es } Q \text{ ENTONCES } Z \text{ es } F(x,y) \tag{2.33}$$

donde P y Q continúan siendo conjuntos difusos, mientras que la función $F(x,y)$ es una función numérica de las entradas del sistema. Usualmente, $F(x,y)$ es una función lineal de las entradas $F(x,y) = p_0 + p_1x + p_2y$, pero en general puede ser cualquier tipo de función polinómica. Cuando el grado del polinomio es uno, el sistema se denomina "Sistema TSK orden 1 o de primer orden", el cual fue originalmente propuesto en [Takagi and Sugeno, 1985], [Sugeno and Kang, 1988]. Igualmente pordrían definirse sistemas de orden 2 o segundo orden, etc. Cuando $F(x,y)$ es una constante, el sistema obtenido es equivalente al modelo de Mamdani utilizando consecuentes no difusos. En la mayoría de los casos, el número de reglas necesarias para la definición de una función difusa es inferior utilizando el modelo de Sugeno que usando el modelo de Mamdani. Es esa mayor capacidad aproximativa la que hace que se usen normalmente los sistemas TSK para problemas de aproximación funcional. Sin embargo, la interpretación de dichas reglas no resulta sencilla debido a que la conclusión es una función de las entradas y no una constante o un valor lingüístico como en el caso del sistema de Mamdani. Además, el conocimiento de un operador humano es difícil de plasmar en términos de reglas con este tipo de estructura. Los sistemas TSK presentan la peculiaridad de que no necesitan una fase de defuzzificación, sino tan solo constan de una fase final de agregación que agrega las salidas de las reglas activadas.

Combinación de las salidas y decodificación: agregación y "defuzzificación"

La fase de agregación combina los consecuentes de cara regla del sistema difuso, para obtener una salida en el caso de los sistemas TSK, o bien como preparación para la fase de "defuzzificación" en el caso de sistemas Mamdani. La fase de "defuzzificación" es necesaria para convertir la salida difusa de un sistema difuso Mamdani en una representación numérica. En general la combinación y decodificación son pasos necesarios pues deseamos obtener un valor concreto de salida a la entrada proporcionada. En sistemas de control por ejemplo, la salida de un controlador debe ser de forma numérica debido a que los actuadores mecánicos, eléctricos, etc. sólo aceptan señales de esta naturaleza. Hay una amplia literatura sobre la investigación de las propiedades de estas operaciones [Zhao and Govind, 1991], [Filev and Yager, 1993], [Hellendoorn and Thomas, 1993], [Mabuchi, 1993].

Los métodos de agregación más utilizados para sistemas de tipo TSK son la media ponderada o "weighted average" y la suma ponderada o "weighted sum". El método más usado es la media ponderada, que como se comentará en sucesivos temas, proporciona un mejor rendimiento, mientras que la suma ponderada se suele asociar a las redes RBF que se introducen más adelante en este mismo tema.

Para sistemas tipo Mamdani, los métodos de agregación más utilizados son el máximo y la suma de conjuntos difusos. Además, para sistemas Mamdani, existen diversas estrategias de "defuzzificación". La estrategia principal de la fase de "defuzzificación" es obtener el elemento numérico que mejor represente el conjunto difuso de salida. Desgraciadamente, no existe hasta la fecha una forma sistemática para la elección del método de "defuzzificación" más adecuado para cada aplicación. En la bibliografía, los métodos mayoritariamente utilizados son el Centro de Área (CDA), el Centro de Sumas (CDS), el Centro de Picos (CDP), el Centro del Área Mayor (CDAM), el Primer Máximo (PM), la Media de Máximos (MDM) y el Método de calidad. Se puede encontrar una descripción más en profundidad de estos métodos y una comparativa en [Pomares, 2000].

2.1.5. La lógica difusa como aproximador universal

Una de las preguntas que más puede interesar en la concepción de un sistema difuso es la precisión con la que una función arbitraria puede ser aproximada mediante un conjunto de reglas y funciones de pertenencia. En el campo de las redes neuronales artificiales, se ha demostrado que un perceptrón de tres niveles puede aproximar tan bien como se requiera cualquier tipo de función real continua en un dominio compacto, con un número adecuado de neuronas en sus niveles [Hornick et al., 1989]. El mismo resultado ha sido probado para un controlador difuso utilizando el diseño utilizado por Mamdani, pero con consecuentes numéricos en lugar de con variables difusas [Wang and Mendel, 1992a]. Utilizando el teorema de Stone-Weierstrass, [Buckley, 1992] demostró que un conjunto de controladores difusos pueden aproximar cualquier tipo de función real continua en un dominio compacto, donde los elementos básicos del controlador son valores lingüísticos representados mediante gaussianas, se utiliza el operador producto para la inferencia e implicación, y como defuzzificador el centro de picos. Un conjunto o tipo de sistemas difusos 'Ψ' es universal si y sólo si dado un proceso cualquiera P, existe un sistema $\Phi \in \Psi$ de forma que la diferencia entre la salida del sistema difuso y la salida del proceso es menor que una constante ϵ, tan pequeña como se requiera.

Kosko [Kosko, 1992a], [Kosko, 1992b], [Kosko, 1994] realizó un análisis muy similar, aunque empleando el concepto de regiones difusas (fuzzy patches). Buckley, en el trabajo [Buckley, 1993] demostró que utilizando el modelo propuesto por Sugeno, también podía construir sistemas difusos con capacidad de aproximar cualquier tipo de función continua. Las primitivas son:

- Los consecuentes de las reglas son funciones polinómicas.

- El proceso de "defuzzificación" es $Z* = 3\lambda_i$, donde λ_i es el grado de semejanza entre las entradas reales del sistema y el antecedente de la regla R_i, en lugar de utilizar la normalización $\lambda_i^* = \lambda_i / 3\lambda_i$.

No obstante, como queda reflejado por J.L.Castro en los trabajos [Castro, 1995], [Castro and Delgado, 1996], existen diversos sistemas difusos que no utilizan las primitivas ni la estructura de la que parten los anteriores resultados y son aproximadores universales. La principal razón es que en muchos diseños se requieren otros tipos de funciones

de pertenencia como, por ejemplo, triangulares o trapezoidales. Con respecto al modelo de inferencia empleado, existe una gran diversidad de operadores de implicación al igual que metodologías de "defuzzificación". En [Castro, 1995] se demuestra que los sistemas difusos formados por:

- funciones de pertenencia triangulares, trapezoidales, gaussianas,
- funciones de conjunción utilizando cualquier tipo de operador T-norma,
- funciones de implicación que solo deben satisfacer unas propiedades elementales.
- funciones de "defuzzificación" usuales en la bibliografía (Media de Máximos, Centro de Área, Primer Máximo, etc.) que sólo deben satisfacer que el punto de "defuzzificación" pertenezca al soporte,

son todos ellos aproximadores universales. También la fusión de los paradigmas neuronales y lógica difusa proporciona una herramienta eficaz para aproximar cualquier tipo de función [Buckley and Hayashi, 1994]. Pese a estas demostraciones, no existe aún una metodología definida para la construcción de un sistema difuso (elección del número de funciones de pertenencia para cada una de las reglas, forma y localización de cada uno de los valores lingüísticos, número de reglas y significado de cada una de ellas, operadores de fuzzificación, implicación, defuzzificador, T-normas, T-conormas, etc.) que nos asegure que el grado de aproximación quede siempre por debajo de una determinada cota dada.

Estructura típica de un sistema difuso tipo TSK

Funciones de pertenencia

En la literatura, las funciones de pertenencia más comúnmente utilizadas son las funciones triangulares, y las funciones gaussianas o pseudo-gaussianas. En el apéndice A se presentan en detalle las configuraciones de funciones de pertenencia empleadas.

Particionamiento del espacio de entrada

En secciones anteriores se presentaron los dos modelos de reglas que más se utilizan en las aplicaciones de loss sistemas difusos. Este trabajo se centra en los modelos tipo TSK, que utilizan reglas de la forma

$$\text{Si } x_1 \text{ es } \mu_1^k(x_1) \text{ Y } \ldots \text{ Y } x_n \text{ es } \mu_n^k \text{ ENTONCES } Y \text{ es } Y_k(\vec{x}) \tag{2.34}$$

siendo n el número de variables de entrada, y $Y_k(\vec{x})$ un polinomio de orden v de las variables de entrada $\vec{x} = \{x_1, \ldots, x_n\}$. Para orden $v = 0$ tenemos un sistema TSK de orden 0, esto es con consecuentes constantes, que también se puede considerar de tipo Mamdani. Para orden $v = 1$ tenemos un sistema TSK de orden 1 o lineal, etc. Sin embargo para órdenes $v = 1$ o superior, se suele considerar que la interpretación de las reglas se complica, debido a que los consecuentes son funciones de las entradas, y no una constante o un valor lingüístico como en los sistemas de tipo Mamdani.

Expresión funcional del modelo difuso

En cuanto al uso de los distintos operadores que intervienen en el proceso de inferencia, se utilizará el operador producto '$\cdot$ ' para la función de implicación, para la intersección en la composición y para la conjunción de las premisas y el operador máximo '$\vee$ ' para la

proyección. En definitiva, se usarán los mismos operadores con los que se obtuvo la expresión 2.28. De esta forma, utilizando la media ponderada como operador de agregación tendremos que el conjunto difuso inferido en la salida será de la forma:

$$\tilde{F}(\vec{x}) = \frac{\underbrace{\sum_{k=1}^{nreglas}}_{\textit{agregación}} \alpha_k \underbrace{\bullet}_{\textit{implicación}} Y_k(\vec{x})}{\underbrace{\sum_{k=1}^{nreglas}}_{\textit{agregación}} \alpha_k} \tag{2.35}$$

siendo:

- α_k: grado con el que el vector de entrada activa la regla k
- $Y_k(\vec{x})$: valor devuelto por el consecuente de la regla j en el punto $\vec{x}$.

Teniendo en cuenta que las entradas al sistema son numéricas, debemos proporcionar el valor de α_j en función de los conjuntos difusos definidos en las premisas de las reglas teniendo en cuenta que las entradas son valores numéricos exactos (no difusos). A partir de 2.28:

$$\alpha_k \equiv \alpha_k(\vec{x}') = \underbrace{\prod_{i=1}^{N}}_{\textit{conjunción}} \alpha_{i,k}(x_i') \tag{2.36}$$

siendo $\alpha_{i,k}$ el grado de semejanza entre el conjunto difuso X_i^k que aparece en la premisa de la regla k y el conjunto difuso en la entrada de la variable k (que en este caso es el valor numérico x_i'), y $\vec{x}'$ el vector de entrada no difuso. De nuevo, recurriendo a la ecuación 2.28:

$$\alpha_{i,k}(x_i') = \underbrace{\bigvee_{x_i}}_{\textit{proyección}} \left\{ \mu_{X_i'}(x_i) \overbrace{\bullet}^{\textit{intersección}} \mu_{X_i^k}(x_i) \right\} \tag{2.37}$$

y, teniendo en cuenta que:

$$\mu_{X_i'}(x_i) = \begin{cases} 1 & x_i = x_i' \\ 0 & \textit{otro caso} \end{cases} \tag{2.38}$$

tenemos finalmente que:

$$\alpha_{i,k}(x_i') = \mu_{X_i^k}(x_i') \tag{2.39}$$

En resumen, a partir de 2.35, 2.36 y 2.39, la salida numérica de nuestro sistema para un vector de entrada no difuso genérico vendrá dada por:

$$\tilde{F}(\vec{x}) = \frac{\sum_{k=1}^{nreglas} Y_k(\vec{x}) \cdot \prod_{i=1}^{N} \mu_{X_i^k}(x_i)}{\sum_{k=1}^{nreglas} \prod_{i=1}^{N} \mu_{X_i^k}(x_i)} \tag{2.40}$$

Hay autores que denominan a esta forma de inferencia "razonamiento difuso simplificado" mientras que otros lo denominan "media ponderada".

Estructura neuro-difusa del sistema TSK

Una posible implementación de este tipo de sistemas es la que se presenta en la figura 2.3. La capa 1 es la encargada de recoger las entradas del sistema. La capa 2 calcula el grado de pertenencia de cada componente de la entrada a cada una de las funciones de pertenencia definidas para dicha componente. Cada neurona de la capa 3 (habrá tantas como reglas) simplemente realiza el producto de sus entradas de modo que su salida representará el grado de activación de la regla que tiene asignada. Finalmente, la neurona de salida calcula tanto el numerador como el denominador de 2.40 para así obtener el valor de salida del sistema.

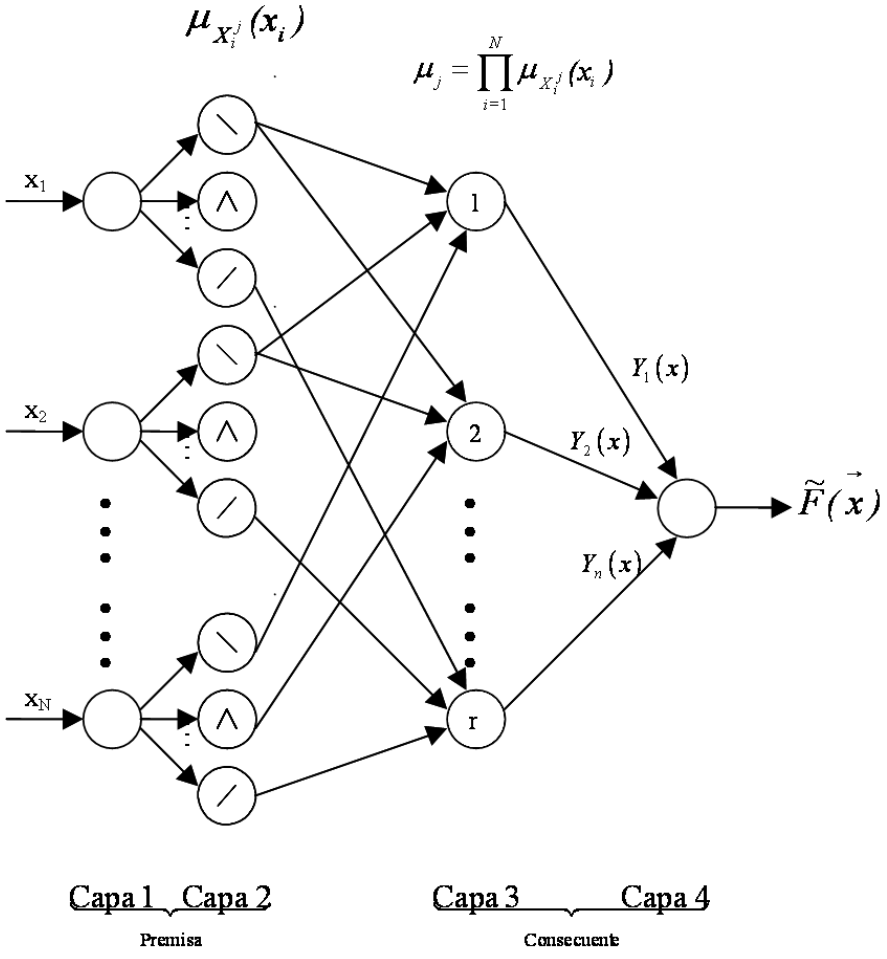


Figura 2.3. *Estructura de un sistema Difuso*

Como es fácil de apreciar a simple vista, la arquitectura de la figura 2.3 presenta gran analogía con una red neuronal por lo que a esta forma de implementación se le suele denominar red neuro-difusa. La ventaja principal de utilizar este tipo de diseño es que el proceso

de inferencia se puede realizar en paralelo, reduciendo considerablemente el tiempo necesario para realizar dicha inferencia. Existen otras maneras de realizar la implementación hardware de los sistemas que se tratarán aquí, pero ésta es una cuestión que queda fuera del ámbito de esta tesis.

2.2. Redes de Función de Base Radial

Las redes de Función de Base Radial (o redes RBF) son un tipo de red neuronal, que tiene su origen, al igual que los otros tipos de redes neuronales, en ciertos aspectos del funcionamiento estructural del cerebro. Son bien conocidas las estructuras cerebrales formadas por unidades receptivas con ajuste local y solapamiento, que se encuentran en el cortex cerebral, el cortex visual y otros. Así, Moody y Darken en [Moody and Darken, 1989], [Moody and Darken, 1988] propusieron una estructurar de red que empleaba unidades receptivas locales para realizar mapeos funcionales. Este modelo, junto con otros similares propuestos en otros trabajos [Powell, 1987], [Broomhead and Lowe, 1988] en el área de la interpolación y la teoría de la aproximación funcional, se denominan colectivamente aproximadores (redes) de función de base radial.

Las redes RBF constan de tres capas: una capa interna de conexión de la red al entorno (a los valores de entrada, por esto a veces se considera que solo poseen dos capas reales); una segunda capa o capa oculta cuyas neuronas llevan asociadas un centro y una varianza de dimensiones iguales a los datos de entrada de la red, y que aplican alguna función de base radial obteniendo una salida para cada dado de entrada; y finalmente una tercera capa o capa de salida que obtiene una salida (o varias salidas) que es función lineal de las salidas obtenidas por las neuronas en la capa oculta a partir de los datos de entrada, según los pesos de la red w_k(ver figura 2.4).

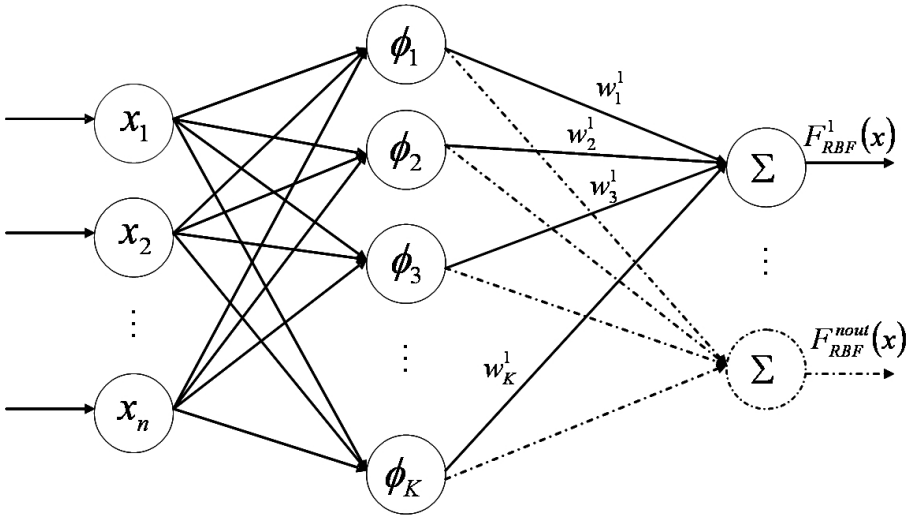


Figura 2.4. *Estructura de una red neuronal de funciones de base radial.*

Así, en general la capa de salida funciona como una unidad de suma, de forma que la salida de la red RBF se expresa como

$$F_{RBF}(\vec{x}) = \sum_{k=1}^{K} w^k \phi^k(\vec{x}, c^k, \sigma^k) \tag{2.41}$$

Las funciones de base radial $\phi^k(\vec{x}, c^k, \sigma^k)$ son funciones no lineales

$$\phi^k(\vec{x}, c^k, \sigma^k) = \phi^k(\frac{\|\vec{x} - c^k\|}{\sigma^k}) \tag{2.42}$$

donde la función ϕ es normalmente una gaussiana,

$$\phi(r) = exp(\frac{r^2}{2}) \tag{2.43}$$

aunque puede utilizarse cualquier función de base radial (multi-cuadráticas, cúbicas, logística...), y donde $\|\cdot\|$ es la norma euclídea, que se define como

$$\|\vec{x} - c^k\| = \sqrt{\sum_{i=1}^{n}(x_i - c_i^k)^2} \tag{2.44}$$

donde r es el radio calculado según las ecuaciones 2.42 y 2.44. Así la función de salida de una neurona oculta de la red puede expresarse como

$$\phi^k(\vec{x}, c^k, \sigma^k) = \prod_{i=1}^{n} exp(-\frac{(c_i^k - x_i)^2}{{\sigma^k}^2}) \tag{2.45}$$

Y la función de salida global del sistema como

$$F_{RBF}(\vec{x}) = \sum_{k=1}^{K} w^k \prod_{i=1}^{n} exp(-\frac{(c_i^k - x_i)^2}{{\sigma^k}^2}) \tag{2.46}$$

2.2.1. Alternativas en el diseño de redes de función de base radial

Utilización de media ponderada en la salida

Una alternativa en la expresión lineal de la salida de la red para la capa de salida es calcular la media ponderada de las funciones de base radial existentes en la red. En este tipo de redes por tanto la salida queda normalizada por el total de las activaciones de la capa oculta. La expresión de la salida para la red $F_{RBF}(\vec{x})$ queda expresada de la siguiente forma

$$F_{RBF}(\vec{x}) = \frac{\sum_{k=1}^{K} w^k \phi^k(\vec{x}, c^k, \sigma^k)}{\sum_{k=1}^{K} \phi^k(\vec{x}, c^k, \sigma^k)} \tag{2.47}$$

Redes de función de base radial pseudo-gaussianas

En la ecuación 2.45, las anchuras de la función de base radial ${\sigma^k}^2$ pueden ser distintas para cada dimensión del espacio de entrada ${\sigma_i^k}^2$, o igual para todas las dimensiones

(simplemente ${\sigma^k}^2$ para la RBF k). En el primer caso nos encontramos con las llamadas redes de función de base radial pseudo-gaussianas. En este caso, la flexibilidad del modelo aumenta al contar con un número mayor de parámetros para la aproximación. Sin embargo, por otro lado, la complejidad de entrenamiento también aumenta considerablemente al incrementarse como se ha dicho el número de parámetros a optimizar.

Redes de función de base radial con pesos de regresión

En las funciones de base radial, los parámetros que conectan las activaciones de las neuronas ocultas con la neurona de salida w^k, normalmente son parámetros sencillos (constantes). Sin embargo existe la posibilidad de generalizar la expresión de dichos parámetros, aumentando por tanto su capacidad expresiva, utilizando lo que se llaman pesos de regresión ("regression weights"). Así, los pesos de la red RBF w^k, pasan a ser función lineal de las variables de entrada $w^k(\vec{x})$, mediante la adición de conexiones laterales entre las neuronas de funciones de base radial

$$w^k(\vec{x}) = \sum_{i=1}^{n} b_i^k x_i + b_0^k \tag{2.48}$$

Utilizando esta formulación, aumenta el número de parámetros de la red RBF con lo que el proceso de optimización de la red (así como el proceso de evaluación) aumentan su complejidad. Sin embargo, este tipo de parámetros tienen una optimización relativamente simple, pues se pueden obtener mediante aprendizaje supervisado utilizando optimización lineal, como veremos en la sección de optimización de redes RBF. Así, la capacidad aproximadora de unidad de función de base radial aumenta, y el número de neuronas necesarias para realizar la aproximación funcional se puede ver reducido. La formulación lineal expuesta en la ecuación 2.48, puede ser ampliada para considerar una expresión polinómica de orden mayor. La figura 2.5 muestra la estructura de una red de funciones de base radial con coeficientes de regresión. Las neuronas añadidas mediante conexiones laterales $r^1, r^2, \ldots, r^K$ en forma de sub-red de regresión, realizan el cómputo de los pesos $w^1(\vec{x}), w^2(\vec{x}), \ldots, w^K(\vec{x})$. La capa de salida en la práctica queda dividida en dos subcapas, una para la aplicación de los pesos a las salidas de las neuronas en la capa oculta y otra que aplica la función lineal sumatoria de salida.

2.2.2. Equivalencia funcional entre redes RBF y sistemas difusos TSK

La ecuaciones 2.45 presentan una formulación similar a las de los sistemas difusos tipo TSK según la ecuación 2.40, independientemente del tipo de consecuente utilizado en el sistema TSK (ver ecuación 2.34) o el peso de la red RBF según la ecuación 2.48. Así mismo en el caso de usar suma ponderada en el sistema TSK, la formulación sería similar a la de la red RBF según la ecuación 2.47. Ambas funciones de salida pueden ser por tanto idénticas, supuesto que se escogen adecuadamente el tipo de función de pertenencia, las funciones de base radial y ciertos operadores de ambos sistemas.

Mientras que una red RBF consta de funciones de base radial, los sistemas difusos están compuestos de un conjunto de funciones de pertenencia para realizar la inferencia difusa. Ambos sistemas por tanto constan de un conjunto de funciones radiadas que producen una respuesta ponderada según las activaciones de las distintas unidades receptivas. Así, aunque ambos sistemas tienen un origen distinto, ambos tienen un funcionamiento con raíces similares.

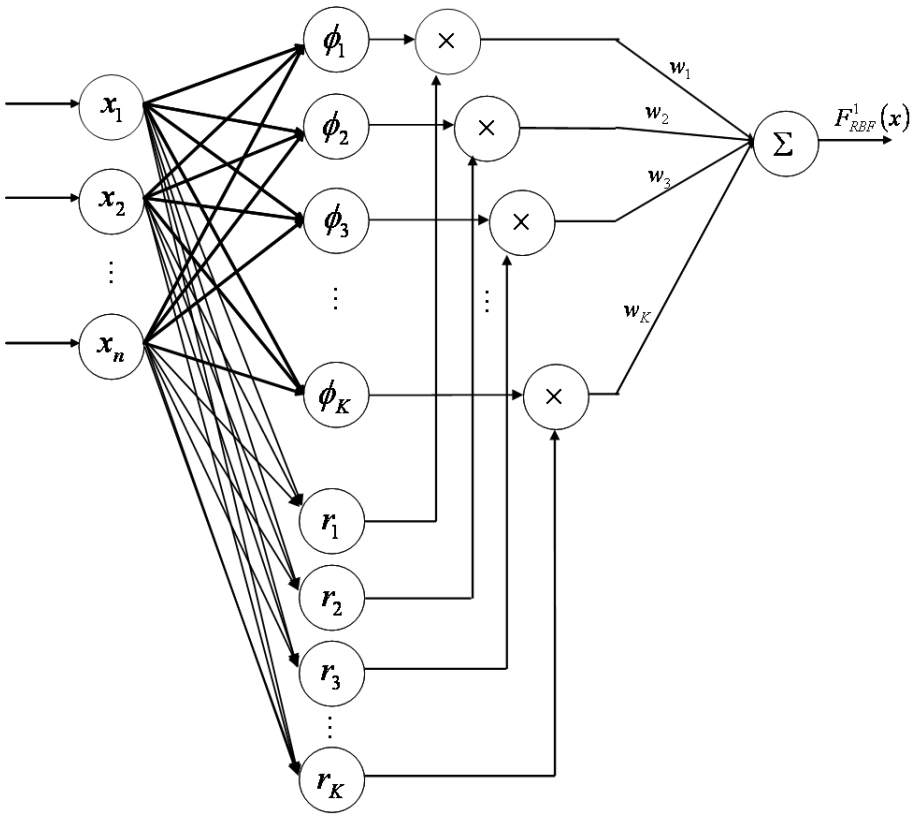


Figura 2.5. *Estructura de una red neuronal de funciones de base radial con pesos de regresión*

Las condiciones bajo las que una red de funciones de base radial y un sistema difuso tipo TSK son funcionalmente equivalentes se resumen en

- La red RBF y el sistema difuso considerados deben tener el mismo método de agregación para obtener la salida, bien sea la media ponderada o la suma ponderada.

- El número de neuronas ocultas en la red RBF es igual al número de reglas del sistema difuso.

- Cada función de base radial de la red RBF es igual a la composición multidimensional de las distintas funciones de pertenencia que presenta la correspondiente regla en su antecedente. Esto implica primeramente que la función de pertenencia utilizada en el sistema difuso es de tipo Gaussiano. Así, haciendo coincidir las componentes centro y varianza de cada dimensión de la función de base radial, con el centro y varianza de cada función de pertenencia de la regla difusa (distinguiendo el caso particular de tener una sola varianza para todas las dimensiones tanto en la red RBF como en el sistema difuso). En cualquier caso, debe ser el producto la T-norma utilizada en el sistema difuso.

- Finalmente la función de salida de ambos sistemas debe ser idéntica, esto es , esto es, los consecuentes de las reglas difusas, y los pesos de las neuronas ocultas deben tener los mismos términos constantes (en el caso de sistemas difusos de orden cero y redes RBF tradicionales), lineales (en el caso de sistemas difusos de orden uno y redes RBF con pesos de regresión lineales) etc.

Así ambos paradigmas quedan unidos bajo esta equivalencia funcional. Ambos son denominados comúnmente sistemas neuro-difusos, y son ampliamente utilizados en la literatura de metodologías de aproximación funcional.

2.3. Máquinas de Vectores Soporte de Mínimos Cuadrados

En esta sección se exponen las bases de las Máquinas de Vectores Soporte de Mínimos Cuadrados. Primeramente se presentan las Máquinas de Vectores Soporte para clasificación, origen del paradigma más en auge en los últimos años en el área del aprendizaje automático. Posteriormente presentamos la extensión de las SVMs para regresión, que utilizan la función de pérdida de Vapnik. Por último se resumen las SVMs de Mínimos Cuadrados para aproximación funcional, que son una modificación de la formulación básica de las SVMs, caracterizadas por un sistema de ecuaciones lineales, y que evitan algunos problemas que presentan las SVMs tradicionales para regresión.

2.3.1. Máquinas de vectores soporte para clasificación

Las Máquinas de Vectores Soporte (SVMs) son un conjunto de técnicas de aprendizaje supervisado basadas en kernels (funciones núcleo), usadas tanto para problemas de clasificación como de regresión. Fueron introducidas por primera vez en el trabajo [et al., 1992] y están directamente relacionadas con la Teoría Estadística del Aprendizaje de Vapnik [Vapnik, 1999]. Las SVMs adquirieron popularidad en el área del aprendizaje automático tras su éxito en un problema de reconocimiento de dígitos escritos, y han recibido una enorme atención en dicha área, aún en aumento, desde entonces.

Dado un conjunto de puntos de patrones de entrada/salida de dos clases distintas, $\left\{(x^1, y^1), ..., (x^m, y^m)\right\} \subset X \times \{\pm 1\}$, un clasificador SVM busca el hiperplano separador óptimo $\langle w, x\rangle + b = 0$ que maximiza el margen de separación entre cualquier punto de entrenamiento de cualquiera de las clases y el hiperplano separador. En el caso hipotético en el que ambos conjuntos de patrones son separables en el espacio de entrada, la superficie separadora puede encontrarse resolviendo el siguiente problema de optimización con restricciones

$$\begin{aligned} &\min_{w,\, b} J_P(w) = \frac{1}{2} w^T w \\ &\text{tal que } y^k[w^T x^k + b] \geq 1, \;\; k = 1, \ldots, m. \end{aligned} \tag{2.49}$$

Ahora bien, en el caso de patrones no separables, hay que permitir errores en la clasificación. Para ello se modifica la formulación, introduciendo las llamadas variables flojas o "slack" ς^k

$$\begin{aligned} \min_{w,\ b,\varsigma^k} J_P(w) &= \frac{1}{2}\|w\|^2 + C\sum_{k=1}^{m}\varsigma^k \\ \text{tal que } y^k(\langle w, x\rangle + b) &\geq 1 - \varsigma^k \text{ with } \varsigma^k \geq 0\ \forall k = 1, ..., m. \end{aligned} \tag{2.50}$$

donde la constante $C > 0$ representa el intercambio equilibrado entre la maximización del margen, y la minimización del error de entrenamiento. De estas ecuaciones, se puede formular una Lagrangiana, cuya solución lleva al siguiente problema de optimización de Programación Cuadrática (PC)

$$\begin{aligned} \max_{\alpha} J_D(\alpha) &= -\frac{1}{2}\sum_{k,l=1}^{m} y^k y^l {x^k}^T x^l \alpha^k \alpha^l + \sum_{k=1}^{m}\alpha^k \\ \text{tal que } &\sum_{k=1}^{m}\alpha^k y^k = 0 \\ &0 \leq \alpha^k \leq C,\ \ k = 1, \ldots, m. \end{aligned} \tag{2.51}$$

Hasta ahora, se ha considerado la búsqueda de un hiperplano separador lineal con margen máximo. Sin embargo, la potencia de las Máquinas de Vectores Soporte es la posibilidad, como métodos de kernel que son, de realizar un mapeo hacia un espacio de dimensionalidad mayor (que llamaremos espacio de características), mediante un mapeo específico desde el espacio de entrada conocido (que denominaremos espacio de entrada), donde se realizará la búsqueda del hiperplano separador con margen mayor. Así cada punto x^k se lleva al espacio de características mediante un mapeo $\phi(x^k)$, con el objetivo de encontrar una superficie lineal de separación en dicho espacio. Así, en la definición del problema de optimización de la ecuación anterior 2.51, el producto escalar ${x^k}^T x^l$ se sustituiría por el producto escalar en el nuevo espacio multidimensional $\phi(x^k)^T\phi(x^l)$.

Sin embargo, el mapeo explícito hacia el espacio de características no es necesario, gracias al "truco del kernel" o "kernel trick". Un kernel $k(\cdot,\cdot)$ es una función que toma dos puntos en el espacio de entrada, y directamente calcula el producto escalar en el espacio de características ϕ

$$k(x^k, x^l) = \phi(x^k)^T\phi(x^l) \tag{2.52}$$

El kernel, siendo un producto escalar en un espacio de características, puede verse como una medida de similaridad entre puntos en el espacio de entrada. Algunas funciones de kernel típicas son

$$\begin{aligned} &k(x, x^k) = {x^k}^T x \quad \text{(SVM lineal)} \\ &k(x, x^k) = (\tau + {x^k}^T x)^d \quad \text{(SVM polinomial de grado } d) \\ &k(x, x^k) = exp(-\|x - x^k\|_2^2/\sigma^2) \quad \text{(kernel RBF)} \end{aligned} \tag{2.53}$$

Así, en la formulación dada en la ecuación 2.51, el producto escalar ${x^k}^T x^l$, se substituye por na función de kernel $k(\cdot,\cdot)$

$$\begin{aligned} \max_{\alpha} J_D(\alpha) &= -\frac{1}{2}\sum_{k,l=1}^{N} y^k y^l k(x^k, x^l)\alpha^k\alpha^l + \sum_{k=1}^{m}\alpha^k \\ \text{tal que } &\sum_{k=1}^{m}\alpha^k y^k = 0 \\ &0 \leq \alpha^k \leq C,\ \ k = 1, \ldots m. \end{aligned} \tag{2.54}$$

Una vez la solución al problema anterior es encontrada, la función de decisión toma la forma

$$f(x) = sgn(\sum_{i=1}^{m} y^i \alpha^i k(x, x^i) + b) \; . \tag{2.55}$$

donde el parámetro b se calcula usando las condiciones de Karush-Kuhn-Tucker (condiciones KKT) $\alpha^i \left[y^i (k(x, x^i) + b) - 1 \right] = 0 \; \forall i = 1, ..., m$

Las Máquinas de Vectores Soporte (SVM) son el estado del arte en clasificación, e incluyen una extensión para problemas de regresión como se verá en la siguiente subsección. El nombre de Máquinas de Vectores Soporte viene del hecho de que tras el entrenamiento de la SVM, solo parte de los vectores de entrada x^k son conservados en el modelo final como "vectores soporte" o vectores frontera, que son aquellos que explícitamente se utilizan en el cálculo del hiperplano separador de ambas clases. El entrenamiento de una SVM se realiza resolviendo un problema de optimización PG. Este es el talón de aquiles de este tipo de técnicas, pues la complejidad computacional del entrenamiento aumenta muy considerablemente al aumentar el número de puntos de entrada. Así para problemas con un número mayor de 1000 puntos de entrada, el tiempo que requiere una optimización de los hiperparámetros de una SVM puede llegar a ser prohibitivo bajo un soporte computacional convencional. Pese a esto, estas técnicas se han aplicado con gran éxito en una gran variedad de problemas en la literatura del área del aprendizaje automático en la última década y media.

2.3.2. Máquinas de vectores soporte para aproximación funcional

Dado un conjunto de entrenamiento, $\left\{ (x^1, y^1), ..., (x^m, y^m) \right\} \subset X \times Y$, un estimador lineal SVM busca el hiperplano que mejor aproxima $\left\langle w, x^k \right\rangle + b \cong y^k$, utilizando como función de pérdida la llamada función ϵ-insensible de Vapnik

$$|y - f(x)|_\epsilon = \begin{cases} 0, & \text{si } |y - f(x)| \leq \epsilon \\ |y - f(x)| - \epsilon, & \text{en otro caso} \end{cases} \tag{2.56}$$

La figura 2.6 muestra la forma de la función de pérdida de Vapnik. La estimación de la función lineal se realiza formulando el siguiente problema con incógnitas w, b

$$\begin{aligned} \min_{w \in X, \; b \in} \; & J_P(w) = \frac{1}{2} w^T w \\ \text{tal que} \; & y^k - w^T x^k - b \leq \epsilon, \;\; k = 1, \ldots, m \\ & w^T x^k + b - y^k \leq \epsilon, \;\; k = 1, \ldots, m \end{aligned} \tag{2.57}$$

El valor de ϵ indica la precisión que se requiere para la aproximación. Esta formulación anterior requiere que todos los puntos entren dentro de un ϵ-tubo multidimensional. Sin embargo, un valor de ϵ pequeño hará que ciertos puntos caigan fuera de dicho tubo. Para controlar los errores que se puedan cometer, se introducen variables flojas o "slack" ς^k, ς^{k^*}

$$\begin{aligned} \min_{w, \; b, \varsigma^k} \; & J_P(w) = \frac{1}{2} \|w\|^2 + C \sum_{k=1}^{m} (\varsigma^k + \varsigma^{k^*} \\ & y^k - w^T x^k - b \leq \epsilon + \varsigma^k, \;\; k = 1, \ldots, m \\ \text{tal que} \; & w^T x^k + b - y^k \leq \epsilon + \varsigma^{k^*}, \;\; k = 1, \ldots, m \\ & \varsigma^k, \varsigma^{k^*} \geq 0, k = 1, \ldots, m \end{aligned} \tag{2.58}$$

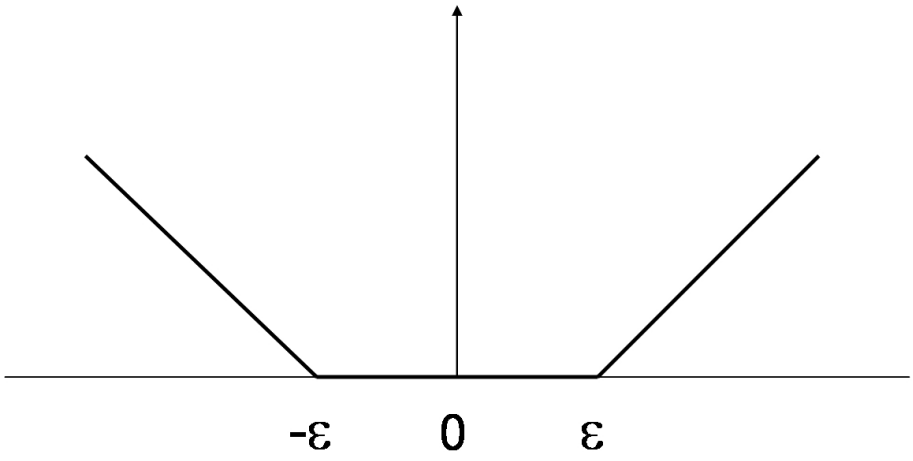


Figura 2.6. *Función de pérdida ϵ-insensible de Vapnik.*

donde la constante $C > 0$ indica la cantidad de desviaciones permitidas fuera del la precisión deseada ϵ que se tolera. De estas ecuaciones, se puede formular una Lagrangiana, cuya solución lleva al siguiente problema de optimización de Programación Cuadrática (PC)

$$\begin{aligned} \max_{\alpha,\alpha^*} J_D(\alpha,\alpha^*) = \quad & -\tfrac{1}{2}\sum_{k,l=1}^{m}(\alpha^k-\alpha^{k^*})(\alpha^l-\alpha^{l^*}){x^k}^T x^l \\ & -\epsilon\sum_{k=1}^{m}(\alpha^k-\alpha^{k^*})+\sum_{k,l=1}^{m} y^k(\alpha^k-\alpha^{k^*}) \\ \text{tal que} \quad & \sum_{k=1}^{m}(\alpha^k-\alpha^{k^*})=0 \\ & \alpha^k,\alpha^{k^*}\in[0,c]. \end{aligned} \tag{2.59}$$

Donde la solución al problema de estimación funcional lineal $f(x) = w^T x + b$ se traduce en el espacio dual, habiendo obtenido los α, α^{k^*} óptimos resolviendo el problema PC anterior, con $w = \sum_{k=1}^{m}(\alpha^k - \alpha^{k^*})x^k$, en

$$f(x) = \sum_{k=1}^{m}(\alpha-\alpha^{k^*}){x^k}^T x + b \tag{2.60}$$

Hasta ahora, de nuevo se ha considerado la búsqueda de un hiperplano lineal que ajuste los puntos dentro de un ϵ tubo n-dimensional. Ahora bien, se puede trasladar este problema lineal, a uno de solución no lineal mediante el "truco del kernel", que nos permite buscar una superficie lineal que ajuste los datos en el espacio de características, de mayor dimensionalidad. Dicha búsqueda en el espacio de características se realiza mediante un mapeo $\phi(x^k)$, implícito gracias a la utilización de una función de kernel $k(x^k, x^l) = \phi(x^k)^T\phi(x^l)$. Así, en la formulación dada en la ecuación 2.59, el producto escalar ${x^k}^T x^l$, se substituye por una función de kernel $k(\cdot,\cdot)$

$$\begin{aligned}\max_{\alpha,\alpha^*} J_D(\alpha,\alpha^*) = \quad & -\tfrac{1}{2}\sum_{k,l=1}^{m}(\alpha^k-\alpha^{k^*})(\alpha^l-\alpha^{l^*})k(x^k,x^l)- \\ & \epsilon\sum_{k=1}^{m}(\alpha^k+\alpha^{k^*})+\sum_{k,l=1}^{m}y^k(\alpha^k-\alpha^{k^*}) \\ \text{tal que} \quad & \sum_{k=1}^{m}(\alpha^k-\alpha^{k^*})=0 \\ & \alpha^k,\alpha^{k^*}\in[0,c].\end{aligned} \tag{2.61}$$

Una vez la solución al problema anterior es encontrada, la función de aproximación toma la forma

$$f(x)=\sum_{k=1}^{m}(\alpha-\alpha^{k^*})k(x^k,x)+b \tag{2.62}$$

donde α, α^{k^*} son la solución al problema de PC anterior, y el parámetro b se calcula usando las condiciones de Karush-Kuhn-Tucker (condiciones KKT).

La solución al problema de Programación Cuadrática es global y único, suponiendo que la función de kernel cumple una serie de requisitos (es positiva definida). De nuevo, la solución obtenida es no dispersa "sparse" en el sentido de que el modelo final conserva solo parte de los puntos de entrada, que son los necesarios para definir las fronteras dentro de la cual se encuentra la estimación funcional obtenida.

Una formulación alternativa que permite controlar el número de vectores soporte obtenidos finalmente es la regresión mediante vectores soporte de tubo-ν o "ν-tube", en el que el objetivo primario de la función de optimización se traduce en

$$\frac{1}{2}w^Tw+c\left(\nu\epsilon+\frac{1}{m}\sum_{k=1}^{m}(\varsigma^k\varsigma^{k^*})\right) \tag{2.63}$$

donde ν controla el número de vectores soporte que se permite que caigan fuera del tubo, que es directamente proporcional al número de vectores soporte en un sentido asintótico.

2.3.3. Máquinas de vectores soporte de mínimos cuadrados para regresión (LS-SVM)

Las máquinas de vectores soporte resuelve un problema de programación cuadrática, y obtienen una solución global dispersa tras dicha optimización. Sin embargo como veremos, es posible simplificar algunos aspectos de la formulación de las SVMs, sin perder sus ventajas.

Las Máquinas de Vectores Soporte de Mínimos Cuadrados (LS-SVM) son una reformulación de las SVMs de Vapnik, en las que la optimización lleva a resolver un sistema de ecuaciones lineales, más sencillo de utilizar que las soluciones a la programación cuadrática. Aquí se verá solo el caso de LS-SVM para problemas de aproximación funcional. La formulación de Vapnik se modifica en dos puntos. En primer lugar, en vez de inecuaciones, la formulación de los LS-SVM utilizan ecuaciones de igualdad, donde el valor a la derecha de la ecuación se considera como un valor objetivo, más que un valor umbral (ver ecuaciones 2.57 y 2.58). Sobre este valor objetivo, se permite un error de estimación variable ς^k. Esta variable de error juega un papel similar a las variables flojas en los SVMs. En segundo lugar, la función de pérdida de Vapnik, se sustituye por una función

de pérdida cuadrática. Estas dos modificaciones simplifican enormemente la formulación del problema como se verá a continuación.

Considerando un modelo en el espacio primario de la siguiente forma

$$y(x) = w^T \phi(x) + b \tag{2.64}$$

donde $x \in \mathbb{R}^n$, $y \in \mathbb{R}$ y $\phi(\cdot) : \mathbb{R}^n \to \mathbb{R}^{n_h}$ es un mapeo hacia un espacio de características multidimensional mayor y posiblemente de infinitas dimensiones. Dado un conjunto de entrenamiento $\{x^k, y^k\}_{k=1}^m$, podemos formular el siguiente problema de optimización en el espacio primario

$$\begin{array}{c} \min\limits_{w,b,\varsigma} J_P(w,\varsigma) = \dfrac{1}{2} w^T w + \gamma \dfrac{1}{2} \displaystyle\sum_{k=1}^{m} {\varsigma^k}^2 \\ \text{tal que } y^k = w^T \phi(x^k) + b + \varsigma^k, \quad k = 1, \ldots, m \end{array} \tag{2.65}$$

Esta formulación es equivalente a la función de coste de la regresión en cresta (o "ridge regression"), formulada en el espacio de características. Sin embargo este problema puede no ser resoluble en el espacio primario, y por tanto se pasa al dual mediante la Lagrangiana,

$$\mathcal{L}(w, b, \varsigma; \alpha) = \mathcal{J}_P(w, \varsigma) - \sum_{k=1}^{m} \alpha^k w^T \phi(x^k) + b + \varsigma^k - y^k \tag{2.66}$$

donde α^k son los multiplicadores de Lagrange. La condiciones para la optimalidad vienen dadas por

$$\left\{ \begin{array}{lll} \frac{\partial \mathcal{L}}{\partial w} = 0 & \to & w = \sum\limits_{k=1}^{m} \alpha^k \phi(x^k) \\ \frac{\partial \mathcal{L}}{\partial b} = 0 & \to & \sum\limits_{k=1}^{m} \alpha^k = 0 \\ \frac{\partial \mathcal{L}}{\partial \varsigma^k} = 0 & \to & \alpha^k = \gamma \varsigma^k, \qquad\qquad k = 1, \ldots, m \\ \frac{\partial \mathcal{L}}{\partial \alpha^k} = 0 & \to & w^T \phi(x^k) + b + \varsigma * k - y^k = 0, \quad k = 1, \ldots, m. \end{array} \right. \tag{2.67}$$

Tras la eliminación de las variables w y ${\varsigma}^k$, se obtiene la siguiente solución

$$\left[\begin{array}{l} \text{solve in } \alpha, b \\ \left[\begin{array}{c|c} 0 & 1_v^T \\ \hline 1_v & \Omega + I/\gamma \end{array} \right] \left[\begin{array}{c} b \\ \alpha \end{array} \right] = \left[\begin{array}{c} 0 \\ y \end{array} \right] \end{array} \right] \tag{2.68}$$

donde $y = [y_1, \ldots, y_m]$, $1_v = [1, \ldots, 1]$ y $\alpha = [\alpha_1, \ldots, \alpha_m]$. El truco del kernel o "kernel trick" se aplica aquí como

$$\Omega_{kl} = \phi(x^k)^T \phi(x_l) = K(x^k, x^l) \quad k, l = 1, \ldots, m \tag{2.69}$$

El modelo LS-SVM para aproximación funcional resultante se convierte entonces en

$$y(x) = \sum_{k=1}^{m} \alpha_k K(x, x^k) + b \tag{2.70}$$

donde α^k, b son la solución al sistema de ecuaciones lineales 2.68

El el caso de utilizar kernels RBF, tan solo dos hiperparámetros (γ, σ) tienen que optimizarse, mientras en para las SVMs tradicionales, debían de optimizarse tres parámetros. Además, a las simplificaciones de la formulación citadas anteriormente, se añade el hecho de que para la función de coste de Vapnik, se necesitan optimizar un total de $2 * m$ parámetros (α y α^*), con el agravante de que se trata de un problema de optimización de PC. Sin embargo para las LS-SVMs, el número de parámetros a optimizar es m. Una desventaja de los LS-SVMs, es que no se obtiene una solución dispersa o "sparse" tras la optimización del sistema de ecuaciones. Esto se puede resolver mediante alguna técnica de poda, que en términos generales eliminará aquellos puntos x^k para los que el valor de α sea pequeño.

Nótese además que en el caso de utilizar kernels RBF, la máquina de vectores soporte obtenida se asemeja a una red RBF donde hay una neurona oculta para cada punto de entrada (ver ecuaciones 2.70 y 2.41). Así, mientras que el sobreajuste se controla en las redes RBF reduciendo el número de neuronas ocultas, en las LS-SVM en general se controla mediante el parámetro de regularización γ [Rossi et al., 2006]. La estructura como red neuronal de una LS-SVM quedaría por tanto como se muestra en la figura

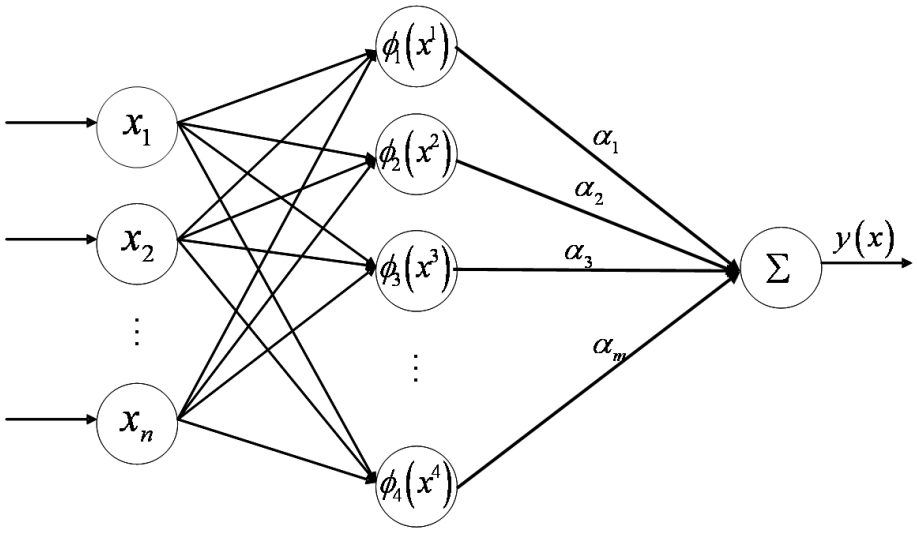


Figura 2.7. *Estructura de una red neuronal de funciones de base radial.*

2.3.4. Obtención de los hiperparámetros en las LS-SVMs

La obtención de los hiperparámetros en una máquina de vectores soporte es una tarea muy complicada. La variación del rendimiento de un modelo SVM no es continua al variar los hiperparámetros, no pudiendo aplicarse por tanto técnicas de descenso en gradiente, y teniendo que probarse distintos valores candidatos aleatoriamente.

La técnica más empleada para evaluar el rendimiento de unos valores dados de hiperparámetros (γ, σ) en una LS-SVM es la validación cruzada. La validación cruzada es una técnica estadística que consiste en particionar el conjunto de muestras de datos en subconjuntos, de forma que se realiza el análisis en uno de los subconjuntos, dejándose los otros para la confirmación y validación de los análisis sobre el subconjunto inicial. En nuestro caso, el análisis llevado a cabo es el entrenamiento de la LS-SVM con los valores prefijados

de hiperparámetros en el primer subconjunto, dejándose la evaluación del rendimiento de la LS-SVM con esos hiperparámetros para el resto del conjunto de datos.

Hay varios tipos de validación cruzada, siendo dos los más conocidos: la validación cruzada en L pliegues (L-fold) y la validación cruzada dejando-uno-fuera (leave-one-out).

La validación cruzada en L pliegues consiste en dividir el conjunto de datos inicial en L subconjuntos. De los L subconjuntos, se deja un subconjunto fuera como datos de validación para comprobar el rendimiento del modelo, y los restantes $L-1$ subconjuntos se utilizan para el entrenamiento. Este procedimiento se realiza L veces, utilizándose cada vez uno de los L subconjuntos como conjunto de validación. Los L resultados de rendimiento obtenidos de cada uno de los subconjuntos pueden usarse para realizar la media y obtener una sola estimación del rendimiento esperado.

La validación dejando-uno-fuera, implica usar una sola muestra de los datos originales como conjunto de validación, dejándose el resto de observaciones como conjunto de entrenamiento. Esto se repite tantas veces como datos disponibles hay, dejando en cada caso una muestra como conjunto de validación. Esta técnica se corresponde con la validación cruzada en L pliegues donde L es el número de datos disponibles.

Dado que disponemos de un método de validación cruzada para estimar el rendimiento de una LS-SVM para unos hiperparámetros dados (γ, σ), el ajuste de los hiperparámetros se suele efectuar realizando una rastreo en un grid bidimensional (una dimensión para cada hiperparámetro) probándose distintas configuraciones de hiperparámetros. La búsqueda en grid ha comprobado ser una buena técnica para encontrar un buen mínimo local de los hiperparámetros. La figura 2.8 muestra un ejemplo de búsqueda en grid de hiperparámetros utilizando LS-SVMlab [LS-,]. En este caso se realiza un doble grid de evaluación, centrándose el segundo en el área del espacio (γ, σ)-dimensional que ha demostrado dar un mejor rendimiento de validación cruzada en la primera pasada en grid.

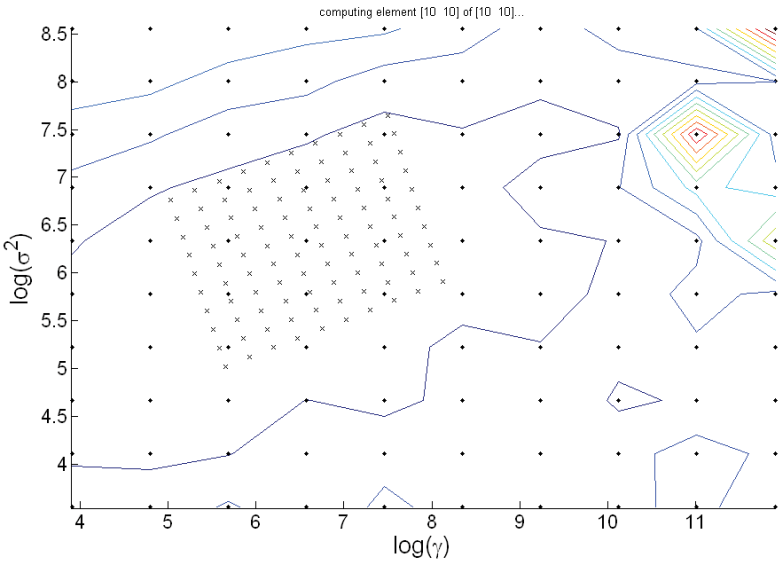


Figura 2.8. *Búsqueda en grid de hiperparámetros* (γ, σ) *en una LS-SVM*

Esta metodología para el cálculo de hiperparámetros no es óptima computacionalmente; nótese que un grid de 10x10 don dos pasadas, con validación cruzada utilizando

10-pliegues, requiere un total de 2000 procesos de entrenamiento de una LS-SVM. Cuando el número de puntos del problema es alto, el coste computacional de la optimización de una LS-SVM se puede volver prohibitivo. Sin embargo, para un número reducido de puntos, este entrenamiento es viable, lo que unido a su eficacia, hacen que los LS-SVM sean una técnica muy deseable para problemas de aproximación funcional. Pueden encontrarse en la literatura otras opciones que intentan evitar el elevado coste computacional de este tipo de técnicas para la optimización de las LS-SVM [Lendasse et al., 2005].

2.4. Selección de Variables para Problemas de Aproximación Funcional. Medida de la Información Mutua

La selección de variables es un paso esencial que hay que llevar a cabo en cualquier problema de modelado a partir de datos de Entrada/Salida. La eliminación de variables redundantes e irrelevantes de un problema, reduce la complejidad en el aprendizaje, aumenta la capacidad de generalización de los modelos aprendidos, y facilita la interpretación del problema y la utilización de los modelos. Un criterio muy utilizado para realizar la selección es la medida de información mutua (IM) de la Teoría de la Información de Shannon. Numerosos trabajos abordan en la literatura la selección de variables usando la información mutua para problemas de clasificación. Sin embargo la estimación de la información mutua entre grupos de variables continuas es un problema mucho más complicado, lo que ha restringido su aplicación en selección de variables. Recientemente sin embargo, gracias a la aparición de un estimador de la IM basado en los k-vecinos más cercanos [Kraskov et al., 2004], han aparecido algunos algoritmos robustos para selección de variables continuas usando IM. En esta sección se introduce el problema de la selección de variables, se explica el concepto de la información mutua, introduciendo brevemente el estimador de la IM basado en los k-vecinos más cercanos de Kraskov et al., revisándose por último las diferentes estrategias que se pueden seguir para diseñar un algoritmo de selección de variables.

2.4.1. Introduction

Supongamos un problema de modelado de una sola salida (Multiple Input, Single Output MISO), disponemos de un conjunto D de M datos de entrada/salida, con variables de entrada $X = [x_1, x_2, \ldots, x_n]$ y variable de salida $Y = y$

$$D = \{z^m = \{\vec{x}^m, y^m\}, 1 \leq m \leq M / y^m = F(\vec{x}^m)\}. \tag{2.71}$$

La selección de variables es un proceso por el que se seleccionan variables útiles en X para obtener una solución eficiente y mejorada al problema de modelado. Idealmente, el proceso de selección de variables, obtendría un subconjunto óptimo $X_G \subset X$ necesario y suficiente para resolver el problema. Dado un conjunto de variables de entrada, definiremos tres tipos de variables en función de su relación con la variable de salida

- Variables irrelevantes: son aquellas variables que carecen de información o relevancia respecto de la variable de salida

- Variables redundantes: son aquellas variables, que pese a tener información respecto de la variable de salida, pueden ser ignoradas al haber otra u otras variables en el

conjunto de variables X que ya contienen esa misma información

- Variables necesarias: son aquellas variables relevantes respecto de la variable de salida, y cuya eliminación puede causar una pérdida de rendimiento del modelo de predicción

La selección del conjunto de variables necesarias para realizar el modelado es un problema muy complejo. Sin embargo, es a su vez un paso necesario a la hora de obtener modelos predictivos. La gran importancia de realizar un proceso adecuado de selección de variables radica en que:

a Existe un gran número de modelos de predicción sufren de la "maldición de la dimensionalidad" en complejidad computacional. El coste computacional en el entrenamiento de dichos modelos crece dramáticamente (a veces exponencialmente) con el número de variables de entrada del problema [Bellman, 1961]

b En general, y aún más acentuadamente en aquellos modelos en los que la predicción se realiza directamente en función de valores cercanos conocidos (k-vecinos más cercanos), está demostrado que la presencia de variables redundantes e irrelevantes conlleva la posibilidad de caer en sobreajuste y en una pobre capacidad de generalización del modelo [Haykin, 1998].

c Desde el punto de vista de la obtención de datos, en principio es más económico recolectar información para un modelo con pocas variables.

d En los modelos en los que se da importancia al entendimiento de éste y explicación de los resultados, es muy importante reducir en lo posible el número de variables de entrada.

Selección de variables y extracción de características

La selección de variables puede englobarse dentro de un procedimiento más general que podríamos llamar "reducción del espacio de entrada". Dentro de la reducción del espacio de entrada tendríamos dos tipos de procedimientos, la selección de variables y la extracción de características. La extracción de características realiza un reemplazo de las variables originales por otro conjunto de variables, que se obtiene normalmente mediante transformaciones lineales o no lineales del conjunto original. De aquí que se diferencie normalmente de forma ficticia entre *variables* (las originales) y *características* (las transformadas) para denotar las variables con las que trabajan un proceso y otro. Los algoritmos de Análisis de Componentes Principales (principal component analysis, PCA) y el PCA basado en kernels (kernel-PCA) son ejemplos muy conocidos de métodos de extracción de características [Haykin, 1998], [Schoelkopf and Smola, 2002], [Suykens et al., 2002].

Los métodos de extracción de características tienen la ventaja de que en el mismo proceso de construcción del nuevo espacio de características, se van seleccionando según su importancia las que mejor explican la distribución de los datos en el espacio original. Por otro lado, los métodos de selección de variables no modifican el espacio original de variables, con lo que el significado (físico, matemático, etc.) original de las variables se conserva a la hora de construir el modelo predictor. Con todo, hay algunos trabajos que

intentan seleccionar variables en función de los componentes principales obtenidos en el PCA [Mao, 2005].

Métodos de envoltura y métodos de filtrado

En general, los métodos de selección de variables se pueden clasificar en dos tipos: métodos de filtrado (filter methods), también llamados métodos independientes del modelo, y métodos de envoltura (o wrapper methods; que ante la dificultad de la traducción de este término, a veces se utiliza la denominación original en inglés) también llamados métodos basados en el modelo. Esta clasificación también es aplicable a los métodos de extracción de características.

Los métodos dependientes del modelo, o métodos de envoltura, utilizan una evaluación del rendimiento del modelo usado para seleccionar el subconjunto de variables más apropiado. Las metodologías independientes del modelo, o métodos de filtrado, no utilizan el modelo final para seleccionar las variable sino que usan tests estadísticos o medidas como la información mutua [Rossi et al., 2006], los gamma tests [Reyhani et al., 2005], etc. para seleccionar el conjunto de variables óptimo, que será después pasado a la metodología de modelado.

La gran ventaja de los métodos de envoltura es que puesto que se utiliza la metodología de modelado para evaluar las variables a seleccionar, el rendimiento final de las variables seleccionadas queda garantizado. En general, los métodos de envoltura suelen obtener mejores resultados que los métodos de filtrado. Por otro lado, al usarse la metodología de modelado en la selección, el coste computacional de este tipo de técnicas puede llegar a ser insostenible.

Los métodos de filtrado son en general una forma "barata" computacionalmente de seleccionar un subconjunto de variables relevantes de un problema. Son técnicas mucho más rápidas que los métodos de envoltura y por ello suelen ser preferibles desde el punto de vista del ratio coste/rendimiento obtenido. En la práctica se pueden utilizar técnicas híbridas, en las que se realiza una primera selección o ranking utilizando una técnica de filtrado, y posteriormente se verifica el rendimiento de las diferentes opciones mediante un método sencillo de envoltura.

2.4.2. Selección de variables mediante información mutua. Estrategias de selección.

La teoría de la información de Shannon ofrece un buen marco teórico para el diseño de técnicas de filtrado de variables, gracias a los conceptos de entropía e información mutua [Cover and Thomas, 1991]. No obstante, la utilización de estos conceptos para problemas continuos (o de aproximación funcional) es un problema muy complejo, al requerir la utilización de técnicas complejas para estimar la distribución de probabilidad. Entre las técnicas para estimar la función de densidad de probabilidad (PDF), podemos encontrar principalmente los histogramas, y las técnicas basadas en kernels [Bonnlander and Weigend, 1994]. Estas técnicas pueden presentar maldición de la dimensionalidad, y en general reducen su aplicabilidad a problemas con pocas variables, y donde el número de muestras disponibles es suficientemente grande.

También en la literatura pueden encontrarse estimadores de la entropía continua basados en los k-vecinos más cercanos. Su extensión a la información mutua se ha producido recientemente por Kraskov et al. [Kraskov et al., 2004] [MIL,]. Una propiedad interesante de este estimador es que se puede usar fácilmente para grupos de variables. Además no sufre de la maldición de la dimensionalidad.

En esta sección introduciremos primeramente los conceptos de entropía continua e información mutua. Posteriormente se revisará el estimador de la información mutua basado en los k-vecinos más cercanos. Finalmente se revisarán las diferentes estrategias de selección de variables que se pueden utilizar para el diseño de algoritmos de selección de variables, explicadas usando el criterio concreto de la información mutua.

Información mutua en variables continuas

Dado un problema de modelado MISO con un conjunto de datos D (ver *eq.* 2.71), el objetivo puede verse desde el punto de vista de la teoría de la información, como reducir en lo posible la incertidumbre sobre la variable dependiente Y. Según la formulación de Shannon, la incertidumbre sobre Y viene medida por su entropía, que se define en el caso continuo como

$$H(Y) = -\int \mu_Y(y) \log \mu_Y(y) dy, \tag{2.72}$$

dado que la función de densidad marginal $\mu_Y(y)$ puede definirse usando la PDF conjunta $\mu_{X,Y}$ de X e Y como

$$\mu_Y(y) = \int \mu_{X,Y}(x, y) dx. \tag{2.73}$$

Dado que se conoce X, la incertidumbre de Y condicionada a que se conoce X, viene dada por la entropía condicionada, que se define como

$$H(Y|X) = -\int \mu_X(x) \int \mu_Y(y|X = x) \log \mu_Y(y|X = x) dy dx. \tag{2.74}$$

La incertidumbre conjunta de $[X, Y]$ viene dada por la entropía conjunta, que se define como

$$H(X, Y) = -\int \mu_{X,Y}(x, y) \log \mu_{X,Y}(x, y) dx dy. \tag{2.75}$$

La información mutua (también llamada entropía cruzada) entre X e Y se define como la cantidad de información que el grupo de variables X me proporciona sobre Y, y se puede expresar como

$$I(X, Y) = H(Y) - H(Y|X). \tag{2.76}$$

En otras palabras, la información mutua $I(X, Y)$ es la pérdida de incertidumbre sobre Y, que se produce cuando se conoce X. Debido a las propiedades de la entropía, la información mutua también se puede definir como

$$I(X, Y) = H(X) + H(Y) - H(X, Y), \tag{2.77}$$

que lleva a

$$I(X, Y) = \int \mu_{X,Y}(x, y) \log \frac{\mu_{X,Y}(x, y)}{\mu_X(x)\mu_Y(y)} dx dy. \tag{2.78}$$

Por tanto, solo se necesita el estimador de la PDF conjunta entre X e Y para estimar la información mutua entre ambos grupos de variables. La estimación de la distribución de probabilidad conjunta puede realizarse mediante varias técnicas. Ahora revisamos una técnica basada en los k-vecinos más cercanos para la estimación de la información mutua.

Estimador de la información mutua usando los k-vecinos más cercanos

El estimador de la IM que se presenta en esta sección presenta las ventajas, frente a otros estimadores, de que sencillo de utilizar para conjuntos de variables, y que es menos sensible al problema de la maldición de la dimensionalidad. Al usar los k-vecinos más cercanos como técnica de base, se reducen los problemas inherentes de trabajar en espacios de dimensionalidad alta.

Se define el espacio $Z = \{X, Y\}$, y se hará uso de la norma máximo para cualquier par de puntos $z = (x, y)$ y $z' = (x', y')$

$$\|z - z'\| = \text{máx}\{\|x - x'\|, \|y - y'\|\}, \tag{2.79}$$

aunque cualquier otra norma podría ser usada. Notamos por $\varepsilon(i)$ la distancia de un punto z_i a su k-ésimo vecino más cercano y por $\varepsilon_x(i)$ y $\varepsilon_y(i)$ las distancias entre los mismos puntos proyectados en los subespacios X e Y respectivamente. Obviamente se cumple que $\varepsilon(i) = \text{máx}\{\varepsilon_x(i), \varepsilon_y(i)\}$.

Contaremos el número $n_x(i)$ de puntos x_j cuya distancia desde x_i es estrictamente menor que $\varepsilon(i)/2$, y de forma similar para y en vez de x. El estimador de la IM es entonces (véase en [Kraskov et al., 2004] la demostración de la convergencia de este estimador)

$$\hat{I}_1(X, Y) = \psi(k) - \frac{1}{N}\sum_{i=1}^{N}[\psi(n_x(i) + 1) + \psi(n_y(i) + 1)] + \psi(N), \tag{2.80}$$

donde ψ es la función digamma dada por

$$\psi(t) = \frac{\Gamma'(t)}{\Gamma(t)} = \frac{d}{dt}\ln\Gamma(t). \tag{2.81}$$

La función ψ satisface la recursión $\psi(x + 1) = \psi(t) + 1/x$ y $\psi(1) = C$ donde $C = -0{,}5772156\ldots$ es la constante de Euler-Mascheroni.

Otra alternativa es reemplazar $n_x(i)$ y $n_y(i)$ por el númer de puntos que cumple que $\|x_i - x_j\| \leq \varepsilon_x(i)/2$ y $\|y_i - y_j\| \leq \varepsilon_y(i)/2$. El estimador de la IM quedaría por tanto

$$\hat{I}_2(X, Y) = \psi(k) - \frac{1}{k} - \frac{1}{N}\sum_{i=1}^{N}[\psi(n_x(i)) + \psi(n_y(i))] + \psi(N). \tag{2.82}$$

En general, ambas formulaciones dan resultados similares. Aunque para dimensiones muy altas, $\varepsilon(i)$ tenderá a ser mayor que el marginal $\varepsilon_x(i)$, y el segundo algoritmo sería preferible.

Una implementación de ambos estimadores puede obtenerse de [MIL,]. Para una explicación más detallada de estos estimadores véase la referencia [Kraskov et al., 2004].

Un parámetro que es necesario fijar para estos estimadores es el valor de k. Ambos estimadores dependen en su funcionamiento de este parámetro. En [Harald et al., 2004] por ejemplo, los autores recomiendan, que se de a k un valor intermedio para resolver el compromiso entre bias y varianza.

Estrategias de selección de variables

Como se ha explicado, la selección de variables pretende identificar el subconjunto de variables óptimo $X_G \cong X$ para realizar el modelado. Vamos a explicar las estrategias más comunes que se pueden encontrar en algoritmos de selección de variables, tomando

como criterio la información mutua. Estas estrategias son genéricas, y no exclusivas de un criterio específico, las explicamos ahora para una mejor comprensión tanto de la IM como de las mismas estrategias. En cualquier caso estas estrategias son generales, pudiéndose encontrar algoritmos híbridos entre estas diferentes estrategias o bien que presenten alguna estrategia específica distinta de las aquí expuestas.

- Selección hacia adelante: En la selección hacia adelante, se parte de un subconjunto vacío $X_G = []$, y se van añadiendo paso a paso variables relevantes a dicho conjunto [François et al., 2007], [Chow and Huang, 2005]. En esta estrategia se añade en cada paso al conjunto X_G la variable x_j que aumente más la información del conjunto seleccionado respecto de la variable de salida, esto es

$$X_{G_{new}} = \left\{ X_G \cup x_j : \begin{array}{r} I(\{X_G \cup x_j\}, Y) > I(\{X_G \cup x_v\}, Y) \\ \forall v \in \{X - X_G\} \, / v \neq j \end{array} \right\}. \tag{2.83}$$

  Esta estrategia de selección se detendría en el momento en que la información mutua al añadir una nueva variables sea similar a la información mutua anterior ($I(X_{G_{new}}, Y) \simeq I(X_G, Y)$). Esto indicaría que se ha encontrado el conjunto más pequeño de variables que proporcionan más información respecto de la variable de salida. Este tipo de estrategia puede funcionar muy bien cuando las variables están cerca de ser independientes entre sí. Sin embargo, esta estrategia presenta el problema de que puede ocurrir que no se identifique algún conjunto de variables tal que separadas no ofrecen información respecto a la variable salida, pero en conjunto sí, no seleccionándose por tanto el conjunto óptimo.

- Selección hacia atrás: Esta estrategia es contraria a la anterior, es decir, comienza considerando el conjunto completo de variables $X_G = X$ y va eliminando de X_G una variable en cada paso [Koller and Sahami, 1996]. Al contrario que la selección hacia adelante, al comenzar con el conjunto completo de variables, es de alguna manera más fácil diseñar criterios más restrictivos que no eliminen del conjunto ninguna variable relevante. Sin embargo, presenta el inconveniente de tener que trabajar desde el comienzo con el conjunto completo de variables, que puede ser costoso y contraproducente desde el punto de vista de la dimensionalidad.

  En esta estrategia se elimina en cada paso del conjunto X_G la variable x_j tal que al eliminarla no disminuya la información mutua del conjunto de variable restante respecto de la variable de salida, esto es

$$X_{G_{new}} = \{X_G - \{x_j\} / I(\{X_G - x_j\}, Y) \simeq I(\{X_G\}, Y)\} \tag{2.84}$$

  Esta estrategia de selección se detendría en el momento en que la información mutua al eliminar una variables sea menor que la información mutua anterior ($I(X_{G_{new}}, Y) < I(X_G, Y)$). Esto indicaría que se ha encontrado el conjunto más pequeño de variables que proporcionan más información respecto de la variable de salida.

- Selección paso a paso: Esta estrategia es una mezcla de las anteriores. En cada paso se puede eliminar o añadir una variable, dependiendo del criterio seleccionado. Pretende unir las ventajas de las dos estrategias [Rossi et al., 2006]

- Selección del subconjunto óptimo: Visto el inconveniente de trabajar con estrategias iterativas, esta estrategia pretende seleccionar directamente el subconjunto óptimo

$X_G \subset X$ de entre todas las $2^n - 1$ posibilidades, que es el más pequeño y que proporciona más información mutua respecto de la variable de salida

$$X_G = \left\{ X_G \subset X / \begin{array}{l} I(X_G, Y) > I(X_v, Y) \text{ ó} \\ I(X_G, Y) \simeq I(X_v, Y) \text{ y } |X_G| < |X_v| \end{array} \forall X_v \cong X \right\} \quad (2.85)$$

Esta metodología fue propuesta para información mutua en [Sorjamaa et al., 2005], [Bonnlander and Weigend, 1994] y parcialmente en [Benoudjit et al., 2004]. La ventaja de esta metodología es que en principio, teniendo un estimador preciso que me pueda proporcionar $I(X_G, Y)$ para cualquier tamaño de X_G, esta selección sería óptima. Sin embargo al trabajar con conjuntos con un gran número de variables, por un lado, los estimadores de la IM pierden eficacia al disponer de un conjunto limitado de puntos. Además, el número de todos los subconjuntos posibles es de 2^n, lo que puede ser en muchos casos computacionalmente inviable [Rossi et al., 2006].

Capítulo 3

Sistemas Neuro-Difusos tipo Takagi-Sugeno-Kang y Redes de Función de Base Radial para Problemas de Aproximación Funcional

3.1. Introducción

En este capítulo se sientan las motivaciones de los modelos difusos TSK presentados en los siguientes tres capítulos. El aprendizaje automático de sistemas difusos para problemas de aproximación funcional viene siendo un área muy importante de investigación en los últimos años. Existen un número de escollos importantes que surgen a la hora de tratar este tipo de problemas de manera automática. Vistas las bases teóricas de los sistemas difusos y de las redes RBF en el tema anterior, este tema introduce brevemente el aprendizaje de los sistemas neuro-difusos TSK, junto con algunas de las opciones existentes en su diseño y los problemas que se presentan en dicho aprendizaje; también se resume el aprendizaje de las redes RBF, que presenta similitudes como se comentó a los sistemas TSK. Finalmente se introducen los tres modelos propuestos y la motivación en su investigación con las ventajas que presentan respecto de los modelos neuro-difusos TSK y RBF tradicionales.

3.2. Alternativas para el Particionamiento del Espacio de Entrada en Sistemas Difusos tipo TSK

Un sistema neuro-difuso tipo Takagi-Sugeno-Kang (TSK) consta en un conjunto de 'K' reglas "SI-ENTONCES"que típicamente tienen la forma

$$Regla^k : \text{SI } x_1 \text{ es } \mu_1^k \text{ Y } \ldots \text{ Y } x_n \text{ es } \mu_n^k \text{ ENTONCES } y = Y(\vec{x})^k \tag{3.1}$$

donde mu_i^k son conjuntos difusos caracterizados por funciones de pertenencia $mu_i^k(x_i)$, y $Y(\vec{x})^k$ es un polinomio función de el vector de entrada $\vec{x}$.

La salida de un sistema difuso con reglas en la forma mostrada en la ecuación 3.1 se puede expresar utilizando media ponderada como operador de agregación como

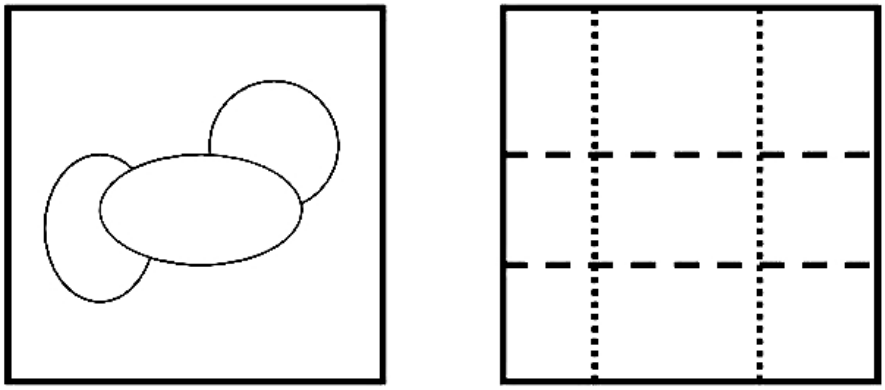
$$F(\vec{x}) = \frac{\sum_{k=1}^{K} \mu^k(\vec{x}) Y(\vec{x})^k}{\sum_{k=1}^{K} \mu^k(\vec{x})} \tag{3.2}$$

dado que cada $\mu^k(\vec{x})$ es el valor de activación para los antecedentes de la regla k, que se

puede expresar como

$$\mu^k(\vec{x}) = \mu_1^k(x_1)\mu_2^k(x_2)\dots\mu_n^k(x_n). \tag{3.3}$$

Los sistemas difusos, según esta forma genérica de sus reglas, pueden presentar dos tipos de particionamiento del espacio de entrada (ver figura 3.1). Por un lado, el particionamiento basado en clustering realiza una subdivisión marginal del espacio de entrada, que depende del número de reglas usadas para alcanzar el objetivo [Gonzalez et al., 2002], [Angelov and Filev, 2004], [Paiva and Dourado, 2001]. Esta subdivisión del espacio de entrada es la que está asociada normalmente a las redes RBF (ver subsección 2.2.2). Las reglas, o neuronas ocultas, se localizan en las zonas del espacio donde se necesitan. Este sistema es más adecuado por ejemplo para problemas de predicción de series temporales en los que los datos disponibles están más centralizados en algunas regiones del espacio de entrada. Para problemas de complejidad media, además, es en principio la única opción a utilizar debido al inconveniente de los sistemas basados en grid de la maldición de la dimensionalidad en complejidad computacional que se comenta a continuación. Sin embargo, este tipo de particionamiento tiene el inconveniente de que puede que no se cubra todo el espacio de entrada del problema. Además este tipo de particionamiento tiene asociada una capacidad de interpretabilidad menor, al disponer de una función de pertenencia en cada variable por cada regla en el sistema, dificultando la asignación de variables lingüísticas a las mismas [Guillaume, 2001].



$a)$ $b)$

Figura 3.1. *a) Particionamiento del espacio basado en clustering. b) Particionamiento del espacio basado en grid*

Por otro lado, los sistemas difusos basados en grid (GBFS) realizan una cobertura exhaustiva de la totalidad del espacio de entrada [Pomares et al., 2000], [Rojas et al., 2000], [Wang and Mendel, 1992b]. Sin embargo, este último enfoque sufre de un inconveniente muy importante que es la maldición de la dimensionalidad en complejidad del modelo [Bellman, 1961]: el número de reglas del modelo se incrementa exponencialmente con el número de variables de entrada, y el número de funciones de pertenencia por variable. El número de reglas K de un sistema difuso basado en grid viene dado por

$$K = \prod_{i=1}^{n} numMF_i \tag{3.4}$$

donde $numMF_i$ es el número de funciones de pertenencia en la variable de entrada i. Dado un problema de modelado de complejidad media, puede ocurrir que obtengamos un sistema con cientos o incluso miles de reglas. Esto tiene serios inconvenientes como son: la dificultad de poder realizar cualquier tipo de tratamiento computacional posterior sobre un conjunto excesivamente grande de reglas, pero sobre todo la pérdida de el comprensión sobre el funcionamiento del modelo. Esta característica es deseable para utilizar los sistemas neuro-difusos en problemas de modelado. Para problemas de cierta complejidad, los sistemas difusos basados en grid pueden perder el entendimiento y la interpretabilidad del modelo y por tanto las ventajas de aplicar la lógica difusa desaparecen.

3.3. Aprendizaje de Sistemas Difusos tipo TSK

Dado un problema de aproximación funcional, el aprendizaje de un sistema difuso tipo TSK para el modelado de dicho problema consta de tres tareas diferentes:

- cálculo de los consecuentes óptimos de las reglas
- ajuste de los parámetros de las funciones de pertenencia en los antecedentes (o ajuste de parámetros)
- identificación de particionamiento óptimo del espacio de entrada (identificación de la estructura)

Existen un gran número de metodologías de aprendizaje de sistemas difusos. De entre éstas distinguimos principalmente entre 2 tipos, técnicas automáticas y técnicas basadas en algoritmos genéticos o sistemas difusos genéticos (Genetic Fuzzy Systems, ver los trabajos [Cordon et al., 2004], [Cordon et al., 2001]). Los algoritmos genéticos son altamente eficaces en la búsqueda de soluciones, pero sin embargo presentan el problema del coste computacional, que para sistemas difusos complejos puede ser excesivo. En los modelos presentados en los siguientes capítulos, se utilizan metodologías automáticas para la optimización de los modelos [Pomares et al., 2000], [Pomares et al., 2002], [Rojas et al., 2000]. Dada la formulación general expuesta en el Capítulo 2, exponemos brevemente ahora los puntos básicos de cada paso de la optimización

3.3.1. Cálculo de consecuentes óptimos

La función de salida de un sistema difuso tipo TSK es lineal con respecto al todos los parámetros de los consecuentes (ver ecuación 2.40, ecuación B.1 y apéndice B). Así pues, se podrían usar un gran número de métodos matemáticos para extraer la solución óptima de los coeficientes de los consecuentes de las reglas. Entre estos métodos, la descomposición de valores singulares (SVD) [Golub and Loan, 1961] se utiliza en esta memoria puesto que además de calcular los valores óptimos, permite realizar un ranking de los coeficientes más relevantes, pudiéndose eliminar valores con poca peso que en algunos casos llevan a redundancia y a soluciones inmanejables [Golub and Loan, 1961].

3.3.2. Ajuste de parámetros

El término ajuste de parámetros se puede utilizar en la literatura, para referirse tanto a los parámetros en los consecuentes de las reglas como a los parámetros en los antecedentes.

En este trabajo referiremos como ajuste de parámetros a la optimización de los parámetros de las funciones de pertenencia. Las localización de las funciones de pertenencia de las variables de entrada, determina la subdivisión del espacio de entrada, según la organización en reglas del sistema. Los parámetros de las funciones de pertenencia son parámetros no lineales respecto de la función de salida, y por tanto deben ser optimizados mediante alguna función de optimización no lineal como las basadas en descenso en gradiente. Dependiendo del tipo de función de pertenencia utilizado en el modelo difuso, se tendrán que optimizar más o menos parámetros. En general es preferible, por eficiencia, utilizar una partición de funciones de pertenencia (ver apéndice A) en la que tan solo haga falta optimizar los centros de las mismas, y obviar los extremos y/o parámetros de anchura de las mismas.

Las técnicas de descenso en gradiente presentan el inconveniente de que pueden caer en mínimos locales. Así pues siempre es conveniente realizar una fase previa de inicialización de los parámetros, antes de la ejecución del descenso en gradiente, de forma que se asegure la obtención de un "buen" óptimo local. Numerosos trabajos han indicado la necesidad de disponer de una buena localización inicial de los centros [Pomares et al., 2000], [Herrera et al., 2005a], pues es clave en el rendimiento final del sistema tras el proceso de búsqueda local de parámetros.

3.3.3. Identificación de la estructura

La identificación de la estructura consiste en encontrar la mejor forma de partición del espacio de entrada, según unos objetivos de precisión, interpretabilidad y/o complejidad del modelo final. En el caso de sistemas difusos basados en clustering, la identificación de la estructura consiste en encontrar el número óptimo de reglas. Igual ocurre en el caso de sistemas difusos basados en grid, aunque el enfoque varía debido al particionamiento del espacio de entrada que se realiza, consistiendo la identificación de la estructura óptima en encontrar el número de funciones de pertenencia óptimo en cada variable (que igualmente definen el número de reglas del sistema). En cualquier caso, a identificación de la estructura óptima requiere de las fases de ajuste de parámetros y de obtención de consecuentes óptimos. Averiguar el particionamiento del espacio de entrada óptimo requiere de una metodología eficiente y eficaz para las otras dos fases a la hora de comprar distintas estructuras.

3.4. Aprendizaje de Redes RBF

Las redes RBF presentan un procedimiento de aprendizaje similar a los sistemas TSK basados en clustering. Disponen igualmente de una tarea de identificación del número óptimo de neuronas, que es siempre la tarea más compleja. Una vez fijado el número de neuronas, el procedimiento de optimización conlleva los siguientes pasos

- Inicialización de los centros de las RBF
- Inicialización de los radios de las RBF
- Cálculo de los valores óptimos para los pesos
- Optimización de los parámetros (centros y radios) de las neuronas ocultas

De nuevo en el caso de las redes RBF, hay diversas ramificaciones en la investigación para la optimización de estos sistemas. Distinguimos también aquí entre algoritmos genéticos y métodos automáticos. En este último caso, al igual que pasaba con los sistemas

difusos TSK, es fundamental disponer de un algoritmo eficiente para la obtención de los parámetros óptimos de las neuronas ocultas, que vaya compaginado con un método adecuado para la inicialización de los centros de la red, con el fin de obtener resultados óptimos. La obtención de los consecuente óptimos se realiza de forma similar a los sistemas difusos (ver sección 2.2.2), también mediante la obtención de un sistema de ecuaciones lineales, que se puede resolver utilizando por ejemplo descomposición de valores singulares (SVD).

3.5. Problemas de Interpretabilidad en el Aprendizaje de los Sistemas Difusos tipo TSK

En la mayoría de los sistemas TSK presentes en la literatura, el consecuente de la regla es simplemente un escalar, esto es, un sistema TSK de orden 0. Sin embargo, al no utilizar conjuntos difusos en la salida, los sistemas TSK [Takagi and Sugeno, 1985] pueden presentar problemas de interpretabilidad desde su misma base, especialmente cuando se utilizan consecuentes que son funciones lineales o polinómicas de grado superior de las variables de entrada [Yen et al., 1998].

La utilización de consecuentes constantes, pese a obtener excelentes resultados de aproximación para muchos problemas, puede hacer que el número de reglas necesarias para alcanzar un cierto umbral de rendimiento sea excesivamente alto. Esto puede hacer que la lógica difusa no sea útil en algunos problemas donde la comprensión y la interpretabilidad del modelo es un requisito. Así, para ciertos problemas puede ser conveniente utilizar reglas con consecuentes de orden mayor con el objetivo de reducir el número de reglas; sin embargo como se ha comentado, la interpretabilidad de las reglas también puede dificultarse.

El objetivo de los modelos que se presentan en los temas siguientes de esta tesis es superar ambos problemas clave en el diseño de modelos difusos TSK para aproximación funcional: la maldición de la dimensionalidad en la complejidad del modelo, y la pérdida de interpretabilidad del modelo.

Un gran número de trabajos han tratado el problema de la pérdida de la interpretabilidad en el modelado difuso [Guillaume, 2001], estando la mayoría de ellos enfocados sobre los sistemas Mamdani [Paiva and Dourado, 2004]. En general hay diferentes aspectos que se relacionan con la interpretabilidad en los sistemas difusos en general [Guillaume, 2001]

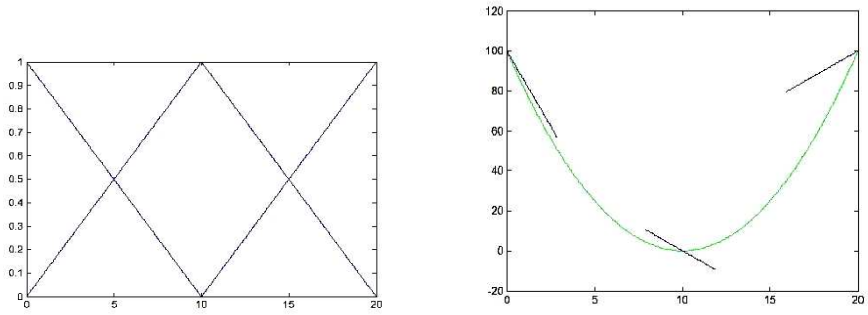
- Es necesario que tras el entrenamiento del modelo se mantenga un número reducido de reglas, con el objetivo de conservar la comprensión y entendimiento de funcionamiento descriptivo del sistema difuso. El funcionamiento descriptivo del sistema viene dado por las reglas obtenidas tras el proceso de modelado. Si se dispone de un número excesivo de éstas, la capacidad de interpretabilidad del modelo se pierde.

- En segundo lugar es muy importante mantener la transparencia en el particionamiento del espacio de entrada, esto es, mantener limitado el solapamiento del soporte de las distintas reglas que conforman el sistema. Cuando existe un solapamiento excesivo entre los conjuntos difusos, la asignación de etiquetas lingüísticas se dificulta, haciendo que la capacidad de comprensión de las reglas pueda menguar [Paiva and Dourado, 2004], [Setnes et al., 1998].

- El número de antecedentes de las reglas difusas se considera inversamente proporcional a la interpretabilidad de las mismas. Así siempre serán preferibles sistemas de

reglas con pocos antecedentes [Guillaume, 2001].

- Por último, y más íntimamente relacionado con los sistemas TSK, que carecen de variables lingüísticas en la salida, los modelos locales (reglas) deben ser coherentes con las salidas del modelo en su área de influencia. Los consecuentes polinómicos de las reglas deben ser reflejo del funcionamiento del modelo en el área del espacio de entrada cubierta por los antecedentes de las mismas. Esto es, se debe mantener la optimización global del sistema, pero sin perder la optimización de los submodelos o modelos locales del sistema (reglas). En general esto no ocurre en los sistemas que se centran en la optimización global (ver figura 3.2) [Yen et al., 1998], [Johansen and Babuska, 2003]



$a)$ $b)$

Figura 3.2. *a) Distribución de MF utilizadas para el ejemplo con función objetivo a aproximar:* $y = (x-10)2$ *b) Función original, aproximación obtenida y modelos locales lineales para cada una de las tres reglas, usando un modelo TSK de orden 1. La salida del modelo global aproximador obtiene un error indistinguible visualmente, sin embargo, los modelos locales carecen de significado en sus áreas de influencia.*

3.6. Problemas en el Aprendizaje de Redes RBF

Las redes RBF, teniendo en cuenta su equivalencia con los sistemas TSK, y como sistemas neuro-difusos, sufren en principio de los mismos inconvenientes anteriormente comentados de los sistemas TSK desde el punto de vista de la interpretabilidad. Una de las grandes ventajas de afrontar un problema de modelado mediante redes RBF es la posibilidad de extraer un conjunto de reglas interpretables del modelo.

Es por esto que un excesivo número de neuronas, un alto solapamiento entre las áreas de influencia de las neuronas, la cobertura de un número excesivo de variables en cada función de base radial [Awad, 2005], y la falta de significado de los pesos de las funciones en su ámbito de influencia, son características indeseables en el resultado de un proceso de entrenamiento de una red RBF.

3.7. Modelo Neuro-Difusos Avanzados Propuestos en esta Tesis

Dados los conceptos básicos sobre sistemas difusos TSK y redes RBF expuestos en el capítulo 2, con las posibilidades de diseño, con sus ventajas e inconvenientes, que se presentan a la hora de abordar un problema de aproximación funcional, y los inconvenientes que se presentan en el entrenamiento de este tipo de sistemas que acabamos de comentar en las secciones 3.5 y 3.6, pasamos a introducir brevemente los tres modelos neuro-difusos TSK modificados que se proponen en los siguientes tres temas de esta memoria y que pretenden resolver al menos parcialmente estos inconvenientes.

En primer lugar, los sistemas TaSe y TaSe-NF se centran en la optimización de la propiedad de interpretabilidad propia de los sistemas TSK que es la consistencia de las salidas de las reglas con el comportamiento en su área de influencia. Ambos sistemas presentan la propiedad de que los consecuentes (pesos) de sus reglas (neuronas) son un polinomio que equivale al desarrollo en serie de Taylor truncado de la salida del sistema en torno al centro de la regla (neurona). Al utilizar este concepto fundamental en la teoría de la aproximación de funciones, estos modelos resuelven de forma satisfactoria el susodicho problema del entrenamiento de estos modelos.

El modelo difuso TSK modificado TaSe presentado en el capítulo 4 utiliza particionamiento del espacio de entrada en grid. Gracias a la utilización de un tipo de función de pertenencia especial obtiene, además de la ventaja comentada, un particionamiento transparente del espacio de entrada. Se consigue además una reducción del número de reglas necesarias para obtener una aproximación con un buen rendimiento, gracias a la utilización de consecuentes de orden mayor de 0 o 1. Esta última ventaja viene avalada además por un estudio estadístico ANOVA [Fisher, 1936] de los parámetros en un sistema difuso que influyen más en el rendimiento de los modelos.

El sistema TaSe-NF, presentado en el capítulo 5, explota la equivalencia entre los modelos TSK y RBF, y obtiene un modelo neuro-difuso con características equivalentes a las del modelo TaSe, pero para un particionamiento en clustering en vez de basado en grid. Obtiene un particionamiento transparente en el espacio n-dimensional, gracias a la utilización de un operador especial de agregación, que además permite la optimización adecuada de los pesos de las neuronas para obtener el desarrollo en serie de Taylor truncado respecto de los centros de las mismas. Este modelo en principio solo puede utilizar consecuentes de orden 0 y 1, aunque admitiría usar consecuentes de orden mayor cambiando el tipo de función de base radial (gaussiana por defecto).

Por otro los sistemas multigrid, presentados en el capítulo 6, intenta evitar la maldición de la dimensionalidad en la complejidad de los sistemas difusos mediante la utilización de diversos sub-grids (en vez de uno solo n-dimensional), con un número reducido de variables de entrada en cada uno de ellos. El objetivo es por tanto perder parcialmente al menos el crecimiento exponencial en el número de reglas del sistema TSK dado en la ecuación 3.4. La transparencia se garantiza también en estos sistemas gracias a la utilización de particionamiento triangular del espacio de entrada (ver apéndice A).

Capítulo 4

El Modelo TaSe para Problemas de Aproximación Funcional

Típicamente, los modelos TSK se han considerado como una herramienta potente para problemas de aproximación funcional, debido a su capacidad para contemplar relaciones complejas entre variables de entrada utilizando reglas sencillas, cuyos consecuentes son funciones simples de las variables de entrada. Sin embargo los sistemas TSK presentan el problema en su entrenamiento de la pérdida de interpretabilidad, lo que hace que la idoneidad de estos sistemas para problemas complejos en los que la interpretabilidad del modelo obtenido sea fundamental sea cuestionable. En este capítulo se presenta una nueva propuesta de modelo TSK, que denominamos modelo TaSe, que obtiene un sistema eficaz e interpretable, que hace uso de la expansión en serie de Taylor de una función en torno a un punto, para aproximar la función objetivo, usando un número reducido de reglas. Asimismo, se proporciona una metodología completa de aprendizaje para la obtención de la estructura óptima del modelo TaSe, así como sus parámetros óptimos en los antecedentes y consecuentes.

Además este capítulo se complementa con un análisis estadístico sobre efecto en el rendimiento, de los parámetros más importantes del diseño de un sistema difuso tipo TSK. Se da un relevancia especial al orden polinómico de los consecuentes de las reglas del modelo. Como se muestra en los resultados del análisis ANOVA, el orden de las reglas usadas es estadísticamente significativo en el rendimiento del modelo, lo que indica la conveniencia, al menos bajo ciertas condiciones, de utilizar consecuentes de orden alto. Los resultados muestran la conveniencia de utilizar el modelo TaSe para una amplia gama de problemas de modelado, puesto que puede usar reglas de orden alto para obtener sistemas más sencillos, con reglas interpretables y sin afectar negativamente a la complejidad computacional en el entrenamiento.

4.1. Introducción

El modelo TaSe es un sistema difuso TSK con particionamiento del espacio de entrada basado en grid. Esto permite obtener una partición transparente del espacio de entrada que mejora la interpretabilidad de las reglas. Con respecto a la reducción en el número de reglas, se conocen pocos trabajos que hayan abordado la posibilidad de utilizar en las reglas consecuentes de orden alto. Demostraremos la aptitud de este tipo de reglas para problemas de aproximación funcional y su eficacia, a la vez que reducen el número de reglas necesarias para realizar la aproximación. La metodología propuesta además, obtiene un conjunto de reglas interpretable, incluyendo el hecho de que los modelos locales obtenidos

tienen una alta información sobre el comportamiento del sistema en torno a esos puntos, gracias a la utilización de un concepto clave en la teoría de la aproximación de funciones como es la expansión en serie de Taylor de una función en torno a un punto.

Presentamos también un método automático para la optimización de los parámetros dado un conjunto de datos de entrada/salida y una configuración dada de funciones de pertenencia por variable. Este algoritmo automático encuentra una localización pseudo-óptima de los centros de las funciones de pertenencia para cada variable, con el objetivo de minimizar el error de aproximación (en el sentido de mínimos cuadrados). La identificación de la estructura es también tratada en el modelo TaSe. Se presenta un algoritmo de identificación automático incremental, basado en previos métodos para sistemas TSK tradicionales, adaptado a reglas con consecuentes polinomiales de orden.

Este capítulo se organiza de la siguiente manera:

- La sección 2 describe los sistemas TSK con reglas de orden alto, así como la obtención de los coeficientes óptimos en los consecuentes y las ventajas de utilizar este tipo de reglas.

- La sección 3 trata la estimación de los modelos locales del sistema TaSe y su interpretabilidad; propiedades que se alcanza gracias al uso de un tipo especial de funciones de pertenencia y una estructura en los consecuentes que asemeja la expansión en serie de Taylor en torno a un punto (centro de la regla).

- La sección 4 expone el método utilizado para el ajuste de parámetros, esto es, para la localización de un óptimo de los parámetros que definen el particionamiento del espacio de entrada del sistema.

- La sección 5 presenta el algoritmo que trata el problema de la identificación de la estructura en el sistema TaSe.

- La sección 6 presenta y analiza los resultados que se obtienen al aplicar el modelo a tres ejemplos significativos utilizados habitualmente en la literatura.

- La sección 7 presenta un análisis ANOVA que compara los resultados al utilizar el sistema Tase con otras posibilidades de diseño de sistemas TSK, mostrándose la conveniencia de utilizar el sistema TaSe en problemas de aproximación funcional, tanto por las ventajas respecto de complejidad e interpretabilidad, como por la capacidad de generalización que presenta.

- Finalmente la sección 7 presenta las conclusiones que se sacan del trabajo expuesto en este capítulo.

4.2. Sistemas Difusos TSK con Consecuentes de Orden Polinómico alto

La utilización de consecuentes de orden cero (constantes) en un sistema TSK ha demostrado proporcionar un buen rendimiento para numerosos ejemplos de aplicación

[Pomares et al., 2002], [Rojas et al., 2000], [Wang, 1994]. El principal problema que presentan estos modelos TSK es el inmanejable número de reglas que pueden surgir para un problema de complejidad media, según se deriva de la ecuación 3.4.

Esta función exponencial del número de reglas de un sistema TSK basado en grid, puede ocasionar que el sistema resultante sea inmanejable e incomprensible, así pues perdiendo su utilidad. Han surgido en los últimos años en la literatura, algunas metodologías de aprendizaje que intentan evitar este problema con éxito parcial (ver refer. [Chung, 2000], [Brockman and Huwendiek, 2000], [de Souza et al., 2002]). En cualquier caso estas metodologías rompen con la estructura en grid original para resolver el problema, por tanto perdiendo las ventajas e intuición que proporciona la división del espacio de entrada en grid.

Como J.J.Buckley [Buckley, 1993] propuso, los consecuentes de las reglas TSK se pueden generalizar de la forma

$$Regla^k : \text{SI } x_1 \text{ es } \mu_1^k \text{ Y } \ldots \text{ Y } x_n \text{ es } \mu_n^k \text{ ENTONCES } y = Y(\vec{x})^k \tag{4.1}$$

donde $Y(\vec{x})^k$ es un polinomio de cualquier orden. Así para polinomios de orden s, los consecuentes tomarán la forma

$$Y_k(\vec{x}) = w_0^k + \vec{w}_1^k \cdot \vec{x} + \vec{x}^T W_2^k \vec{x} + ... + \left\langle W_s^k (\vec{x} \otimes ... \otimes \vec{x}) \right\rangle \tag{4.2}$$

donde w_0^k es un coeficiente de orden 0, $\vec{w}_1^k$ es un vector con coeficientes de orden 1, W_2^k es una matriz triangular con coeficientes de orden dos del polinomio y W_s^k es una matriz triangular s-dimensional de coeficientes, siendo $\otimes$ el producto tensorial.

En este trabajo, se usarán consecuentes polinomiales de orden alto con el objetivo de vencer el problema de la "maldición de la dimensionalidad" en la complejidad del modelo TSK. La principal ventaja de este tipo de reglas difusas, comparando con los sistemas que incluyen reglas de orden 0 o de orden 1 (o lineales), es el incremento en poder de aproximación que cada regla es capaz de proporcionar por sí misma. Esto es, es de esperar que se necesite un número menor de reglas con consecuente de orden polinomial superior, para identificar la función subyacente de un conjunto de entrenamiento de entrada/salida en un problema de modelado continuo.

Con el objetivo de ver cómo se modifica el funcionamiento del sistema difuso aproximador utilizando consecuentes de orden superior, recuperemos la expresión de salida de un sistema difuso para una entrada $\vec{x}$, considerando agregación por media ponderada

$$F(\vec{x}) = \frac{\sum_{k=1}^{K} \mu^k(\vec{x}) \cdot Y^k(\vec{x})}{\sum_{k=1}^{K} \mu^k(\vec{x})} \tag{4.3}$$

siendo $\mu^k(\vec{x})$ el valor de activación de la regla k. Por simplicidad, considérese un ejemplo unidimensional y una partición triangular sobre la variable de entrada [Ruspini, 1969], [Rojas et al., 2000] con dos funciones de pertenencia. En este caso, dos reglas $\{k = 1, 2\}$ se activarán dado cualquier punto en el intervalo de entrada.

Tomando reglas de orden 0, la salida puede expresarse (dada la propiedad de suma-a-uno [Pomares et al., 2000]) como

$$\begin{aligned} F(x) = \mu_1(x) \cdot y_1 + \mu_2(x) \cdot y_2 = \mu_1(x) \cdot y_1 + (1 - \mu_1(x)) \cdot y_2 = \\ = y_2 + \mu_1(x) \cdot (y_1 - y_2) \end{aligned} \tag{4.4}$$

por tanto resulta en una expresión polinomial de orden uno en x (al ser $\mu_1(x)$ un polinomio de primer orden [Pomares et al., 2000]).

Considerando ahora reglas con consecuentes polinomiales de orden s, la salida de la regla pasa a ser

$$F(x) = Y_2(x) + \mu_1(x) \cdot (Y_1(x) - Y_2(x)) \tag{4.5}$$

lo que resulta en una expresión polinomial de orden $s+1$ con coeficientes que se obtienen a partir de las expresiones de los consecuentes de las dos reglas adyacentes, siendo por tanto un aproximador mucho más potente.

Ahora pasamos a comentar la estimación de los coeficientes óptimos de los consecuentes polinomiales de orden alto de las reglas, a partir del conjunto de datos de entrada/salida. El enfoque de mínimos cuadrados (Least Squares Error, LSE) es comúnmente el método utilizado para optimizar dichos coeficientes. El LSE intenta minimizar la función de error

$$J = \sum_{m \in D} \left(y^m - F(\vec{x}^m)\right)^2 \tag{4.6}$$

donde y^m es la salida deseada para el punto $\vec{x}^m$ en el conjunto de datos D y $F(\vec{x}^m)$ es la salida de nuestro sistema aproximador.

En el caso particular de coeficientes polinomiales de segundo orden, el número de parámetros a ser optimizados por mínimos cuadrados viene dado por

$$K \cdot \left(1 + n + {}^1\!/_2 \left(n^2 + n\right)\right) \tag{4.7}$$

donde K es el número de reglas y n es la dimensión del espacio de entrada.

Nótese la gran complejidad del problema cuando se dispusiese de un gran número de variables de entrada con un número moderado de funciones de pertenencia por variable. El grado de redundancia en los parámetros podría ser considerable, haciendo que el problema esté "mal condicionado". Debido a que la salida de la función es lineal con respecto a todos los parámetros de los consecuentes (ver apéndice B), se podrían usar un buen número de métodos matemáticos para extraer una solución óptima [Golub and Loan, 1961]. De entre estos métodos, la descomposición en valores singulares (SVD) ha sido implementada satisfactoriamente puesto que permite descartar valores poco significativos, evitando la redundancia y soluciones inmanejables. La eficacia de este método se puede comprobar en los ejemplos propuestos

Cuando se plantea un problema concreto, sería discutible si es más conveniente usar un gran número de reglas de orden 0 o 1, o bien un número menor de reglas de orden alto. Desde el punto de vista computacional, la diferencia para una tolerancia en el error dada no está clara. Desde el punto de vista del número de reglas difusas generadas, ya se ha comentado que al usar reglas de orden mayor, es de esperar una reducción drástica en el número de reglas necesarias para obtener la aproximación. Finalmente desde el punto de vista de la interpretabilidad de los modelos locales, tradicionalmente las reglas de orden cero han sido preferentemente escogidas debido a la facilidad de comprensión por parte del diseñador. Sin embargo al aumentar el número de reglas del sistema, la interpretabilidad del modelo completo también se pierde (problema de maldición de la dimensionalidad). Las reglas de orden mayor son vistas normalmente como no interpretables [Yen et al., 1998], sin embargo como se expone en la siguiente sección, bajo ciertas condiciones es posible dotar a este tipo de reglas difusas de interpretabilidad, así pues dotando a estos sistemas de ambas propiedades: bajo número de reglas y modelos locales interpretables.

4.3. Interpretabilidad y Obtención de Modelos Locales Interpretables

Hasta ahora se ha visto como se pueden utilizar consecuentes polinomiales de orden alto en las reglas de un sistema difuso tipo TSK. Esta sección analiza las propiedades adicionales de interpretabilidad que se pueden añadir al modelo difuso, mediante la utilización de un tipo especial de función de pertenencia.

4.3.1. Breve introducción al desarrollo en serie de Taylor

Sea $f(x)$ una función definida en un intervalo, con un punto intermedio a, para el que se conocen las derivadas de todos los órdenes. El polinomio de primer orden

$$p_1(x) = f(a) + f'(a)(x-a) \tag{4.8}$$

tiene el mismo valor que $f(x)$ en el punto $x = a$ y también el mismo valor en la primera derivada en ese punto. Su representación gráfica es una línea tangente al gráfico de $f(x)$ en el punto $x = a$.

Tomando la segunda derivada de $f(x)$ en $x = a$, el polinomio de segundo orden

$$p_2(x) = f(a) + f'(a)(x-a) + \frac{1}{2}f(a)(x-a)^2 \tag{4.9}$$

tiene el mismo valor que $f(x)$ en $x = a$, y la mismas derivadas primera y segunda en este punto. Para los puntos en la vecindad de $x = a$, $f(x)$ será más similar a $p_2(x)$ que a $p_1(x)$. Por tanto podemos esperar que formando un polinomio de orden n, $p_n(x)$, a partir de las n primeras derivadas de $f(x)$ en $x = a$, en la misma forma como hicimos para $p_1(x)$ y $p_2(x)$, el polinomio resultante será muy cercano a $f(x)$ en la vecindad de $x = a$. Ver la figura 4.1

El teorema de Taylor establece que si una función $f(x)$, definida en un intervalo tiene derivadas de todos los órdenes, se puede aproximar en torno a un punto $x = a$, como su expansión en serie de Taylor en torno a ese punto

$$\begin{aligned} f(x) = f(a) + f'(a)(x-a) + \tfrac{1}{2}f''(a)(x-a)^2 ... \\ + \tfrac{1}{n!}f^{(n)}(a)(x-a)^n + \tfrac{1}{(n+1)!}f^{(n+1)}(c)(x-a)^{n+1} \end{aligned} \tag{4.10}$$

donde el último término se refiere al error, en el que c es un punto entre x y a.

La expansión en serie de Taylor abre una puerta a la aproximación de cualquier función mediante polinomios, esto es, mediante la adición de funciones sencillas. Es por tanto una clave fundamental en el campo de la Teoría de la Aproximación Funcional y el Análisis Matemático. La expansión en serie de Taylor también nos proporcionará una forma de dar interpretabilidad a los sistemas difusos tipo TSK tomando un cierto tipo de antecedentes de las reglas que tienen un número de propiedades interesantes.

En el caso n dimensional, la expansión en serie de Taylor se adapta a la siguiente forma

$$\begin{aligned} f(\vec{x}) = f(\vec{a}) + (\vec{x}-\vec{a})^T \left[\tfrac{\partial f}{\partial \vec{x}_i}(\vec{a})\right]_{i=1..n} + \tfrac{1}{2}(\vec{x}-\vec{a})^T W(\vec{x}-\vec{a}) + \\ + \tfrac{1}{3!}\left\langle W^3(\vec{x}-\vec{a}\otimes\vec{x}-\vec{a}\otimes\vec{x}-\vec{a})\right\rangle + ... \end{aligned} \tag{4.11}$$

donde W es una matriz triangular de dimensiones $n \times n$ con valores

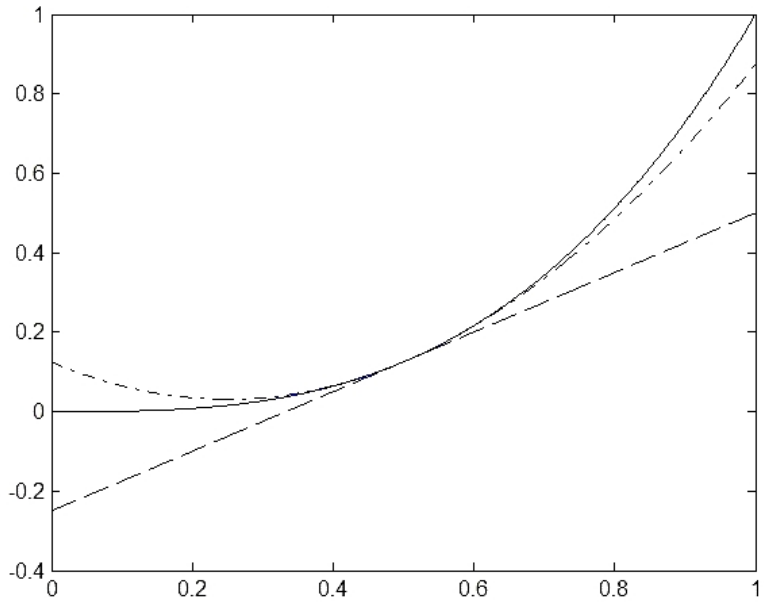


Figura 4.1. *Ejemplo de expansión en serie de Taylor. La función original (línea continua) es* $f(X) = x^3$*. Obsérvese como el polinomio* $p_1(x)$ *para* $a = 0{,}5$ *es tangente a* $f(x)$ *en ese punto (línea discontinua).* $p_2(x)$ *(línea discontinua y punteada) realiza una muy buena aproximación para los puntos en la vecindad de* $a = 0{,}5$*. Nótese que puesto que* $f(x)$ *es un polinomio de tercer orden,* $p_3(x)$ *sería exactamente igual que* $f(x)$*.*

$$\left[\frac{\partial f}{\partial \vec{x}_i \partial \vec{x}_j}(\vec{a})\right]_{i=1..n;\ j=i..n} \tag{4.12}$$

y W^3 es una matriz tridimensional conteniendo los correspondientes valores de la derivada multi-parcial para $\vec{x} = \vec{a}$.

Sería estimable el poder interpretar los consecuentes de las reglas como la expansión en serie de Taylor (truncada) para la función aproximada en el centro de cada regla. Antes de continuar, estudiaremos qué requerimientos deben cumplir las funciones de pertenencia con el objetivo de poder dar esta interpretación a la salida del sistema.

4.3.2. Funciones de pertenencia que preservan la interpretabilidad de las reglas difusas TSK de orden alto

Con el objetivo de dar interpretabilidad a los modelos locales como la expansión en serie de Taylor truncada en torno al centro de la regla, se requieren dos propiedades principalmente. Primero, el grado de solapamiento de todas las funciones de pertenencia debería anularse en el centro de cada regla, para que la salida del sistema en dicho punto esté solo influenciada por la regla respectiva. Y segundo, el tipo de función de pertenencia debe permitir que la salida del sistema sea continua y v veces derivable, donde v indica el

orden del consecuente polinomial de la regla difusa, con el objetivo de poder identificar los coeficientes de los consecuentes de las reglas como las derivadas parciales de orden v de la función de salida en los centros.

Las bases de funciones de pertenencia OLMF de orden p sobre un grid n-dimensional tienen una serie de propiedades interesantes como notó Marwan Bikdash [Bikdash, 1999]. Los requerimientos que un conjunto de funciones de pertenencia en un grid tienen que completar para formar una base OLMF son básicamente

- todas las funciones de pertenencia son locales (esto es, son no negativas y se anulan con la distancia), están definidas en un dominio delimitado y son del mismo tipo
- cada extremo de la función de pertenencia coincide con el centro de las funciones de pertenencia adyacentes (forman una partición, por tanto evitando un solapamiento incontrolado de las funciones de pertenencia, ver referencias [Setnes et al., 1998], [Paiva and Dourado, 2004])
- todas las funciones de pertenencia son p veces diferenciables y la p-ésima derivada de la misma es continua en todo su dominio
- la p-ésima derivada de la función de pertenencia se anula en su centro y en sus extremos
- las bases deben cumplir la propiedad de suma-a-1 [Ruspini, 1969]

Nótese que para los tipos más conocidos de funciones de pertenencia, estos requerimientos no se cumplen. Por ejemplo, para la partición triangular, las FP no son derivables en los centros de las reglas (tampoco ocurre esto para las particiones trapezoidales); para la partición Gaussiana, aunque el gradiente se anula en los centros, y las FPs son p veces derivables, las FP no tiene un intervalo delimitado (ver apéndice A. Las bases OLMF aseguran que la salida del sistema será continua y p veces derivable, que que las FP componentes son locales.

Para construir una función de pertenencia local que es consistente con esos requisitos, consideremos una FP de tipo "spline" definida por los tres siguientes parámetros $[a, b, c]$, donde a y c son los extremos de la FP local y b es el centro de la FP.

En el caso especial en el que $p = 2$, esto es, bases de orden dos, los requerimientos anteriores llevan a las siguientes condiciones sobre las FP

$$\begin{array}{ll} MF\,(\mathrm{a}) = 0; & MF\,(\mathrm{b}) = 1; \\ MF'\,(\mathrm{a}) = 0; & MF'\,(\mathrm{b}) = 0; \\ MF\ (\mathrm{a}) = 0; & MF\ (\mathrm{b}) = 0; \end{array} \tag{4.13}$$

Utilizando la Interpolación de Hermite, la función de pertenencia $FP(x, [a, b, c])$ para $x \in [a, b]$, esto es, la parte izquierda de la FP viene dada por

$$MF_i\,(x, [a, b, c]) = \frac{1}{(b-a)^3}\,(x-a)^3 \left[1 - \frac{3\,(x-b)}{(b-a)} + \frac{6\,(x-b)^2}{(b-a)^2}\right] \qquad for\, x\, \in [a, b] \tag{4.14}$$

Debido a la propiedad de suma-a-1, la parte derecha de la FP queda de la siguiente forma

$$MF_i\left(x,[a,b,c]\right) = 1 - MF_{i+1}\left(x,[b,c,d]\right) \qquad for\, x \in [b,c] \tag{4.15}$$

La figura 4.2 muestra un ejemplo gráfico de este tipo de FP. Puede observarse que aparte de la continuidad y derivabilidad de la función, el gradiente de la FP se anula en el centro y en los extremos del intervalo donde está definido.

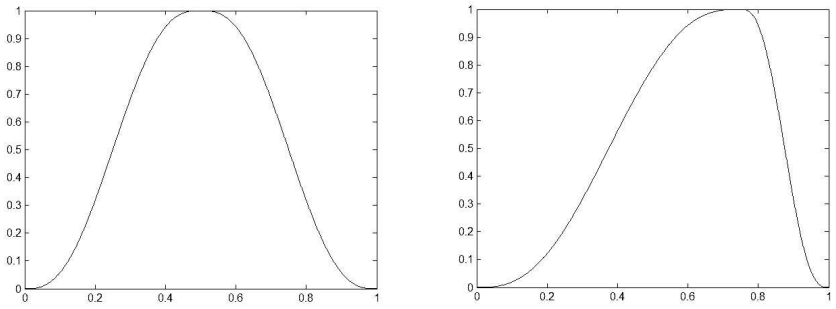


Figura 4.2. *Dos ejemplos de función de pertenencia para el tipo de función de pertenencia OLMF.*

Ahora, dado cualquier punto de entrada, la fórmula final para la salida del sistema se simplifica de la siguiente forma

$$F(\vec{x}) = \sum_{k=1}^{K} \mu_k\left(\vec{x}\right) Y_k\left(\vec{x}\right) \tag{4.16}$$

donde el denominador de la ecuación anterior 4.3 pasa a ser 1 debido a la propiedad de la suma-a-1, y donde debe recordarse que las reglas tienen la forma dada en 4.1, con salida polinomial en la forma 4.11.

4.3.3. Interpretabilidad de las reglas difusas

Marwan Bikdash demostró en [Bikdash, 1999] que dado un sistema difuso TSK basado en reglas en el que

1. las funciones de pertenencia en las variables de entrada forma una base OLMF de orden p en cada dimensión de entrada

2. la parte del consecuente de cada regla está escrito en la forma centrada mostrada en 4.1 y 4.14, siendo $Y^k\left(\vec{x}\right)$ polinomios de grado n

entonces para $n \leq p$, cada $Y^k\left(\vec{x}\right)$ puede ser interpretado como la expansión en serie de taylor truncada de orden n sobre la salida del sistema difuso en torno al punto $x = a$, centro de la k-ésima regla.

Considérese entonces que disponemos de un método para obtener los coeficientes óptimos de los consecuentes de las reglas TSK 4.11 para aproximación funcional dado un conjunto de ejemplos de entrada/salida, y una distribución de funciones de pertenencia

que forman una base OLMF; entonces podemos interpretar los consecuentes de las reglas $Y^k(\vec{x})$ como la expansión en serie de Taylor truncada en torno a los centros de las reglas de la salida del sistema; y además, de la función modelada subyacente dada por el conjunto de ejemplos de entrada/salida. En el caso límite en el que la función está perfectamente aproximada por el sistema, los consecuentes de las reglas coincidirán con la expansión en serie de Taylor de dicha función en torno al centro de cada regla, habiéndose obtenido una total aproximación de la función, pero también una total interpretación de dicha función a aproximar, tanto local como global. El sistema difuso TaSe que consta del tipo de antecedentes de reglas expuesto en la subsección 4.3.2, y con el tipo de consecuentes de las reglas expuesto en la subsección 4.3.1, proporciona un modelo general altamente interpretable, y como se verá en las siguientes secciones, un modelo preciso y eficiente para problemas de aproximación funcional a partir de un conjunto de entrenamiento de entrada/salida.

En el trabajo [Bikdash, 1999], Bikdash utilizó directamente los valores (disponibles) de la expansión en serie de Taylor en torno a los centros de las reglas para aproximar la función mediante el sistema difuso TSK. Nótese que estos consecuentes de las reglas, aunque tienen una interpretabilidad importante como modelos locales, no son los consecuentes óptimos desde el punto de vista del enfoque de mínimos cuadrados; nótese que la expansión en serie de Taylor es una aproximación para una función en la vecindad del punto referencia. Por tanto, incluso aún cuando se usa un gran número de FPs, el error obtenido por el método en [Bikdash, 1999] es difícilmente lo suficientemente pequeño (comparando con un sistema con complejidad similar cuyos coeficientes en los consecuentes estén optimizados mediante LSE) y por tanto, la salida del sistema es una vaga aproximación de los datos que se modelan.

Considérese también que para aproximación funcional, normalmente disponemos exclusivamente de la información dada por los puntos de entrada/salida. No hay dada información adicional sobre la función a aproximar, ni tampoco sobre las derivadas parciales de la función con respecto a los puntos. Además no hay método preciso para obtener las derivadas a partir de los puntos de entrenamiento.

Finalmente, el método que se presenta en este capítulo además trata la búsqueda de los puntos óptimos en los que establecer los centros de las reglas con el objetivo de encontrar tanto la interpretabilidad como la aproximación funcional como se verá en la siguiente sección.

4.4. Ajuste de Parámetros en el Modelo TaSe

Numerosos trabajos en la literatura han tratado el problema del ajuste de parámetros para aproximación funcional utilizando sistemas difusos basados en grid [Wang, 1994], [Pomares et al., 2000], [Kim and Kim, 1997], [Ning et al., 2003], [Wu and Yu, 2000]. Algunos de éstos autores centran su investigación en el uso de algoritmos genéticos debido a su alta capacidad de exploración en el espacio de búsqueda. Sin embargo, el coste computacional que este tipo de métodos requiere es a veces excesivo. En nuestro caso, como se mencionó anteriormente en la sección 4.2, los coeficientes de los consecuentes de las reglas se obtienen de forma óptima utilizando el enfoque de mínimos cuadrados, y SVD para resolver el sistema de ecuaciones lineales resultante. Con respecto a los centros de las FP, el procedimiento de optimización de Levenberg-Marquardt [More, 1978] se emplea como método de búsqueda local. Utiliza una configuración inicial de los centros que se ha seleccionado previamente mediante otra técnica (Equidistribución de Errores), con el objetivo de minimizar los riesgos de la búsqueda local.

Dicha configuración inicial de los centros de las FPs para el algoritmo Levenberg-

Marquardt se haya utilizando la técnica propuesta por H. Pomares en [Pomares et al., 2000] y que se resume a continuación. El uso de este método, denominado "Método de Equidistribución de Errores" (EE), fue satisfactoriamente empleado en [Pomares et al., 2000] y [Pomares et al., 2002] para una configuraciónque utilizaba partición triangular y consecuentes constantes. Como se comprobará en las secciones de simulaciones, este enfoque funciona satisfactoriamente también para el modelo TaSe con bases OLMF y consecuentes de las reglas basadas en la expansión en serie de Taylor.

4.4.1. Método de Equidistribución de Errores para la inicialización de los centros

El objetivo esta técnica es buscar la distribución de centros que distribuya el error homogéneamente en todas las regiones cubiertas por el grid de FP. Esta idea puede trasladarse matemáticamente a tener a cada lado del centro de cada FP, la misma cantidad de error, según la función de error dada en 4.6 y el conjunto de entrenamiento D. Esto es

$$\sum_{\substack{m\in D \\ x_n^m\in\left[c_n^{i_n-1},c_n^{i_n}\right]}} e^2(\vec{x}^m) \approx \sum_{\substack{m\in D \\ x_n^m\in\left[c_n^{i_n},c_n^{i_n+1}\right]}} e^2(\vec{x}^m) \tag{4.17}$$

para el i_n-ésimo centro definido en la variable x_n, $c_n^{i_n}$.

Este método consiste en un proceso iterativo con dos fases: una para calcular un parámetro de pendiente para cada centro, y un segundo para mover los centros de acuerdo a dicho parámetro de pendiente.

En la primera fase, el centro $c_n^{i_n}$ es asociado con un valor de pendiente $p_n^{i_n}$

$$p_n^{i_n} = \frac{1}{\sigma_y^2} \left(\sum_{\substack{m\in D \\ x_n^m\in\left[c_n^{i_n-1},c_n^{i_n}\right]}} \left(y^m - F\left(\vec{x}^m\right)\right)^2 \; - \sum_{\substack{m\in D \\ x_n^m\in\left[c_n^{i_n},c_n^{i_n+1}\right]}} \left(y^m - F\left(\vec{x}^m\right)\right)^2 \right) \tag{4.18}$$

siendo σ_y^2 la desviación típica de los datos de salida, usada aquí como una constante de normalización. Un valor positivo del parámetro indica que la contribución al error de la parte izquierda es mayor que la contribución a la parte derecha; por tanto, desearíamos que el centro de la FP se moviese a la parte izquierda para contrarrestar este efecto, y viceversa.

Una vez que el parámetro de pendiente se ha calculado para todos los centros de las FPs del sistema, en la segunda fase del proceso, se realiza el siguiente movimiento de centros

$$\Delta c_n^{i_n} = \begin{cases} \frac{c_n^{i_n-1}-c_n^{i_n}}{b}\frac{p_n^{i_n}}{p_n^{i_n}+\frac{1}{T_n^{i_n}}} & if \quad p_n^{i_n} \geq 0 \\ \frac{c_n^{i_n+1}-c_n^{i_n}}{b}\frac{\left|p_n^{i_n}\right|}{\left|p_n^{i_n}\right|+\frac{1}{T_n^{i_n}}} & if \quad p_n^{i_n} < 0 \end{cases} \tag{4.19}$$

Aquí, b es el radio activo, que es la mayor distancia de variación y se usa para garantizar que el orden de las funciones de pertenencia permanece sin cambios; $T_n^{i_n}$ es la temperatura que controla cuan lejos la variable se moverá en cada iteración y que se decrementará conforme el sistema encuentra su equilibrio.

El procedimiento descrito trabaja iterativamente, moviendo los centros hasta que el error a cada lado de los centros de las FPs queda balanceado. En este trabajo, usamos $b = 2$ y $T_n^{i_n} = 100$

4.4.2. Calculando la distribución óptima de los centros mediante el algoritmo de Levenberg - Marquardt

En el paso previo, se ha alcanzado una configuración de centros que se asume es una buena solución, pero que se refina en un proceso de búsqueda local basada en gradiente para obtener una configuración óptima. Para llevar a cabo la tarea, hay disponibles numerosos métodos en la bibliografía (descenso en gradiente, gradiente conjugado, método de Newton-Raphson, método Levenberg-Marquardt, etc.). En este trabajo se ha usado el método Levenberg-Marquardt puesto que sus características de eficiencia y robustez lo hacen especialmente adecuado para este tipo de problemas de optimización. Las expresiones explícitas para las derivadas parciales con respecto a cada parámetro implicado en el sistema difuso pueden encontrarse en el apéndice B.

4.5. Identificación de la Estructura en el Modelo TaSe

Es más escaso el trabajo presente en la literatura sobre identificación de la estructura en sistemas difusos basados en grid. Los algoritmos genéticos han sido también el mayor foco de investigación para este problema, habiéndose propuesto pocos algoritmos automáticos [Ning et al., 2003], [Pomares et al., 2002], [Sugeno and Kang, 1988].

Suponemos que disponemos de un conjunto de entrenamiento y de un conjunto de validación para el algoritmo de identificación de la estructura. El punto de partida para el método propuesto es la estructura más simple posible, que consta de una sola FP por variable. Por tanto se comienza con una sola regla en la forma dada en la ecuación 4.1, con consecuente en la forma dada en 4.11. Como se estableció antes, utilizando mínimos cuadrados y métodos de regresión lineal, podemos obtener los coeficientes óptimos de los consecuentes de dicha regla para aproximar el conjunto de entrenamiento.

Partiendo de esta configuración sencilla, el procedimiento propuesto añadirá funciones de pertenencia a las variables de entrada mientras que el error de validación no se incremente. Se añadirá iterativamente una función de pertenencia a una variable; para añadir una FP en cada paso se seleccionará la variable en la que el error se decremente más. Así, el objetivo es alcanzar una configuración final con un error de validación mínimo. Dado este enfoque, la cuestión es cómo se identifica la variable en la que el error se decremente más, que será en la que se añada una nueva FP. Primero nótese que añadiendo una FP en cualquier variable j en el sistema TaSe implica

- Un incremento considerable del poder aproximativo del sistema, puesto que implica la adición de $\prod_{i=1, i\neq j}^{n} mf_i$ reglas

- Debido a este poder aproximativo y a la necesidad de evitar redundancia, el ajuste de parámetros debe tenerse en cuenta con el objetivo de tomar la decisión correcta. Si no se posicionan correctamente los centros de las FP cuando se toma la decisión, puede añadirse FPs y reglas innecesarias, incrementando por tanto innecesariamente la complejidad del sistema, de acuerdo a las ecuaciones 3.4 y 4.7.

- Aparte, la adición de una sola regla de por ejemplo segundo orden al sistema, implica la adición de

$$\left(n^2 + n\right)/2 + n + 1 \tag{4.20}$$

parámetros lineales por parte del consecuente de la regla, donde n es el número de variables de entrada.

Nótese además que debido al alto poder aproximativo de cada una de las reglas del sistema TaSe, es de esperar que solo se necesiten pocas FPs por cada variable de entrada.

Estas razones fuerzan a que la selección del algoritmo de identificación de arquitectura sea cuidadosa. Los algoritmos genéticos por ejemplo, son inviables debido a su poca eficiencia cuando tratan este tipo de problemas. Se propone por tanto un enfoque greedy que asegura encontrar una estructura sub-óptima mínima de forma automática. Sencillamente se comprobará cada posible nueva configuración, añadiendo una FP en cada variable de entrada, y se conservará la configuración en la que el error de validación sea menor. Esta técnica es posible puesto que hemos proporcionado un método automático eficiente para el ajuste de parámetros que obtiene los coeficientes óptimos de los consecuentes y un ajuste sub-óptimo de los centros de las FPs. El algoritmo final queda de la siguiente forma

1: **while** validation error decreases (or minimum validation error limit not reached) **do**
2: **for** I = 1..number of input variables **do**
3: Consider adding 1 MF to variable I
4: Optimise centres and consequents
5: Get the error
6: **end for**
7: Add 1 MF to the variable where adding 1 MF resulted in a lower validation error.
8: **end while**

Este método sencillo asegura que se encontrará una configuración de FPs en las variables de entrada sencilla, sub-óptima y que proporciona el menor error de validación esperado dados los conjuntos de entrenamiento, para el ajuste de parámetros, y validación para la selección de la mejor estructura.

4.6. Ejemplos

Esta sección proporciona tres ejemplos de aplicación, que han sido utilizados previamente en la literatura, para clarificar las principales características de la metodología TaSe propuesta. Como es común en trabajos que tratan el problema de aproximación funcional, se utilizarán ejemplos de funciones analíticas, puesto que éstas permiten que la función estimada sea comparada con la original para cualquier punto deseado. En todos los casos, el algoritmo comienza asignando una función de pertenencia por variable, habiendo por tanto una sola regla en el sistema difuso, como se comentó en la sección anterior.

La medida del error utilizada en los dos primeros ejemplos es la raiz del error cuadrático medio normalizado (Normalized Root-Mean-Square Error, NRMSE) que se define como

$$NRMSE = \sqrt{\frac{\overline{e^2}}{\overline{\sigma_y^2}}} \tag{4.21}$$

donde σ_y^2 es la varianza de los datos de salida, y $\overline{e^2}$ es el error cuadrático medio entre la salida del sistema y las salidas del conjunto de datos D. De esta forma, el índice NRMSE

describe el rendimiento de la aproximación, haciéndolo independiente de factores de escala o del número de datos. Para el tercer ejemplo (la serie Mackey-Glass), se utilizará la raiz del error cuadrático medio (RMSE) puesto que es la medida normalmente utilizada en la literatura para este benchmark específico.

4.6.1. Aplicación detallada del algoritmo a una función 1-dimensional

Para mostrar la metodología propuesta, primeramente consideraremos un ejemplo representativo desde el punto de vista del problema de aproximación funcional y de la pérdida de interpretabilidad en los sistemas difusos TSK. Vamos a aplicar el algoritmo completo a un ejemplo de función unidimensional, presente en [Nie and Lee, 1996] y estudiado también en [Pomares et al., 2000]. Como se comprobará en los resultados obtenidos que el modelo TaSe presenta excelentes resultados de aproximación funciona, además de proporcionar interpretabilidad a los modelos locales gracias al las reglas basadas en el desarrollo de Taylor y a la estructura en los antecedentes del sistema TaSe.

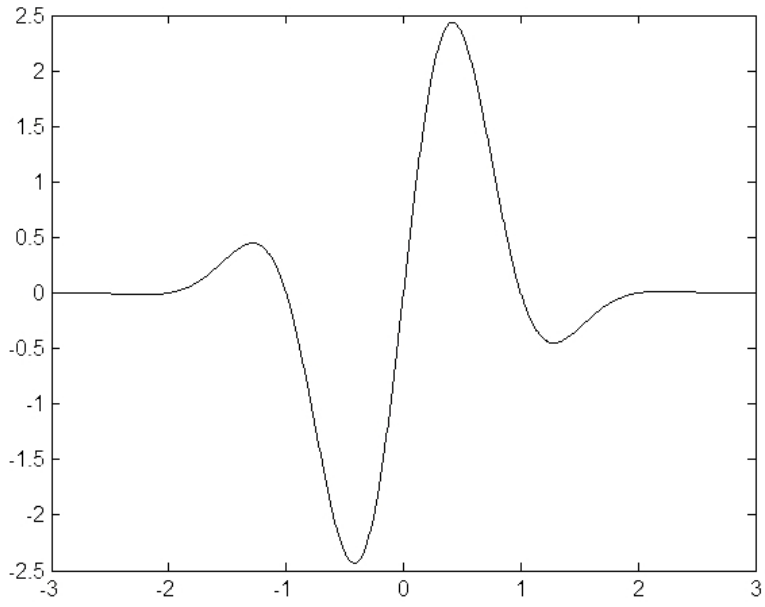


Figura 4.3. *Función original sin ruido a aproximar.*

La función presentada por J.H. Nie, (ver figura 4.3) se define por la expresión siguiente

$$y(x) = 3e^{-x^2} sin(\pi x) \ + \xi \quad donde \ x \in [-3, 3] \ \ y \ \ \xi \ \ ruido \tag{4.22}$$

Puesto que es una función unidimensional, el algoritmo de identificación de estructura trabaja de manera trivial, empezando por una FP en la variable, y añadiendo nuevas

FPs hasta que el error de validación deja de disminuir. En cada etapa del algoritmo, los centros se calculan usando el procedimiento expuesto en la sección 4.4. Mínimos cuadrados trabajará en cada iteración para el cálculo de los coeficientes de los consecuentes óptimos para la distribución de centros de FPs dada.

Para demostrar la ejecución del algoritmo, tomaremos una ejecución ejemplo que tiene 100 puntos de entrenamiento aleatoriamente distribuidos con ruido aditivo Gaussiano 0.1, y 100 puntos de validación de las mismas características. Ahora veremos en detalle como evoluciona el algoritmo. Primero, tomando solo una FP, la única regla del sistema, centrada en $x = 0$ (denotaremos la única función de pertenencia centrada en 0 como $\widehat{0}$ queda

$$\text{SI } x \text{ es } \widehat{0} \text{ ENTONCES } y = -0{,}0(x-0)^2 + 0{,}037(x-0) + 0{,}11 \tag{4.23}$$

con error de validación NRMSE = 0.998.

Añadiendo una FP más al sistema, resulta en un sistema con las dos siguientes reglas, centradas en los extremos del intervalo de definición

$$\begin{array}{l} \text{SI } x \text{ es } -\widehat{3} \text{ ENTONCES } y = 4{,}77(x+3)^2 + 0{,}63(x+3) - 0{,}51 \\ \text{SI } x \text{ es } \widehat{3} \text{ ENTONCES } y = -4{,}78(x-3)^2 + 0{,}59(x-3) + 0{,}47 \end{array} \tag{4.24}$$

con NRMSE = 0.846.

Una vez que una tercera FP es añadida, el ajuste de parámetros expuesto en la sección 4.4 se ejecuta para optimizar la posición de la FP central. Puesto que los extremos de cada FP coinciden con los centros de las FPs adyacentes en cada variable de entrada (ver sección 4.3.2), solo los centros de las FPs tienen que optimizarse, descartándose por tanto los dos centros situados en los extremos del dominio $[-3, 3]$. El valor inicial del centro móvil será 0, puesto que las FPs están inicialmente equidistribuidas en el intervalo $[-3, 3]$. Después de la ejecución del método de Equidistribución de errores, el valor para el centro de la variables es 1.029; y con este nuevo valor inicial de la configuración, el algoritmo de Levenberg-Marquardt alcanza el valor sub-óptimo para la variable centro igual a 1.066, y el sistema de regla finalmente queda como

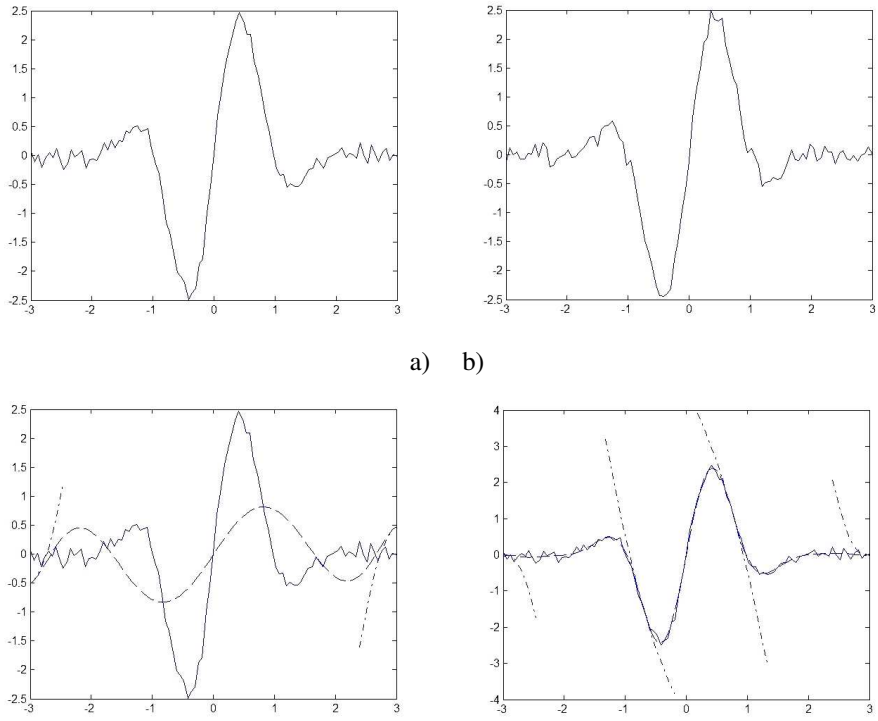
$$\begin{array}{l} \text{SI } x \text{ es } -\widehat{3} \text{ ENTONCES } y = -18{,}7(x+3)^2 + 0{,}81(x+3) + 0{,}35 \\ \text{SI } x \text{ es } \widehat{1{,}07} \text{ ENTONCES } y = 16{,}1(x-1{,}07)^2 - 3{,}41(x-1{,}07) - 0{,}08 \\ \text{SI } x \text{ es } \widehat{3} \text{ ENTONCES } y = -8{,}41(x-3)^2 + 4{,}09(x-3) + 0{,}50 \end{array} \tag{4.25}$$

Resultando en un modelo con un NRMSE de validación de 0.434

Añadiendo una FP más implica que hay que optimizar la posición de dos centros de FP. Ambos empiezan con valores equidistribuidos en el intervalo $[-3, 3]$, por tanto -1 y 1. El método de equidistribución de errores encuentra la configuración inicial idónea para los centros de valores $-0{,}71$ y $0{,}71$. El error obtenido con esta configuración es NRMSE = 0.096

$$\begin{array}{l} \text{SI } x \text{ es } -\widehat{3} \text{ ENTONCES } y = -7{,}60(x+3)^2 + 0{,}93(x+3) - 0{,}04 \\ \text{SI } x \text{ es } -\widehat{0{,}71} \text{ ENTONCES } y = 2{,}62(x+0{,}71)^2 - 6{,}03(x+0{,}71) - 1{,}37 \\ \text{SI } x \text{ es } \widehat{0{,}71} \text{ ENTONCES } y = -2{,}24(x-0{,}71)^2 - 5{,}96(x-0{,}71) + 1{,}30 \\ \text{SI } x \text{ es } \widehat{3} \text{ ENTONCES } y = 7{,}22(x-3)^2 + 0{,}91(x-3) + 0{,}01 \end{array} \tag{4.26}$$

La figura 4.4 muestra los conjuntos de entrenamiento (a) y validación (b) utilizados para el presente ejemplo. Además, en (c) se muestra el estado del modelo tras el análisis del sistema de dos reglas. La figura (c) contiene el conjunto de datos original, la salida obtenida con el sistema de dos reglas y cómo ambos consecuentes polinómicos de las reglas se asemejan a la salida del sistema en torno a ambos centros. El apartado (d) muestra la salida final del modelo TaSe junto con el conjunto de datos original. El modelo final deja filtrado completamente el ruido añadido. Nótese además que para las cuatro reglas centradas en $[-3, -0{,}71, 0{,}71, 3]$, la representación de los consecuentes polinomiales es prácticamente idéntica a la salida del modelo en la vecindad de los centros de las reglas. De acuerdo con el ejemplo presentado, el algoritmo automático presentado en este trabajo obtiene un modelo que presenta precisión en la aproximación y cuyos modelos locales son interpretables gracias al concepto del desarrollo en serie de Taylor.



a) b)

c) d)

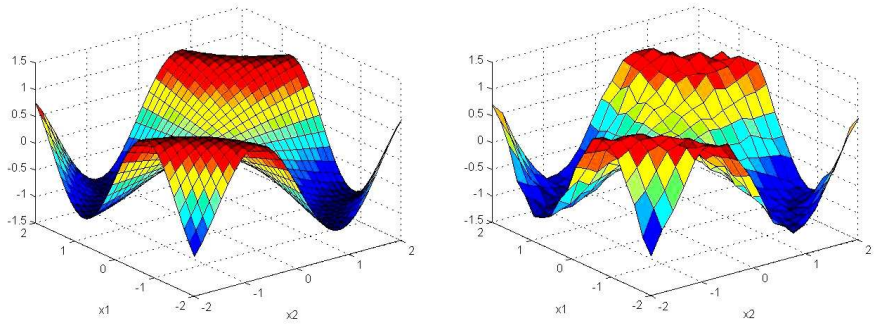
Figura 4.4. *a) Conjunto de entrenamiento con 100 datos de entrada para la función $y(x)$ con ruido aditivo Gaussiano $N \sim (0, 0{,}1)$.b) Conjunto de validación con 100 datos para la función $y(x)$ con ruido aditivo Gaussiano $N \sim (0, 0{,}1)$. c) Aproximación funcional utilizando dos reglas (línea discontinua). Los consecuentes polinomiales de ambas reglas también se muestran (líneas discontinuas-punteadas). d) Aproximación funcional con cuatro reglas. Obsérvese como con cuatro reglas, la función objetivo es aproximada (línea discontinua). Obsérvese también como la gráfica de los consecuentes polinomiales es similar a la salida de la función en los puntos cercanos a los centros de las FPs = $[-3, -0{,}71, 0{,}71, 3]$ (líneas discontinuas-punteadas).*

Añadiendo una FP más lleva a un incremento del error de validación. Por tanto el modelo óptimo obtenido es el anterior con 4 reglas. El error medio obtenido para 10 ejecuciones diferentes es igual a 0.105 con desviación típica 0.012. Nótese además el bajo número de reglas utilizadas para aproximar la función, definida por los conjuntos de entrenamiento y validación. Realizando una breve comparación con otros modelos difusos tipo TSK, el número de reglas necesarias para aproximar esta función utilizando consecuentes constantes es mucho mayor como se nota en [Pomares et al., 2000].

4.6.2. Aplicación a una función bidimensional

Considérese ahora una función bidimensional f_1 tomada de [Cherkassky et al., 1996]. Se muestran ahora los resultados de la ejecución completa del algoritmo de aprendizaje, incluyendo el sub-algoritmo de identificación de estructura. La expresión de la función f_1 (ver figura 4.5) es

$$f_1(x_1, x_2) = \sin(x_1 \cdot x_2) + \xi \qquad donde \quad \begin{matrix} x_1,\ x_2\ uniforme\ en\ [-2,2] \\ \xi\ ruido \end{matrix} \tag{4.27}$$



a) b)

Figura 4.5. *a) Función f_1 con 1000 muestras equidistribuidas. b) Función f_1 con ruido Gaussiano $N \sim (0, 0{,}05)$ y 400 muestras aleatoriamente distribuidas.*

Como se citó para el ejemplo anterior, el algoritmo comienza con la configuración más sencilla (1 FP por variable de entrada = 1 sola regla), y explora configuraciones más complejas, obteniendo en cada paso los parámetros óptimos de la configuración. La decisión sobre qué variable debe incrementar su número de FPs se discutió en la sección 4.5. En el ejemplo que se considera ahora (ver ecuación 4.27, es claro que ambas variables de entrada influencian la salida de forma similar. Así cualquier algoritmo de identificación de estructura, debe proporcionar a ambas variables un número similar de funciones de pertenencia, de forma que se obtenga un resultado de aproximación lo más óptimo posible, manteniendo lo más baja posible la complejidad del sistema.

Para esta función se extraen 400 puntos para entrenamiento y 400 para validación, ambos distribuidos aleatoriamente y distorsionados con ruido aditivo Gaussiano $N \sim (0, 0{,}05)$ en la salida. En el inicio del algoritmo, con 1 FP por variable, la única regla del sistema, centrada en $x_1 = 0, x_2 = 0$, es

$$\begin{array}{c}\text{SI } x_1 \text{ es } -\widehat{0} \text{ Y } x_2 \text{ es } \widehat{0} \text{ ENTONCES } y = -0{,}0(x_1-0)^2 + \\ 0{,}27(x_1-0)(x_2-0) - 0{,}0(x_2-0)^2 + 0{,}54(x_1-0) + 0{,}54(x_2-0) + 1{,}09\end{array} \tag{4.28}$$

con un error NRMSE = 0.774. Ahora se debe tomar una decisión de en qué variable se debe añadir una nueva FP de forma que el error sea menor. Como se expuso en la sección 4.5, se chequearán las dos alternativas posibles, es decir, añadir una FP en la primera variable y añadir una FP en la segunda variable. La alternativa que proporcione el menor error se escogerá, de forma que dicha FP queda añadida permanentemente para las próximas iteraciones del algoritmo de identificación de la estructura. La configuración de funciones de pertenencia 2×1 proporciona un error de NRMSE = 0.562, y la configuración 1×2 da un error de NRMSE = 0.563. Nótese la igualdad en rendimiento para ambas posibles estructuras. Como se mencionó anteriormente, la función f_1 que se está considerando es simétrica, y por tanto esta situación es de esperar. Según los errores obtenidos por el chequeo de ambas estructuras, se añade una FP a la primera variable. Las dos reglas, centradas en $x_1 = -2, x_2 = 0$, y $x_1 = 2, x_2 = 0$ en el nuevo sistema eventualmente son

$$\begin{array}{l}\text{SI } x_1 \text{ es } -\widehat{2} \text{ Y } x_2 \text{ es } \widehat{0} \text{ ENTONCES } y = 0{,}05(x_1+2)^2 - \\ 0{,}97(x_1+2)(x_2-0) - 0{,}0(x_2-0)^2 - 1{,}93(x_1+2) - 0{,}07(x_2-0) - 0{,}14 \\ \text{SI } x_1 \text{ es } \widehat{2} \text{ Y } x_2 \text{ es } \widehat{0} \text{ ENTONCES } y = -0{,}04(x_1-2)^2 - \\ 0{,}95(x_1-2)(x_2-0) + 0{,}0(x_2-0)^2 - 1{,}94(x_1-2) + 0{,}05(x_2-0) + 0{,}11\end{array} \tag{4.29}$$

En el siguiente paso, el sub-algoritmo de identificación de estructura chequeará el rendimiento de las dos posibles alternativas, esto es, una estructura de funciones de pertenencia 3×1 y otra estructura de FP 2×2. La primera alternativa da un error NRMSE = 0.561, y la segunda da un error NRMSE = 0.113. Véase la gran diferencia entre añadir una FP en una u otra variable de entrada. Esto es debido de nuevo a la simetría de la función f_1 que se está considerando, que debe llevar a una distribución similar de FPs a ambas variables de entrada, siempre que el error de validación decrezca. La ejecución completa del algoritmo de identificación de estructura se puede contemplar en la tabla 4.1. Se muestra también para cada paso del algoritmo, el error proporcionado por la configuración escogida, así como el error de validación obtenido para cada una de las dos posibles alternativas que pueden ocurrir en cada paso, de forma que el lector pueda fácilmente comprobar las decisiones tomadas por el algoritmo.

La estructura óptima encontrada para el modelo tiene 3 FPs en cada variable de entrada. El NRMSE obtenido es de 0.0820, y las nueve reglas del sistema tras la ejecución del algoritmo son

#MFs		NRMSE (5 % noise)		Añadiendo 1 FP en la variable X1	Añadiendo 1 FP en la variable X2
X1	X2	Training	Validation		
1	1	0.776	0.772	0.562	0.563
2	1	0.560	0.562	0.561	0.113
2	2	0.108	0.113	0.0954	0.0975
2	3	0.0933	0.0954	0.0844	0.0820
3	3	0.0731	0.0820	0.0830	0.0844
3	4	El error aumenta al añadir 1 FP a cualquier variable			

Tabla 4.1. *Datos de la ejecución del algoritmo de identificación de la estructura para el ejemplo f_1*

$$
\begin{array}{l}
\text{SI } x_1 \text{ es } -\widehat{2} \text{ Y } x_2 \text{ es } -\widehat{2} \text{ ENTONCES } y = -0{,}24(x_1+2)^2 - \\
0{,}34(x_1+2)(x_2+2) - 1{,}08(x_2+2)^2 + 2{,}12(x_1+2) + \\
1{,}81(x_2+2) - 0{,}83 \\
\text{SI } x_1 \text{ es } -\widehat{2} \text{ Y } x_2 \text{ es } -\widehat{0{,}06} \text{ ENTONCES } y = -0{,}24(x_1+2)^2 + \\
1{,}12(x_1+2)(x_2+0{,}06) - 0{,}05(x_2+0{,}06)^2 + 0{,}05(x_1+2) - \\
1{,}97(x_2+0{,}06) - 0{,}10 \\
\text{SI } x_1 \text{ es } -\widehat{2} \text{ Y } x_2 \text{ es } \widehat{2} \text{ ENTONCES } y = 0{,}74(x_1+2)^2 - \\
0{,}23(x_1+2)(x_2-2) + 1{,}05(x_2-2)^2 - 1{,}98(x_1+2) + \\
1{,}77(x_2-2) + 0{,}82 \\
\text{SI } x_1 \text{ es } \widehat{0{,}13} \text{ Y } x_2 \text{ es } -\widehat{2} \text{ ENTONCES } y = -1{,}05(x_1-0{,}13)^2 + \\
0{,}92(x_1-0{,}13)(x_2+2) + 0{,}14(x_2+2)^2 - 2{,}05(x_1-0{,}13) - \\
0{,}03(x_2+2) - 0{,}22 \\
\text{SI } x_1 \text{ es } \widehat{0{,}13} \text{ Y } x_2 \text{ es } -\widehat{0{,}06} \text{ ENTONCES } y = 0{,}10(x_1-0{,}13)^2 + \\
1{,}10(x_1-0{,}13)(x_2+0{,}06) + 0{,}11(x_2+0{,}06)^2 - 0{,}12(x_1-0{,}13) + \\
0{,}18(x_2+0{,}06) - 0{,}01 \\
\text{SI } x_1 \text{ es } \widehat{0{,}13} \text{ Y } x_2 \text{ es } \widehat{2} \text{ ENTONCES } y = 0{,}46(x_1-0{,}13)^2 + \\
1{,}01(x_1-0{,}13)(x_2-2) - 0{,}29(x_2-2)^2 + 2{,}10(x_1-0{,}13) + \\
0{,}02(x_2-2) + 0{,}26 \\
\text{SI } x_1 \text{ es } \widehat{2} \text{ Y } x_2 \text{ es } -\widehat{2} \text{ ENTONCES } y = 2{,}64(x_1-2)^2 - \\
0{,}03(x_1-2)(x_2+2) + 1{,}94(x_2+2)^2 + 1{,}88(x_1-2) - \\
1{,}82(x_2+2) + 0{,}79 \\
\text{SI } x_1 \text{ es } \widehat{2} \text{ Y } x_2 \text{ es } -\widehat{0{,}06} \text{ ENTONCES } y = 0{,}19(x_1-2)^2 + \\
0{,}99(x_1-2)(x_2+0{,}06) - 0{,}63(x_2+0{,}06)^2 + 0{,}15(x_1-2) + \\
2{,}04(x_2+0{,}06) - 0{,}09 \\
\text{SI } x_1 \text{ es } \widehat{2} \text{ Y } x_2 \text{ es } \widehat{2} \text{ ENTONCES } y = -2{,}22(x_1-2)^2 - \\
0{,}08(x_1-2)(x_2-2) - 0{,}49(x_2-2)^2 - 1{,}90(x_1-2) - \\
1{,}81(x_2-2) - 0{,}78
\end{array}
\tag{4.30}
$$

Obsérvese también en los datos de la tabla 4.1 como añadir una FP más a cualquiera de las dos variables de entrada conlleva un incremento del error de validación, por tanto forzando al algoritmo a parar y retornar la configuración de FPs 3×3 con centros de reglas mostrados en la ecuación 4.30. El error medio obtenido tras 10 ejecuciones diferentes es

igual a 0.085, con una desviación típica de 0.005.

La figura 4.6 muestra la salida obtenida por el modelo, así como las representaciones de los consecuentes polinomiales para las configuraciones intermedias 2×2 y 3×3. Obsérvese como la salida del modelo es similar, ya incluso con la configuración 2×2 a la función original f_1. La configuración final 3×3 proporciona una salida sin ruido muy similar a la función original f_1 (NRMSE = 0.0820). Obsérvese también en (b) y (e), las salidas proporcionadas por los polinomios de las reglas para ambas configuraciones 2×2 y 3×3. Nótese la alta similitud en los gradientes de los polinomios en (b) y (e), y la salida del sistema difuso en torno a los centros de las reglas. Puede encontrarse una comparación más precisa en (d) y (f). Así pues, se ha obtenido un modelo preciso e interpretable, que presenta interpretabilidad también en los modelos locales, para el conjunto de datos proporcionado de la función f_1.

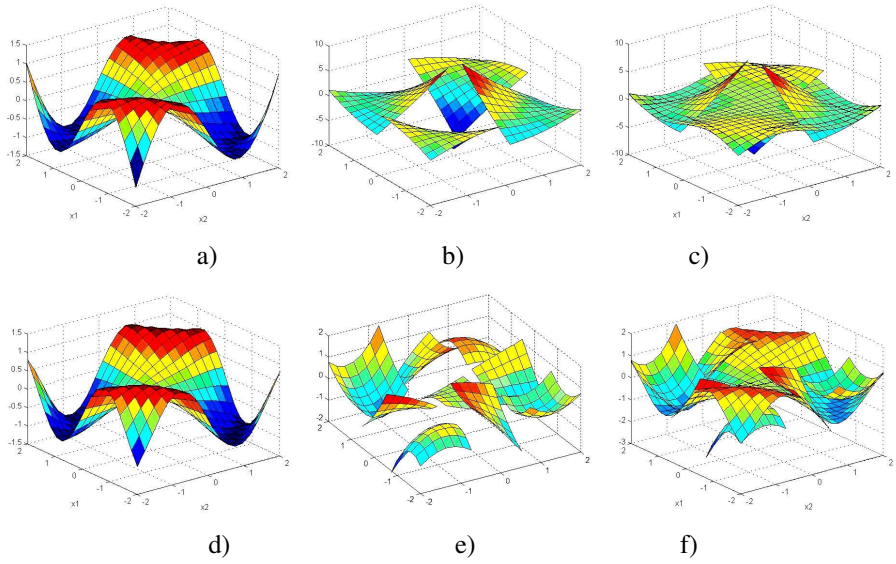


Figura 4.6. *a) Salida del modelo con dos FPs por variable, esto es, 4 reglas en el sistema. b) Consecuentes polinomiales de las 4 reglas del modelo con configuración de FP* 2×2. *c) Gráfico combinado. d) Salida del modelo con 3 FPs por variable, esto es, con 9 reglas. e) Consecuentes polinomiales de las 9 reglas del modelo con configuración de FP* 3×3. *f) Gráfico combinado.*

A partir de este ejemplo puede notarse el notable bajo número de reglas que se han necesitado para aproximar con precisión la función objetivo f_1. Con tan solo 9 reglas interpretables, el error de aproximación obtenido para el ejemplo con ruido es de NRMSE = 0.0820. Otros métodos similares con reglas constantes o lineales necesitan un número mucho mayor de reglas para obtener resultados parecidos [Pomares et al., 2000].

4.6.3. Aplicación a la serie temporal Mackey-Glass

La serie temporal caótica Mackey-Glass [Mackey and Glass, 1977] es un benchmark muy conocido para el modelado de sistemas, que ha sido ampliamente utilizado en la literatura de este tipo de problemas. Esta serie temporal está descrita por la siguiente ecuación

diferencial

$$\frac{dx\left(t\right)}{dt}=\frac{ax\left(t-\tau\right)}{1+x^{10}\left(t-\tau\right)}-bx\left(t\right) \tag{4.31}$$

Se generaron 1000 muestras a partir de la condición inicial $x(0) = 1{,}2$ y $\tau = 17$ utilizando el método de Runge-Kutta de 4º orden. Para realizar comparaciones justas con trabajos previos, se han escogido los parámetros de forma que los vectores de entrada/salida para el modelo tienen el siguiente formato

$$\left[x\left(t-18\right),x\left(t-12\right),x\left(t-6\right),x\left(t\right);x\left(t+6\right)\right] \tag{4.32}$$

La figura 7 muestra la sección de 1000 muestras usadas en este estudio, del las que se han seleccionado 500 para entrenamiento y validación (escogiendo aleatoriamente 400 y 100 respectivamente), y las últimas 500 para test.

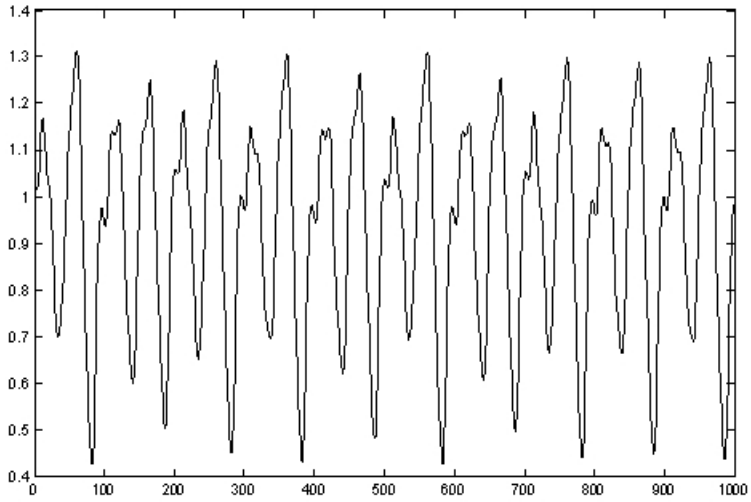


Figura 4.7. *Serie caótica de Mackey-Glass con* $\tau = 17$.

La tabla 4.2 muestra la evolución del algoritmo de identificación de la estructura, que comenzando por una configuración de FP de 1 FP por variable, itera añadiendo 1 FP a la variable seleccionada según el algoritmo presentado en la sección 4.5.

Finalmente la tabla 4.3 compara la precisión en la predicción para diferentes paradigmas computacionales presentes en la bibliografía para este benchmark. Con el objetivo de realizar una comparación equitativa en complejidad, mostramos el error de test para dos configuraciones de FPs del sistema TaSe: $3 \times 2 \times 1 \times 1$ con 91 ($15 \times 6 + 1$ centro móvil) parámetros y $3 \times 2 \times 1 \times 2$ con 181 ´(15×12 + 1 centro móvil) parámetros (ver ecuación 4.7). Los resultados muestran para este ejemplo la conveniencia del modelo TaSe, no solo debido a sus propiedades de interpretabilidad y transparencia y a la interacción entre el modelo global y los modelos locales, sino también debido a su alto grado de precisión para problemas de aproximación funcional y predicción de series temporales. Con tan solo 6 reglas, el modelo TaSe es capaz de identificar el comportamiento no lineal de la serie

#MFs				NRMSE (para predecir $x(t+6)$)			Añad. 1MF en $x(t-18)$	Añad. 1MF en $x(t-12)$	Añad. 1MF en $x(t-6)$	Añad. 1MF en $x(t)$
$x(t-18)$	$x(t-12)$	$x(t-6)$	$x(t)$	Entren.	Valid.	Test				
1	1	1	1	0.032	0.029	0.034	0.013	0.0080	0.016	0.021
1	2	1	1	0.0084	0.008	0.0098	0.0037	0.0069	0.0056	0.0046
2	2	1	1	0.0032	0.0037	0.0035	0.0027	0.0031	0.0034	0.0033
3	2	1	1	0.0024	0.0027	0.0028	0.0025	0.0025	0.0027	0.0024
3	2	1	2	0.0020	0.0024	0.0024	0.0022	0.0020	0.0025	0.0021
3	3	1	2	0.0014	0.0020	0.0017	0.0019	0.0021	0.0022	0.0018
3	3	1	3	0.0011	0.0018	0.0013	El error aumenta al añadir 1 FP a cualquier variable			

Tabla 4.2. *Evolución del algoritmo de identificación de estructura para el problema de predicción de series temporales Mackey-Glass*

Metodo		RMSE de Test
Modelo Auto Regresivo		0.19
RNN de Correlación en Cascada		0.06
RNN Back-Prop.		0.02
Polinomio 6º orden		0.04
Método predictivo lineal		0.55
Kim y Kim (Sistema difuso entrenado con algoritmo genético [Kim and Kim, 1997]	5FPs	0.0492
	7FPs	0.0422
	9 FPs	0.037
Sistema difuso y ANFIS [Jang et al., 1997]		0.007
Red RBF clásica (23 neuronas) [Cho and Wang, 1995]		0.0114
PG-RBF [Rojas et al., 2002]		0.0030
Modelo TaSe con 6 reglas		0.0024
Modelo TaSe con 12 reglas		0.0017
Modelo TaSe Óptimo con 36 reglas		0.001

Tabla 4.3. *Comparación de resultados en error de predicción para diferentes métodos de predicción de series temporales para la serie Mackey Glass*

Mackey-Glass, mejor que muchos otros métodos de predicción de series temporales presentes en la literatura (para una complejidad del modelo similar), disponiendo además de reglas cuyos consecuentes son el desarrollo en serie de Taylor de la salida del modelo en torno a los centros de dichas reglas.

4.7. Estudio Estadístico ANOVA sobre el Rendimiento de Reglas de Orden Alto en los Sistemas TSK y el Sistema TaSe

El objetivo de este análisis estadístico es comparar el efecto en el rendimiento de los parámetros más importantes del diseño de un sistema difuso tipo TSK. Se da un relevancia especial al orden de las reglas del modelo (orden polinómico de los consecuentes: cero, uno y dos). Los sistemas con reglas de orden mayor han estado normalmente asociados a una pérdida en la interpretabilidad de los modelos y a un incremento en el coste computacional en el entrenamiento de los mismos. Sin embargo, en principio, el uso de reglas de orden alto puede producir sistemas con un número mucho menor de reglas. Además como se ha

comprobado el modelo TaSe puede utilizar reglas de orden alto sin perder la interpretabilidad de las mismas, gracias al concepto de expansión en serie de Taylor. Como se muestra en esta sección, los resultados del análisis ANOVA indican que el orden de las reglas usadas es estadísticamente significativo en el rendimiento del modelo. Este estudio se realiza sobre un conjunto de ejemplos de aproximación funcional y de problemas de predicción de series temporales tomados de la literatura. Los resultados muestran la conveniencia de utilizar el modelo TaSe para una amplia gama de problemas de modelado, puesto que puede usar reglas de orden alto para obtener sistemas más sencillos, con reglas interpretables y sin afectar negativamente a la complejidad computacional (si no afectando positivamente en este sentido).

4.7.1. Introducción

Vista la definición de un sistema difuso TSK en temas anteriores, la salida de un sistema TSK puede generalizarse de la siguiente forma:

$$F\left(\vec{x}\right) = G^R\left(I\left(\mu_k\left(\vec{x}\right), Y_k\left(\vec{x}\right)\right), \;\; k = 1..R\right) \tag{4.33}$$

donde I es el método de implicación, que está definido en términos de los llamados operadores T-norma y T-conorma y G^R es el metodo de agregación que toma las salidas de las R reglas para obtener una sola salida. El término $\mu_k\left(\vec{x}\right)$ es el valor de activación de la regla k (considerando solo conexiones AND entre los antecedentes de las reglas) y se expresa utilizando la T-norma T

$$\mu_k\left(\vec{x}\right) = T\left(MF_1^k\left(x_1\right), MF_2^k\left(x_2\right), \cdots, MF_n^k\left(x_n\right)\right) \tag{4.34}$$

Según esta formulación general, existen muchas posibilidades de seleccionar los operadores básicos que intervienen en el proceso de inferencia difusa. Hay numerosas alternativas, que han sido analizadas en la literatura, para el operador de agregación, para el operador de implicación, para los operadores T-norma y T-conorma, el tipo de función de pertenencia, etc. [Rojas et al., 1998], [Cordón et al., 1997], [Valenzuela et al., 2004]. Como se ha comentado en este capítulo y anteriormente en el capítulo de conceptos previos, otro factor a tener en cuenta al diseñar un sistema difuso es el orden de los consecuentes de las reglas.

Esta sección presenta un análisis preciso, que compara la influencia en la salida del sistema cuando los operadores básicos que toman parte en el proceso de inferencia difuso son modificados. Se hace hincapié como se ha comentado en la forma de los consecuentes, más específicamente en el orden de los consecuentes de las reglas. El comportamiento no lineal de este tipo de sistemas obliga a usar herramientas adicionales para estudiar la influencia de los diferentes factores considerados, incluyendo el orden de las reglas, en la complejidad y rendimiento del sistema. Primeramente es de esperar, como se ha justificado anteriormente, que dado un objetivo de rendimiento (medido como error de aproximación), el número de reglas en modelos con reglas de orden alto es mucho menor que en modelos con reglas de orden menor. Sin embargo, ocurre también que al usar reglas de orden superior, el número de parámetros en los consecuentes también se incrementa, y por tanto se suelen asociar las reglas de orden alto a pérdida en la interpretabilidad y a un incremento en la complejidad del modelo. Sin embargo el modelo TaSe evita el problema general de la pérdida de la interpretabilidad en los modelos locales, permitiendo además el uso de reglas de orden mayor. Adicionalmente el análisis que se presenta ahora, muestra que la complejidad computacional no aumenta al usar reglas de orden mayor. Incluso, para el conjunto de ejemplos usados en este trabajo, el rendimiento para complejidades similares (medida la

complejidad como número de parámetros a optimizar) es mejor para reglas de orden mayor que para reglas de orden menor.

La herramienta estadística usada para realizar el análisis es el "ANálisis de la VArianza" (ANOVA) [Box et al., 1978]. La herramienta ANOVA consiste en un conjunto de técnicas estadísticas que permiten el análisis y comparación de experimentos, describiendo las interacciones e interrelaciones entre las variables cualitativas o cuantitativas (llamadas factores en este contexto) del sistema difuso. Los factores que se han considerado como variables más relevantes en el diseño de un sistema difuso tipo TSK son: el operador de agregación, el tipo de función de pertenencia, la T-norma, el orden de las reglas del sistema, y la complejidad del modelo.

4.7.2. Metodología de aprendizaje propuesta

El análisis ANOVA que se presenta en la siguiente subsección, realiza una comparación de los efectos en el rendimiento de las diferentes alternativas de operadores del modelo TSK. Para realizar una comparación justa, se necesita un algoritmo de aprendizaje genérico para todas esas alternativas. En este caso se ha utilizado la misma metodología que se ha expuesto para el sistema TaSe en las secciones 4.4 y 4.5, que es independiente para las modificaciones de los parámetros propuestos (operador de agregación, operador T-norma, tipo de función de pertenencia y orden de los consecuentes de las reglas). La metodología de aprendizaje recordemos se subdivide en identificación de la estructura (o identificación de la partición óptima del espacio de entrada, ver figura 4.8) y ajuste de parámetros (localización óptima de los centros y consecuentes de las reglas, ver figura 4.9).

Notamos aquí que respecto del ajuste de parámetros, en los casos en los que se usan particiones en las variables de entrada, tan solo se necesita optimizar los centros de las FPs. En el caso de utilizar otro tipo de FP, quizás haya que optimizar otros parámetros. En este trabajo se comparan tres tipos de funciones de pertenencia típicos: OLMF, partición triangular y FP Gaussiana (ver apéndice A. En el caso de las FP Gaussianas, para el análisis ANOVA se realiza el cálculo automático de los radios, según las distancias a los centros más cercanos [Moody and Darken, 1989] con el objetivo de realizar una comparación equitativa.

4.7.3. Aplicación del análisis estadístico ANOVA al diseño de inferencia difusa TSK usando reglas de orden alto

La teoría y la metodología del Análisis de la Varianza (ANOVA) fue desarrollada principalmente primera vez por R.A. Fisher [Fisher, 1936], [Fisher, 1950]. Consiste en una serie de técnicas tomadas de la Estadística que permiten analizar y comparar experimentos en los que existe una salida cuantitativa unidimensional descrita como función de una serie de variables, cuantitativas o cualitativas, que se denominan factores [Box, 1989]. ANOVA se utiliza para estimar los efectos directos y de interacción que los factores tienen sobre la salida. El efecto directo es la relación directa entre un factor y la salida; los efectos de interacción son es la relación conjunta de varios factores sobre la salida.

Para la aplicación del análisis de la varianza [Montgomery, 1991], las observaciones de la variables respuesta se expresan como el resultado aditivo de una serie de componentes. En general, si tenemos múltiples factores, la observación $Y_{a,b\ldots l,s}$, donde:

- $a, b \ldots l$ son las diferentes variables o factores del problema

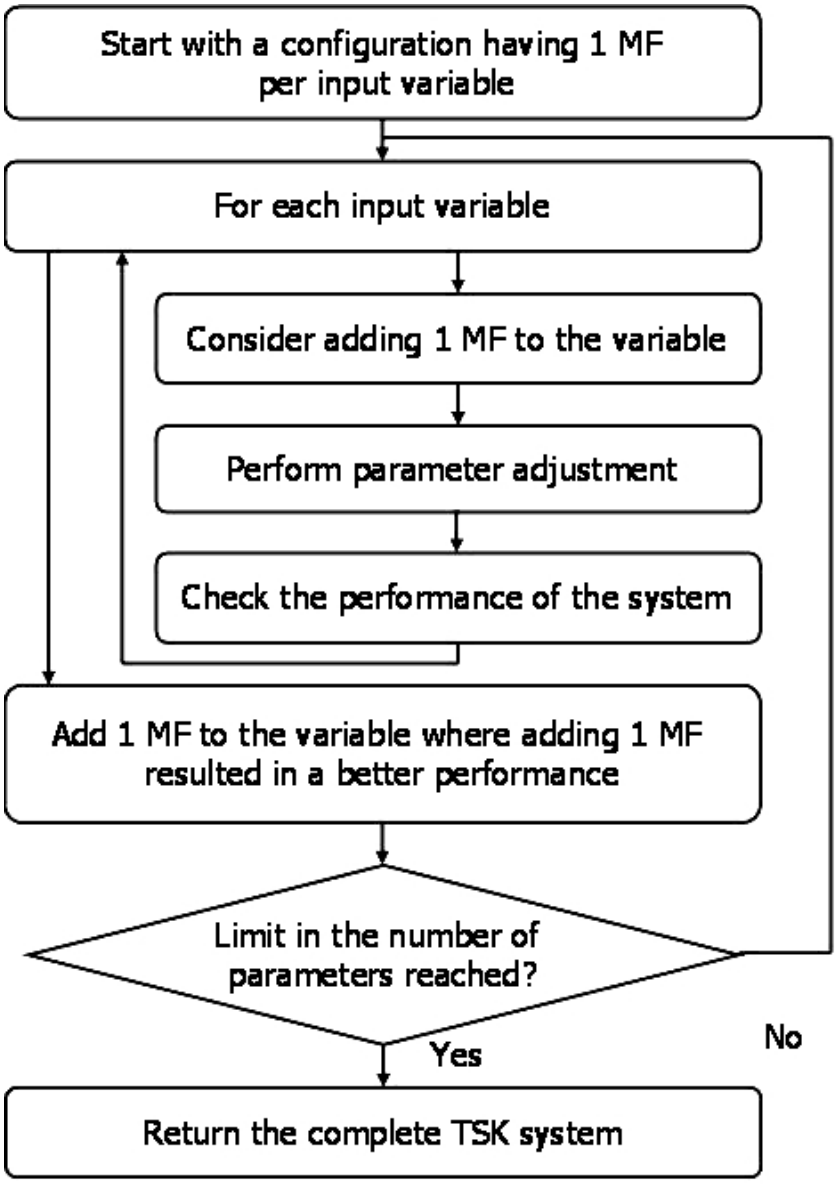


Figura 4.8. *Algoritmo completo de aprendizaje, válido para un modelo TSK genérico.*

- s es el número de observaciones cuando existe repetición de experimentos reales bajo las mismas condiciones o valores de los factores.

puede ser admitida, en una primera aproximación, que es consecuencia aditiva de los efectos de los factores $a, ..., l$. Por tanto la observación puede ser expresada como una suma lineal

$$Y_{a,b,...,l,s} = a + b + \cdots + l + \epsilon \tag{4.35}$$

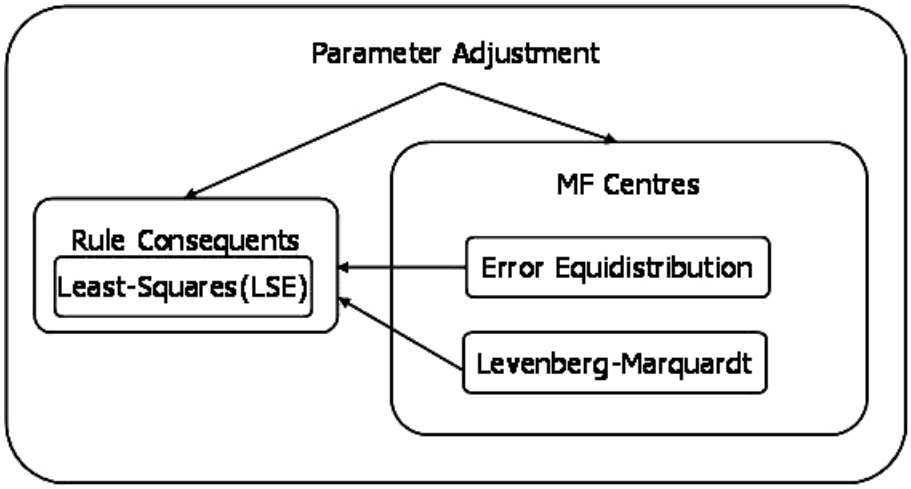


Figura 4.9. *Procedimiento de ajuste de parámetros para una estructura dada en un modelo TSK.*

siendo ϵ el efecto llamado de causas no asignables o de azar. Si los factores $a, b, ..., l$ se mantienen constantes, las medidas de $Y_{a,b...l,s}$ presentarán variaciones que deberán poder considerarse como atribuibles a un gran número de pequeñas causas indistinguibles entre sí, que es lo que se designa como variación aleatoria [Degroot, 1988], [Box, 1989]. Si disponemos un experimento en que midamos $Y_{a,b...l,s}$ a niveles diferentes de uno o más factores, el conjunto de medidas obtenidas puede que no sea homogéneo, estando formado por dos o más grupos. El propósito de la técnica introducida por Fisher es precisamente contrastar esta heterogeneidad, para ver si tales factores son realmente causas asignables en la variación que se trata de estudiar, o bien, se debe atribuir dicha variación al efecto de azar. Es decir, en el análisis de la varianza se trata de separar las componentes de la variación que aparecen en un conjunto de datos estadísticos, determinando si la discrepancia *entre* las medias de los factores son mayores de lo que podría esperarse razonablemente de las variaciones que ocurren *dentro* de los factores.

Debido a la abundante bibliografía, a veces con resultados contradictorios sobre el tipo de orden del modelo TSK ideal para ser utilizado en la lógica difusa, junto con otros parámetros como el tipo de función de pertenencia, tipo de T-norma, complejidad del sistema, etc., en este estudio se pretende abordar el problema desde otro punto de vista. Lo que se pretende es responder a la pregunta: ¿La elección del tipo de orden del modelo TSK para el diseño de un sistema difuso es más significativa para el comportamiento del sistema, que las propias operaciones (T-normas, tipo de defuzzificador, tipo de función de pertenencia, etc.) involucradas en su definición y en el proceso de inferencia?. Recordamos que los factores considerados en este análisis son: el operador de agregación, el tipo de función de pertenencia, la T-norma y el orden del modelo TSK. Con el objetivo de obtener conclusiones significativas, se añaden dos factores más: la función a aproximar (representada por una serie de funciones ejemplo) y la complejidad del modelo (representada por una serie de niveles de complejidad del sistema). Respecto de este último parámetro, debe notarse que al incrementar el orden de la regla TSK, para un número fijo de funciones de pertenencia, el número de parámetros que hay que optimizar se incrementa muy considerablemente. Por tanto para una complejidad fija (número de parámetros que hay que optimizar), los

sistemas difusos con reglas TSK de orden menor tendrán un número más alto de reglas que los sistemas difusos con reglas TSK de orden mayor.

La variable respuesta utilizada para realizar el análisis estadístico es el la raíz del error cuadrático medio normalizado (NRMSE) de la salida del sistema, cuando se modifican los niveles de los factores considerados respecto de un diseño de referencia. Nótese que para obtener un análisis preciso, se deberían haber considerado un número muy alto de funciones ejemplo, con el objetivo de tomarlas como casos aleatorios, y analizar la respuesta del sistema cuando se modifican los niveles de los factores. Esto implica que se debería considerar un número infinito de ejemplos. Esto es obviamente inmanejable, y por tanto, se ha tomado un conjunto de funciones fijo que se ha considerado como significativo. En cualquier caso, como se verá en el análisis de los resultados, el ejemplo considerados muestra ser un factor irrelevante cuando se modifican los niveles del resto de factores con respecto al sistema de referencia. Los cambios en la variable de respuesta ocurren en mayor parte cuando una nueva combinación de T-norma, operador de agregación, tipo de FP y orden del sistema TSK pasa a ser considerada, independientemente del ejemplo tomado. Esto respalda la robustez del análisis realizado.

4.7.4. Configuración de los experimentos

Esta sección introduce la configuración de los experimentos y los diferentes problemas que se han usado en el análisis ANOVA realizado. Hemos seleccionado un amplio rango de problemas de aproximación funcional, así como un conjunto de problemas de predicción de series temporales, con el objetivo de dotar a los resultados obtenidos del análisis ANOVA de suficiente respaldo experimental.

Problemas de aproximación funcional

Sería muy complejo seleccionar un conjunto de funciones significativas que engloben las características más importantes y comportamientos que los problemas de aproximación funcional pueden presentar en general. En [Cherkassky et al., 1996], Cherkassky y otros presentaron una comparación de diferentes metodologías (Vecinos Más Cercanos, Aprendizaje Basado en Memoria Generalizado, Búsqueda por Proyección, Redes Neuronales Artificiales, Splines de Regresión Adaptativa Multivariada y Mapeo Topológico con Restricciones) para aproximación funcional utilizando un conjunto de funciones significativas. Otro trabajo de Rovatti y otros [Rovatti and Guerrieri, 1996] presenta un método de identificación de la estructura para sistemas difusos que se evalúa utilizando cinco funciones ejemplo bidimensionales. El análisis ANOVA toma seis funciones tomadas del trabajo de Cherkassky (f_1-f_6) y cuatro del trabajo de Rovatti (f_7-f_{10}). Éstas son las expresiones de las funciones utilizadas por el análisis

$$f_1 = \sin(x_1 x_2) \qquad x_1, x_2 \in [-2, 2] \tag{4.36}$$

$$f_2 = \exp(x_1 \sin(\pi x_2)) \qquad x_1, x_2 \in [-1, 1] \tag{4.37}$$

$$\begin{aligned} &f_3 = a/(b+c) \qquad \text{con:} \\ &a = 40 \exp\left(8\left((x_1 - 0{,}5)^2 + (x_2 - 0{,}5)^2\right)\right) \\ &b = \exp\left(8\left((x_1 - 0{,}2)^2 + (x_2 - 0{,}7)^2\right)\right) \\ &c = \exp\left(8\left((x_1 - 0{,}7)^2 + (x_2 - 0{,}2)^2\right)\right) \end{aligned} \qquad x_1, x_2 \in [0, 1] \tag{4.38}$$

$$f_4 = 42{,}659\left(0{,}1 + x_1\left(0{,}05 + x_1^4 - 10x_1^2x_2^2 + 5x_2^4\right)\right) \qquad x_1, x_2 \in [-0{,}5, 0{,}5] \tag{4.39}$$

$$f_5 = 1{,}3356\left[\begin{array}{l} 1{,}5\left(1 - x_1\right) + \exp\left(2x_1 - 1\right)\sin\left(3\pi\left(x_1 - 0{,}6\right)^2\right) + \\ \exp\left(3\left(x_2 - 0{,}5\right)\right)\sin\left(4\pi\left(x_2 - 0{,}9\right)^2\right) \end{array}\right] \quad x_1, x_2 \in [0, 1] \tag{4.40}$$

$$f_6 = 1{,}9\left[1{,}35 + \exp\left(x_1\right)\sin\left(13\left(x_1 - 0{,}6\right)^2\right)\exp\left(-x_2\right)\sin\left(7x_2\right)\right] \qquad x_1, x_2 \in [0, 1] \tag{4.41}$$

$$f_7 = \tfrac{1}{2} + 64\tfrac{\left(x_1 - \frac{1}{2}\right)\left(x_2 - \frac{1}{2}\right)\left(x_1 + \frac{1}{8}\right)}{1 + (4x_1 - 2)^2 + (4x_2 - 2)^2} \qquad x_1, x_2 \in [0, 1] \tag{4.42}$$

$$f_8 = \tfrac{1}{2}\exp\left(-20\left[\left(x_1 - \tfrac{1}{2}\right)^2 + \left(x_2 - \tfrac{1}{2}\right)^2\right]\right) + \exp\left(-10\left(x_1^2 + x_2^2\right)\right) \qquad x_1, x_2 \in [0, 1] \tag{4.43}$$

$$f_9 = \tfrac{1}{1 + \exp(10(x_1 - x_2))} \qquad x_1, x_2 \in [0, 1] \tag{4.44}$$

$$f_{10} = \tfrac{1}{2}\left[1 + \sin\left(2\pi x_1\right)\cos\left(2\pi x_2\right)\right] \qquad x_1, x_2 \in [0, 1] \tag{4.45}$$

Problemas de predicción de series temporales

En los últimos años, muchos artículos han tratado el problema del análisis de series temporales mediante redes neuronales artificiales y sistemas difusos. Se ha considerado en primer lugar una serie temporal generada a partir de una ecuación diferencial gobernada mediante determinismo (en el que, una vez el valor inicial es dado, los siguientes estados son determinados) como es el sistema Lorenz [Yu, 1999]. El atractor de Lorenz se deriva de un modelo simplificado de la convección en la atmósfera terrestre. También surge de manera natural el modelos de lasers y dinamos, y es un ejemplo de caos determinístico. La predicción de esta serie temporal está reconocida como un benchmark para la comparación de arquitecturas de redes neuronales y sistemas difusos. Otros tipos de series temporales reales a partir de fenómenos naturales y sociales, por ejemplo, manchas en el sol, cambios estacionales en la temperatura terrestre, demanda de suministro eléctrico, densidad del tráfico, etc. se podrían considerar. Una serie que mide el flujo mensual de un rio (Snake River near Moran, Wyoming; 1904-1994) obtenida de [Tim,] se ha considerado también en el conjunto de ejemplos.

En general, cualquier problema de predicción de series temporales intenta identificar una función F tal que

$$\hat{y}\left(t + h\right) = F\left(y\left(t - \tau_1\right), y\left(t - \tau_2\right), ..., y\left(t - \tau_N\right)\right) \tag{4.46}$$

donde normalmente los parámetros $\tau_1, \ldots, \tau_N$ toman los valores $0, \delta, \ldots\ (N-1)\delta$. En los dos ejemplos considerados $N = 4$, tomando el parámetro δ diferentes valores dependiendo de la serie: el sistema Lorenz tomará el valor comúnmente considerado $\delta = 4$ y el la serie flujo del rio tomará el valor $\delta = 3$ (referimos al trabajo [McNames et al., 1999] para estudiar cómo encontrar un valor adecuado para δ y N para una serie temporal dada).

Niveles de los factores	*Función Ejemplo*	*Operador de Agregación*	*Tipo de FP*	*T-Norma*	*Orden del TSK*	*Complejidad*
1	f_1	Media Ponderada	OLMF	Producto	Orden 0	Baja (54 params)
2	f_2	**Suma Ponderada**	**Triangular**	Minimo	Orden 1	Media (120 params)
3	f_3		Gaussiana	Minimo	**Orden 1**	**Media (120 params)**
4	f_4			Hamacher $\lambda = 0{,}25$	Orden 2	Alta (180 params)
5	f_5			**Hamacher** $\lambda = 0{,}5$		
6	f_6			Hamacher $\lambda = 0{,}75$		
7	f_7					
8	f_8					
9	f_9					
10	f_{10}					
11	Lorenz					
12	River Flow					

Tabla 4.4. *Factores utilizados en el estudio estadístico. Todas las posibles configuraciones de niveles de factores (Operador de Agregación, Tipo de FP, T-norma, Orden TSK, Complejidad) se evalúan para cada uno de los diferentes experimentos considerados.*

4.7.5. Resultados del análisis

Para llevar a cabo el estudio estadístico, se ha hecho una selección de un conjunto de alternativas que son representativas de cada uno de los factores considerados. Analizando los diferentes niveles en estos factores, es posible determinar su influencia en el rendimiento de un sistema difuso tipo TSK. Como se resaltó previamente, la variable respuesta utilizada para realizar este análisis estadístico es el NRMSE de la salida de un sistema difuso, cuando los factores considerados cambian con respecto a una referencia. Esta referencia es la combinación de operador de agregación, tipo de FP, T-norma, orden de las reglas y complejidad que se muestran en negrita en la tabla 4.4, que muestra los diferentes niveles considerados en cada factor para llevar a cabo el análisis ANOVA multi-factorial.

La tabla 4.5 proporciona en análisis de la varianza a seis vías de la metodología difusa TSK estudiada, para todo el conjunto de ejemplos. Para todas las configuraciones de los factores, se ha usado el algoritmo de aprendizaje presentado en la sección 4.7.2. La tabla ANOVA que contiene la suma de cuadrados, los grados de libertad, la media de cuadrados, los tests estadísticos, etc. representa el análisis inicial de forma compacta. Este tipo de representación tabular es la utilizada normalmente para mostrar los resultados de los cálculos del ANOVA.

De toda la información presente en la tabla ANOVA, el mayor interés del investigador se centrará con más probabilidad en los valores de las columnas "F-Ratio" y "Nivel Sig.". Si los números encontrados en estas columnas son menores que un valor crítico (α) establecido por el analista, entonces los efectos se consideran significativos. Este valor se establece normalmente a $0{,}05$, y cualquier valor mejor que este resultará en efectos signi-

Fuente	Suma de Cuadrados	DF	Media de Cuadrados	F-ratio	Nivel Sig.
Factores Principales					
Funciones	0,0548	11	0,00498	3,63	0.0000
Agregación	2,13	1	2,13	1553	0.0000
Tipo de FP	0,184	2	0,0921	67,1	0.0000
T-Norma	0,990	4	0,247	180	0.0000
Orden TSK	0,526	2	0,263	192	0.0000
Complejidad	1,64	2	0,825	601	0.0000
Interacciones Significativas					
Agregación y T-norma	0,782	4	0,196	142	0.0000
Aggregación y Tipo FP	0,315	2	0,158	115	0.0000
Complejidad y Orden TSK	0,741	4	0,185	135	0.0000

Tabla 4.5. *Tabla ANOVA para el análisis de las variables que influyen en el rendimiento del modelo difuso.*

ficativos, mientras cualquier valor mayor que α resultará en efectos no significativos. Si los efectos se consideran significativos utilizando el procedimiento anterior, esto implicará que las medias difieren más de lo que se espera por azar. Como puede verse en la tabla 4.5 todos los factores principales son estadísticamente significativos, pero es importante notar que el orden de las reglas TSK es un factor muy importante (que será analizado después con el test de rangos múltiples) junto con el operador de agregación. También es notable que el factor "Función" es prácticamente irrelevante comparado con el resto de factores, lo que respalda la robustez del análisis.

Si los efectos mostrasen ser no significativos, entonces las diferencias entre las medias no serían lo suficientemente grandes para permitir al investigador afirmar que son diferentes. En ese caso, no se intentaría una interpretación más allá. Cuando los efectos se consideran significativos, esto implica que las medias deben ser examinadas para identificar la naturaleza de los efectos. Hay procedimientos adicionales para asistir al investigador en esta tarea, pero frecuentemente el análisis inicial es evidente al simplemente observar las diferencias entre las medias.

Por tanto, un análisis detallado se realiza ahora para cada uno de los factores examinados, utilizando el test de rangos múltiples. El estudio ANOVA utilizaba dos factores que se necesitaban para obtener resultados sitnificativos, pero que no son relevantes en nuestro objetivo. Estos factores son la "Complejidad" y "Funciones", y se discutirán en primer lugar. Posteriormente se analizarán el resto de los factores. Para el factor complejidad, es obvio que para un mayor número de parámetros empleados en obtener la aproximación, más precisa será la aproximación (ver tabla 4.6). Con respecto al factor "Funciones", se observa (ver tabla 4.7) que es un factor prácticamente irrelevante. Hay cinco grupos homogéneos, pero con bastantes intersecciones no vacías, lo que quiere decir que hay muchas similitudes entre ellos. Esto respalda los resultados estadísticos obtenidos en el análisis del resto de factores; no importa la función ejemplo utilizada para realizar la aproximación

Nivel de la variable "Agregación"	Media Cuad.	Grupos Homogéneos		
Nivel 3: Complejidad alta	-0.0169	X		
Nivel 2: Complejidad media	-0.00237		X	
Nivel 1: Complejidad baja	0.0365			X
Límite para establecer diferencias significativas: 0.003				

Tabla 4.6. *Test de rangos múltiples para el factor "Complejidad"*

Nivel de la variable "Agregación"	Media Cuad.	Grupos Homogéneos				
Nivel 2: f_2	0.000258	X				
Nivel 7: f_7	0.00107	X	X			
Nivel 12: River Flow	0.0173	X	X			
Nivel 3: f_3	0.00210	X	X	X		
Nivel 11: Lorenz System	0.00258	X	X	X		
Nivel 1: f_1	0.00351	X	X	X	X	
Nivel 9: f_9	0.00701		X	X	X	X
Nivel 4: f_4	0.00810			X	X	X
Nivel 6: f_6	0.00941				X	X
Nivel 5: f_5	0.00986					X
Nivel 10: f_{10}	0.0107					X
Nivel 8: f_8	0.0124					X
Límite para establecer diferencias significativas: 0.006						

Tabla 4.7. *Test de rangos múltiples para el factor "Funciones"*

utilizando un modelo TSK que las variaciones en el rendimiento que muestra el análisis cuando se modifican el resto de factores principales, se mantienen.

La tabla 4.8 muestra el test de rangos múltiples para el primer factor "Agregación" (que representa el tipo de operador de agregación utilizado en el sistema de inferencia difuso). A este respecto, debe notarse que la suma ponderada está normalmente más asociada a las redes de función de base radial, en las que la salida del modelo neuro-difuso se obtiene por suma ponderada de las salidas de las neuronas ocultas individuales. Puede notarse claramente que cuando se utiliza media ponderada, el error medio es mucho menor que cuando se utiliza suma ponderada. No obstante, hay casos para los que ambas media ponderada y suma ponderada proporcionan el mismo resultado. Es en el caso en que la propiedad de "suma-a-uno" se cumple: esto es, los casos en los que el operador T-norma se corresponde con el producto, con dos tipos de función de pertenencia que son las bases OLMF y la partición triangular. Estos efectos también se muestran en las dos "interacciones significativas" seleccionadas en la segunda parte de la tabla 4.5. Esto es, hay variaciones importantes en el error cuando el operador de agregación y, la T-norma o el tipo de FP, varían.

Respecto del factor "FPs", que se refiere al tipo de función de pertenencia utilizada en el sistema difuso, hay pequeñas diferencias en el rendimiento (ver tabla 4.9). La base OLMF es el tipo de FP más adecuado de acuerdo con el conjunto de ejemplos de funciones seleccionado para este análisis. En relación al operador T-norma, hay importantes diferen-

Nivel de la variable "Agregación"	Media Cuad.	Grupos Homogéneos	
Nivel 1: Media Ponderada	-0.0199	X	
Nivel 2: Suma Ponderada	0.0314		X
Límite para establecer diferencias significativas: 0.003			

Tabla 4.8. *Test de rangos múltiples para el factor "Agregación"*

Nivel de la variable "FPs"	Media Cuad.	Grupos Homogéneos		
Nivel 2: OLMF	-0.00260	X		
Nivel 1: Triangular	0.00412		X	
Nivel 3: Gaussiana	0.0157			X
Límite para establecer diferencias significativas: 0.003				

Tabla 4.9. *Test de rangos múltiples para el factor "FPs"*

cias. El producto es la T-norma más adecuada, y el operador Hamacher con $\lambda = 0{,}25$ y el mínimo son las que presentan un rendimiento peor (estando el mínimo más relacionado con los sistemas difusos de tipo Mamdani) (ver tabla 4.10). El operador Hamacher con $\lambda = 0{,}75$ es muy similar en rendimiento al producto, debido a su parecido (el operador Hamacher con $\lambda = 1$ equivale al producto). Para estos dos últimos factores, obsérvese que la relevancia en el rendimiento no es tan alta como para el resto de factores; por tanto estos resultados son muy similares a los obtenidos anteriormente en [Rojas et al., 1998].

En relación al factor "orden TSK", que refiere al grado del consecuente polinomial en las reglas TSK, se muestra que el rendimiento de los sistemas que utilizan consecuentes polinomiales de segundo orden es mejor en media que aquellos que utilizan polinomios de primer o de cero orden (ver tabla 4.11). El modelo TSK de orden cero presenta en media el mayor error, con poca diferencia respecto del modelo TSK de primer orden. Debe resaltarse de nuevo que a igualdad en el número de parámetros que optimizar,los sistemas con reglas de consecuente constante tienen un número mucho mayor de reglas que aquellos que utilizan reglas de orden primero o segundo. Este resultado apunto a que la explotación de los parámetros cuando se utilizan reglas de orden segundo es mayor que cuando se utilizan reglas de orden menor. Cuando se utilizan reglas de orden alto, hay un mayor número de parámetros que se obtienen resolviendo un problema que tiene un mínimo global. El tener más parámetros no lineales implica que hay más mínimos locales, incrementándose por tanto las oportunidades de tener un peor rendimiento (como pasa para reglas de orden menor).

Históricamente, las reglas TSK de orden alto no se han utilizado debido a la pérdida de interpretabilidad que conllevaban, comparando con reglas de orden menor; sin embargo este inconveniente puede eliminarse utilizando la metodología TaSe como se ha expuesto en las secciones anteriores. Este resultado apoya la utilidad e importancia del sistema difuso TaSe con reglas basadas en la serie de Taylor propuesto. En este análisis, el modelo TaSe es un caso específico del modelo TSK con el producto como T-norma, media ponderada como operador de agregación, OLMF como tipo de FP, para cualquier complejidad y cualquier orden de reglas TSK. El estudio realizado ha mostrado que para una complejidad

Nivel de la variable "T-norma"	Media Cuad.	Grupos Homogéneos			
Nivel 1: Producto	-0.0142	X			
Nivel 5: Hamacher $\lambda = 0{,}75$	-0.0103	X			
Nivel 4: Hamacher $\lambda = 0{,}5$	0.0012		X		
Nivel 2: Mínimo	0.0219			X	
Nivel 3: Hamacher $\lambda = 0{,}25$	0.0300				X
Límite para establecer diferencias significativas: 0.003					

Tabla 4.10. *Test de rangos múltiples para el factor "T-norma"*

Nivel de la variable "Orden TSK"	Media Cuad.	Grupos Homogéneos		
Nivel 3: TSK orden 2	-0.0122	X		
Nivel 2: TSK orden 1	0.0140		X	
Nivel 1: TSK orden 0	0.0154			X
Límite para establecer diferencias significativas: 0.001				

Tabla 4.11. *Test de rangos múltiples para el factor "Orden TSK"*

dada, las reglas de orden alto necesitan un número mucho menor de reglas, y la precisión en la aproximación puede ser mejor. Adicionalmente, la metodología TaSe permite conservar la interpretabilidad de los sub-modelos locales polinomiales de orden alto, proporcionando la información sobre el valor y las derivadas parciales de p-ésimo orden de la función a aproximar. Así, el modelo TaSe permite condensar precisión y un número pequeño (utilizando reglas de orden alto sin incrementar al menos la complejidad computacional) de reglas interpretables en un modelo difuso TSK para aproximación funcional. Que son tres características importantes y deseables cuando se trata un problema de este tipo.

4.8. Conclusiones

Con el objetivo de evitar la pérdida de la interpretabilidad en las reglas difusas tipo Takagi-Sugeno-Kang, en este artículo se ha presentado una herramienta poderosa para problemas de aproximación funcional utilizando un sistema difuso novedoso con un tipo específico de antecedentes de reglas y consecuentes de reglas basados en la serie de Taylor (sistemas difusos TaSe). Este sistema está por tanto dotado de ambas capacidades, la capacidad aproximativa de las reglas TSK y las ventajas de interpretabilidad de los sistemas difusos tradicionales, puesto que cada consecuente puede verse como la expansión en serie de Taylor en torno a los centros de las reglas. Se ha proporcionado también una metodología automática para realizar tanto la estimación de los parámetros como la identificación de la estructura del sistema difuso TaSe. Se han realizado tres ejemplos de aplicación de detallados del algoritmo de aprendizaje con el objetivo de comprobar la bondad y adaptación de la metodología propuesta a este tipo de problemas.

Adicionalmente y con el objetivo de ganar una mejor comprensión de cómo mejorar el rendimiento de los modelos TSK para problemas de aproximación funcional y predicción de series temporales, en este trabajo se ha llevado también a cabo un análisis ANO-

VA considerando los siguientes factores: el operador de agregación, el tipo de función de pertenencia, la T-norma y el orden de los consecuentes polinomiales de las reglas TSK, para diferentes ejemplos de funciones y diferentes niveles de complejidad del sistema. Se ha dado una relevancia especial al orden de los consecuentes de las reglas TSK, puesto que los modelos de reglas TSK de orden alto pueden presentar interesantes propiedades. Este análisis identifica el operador de agregación y el orden de los consecuentes polinomiales de las reglas como los factores más importantes cuando se diseña un modelo de inferencia difuso tipo TSK. Los niveles que proporcionan el mejor rendimiento para estos dos factores son la media ponderada para el operador de agregación, y los consecuentes polinomiales de segundo orden para el orden de las reglas TSK. Por tanto, puede afirmarse que, utilizando reglas de orden alto, para igualdad en la complejidad del sistema, los sistemas de reglas difusas de orden dos son capaces de alcanzar al menos un rendimiento equivalente que los sistemas de orden menor. Para el conjunto de ejemplos presentado, el análisis muestra que el rendimiento fue incluso mejor para reglas de segundo orden, mostrando que, al menos en algunos casos, la explotación de los parámetros cuando se utilizan reglas de orden mayor es mejor que cuando se usan reglas de orden bajo. Conforme se utilizan reglas de orden mayor, el número de parámetros no lineales decrece; por tanto la posibilidad de caer en mínimos locales también decrece, afectando positivamente al rendimiento.

El problema de este tipo de sistemas es que se han visto tradicionalmente como no interpretables, sin embargo gracias a la utilización del modelo TaSe, es posible utilizar consecuentes de orden mayor con el objetivo de reducir el número de reglas necesarias para obtener la aproximación, sin pérdida de interpretabilidad o precisión. Estos resultados muestran que para problemas de aproximación funcional y predicción de series temporales, para los que el número de reglas esperadas para obtener un buen rendimiento es alto, se puede utilizar un conjunto menor de reglas, con más parámetros por regla explicando el funcionamiento de la función en torno a los centros de las reglas, y con un rendimiento similar o mejor que lo que se podría esperar de los sistemas tradicionales de orden menor.

Capítulo 5

El Modelo Neuro-Difuso TaSe-NF para Problemas de Aproximación Funcional

En el capítulo anterior hemos visto como se puede proporcionar interpretabilidad a los modelos locales en sistemas difusos basados en grid. Sin embargo, en sistemas basados en clustering, y en redes de funciones de base radial y modelos neuro-difusos en general, esta tarea no es tan trivial. En este capítulo se presenta un modelo neuro-difuso que mantiene las propiedades de optimización de los modelos locales a la vez que entrena la red optimizándola globalmente. Además, debido a los problemas de falta de transparencia en los modelos basados en clustering, el trabajo propuesto implementa un enfoque de compartición de funciones de pertenencia, con el objetivo de mejorar la interpretabilidad del conjunto de reglas difusas extraídas del modelo. Para la mejora del rendimiento, se ha implementado también un enfoque apropiado para la inicialización de centros, que supera en rendimiento a otros enfoques presentes en la literatura con este objetivo.

5.1. Introducción

El término neuro-difuso se refiere a híbridos entre redes neuronales artificiales y sistemas difusos. La hibridación resulta en un sistema inteligente que fusiona estas dos técnicas, combinando el estilo razonamiento parecido al humano de los sistemas difusos con la estructura de conexión de las redes neuronales. Los puntos más fuertes de los modelos neuro-difusos son, que se consideran aproximadores universales [Ying, 1998], y que tienen la habilidad de construir un modelo del que se obtiene un conjunto de reglas interpretables SI-ENTONCES.

Los modelos neuro-difusos se utilizan para tratar con problemas de modelado debido a su habilidad para explicar relaciones no-lineales utilizando un número relativamente reducido de neuronas o reglas. Cara uno de estos sub-modelos que forman el sistema neuro-difuso, realiza una aproximación de una parte del espacio de entrada, y conjuntamente colaboran para modelar globalmente la totalidad del espacio de entrada del problema. Estos sub-modelos vienen determinados por las neuronas ocultas en el caso de las redes de función de base radial (redes RBF), y en cualquier caso vienen determinadas por las reglas SI-ENTONCES que se extraen del sistema. Estas reglas pretenden proporcionar entendimiento sobre el funcionamiento global del sistema, pero en particular sobre el funcionamiento del modelo en torno a su área de influencia, que viene dado por su centro c_k y por su anchura σ_k (en el caso particular de funciones de pertenencia gaussianas).

Los modelos neuro-difusos se pueden clasificar según sus dos requisitos: interpretabilidad y precisión. Estos dos requisitos son normalmente contradictorios, y el reflejo de

cada uno de ellos depende del tipo específico de modelo neuro-difuso utilizado. Por tanto, los modelos difusos de Mamdani están más centrados en la interpretabilidad, puesto que utilizan conjuntos difusos tanto en los antecedentes como en los consecuentes de las reglas [Casillas et al.,]. Sin embargo se utilizan raramente para problemas de aproximación funcional y están más centrados en operaciones de control en las que la utilización de términos lingüísticos es esencial. Los modelos Takagi-Sugeno-Kang (TSK) [Takagi and Sugeno, 1985], [Kim et al.,] se consideran más adaptados a la precisión puesto que utilizan funciones lineales de las variables de entrada en los consecuentes de las reglas. Esto proporciona una mayor capacidad de aproximación, siendo por tanto más apropiados para este tipo de problemas en el que los datos de salida son continuos.

Las redes RBF [Guillén et al., 2007], que son equivalentes bajo ciertas condiciones a los sistemas TSK [Jang and Sun, 1993] (ver sección 2.2.2), es también un paradigma neuro-difuso muy utilizado habitualmente, y cuya mayor característica es el alto grado de rendimiento que pueden proporcionar. La diferencia esencial entre las redes RBF y los sistemas difusos TSK es la interpretabilidad, que permite a los sistemas difusos ser más fácilmente comprensibles. Existen algunos trabajos que examinan el problema de la extracción de reglas de redes RBF con el objetivo de dar interpretabilidad al modelo obtenido [Jin and Sendhoff, 2003].

Sin embargo, en general, la optimización de un sistema neuro-difuso en un problema de modelado puede llevar a sistemas en los que la interpretabilidad del conjunto resultante de reglas se pierde o en el que el modelado local carece de significado. El problema de la optimización de los modelos locales rara vez se tata en la optimización de los modelos neuro-difusos [Johansen and Babuska, 2003] [Zhou and Gan, 2004], y el error global del modelo es normalmente el único objetivo a minimizar.

$$J = \sum_{m \in D} \left(f\left(\vec{x}^m\right) - z^m \right)^2 . \tag{5.1}$$

Es importante tener en cuenta en el entrenamiento de un modelo neuro-difuso, que el número resultante de reglas no sea excesivo, que haya poco solapamiento entre las reglas, y que los modelos locales conserven su significado, junto con el funcionamiento correcto del modelo global. Cuando se utiliza particionamiento basado en grid, en el capítulo anterior hemos visto como se puede conservar el significado de los modelos locales durante el proceso de entrenamiento [Herrera et al., 2005f], [Bikdash, 1999]. Sin embargo, los sistemas basados en grid presentan en principio un crecimiento exponencial en el número de reglas que los hace no sables para problemas de cierta complejidad. Los sistemas basados en clustering o las redes RBF son en principio la única solución interpretable a este problema, pero el mantenimiento de la interpretabilidad y de las propiedades de modelado local son un problema mucho más complejo.

En este capítulo se presenta un modelo neuro-difuso modificado que llamaremos TaSe-NF, que permite realizar un aprendizaje global del modelo, mientras que debido a sue estructura interna, optimiza además los modelos locales. Se trata de un modelo TSK mordicado, que presenta también equivalencia con las redes RBF. Permitirá obtener modelos con un número bajo de reglas/neuronas, con un particionamiento del espacio de entrada transparente y con la posibilidad de extraer un conjunto de reglas interpretables gracias al concepto de la expansión en serie de Taylor. Además se ha implementado un enfoque sencillo de compartición de funciones de pertenencia con el objetivo de obtener conjuntos de funciones de pertenencia transparentes en cada variable de entrada, y mejorar así la interpretabilidad del conjunto de reglas extraído. Sin embargo, el modelo propuesto, debido a sus características especiales, reduce parcialmente la flexibilidad esperada del modelo

neuro-difuso diseñado al resolver problemas de modelado. Para reducir esta pérdida, se ha implementado una técnica reciente de inicialización de centros, que supera el rendimiento de otras técnicas clásicas y avanzadas para resolver este objetivo.

El resto del capítulo se organiza de la siguiente forma. La sección 5.2 trata las condiciones de equivalencia de las redes RBF y los sistemas TSK, que es el punto de partida del diseño del modelo TaSe-NF. La sección 5.3 presenta y describe el modelo TaSe-NF propuesto. La sección 5.4 presenta un algoritmo completo de aprendizaje para el modelo TaSe-NF que incluye el Método de Equidistribución de Errores para la inicialización de los centros del modelo neuro-difuso. La sección 5.5 muestra varios ejemplos significativos de aplicación del algoritmo de aprendizaje propuesto utilizando el modelo TaSe-NF. La sección 5.6 finaliza el capítulo.

5.2. Equivalencia entre Redes RBF y Sistemas Difusos TSK

En la sección 2.2.2 ya se expusieron las condiciones de equivalencia entre las redes RBF y los sistemas TSK. Aquí repasamos brevemente esos conceptos como punto de partida para el sistema TaSe-NF. Recordemos que la estructura de un modelo neuro-difuso TSK tradicional [Takagi and Sugeno, 1985] y su proceso de inferencia asociado comprende un conjunto de K reglas SI-ENTONCES en la forma dada en la ecuación 3.1. La salida del sistema TSK, utilizando media ponderada se obtiene según la ecuación 3.2. Es bien conocida la equivalencia entre los sistemas difusos TSK y las redes RBF [Jang and Sun, 1993]; esta equivalencia ocurre esencialmente cuando la salida de la red está normalizada y se utilizan funciones Gaussianas (de base radial) como funciones de pertenencia,

$$\mu_i^k(\vec{x}) = e^{\frac{(x_i - c_i)^2}{2\sigma^2}}. \tag{5.2}$$

Por tanto, dado el enfoque de red RBF, la ecuación 3.2 es la salida de la red neuronal RBF, donde los $\mu^k(\vec{x})$ son las activaciones de las neuronas ocultas, y $Y(\vec{x}^k)$ son los pesos.

Está la posibilidad de utilizar una sola anchura σ para todos los nodos y variables de entrada, en este caso, la optimización del modelo neuro-difuso resulta más sencilla. Sin embargo, los modelos TSK están asociados con tomar una anchura diferente para cada variable en cada nodo. En este caso, la red RBF se denomina red de función de base radial pseudo-gaussiana (PGBFN) [Rojas et al., 2002]. Esto implica un coste computacional más alto en el entrenamiento, puesto que se requiere optimizar un número mayor de parámetros no lineales.

En relación con los pesos de las reglas $Y(\vec{x})$, los modelos TSK de orden 0, 1 o 2 utilizan polinomios de orden 0, 1 o 2. En redes RBF, cuando los pesos de orden mayor que 0 se utilizan, pasan a denominarse redes RBF con pesos de regresión [Rojas et al., 2002].

El modelo TaSe-NF presentado en este capítulo es un modelo neuro-difuso modificado cuyo punto de partida es un modelo difuso TSK de orden 1, equivalente a un modelo general PGBFN con pesos de regresión de primer orden. Sin embargo, el modelo TaSe-NF presentado puede admitir fácilmente un modelo de red RBF más sencillo con una sola anchura para todas las neuronas y/o con pesos de orden 0. En este caso la optimización pasaría a ser mucho más sencilla, pero la flexibilidad del modelo también se reduciría considerablemente, llevando por tanto a un sistema con un número mucho mayor de reglas/nodos.

5.3. El Modelo TaSe-NF

En general, en problemas complejos, los sistemas basados en grid son inviables debido al crecimiento exponencial en el número de reglas. Los sistemas difusos basados en clustering y equivalentemente las redes RBF pueden tratar estos problemas, pese a que en principio son sistemas con más carencias de interpretabilidad. Son también más adecuados por ejemplo para problemas de predicción de series temporales, en los que los datos de entrada se encuentran en torno a algunas zonas del espacio de entrada

El trabajo que aquí se presenta, tiene características similares al modelo TaSe presentado en el capítulo 4, respecto de la optimización global del sistema, y local de los sub-modelos componentes, sin necesidad de sacrificar un factor por otro otro, pero para sistemas basados en clustering. Se permite obtener un número menor de reglas gracias a la utilización de consecuentes de orden primero (o pesos de regresión de primer orden en redes RBF). La gran ventaja del modelo TaSe-NF respecto del modelo TaSe y otros enfoques que pretenden resolver este problema es que el modelo TaSe-NF admite un particionamiento cualquiera del espacio de entrada, problema mucho más difícil de tratar. Además con el objetivo de mejorar la interpretabilidad en sistemas basados en clustering, se ha implementado un enfoque sencillo de compartición de funciones de pertenencia (ver la referencia [Jin and Sendhoff, 2003]), para obtener un conjunto más transparente y sencillo de funciones de pertenencia en cada variable de entrada, para los casos en los que por ejemplo se desee asignar valores lingüísticos a las FP de las variables.

5.3.1. Descripción

El modelo TaSe-NF es un modelo neuro-difuso modificado cuyas consecuentes de las reglas se pueden interpretar como la expansión en serie de Taylor de la salida del modelo en torno a los centros de las reglas. Este resultado es posible gracias a dos características principales de este modelo.

1. Los consecuentes de las reglas tienen una forma general de polinomio centrado que admite la serie de Taylor. El teorema de Taylor establece que si una función $f(x)$ definida en un intervalo tiene derivadas de todos los órdenes, ésta se puede aproximar en torno al punto $x = a$ como su expansión en serie de Taylor en torno a dicho punto (ver adicionalmente la sección 4.3.1)

$$\begin{array}{rl} f(\vec{x}) = & f(\vec{a}) + (\vec{x}-\vec{a})^T \left[\frac{\partial f}{\partial \vec{x}_i}(\vec{a})\right]_{i=1\ldots n} + \\ & \frac{1}{2}(\vec{x}-\vec{a})^T W(\vec{x}-\vec{a}) + \cdots + \\ & \frac{1}{(l+1)!} f^{(l+1)}(\vec{c})(\vec{x}-\vec{a})^{l+1} \end{array} \tag{5.3}$$

   donde $\vec{c}$ es un punto entre $\vec{x}$ y $\vec{a}$, y donde W es una matriz triangular de dimensiones $n \times n$ con las derivadas parciales de segundo orden. La serie de Taylor abre una puerta a la aproximación de cualquier función a través de polinomios, esto es, a través de la adición de un cierto número de funciones sencillas. Es una clave fundamental por tanto en los campos de la Teoría de la Aproximación Funcional y el Análisis Matemático. Utilizando la formulación general del modelo TSK, podemos os utilizar consecuentes $Y(\vec{x})^k$ (ecuación 3.1) que tienen la forma centrada simulando 5.3 (ver subsección siguiente).

2. Los antecedentes de las reglas siguen una estructura que permite dicha interpretabilidad. Esto es una condición doble. Primero, la salida del modelo debe ser continua y

derivable, de forma que el teorema de Taylor se pueda aplicar. Segundo, el grado de solapamiento de todas las FPs debe desvanecerse en cada centro de la regla, de forma que cada punto en el espacio identificado por el centro de una regla, quede afectado tan solo por el consecuente de su regla respectiva en la salida global del sistema.

Revisamos ahora tanto la parte del antecedente de las reglas como la parte del consecuente de las reglas TaSe-NF en detalle. Después se verá la justificación de la interpretabilidad de los modelos locales en el modelo TaSe-NF.

5.3.2. Pesos de las reglas

Considerando un caso 1-dimensional, un modelo difuso TSK con consecuentes polinomiales de orden 1, en el modelo TaSe-NF, la forma de los pesos de las reglas se asemejarían a la fórmula de Taylor según la ecuación 5.3 como sigue

$$\begin{aligned} &\text{SI } x \text{ es } \widehat{c^1} \text{ ENTONCES} \\ &\quad y = A + B(x - c^1) \end{aligned} \tag{5.4}$$

donde $\widehat{c^1}$ es una función de pertenencia sobre x, c^1 es el centro de la regla (traslación a valor discreto del valor difuso en el antecedente) y A (valor de la función) y B (primera derivada respecto de x) son los coeficientes de la serie de Taylor respecto de la salida del modelo en torno a c^1.

La utilización de polinomios de orden alto puede implicar una importante reducción en el número de reglas necesarias para realizar el modelado [Herrera et al., 2005c] (ver sección 4.7). La principal ventaja de este tipo de reglas difusas, comparando con reglas TSK de orden cero, es el incremento en poder expresivo que cada regla puede proporcionar por si misma (ver sección 4.2). Esto es, un número menor de sub-modelos puede ser capaz de identificar la función subyacente que lleva un conjunto de puntos de entrenamiento de entrada/salida. Esta ventaja se complementa con el hecho de que las reglas modelan su área de cobertura del espacio de entrada, explicando el comportamiento (valor de la función y derivadas parciales) de la salida del modelo en torno a los centros de las reglas.

Nótese que los coeficientes de los pesos (w_0^k, $\vec{w}_1^k$, W,...W_s^k) pueden obtenerse de forma óptima utilizando mínimos cuadrados (LSE), y minimizando la función de error 5.1. Ver apéndice B.

5.3.3. Particionamiento del espacio de entrada que permite la interpretabilidad y optimización de los modelos locales

En el capítulo anterior, se ha visto un tipo de función de pertenencia que permite la optimización e interpretabilidad de los modelos locales. Así, el modelo TaSe incluye estas características gracias a la utilización de bases OLMF de FP. Sin embargo, para sistemas difusos basados en clustering, este problema es más complicado. No es fácil obtener un particionamiento del espacio de entrada que cubra la totalidad del espacio y que cumpla con los requisitos previamente mencionados, ni existe un tipo de función de pertenencia que pueda proporcionar estas características a un CBFS. El particionamiento no basado en grid evita esta última posibilidad.

El punto de partida para la siguiente explicación será el tipo de FP Gaussiana, utilizada ampliamente en redes RBF y sistemas TSK basados en clustering. Sin embargo, como se verá, carecen de la propiedad de proporcionar interpretabilidad y modelado local para

sistemas con pesos en las reglas/neuronas de orden más alto. Notamos aquí que una función de pertenencia de base radial distinta de la Gaussiana tradicional, se podría utilizar para proporcionar esta propiedad al sistema.

Considérese una FP Gaussiana, y supóngase el caso sencillo en el que tenemos un espacio unidimensional con dominio [0,1] y dos FPs (por tanto dos reglas) centradas por ejemplo en $c^1 = 0{,}2$ y $c^2 = 0{,}8$ con $\sigma = 0{,}3$ (ver figura 5.1.(a)). En este caso, hay un fuerte solapamiento entre ambas FPs en ambos centros de las reglas. Para evitar este solapamiento permitiremos que el dominio de la primera FP $\mu^1(x)$ esté limitado por la función $1-\mu^2(x)$, esto es, cuando el valor de activación de la otra regla sea 1, la activación de la primera será forzado a tomar valor 0. Más específicamente, el valor de activación para la primera regla $\mu^1(x)$ estará limitado por

$$1-\mu'^2(x);\;\; \mu'^2(x) = \left\{ \begin{array}{cc} \mu^2(x) & \text{if } \; x < c^2 \\ 1 & \text{if } \; x \geq c^2 \end{array} \right. \; labeleq_m fprima_2 \tag{5.5}$$

y de forma similar, el valor de activación de la segunda regla $\mu^2(x)$ estará limitado por

$$1-\mu'^1(x);\;\; \mu'^1(x) = \left\{ \begin{array}{cc} \mu^1(x) & \text{if } \; x > c^1 \\ 1 & \text{if } \; x \leq c^1 \end{array} \right. \tag{5.6}$$

esto es, el valor final de activación para cualquier punto por cualquier regla (ver figura 5.1.(b)) será

$$\mu^{1*}(x) = \mu^1(x)\left(1-\mu'^2(x)\right) \tag{5.7}$$

$$\mu^{2*}(x) = \mu^2(x)\left(1-\mu'^1(x)\right) \tag{5.8}$$

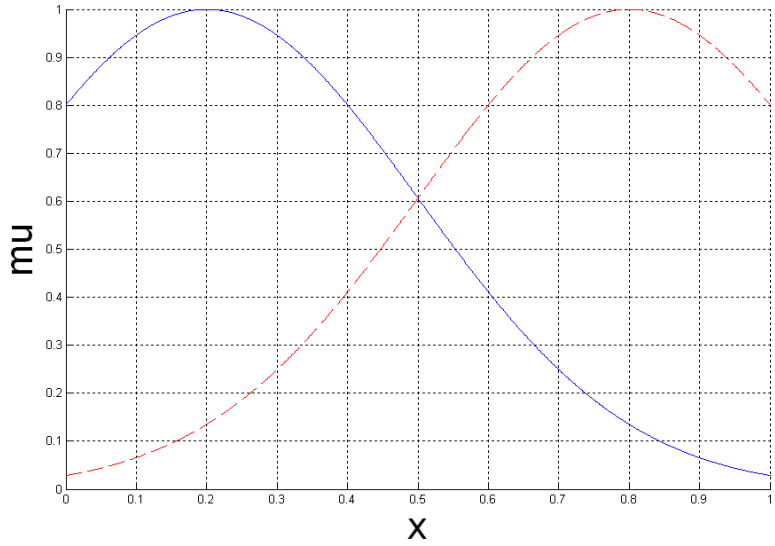
En general, el valor de activación para la regla k se obtiene mediante la siguiente ecuación

$$\mu^{k*}(x) = \mu^k(x) \prod_{j=1}^{\substack{j=K;\\ j\neq k}} \left(1-\mu'^j(x)\right) \tag{5.9}$$

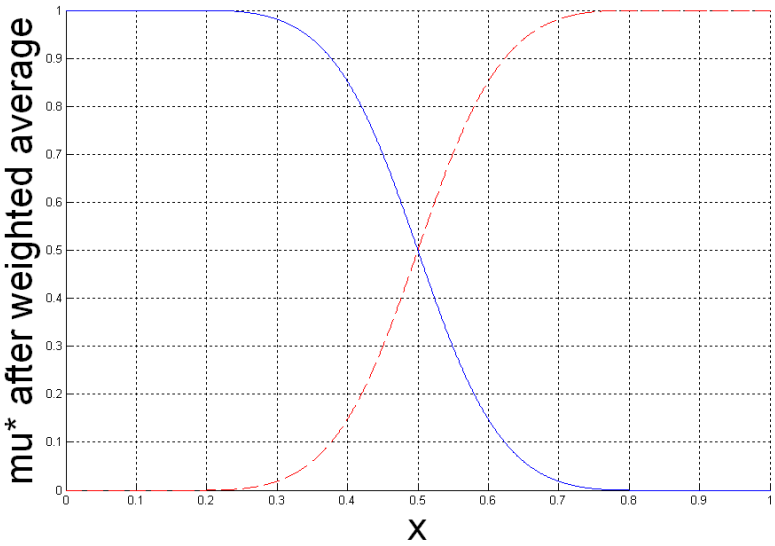
donde utilizando el producto T-norma, y expresando la dependencia en la posición relativa de los centros c_i^j

$$\mu'^j(x) = \prod_{i=1}^{i\leq n} \left\{ \begin{array}{cc} 1 & \text{if } \; \begin{array}{c}(x_i > c_i^j)(x_i \leq c_i^j)\\ (x_i < c_i^j)(x_i \geq c_i^j)\end{array} \\ \mu_i^j(x_i) & \text{elsewhere} \end{array} \right. \tag{5.10}$$

Por tanto, generalizando al caso n-dimensional con cualquier número de reglas K, la expresión general para la salida del sistema neuro-difuso, utilizando media ponderada (que fuerza a que el valor de activación de las reglas en sus centros sea 1) se puede calcular como



(a)



(b)

Figura 5.1. *a) FPs originales para el ejemplo unidimensional. b) Activaciones utilizando el operador de agregación modificado.*

$$F(\vec{x}) = \sum_{k=1}^{K} \hat{\mu}^{k*}(\vec{x}) Y^k(\vec{x}) = \frac{\sum_{k=1}^{K} \mu^{k*}(\vec{x}) Y^k(\vec{x})}{\sum_{k=1}^{K} \mu^{k*}(\vec{x})} = \frac{\sum_{k=1}^{K} \left(\mu^k(x) \prod_{\substack{j=1 \\ j \neq k}}^{j=K} \left(1 - \mu'^j(x)\right) \right) Y^k(\vec{x})}{\sum_{k=1}^{K} \left(\mu^k(x) \prod_{\substack{j=1 \\ j \neq k}}^{j=K} \left(1 - \mu'^j(x)\right) \right)} \tag{5.11}$$

donde $\hat{\mu}^{k*}(\vec{x}) = \mu^{k*}(\vec{x}) / \sum_{k=1}^{K} \mu^{k*}(\vec{x})$ es la activación con media ponderada final para la regla k. Esta nueva formulación de la salida del sistema en la ecuación 5.11 se puede ver como un operador de agregación modificado con comportamiento de media ponderada.

5.3.4. Modelado local de las reglas

Para el modelo modificado presentado, tomando este ejemplo unidimensional con reglas según la ecuación 5.4, se cumplen las siguientes propiedades

$$\hat{\mu}^{2*}(c_1) = 0; \hat{\mu}^{1*}(c_1) = 1; \Rightarrow F(c_1) = Y^1(c_1) = A_1$$
$$\hat{\mu}^{1*}(c_2) = 0; \hat{\mu}^{2*}(c_2) = 1; \Rightarrow F(c_2) = Y^2(c_2) = A_2$$

además, gracias a las propiedades de derivabilidad de la función de pertenencia gaussiana

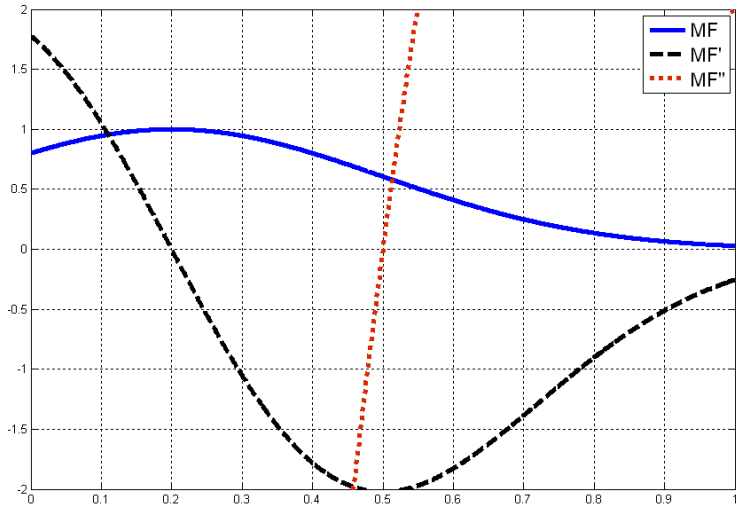
$$\frac{\delta \hat{\mu}^{1*}}{\delta x}(c_1) = 0 \wedge \frac{\delta \hat{\mu}^{2*}}{\delta x}(c_1) = 0 \Rightarrow \frac{\delta F}{\delta x}(c_1) = \frac{\delta Y^1}{\delta x}(c_1) = B_1 \tag{5.12}$$
$$\frac{\delta \hat{\mu}^{2*}}{\delta x}(c_2) = 0 \wedge \frac{\delta \hat{\mu}^{1*}}{\delta x}(c_2) = 0 \Rightarrow \frac{\delta F}{\delta x}(c_2) = \frac{\delta Y^2}{\delta x}(c_2) = B_2 \tag{5.13}$$

Sin embargo, la segunda derivada de la función de base radial gaussiana no se anula en el centro, y por tanto, la continuidad de la segunda derivada de la función μ^* no se garantiza. En efecto, la figura 5.2 muestra la primera y segunda derivadas de $\mu^1(x)$ y de $\hat{\mu}^{1*}(x)$ para este ejemplo.

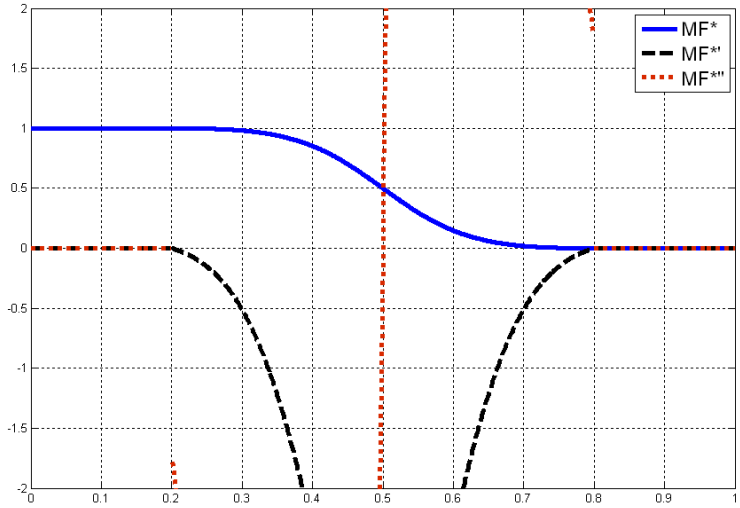
Para casos de dimensionalidad mayor, la figura 5.3 muestra un ejemplo de un caso bidimensional con dominio [0,10] con tres FPs de tipo gaussiano, con centros en [3,3], [5,5] y [7,7] con $\sigma = 2$. De nuevo, los valores de activación para los centros de las reglas son 0 excepto para la regla respectiva. Además tememos que mencionar que aunque los CBFSs se han visto normalmente como sistemas poco interpretables debido al solapamiento de las FPs, y puesto que cada regla tiene su propia FP en cada variable de entrada [Guillaume, 2001], este modelo realiza un pseudo-particionamiento del espacio de entrada que es muy intuitivo como se observa en la figura 5.3; ninguna regla tiene efecto en el resto de centros de reglas, y además, el valor de activación está limitado según la localización del resto de centros en el espacio n dimensional de entrada. Cada regla define una región en el espacio de entrada donde tiene un valor de activación positivo en torno a su respectivo dentro de regla, que está parcialmente limitado por la posición del resto de centros de las reglas.

Trasladando este nuevo enfoque explícitamente a la metodología de redes RBF, la estructura tradicional en tres capas de la red RBF se podría modificar de forma que se introduce una nueva capa entre la capa oculta y la capa de salida, que realizase los cálculos necesarios según la ecuación 5.11. Además, para los coeficientes de los consecuentes de las reglas basadas en la serie de Taylor, los ya comentados pesos de regresión [Rojas et al., 2002] se pueden expandir para admitir pesos polinomiales de orden mayor que 0.

Dadas las características del modelo TaSe-NF, que utiliza la formulación de consecuentes de las reglas dada en las ecuaciones 5.3 y 5.4, y según el operador de agregación dado en las ecuaciones 5.9 y 5.11 que hacen que las derivadas de la salida del sistema en torno a los centros de las reglas tomen valores según las ecuaciones 5.12 y 5.12, entonces los consecuentes $Y_k(\vec{x})$ se pueden interpretar como la expansión en serie de Taylor truncada de orden 1 de la salida del modelo en torno a los centros de cada regla respectiva k. Este



(a)

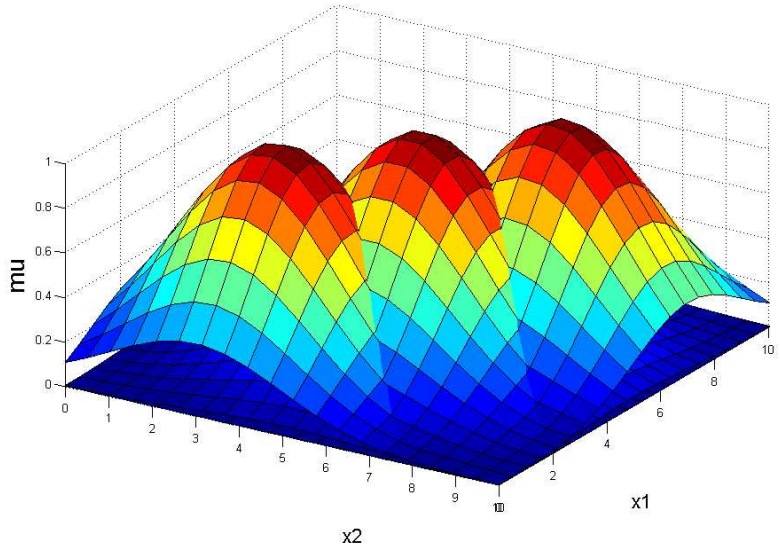


(b)

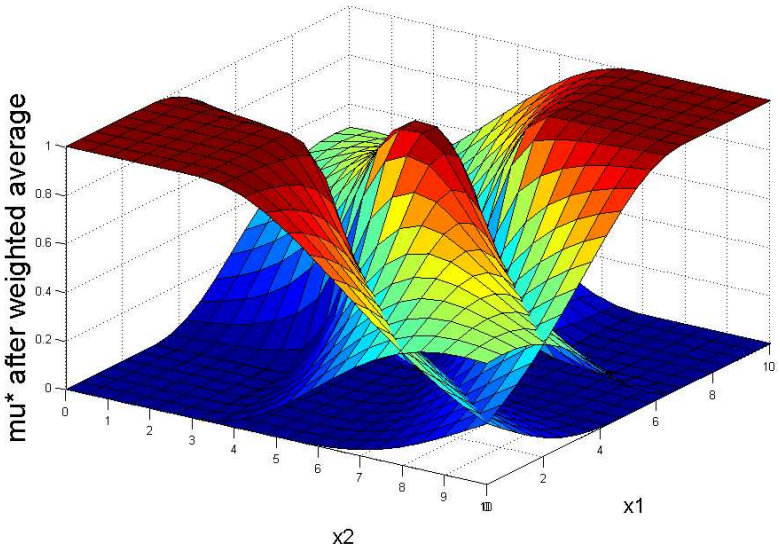
Figura 5.2. *a) Función de pertenencia $\mu^1(x)$, con sus primera y segunda derivadas. b) Función de pertenencia $\hat{\mu}^{1*}(x)$, con sus primera y segunda derivadas.*

resultado es similar al teorema de Bikdash para GBFS que se utilizó en el capítulo anterior, al ser similares las bases de ambos sistemas TSK modificados.

La prueba de esta afirmación viene de las propiedades de derivabilidad de la función Gaussiana, de los consecuentes polinomiales y del operador T-norma producto. Nótese que como se ha mencionado, estas propiedades ocurren para pesos de polinomios de primer



(a)



(b)

Figura 5.3. *a) Original MFs for a two-dimensional example. b) Activations using the modified Aggregation Operator.*

orden; para utilizar polinomios de orden mayor, se tendría que utilizar una función de base radial diferente, para la que la n-ésima derivada en el centro se anulase. En este trabajo se utilizará la función gaussiana tradicional y pesos de primer orden.

5.3.5. Compartición de funciones de pertenencia

Visto como modelos separados, la mayor diferencia entre los modelos difusos y las redes RBF es la interpretabilidad. El modelo TaSe-NF que se presenta en este trabajo es un modelo neuro-difuso modificado basado en un equivalente TSK-RBF, que obtiene un conjunto de reglas interpretables y que optimiza tango el modelo global como los modelos locales. Sin embargo, en general, las redes RBF y los CBFSs padecen de la falta de tansparencia en el particionamiento del espacio de entrada. En el modelo TaSe-NF, como se ha visto, la transparencia en el espacio n-dimensional es muy buena (ver las figuras 5.1 y 5.3). Sin embargo, desde el punto de vista de los conjuntos difusos en cada variable de entrada, esos modelos pueden presentar un particionamiento no transparente y el número de conjuntos difusos iguala al número de reglas (neuronas ocultas), entorpeciendo la utilización de etiquetas lingüísticas en las reglas. Esta subsección presenta un método sencillo de compartición de funciones de pertenencia que será incluido en el algoritmo de aprendizaje, con el objetivo de mejorar la interpretabilidad del conjunto de reglas obtenido.

La idea de la compartición de FPs, es identificar FPs similares en diferentes neuronas y hacer que compartan el mismo centro y anchura con el objetivo de reducir el número de FPs solapadas en cada variable. Se han utilizado numerosas medidas de similitud entre FPs; entre éstas, la distancia euclídea es sencilla y se ha utilizado frecuentemente. La distancia euclídea entre dos funciones de pertenencia gaussianas con parámetros $\mu_1(c_1, \sigma_1)$ y $\mu_2(c_2, \sigma_2)$ se define como

$$d(\mu_1, \mu_2) = \sqrt{(c_1 - c_2)^2 + (\sigma_1 - \sigma_2)^2} \tag{5.14}$$

Dada esta medida de la distancia, la idea bajo la compartición de FPs es unir FPs similares. Dos FPs se unirán si la distancia entre ellas es menor que un cierto umbral δ, creándose una nueva FP con centro $c' = (c_1 + c_2)/2$ y anchura $\sigma' = (\sigma 1 + \sigma 2)/2$. Esto implica que por un lado, la transparencia a lo largo de la variable de entrada correspondiente aumentará, puesto que las FPs solapadas se unirán, y por otro lado, esto permitirá el uso de etiquetas lingüísticas en las FPs. Sin embargo, el rendimiento del sistema cuando se comparten FPs puede disminuir considerablemente.

Así, el algoritmo de aprendizaje completo presentado en la siguiente sección, presenta dos fases de descenso en gradiente, donde la segunda de ellas optimiza los parámetros de las FPs después de que la fase de compartición se haya realizado, con el objetivo de optimizar el rendimiento del modelo con el conjunto de FPs interpretable reducido.

5.4. Metodología de Aprendizaje para el Modelo TaSe-NF con Inicialización de Centros mediante Equidistribución de Errores

Esta sección presenta una metodología de aprendizaje para el modelo TaSe-NF. En general, el procedimiento de aprendizaje en un modelo CBFS o en una red RBF se puede resumir como

- inicialización de los centros de las reglas y otros parámetros de las FPs
- obtención de los coeficientes óptimos de los consecuentes utilizando LSE
- procedimiento de búsqueda local para optimizar centros y otros parámetros de las FPs

La metodología de aprendizaje del modelo TaSe-NF sigue esos mismos pasos. Con la peculiaridad de que el procedimiento de búsqueda local tiene dos fases, con una segunda etapa de optimización de las FPs compartidas. Además la fase de inicialización de centros se realizará utilizando el Método de Equidistribución de Errores (EEM), que se presentó por primera vez en [Herrera et al., 2005a]. Ahora revistamos ambas fases del procedimiento de aprendizaje, de inicialización de centros y de búsqueda local. Los consecuentes óptimos se obtienen utilizando LSE y resolviendo un sistema de ecuaciones lineales como se realiza habitualmente [Herrera et al., 2005f], [Gonzalez et al., 2002].

5.4.1. Inicialización de centros utilizando el Método de Equidistribución de errores en sistemas basados en clustering

La inicialización de los centros de las reglas es un paso crítico que ha demostrado tener una alta influencia en el rendimiento final de los modelos neuro-difusos (ver referencia [Gonzalez et al., 2002]). En la literatura, los algoritmos de clustering se han utilizado para la inicialización de centros en modelos TSK y en redes RBF. El uso de enfoques de clustering para la inicialización de centros de las reglas se basa sobre todo en la idea de obtener una partición en clusters del espacio de entrada y asignar un peso o valor funcional a esa región del espacio de entrada. Sin embargo, esta idea puede ser más apropiada en problemas de clasificación; en problemas de aproximación funciona, la interrelaciones entre clusters en el espacio de entrada no lleva consigo necesariamente dicha interrelación en la salida. Las técnicas de clustering de entrada-salida [Gonzalez et al., 2002], [Uykan et al., 2000] resuelven parcialmente este problema puesto que consideran la variable de salida en el proceso de clustering.

El principal objetivo del enfoque de equidistribución de errores, en vez de pretender minimizar una medida de distorsión clásica basada en la distancia de los datos de entrenamiento a los centros de las reglas es intentar situar los centros de las reglas de forma que los errores (según la función de error 5.1) se distribuyan homogéneamente sobre la totalidad del rango de salida.

Comenzando con una inicialización aleatoria (o usando cualquier enfoque como las k-medias [Duda and Hart, 1973]) de distribución de los centros, consideraremos que un centro k es responsable para el error correspondiente a cada punto de entrenamiento $\vec{x}^m$ utilizando la siguiente fórmula

$$J^k(\vec{x}^m) = \frac{\mu_k(\vec{x})}{\sum_{j=1}^{K} \mu_j(\vec{x})} \left(f\left(\vec{x}^m\right) - z^m\right)^2 \tag{5.15}$$

siendo por tanto

$$J = \sum_{m \in D} \sum_{j=1}^{K} J^k(\vec{x}^m) \tag{5.16}$$

Cada centro k por tanto tendrá parámetros asociados S_{i-}^k y S_{i+}^k, que reflejarán el error según la ecuación 5.1 a la "izquierda" (signo negativo) y a la "derecha" (signo positivo) del centro c_i^k (esto es, centro de la FP en la regla k para la dimensión i).

$$S_{i+}^k = \sum_{\substack{m \in D \\ x_i^m \geq c_i^k}} J^k(\vec{x}^m) \tag{5.17}$$

$$S_{i-}^k = \sum_{\substack{m \in D \\ x_i^m < c_i^k}} J^k(\bar{x}^m) \tag{5.18}$$

Utilizando estos dos parámetros S_{i-}^k y S_{i+}^k, definiremos un parámetro de pendiente

$$S_i^k = \frac{S_{i+}^k - S_{i-}^k}{\sigma_y^2} \tag{5.19}$$

(donde σ_y^2 es la varianza de salida) que reflejarán la necesidad de ese centro de moverse de "izquierda" a "derecha" con el objetivo de equidistribuir el error J a lo largo del espacio de entrada.

Se realiza un proceso iterativo que calculará este parámetro de pendiente para cada dimensión en cada centro, que posteriormente moverá los centros de las reglas según la siguiente fórmula

$$\triangle c_i^k = \begin{cases} \frac{d_{min}}{b} \frac{S_i^k}{S_i^k + T_i^k}, & \text{if } S_i^k \geq 0 \\ \frac{d_{min}}{b} \frac{|S_i^k|}{|S_i^k| + T_i^k}, & \text{if } S_i^k < 0 \end{cases} \tag{5.20}$$

donde d_{min} es la distancia Euclídea del centro k al centro más cercano (o a los límites del espacio de entrada), b es el radio activo que es la distancia de variación máxima y se usa para asegurar que el cluster no sustituye su posición relativa en el espacio de entrada (un valor típico es $b = 4$, significando que, como mucho, un centro se puede mover tanto como la mitad del punto medio entre un centro y su vecino), y T_i^k es un parámetro de temperatura inversa que dotará nuestra metodología con un comportamiento de "enfriamiento simulado", en el sentido de que comenzará con un valor pequeño (permitiendo movimientos grandes de los centros) y conforme el signo de S_i^k cambie, se hará más grande (por tanto permitiendo movimientos más cortos de los centros. En este proceso, la anchura de la función gaussiana se calcula automáticamente utilizando un enfoque sencillo [Moody and Darken, 1989].

En este proceso iterativo, se realiza un proceso de migrado como el que presenta el algoritmo ELBG [Russo and Patane, 1999], con el objetivo de situar los centros de las reglas para los que (para cualquier i) la distorsión total

$$D^k = S_{i+}^k + S_{i-}^k \tag{5.21}$$

es menor que el valor medio de distorsión, cerca de aquellos centros para los que la distorsión es mayor que la media. Este proceso de migración asegurará que cualquier región del espacio de entrada no quedará insuficientemente cubierto y acelerará la convergencia del proceso.

El proceso finaliza cuando la configuración de centros alcanza una distribución equilibrada de los errores, esto es, cuando la suma de pendientes

$$E = \sum_{\substack{k=1\ldots K \\ i=1\ldots n}} |S_i^k| \tag{5.22}$$

no disminuya con respecto a iteraciones anteriores. Este procedimiento requiere pocas iteraciones para obtener un sistema neuro-difuso que, aunque por si mismo no representa una buena aproximación, se asume será un buen punto de comienzo para buscar una aproximación cercana al óptimo de la función objetivo, utilizando técnicas de búsqueda de mínimos locales. la figura 5.4 muestra el diagrama de flujo del algoritmo completo de EEM.

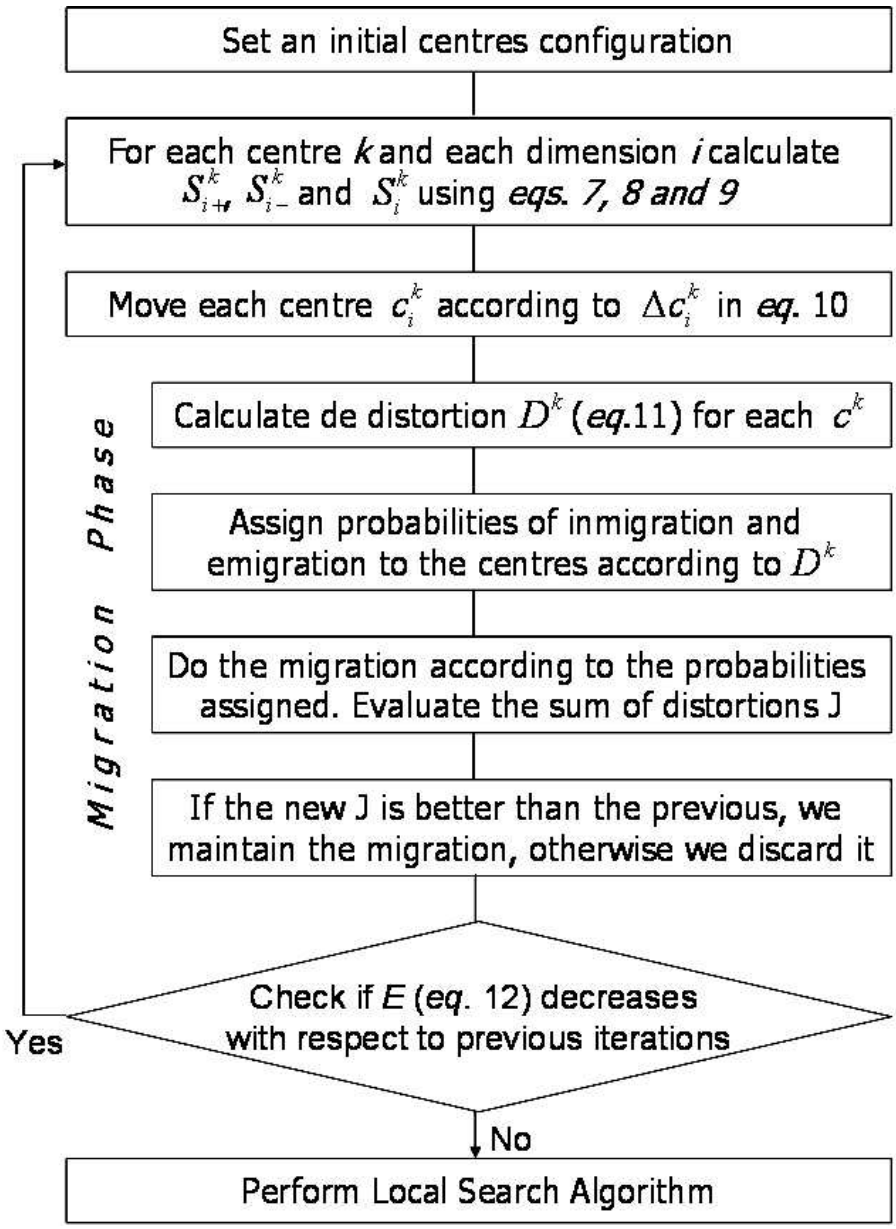


Figura 5.4. *Método de Equidistribución de Errores para la inicialización de los centros en CBFSs*

5.4.2. Procedimiento de búsqueda local

Una vez que tenemos una buena configuración inicial de los centros, cualquier procedimiento de búsqueda local se podría usar para encontrar un buen óptimo local de los parámetros de las neuronas. Aunque en la inicialización de centros la anchura de las funciones gaussianas se calcula automáticamente, en la búsqueda local, ambos los centros y

las anchuras se pueden optimizar para obtener un sistema más preciso según la función de error en la ecuación 5.1. En este trabajo, hemos utilizado el algoritmo de Levenberg-Marquardt [More, 1978] debido a su eficiencia; no obstante, cualquier procedimiento de búsqueda local podría tenerse en cuenta según la naturaleza del problema.

Como se explicó, el procedimiento de compartición de FPs realiza la unión de dos o más centros similares (en una dimensión dada), por tanto compartiendo un solo parámetro. Es una decisión delicada cuándo realizar el procedimiento de compartición de FPs desde el punto de vista de la optimización y del rendimiento final del modelo. En trabajos anteriores [Jin and Sendhoff, 2003] se incluyó en cada paso de la búsqueda local, junto con una compartición adaptativa de los pesos. Sin embargo, el coste computacional puede incrementarse considerablemente. En este trabajo, hemos realizado la compartición de las FPs al final del proceso, entonces, un segundo procedimiento breve de búsqueda local se ha ejecutado para optimizar los nuevos centros compartidos. Este procedimiento muestra obtener buenos resultados. La figura 5.5 muestra un diagrama con el algoritmo completo de aprendizaje para el modelo TaSe-NF.

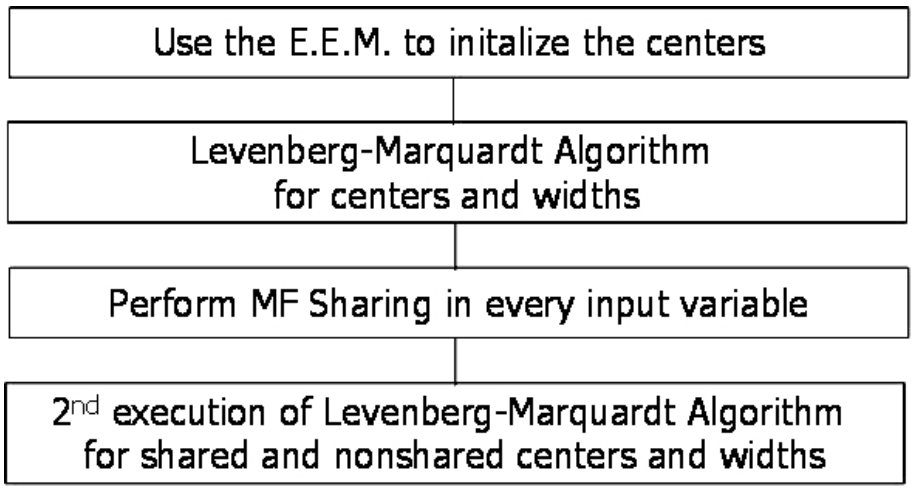


Figura 5.5. *Complete learning algorithm for the TaSe-NF model*

5.5. Simulaciones

5.5.1. Ejemplo Unidimensional

Ahora presentamos un ejemplo de aplicación del modelo TaSe-NF para aproximación funcional. Dada la función f_3 [Cherkassky et al., 1996] que se expresa como

$$\begin{aligned} f_3 &= a/(b+c) \qquad \text{con:} \\ a &= 40\exp\left(8\left((x_1-0{,}5)^2+(x_2-0{,}5)^2\right)\right) \\ b &= \exp\left(8\left((x_1-0{,}2)^2+(x_2-0{,}7)^2\right)\right) \\ c &= \exp\left(8\left((x_1-0{,}7)^2+(x_2-0{,}2)^2\right)\right) \end{aligned} \qquad x_1, x_2 \in [0,1] \tag{5.23}$$

se generaron 400 puntos equidistribuidos como conjunto de entrenamiento. Se usaron 400

datos más generados aleatoriamente como conjunto de test. Como ejemplo de ejecución, se inicializaron 10 centros utilizando el algoritmo de clustering de las k-medias (ver referencia [Duda and Hart, 1973]), y la compartición de FPs se hizo con umbral δ establecido a $0{,}09$. La figura 5.6 muestra la forma de la función original f_3. Como se observa, es una función de fácil de caracterización, pero que se ha utilizado como ejemplo para ver las características de interpretabilidad y modelado local óptimo que presenta el modelo TaSe-NF propuesto.

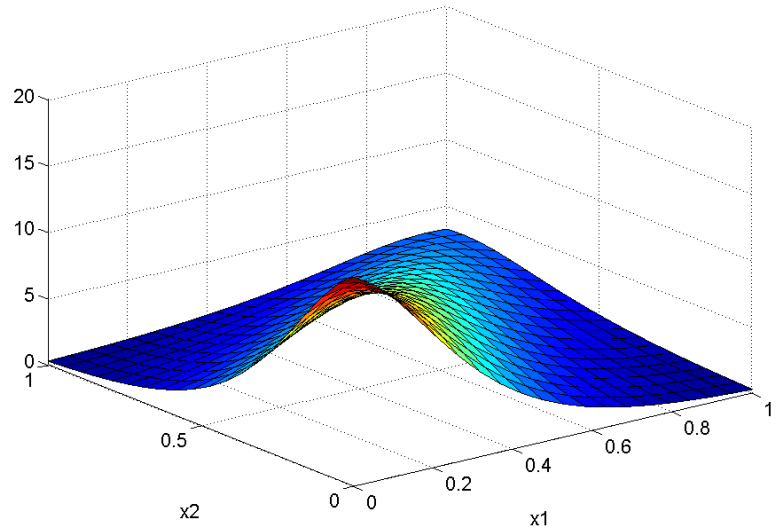


Figura 5.6. *Función original f_3.*

La configuración inicial de centros da 10 FPs por variable, al disponerse de 10 centros en el espacio bi-dimensional. La tabla 5.5.1 muestra la configuración inicial y final de centros antes, y después de ejecutar el algoritmo completo. Tras la aplicación de la metodología de aprendizaje completa, que incluye el mezclado de FPs, solo 7 FPs diferentes quedan en la primera variable, y 5 en la segunda variable. Las figuras 5.7 y 5.8 muestran las funciones de pertenencia que han quedado en cada variable tras el proceso de optimización completo. Obsérvese la separación y trasparencia del conjunto resultante de FPs en cada variable gracias al proceso de compartición de FPs.

La figura 5.9 muestra los valores de activación para las 10 neuronas finales del sistema Tase-NF que incluye agregación mediante media ponderada (con la que la suma de las activaciones para cualquier punto es 1, y la activación en cada centro de cada regla es 1 para su respectiva regla). Ahí se puede observar la distribución de centros obtenida tras la ejecución del algoritmo y cómo para algunos centros se comparten los conjuntos difusos de alguna de las variables.

La figura 5.10 muestra la aproximación final obtenida, utilizando la configuración de centros y sigmas final obtenida. La Raíz del Error Normalizado Cuadrático Medio (NRMSE [Pomares et al., 2000]) de entrenamiento es 0.108, mientras que el NRMSE de test obtenido es de 0.112 para este ejemplo.

Finalmente respecto de la interpretabilidad de los modelos locales, las figuras 5.11 y 5.12 muestran la representación de los polinomios en los consecuentes de las reglas que

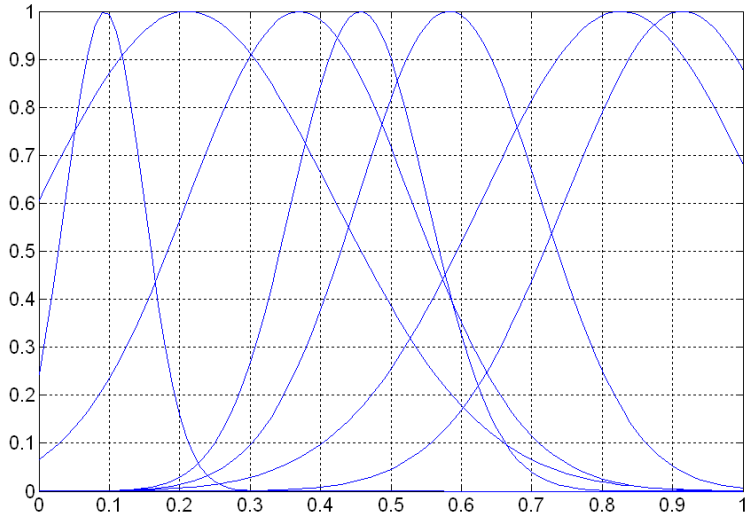


Figura 5.7. *Funciones de pertenencia resultantes para la variable x_1 en el problema.*

	centros k-medias		centros óptimos	
1	0.4858	0.8745	0.3702	0.4519
2	0.8421	0.8947	0.5826	0.9346
3	0.8467	0.1022	0.2103	0.1087
4	0.8467	0.3653	0.8268	0.2043
5	0.5043	0.1591	0.3702	0.1087
6	0.8467	0.6347	0.9143	0.5256
7	0.1476	0.8352	0.2103	0.9346
8	0.1579	0.5000	0.2103	0.5256
9	0.1579	0.1579	0.0936	0.2043
10	0.5084	0.5251	0.4559	0.5256

Tabla 5.1. *Configuración inicial y final de centros antes, y después de ejecutar el algoritmo completo de aprendizaje. En la configuración final, hay tan solo 7 FPs en la dimensión 1 y 5 en la dimensión 2, compartidas entre los 10 centros bidimensionales*

se muestran a continuación (centrados en torno a la posición del centro de cada regla respectiva). Puede observarse como los consecuentes de las reglas indican exactamente el valor de la función en ese punto, así como los valores de las pendientes en cada dimensión, definiéndose en cada caso un plano que indica exactamente el valor de la pendiente de la salida del modelo en torno al centro de la regla.

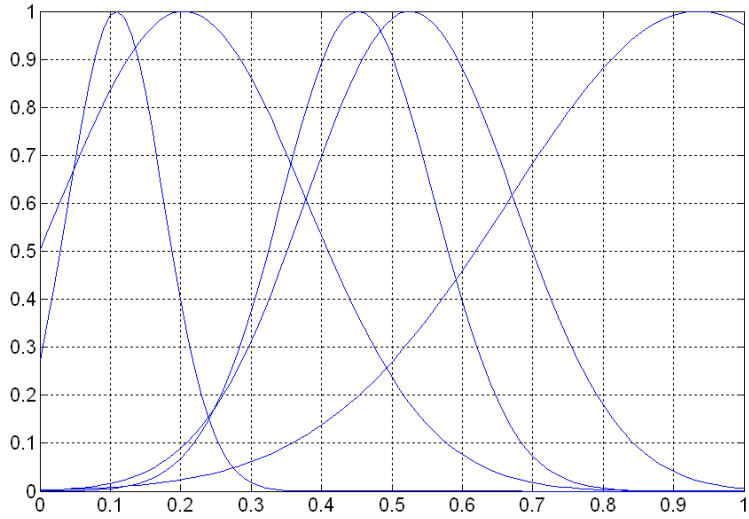


Figura 5.8. *Funciones de pertenencia resultantes para la variable x_2 en el problema.*

$$
\begin{array}{l}
\text{SI } x_1 \text{ es } \widehat{0{,}4519} \text{ Y } x_2 \text{ es } \widehat{0{,}3702} \text{ ENTONCES } y = 7{,}6954 - \\
12{,}6659(x_1 - 0{,}4519) - 6{,}6870(x_2 - 0{,}3702) \\
\text{SI } x_1 \text{ es } \widehat{0{,}9346} \text{ Y } x_2 \text{ es } \widehat{0{,}5826} \text{ ENTONCES } y = 2{,}2264 - \\
11{,}1942(x_1 - 0{,}9346) + 4{,}8117(x_2 - 0{,}5826) \\
\text{SI } x_1 \text{ es } \widehat{0{,}1087} \text{ Y } x_2 \text{ es } \widehat{0{,}2103} \text{ ENTONCES } y = 10{,}9094 + \\
8{,}6965(x_1 - 0{,}1087) - 24{,}8071(x_2 - 0{,}2103) \\
\text{SI } x_1 \text{ es } \widehat{0{,}2043} \text{ Y } x_2 \text{ es } \widehat{0{,}8268} \text{ ENTONCES } y = 1{,}3014 + \\
1{,}9907(x_1 - 0{,}2043) - 5{,}3891(x_2 - 0{,}8268) \\
\text{SI } x_1 \text{ es } \widehat{0{,}1087} \text{ Y } x_2 \text{ es } \widehat{0{,}3702} \text{ ENTONCES } y = 6{,}6848 + \\
14{,}7067(x_1 - 0{,}1087) - 18{,}7718(x_2 - 0{,}3702) \\
\text{SI } x_1 \text{ es } \widehat{0{,}5256} \text{ Y } x_2 \text{ es } \widehat{0{,}9143} \text{ ENTONCES } y = 2{,}0615 + \\
3{,}1849(x_1 - 0{,}5256) - 8{,}8371(x_2 - 0{,}9143) \\
\text{SI } x_1 \text{ es } \widehat{0{,}9346} \text{ Y } x_2 \text{ es } \widehat{0{,}2103} \text{ ENTONCES } y = 0{,}7845 - \\
4{,}6238(x_1 - 0{,}9346) + 2{,}5731(x_2 - 0{,}2103) \\
\text{SI } x_1 \text{ es } \widehat{0{,}5256} \text{ Y } x_2 \text{ es } \widehat{0{,}2103} \text{ ENTONCES } y = 4{,}4525 - \\
11{,}8354(x_1 - 0{,}5256) + 8{,}0679(x_2 - 0{,}2103) \\
\text{SI } x_1 \text{ es } \widehat{0{,}2043} \text{ Y } x_2 \text{ es } \widehat{0{,}0936} \text{ ENTONCES } y = 11{,}0061 - \\
27{,}9078(x_1 - 0{,}2043) + 13{,}1181(x_2 - 0{,}0936) \\
\text{SI } x_1 \text{ es } \widehat{0{,}5256} \text{ Y } x_2 \text{ es } \widehat{0{,}4559} \text{ ENTONCES } y = 6{,}7246 - \\
10{,}6404(x_1 - 0{,}5256) + 3{,}4594(x_2 - 0{,}4559)
\end{array}
\tag{5.24}
$$

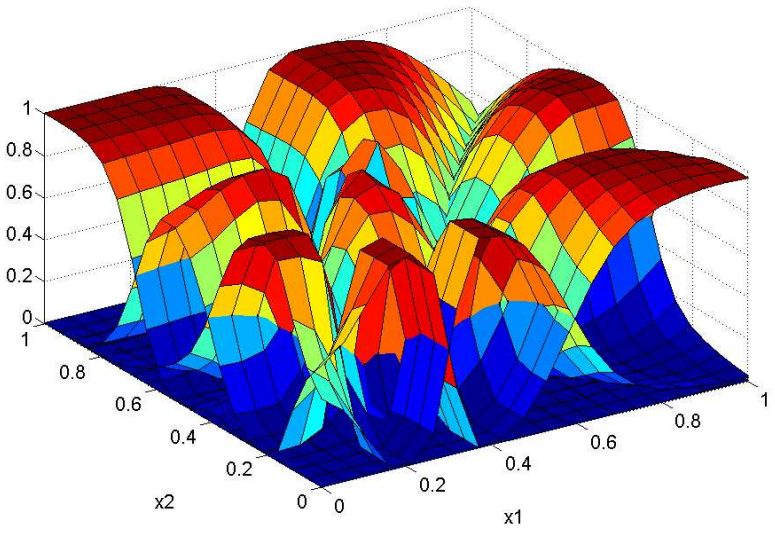


Figura 5.9. *Activaciones de las 10 neuronas en el sistema TaSe-NF óptimo interpretable obtenido para el ejemplo de la función f_3.*

5.5.2. Serie Temporal Mackey-Glass

La serie Mackey-Glass [Mackey and Glass, 1977] es un ejemplo de aplicación muy utilizado en la literatura de sistemas neuro-difusos y redes neuronales. Su descripción ya se discutió en la sección 4.6.3 (ver ecuación 4.32); aquí igualmente se generaron 500 puntos para test y 500 para training.

El algoritmo de aprendizaje del modelo TaSe-NF se lanzó para 10 centros, con $\delta = 0{,}05$. Tras el proceso de optimización, quedaron tan solo 5 centros diferentes (el resto coincidieron en el espacio de entrada tras la compartición de FPs). El obtuvo un RMSE (ver sección 4.6.3) de 0.0068 con tan solo 5 reglas. Nótese que los resultados son peores que los obtenidos con otros modelos, pero hay que recordar que la compartición de FPs hace que el rendimiento se reduzca, y que tan solo se han utilizado 5 reglas. En las cuatro variables de entrada quedan respectivamente 4, 3, 4 y 4 FPs finales. La distribución final de FPs en las variables se muestra en la figura 5.14. Como ejemplo, una de las reglas obtenidas tras el proceso de optimización es:

$$\begin{gathered}\text{SI } x(t) \text{ es } \widehat{0{,}6171} \text{ Y } x(t-6) \text{ es } \widehat{1{,}0908} \text{ Y } x(t-12) \text{ es } \widehat{1{,}1472} \\ \text{Y } x(t-18) \text{ es } \widehat{1{,}1797} \text{ ENTONCES} \\ y = 0{,}8657 - -1{,}1442(x(t) - 0{,}6171) - 1{,}1116(x(t-6) - 1{,}0908) \\ +12{,}6659(x(t-12) - 1{,}1472) - 0{,}8912(x(t-18) - 1{,}1797)\end{gathered} \tag{5.25}$$

5.6. Conclusiones

En este capítulo se ha presentado un modelo neuro-difuso modificado, denominado TaSe-NF, cuyo punto de partida es un sistema equivalente entre TSK CBFSs y redes

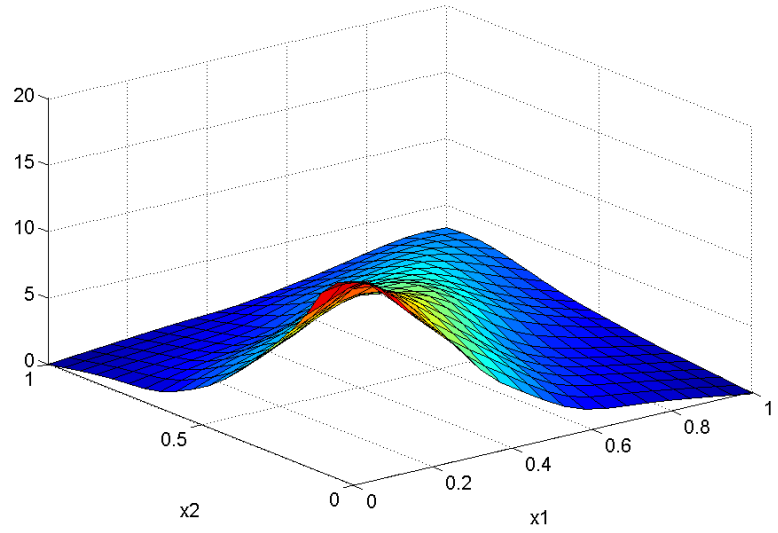


Figura 5.10. *Aproximación final obtenida para la función* f_3.

RBF, que mantiene las propiedades de optimización de los modelos locales a la vez que entrena la red optimizándola globalmente. Dichas propiedades se obtienen gracias a la inclusión de un operador de agregación modificado que proporciona un particionamiento del espacio de entrada adecuado para llevar a cabo dichos objetivos. Adicionalmente, debido a los problemas de falta de transparencia en los modelos basados en clustering, el trabajo propuesto implementa un enfoque de compartición de funciones de pertenencia, con el objetivo de mejorar la interpretabilidad del conjunto de reglas difusas extraídas del modelo. Además, para la mejora del rendimiento, se ha implementado también el enfoque de equidistribución de errores para la inicialización de centros en sistemas basados en clustering. Las propiedades de este sistema y de la metodología de aprendizaje que lleva paralela, se muestran en varios ejemplos significativos de aproximación funcional.

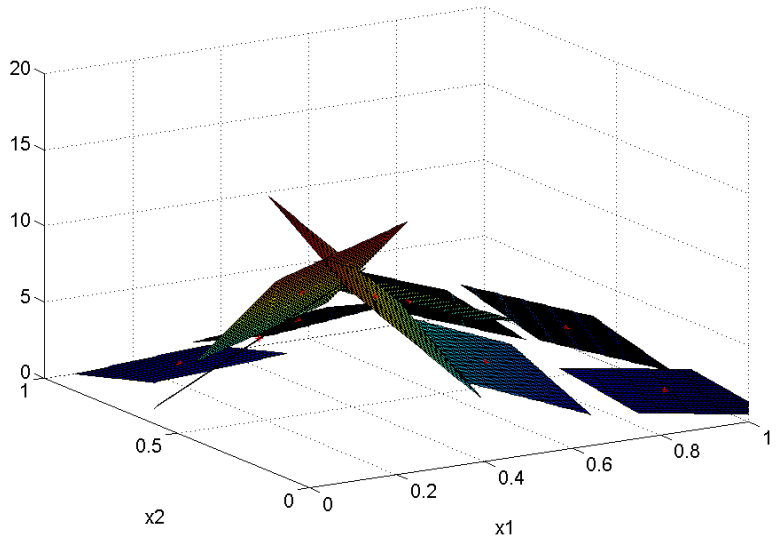


Figura 5.11. *Representación de los polinomios en los consecuentes de salida de las 10 reglas del modelo, en el área en torno al centro de cada una de ellas.*

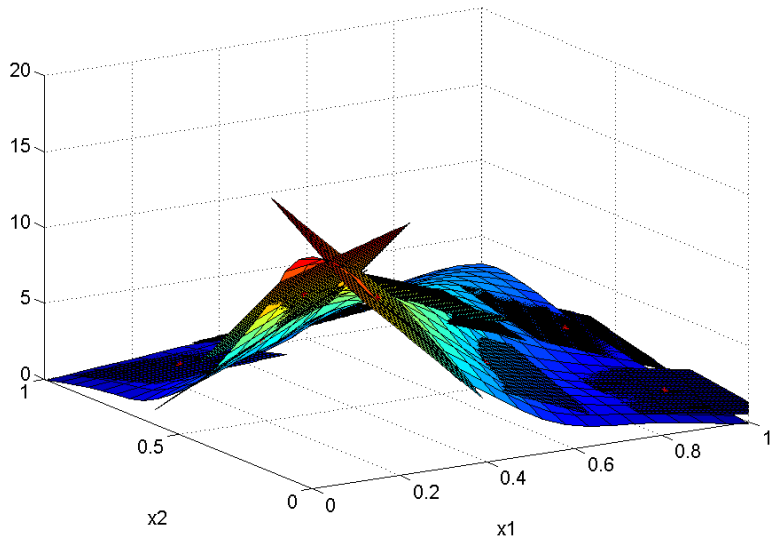


Figura 5.12. *Representación conjunta de la salida del modelo junto con los polinomios en los consecuentes de salida de las 10 reglas del modelo, en el área en torno al centro de cada una de ellas.*

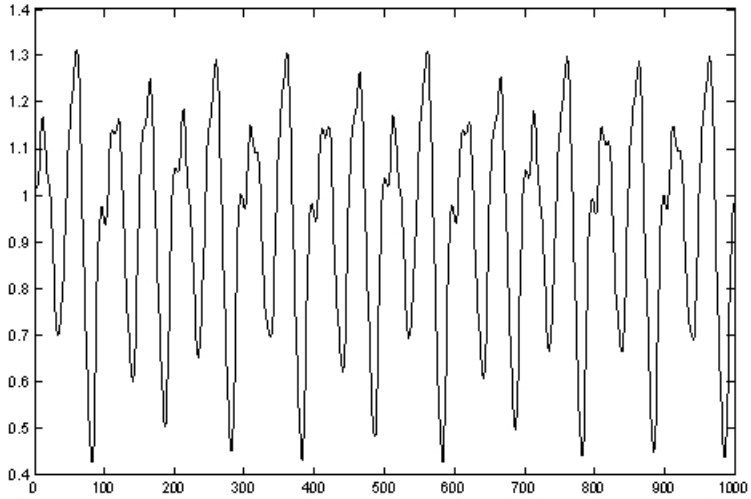


Figura 5.13. *Serie caótica de Mackey-Glass con* $\tau = 17$.

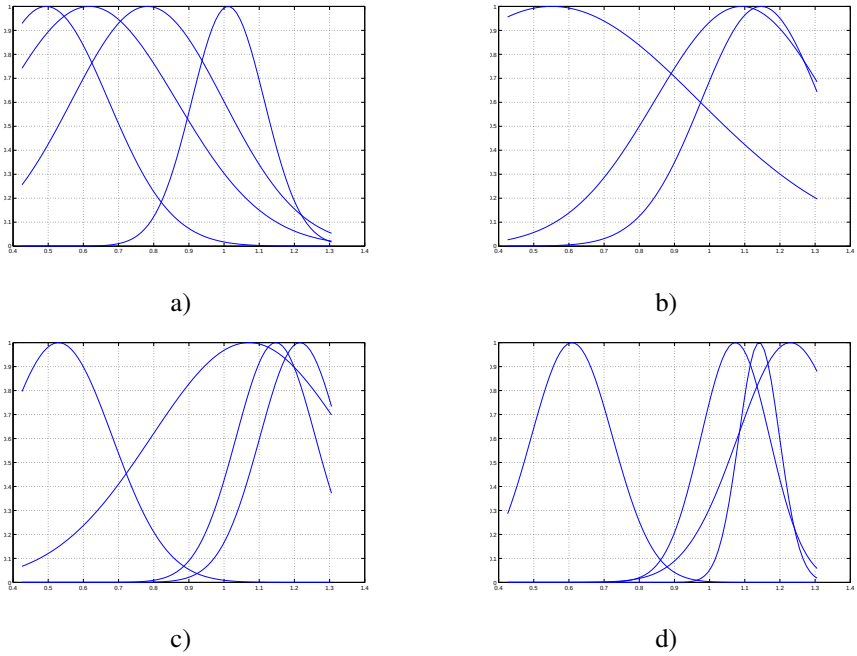


Figura 5.14. *Funciones de pertenencia resultantes para las variables a)* $x(t)$*, b)* $x(t-6)$*, c)* $x(t-12)$ *y d)* $x(t-18)$ *en el problema.*

Capítulo 6

Sistemas Difusos Multigrid

Este capítulo presenta los sistemas difusos Multigrid, como metodología alternativa para evitar la maldición de la dimensionalidad en la complejidad de los sistemas difusos basados en grid para problemas complejos. La arquitectura de los sistemas Multigrid permite reducir exponencialmente el número de reglas, mediante el descarte de relaciones irrelevantes entre las variables. Este trabajo presenta un algoritmo de selección de arquitectura para sistemas difusos Multigrid a partir de un conjunto de datos de entrada/salida, que incorpora un proceso de selección de variables. Este algoritmo de selección de arquitectura se complementa con un procedimiento de identificación de la estructura, que obtiene el particionamiento óptimo del espacio en los diferentes grids que forman el modelo Multigrid y sus parámetros (número y posición de las funciones de pertenencia y consecuentes de las reglas). Las ventajas que proporciona esta metodología se verá en la sección de simulaciones, donde cabe resaltar la aplicación al problema de predicción de series temporales CATS, propuesto inicialmente como competición en la conferencia IJCNN'2004 (International Joint Conference on Neural Networks) [Lendasse et al., 2004].

6.1. Introducción

Como se ha comentado en temas anteriores, los sistemas difusos basados en grid, tanto TSK como Mamdani, [Pomares et al., 2000], [Wang and Mendel, 1992b], presentan el problema de la maldición de la dimensionalidad (crecimiento exponencial del número de reglas) [Bellman, 1961] para problemas de complejidad moderada, especialmente para problemas de dimensionalidad media o alta.

Los sistemas difusos Multigrid, abren una puerta al tratamiento de problemas de aproximación funcional y predicción de series temporales de dimensionalidad alta mediante sistemas difusos basados en grid. Un sistema difuso Multigrid (MGFS) es un modelo difuso aditivo modificado compuesto por un conjunto de sub-grids; la salida del modelo es la media ponderada de las salidas de las reglas de los sub-grids componentes. La idea es que para un problema de dimensión del espacio de entrada n, en vez de disponer de un solo grid n-dimensional, que puede generar un número de reglas excesivamente grande, dispondremos de un número indeterminado de diferentes sub-grids, cada uno disponiendo de su propio subconjunto de variables de entrada.

En este capítulo, aparte de explicarse este tipo de sistema difuso, proporcionaremos un algoritmo para seleccionar la mejor arquitectura para un MGFS dado un conjunto de entrenamiento de E/S. Este algoritmo permite tener en cuenta cualquier número de variables

de entrada, y selecciona las interrelaciones más relevantes entre las variables de entrada para construir los sub-grids del modelo. De estar forma se pierde en gran parte el crecimiento exponencial en el número de reglas que los sistemas basados en grid presentan, pero conservándose sus ventajas. La metodología de selección de arquitectura se completa posteriormente con un procedimiento de identificación de la estructura basado en el trabajo de Pomares y otros [Pomares et al., 2002], que encontrará un particionamiento del espacio de entrada óptimo para el modelo MGFS. Se identifica el número óptimo de reglas para realizar la aproximación, situando un número adecuado de FPs en las variables de entrada correspondientes.

La metodología propuesta se aplicará a algunos ejemplos sencillos, así como al problema de predicción de series temporales CATS. La serie temporal CATS [CAT,] consta de 5000 puntos en los que faltan 100 valores. Estos valores desconocidos están distribuidos en cinco bloques de 20 puntos. La serie temporal CATS se propuso para la competición de series temporales del congreso IJCNN'2004 que tuvo lugar en Budapest [Lendasse et al., 2004]. El algoritmo de aprendizaje para MGFS propuesto se usará para obtener un conjunto de modelos de predicción directos [Yongnan et al., 2005]. Para los cuatro primeros bloques de valores desconocidos, se obtendrán un conjunto de modelos de predicción hacia adelante y hacia atrás, y para el último bloque solo se usarán modelos hacia adelante.

El resto del capítulo se organiza como sigue. La sección 6.2 introduce los conceptos básicos sobre el modelo difuso Multigrid. La sección 6.3 presenta el algoritmo de aprendizaje propuesto para MGFS. Este algoritmo incluye un primer paso de selección de arquitectura MGFS y se complementa en un segundo paso con una metodología de identificación de la estructura para la arquitectura MGFS obtenida. La sección 6.4 revisa el problema de predicción de series teporales CATS. Lase secciones 6.5-6.7 tratarán la aplicación del modelo MGFS al problema de predicción de la serie CATS, así como algunas mejoras propuestas con respecto al trabajo inicial [Herrera et al., 2004a]. Finalmente, la sección 6.8 finaliza el capítulo y trata el trabajo futuro.

6.2. Sistemas Difusos Multigrid

Recordamos que un sistema difuso tradicional basado en grid consiste en un sistema con K reglas tipo Takagi-Sugeno-Kang (TSK) [Takagi and Sugeno, 1985] según la ecuación 3.1, donde éstas están distribuidas en el espacio de entrada según un grid n-dimensional, como el que se muestra en la figura 6.1 para un ejemplo sencillo bidimensional.

Cuando tratamos con un problema complejo, con un número alto de variables de entrada, un grid n-dimensional cae en el problema de la maldición de la dimensionalidad en la complejidad del modelo, dada la ecuación 3.4 del número de reglas para un sistema basado en grid. Además, los modelos que presentan un gran número de reglas y un gran número de antecedentes por cada regla, acaban por ser incomprensibles para el diseñador [Casillas et al., 2003].

La figura 6.2 muestra la arquitectura MGFS propuesta [Herrera et al., 2004c] para tratar con espacios de entrada de alta dimensionalidad. Cada grupo de variables se utiliza para definir un sistema en grid del que se extraen reglas en la forma

$$\begin{array}{l} \text{SI } x_1^p \text{ es } \mu_1^{k_p} \text{ Y } \ldots \text{ Y } x_{n_p}^p \text{ es } \mu_{n_p}^{k_p} \\ \text{ENTONCES } y = R^{k_p} \end{array} \tag{6.1}$$

siendo R^{k_p} el consecuente de la k-ésima regla del p-ésimo GBFS. Aquí se usaran solo

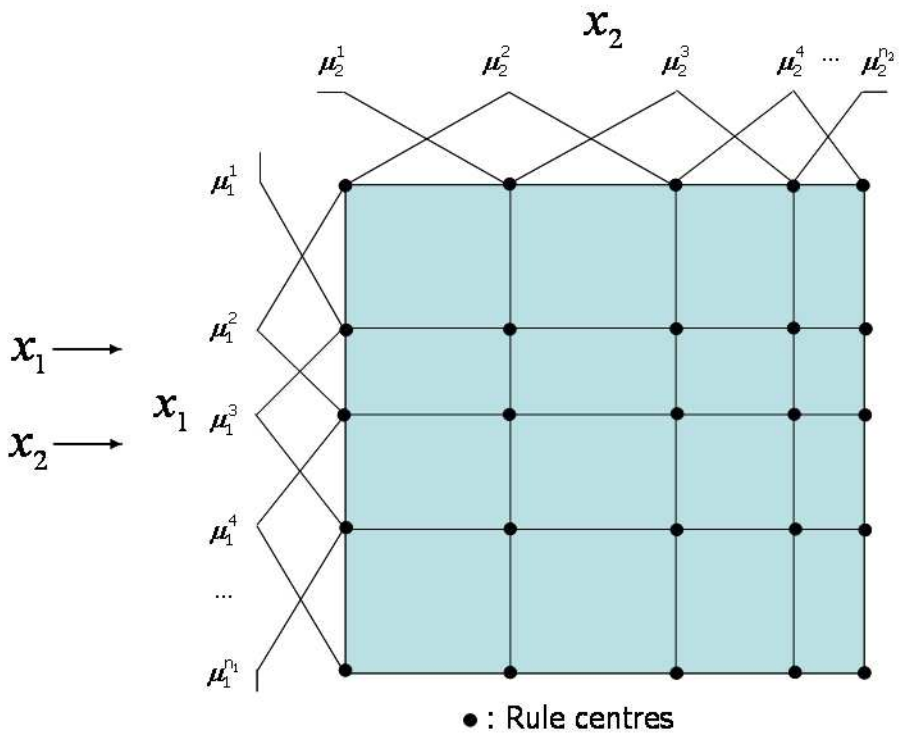


Figura 6.1. *Distribución de las reglas en un grid bidimensional en un sistema difuso basado en grid tradicional*

consecuentes de regla constantes (los lineales o de segundo orden, vistos en temas anteriores no se analizan aquí). Todas las reglas, provenientes de los sistemas en grid que forman el modelo MGFS, forman el modelo final Multigrid, cuya salida se obtiene utilizando media ponderada. Por tanto, la salida final del sistema para cualquier valor de entrada $\vec{x} = (x_1, x_2, \ldots, x_N)$, se puede expresar como

$$F(\vec{x}, MF, R, A) = \frac{\sum_{p=1}^{P} \sum_{k=1}^{K_p} R^{k_p} \prod_{m=1}^{n_p} \mu_m^{k_p}(x_m)}{\sum_{p=1}^{P} \sum_{k=1}^{K_p} \prod_{m=1}^{n_p} \mu_m^{k_p}(x_m)} \tag{6.2}$$

donde se hace mención explícita a la dependencia de la función de salida a la estructura de funciones de pertenencia *MF* del sistema, con los consecuentes del conjunto completo de K reglas R, y con la arquitectura del sistema MGFS $A = \left\{\left\{x_1^1, x_2^1, \ldots x_{n1}^1\right\}, \left\{x_1^2, x_2^2, \ldots x_{n2}^2\right\}, \ldots, \left\{x_1^P, x_2^P, \ldots x_{np}^P\right\}\right\}$. El número total K de reglas es igual a la suma de los valores $\{K_1, K_2, \ldots, K_P\}$, esto es, las reglas función de las variables de entrada que entran en cada grid individual según la estructura *MF*.

El modelo MGFS reduce la complejidad computacional del GBFS cuando se tratan problemas complejos, y es capaz de obtener un conjunto de reglas más reducido, con un número menor de antecedentes. Los sistemas MGFS por tanto abren una puerta a la ex-

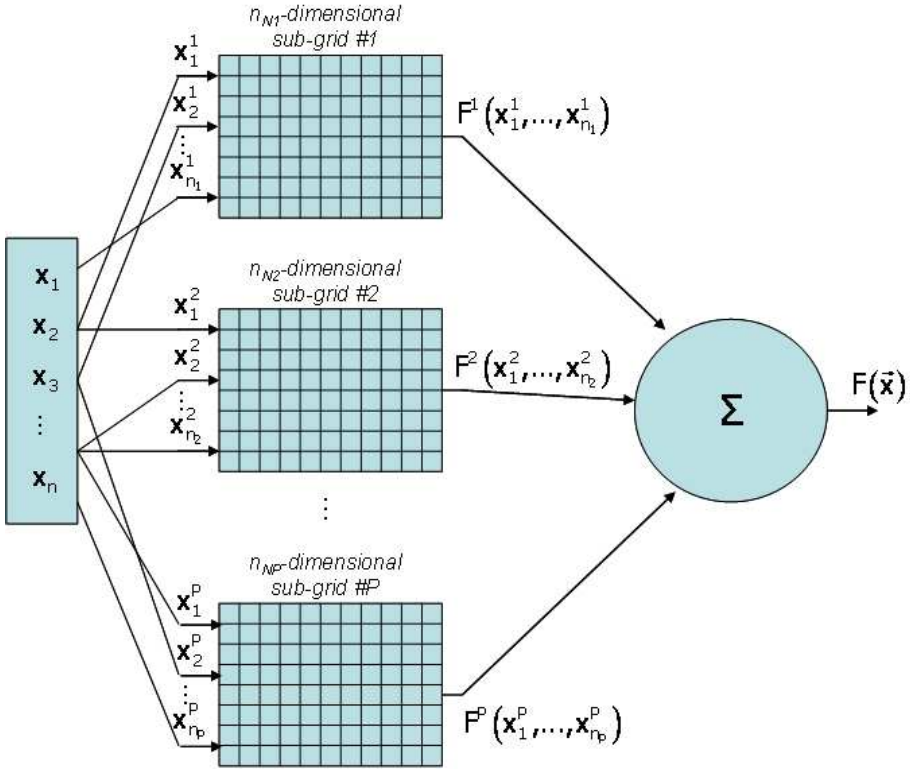


Figura 6.2. *Sistemas difusos Multigrid (MGFS)*

presión de funciones no lineales complejas como una suma de funciones más sencillas; se reduce la complejidad conceptual de problemas complejos, y se cubre el espacio de entrada mediante sistemas de particionamiento en grid. Además, gracias al método de aprendizaje propuesto, se podrán seleccionar las variables más relevantes (y sus interrelaciones) que influencian la variable de salida.

Sin embargo, los sistemas MGFS, como modelos basados en grid, pueden no ser útiles para problemas muy complejos en los que hay un gran número de variables de entrada relevantes (y relaciones complejas entre ellas. La arquitectura óptima Multigrid necesaria para obtener el modelado de los datos de E/S puede ser inviable (ver ecuación 3.4). Desde nuestro punto de vista, este sería el caso de problemas sin una solución interpretable basada en grid; para problemas con más de 10-15 variables relevantes, las posibilidades de aplicabilidad se reducen. Para estos problemas deberían usarse por tanto modelos difusos basados en clustering (CBFS, o similarmente redes RBF [Rojas et al., 2002]), u otros paradigmas como las Máquinas de Vectores Soporte [Schoelkopf and Smola, 2002], [Suykens et al., 2002] o redes neuronales artificiales [Barbounis and Theocharis, 2006].

En el caso en que se necesite un modelo interpretable, el modelo MGFS extiende la aplicabilidad de los sistemas basados en grid tradicionales. El modelo MGFS junto con el algoritmo de aprendizaje propuesto, como se verá, obtiene un modelo interpretable con altas capacidades de rendimiento y generalización.

6.3. Algoritmo de Aprendizaje para Sistemas Multigrid

Esta sección presenta un algoritmo automático y efectivo para el aprendizaje de un modelo MGFS a partir de un conjunto de datos de E/S. El aprendizaje de un sistema difuso Multigrid implica primeramente la determinación de los grupos de variables que forman cada sub-grid; este proceso se denomina selección de arquitectura. Todos los sub-grids juntos con las variables de entrada que entran en cada sub.grid forman la arquitectura del sistema, como se muestra en la figura 6.2. En segundo lugar, se puede realizar un paso de identificación de la estructura, con el objetivo de determinar el particionamiento del espacio de entrada más idóneo en cada sub-grid. Junto con la identificación de la estructura, debe realizarse también una fase de ajuste de parámetros (encontrar los parámetros óptimos de las FPs) [Pomares et al., 2002].

Esta sección se estructura de la siguiente forma: la subsección 6.3.1 presenta el algoritmo de selección de arquitectura propuesto. Posteriormente, la subsección 6.3.2 introduce un algoritmo complementario para identificación de estructura y ajuste de parámetros para MGFSs. Finalmente se presentan dos ejemplos de aplicación significativos en las subsecciones 3.3 y 3.4. La subsección 3.5 finalmente trata algunos aspectos sobre la metodología propuesta.

6.3.1. Selección de arquitectura para MGFS

La obtención de una arquitectura óptima es una tarea muy compleja, dada la naturaleza del problema. En principio habría que evaluar todas las posibles arquitecturas, sin embargo esto es inviable; el número de arquitecturas posibles para un número de variables de entrada dado es un número combinatorio. Esta sección presenta un algoritmo incremental que evita un escaneo completo del espacio de búsqueda, pero que obtiene una arquitectura con un buen rendimiento dado un problema con un conjunto de datos de E/S.

En el trabajo [Herrera et al., 2004c], se presentó un algoritmo con funcionamiento arriba-abajo, esto es, que a partir de un GBFS completo (es decir, un MGFS con un solo sub-grid n-dimensional) descartaba estructuras más complejas en favor de otras más sencillas, mientras se mantenía un umbral de error, y manteniendo fijo el número de FPs por variable. Este enfoque, aunque es muy potente para descubrir relaciones intrínsecas entre las variables de entrada, tiene una desventaja muy importante que lo hace inapropiado para problemas complejos: el punto de partida es un sistema GBFS completo, que puede ser computacionalmente demasiado caro cuando se dispone de un problema con un número alto de variables de entrada (ver ecuación 3.4). Este enfoque es por tanto inviable para problemas de complejidad media o alta.

Aquí se presenta un enfoque de abajo-arriba, tratado por primera vez en el trabajo presentado en [Herrera et al., 2004d]. Se comienza con la arquitectura más sencilla posible, que dispone de un sub-grid por variable de entrada, y realiza una búsqueda en anchura en complejidad de grupos de variables. La idea de base de esta segunda alternativa es que en vez de buscar conservar un rendimiento y buscar una arquitectura lo más sencilla posible, en este enfoque, se pretende conservar un límite de complejidad aceptable (número de parámetros a optimizar = número de reglas) mientras se busca por el sistema que se comporta mejor, dado este número de reglas. El límite en el número de reglas es un parámetro importante del algoritmo, y su obtención se discutirá después en esta subsección.

Con el objetivo de comparar diferentes arquitecturas MGFS a través del algoritmo incremental propuesto, para un límite en el número de parámetros dado, se usará la siguiente heurística (en ausencia de información adicional específica sobre el problema concreto) con

el objetivo de conservar la simplicidad y eficiencia del enfoque: las reglas se distribuirán homogéneamente entre los sub-grids, y dentro de cada sub-grid, el número de FPs por variable se distribuirá homogéneamente de forma que se conserve aproximadamente el número de reglas asignadas. Nótese que este límite en el número de reglas puede no ajustar el número de reglas totales asignadas a todos los sub-grids según esta heurística; sin embargo como se ha mencionado, el algoritmo intentará obtener la mejor distribución de reglas y FPs para conservar dicho número de reglas totales. Notamos aquí que los consecuentes de las reglas se obtendrán de forma óptima utilizando el enfoque de mínimos cuadrados. El modelo MGFS usará aquí FPs en forma de partición triangular [Pomares et al., 2000], de forma que los consecuentes pueden obtenerse mediante métodos lineales [Herrera et al., 2004c], [Pomares et al., 2000].

El algoritmo funciona de manera greedy (ver figura 6.3). Comienza con la arquitectura sencilla que tiene n sub-grids con una variable cada uno, donde n es el número inicial de variables de entrada. Secuencialmente probará el rendimiento del problema cuando se añade una variable a un sub-grid, pero sin testear más allá con sub-grids de orden $m+1$ hasta que todas las combinaciones de orden m se hayan explorado. De entre todas las posibilidades de añadir una variable a un sub-grid (formándose un sub-grid de orden m), el algoritmo selecciona en cada caso aquel que reduzca más el error de entrenamiento. Este procedimiento se repite hasta que el error no se decremente, y entonces explorará sub-grids de orden $m+1$.

El número fijo de reglas no se conserva permanentemente, distribuyendo las reglas y FPs como se ha comentado anteriormente. Esto es, el añadir una variable a un sub-grid implica la reducción en el número de FPs por variable en ese sub-grid. La distribución de reglas propuesta, lleva a cabo un intercambio entre alta dimensionalidad y un número menor de FPs por variable en cada sub-grid concreto. Esto implica que tan solo las interrelaciones significativas entre variables reflejarán una reducción en el error de entrenamiento. La arquitectura MGFS que proporcionará el menor error de entrenamiento dado sobre el conjunto de datos D será el candidato escogido.

De acuerdo con la figura 6.3, tras la inicialización del modelo, se realiza un procedimiento de eliminación de sub-grids mediante la técnica de dejar-uno-fuera (leave-one-out). De esta forma se comprueba que todos los sub-grids iniciales son relevantes para el problema concreto de modelado. Si dejando un sub-grid fuera del modelo, nos deja un error de entrenamiento similar o mejor, entonces este sub-grid puede eliminarse de la arquitectura. Resaltamos aquí que la eliminación de un sub-grid no relevante, debido a la distribución de reglas que se utiliza, implica que otros sub-grids que sí son relevantes obtendrán más FPs. Esto podría implicar incluso un incremento en el rendimiento. La fase de dejar-uno-fuera asegura que se eliminan variables irrelevantes y redundantes del modelo, descartándose sus respectivos sub-grids de una sola variable. De esta forma, el espacio de búsqueda se limita todavía más, con un riesgo bajo (las variables que en principio no proporcionan información por si solas son eliminadas, aunque sin embargo, podrían recuperarse posteriormente en las extensiones de los sub-grids), pero con una importante reducción en el coste computacional.

El límite en el número de reglas para el que se seleccionará la mejor arquitectura, se debe seleccionar teniendo en cuenta el intercambio entre la interpretabilidad y la precisión del sistema que se desea diseñar, el número de puntos de entrenamiento de los que disponemos (considerando que normalmente no es conveniente tener un número mayor de parámetros que de datos), el número de variables de entrada, y el coste computacional del algoritmo. Para obtener un valor conveniente y preciso, proponemos la utilización de validación cruzada, con el objetivo de ajustar el intercambio entre precisión y generalización de la arquitectura a seleccionar. Así, los datos de entrenamiento se subdividirán en un número

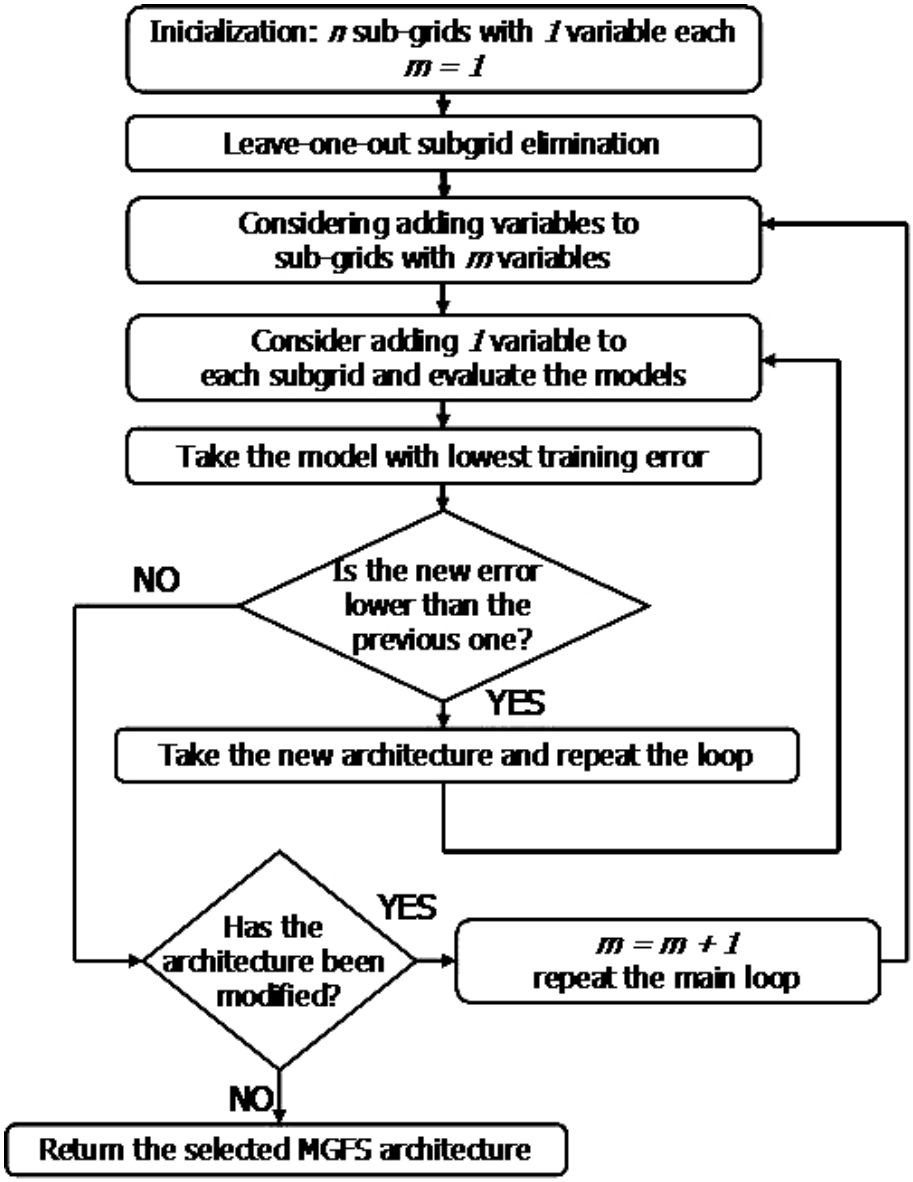


Figura 6.3. *Algoritmo de Selección de Arquitectura para MGFS*

de bloques. Para un límite candidato, el algoritmo de selección de arquitectura (ver figura 6.3) se lanzará usando todos los bloques de datos menos uno, que se utilizará para evaluar la arquitectura obtenida utilizando la distribución de reglas obtenida. Este proceso se repite para todos los bloques, dejando uno fuera cada vez, y el error de validación medio se toma como criterio para la selección del límite óptimo en el número de reglas. Este procedimiento se ejecuta utilizando un rango de valores candidatos para dicho límite. El límite seleccionado será aquel que proporcione un error de validación medio menor.

La arquitectura seleccionada serla la que se obtenga al lanzar el algoritmo de selec-

ción de arquitectura utilizando el conjunto completo de entrenamiento, para el límite en el número de reglas seleccionado. Debido a que es un procedimiento de optimización no lineal, y a que en el procedimiento se han utilizado heurísticas debido a su complejidad, no podemos asegurar que la arquitectura obtenida sea óptima en todos los casos. Pero debido a las características del funcionamiento del algoritmo propuesto, si podemos decir que será una arquitectura sub-óptima.

Una vez que se selecciona una arquitectura, es interesante el determinar el número óptimo de FPs en cada variable de entrada, con los parámetros de las FPs también optimizados. La distribución y colocación de las reglas y FPs utilizados en el algoritmo de selección de arquitectura puede ser ineficiente en muchos casos, llevando a un rendimiento deficiente del sistema. Así pues proponemos ahora un procedimiento de optimización en la siguiente subsección.

6.3.2. Optimización de un modelo MGFS con una arquitectura proporcionada

Esta subsección trata sobre la búsqueda de un particionamiento adecuado del espacio de entrada para realizar el modelado de E/S para una arquitectura dada. Una distribución correcta de las FPs en las variables de entrada ayuda a controlar el intercambio entre las capacidades de precisión y generalización del modelo. Para un sistema Multigrid dado, esto implica particionar el espacio de entrada en cada uno de los sub-grids, de forma que el modelo global rinda óptimamente. El proceso de optimización de un MGFS con una arquitectura dada, se divide en dos sub-procesos intercalados, identificación de la estructura y ajuste de parámetros (ver figura 6.4).

Identificación de la estructura

La identificación de estructura plantea la búsqueda del número óptimo de FPs por variable de entrada, o, dicho de otra forma, la búsqueda del número óptimo de reglas para el modelo. El algoritmo para sistemas Multigrid propuesto está basado en la metodología presentada para sistemas basados en grid en [Pomares et al., 2002], y que también se utilizó para el sistema TaSe en capítulos anteriores. Se comienza con la estructura más sencilla posible, que consta de una FP por variable. Entonces comienza un proceso iterativo que a partir de la estructura más simple posible con una FP por variable de entrada, hasta que se alcanza un umbral concreto, o cuandoquiera que el error de validación deja de decrementarse.

La variable en la que se añade la función de pertenencia, se selecciona evaluando la adición de l FPs a cada variable en cada sub-grid del MGFS. La variable que mostró que añadiendo las l FPs adicionales reduce más el error de entrenamiento se selecciona para añadirle 1 FP eventualmente. El valor l es proporcional (un 200 % o un 150 %) al mayor número de FPs en una variable en el paso actual (ver [Pomares et al., 2002] para más detalles), y se utiliza para estimar la necesidad de una variable de tener más FPs para mejorar el rendimiento global. Una vez que la variable se selecciona y la nueva FP se añade, se realiza validación cruzada para comprobar si la nueva adición de una FP mejora la generalización y eficacia del modelo. En cada una de las iteraciones del procedimiento de validación cruzada se realiza la fase de ajuste de parámetros (que se explica en la siguiente subsección. La estructura final Multigrid es aquella para la que el error de validación cruzada fue menor. Los parámetros de las funciones de pertenencia del modelo (con la arquitectura y estructura seleccionadas) pueden optimizarse finalmente utilizando la totalidad de los datos de entrenamiento para obtener el modelo definitivo.

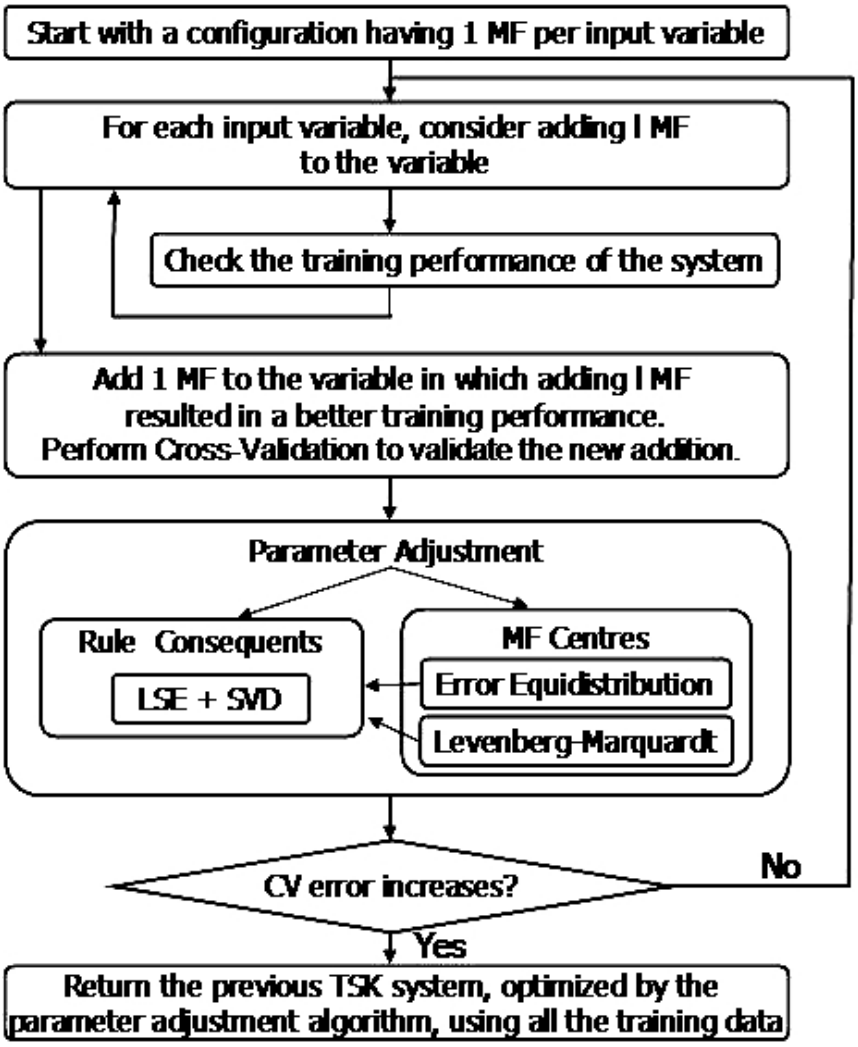


Figura 6.4. *Optimización de un MGFS*

Se debe resaltar que una variable de entrada x_i puede pertenecer a varios sub-grids diferentes en la arquitectura Multigrid seleccionada. Cada variable de entrada x_i puede tener una representación diferente de su dominio para cada sub-grid en el que esté presente. Sin embargo, para el procedimiento de optimización realizado aquí, cada variable x_i, aunque esté presente en más de un sub-grid, se considerará como una sola variable. La razón de este enfoque es que de esta forma se conserva mejor la interpretabilidad del modelo, al disponer de un solo conjunto de FPs por cada variable. La flexibilidad del modelo podría ser mayor si para dichas variables de entrada se considerasen las múltiples posibles representaciones de su dominio. Además la metodología propuesta es fácilmente modificable para tener en cuenta ese extremo, sin embargo, se ha dado más importancia al hecho de conservar la interpretabilidad del modelo, disponiendo de una relación uno a uno entre variables reales del problema y variables del modelo, cada una con un solo particionamiento

en FPs.

Ajuste de parámetros

Referimos como ajuste de parámetros a la optimización de la localización de las funciones de pertenencia y los consecuentes de las reglas. Los consecuentes de las reglas de los diferentes sub-grids, se pueden optimizar utilizando el enfoque de mínimos cuadrados, al ser la salida del modelo, la suma ponderada de las salidas de todas las reglas del sistema. Así pues, de nuevo mínimos cuadrados llevan a un sistema de ecuaciones lineales que pueden resolverse utilizando cualquier método matemático para ese objetivo. Utilizaremos la descomposición de valores singulares, debido a su eficiencia y a que puede eliminar coeficientes innecesarios [Herrera et al., 2005f].

Con respecto al ajuste de los parámetros de las FPs, debido a que en cada variable se utiliza una partición de FPs triangulares (ver la figura 6.1), solo los centros de las FPs no situadas en los extremos de cada dominio necesitan ser ajustados. El ajuste de parámetros es un paso necesario, puesto que hay algunos casos en los que hay una mayor variabilidad de la función a aproximar en algunas zonas del espacio que en otras; por tanto se deberían situar más FPs en aquellas áreas [Gonzalez et al., 2002]. Este trabajo utiliza un enfoque basado en gradiente para situar los centros de las FPs de cada variable del MGFS. Se utilizará el algoritmo de Levenberg-Marquardt [More, 1978] debido a su robustez y a su eficiencia computacional. Con el objetivo de evitar caer en mínimos locales se utilizará un procedimiento inicial de situación de los centros, adaptado de la Equidistribución de Errores [Pomares et al., 2000] para los sistemas Multigrid. Así pues la utilización de la inicialización de centros mediante la Equidistribución de Errores y el algoritmo de Levenberg-Marquardt garantiza que se encontrará un buen mínimo local para la distribución de los centros de las FPs del sistema.

6.3.3. Ejemplo 1

Esta subsección presenta una aplicación de la metodología completa de aprendizaje para sistemas difusos Multigrid a un ejemplo 9-dimensional cuyos datos de entrada/salida se sacan de la siguiente expresión

$$F(x_1, \ldots, x_9) = \sin(\pi x_1 x_2) + \exp x_3 x_4 + 2x_5^2 + 2x_6 x_7 x_8 + 0x_9 + \xi \qquad (6.3)$$

donde $x_1, \ldots, x_9 \in [0, 1]$, y ξ es un ruido aditivo gaussiano con desviación estándar 0.1. En este ejemplo artificial, las variables se agrupan en tres asociaciones bien diferenciadas, que el algoritmo de aprendizaje para MGFS debería identificar como sub-grids. Es un ejemplo artificial sencillo, pero ayudará a comprender y verificar el buen funcionamiento del algoritmo de aprendizaje propuesto. Nótese que para las condiciones de los problemas dadas, un modelo tradicional basado en grid no puede aplicarse (utilizando consecuentes constantes como se comentó); tomando por ejemplo tres MFs por variable, resultaría en un número inmanejable de reglas según la expresión dada en la ecuación 3.4.

Para el ejemplo propuesto, se generaron 1000 datos. Se usaron 800 para el entrenamiento y los restantes 200 para testar la metodología de aprendizaje. Primeramente se lanzó el algoritmo de selección de arquitectura. El límite en el número de reglas del modelo se seleccionó mediante validación cruzada con 5 bloques sobre el conjunto de entrenamiento. La figura 6.5 muestra los valores medios de la validación cruzada en la selección del valor óptimo para el límite de reglas.

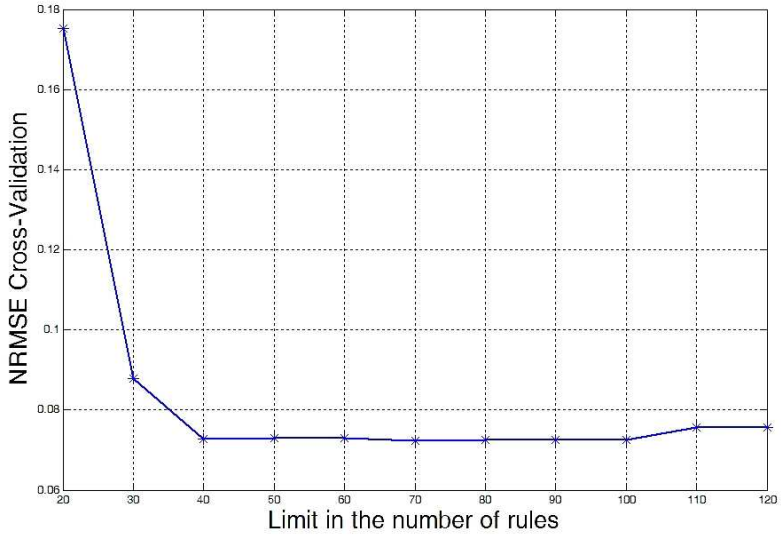


Figura 6.5. *Errores de validación cruzada para la selección del límite en el número de reglas para el algoritmo de selección de arquitectura en el ejemplo 1*

El límite seleccionado según esta gráfica es de 70, que es el valor para el que se obtiene un valor medio de error de validación cruzada menor. Sin embargo, para este ejemplo particular, podemos observar que el error de validación cruzada está estabilizado. Esto significa que hay un amplio rango de valores de número de reglas, para el que la selección de arquitectura es de esperar que obtenga un resultado óptimo. La tabla 6.1 muestra la evolución del algoritmo de selección de arquitectura utilizando 70 reglas con las 800 muestras de entrenamiento (utilizando como medida del error el error cuadrático medio normalizado, NRMSE [Pomares et al., 2000]). Obsérvese como primeramente el proceso de eliminación de sub-grids deja-uno-fuera, elimina eventualmente la variable x_9 del modelo. Posteriormente, se evalúan los sub-grids de dos variables que se pueden formar en el modelo. Nótese que cuando un sub-grid más grande se añade al modelo, los sub-grids que puedan ser subsumidos por éste se eliminan eventualmente del modelo.

Posteriormente se evalúan los sub-grids de tres variables, partiendo de la adición de una nueva variable a los sub-grids de dos variables integrados en la arquitectura. Después, se evalúan los sub-grids de cuatro variables; el algoritmo se detiene puesto que no se detectan más interrelaciones relevantes entre las variables.

Posteriormente se ejecuta el procedimiento de identificación de la estructura. La tabla 6.2 muestra la ejecución del proceso iterativo. Como se explica en la sección 6.3.2, cuando el número de centros en una variable es mayor que 2, el algoritmo de Levenberg-Marquardt (con el proceso previo de Equidistribución de Errores) localiza óptimamente la posición de los centros de las FPs intermedias. La estructura óptima encontrada por el procedimiento de optimización es $\{\{3,3\},\{5,4\},\{2,2,2\},\{5\}\}$, con un total de 42 reglas.

Así pues, el sistema difuso Multigrid devuelto por el algoritmo de aprendizaje para el ejemplo propuesto tiene 4 sub-grids, dos de dos variables ($\{x_3,x_4\},\{x_1,x_2\}$), uno de tres ($\{x_6,x_7,x_8\}$) y uno de una ($\{x_5\}$), con la distribución de funciones de pertenencia correspondiente, y posicionamiento de los centros de las FPs obtenida mediante el algoritmo

Tabla 6.1. *Ejecución del algoritmo de selección de arquitectura para MGFS para el ejemplo 1*

Arquitectura	Distribución de FPs	NRMSE de entre.
$\{\{x_1\},\{x_2\},\{x_3\},\{x_4\},\{x_5\},\{x_6\},\{x_7\},\{x_8\},\{x_9\}\}$	$\{\{8\},\{8\},\{8\},\{8\},\{8\},\{8\},\{8\},\{8\},\{8\}\}$	0.278
Tras el paso de eliminación deja-uno-fuera de sub-grids		
$\{\{x_1\},\{x_2\},\{x_3\},\{x_4\},\{x_5\},\{x_6\},\{x_7\},\{x_8\}\}$	$\{\{9\},\{9\},\{9\},\{9\},\{9\},\{9\},\{9\},\{9\}\}$	0.278
Búsqueda de sub-grids de dos variables		
$\{\{x_3,x_4\},\{x_1\},\{x_2\},\{x_5\},\{x_6\},\{x_7\},\{x_8\}\}$	$\{\{3,3\},\{10\},\{10\},\{10\},\{10\},\{10\},\{10\}\}$	0.209
$\{\{x_3,x_4\},\{x_1,x_2\},\{x_5\},\{x_6\},\{x_7\},\{x_8\}\}$	$\{\{3,3\},\{3,3\},\{12\},\{12\},\{12\},\{12\}\}$	0.137
$\{\{x_3,x_4\},\{x_1,x_2\},\{x_6,x_8\},\{x_5\},\{x_7\}\}$	$\{\{4,4\},\{4,4\},\{4,4\},\{14\},\{14\}\}$	0.113
$\{\{x_3,x_4\},\{x_1,x_2\},\{x_6,x_8\},\{x_6,x_7\},\{x_5\}\}$	$\{\{4,4\},\{4,4\},\{4,4\},\{4,4\},\{14\}\}$	0.0992
Búsqueda de sub-grids de tres variables		
$\{\{x_3,x_4\},\{x_1,x_2\},\{x_6,x_7,x_8\},\{x_5\}\}$	$\{\{4,4\},\{4,4\},\{3,3,3\},\{18\}\}$	0.0784

de optimización. El error de test obtenido con esta configuración del sistema Multigrid es de NRMSE = 0.0737.

Tabla 6.2. *Evolución del algoritmo de identificación de la estructura para la arquitectura* $\{\{x_3,x_4\},\{x_1,x_2\},\{x_6,x_7,x_8\},\{x_5\}\}$ *(ver la figura 6.3)*

Distribución de FPs en el MGFS	Número de reglas	NRMSE de validación cruzada
$\{\{1,1\},\{1,1\},\{1,1,1\},\{1\}\}$	4	1
$\{\{1,1\},\{1,1\},\{1,1,1\},\{2\}\}$	5	0,665
$\{\{1,1\},\{2,1\},\{1,1,1\},\{2\}\}$	6	0,600
$\{\{1,1\},\{2,2\},\{1,1,1\},\{2\}\}$	8	0,549
$\{\{1,2\},\{2,2\},\{1,1,1\},\{2\}\}$	9	0,481
$\{\{2,2\},\{2,2\},\{1,1,1\},\{2\}\}$	11	0,371
$\{\{2,2\},\{2,2\},\{1,1,1\},\{3\}\}$	12	0,323
$\{\{2,2\},\{3,2\},\{1,1,1\},\{3\}\}$	14	0,275
$\{\{2,2\},\{3,3\},\{1,1,1\},\{3\}\}$	17	0,220
$\{\{2,2\},\{3,3\},\{1,2,1\},\{3\}\}$	18	0,191
$\{\{2,2\},\{3,3\},\{2,2,1\},\{3\}\}$	20	0,166
$\{\{2,2\},\{3,3\},\{2,2,2\},\{3\}\}$	24	0,113
$\{\{2,2\},\{3,3\},\{2,2,2\},\{4\}\}$	25	0,104
$\{\{2,2\},\{4,3\},\{2,2,2\},\{4\}\}$	28	0,0985
$\{\{2,2\},\{4,4\},\{2,2,2\},\{4\}\}$	32	0,0937
$\{\{2,3\},\{4,4\},\{2,2,2\},\{4\}\}$	34	0,0881
$\{\{3,3\},\{4,4\},\{2,2,2\},\{4\}\}$	37	0,0822
$\{\{3,3\},\{4,4\},\{2,2,2\},\{5\}\}$	38	0,0785
$\{\{3,3\},\{5,4\},\{2,2,2\},\{5\}\}$	42	0,0795

6.3.4. Ejemplo 2

La serie Mackey-Glass es bien conocida como problema de prueba de diferentes metodologías. Es una serie artificial sin ruido que se ha usado ampliamente en la literatura de sistemas neuro-difusos y otros modelos no lineales como comparativa. Esta serie temporal se describe por la siguiente ecuación diferencial con retardo

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx(t) \tag{6.4}$$

Se genero una secuencia de 1200 muestras con la condiciones inicial $x(0) = 1{,}2$ y $\tau = 17$, usando el método de Runge-Kutta de 4º orden. De los 1200 puntos, se tomaron los últimos 1000, usando 500 para entrenamiento y los últimos 500 para test. Para hacer

Tabla 6.3. *Resultados de comparación del error de predicción de diferentes métodos para predicción de la serie Mackey-Glass con horizonte de predicción igual a 6. Las referencias se han tomado de [Rojas et al., 2002]*

Método		RMSE de test
Auto Regressive Model		0.19
Cascade Correlation NN		0.06
Back-Prop. NN		0.02
6th-order Polynomial		0.04
Linear Predictive Method		0.55
Kim and Kim (Genetic Algorithm and Fuzzy System)	5 MFs	0.049
	7 MFs	0.042
	9 MFs	0.038
Classical RBF with 23 neurons ($\simeq$ 200 params)		0.0114
PG-RBF ($\simeq$ 200 params)		0.0030
MGFS 3 sub-grids with 120 rules (136 params)		0.0031
GBFS with 192 rules (201 params)		0.0041
Neuro-fuzzy network [Paiva and Dourado, 2001] 9 rules (92 params)		0.0239

las comparaciones con otros trabajos previos justas, se escogieron los parámetros de forma que los vectores de entrenamiento del modelo tienen el siguiente formato

$$[x\,(t-18)\,,\ x\,(t-12)\,,\ x\,(t-6)\,,\ x\,(t)\,;\ x\,(t+6)] \tag{6.5}$$

Para este problema, al contrario que para el ejemplo anterior, se desconoce el agrupamiento óptimo de variables de entrada que debería formar la arquitectura del modelo MGFS. Utilizando validación cruzada, se seleccionó el número óptimo de reglas que debe de utilizar el algoritmo de selección de arquitectura. El límite seleccionado fue 190. Para este límite, se lanzó el algoritmo utilizando todos los puntos de entrenamiento. Empezando por la configuración más sencilla posible que tiene cuatro sub-grids con una variable cada uno. El algoritmo alcanzó una arquitectura sub-óptima con tres sub-grids de dos variables cada uno, descartándose cualquier otra estructura más compleja que ésta. Los sub-grids que forman la arquitectura MGFS seleccionada son $\{x(t-12), x(t)\}$, $\{x(t-12), x(t-6)\}$, $\{x(t-12), x(t-18)\}$, cada uno disponiendo de una configuración de FPs de 8×8. La raíz del error cuadrático medio (RMSE [Rojas et al., 2002], que es la medida utilizada normalmente para este ejemplo en la literatura) obtenida con esta arquitectura es 0.0039. La identificación de la estructura mejoró ligeramente el rendimiento del modelo inicial hasta un RMSE de 0.0033, con una estructura $\{\{6,6\}, \{6,6\}, \{6,8\}$. La tabla 6.3 muestra una comparativa de la metodología MGFS con otros paradigmas y metodologías que incorporan soluciones de lógica difusa presentes en la literatura. Obsérvese como el modelo tradicional basado en grid, obtiene un sistema óptimo que tiene un número mayor de parámetros (con una estructura de funciones de pertenencia igual a $\{2,3,8,4\}$) y aún obtiene un rendimiento peor que el modelo MGFS propuesto para este ejemplo.

6.3.5. Notas sobre la efectividad y eficiencia computacional del algoritmo de aprendizaje

El modelo MGFS como se ha visto presenta un buen rendimiento con ventajas adicionales sobre los sistemas basados en grid tradicionales. Primeramente el número de reglas para un rendimiento similar puede ser mucho menor, y con un número menor de antecedentes. Además, desde el punto de vista de la interpretabilidad, un dato, para el que se espera obtener la salida, necesita realizar la media ponderada de la salida de 2^n reglas cuando se usan los modelos GBFS. Utilizando MGFS, esto se traduce en un número menor de reglas igual a $\sum_{p=1}^{P} 2^{n_p}$ (donde n_p es el número de variables de entrada en el sub-grid p); además las reglas en MGFS tienen un número menor de antecedentes, lo que facilita la interpretabilidad, desde ese punto de vista.

Con respecto al coste computacional, dada una cierta arquitectura, la metodología MGFS presenta un cost de entrenamiento similar en comparación con lo que un GBFS tradicional conllevaría. Hay que obtener un conjunto fijo de FPs por variable de entrada. Esto implica encontrar el número óptimo de FPs por variable y sus parámetros óptimos. Los parámetros lineales del modelo (consecuentes de las reglas) se obtienen de forma similar también. Sin embargo el numero de reglas en un MGFS puede ser mucho menor que en un GBFS para un objetivo de rendimiento similar como se ha visto.

Adicionalmente, hay que realizar un proceso de selección de arquitectura cuando se entrena un sistema Multigrid. Es un problema muy complejo, pero gracias a la metodología incremental propuesta, el espacio de búsqueda se reduce considerablemente. Además gracias al control de la búsqueda utilizando un límite en el número de reglas estimadas para el modelo, se evitan cuellos de botella computacionales. El número total de reglas se divide automáticamente entre los sub-grids, y el entrenamiento de cada arquitectura eventual para esta evaluación se hace mediante mínimos cuadrados (resolviendo un sistema de ecuaciones lineales). Como se ha comentado, este límite en el número de reglas se obtiene utilizando validación cruzada, con el objetivo de ajustar el intercambio entre precisión y capacidad de generalización del modelo conforme se aumenta dicho límite. Los esfuerzos computacionales para el procedimiento de selección de arquitectura, dependen mayormente en el número de muestras de entrenamiento usadas y en el número de variables de entrada relevantes.

El control del sobre-ajuste en la metodología propuesta es doble. Por un lado, la identificación del número de reglas a utilizar por el algoritmo de selección de arquitectura se realiza utilizando validación cruzada. Este límite se usa para obtener la arquitectura más apropiada para el conjunto de datos de E/S. Por otro lado, la estructura óptima del modelo se calcula, dada la arquitectura del modelo, utilizando un proceso incremental, cuyo criterio de parada depende del decremento del error de validación cruzada. Esta metodología doble asegura la eficacia y la capacidad de generalización del modelo final obtenido.

Ahora, el resto del artículo trata sobre el problema de predicción en serie temporal CATS, y su solución utilizando un conjunto de modelos de predicción directos MGFS.

6.4. Predicción de la Serie Temporal CATS

6.4.1. La serie temporal CATS

La serie temporal CATS consta de 5000 puntos de datos en los que 100 de estos valores son desconocidos. Estos valores que faltan se dividen en cinco bloques

1. - elementos 981 to 1,000;

2. - elementos 1,981 to 2,000;
3. - elementos 2,981 to 3,000;
4. - elementos 3,981 to 4,000;
5. - elementos 4,981 to 5,000;

La tarea en este problema es predecir estos 100 valore, y la calidad de la predicción se evalúa mediante la medida del error cuadrático medio E en los 100 valores utilizando la fórmula

$$E = \frac{\sum_{m=0}^{4} \sum_{t=1000*m+981}^{1000*(m+1)} (y_t - \hat{y}_t)^2}{100} \tag{6.6}$$

Una representación completa de estos 5000 datos, que se proporcionó tras la competición en el IJCNN'2004 [Lendasse et al., 2004], se muestra en la figura 6.6. Una segunda secuencia de la serie temporal con 5000 nuevos elementos y cinco nuevos huecos que predecir, proporcionada tras ese encuentro, se muestra en la figura 6.7.

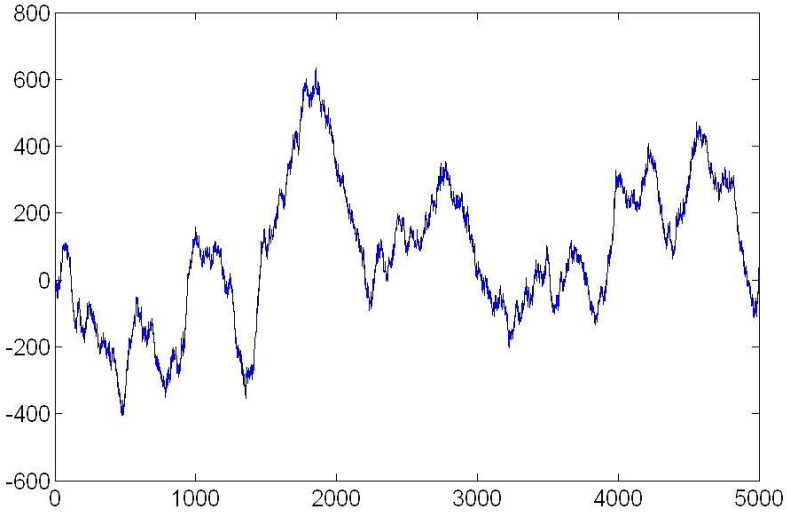


Figura 6.6. *Primera secuencia de 5000 datos de la serie temporal CATS, incluyendo la solución de los cinco huecos*

6.4.2. Análisis de la serie

Para este problema de predicción de la serie temporal CATS, se ha proporcionado una sola serie unidimensional de datos. No hay información adicional sobre el modelo de datos subyacente que produce la serie.

Echando un primer vistazo a la serie original, se observa que en principio, no hay ningún patrón repetitivo. Los datos siguen una cierta tendencia con ruido, pero sin un patrón

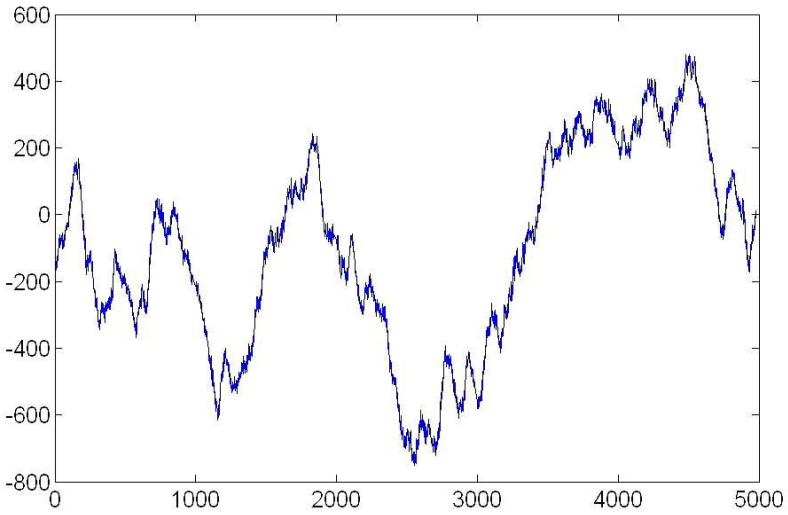


Figura 6.7. *Segunda secuencia de 5000 datos de la serie temporal CATS con cinco huecos de 20 puntos que predecir*

identificable (ver figuras 6.6 y 6.7), al contrario que sucede con otras series bien conocidas [Chang et al., 2001], [Mackey and Glass, 1977], [McNames et al., 1999]

En general, un modelo predictivo que tiene en cuenta salidas previas como variables de entrada tiene la forma

$$\hat{y}(t+h) = F\left(y(t-i_1), y(t-i_2), \ldots, y(t-i_m)\right) \tag{6.7}$$

donde h es el horizonte de predicción, y típicamente los índices i_1, i_2, $\ldots$, i_m toman la forma 0, τ, $2*\tau$, $\ldots$, $(m-1)*\tau$, donde τ es el periodo de muestreo.

6.4.3. Serie temporal diferenciada

Como se notó en [McNames et al., 1999], en vez de considerar la serie original, el problema de predicción se puede analizar utilizando la serie temporal diferenciada $u(t) = y(t) - y(t-1)$. Se tiene en cuenta nueva información, considerando las diferencias de un punto de datos al siguiente, con respecto del anterior. La serie original puede expresarse por tanto como

$$y(t) = y(t-1) + u(t) \tag{6.8}$$

La serie temporal diferenciada completa, puede verse en la figura 6.8, y una vista parcial se muestra en la figura 6.9. La ventaja de utilizar la serie temporal diferenciada en vez de la original es que todos los datos están dentro de un cierto dominio más estrecho, que permite una mejor cobertura del dominio de las variables de entrada. La aplicación del modelo MGFS a la serie temporal CATS utilizará la serie temporal diferenciada. Las secciones 6.2 y 6.3 presentan el modelo MGFS propuesto y la metodología de aprendizaje. Después, la sección 6.5 trata la aplicación del modelo MGFS a la serie temporal CATS utilizando predicción directa. La sección 6.6 introduce algunas mejoras propuestas sobre

el trabajo inicial [Herrera et al., 2004a], y la sección 6.7 muestra los resultados obtenidos para los nuevos datos de la serie CATS [CAT,].

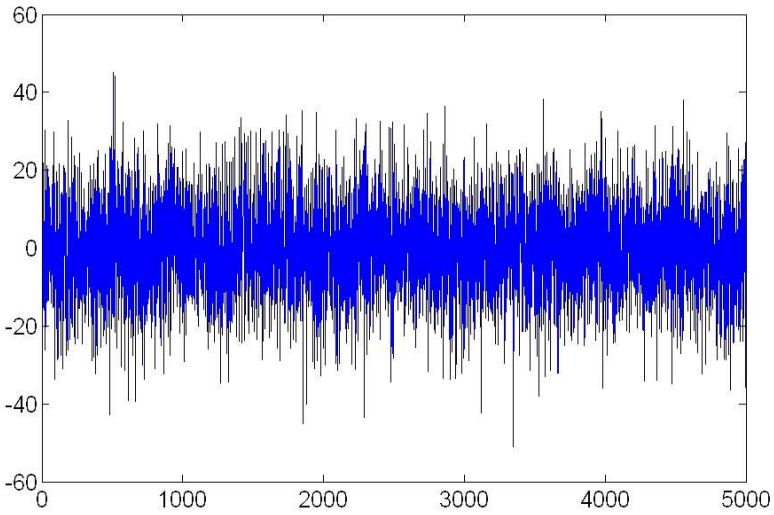


Figura 6.8. *Serie temporal diferenciada* $u(t)$

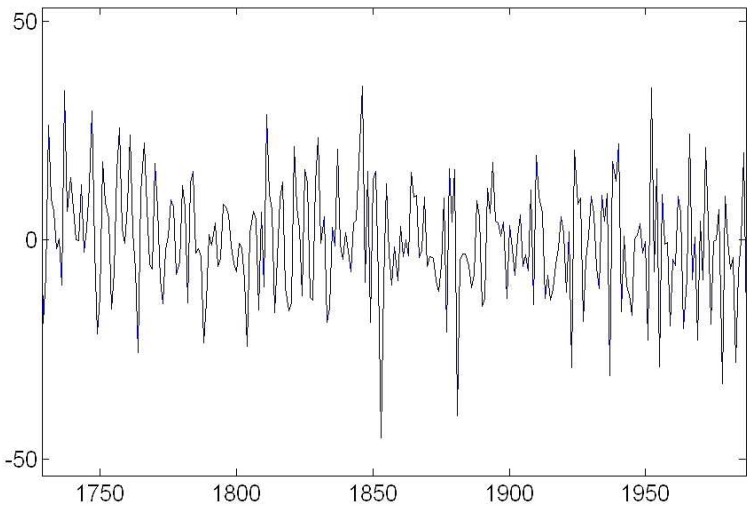


Figura 6.9. *Vista parcial de la serie temporal diferenciada* $u(t)$

6.5. Modelado de la Serie CATS Mediante la Serie Temporal Diferenciada

La función de autocorrelación parcial (PACF) de la serie diferenciada (ver figura 6.10) muestra que solo los primeros 20-25 datos previos pueden tener influencia sobre la predicción de cada dato $\hat{u}(t)$ utilizando un modelo lineal.

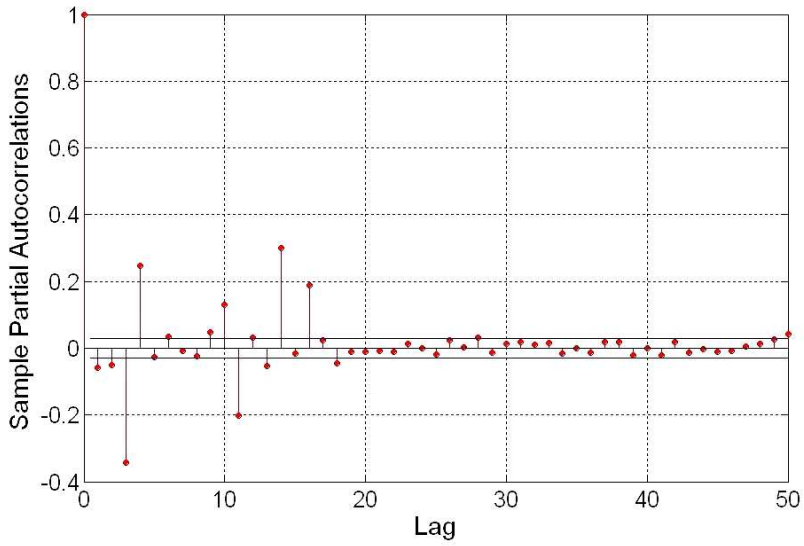


Figura 6.10. *Sample Partial Auto-Correlation function of the differentiated time series*

Para el modelo MGFS se realizaron numerosas pruebas tomando incuso hasta 100 valores previos para predecir $\hat{u}(t+1)$ (ver ecuación 6.7), pero los resultados obtenidos mostraron que considerando valores previos con i_m mayor que 20-25 no se proporcionaba un rendimiento mejor. Nótese en cualquier caso que en la figura 6.10, un lag l corresponde al efecto de la variable $u(t-l+1)$ sobre la variable a predecir $\hat{u}(t+1)$. Los 20 valores previos $(u(t-0)\dots u(t-20))$ se seleccionaron finalmente para lanzar el algoritmo de selección de arquitectura para sistemas Multigrid, con el objetivo inicial de predecir $\hat{u}(t+1)$,

$$\hat{u}(t+1) = F\left(u(t-0), u(t-1), \dots, u(t-20)\right) \tag{6.9}$$

Los datos de entrenamiento de entrada/salida $[u(t+1);\ u(t-0),\ u(t-1),\ \dots,\ u(t-20)]$ se dividieron aleatoriamente en training (4000 muestras aproximadamente) y test (1000 muestras aproximadamente). Estos conjuntos de datos no incluían obviamente las zonas de huecos desconocidas de la secuencia de 5000 valores.

El primer paso es seleccionar el límite en el número de reglas para el algoritmo de selección de arquitectura MGFS. Se realizó un procedimiento de validación cruzada con 5 bloques como se discutió en la sección 6.3.1, comprobando el rango 50-150. El límite seleccionado fue 60; a partir de este valor, el procedimiento de validación cruzada se estabilizó. Este proceso de validación cruzada llevó 4 horas en un PC con procesador P-2.8GHz.

La selección de arquitectura de sistemas Multigrid comienza a partir de una arquitec-

Tabla 6.4. *Evolución del algoritmo de identificación de la estructura en la ecuación 6.10*

Estructura MGFS	Número de reglas	Error de Validación Cruzada
$\{\{1\},\{1\},\{1,1\},\{1,1\}\}$	4	1
$\{\{1\},\{1\},\{1,1\},\{2,1\}\}$	5	0.910
$\{\{2\},\{1\},\{1,1\},\{2,1\}\}$	6	0.871
$\{\{2\},\{2\},\{1,1\},\{2,1\}\}$	7	0.850
$\{\{2\},\{2\},\{2,1\},\{2,1\}\}$	8	0.841
$\{\{2\},\{2\},\{2,2\},\{2,1\}\}$	10	0.831
$\{\{2\},\{2\},\{2,2\},\{2,2\}\}$	12	0.821
$\{\{2\},\{3\},\{2,2\},\{2,2\}\}$	13	0.815
$\{\{2\},\{3\},\{2,2\},\{3,2\}\}$	15	0.813
$\{\{2\},\{3\},\{2,2\},\{3,3\}\}$	18	0.812
$\{\{2\},\{3\},\{3,2\},\{3,3\}\}$	20	0.810
$\{\{2\},\{3\},\{3,2\},\{3,4\}\}$	23	0.809
$\{\{2\},\{3\},\{3,3\},\{3,4\}\}$	26	0.808
$\{\{2\},\{3\},\{3,3\},\{3,5\}\}$	29	0.807
$\{\{2\},\{3\},\{4,3\},\{3,5\}\}$	32	0.806
$\{\{2\},\{3\},\{4,3\},\{4,5\}\}$	37	0.805

tura con n sub-grids ($n = 20$ en este caso) de una variable. El proceso de eliminación de sub-grids deja-uno-fuera se realizó tras esta inicialización (como se explica en la sección 6.3.1). Así pues, el punto de comienzo pasa a ser una arquitectura con p sub-grids de una variable, que son aquellos que muestran tener relevancia respecto de la salida. Tras el proceso de eliminación deja-uno-fuera la arquitectura MGFS queda como: $A = \{\{u(t-2)\}, \{u(t-3)\}, \{u(t-13)\}, \{u(t-15)\}\}$. El número de regla se distribuye como se indica en la sección 6.3.1 (esto es, 15 para cada sub-grid). La primera iteración del algoritmo intenta encontrar sub-grids de orden segundo, que se pueden obtener añadiendo cualquiera de las 20 variables a los sub-grids de orden primero. Iteraciones sucesivas (m-ésimas) tendrían el mismo efecto en la búsqueda de sub-grids de orden m. El algoritmo finaliza seleccionando la siguiente arquitectura MGFS: $A = \{\{u(t-2)\}, \{u(t-3)\}, \{u(t-15), u(t-1)\}, \{u(t-13), u(t-12)\}\}$ (el tiempo de ejecución necesario para este caso son 10 minutos en un PC con procesador P-2.8GHz).

$$\begin{aligned}\hat{u}(t+1) = {} & F^1\left(u(t-2)\right) + F^2\left(u(t-3)\right) + \\ & + F^3\left(u(t-15), u(t-1)\right) + \\ & + F^4\left(u(t-13), u(t-12)\right)\end{aligned} \tag{6.10}$$

El algoritmo de identificación de la estructura se realiza posteriormente para obtener la estructura óptima para la arquitectura dada. La tabla 6.4 muestra la evolución del proceso de identificación de la estructura para el problema actual. Iteraciones posteriores del algoritmo no trajeron una mejora del error de validación cruzada. El algoritmo obtuvo una distribución de funciones de pertenencia que, con tan solo 37 reglas obtiene un rendimiento similar que con las 60 reglas obtenidas tras la ejecución del algoritmo de selección de arquitectura. El ajuste de parámetros de las funciones de pertenencia se ejecuta a partir de la séptima iteración (esto es, cuando el número de FPs de cualquiera de las variables llega a 3; ver sección 6.3.2).

El error de test obtenido con el modelo MGFS final con la arquitectura especificada en la ecuación 6.10, y con estructura $\{\{2\},\{3\},\{4,3\},\{5,5\}\}$ es NRMSE = 0.815 (con un valor medio similar de validación cruzada 0.805).

Para predecir los puntos restantes de los huecos ($u(t+2), u(t+3), \ldots, u(t+20)$), este trabajo utilizará un enfoque de predicción directa [Yongnan et al., 2005]). La predicción

directa presenta el inconveniente de que se necesitan entrenar varios modelos, uno para cada horizonte de predicción. Sin embargo, cada modelo puede entrenarse por separado para obtener un mejor rendimiento en la evaluación final considerando todos los horizontes de predicción. La alternativa es la predicción recursiva, que utiliza el mismo modelo $\hat{u}(t+1)$ para predecir todos los posibles horizontes de predicción. Sin embargo, el error obtenido para predecir $\hat{u}(t+1)$ es demasiado alto, y este error se arrastraría a la predicción de los siguientes puntos en los huecos.

Ahora pasamos a discutir la obtención de los modelos restantes para predecir los valores $\hat{u}(t+2)$, $\hat{u}(t+3)$, ..., $\hat{u}(t+20)$. Nótese que el modelo aprendido que aproxima $\hat{u}(t+1)$, puede utilizarse también para predecir $\hat{u}(t+2)$, puesto que $u(t)$ no es una variable de entrada para el modelo. En principio, se podrían tomar dos alternativas distintas para obtener los modelos restantes. Esto es, el objetivo podría ser predecir (trabajando siempre en la serie diferenciada), para el H-ésimo punto desconocido

$$\hat{u}(t+H) = F\left(u(t-0), u(t-1), \ldots, u(t-20)\right) \tag{6.11}$$

o bien

$$\hat{y}(t+H) - y(t) = \sum_{h=1}^{H} \hat{u}(t+h) = F\left(u(t-0), u(t-1), \ldots, u(t-20)\right) \tag{6.12}$$

esto es, la diferencia en la serie original entre los puntos $y(980+H)$ - $y(980)$.

Los modelos obtenidos para la primera posibilidad al aplicar el algoritmo de selección de arquitectura, así como los errores medios de validación cruzada y el error de test obtenidos se muestran en la tabla 6.5. El mismo límite (60) en el número de reglas se utilizó para todos los modelos por simplicidad. El tiempo de computación necesario para obtener todos los modelos fue de aproximadamente 3 horas en un PC con procesador P-800Mhz para el algoritmo de selección de arquitectura, y un tiempo similar de computación para el algoritmo adicional de optimización de la estructura de funciones de pertenencia y sus parámetros. Debe mencionarse sobre estos resultados, que el error aumenta conforma aumenta el horizonte. Antes se discutió que los 20 valores previos mostraron dar más información sobre $u(t+1)$; esto explica por qué al aumentar el horizonte, la información disponible para su predicción disminuye. Nótese también como el algoritmo obtiene a veces modelos sub-óptimos similares para horizontes de predicción cercanos H en $\hat{y}(t+H)$ (por ejemplo $\hat{y}(t+1)$ y $\hat{y}(t+2)$, $\hat{y}(t+5)$ y $\hat{y}(t+6)$, etc.). Estos coincidencias confirman la robustez del algoritmo de selección de arquitectura. La segunda posibilidad, esto es, la mostrada por la ecuación 6.12, mostró proporcionar un error de predicción peor para el entrenamiento y el test que la primera alternativa, esto es la mostrada por la ecuación 6.11.

Un ejemplo de la aproximación de una sección completa de 20 datos utilizando los modelos de la tabla 6.5 se muestra en la figura 6.11. Los veinte valores del hueco se predicen utilizando los modelos de la tabla 6.5 para la serie temporal diferenciada, y estos valores predichos se utilizan para recuperar la serie original esperada al invertir la serie temporal según la ecuación 6.8. El MSE estimado total es de 1000 para este enfoque multi-modelo directo de predicción.

Sin embargo, para los primeros cuatro huecos (primeros 80 puntos desconocidos) a predecir, conocemos la continuación de la serie temporal. La misma metodología puede aplicarse, invirtiendo la serie temporal para predecir las cuatro secciones de 20 puntos (teniendo en cuenta los datos que siguen además de los datos anteriores).

Un nuevo conjunto de 20 modelos similares se obtuvieron para este modelo de predicción hacia atrás

Tabla 6.5. *Error de predicción para $\hat{y}(t+p)$ utilizando $\hat{u}(t+p)$ para $p = 1:20$, y valores de lag en la estructura MGFS*

Para predecir	Err V.C.	Err Tst	Arquit. MGFS ($\{\{u(t-lag),\dots\}\dots\}$	Estruct. MGFS
$\hat{y}(t+1)$	0.805	0.815	$\{\{2\},\{3\},\{15,1\},\{13,12\}\}$	$\{\{2\},\{3\},\{4,3\},\{5,5\}y\}$
$\hat{y}(t+2)$	0.805	0.815	$\{\{1\},\{2\},\{14,0\},\{12,11\}\}$	$\{\{2\},\{3\},\{4,3\},\{5,5\}\}$
$\hat{y}(t+3)$	0.822	0.820	$\{\{0\},\{1\},\{13\},\{11,10\}\}$	$\{\{2\},\{3\},\{2\},\{3,3\}\}$
$\hat{y}(t+4)$	0.854	0.836	$\{\{0\},\{12\},\{10,9\}\}$	$\{\{3\},\{2\},\{4,3\}\}$
$\hat{y}(t+5)$	0.870	0.858	$\{\{11\},\{9,8\},\{9,1\}\}$	$\{\{4\},\{5,4\},\{5,4\}\}$
$\hat{y}(t+6)$	0.852	0.863	$\{\{10\},\{8,7\},\{8,0\}\}$	$\{\{4\},\{5,4\},\{5,4\}\}$
$\hat{y}(t+7)$	0.845	0.869	$\{\{9,10\},\{7,6,2,0\},\{7,6,2,15\}\}$	$\{\{2,2\},\{3,5,4,3\},\{3,5,4,2\}\}$
$\hat{y}(t+8)$	0.862	0.863	$\{\{8,9\},\{6,5,1\}\}$	$\{\{4,2\},\{3,4,4\}\}$
$\hat{y}(t+9)$	0.839	0.863	$\{\{7,8\},\{5,4,0\}\}$	$\{\{4,2\},\{3,4,4\}\}$
$\hat{y}(t+10)$	0.865	0.855	$\{\{6,7\},\{4,3,1\}\}$	$\{\{3,2\},\{3,4,3\}\}$
$\hat{y}(t+11)$	0.865	0.855	$\{\{5,6\},\{3,2,0\}\}$	$\{\{3,2,\},\{3,4,3\}\}$
$\hat{y}(t+12)$	0.861	0.864	$\{\{2,1\},\{4,5,0\}\}$	$\{\{3,4\},\{4,4,4\}\}$
$\hat{y}(t+13)$	0.871	0.861	$\{\{1,0\},\{3,4\}\}$	$\{\{3,4\},\{4,3\}\}$
$\hat{y}(t+14)$	0.885	0.862	$\{\{2\},\{0,3\}\}$	$\{\{4\},\{3,2\}\}$
$\hat{y}(t+15)$	0.951	0.943	$\{\{2\},\{13\},\{15\}\}$	$\{\{4\},\{3\},\{3\}\}$
$\hat{y}(t+16)$	0.951	0.943	$\{\{1\},\{12\},\{14\}\}$	$\{\{4\},\{3\},\{3\}\}$
$\hat{y}(t+17)$	0.951	0.943	$\{\{0\},\{11\},\{13\}\}$	$\{\{4\},\{3\},\{3\}\}$
$\hat{y}(t+18)$	0.950	0.945	$\{\{10\},\{12\}\}$	$\{\{4\},\{5\}\}$
$\hat{y}(t+19)$	0.950	0.945	$\{\{9\},\{11\}\}$	$\{\{4\},\{5\}\}$
$\hat{y}(t+20)$	0.950	0.945	$\{\{8\},\{10\}\}$	$\{\{4\},\{5\}\}$

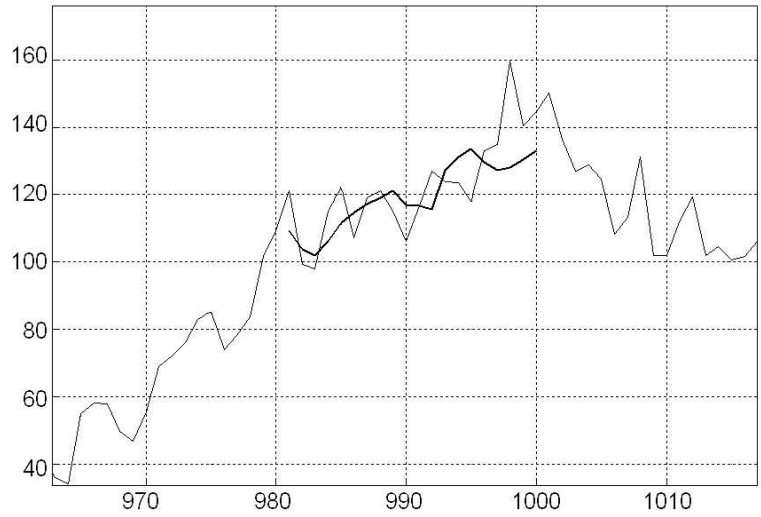


Figura 6.11. *Primera aproximación utilizando el sistema multi-modelo hacia adelante completo mostrado en la tabla 6.5*

$$\hat{u}(t-H) = F\left(u(t+0), u(t+1), \dots, u(t+20)\right) \tag{6.13}$$

Para combinar las salidas de los dos conjuntos de modelos directos (hacia adelante y hacia atrás) en los cuatro primeros huecos de las erie, se aplicó una estrategia de media ponderada. Los pesos de las salidas de los dos grupos de modelos se obtuvieron considerando el rendimiento de los mismos, ambos para $\hat{y}(t+H)$ y para $\hat{y}(t-H)$.

El error cuadrático medio esperado para esta metodología combinada, que tiene en cuenta predicción hacia adelante y hacia atrás para los puntos desconocidos es de 330. Este error cuadrático medio esperado se calculó tras 500 ejecuciones tomando huecos artificiales de 20 puntos en la serie de 5000 valores, e intentando predecirlos. Para los últimos 20 datos a predecir, esto es, de 4980 a 5000, solo el modelo hacia adelante $\hat{u}(t+1)\ldots\hat{u}(t+20)$ se usa.

La predicción final para los cinco huecos (100 datos) para la primera parte de la serie con 5000 datos se muestra en las figuras 6.12-6.16. El MSE obtenido para la predicción de los 80 primeros valores (primeros cuatro huecos) es de 352, y para los últimos 20 datos (último hueco) es de 1186 y finalmente, el MSE para el total de 100 puntos es de 518 (nótese que hay diferencias con los resultados publicados en [Herrera et al., 2004a], debido a un error de implementación de la media ponderada para combinar los resultados de los modelos hacia adelante y hacia atrás).

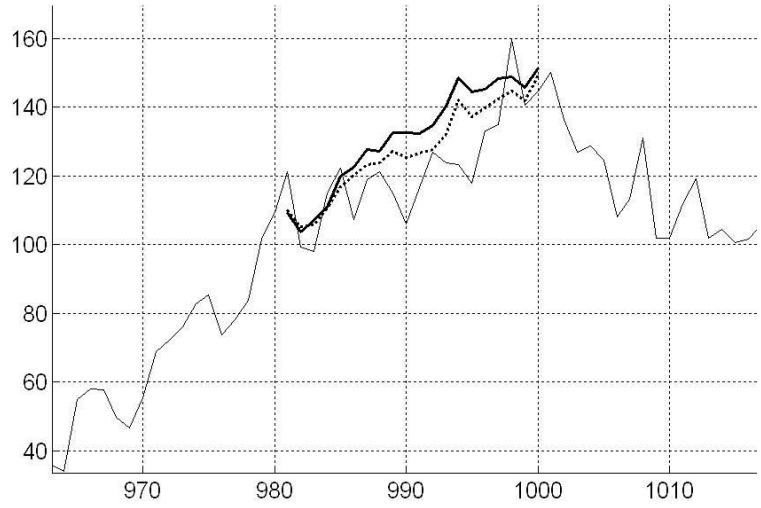


Figura 6.12. *Predicción para el primer hueco. La línea punteada indica la predicción final mejorada según la sección 6*

6.6. Mejoras Propuestas al Modelado: Reflejo de la Tendencia de la Serie Temporal en la Predicción Combinada Hacia Adelante y Hacia Atrás

La última sección trató sobre cómo para los cuatro primeros huecos (en los que disponemos de los datos anteriores y los siguientes), se puede utilizar una predicción directa combinada hacia adelante y hacia atrás. La media ponderada de ambas predicciones demuestra funcionar bien, pero no es óptima en el sentido de que no refleja completamente la tendencia de la serie temporal.

Para la predicción hacia adelante, los 20 valores predichos $\hat{y}(t+i)$, $i = 1:20$ se estiran de forma que el último valor predicho $\hat{y}(t+20)$ coincida con el último punto de una línea que una los puntos conocidos $y(t)$ y $y(t+21)$. Esta modificación el la predicción

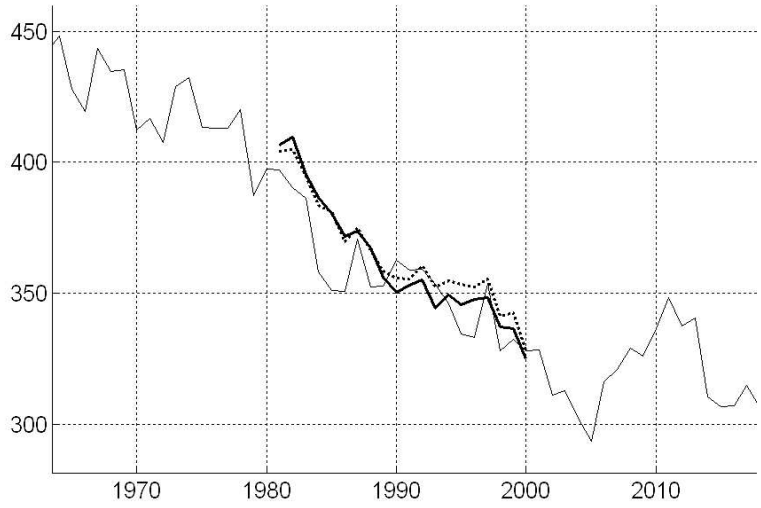


Figura 6.13. *Predicción para el segundo hueco. La línea punteada indica la predicción final mejorada según la sección 6*

hacia adelante lleva a obtener un MSe de 314 por si mismo. El mismo procedimiento se ha realizado para la predicción hacia atrás, estirando los valores predichos para que el primer valor predicho $\hat{y}(t+1)$ coincida con el primer punto de la línea que une $y(t)$ y $y(t+21)$. La predicción hacia atrás con esta modificación obtiene un MSE de 318.

Finalmente las dos predicciones obtenidas para los cuatro primeros huecos se han unido utilizando media ponderada para alcanzar un MSE esperado de 284. Ver la referencia [Lendasse et al., 2004] para una comparación con otros métodos. Ver las figuras 6.12-6.15 (en línea punteada) para verificar las nuevas predicciones para los cuatro huecos de los 5000 primeros datos de la serie CATS.

6.7. Resultados Obtenidos en los Nuevos Valores de la Serie CATS

Según el procedimiento explicado en las secciones 6.5 y 6.6, las figuras 6.17-6.20 muestran la predicción para los cuatro primeros huecos de los segundos 5000 datos proporcionados mostrados en la figura 6.7, que tienen un MSE esperado de 284.

Y finalmente, la figura 6.21 muestra la predicción para el hueco final de los segundos 5000 datos proporcionados, utilizando los 20 modelos directos hacia adelante obtenidos en la sección 6.5.

6.8. Conclusiones

En este capítulo se ha presentado un algoritmo de aprendizaje novedoso y efectivo para los sistemas difusos Multigrid, que incluye seleción de arquitectura e identificación de la estructura. Este tipo de sistemas difusos basados en grid, permite mantener un coste computacional bajo, a la vez que es capaz de considerar un mayor número de variables de

Figura 6.14. *Predicción para el tercer hueco. La línea punteada indica la predicción final mejorada según la sección 6*

entrada en el entrenamiento. Mantiene gran parte de las ventajas de los sistemas basados en grid tradicionales, pero se puede aplicar a sistemas de dimensionalidad mayor. El algoritmo propuesto selecciona las interrelaciones más relevantes entre las variables de entrada para realizar la aproximación. Este enfoque por tanto es muy adecuado para problemas de aproximación funcional y predicción de series temporales en los que las variables de entrada implicadas no están determinadas, y se necesita una solución efectiva e interpretable.

El modelo MGFS, se ha aplicado al problema de predicción de la serie temporal CATS utilizando un enfoque directo de predicción [Yongnan et al., 2005]. Para los cuatro primeros huecos a predecir, en los que la continuación de la serie está disponible, se han utilizado un conjunto de predictores hacia adelante y hacia atrás. El modelo global de predicción además tiene en cuenta la tendencia que presenta la serie temporal para obtener una mejor aproximación. Para el último hueco de la serie solo se han utilizado los modelos directos hacia adelante. Todos los modelos MGFS obtenidos están compuestos por un número reducido de reglas sencillas con un número bajo de antecedentes. Los resultados obtenidos muestran que este enfoque de predicción directa basado en MGFS presenta un buen rendimiento para este problema de predicción de series temporales.

Como trabajo futuro, por un lado se pretende mejorar la robustez del algoritmo de selección de arquitectura, mediante un mejor particionamiento de las reglas en la arquitectura del sistema Multigrid para cada problema específico. Además se pretende estudiar el uso de consecuentes de orden alto (no solo constante), lo que podría llevar a obtener modelos con un número menor de reglas sin pérdida de la interpretabilidad [Herrera et al., 2005f].

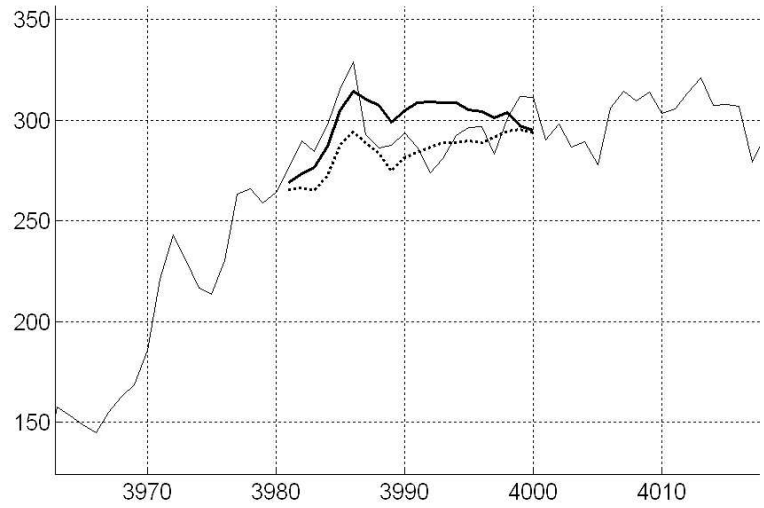


Figura 6.15. *Predicción para el cuarto hueco. La línea punteada indica la predicción final mejorada según la sección 6*

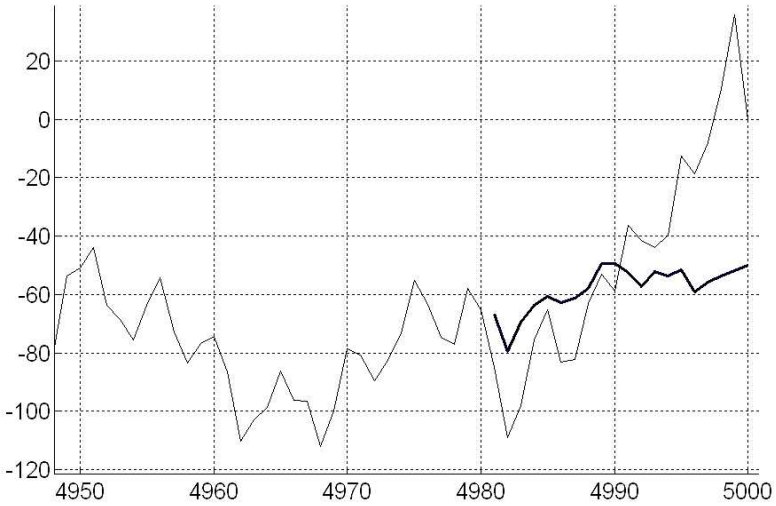


Figura 6.16. *Predicción para el quinto hueco*

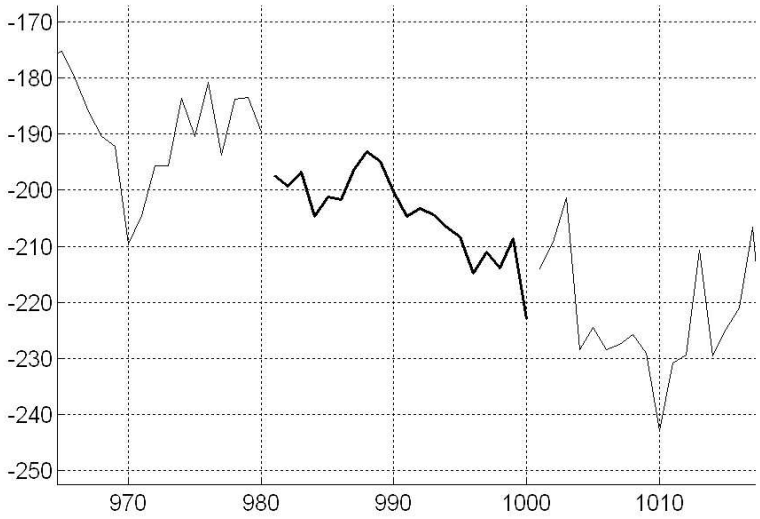


Figura 6.17. *Predicción final para el primer hueco de los segundos 500 datos proporcionados*

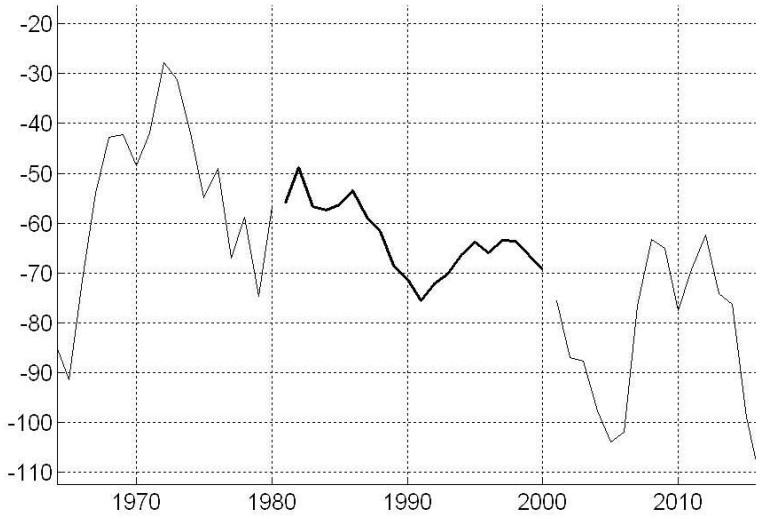


Figura 6.18. *Predicción final para el segundo hueco de los segundos 500 datos proporcionados*

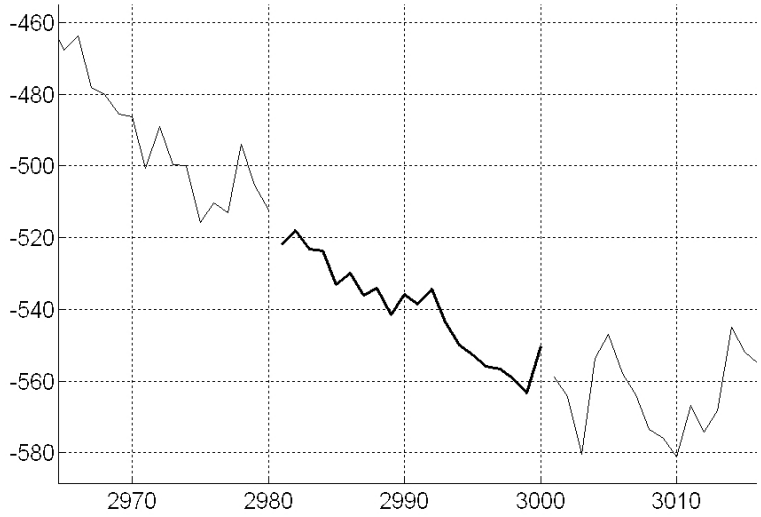


Figura 6.19. *Predicción final para el tercer hueco de los segundos 500 datos proporcionados*

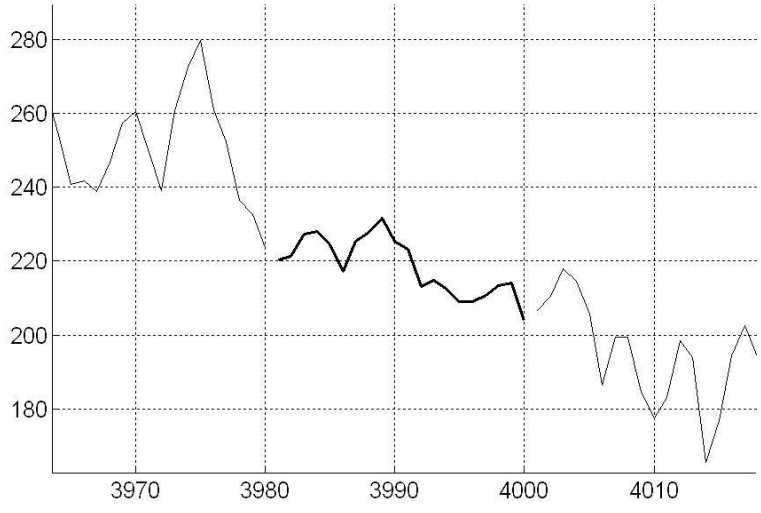


Figura 6.20. *Predicción final para el cuarto hueco de los segundos 500 datos proporcionados*

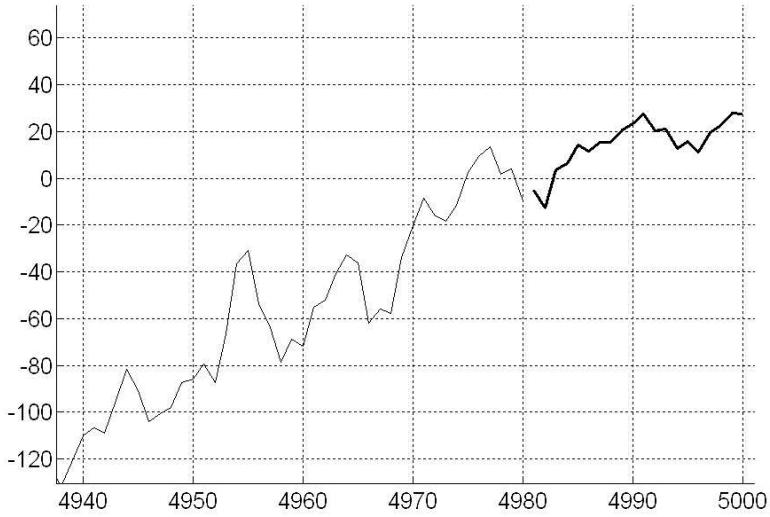


Figura 6.21. *Predicción final para el quinto hueco de los segundos 500 datos proporcionados*

Capítulo 7

Mejoras en el Modelado Recursivo de Series Temporales Usando LS-SVM y Modelos TSK Avanzados

Hay un gran rango de paradigmas y un gran número de metodologías que se aplican al problema de predicción de series temporales. La mayoría se presentan como un problema de aproximación funcional modificado utilizando datos de entrada/salida, en el que las entradas se obtienen utilizando valores de la serie en momentos anteriores. Así, el modelo obtenido normalmente predice el valor de la serie en el momento $(t+h)$ utilizando momentos de tiempo anteriores $(t-\tau_1), (t-\tau_2), \ldots, (t-\tau_n)$. Sin embargo, el aprendizaje de un modelo para predicción de series temporales a largo plazo puede verse como un problema mucho más complicado, puesto que puede utilizar sus propias salidas como valores de entrada para predecir valores sucesivos (predicción recursiva). El trabajo presentado en este capítulo muestra la aptitud de dos metodologías diferentes, el modelo TaSe y las Máquinas de Vectores Soporte de Mínimos Cuadrados, para resolver el problema de predicción de series temporales a largo plazo utilizando predicción recursiva. Este trabajo además introduce algunas técnicas que incrementan el rendimiento de este tipo de modelos avanzados de predicción a un paso (y en general de cualquier modelo de predicción a un paso), cuando se utilizan recursivamente para predicción de series temporales a largo plazo.

7.1. Introducción

Hay un gran número de metodologías que tratan el problema de la predicción de series temporales [Weigend and Gershenfeld, 1993]. Las Redes Neuronales Artificiales, Sistemas Difusos, Máquinas de Vectores Soporte, Modelos AR-ARMA, etc. están dentro de los paradigmas que se han aplicado a este problema. En el caso de la predicción de series temporales a largo plazo, se pueden tomar dos tendencias a la hora de predecir los datos: predicción directa y predicción recursiva [Yongnan et al., 2005]. La predicción directa implica la construcción de modelos distintos para cada horizonte de predicción que se necesite. Por otra parte, la predicción recursiva solo utiliza un modelo para predecir todos los horizontes, pero con el inconveniente de que el error cometido en los horizontes cercanos se transmite a los horizontes más lejanos, al usarse las mismas salidas del modelo como entradas para los subsiguientes datos desconocidos. Este trabajo trata con predicción recursiva, mostrando dos metodologías diferentes para construir el modelo recursivo, y presentando una técnica general que mejora la precisión en la predicción para este tipo de problemas.

Las metodologías que se utilizan en este trabajo son el modelo TSK TaSe, ver referencias [Herrera et al., 2004c], [Herrera et al., 2005f], [Herrera et al., 2005e] y las Máquinas

de Vectores Soporte de Mínimos Cuadrados [Suykens et al., 2002]. El primero como se ha visto en el capítulo 4, es una metodología de aprendizaje difusa avanzada que proporciona un modelo interpretable de aproximación dado un conjunto de datos de entrada y salida. Este modelo elimina, al menos parcialmente, los dos inconvenientes principales del entrenamiento de sistemas difusos tipo TSK basados en grid: la maldición de la dimensionalidad y la pérdida de interpretabilidad. El modelo TaSe obtiene un buen rendimiento en problemas de aproximación funcional y predicción de series temporales comparando con otras técnicas de soft-computing. El segundo es una reformulación del modelo genérico bien conocido de Maquinas de Vectores Soporte, que está basado en la teoría estadística del aprendizaje. Esta metodología proporciona un buen rendimiento con una buena generalización en problemas de clasificación y aproximación funcional. Su formulación lleva a resolver un sistema de ecuaciones lineales de Karush-Kuhn-Tucker (KKT) [Suykens et al., 2002]. Las LS-SVM están íntimamente relacionadas con las redes de regularización y con los procesos gaussianos, pero adicionalmente enfatizan y explotan la interpretación primal-dual de la teoría de la optimización [Schoelkopf and Smola, 2002]. Las LSSVM, al igual que las SVM tradicionales, también sufren de la maldición de la dimensionalidad, pero en el número de puntos de entrenamiento. Las LSSVM de tamaño fijo (fixed-size) [Suykens et al., 2002] son uno de los métodos que han aparecido en la literatura para resolver este problema, que es una de las tareas que sufre una tarea de investigación más intensiva para este tipo de paradigmas. Este trabajo no pretende comparar estas metodologías diferentes, sino que trata con el problema general de predicción de series temporales a largo plazo utilizando modelos recursivos (independientemente del paradigma utilizado), y con una técnica general para obtener un mejor rendimiento bajo este planteamiento.

El resto del capítulo se organiza como sigue: la sección 7.2 profundiza un poco más en las diferencias y ventajas más notables de las dos metodologías de aprendizaje propuestas, así como un ejemplo de aplicación al problema bien conocido de predicción de series temporales para la serie Mackey-Glass [Mackey and Glass, 1977]. La sección 7.3 presenta brevemente un punto de vista modificado de la predicción de series temporales a largo plazo. La sección 7.4 presenta una metodología de aprendizaje modificada, mostrando su aplicación para los modelos TaSe y LSSVM, para mejorar la predicción para series a largo plazo. Finalmente la sección 7.5 finaliza el capítulo.

7.2. Metodologías de Aprendizaje Utilizadas y Aplicación a la Serie Temporal Mackey-Glass

7.2.1. Modelo TaSe

En el capítulo 4 las secciones 4.2, 4.3, 4.4 y 4.5 explican el modelo TaSe con detenimiento, su algoritmo de aprendizaje y la sección 4.6.3 su aplicación al problema de predicción de series temporales de la serie Mackey-Glass.

Debemos notar aquí de nuevo, que el uso de particionamiento del espacio de entrada en grid para predicción de series temporales tiene la desventaja de que puesto que normalmente los puntos se distribuyen en ciertas zonas del espacio de entrada, existe un desperdicio de esfuerzos al cubrir con equidad la totalidad del espacio de entrada. Esto tiene dos consecuencias importantes. Primero, hay un esfuerzo extra al realizar la aproximación que puede llevar a un incremento innecesario en complejidad del sistema, y a un decremento en rendimiento. Y segundo, con respecto a la interpretabilidad, algunas reglas podrían quedar centradas en puntos en el espacio n-dimensional que no representa puntos de datos reales

Tabla 7.1. *Comparación del error de aproximación para diferentes métodos para la serie temporal Mackey-Glass con paso de predicción igual a 6. Ver [Rojas et al., 2002]*

Método		Error de Predicción (RMSE)
Auto Regressive Model		0.19
Cascade Correlation NN		0.06
Back-Prop. NN		0.02
6th-order Polynomial		0.04
Linear Predictive Method		0.55
Kim and Kim (Genetic Algorithm and Fuzzy System)	5 MFs	0.049
	7 MFs	0.042
	9 MFs	0.038
ANFIS and Fuzzy System (16 rules $\approx$ 100 par.)		0.007
Classical RBF (with 23 neurons $\approx$ 200 par.)		0.0114
PG-RBF [Rojas et al., 2002] (with 12 neurons $\approx$ 200 par.)		0.0029
TaSe Fuzzy System with 6 rules		0.0024
TaSe Fuzzy System with 12 rules		0.0017
Optimal TaSe Fuzzy System with 36 rules		0.001
LS-SVM		0.0005

de la función. Para resolver estos problemas, se podría usar un particionamiento basado en clustering en las áreas del espacio de entrada en las que se necesita, así resolviendo estos problemas (ver capítulo 5.

7.2.2. Máquinas de vectores soporte de mínimos cuadrados

Las condiciones del problema de predicción de series temporales de Mackey-Glass se han presentado ya anteriormente en la sección 4.6.3. Los primeros 500 datos se han usado para entrenamiento y los últimos 500 para test. La implementación de las LS-SVM utilizada puede encontrarse en [LS-,].

El entrenamiento del modelo LS-SVM comienza por averiguar los hiperparámetros óptimos para el problema dado. Utilizando validación cruzada con búsqueda en grid, los hiperparámetros óptimos obtenidos para el modelo LS-SVM son $\sigma = 0{,}655$ y $\gamma = 4864$. Con estos hiperparámetros, el entrenamiento del modelo lleva a obtener un RMSE de entrenamiento igual a 0.00041 y de test de 0.00052. Como puede observarse, el rendimiento del modelo LS-SVM sobrepasa a todos los modelos presentados en la tabla 7.1. Recuérdese que el el modelo LS-SVM tan solo hay que optimizar el valor de dos hiperparámetros utilizando un método de búsqueda local, el parámetro de regularización y la anchura del kernel. La estructura del modelo LS-SVM obtenido, aunque se dice que es un modelo no paramétrico, se asemeja a una red RBF con una neurona por cada punto de entrenamiento. Además, como se ha comentado anteriormente, el control del sobreajuste se realiza mediante el uso de un parámetro de regularización γ. Estas condiciones proporcionan una flexibilidad alta en el comportamiento del modelo para adaptarse a cada problema particular, y como se ha visto para este problema específico, la precisión en la predicción es excelente.

7.3. Predicción a Largo Plazo

Como se ha mencionado, hay dos tendencias principales al tratar predicción de series temporales a largo plazo: predicción directa y predicción recursiva [Yongnan et al., 2005]. Debido a las características de cada enfoque, puede esperarse que la predicción directa proporciona un mejor rendimiento en precisión de la predicción, puesto que utiliza un modelo específico para cada horizonte deseado. Sin embargo, tiene dos inconvenientes principales que podrían hacer que el uso de los modelos recursivos sea más conveniente o deseable: por un lado, el modelado y la comprensión del comportamiento de la serie temporal se pierde cuando se utilizan diferentes modelos para cada horizonte de predicción. Por otro lado, para horizontes de predicción a muy largo plazo, como ocurre en algunos casos [Jung et al., 2005], [EUN,], en los que se necesita un gran número de predicciones, un enfoque de predicción directa aparentemente es inútil al requerirse un número excesivamente grande de modelos. Aún para horizontes largos, cuando el ruido de la serie es considerable, un enfoque de predicción directa puede ser deseable para obtener mejores predicciones. Pero para niveles de ruido aceptables, la predicción recursiva puede obtener buenas predicciones utilizando un solo, y por tanto globalmente más comprensible, modelo.

7.3.1. Predicción recursiva

Ahora el problema de la predicción recursiva se estudia más en profundidad. Hasta ahora, se han visto dos enfoques diferentes para el modelado de un conjunto de datos de entrada/salida en los que el objetivo (por ejemplo para la serie temporal de Mackey-Glass) es predecir la salida $\hat{x}(t+6)$, asumiendo que las entradas especificadas en la ecuación 4.32 son conocidas. Pero para problemas de predicción a largo plazo, el objetivo podría ser expresado como: "teniendo una secuencia cortada de datos, intentar predecir el resto de la secuencia hasta que se alcance un cierto momento". Así, los datos de entrenamiento para este modelo serían los de la ecuación 4.32, pero en realidad, la aplicación del modelo para predecir la serie completa debería utilizar iterativamente los datos de entrenamiento

$$\hat{x}\left(t+6\right) = f\left(\hat{x}\left(t-18\right),\ \hat{x}\left(t-12\right),\ \hat{x}\left(t-6\right),\ \hat{x}\left(t\right)\right) \tag{7.1}$$

y en principio, esto no es para lo que se ha entrenado nuestro modelo inicial.

Supóngase un ejemplo sencillo de serie temporal $x(t)_{t=1.,4} = \{1, 2{,}5, 3{,}5, 4\}$. Un modelo que intenta predecir $\hat{x}(t+1) = f(x(t))$ utilizaría como entrenamiento los pares de datos $[1; 2{,}5]$, $[2{,}5; 3{,}5]$, $[3{,}5; 4]$. Así, utilizando mínimos cuadrados, un modelo lineal sencillo que utiliza un solo parámetro constante $\hat{x}(t+1) = a * x(t)$ resolvería la ecuación

$$[1, 2{,}5, 3{,}5]' * a = [2{,}5, 3{,}5, 4]'. \tag{7.2}$$

La solución es $a = 1{,}29$ y el MSE = 0.6. Pero de hecho, el error de predicción a largo plazo, comenzando con el valor inicial $x(t_0) = 1$ y prediciendo los siguientes valores utilizando la salida del modelo recursivamente es MSE = 2.72 (la predicción "a largo plazo" sería $[1{,}29, 1{,}29^2, 1{,}29^3]$). Así pues, la manera correcta de entrenar el modelo recursivo a largo plazo sería

$$\left[1, 1 * a, 1 * a^2\right]' * a = [2{,}5, 3{,}5, 4]' \tag{7.3}$$

cuya solución es $a = 1{,}67$ y el error de predicción MSE a "largo plazo" es 0.54. No obstante, incluso para este ejemplo sencillo, esto es un problema de optimización no lineal,

con una complejidad muy alta, y por tanto las técnicas de modelado de aproximación funcional tradicionales no funcionan aquí.

7.4. Mejorando la Predicción a Largo Plazo Utilizando Modelos de Predicción a Un Paso

En la sección anterior, se ha visto brevemente la tarea recursiva tan grande que sería modelar directamente un problema de predicción a largo plazo a partir de un conjunto de entrada/salida. Como se ha visto en [Yongnan et al., 2005], en principio el modelo de predicción a un paso puede utilizarse directamente para realizar dicha tarea. Esta sección presenta una metodología general para mejorar los predictores a un paso cuando se utilizan como predictores recursivos.

7.4.1. Utilizando el modelo difuso TSK TaSe

Considérese la serie temporal Mackey-Glass, y el modelo TaSe óptimo obtenido como se ha visto en la tabla 4.6.3. Partamos de que utilizamos este modelo predictor a un paso para realizar predicciones a largo plazo sobre la serie temporal Mackey-Glass. Suponiendo que se conoce una parte inicial de la serie, y que el objetivo es predecir el resto utilizando este modelo (al menos se necesitarían un total de 23 (18+5) ejemplos como parte conocida inicial de la serie para predecir el resto de 500 valores). Así, a partir de este trozo inicial, las salidas estimadas se usarían iterativamente como entradas nuevas (ver ecuación 7.1). La figura 7.1 muestra la predicción a largo plazo (ecuación 7.1), con $\hat{x}(t) = x(t)$ para los primeros valores $t = 1 \ldots 23$ para el conjunto de test. El RMSE obtenido para el conjunto de entrenamiento es igual a 0.030, y para el conjunto de test es 0.031.

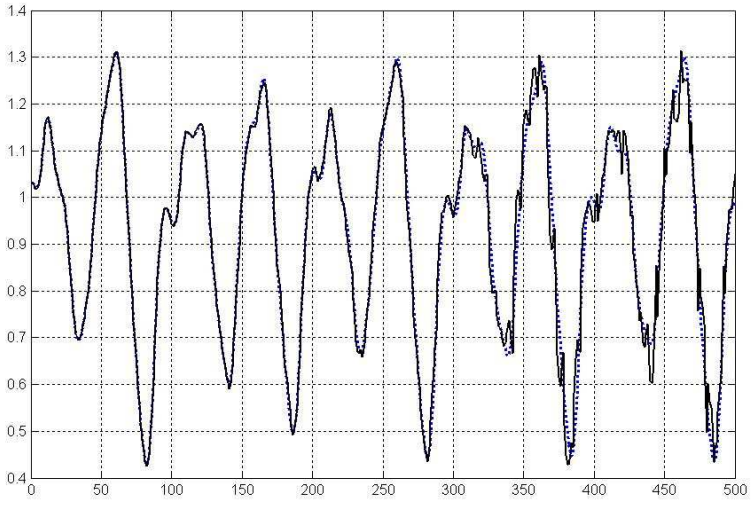


Figura 7.1. *Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para el conjunto de test utilizando el modelo TaSe. Error de test RMSE = 0.031.*

Estos resultados muestran que se obtiene una solución bastante buena para este problema de predicción a largo plazo, utilizando recursivamente un predictor a un paso. Gracias a la precisión del modelo TaSe, se obtiene una secuencia de salidas muy similar a los datos originales.

Sin embargo, se presenta ahora una modificación en el proceso de entrenamiento que permitirá obtener mejores resultados. En [Decoste and Schoelkopf, 2002], Schoelkopf y otros añadieron pequeñas modificaciones en cada muestra de entrenamiento, expandiendo el conjunto de entrenamiento para un problema de clasificación de dígitos escritos a mano. De forma similar, el conjunto de datos de entrenamiento de la serie temporal para este caso podría expandirse de forma que pequeñas modificaciones de la serie se considerasen también en el entrenamiento. Sería posible por ejemplo, añadir un nuevo conjunto de ejemplos obtenidos a partir de la serie original, calculando la serie $x(t + 0{,}5)$. Sin embargo, éste es un problema de interpolación y esta tarea puede ser en algunos casos tan difícil o más como el problema de predicción original. Con todo, para la serie Mackey-Glass, hemos obtenido una muy buena aproximación a un paso para la serie (RMSE = 0.0011). Así, utilizando el mismo modelo original, el conjunto de datos de entrenamiento se ha expandido para considerar una nueva sección de la serie temporal, que es una pequeña modificación de la original y que, de hecho, presenta el mismo comportamiento. El conjunto expandido de muestras de entrenamiento consta ahora por tanto de la serie original $x(t), t = 1.{,}500$ más un nuevo trozo ficticio de serie $\hat{x}, t = 1.{,}478$, donde el $\hat{x}$ ha sido obtenido utilizando el modelo predictivo TaSe.

Para el reentrenamiento del modelo TaSe utilizando el conjunto de entrenamiento extendido, nótese que tan solo se han actualizado los coeficientes en los consecuentes de las reglas. La estructura y parámetro en los antecedentes se quedan como están, con el objetivo de evitar un coste computacional excesivo y para conservar la simplicidad del enfoque propuesto. La figura 7.2 muestra la predicción a largo plazo del conjunto de test, obtenida con el modelo TaSe reentrenado utilizando el conjunto de entrenamiento extendido. El RMSE obtenido para la predicción a largo plazo del conjunto de entrenamiento es de 0.015 y para el conjunto de test es 0.019. Intentos sucesivos de reextender el conjunto de entrenamiento, repitiendo el mismo proceso (reextendiendo el conjunto de entrenamiento, utilizando la salida predicha a un paso para el modelo reentrenado) no produjo mejoras significativas para la predicción a largo plazo para este ejemplo.

Ahora considérese la serie original Mackey-Glass con un componente añadido de ruido con varianza $0{,}05 * \sigma_x$, donde σ_x es la desviación estándar de la serie temporal original. Utilizando este nuevo ejemplo, el modelo TaSe se utilizó de nuevo para aprender un predictor a un paso. El modelo TaSe óptimo tenía en este cas tan solo 12 reglas (la optimización se realizó de manera similar a como se vio en el capítulo 4 para la serie Mackey-Glass original), debido al comportamiento ruidoso y a la utilización de un conjunto de validación para evitar el sobreajuste. Como se hizo para la serie temporal sin ruido, el modelo se reentrenó utilizando la salida del modelo de predicción a un paso como nuevos datos de entrada (un nuevo trozo de la serie temporal). Sin embargo, en este caso se realizó un proceso iterativo, repitiendo esta técnica. La salida obtenida se utilizó una y otra vez como nuevos datos de entrenamiento, obteniéndose una secuencia de nuevos modelos a un paso reentrenados. La figura 7.3 muestra la predicción a largo plazo para el conjunto de test utilizando los datos de entrenamiento originales (figura 7.3a), con RMSE 0.12 y 0.13 para los datos de entrenamiento y test respectivamente. El proceso iterativo se llevó a cabo hasta que la predicción del conjunto de entrenamiento no mejoró. El RMSE obtenido tras este proceso iterativo fue de 0.10 para ambos conjuntos de datos. La predicción a largo plazo del modelo óptimo TaSe reentrenado para el conjunto de test se muestra en la figura 7.3b. Este proceso iterativo por tanto muestra mejorar levemente la precisión en la predicción de las reglas

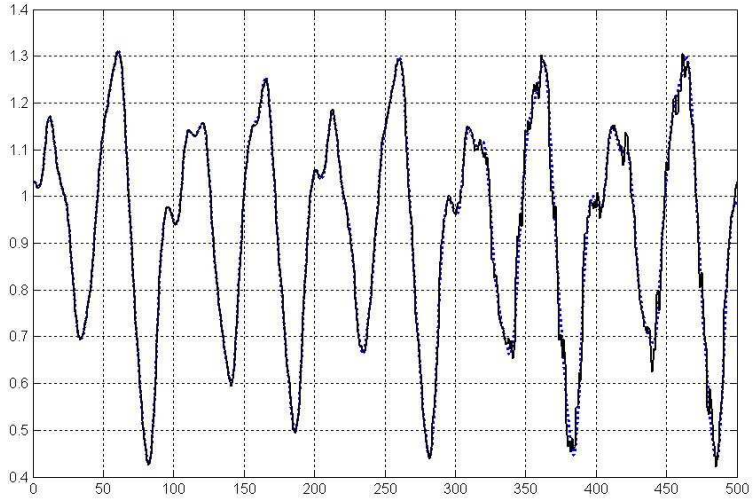


Figura 7.2. *Predicción a largo plazo original (linea punteada) y aproximada (línea sólida) para el conjunto de test utilizando el conjunto de entrenamiento expandido. RMSE de test = 0.019*

para la predicción a largo plazo para algunos casos.

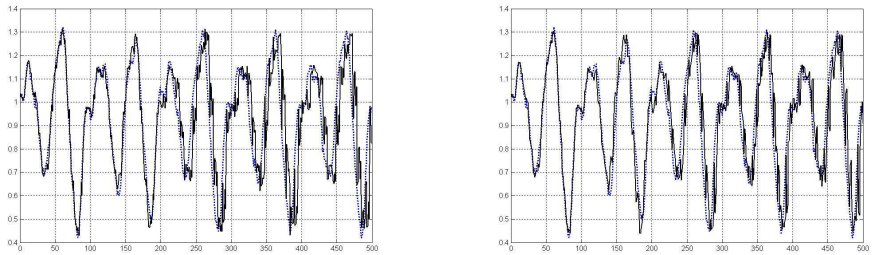


Figura 7.3. *a) Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para el conjunto de test utilizando el conjunto de entrenamiento original para el problema ruidoso. RMSE de test = 0.13 b) Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para el conjunto de test utilizando el conjunto de entrenamiento extendido para el problema ruidoso. RMSE de test = 0.10.*

Los resultados obtenidos muestran que este proceso puede verse como un proceso de eliminación de ruido en una serie temporal. Si se encuentra un modelo a un paso suficientemente preciso, y que es capaz de eliminar parcialmente el ruido de la serie, la salida de dicho modelo puede utilizarse como nuevos datos de entrenamiento, con el objetivo de mejorar los resultados de predicción recursiva a largo plazo.

Como segundo ejemplo, considérese ahora una serie temporal distinta, el flujo mensual en pies cúbicos por segundo del rio Snake, tomada de [Tim,]. La figura 7.4 muestra

la forma de la serie. Nótese cómo para este caso, la serie presenta un alto nivel de ruido, aunque con un patrón bien reconocible en su comportamiento.

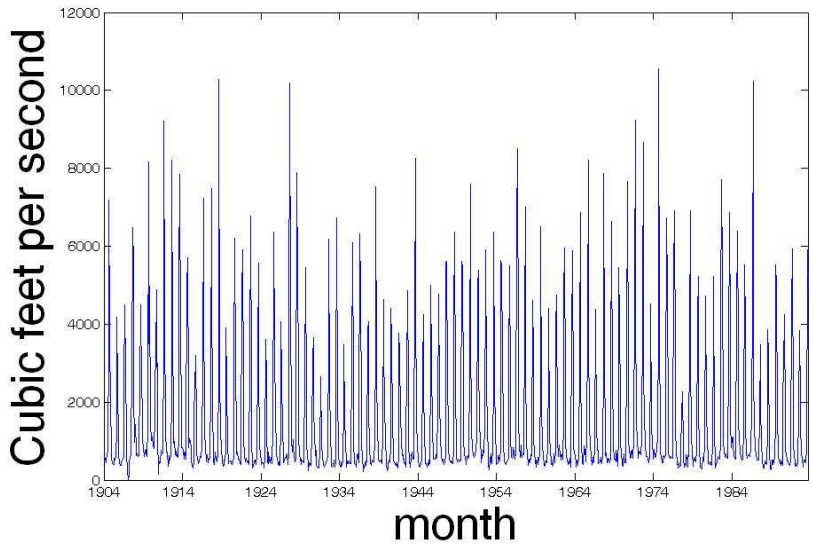


Figura 7.4. *Flujo mensual en pies cúbicos por segundo del rio Snake, durante el periodo de 1904 a 1994, ver la referencia [Tim,]*

De nuevo para esta serie, se tomaron 1000 muestras, 500 para entrenamiento y validación (tomando aleatoriamente 400 y 100 respectivamente) y 500 para test. En este caso los valores tomados por $h, \tau_0, \ldots \tau_n$ en la ecuación $\hat{x}\left(t+h\right) = f\left(x\left(t-\tau_0\right),\ x\left(t-\tau_1\right), \ldots,\ x\left(t-\tau_n\right)\right)$ son [3, 0, 3, 6, 9] (ver la referencia [McNames et al., 1999] sobre cómo encontrar valores adecuados para estos parámetros dada una serie temporal). El modelo TaSe óptimo para este problema de predicción a un paso tiene tan solo dos reglas con RMSE de entrenamiento = 878 y RMSE detest = 890. Nótese que puesto que la misma medida del error se ha utilizado para este ejemplo (la raíz del error cuadrático medio, RMSE), los errores en esta serie ruidosa no normalizada son mucho mayores que para el ejemplo anterior. La predicción a largo plazo para los conjuntos de entrenamiento y test, proporcionan un RMSE total igual a 1769 para el conjunto de entrenamiento y 1235 par el conjunto de test. La figura 7.5 muestra la predicción a largo plazo para el flujo mensual del rio Snake, utilizando el predictor TaSe a un paso. Puede observarse que la capacidad de predicción a largo plazo del modelo a un paso se desvanece en un rango corto de tiempo.

De nuevo para este ejemplo de serie temporal, los datos de entrenamiento se ampliaron con la salida del predictor TaSe a un paso, considerado como un nuevo trozo de la serie temporal. Este trozo ficticio se añadió a los datos de entrenamiento, conservando la estructura óptima anterior del modelo TaSe y tan solo reobteniendo de nuevo los consecuentes óptimos de las reglas. Tras una iteración, la predicción a largo plazo mejoró hasta un RMSE de entrenamiento de 1618 y 1060 de test. La figura 7.6 muestra la predicción a largo plazo para los conjuntos de test y entrenamiento, utilizando el conjunto de datos extendido de entrenamiento para la serie del flujo del rio Snake. Se observa claramente la mejora en la predicción que se obtiene en la predicción de la serie a largo plazo.

De nuevo se ha realizado un proceso iterativo, extendiendo el conjunto de datos de

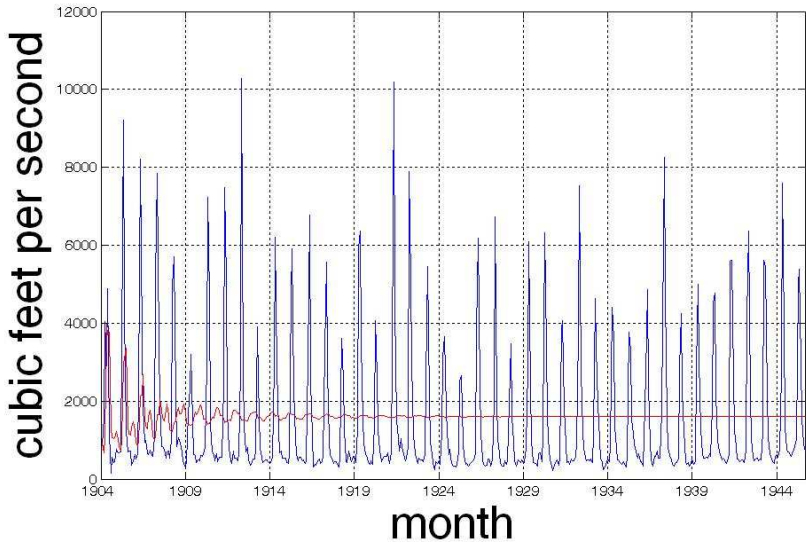


Figura 7.5. *Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para el conjunto de entrenamiento. RMSE de entrenamiento = 1769*

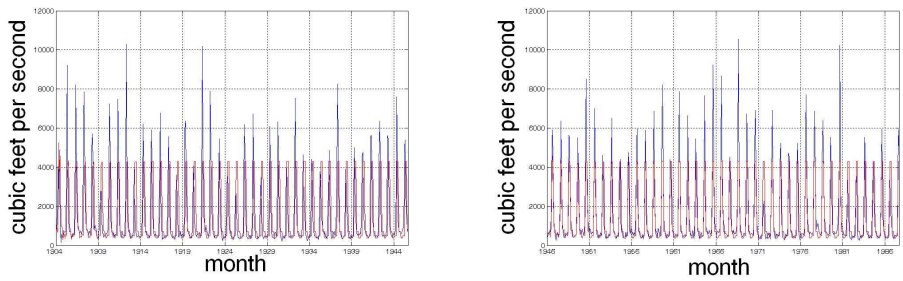


Figura 7.6. *a) Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para los datos de entrenamiento utilizando el conjunto de datos extendido para el problema del flujo de un rio. RMSE de entrenamiento 1618. Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para los datos de test utilizando el conjunto de datos extendido para el problema del flujo de un rio. RMSE de test 1060*

entrenamiento, hasta que la predicción a largo plazo del conjunto de entrenamiento original no mejora. De esta forma, el RMSE final a largo plazo del conjunto de entrenamiento alcanzó 1611, y el RMSE a largo plazo del conjunto de test 1041.

7.4.2. Utilizando LS-SVMs

Ahora considérese el enfoque de las LS-SVMs, aplicado a la predicción de la serie Mackey-Glass. Tomando la serie y utilizando el predictor LS-SVM a un paso para realizar una predicción recursiva a largo plazo, lleva a un RMSE de entrenamiento de 0.014 y a

un RMSE de test de 0.017. De nuevo el mismo procedimiento se realizó para extender el conjunto de entrenamiento con la salida del predictor LS-SVM a un paso. Se utilizaron los mismos valores de los hiperparámetros $\sigma = 0{,}655$ y $\gamma = 4864$ para reentrenar el modelo con el conjunto extendido de datos. La predicción a largo plazo utilizando el conjunto de entrenamiento extendido mejora los resultados con un RMSE de 0.0103 para el conjunto de entrenamiento y un RMSE de 0.0053 para el conjunto de test.

De nuevo con la serie temporal del flujo del rio Snake, se realizó el mismo procedimiento. El predictor óptimo a un paso obtuvo como hiperparámetros $\sigma = 3{,}64$ y $\gamma = 7{,}62$. Los RMSE de entrenamiento y test obtenidos fueron 910 y 898 respectivamente. La predicción a largo plazo se realizó utilizando recursivamente el predictor a un paso, alcanzándose una predicción a largo plazo en el conjunto de entrenamiento con RMSE 2292, y con un RMSE de test de 1961. Entonces se procedió a extender el conjunto de entrenamiento con la aproximación a un paso de la serie de entrenamiento. Se utilizaron los mismos hiperparámetros para reentrenar el predictor LS-SVM a un paso (con el conjunto de entrenamiento extendido). La predicción a largo plazo obtenida para el conjunto de entrenamiento obtuvo un rendimiento con RMSE = 1598, y para el conjunto de test el RMSE fue de 1054. La figura 7.7 muestra la predicción a largo plazo para los conjuntos de test y entrenamiento, utilizando los datos extendidos de entrenamiento para el flujo del rio Snake utilizando el enfoque de las LS-SVMs.

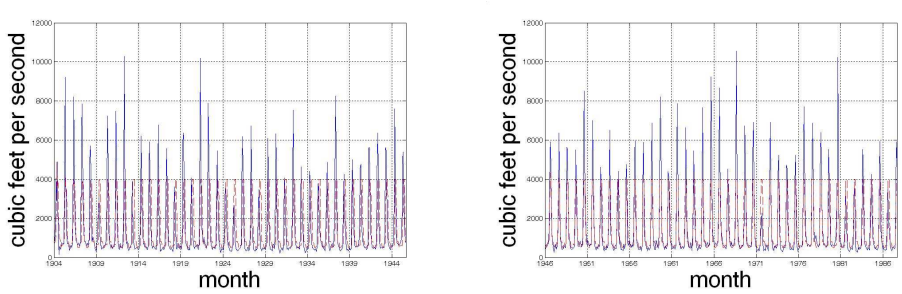


Figura 7.7. *a) Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para los datos de entrenamiento utilizando el conjunto de datos extendido para el problema del flujo de un rio utilizando LS-SVMs. RMSE de entrenamiento 1598. b) Predicción a largo plazo original (línea punteada) y aproximada (línea sólida) para los datos de test utilizando el conjunto de datos extendido para el problema del flujo de un rio utilizando LS-SVMs. RMSE de test 1054.*

De nuevo se realizó el proceso iterativo, generando más datos de entrenamiento al obtenerse nuevas predicciones a un paso del conjunto de entrenamiento mediante el modelo LS-SVM reentrenado. Para estos reentrenamientos posteriores, el RMSE de la predicción a largo plazo recursiva mejoró ligeramente durante 4 iteraciones más. No obstante, para las LS-SVMs, este proceso propuesto incrementa el número de datos de entrenamiento, llevando por tanto a un cuello de botella computacional en el entrenamiento del modelo. La complejidad computacional del entrenamiento en LS-SVMs se incrementa de forma cuadrática con el número de puntos en el conjunto de entrenamiento [Suykens et al., 2002]. Existen varios enfoques en la literatura para evitar este problema de cuello de botella computacional en los métodos de kernel al aumentar el número de puntos [Schoelkopf and Smola, 2002], [Suykens et al., 2002]. Por ejemplo, las LS-SVM de tamaño fijo es un método para tratar con problemas de gran tamaño utilizando LS-SVMs [Suykens et al., 2002]. Otros métodos

pueden encontrarse en [Jung et al., 2005].

7.5. Conclusiones

Este capítulo presenta una técnica general para mejorar los resultados de predicción recursiva para predicción de series temporales a largo plazo. Para introducir y comprobar la eficacia del método, se han aplicado dos metodologías de aprendizaje totalmente diferentes, el modelo TSK difuso TaSe y las LS-SVMs, a dos ejemplos de series temporales, la serie artificial bien conocida de Mackey-Glass y la serie del flujo de un rio. En general, cuando se realiza una predicción a largo plazo de una serie temporal dada, con un modelo de predicción cualquiera, se obtiene una copia sin ruido de la serie original. Las simulaciones presentadas han mostrado cómo el rendimiento de un predictor a un paso puede mejorarse cuando se utiliza dicha salida para realizar predicción recursiva a largo plazo. Esta mejora se obtiene utilizando la predicción a un paso sin ruido como una nueva parte ficticia de la serie temporal, que se añade a los datos de entrenamiento, y reentrenando el modelo con este conjunto de datos de entrenamiento extendido. En las simulaciones, ambos modelos han mejorado su predicción recursiva a largo plazo para ambas series temporales. El enfoque de las LS-SVM obtiene en las simulaciones realizadas, unos excelentes resultados de aproximación, gracias a su mayor nivel de flexibilidad, y al control de la varianza y bias mediante el parámetro de regularización. Sin embargo, el modelo LS-SVM sufre de la maldición de la dimensionalidad en el número de puntos, y en principio, carece de interpretabilidad. El modelo TaSe obtiene un modelo preciso e interpretable, compuesto de un conjunto de reglas cuyos consecuentes son el desarrollo en serie de Taylor de la salida del modelo en torno a los centros de las reglas. Sin embargo, como modelo que realiza un particionamiento en grid del espacio de entrada, éste puede difícilmente aplicarse a problemas complejos con un gran número de variables de entrada. Como trabajo futuro, se seguirán dos tendencias con el objetivo de obtener mejores predictores recursivos. Por un lado, se pretende explorar otras variaciones a la técnica general presentada en la sección 7.4. Por otro lado, se analizarán posibles optimizaciones en la eficacia para el problema general presentado en la sección 7.3.

Capítulo 8

Backward Input Variable Selection for Function Approximation Problems Using Mutual Information

Input variable selection is a key preprocess step in any I/O modeling problem. In general, the elimination of unneeded irrelevant or redundant variables in a problem, reduces the learning complexity, increases the generalization performance, and makes the interpretation and usability of the obtained models easier. Information theory provides a robust theoretical framework for performing input variable selection thanks to the concept of mutual information. This chapter proposes a backward variable selection methodology for function approximation that is based on the mutual information measure. Among the advantages that the proposed methodology presents, we stand out the possibility of avoiding of the curse of dimensionality problem that the mutual information estimators can present, making the method feasible for large problems without losing its robustness. The estimation of the mutual information and the selection of the mutual information estimator parameter, are furthermore improved by using resampling methods as proposed in previous works. The performance of the proposed methodology is shown through significant examples of large function approximation problems, such as spectrometric data and time series prediction problems.

8.1. Introduction

Input variable selection is a very important preprocessing step in any supervised or unsupervised learning problem. Most of the paradigms from the machine learning field suffer from the curse of dimensionality problem, i.e. the problem caused by the exponential increase in volume associated with adding extra dimensions to a (mathematical) space [Bellman, 1961]. Therefore, the existence of a number of irrelevant or redundant input variables can lead to overfitting and to a loss of generalization of the model [Haykin, 1998]. Furthermore in models that suffer from the curse of dimensionality in computational complexity in the number of input variables, like grid-based fuzzy models [Herrera et al., 2005f], input variable selection becomes essential.

In general there are two types of variable selection methods. Filter methods try to select the variables in a preprocess step with the only information that the I/O values bring. Wrapper methods employ the learning methodology that is going to be used, in order to select the subset of variables that brings the best performance. The first ones in general are more efficient techniques, but they can fail in not considering the final methodology that is being used to learn the model, since in principle they don't receive a feedback of the real performance of the variables selected. Wrapper methods on the other side can

be very effective in selecting an optimal subset of features, but the computational cost of evaluating many possible subsets of input variables, by learning a model for each of them, can be overwhelming. Filter methods are normally preferred due to the computational cost reduction that can be obtained, at least as a preprocessing procedure to rank the importance of the variables.

This work deals with a filter method for feature selection on regression problems based on the Mutual Information (MI) measure from Shannon's Information Theory. Filter methods, as mentioned, are used as a completely separated preliminary step to the Input/Output (I/O) modeling problem. For input variable selection, information theory offers a good theoretical background for variable filtering thanks to the concepts of entropy and mutual information among variables [Cover and Thomas, 1991]. In regression problems, the input and output variables can take any value within a continuous domain, and additional techniques have to be used to estimate the probability distribution comparing to classification problems [Bonnlander and Weigend, 1994]. This drawback becomes even more pronounced specially when the number of data points is low in relation to the number of input variables (DNA Micro-arrays, spectra data, etc.).

Among the techniques to estimate the probability density functions (PDF) we can find histogram and kernel-based PDF estimators [Bonnlander and Weigend, 1994]. A number of entropy estimators based on the k-nearest neighbor statistics also exist [Rossi et al., 2006]. Recently they have been extended to the mutual information estimation by Kraskov et al [Kraskov et al., 2004], [MIL,]. This estimator will be used in this work. A nice property that it presents, is that it can be used easily for sets of variables. Using the estimation of mutual information between two or more variables, a number of algorithms could be designed, according to typical strategies such as forward selection [Battiti, 1994], [François et al., 2007], backward selection [Koller and Sahami, 1996], [Herrera et al., 2006], or subset selection [Sorjamaa et al., 2005], [Rossi et al., 2006].

Forward selection methods start with an empty subset $X_G = []$ and relevant variables are added in an iterative process. This type of strategy can work very well when the input variables are near to be independent of each other. However, in cases in which there are variables that have a joint influence on the output, it could happen that a certain subset of variables does not give information to the output variable separately but altogether, and in this case this strategy might not provide optimal results. Backward selection on the other hand starts with the complete set of variables $X_G = X$ and iteratively eliminates variables from X_G. This approach overcomes the previous mentioned problem, and in principle is the best strategy to obtain a more robust subset of relevant features. However, it requires in principle the evaluation of the relevance of large groups of features, that is a more complicated and less reliable task. Both in forward and backward feature selection, a very difficult issue is when to stop the iterative process and return an optimal subset of features. Subset selection would directly obtain the best subset of features for a given problem according to a given criterion. Nevertheless, the number of needed evaluations to test all possible subsets is 2^n where n is the number of input variables, and this is already prohibitive for medium complexity problems.

This work proposes the use of an efficient backward elimination process for feature selection, as a robust way to obtain an optimal subset of relevant variables for a given regression problem. The proposed method uses as criterion a k-nearest neighbors MI estimator [Kraskov et al., 2004], improved by using resampling methods [François et al., 2007]. Thanks to the behavior of the methodology, that makes use of the Markov blanket (MB) concept [Pearl, 1988], [Koller and Sahami, 1996], [Herrera et al., 2006], estimation of MI among large groups of variables is avoided, thus diminishing the curse of dimensionality problem in the k-nearest neighbors estimator and increasing the expected robustness of the

proposed methodology for large problems. In order to return an optimal subset of variables, it is proposed to evaluate the returned subset of variables for different MB sizes, and return the subset of variables for which the performance is best with respect to to the performance using the whole set of variables.

The rest of the chapter is organized as follows. Section 8.2 briefly explains the mutual information concept for continuous variables. It also reviews how to perform the parameter selection for the k-nearest neighbors MI estimator and how to obtain more robust estimations using resampling methods according to the work by D. François et al. [François et al., 2007]. Section 8.3 presents the backward variable selection method proposed in this work, with discussions about its robustness, efficiency and parameter selection. Section 8.4 shortly reviews the basics of LS-SVMs [Suykens et al., 2002], that is the learning methodology used to check the effectiveness of the proposed approach. Section 8.5 presents four significant examples of application of the variable selection methodology. Finally, section 8.6 presents the main conclusions drawn from the study.

8.2. k-Nearest Neighbors Mutual Information Estimation Using Resampling Methods

8.2.1. Mutual information definition

Given a multiple input single-output (MISO) function approximation or classification problem, with input variables $X = [x_1, x_2, \ldots, x_n]$ and output variable $Y = y$, the main goal of a modeling problem is to reduce the uncertainty on the dependent variable Y. The objective of a preprocessing filter variable selection approach is to select the smallest subset of variables $X_G \subset X$ that in principle fulfils this reduction of uncertainty. According to the formulation of Shannon, and in the continuous case, the uncertainty on Y is given by its entropy, that is defined as

$$H(Y) = -\int \mu_Y(y) \log \mu_Y(y) dy, \tag{8.1}$$

considering that the marginal density function $\mu_Y(y)$ can be defined using the joint PDF $\mu_{X,Y}$ of X and Y as

$$\mu_Y(y) = \int \mu_{X,Y}(x, y) dx. \tag{8.2}$$

Given that we know X, the resulting uncertainty of Y conditioned to known X is given by the conditional entropy, defined by

$$H(Y|X) = -\int \mu_X(x) \int \mu_Y(y|X = x) \log \mu_Y(y|X = x) dy dx. \tag{8.3}$$

The joint uncertainty on the $[X, Y]$ pair is given by the joint entropy, defined by

$$H(X, Y) = -\int \mu_{X,Y}(x, y) \log \mu_{X,Y}(x, y) dx dy. \tag{8.4}$$

The mutual information (also called cross-entropy) between X and Y can be defined as the amount of information that the group of variables X provide about Y, and can be expressed as

$$I(X, Y) = H(Y) - H(Y|X). \tag{8.5}$$

In other words, the mutual information $I(X,Y)$ is the decrease of the uncertainty on Y once we know X. Due to the mutual information and entropy properties, the mutual information can also be defined as

$$I(X,Y) = H(X) + H(Y) - H(X|Y), \tag{8.6}$$

leading to

$$I(X,Y) = \int \mu_{X,Y}(x,y) \log \frac{\mu_{X,Y}(x,y)}{\mu_X(x)\mu_Y(y)} dxdy. \tag{8.7}$$

Thus, only the estimate of the joint PDF between X and Y is needed to estimate the mutual information between two groups of variables. Estimating the joint probability distribution can be performed using a number of techniques. As mentioned before, histograms and kernel density estimators have been used for this purpose. In the recent literature, the k-nearest neighbors MI estimator by Kraskov et al. [Kraskov et al., 2004], [MIL,] has shown to be effective and have shown to be less sensitive to dimensionality [Kraskov et al., 2004], [Dijck and Hulle, 2006], [Benoudjit et al., 2004], [Rossi et al., 2006], therefore being more convenient for the present problem. A software for MI estimation using k-nearest neighbors can be found in [MIL,].

8.2.2. Parameter selection for the k-nearest neighbors estimator using resampling methods

The estimation of the MI using the k-nearest neighbors can present a strong dependency in the value of k [François et al., 2007]. Selecting an appropriate value for this parameter is essential in order to obtain robust results in the selection process. In some works it is recommended [Harald et al., 2004] that for a tradeoff between variance and bias, a mid-range value for k (k = 6) should be used. However, recent works have proposed the use of resampling methods in order to select the best value for the parameter k [François et al., 2006], [François et al., 2007] in the k-nearest neighbors MI estimator.

L-fold resampling

L-fold resampling is very similar to the L-fold crossvalidation scheme used for validating prediction models, except that it is used in an unsupervised manner. Given z_1 and z_2, realizations of Z_1 and Z_2 respectively, and some statistic Θ involving z_1 and z_2, the L-fold resampling method consists in computing L estimates $\hat{\Theta}_l$ of Θ where one (or several) data element(s) has(ve) been removed from the analysis. Typically, the sample is partitioned into L clusters of roughly equal size, and the statistic is estimated L times on the sample from which the L-th cluster was excluded. The average of those estimations is often found to be a more robust estimator of Θ, while the variance of the estimations gives an idea of the sensitivity of the estimator to the particular sample. Resampling approaches similar to L-fold resampling (Jackknife, bootstrap, etc.) have also been used to get better estimates of the mutual information [Zhou et al., 2004].

The permutation test

The permutation test or randomized resampling [Good, 1994] is a nonparametric hypothesis test over some estimated statistic $\hat{\Theta}$ involving z_1 and z_2. The statistic $\hat{\Theta}$ can be a difference of means in a classification context, or a correlation, or, as in this context, a

mutual information. Let $\hat{\Theta}$ be the estimation of the statistics for the given z_1 and z_2, both vectors of size m drawn from p_{Z_1} and p_{Z_2} , respectively. The aim of the test is to answer the following question: how likely is the value $\hat{\Theta}$ given the vectors z_1 and z_2 if we suppose that Z_1 and Z_2 are independent? In particular, the value of the mutual information under such hypothesis should be zero.

The permutation test considers the empirical distribution of z_1 and z_2 to be fixed, as well as the sample size. The random variable of interest is the value of the statistic Θ. In such a framework, the distribution of $\hat{\Theta}$ is the set of all values of $\hat{\Theta}_i$ for all $m!$ possible permutations of the elements of the vector z_1, or, equivalently, all permutations of the elements of the vector z_2. The P-value associated to the test is the proportion of $\hat{\Theta}_i$ that are larger than the value of $\hat{\Theta}$ estimated with z_1 and z_2 without permutation. In practice, it is not necessary to perform all $m!$ permutations. Several tens or hundreds of them are randomly performed. In this case, the exact P-value cannot be known but a $95\,\%$ confidence interval around the observed P-value can be estimated. The obtained P-value informs about the probability of the hypothesis that Z_1 and Z_2 are independent.

Estimation of the parameter k for the k-nearest neighbors MI estimator

In order to determine the optimal value for k in the estimation of the MI, the notion of optimality must be explicitly defined since there is no obvious criterion that we could maximize or minimize. Since we are designing a filtering method, we do not want to optimize the number of neighbors with respect to the performances of a prediction model built with the variables chosen by the procedure. The idea is to discriminate between features that are relevant for the problem and features that are useless. We therefore consider the optimal value of k to be the value for which the separation between the relevant features and an independent feature is maximum [François et al., 2007]. Since the estimator of the mutual information has some variance, it is important to take this variance into account in measuring the separability. If we had access to the distribution of the mutual information estimate over the data, we could calculate a separation between $MI(x, Y)$ and $MI(u, Y)$ (considered as random variables) for an important feature x and an useless feature u.

In practice however, the true distribution of $MI(X, Y)$ is unknown. In the work [François et al., 2007], a combined L-fold/permutation test is proposed to try to estimate the bias and the variance of the estimator for relevant features and for independent ones. Consider x_i a feature that is supposed to be relevant to predict Y. Two resampling distributions are built for both $MI(x_i, Y)$ and $MI(x_i^\pi, Y)$ where x_i^π denotes a randomized x_i that is made independent from Y through permutations. This is done by performing several estimations of (i) the mutual information between x_i and Y and (ii) the mutual information between a randomized version of x_i and Y, using several non-overlapping subsets of the original sample, in a L-fold resampling scheme (with L around 20 or 30). The procedure results in two samples of estimates of $MI(x_i, Y)$ and $MI(x_i^\pi, Y)$. The optimal value of k for the k-nearest neighbors MI estimator is the one that best separates those two distributions, for instance according to a Student-like measure:

$$t_{i,k} = \frac{\mu - \mu_\pi}{\sqrt{\sigma^2 + \sigma_\pi^2}}, \tag{8.8}$$

where μ and σ^2 represent the mean and variance of the cross-validated distribution of $MI(x_i, Y)$, μ_π and σ_π^2 are those of the cross-validated distribution of $MI(x_i^\pi, Y)$. The optimal k for all features is chosen as the one corresponding to the largest value of $t_{i,k}$ over all values of k over all features. This way, features that are useless do not participate in the choice of the optimal value for k.

8.2.3. Estimation of the mutual information using L-fold resampling

Given that an optimal value for the k parameter in the k-nearest neighbors MI estimator is obtained, the MI estimator could be improved by using a L-fold resampling scheme. As explained before, the average of the L-fold estimations is often found to be a more robust estimator of the MI, and other approaches such as bootstrap have been used to get better estimates of the mutual information [Zhou et al., 2004]. Obviously the computational cost of performing L-fold resampling to estimate the MI among two groups of variables, increases by L the computational cost order of the designed methodology. However, due to the large number of comparisons of MI estimations among groups of variables that have to be performed along a variable selection procedure, in some cases it can be very convenient to obtain a more robust estimator in spite of the computational cost it entails.

8.3. Backward Variable Selection for Function Approximation Problems Avoiding the Curse of Dimensionality in the MI Estimation

When dealing with a high dimensional variable selection problem, three main strategies can be taken to perform this task. Among them, subset selection implies an exhaustive search of the optimal subset of variables that brings more information about the output variable, but this is unfeasible (2^n possible subsets) for medium or large dimensional problems. In relation to forward feature selection, as previously mentioned, in can work very well when input variables are near to be independent of each other. However, in cases in which there are variables that have a joint influence on the output, it can occur that a certain subset of variables does not give information w.r.t the output variable separately but they do when considered together. In this case this strategy fails to provide optimal results.

In any case, strategies performing comparisons among large groups of variables suffer from the problem of the curse of dimensionality in the MI estimator. Although the k-nearest neighbors MI estimator [Kraskov et al., 2004] is less sensitive to this problem, as the number of input dimensions increases, this estimator shows to systematically decrease the MI of a group of input variables with respect to the output variable [François et al., 2007]. Thus, for subset feature selection, comparisons among different large subsets is not robust. In backward selection the starting point for the selection process is already the whole set of features, which makes an MI estimation-based variable selection algorithm less reliable. And in forward selection, although for the first iterations the results can be optimal, as the number of input variables to be selected gets higher, comparisons among different MI estimations might also not be robust.

In this work, a different approach is proposed that avoids performing MI estimations among big groups of variables, and thus it is able to bring more robust results for large dimensional problems. The objective of any feature selection process is to select a right subset $X_G \subset X$ that comprises the same information that X has of Y. That is, we want to find a subset $X_G \subset X$ such that

$$I(X, Y) \cong I(X_G, Y). \tag{8.9}$$

The proposed method is an iterative backward selection process. A variable x_i is eliminated if and only if it does not bring information about the output variable, considering that we keep the remaining set of variables $x_G - \{x_i\}$. However, the determination of the condition $I(\{X_G \cup \{x_i\}\}, Y) \cong I(X_G, Y)$ is practically unfeasible. To help on the

estimation of the relevance of an input variable in the backward selection process, the concept of Markov blanket [Pearl, 1988], adapted for this problem [Koller and Sahami, 1996], [Herrera et al., 2006], will be used. We will suppose that this concept can be applied for the specific variable selection problem we deal with. The use of Markov blankets implies strong conditioning among the variables. Nevertheless it will be relaxed to help us performing the variable selection.

Definition: Let M be a subset of variables taken from X that does not contain x_i. We say that M is a Markov blanket for x_i if $I(\{M \cup x_i\}, X - \{M \cup x_i\}) \cong I(M, X - \{M \cup x_i\})$

Corollary: Let X_G be a subset of variables and x_i a variable in X_G. Assume that a subset M of X_G is a Markov blanket of x_i. Then $I(X_G, Y) \cong I(X_G - \{x_i\}, Y)$.

As we can see, the Markov blanket condition is stronger than the one we desire. It can be even a harder problem to find a Markov blanket of a variable in a set of variables than the variable selection problem itself. However it brings the idea on how to facilitate a backward variable selection procedure. The difficult evaluation of $I(\{X_G \cup \{x_i\}\}, Y) \cong I(X_G, Y)$ to eliminate variables, will be transformed into estimating, for each x_i a Markov blanket $M_i \subset X_G$, and into evaluating if $I(\{M_i \cup \{x_i\}\}, Y) \cong I(M_i, Y)$. Those x_i can be removed from the current X_G without loss of information.

As already mentioned, calculating the Markov blanket of a variable in a set, or even trying to know if it exists is a very difficult task. Therefore it will be assumed that the Markov blanket exists, and we will derive a heuristics to guess the variables M that compose the Markov blanket of any variable x_i. The Markov blanket candidates for each variable x_i will be calculated as those that bring more information on that variable, that is $M_i = \{x_{j1} \dots x_{jp} \in X / I(x_j l, x_i) > I(x_v, x_i) \forall v \notin \{j_1 \dots j_p\}\}$. For the sake of simplicity, the size p of the Markov blankets will be fixed for all the variables; the operation of the designed algorithm with respect to this parameter will be discussed later. This formation of MB candidates is not optimal and could be improved by properly selecting the optimal candidates in a forward iterative search $M_i = \{x_{j1} \dots x_{jp} / I([x_{j1}, \dots, x_{jp}], x_i)\ is\ \max_{x_i}\}$. However this strategy, considerably increases the computational cost of the algorithm. For the definition of the algorithm and for the examples presented in this work, the simpler construction of the MB candidates has been considered, as it shows to work properly in the examples presented.

A sketch of the algorithm stays as follows:

1. Calculate the MI between every two input variables $MI_{inputVariables}(i, j) = I(x_i, x_j)$

2. Starting from the complete set of input variables $X_G = X$, iterate:

   a) For each variable x_i, let the candidate Markov blanket M_i be the set of p variables in X_G for which $MI_{inputVariables}(i, j) = I(x_i, x_j)$ is highest.

   b) Compute for each x_i

   $$Loss_i = I(\{M_i \cup x_i\}, Y) - I(M, Y). \tag{8.10}$$

   c) Choose the x_i' for which $Loss_i'$ is lowest and eliminate x_i' from X_G.

3. Stop the algorithm when the $Loss_i'$ function show that there is loss of information when a variable is eliminated.

On how to stop the algorithm, it is easy to check if the $Loss'_i$ function returns a positive value, meaning that variable x_i is not subsumed by its MB. However the size of the MB can play a very important role in the robustness of this approach. Next subsections discusses on how an appropriate value of p for a given problem could be selected, and proposes a simple wrapper approach to perform the right selection of the size p of the MB, providing the optimal subset of features selected by the algorithm. Later the computational cost of the whole methodology is also discussed .

8.3.1. Selection of the parameter p

In the previous algorithm, the Markov blanket selected for every variable is just an approximation and the number p of variables is fixed a priori. With respect to parameter p, it is expectable that higher values bring better chances that the pseudo-Markov blankets taken subsume real Markov blankets of the variables. However, as the size of the MB increases, the reliability of the MI estimator may vanish [Kraskov et al., 2004], [François et al., 2007]. Also for problems with a low number of samples in the training data set, the generalization performance of the MI estimator is expected to be more sensitive to higher values of p.

Nevertheless, in order to use an appropriate value of p for a given problem, the matrix $MI_{inputVariables}(i,j)$ calculated in the first step of the algorithm brings the information of the interdependency of the input variables. Thus, on the one hand, for problems in which the input variables show to be independent of each other (values for $MI_{inputVariables}(i,j) \cong 0$), a value of $p = 1$, or even, empty MB (thus a traditional backward or forward selection procedure [François et al., 2007]) should be used. On the other hand, problems showing a high interdependency among the input variables, and with a large enough set of samples, values from $p = 2$ should be considered.

In the examples performed, it has been seen, and it will be shown in section 5 in this work, that values of $p = 3$ will seldom be useful. In fact values of $p = 2$ and $p = 1$ show to be optimal in all cases. Already with $p = 3$ comparisons among MI among groups of four and three variables (see equation 8.11) are needed, thus diminishing the reliability of the proposed approach due to the problem already commented that the MI diminishes as more variables (even relevant ones) are considered.

The effectiveness of the MB estimations in any case might depend on the specific problem. However, considering the output of the variable selection algorithm as a ranking of relevance of the input variables, the proposed algorithm is a robust way to identify the irrelevant and redundant variables and to set up properly the importance of the variables in a given problem [Herrera et al., 2006].

Supposing that an appropriate value of p is considered for a given problem, the algorithm should stop whenever the final step inside the main loop of the algorithm ("*Choose the x'_i for which $Loss'_i$ is lowest and eliminate x'_i from X_G*"), identifies that there is loss of information when eliminating a variable x'_i. The $Loss'_i$ function as seen indicates the loss of information that is committed when the variable x'_i is eliminated, according to its MB. This way, as the MBs of the variables are subsuming the information they contain with respect to the output variable Y, the procedure will stop when the value of $Loss'_i$ indicates that there is in fact loss of information, that is, when it's above 0. However, the right value for p for a given problem is in principle unknown. If too small MBs are selected, the function $Loss'_i$ will take values higher than 0 too in advance, as variables might not be sufficiently covered by their MB, and too large MBs can lead to unexpected negative values of the $Loss'_i$ function, due to the k-nearest neighbors MI estimator trend to decrease the MI estimation as more variables are considered.

In general the problem of when to stop an iterative variable selection procedure is a very important issue with very difficult solution. In [François et al., 2007] it is proposed a very convenient approach to stop the forward iterative process, based on the permutation test reviewed in section 8.2. However these type of criteria can present the same problem previously commented of the curse of dimensionality problem, that can make the robustness of the MI estimator vanish as the number of input variables selected increases. Furthermore, for the specific work in [François et al., 2007], the permutation test can present problems in case we deal with variables very interrelated to each other such as spectral data problems [Herrera et al., 2006]: using the k-nearest neighbors approach, the addition of a permuted variable $\tilde{x_p}$ to a subset of very interrelated variables $x_1 \dots x_{p-1}$ highly diminishes the MI estimation using the k-nearest neighbors approach, i.e., $MI([x_1 \dots x_{p-1}], y]) \gg MI([x_1 \dots x_{p-1}, \tilde{x_p}], y])$. This might mislead in some cases the selection of the optimal number of input variables to perform the approximation.

In several cases as explained, it is very difficult to obtain a robust stopping criterion for very high dimensional problems, rather than stopping after a fixed number of input variables is selected. In other cases, sub-optimal subsets can be obtained using filtering criteria, obtaining very good approximation results as has been reported in previous works [François et al., 2007], [Herrera et al., 2006], [Rossi et al., 2006]. However, optimality is in many cases not guaranteed, and according to the experience, it is convenient to perform a hybrid wrapper-filter selection approach and utilize the learning methodology that is going to be used to evaluate the performance of the currently obtained subset, before performing a final launch of the learning methodology using the selected subset of input variables.

From this principle, since for every value of p we obtain a different subset of variables that keep the desired information w.r.t. the output variable according to the size of MB, we propose here to perform a wrapper selection of the value of p, and retain the one that fulfils the desired properties for a given problem (with the associate subset of selected variables). That is, for different values of p, different subsets of variables fulfilling the condition $I(X, Y) \underset{MB_p}{\cong} I(X_G, Y)$ are obtained (according to their respective MBs). The final subset of variables should be then selected as the one that provides the best training-validation performance according to the learning methodology to be used. This last step can be seen therefore as a wrapper selection technique for the parameter p.

Next subsection discusses the computational cost of the proposed approach. As it will be discussed, the execution of the basic algorithm for different sizes of Markov blanket candidates implies an affordable computational cost due to the operation of the loop in the algorithm. As mentioned, and as it will be corroborated in the simulations, we propose a range of values for p from $1 \dots P$ where $P = 3$.

8.3.2. Computational cost of the algorithm

In general, backward and forward procedures can be associated to a large computational cost, since in each iteration, the MI has to be estimated for every variable with respect to the current situation of the process.

For this procedure, the estimation of the MI has to be obtained for every pair of variables, according to the heuristics performed to obtain the MB candidates for each variable x_i. However, during the elimination process, after the first iteration in which the $Loss$ function is calculated for every input variable, the values for $Loss_i$ only change for those variables for which their MB has changed. That is, when a variable x_i' is eliminated from X_G, the only values of $Loss_i$ that have to be recalculated are those M_j for which $x_i' \in M_j$. Therefore, the previous algorithm for a given p can be implemented as follows

1. Calculate the MI between every two input variables $MI_{inputVariables}(i,j) = I(x_i, x_j)$

2. $M_{toCalculate} = \{1, \ldots, n\}$

3. Starting from the complete set of input variables $X_G = X$, iterate:

   a) For each variable x_i, let the candidate Markov blanket M_i be the set of p variables in X_G for which $I(x_i, x_j)$ is highest.

   b) Compute for each x_i where $i \in M_{toCalculate}$

   $$Loss_i = I(\{M_i \cup x_i\}, Y) - I(M, Y). \tag{8.11}$$

   c) Choose the x'_i for which $Loss'_i$ is lowest and eliminate x'_i from X_G. $M_{toCalculate} = \{j / x'_i \in M_j\}$.

This leads to an average of $Loss$ function evaluations equal to n for the first iteration + $n_{elim} * p$ for the rest of iterations, where n_{elim} is the number of input variables eliminated, and p is the size considered for the MB candidates. Every $Loss$ function calculation implies 2 MI estimations. Thus, adding the first loop to calculate all the $I(x_i, x_j)$, and considering different values of p, leads to the estimation of the MI

$$(n^2 + n)/2 + \sum_{p=1}^{P} 2 * (n + n_e lim * p) \tag{8.12}$$

times.

The computational cost also depends on the use of the L-fold resampling for a better estimation of the MI (between 20 and 30 according to [François et al., 2007]), and on the process of finding the optimal value for k on the k-nearest neighbors estimator for the MI [François et al., 2007].

8.4. Least-Squares Support Vector Machines

This section presents a brief introduction to the learning methodology used in the simulations. LS-SVMs are reformulations to standard SVMs, closely related to regularization networks and Gaussian processes but additionally emphasize and exploit primal-dual interpretations from optimization theory. LS-SVMs are a paradigm specially well suited for function approximation problems [Suykens et al., 2002].

The LS-SVM model [Suykens et al., 2002] is defined in its primal weight space by

$$\hat{y} = \vec{\omega}^T \phi(\vec{x}) + b \tag{8.13}$$

where $\vec{\omega}^T$ and b are the parameters of the model, $\phi(\vec{x})$ is a function that maps the input space into a higher dimensional feature space, and $\vec{x}$ is the n-dimensional vector of inputs x_j. In Least Squares Support Vector Machines for function approximation, the following optimization problem is formulated,

$$\min_{\vec{\omega}, b, e} J(\omega, e) = \frac{1}{2}\vec{\omega}^T\vec{\omega} + \gamma\frac{1}{2}\sum_{i=1}^{N} e_i^2 \tag{8.14}$$

subject to the equality constraints (inequality constraints in the case of SVMs)

$$e_i = y_i - \hat{y}(\vec{x}_i), i = 1 \dots N \tag{8.15}$$

Solving this optimization problem in dual space, leads to finding the λ_i and b coefficients in the following solution

$$\hat{y} = \sum_{i=1}^{N} \lambda_i K(\vec{x}, \vec{x}_i) + b \tag{8.16}$$

where the function $K(\vec{x}, \vec{x}_i)$ is the kernel function defined as the dot product between the $\phi(\vec{x})$ and $\phi(\vec{x}_i)$ mappings.

In case we consider Gaussian kernels, the kernel function $K(\vec{x}, \vec{x}_i)$ takes the form

$$K(\vec{x}, \vec{x}_i) = \exp\left[-\left(\frac{\|\vec{x} - \vec{x}_i\|}{\sqrt{2}\sigma_i}\right)^2\right] \tag{8.17}$$

where σ_i is the width of the kernel, that together with the regularization parameter γ, are the hyper-parameters of the problem. Note that in the case in which Gaussian kernels are used, the models obtained resemble Radial Basis Function Networks (RBFN); with the particularities that there is an RBF node per data point, and that overfitting is controlled by a regularization parameter instead of by reducing the number of kernels [Rossi et al., 2006]. In LS-SVM, the hyper-parameters of the model can be optimized by cross-validation. Nevertheless, in order to speed-up the optimization, a more efficient methodology by Lendasse et al. can be found in [Lendasse et al., 2005].

8.5. Simulations

In this section, the proposed methodology is applied to two important types of problems, presenting a strong need of performing input variable selection, such as spectrometric data and time series prediction problems. This type of data form a relatively small number of vectors with a large number of exploitable variables. However, only a small subset of them is usually required to build a good model. On the other hand, in time series prediction problems, it is a hard task to perform variable selection since any previous value of the series can have an important influence on the current value, that needs to be taken into account. In both types of problems, there is a strong interrelation among the input variables, that makes them suitable for the proposed methodology. For the examples presented in this section, the MI estimator in [MIL,] has been used. A LS-SVM Matlab toolbox can be found in [LS-,].

8.5.1. Spectrometric data

In this subsection we present the application of the variable selection method proposed in this work to two significant spectrometric data examples. The error measure used in these examples is the Normalized Mean Square Error, NMSE [Rossi et al., 2006].

Tecator Meat sample data set

The example considered has been taken from [Rossi et al., 2006] and is a spectrometric data set coming from the food industry. The “tecator meat” data set consists of 100

spectral input variables and one output variable (the original data set has three, but we consider only the fat content). It relates to the determination of the fat content of meat samples analyzed by near infrared transmittance spectroscopy. This data set contains 172 training spectra and 43 test spectra. As in [Rossi et al., 2006], and in order to make comparisons fair, the spectra are reduced to zero mean and unit variance. Also to avoid losing information, the original mean and standard deviation are kept as two additional variables. A selection of training spectra is shown in figure 8.1.

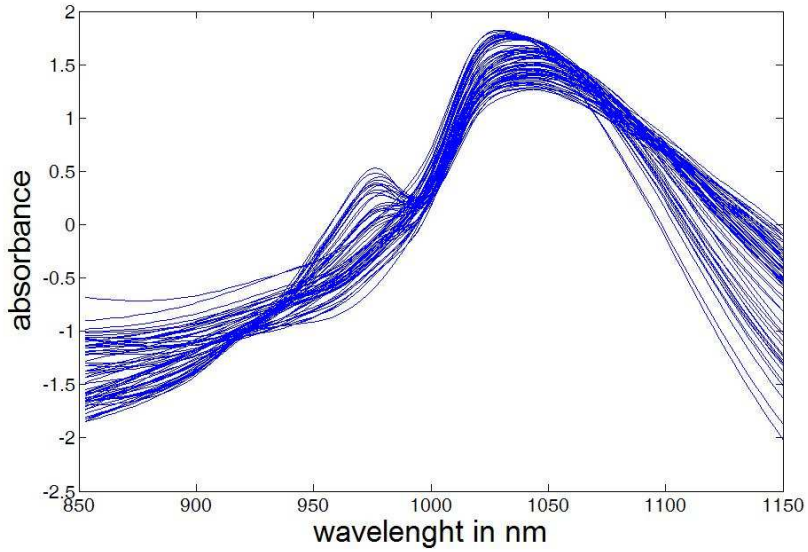


Figura 8.1. *A selection of the spectra from the "tecator meat" data set.*

First, the value of k is optimally selected as is explained in section 2.2. The optimal value selected is $k = 6$, that is the one that corresponds to the highest value for the Mutual Information estimation (using the permutation test) for any input variable of the problem. At the first step of the algorithm, the matrix $MI_{inputVariables}(i, j)$ is built, and for this example it shows that there is a strong interrelation among the input variables. As it can be seen from figure 8.1, the input data of the spectra instances are very similar, just with some differences around some wavelength ranges. Considering the very low number of data points that are available for this problem (172 training data samples) the algorithm will be executed for p ranging from $1 \ldots 3$. The evolution of the $Loss$ function for the three values of p is shown in figure 8.2. From this graph we can see how much information is lost as variables are discarded in the iterative process for different sizes of MB.

According to the execution of the algorithm, for $p = 1$, 80 variables are selected of the 102, for $p = 2$, 20 variables are selected, and for $p = 3$, 13 variables are selected. The performance of a LS-SVM model trained for each subset of variables is shown in table 8.1. The results obtained clearly show that the best performance is obtained for $p = 2$, since for a much lower number of input variables, a similar performance is obtained. Furthermore from the evolution of the $Loss$ function, we observe that the variance of the MI estimator highly increases for $p = 3$ comparing to the evolution for $p = 2$ and $p = 1$. This supports the fact that for too high values of p, the reliability of the MI estimations is lost, due to the curse of dimensionality problem and therefore attention has to be paid to this phenomenon.

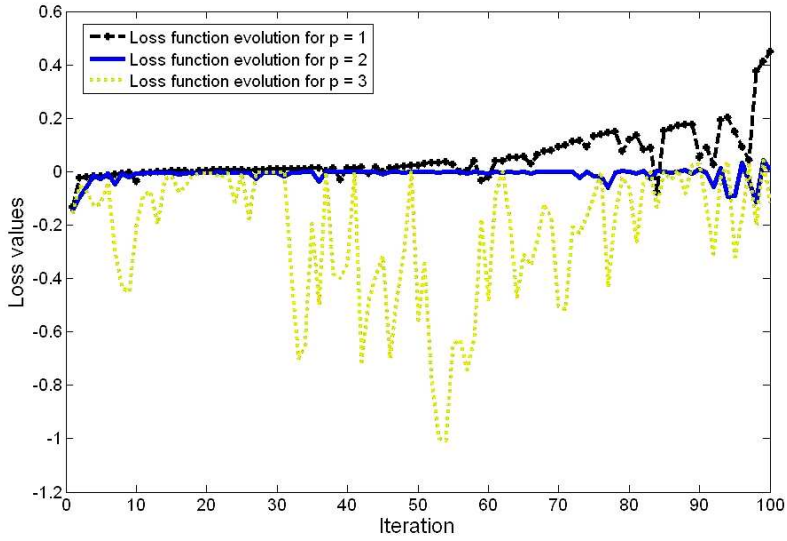


Figura 8.2. *Evolution of the $Loss'$ function in the execution of the algorithm for $p = 1 \ldots 3$*

Tabla 8.1. *Performance obtained for different values of p for the Meat spectrometric data example*

p	Number of Variables	Training NMSE	Test NMSE
Original set X	102	0.0006	0.0016
1	80	0.0007	0.0015
2	20	0.0008	0.0016
3	13	0.0022	0.0044

Figure 8.3 shows the evolution of the NMSE for different LS-SVM models, trained using 10-fold cross validation and grid search in order to find optimal values for the hyperparameters, as the least relevant variables are eliminated according to the ranking of variables obtained for the selected value of $p = 2$. It can be observed that there is a equivalent evolution of the $loss$ function and the NMSE obtained as more variables are eliminated. This supports the robustness of the proposed approach. The results also shows the appropriateness of the proposed approach comparing to the results shown in previous works [Rossi et al., 2006]. Also using a forward selection procedure in order to get the subset of variables that obtains the best MI with respect to the output variable gets a worse performance, reaching a subset of 8 variables, obtaining a test NMSE of 0.0051 and training NMSE = 0.0031.

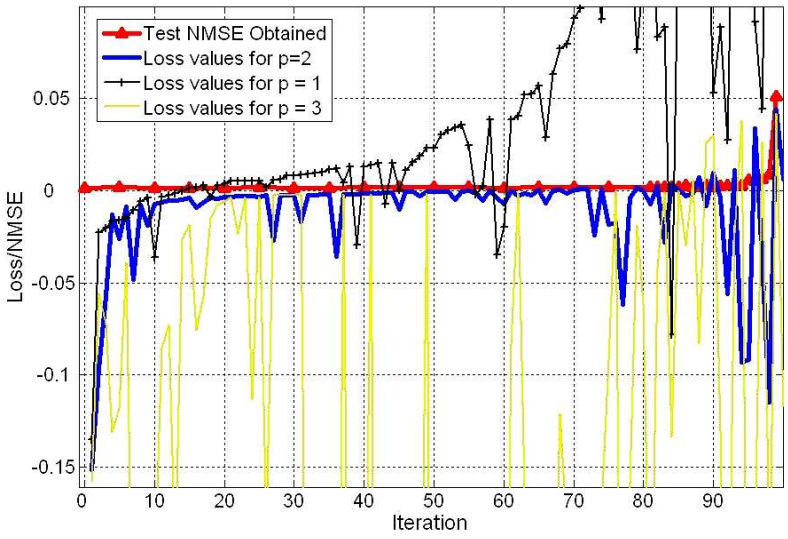


Figura 8.3. *Evolution of the* $Loss'$ *function for* $p = 1 \ldots 3$*, and evolution of the NMSE for the ranking variables for* $p = 2$

Nitrogen data set

The nitrogen data set originates from a software contest organized at the International Diffuse Reflectance Conference 1 held in 1998. It consists of scans and chemistry gathered from Fescue grass (Festuca elatior). The data set contains 141 spectra discretized to 1050 different wavelengths. The goal is to determine the nitrogen content of the grass samples. The data can be obtained from the Analytical Spectroscopy Research Group of the University of Kentucky. As in [François et al., 2007], the data set is split into a test set containing 36 spectra and a training set with the remaining 105 spectra. Instead of applying a functional preprocessing, the brute data was used with two objectives: first due to the convenience of keeping the original meaning of the variables and second to show how the operation of the algorithm can be easily adapted to large problems without losing its robustness.

Figure 8.4 shows the training spectra for the present example.

To obtain the optimal parameter k for the k-nearest neighbor estimator, in order to avoid the excessive computational cost instead of evaluating all the possible values for all the variables, a 10 % of the variables (equally distributed) are considered. The reason for that is that near variables are expected to bring similar information with respect to the output variable. The value for k obtained is $k = 14$.

Next the computationally most expensive part of the algorithm is to obtain the matrix $MI_{inputVariables}(i, j) = I(x_i, x_j)$. The cost of building up this matrix is $(n^2 - n)/2$ in opposite to the selection part of the algorithm in which the cost is $\sum_{p=1}^{P} 2 * (n + n_e lim * p)$. In order to reduce the cost of the algorithm for such a large problem, a simple divide and conquer approach will be performed, by subdividing the 1050 features in groups and executing the algorithm in each group separately in order to decrease the number of available features, and reduce the $O(n^2)$ complexity into $O(n \log n)$. The grouping should be done

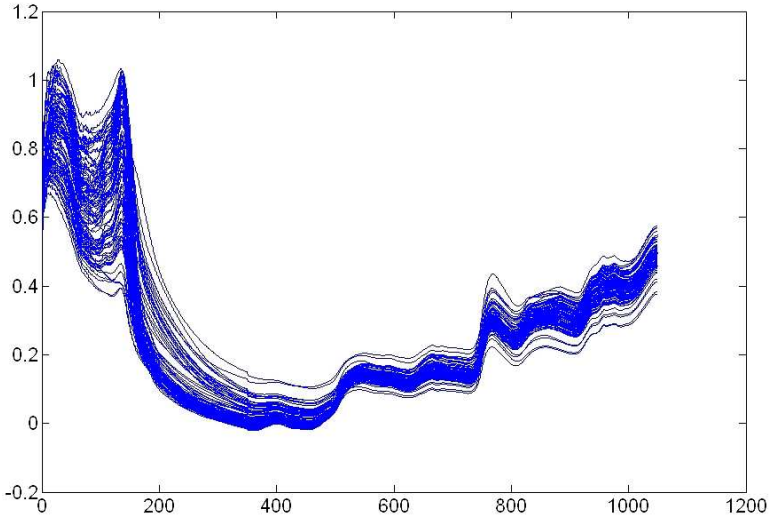


Figura 8.4. *Nitrogen data set training spectra*

using clustering techniques, in order to associate similar features and to eliminate the redundant ones. In this case with spectrometric data, the grouping will be done through nearness in the spectra wavelengths.

A first loop was performed by groups of 50 features, keeping 25 of them, thus remaining 525 features. After another loop was performed taking by nearness groups of 100 features and keeping 50, thus keeping around 250 features. Later pre-final execution took 75 features out of each group of 125 features, and finally, a last execution was performed for the remaining 150 most significant wavelengths.

Figure 8.5 shows the evolution of the Loss function for $p = 1 \ldots 3$. From this figure we can observe that for $p = 1$ the $Loss$ function provides values higher than 0 from the beginning of the variable elimination process. For $p = 2$ it is shown that all values of the $Loss$ function are below 0 until a certain step, from which it starts growing up. For $p = 3$, the variability of the $Loss$ function shows to be very high with plenty of fluctuations that are not observed for lower value of $p = 2$.

Table 8.2 shows the performance for the selected subsets for $p = 1, 2$ and the performance for the whole data set with 1050 variables. As it is shown, for $p = 2$, the algorithm selects an appropriate subset of variables with a similar performance than using the whole set of variables. This shows the good operation of the algorithm for problems with a large number of input variables. However in this specific case, strategies such as the functional preprocessing proposed in [François et al., 2007] are very convenient, in order to obtain an effective low number of input variables identifying the behavior of the subjacent function.

8.5.2. Time series prediction

Two examples of time series are considered now. The well known artificial Mackey-Glass time series [Mackey and Glass, 1977], and a natural series showing the monthly flow of a river [Tim,]. The error measure used in these examples is the Root Mean Square Error,

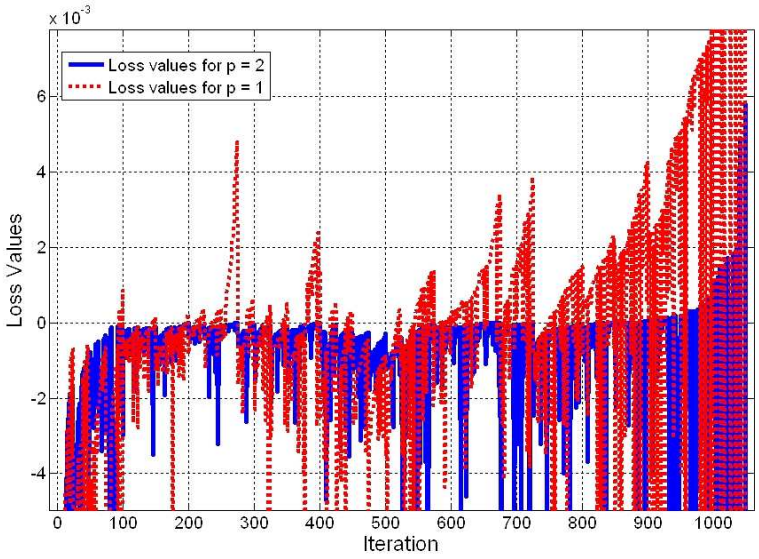


Figura 8.5. *Evolution of the $Loss'$ function in the execution of the algorithm for $p = 1 \ldots 2$ for the Nitrogen spectra data problem*

Tabla 8.2. *Performance obtained for different values of p for the Nitrogen data*

p	Number of Variables	Training NMSE	Test NMSE
Original set X	1050	0.0039	0.0105
1	800	0.0044	0.0110
2	210	0.0045	0.0109

RMSE [Herrera et al., 2007b].

Mackey-Glass time series

This subsection deals with a time series prediction problem. The Mackey-Glass time series is a very well known data series, widely used in the literature of Neural Networks for the evaluation and comparisons of different methodologies. This time series models the dynamics of white blood cell production in the human body, and is described by the following delay differential equation

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx(t) \tag{8.18}$$

A sequence of 1200 data points were generated with an initial condition $x(0) = 1{,}2$ and $\tau = 17$ using the 4th order Runge-Kutta method. From the 1200 data points, the last 1000 samples were taken, 500 for training and 500 for testing. For this example, 50 previous input values $x(t) \ldots x(t-49)$ are selected as candidate regressors in order to predict $x(t+1)$. The training part of the data set is shown in Figure 8.6.

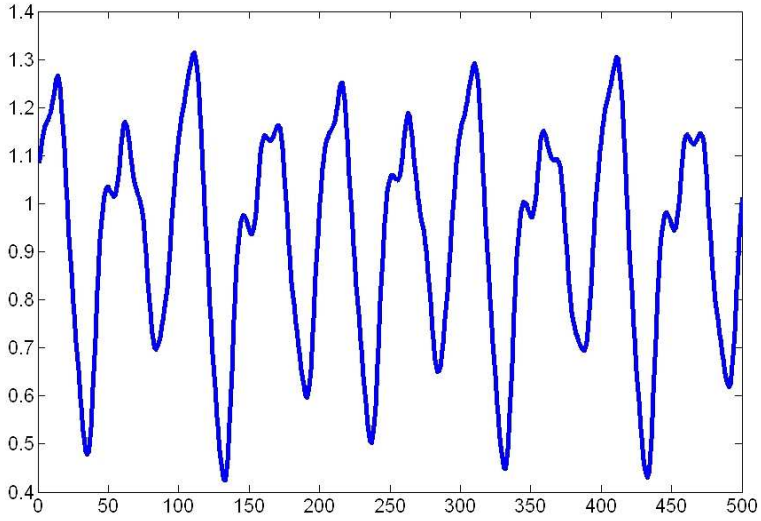


Figura 8.6. *Mackey Glass training series*

Tabla 8.3. *Performance obtained for different values of p for the Mackey-Glass Time series example*

p	Number of Variables	Training RMSE	Test RMSE
Original set X	50	0.0004	0.0005
1	29	0.0005	0.0005
2	11	0.0006	0.0006
3	2	0.0070	0.0070

The optimal value of k for the k nearest neighbors MI estimator in this case equals $k = 20$, that is the one that corresponds to the highest value for the Mutual Information estimation (using the permutation test) for any of the 50 input variables of the problem. The variable selection algorithm starts by building up the matrix $MI_{inputVariables}(i, j)$, and as it is a time series presenting a certain seasonality, it is shown that every variable is interrelated to every other variable in a certain variable amount. This indicates that a "high" value of p could be taken into account, in order to bring more robust results. Considering that 500 data points are available, the algorithm will be executed for p ranking from $1 \ldots 3$. The evolution of the $Loss$ function for the three first values of p is shown in figure 8.7.

According to the execution of the algorithm, for $p = 1$, 29 variables are selected of the 50, for $p = 2$, 11 variables are selected, and for $p = 3$, 2 variables are selected. The performance of a LS-SVM model trained for each subset of variables shows that the best performance is obtained for $p = 2$ (see table 8.3).

Again we show the evolution of the experimental RMSE for the given problem for $p = 2$. As it can be observed in figure 8.8 the Loss function starts providing values higher than 0 approximately at the same moment in which the experimental RMSE obtained by

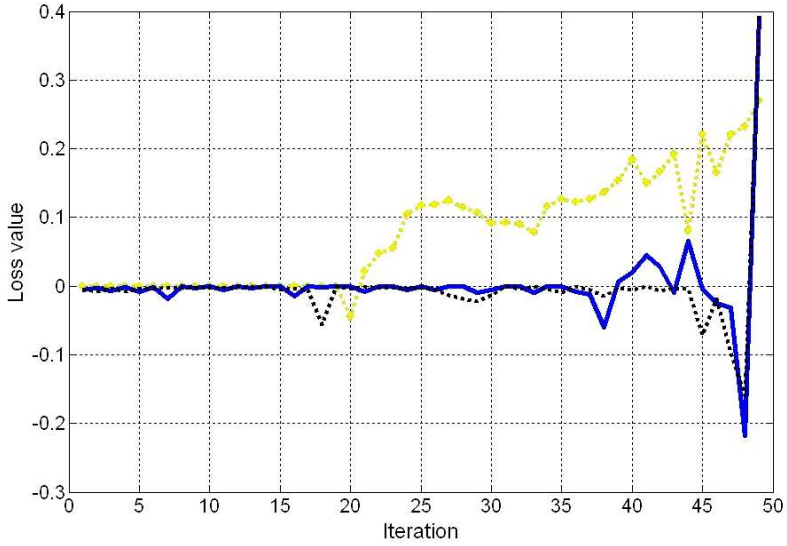


Figura 8.7. *Evolution of the $Loss'$ function in the execution of the algorithm for $p = 1 \ldots 3$ for the Mackey-Glass time series prediction problem*

training LS-SVM models starts growing up. This result supports again the robustness of the proposed approach. A forward ranking of variables performed on this data set in order to obtain the subset with most MI w.r.t. the output variable obtains a subset with just 2 variables with a performance of RMSE = 0.0070.

River Flow time series

Now a natural time series is considered, the monthly river flow series in cubic feet per second of the Snake river taken from [Tim,]. Note how in this case, the series presents a high level of noise, although with a recognizable pattern in the series behavior.

Again for this time series, 1000 samples were taken, 500 for training (see figure 8.9) and 500 for testing. The optimal value of k for the k nearest neighbors MI estimator in this case equals $k = 12$. The complete algorithm is executed for p ranking from $1 \ldots 3$. The evolution of the $Loss$ function for the three first values of p is shown in figure 8.10.

According to the execution of the algorithm, for $p = 1$, 14 variables are selected of the 50, for $p = 2$, 17 variables are selected, and for $p = 3$, 15 variables are selected. In this case, the best performance is obtained for $p = 1$ with a test RMSE of 705.

Again we show the evolution of the experimental RMSE for the given problem for $p = 1$. As it can be observed in figure 8.11 for the ranking of variables selected by the algorithm using $p = 1$, for 14 variables there is a minimum in the test RMSE, that corresponds to the best subset according to the $Loss$ function of the algorithm (see figure 8.10). This result shows the importance of selecting an appropriate subset of variables in order to avoid overfitting and improve the generalization capability of the designed predictors. In this case the search for the subset of variables providing highest MI w.r.t. the output variable leads to a subset with four variables with test RMSE of 802.

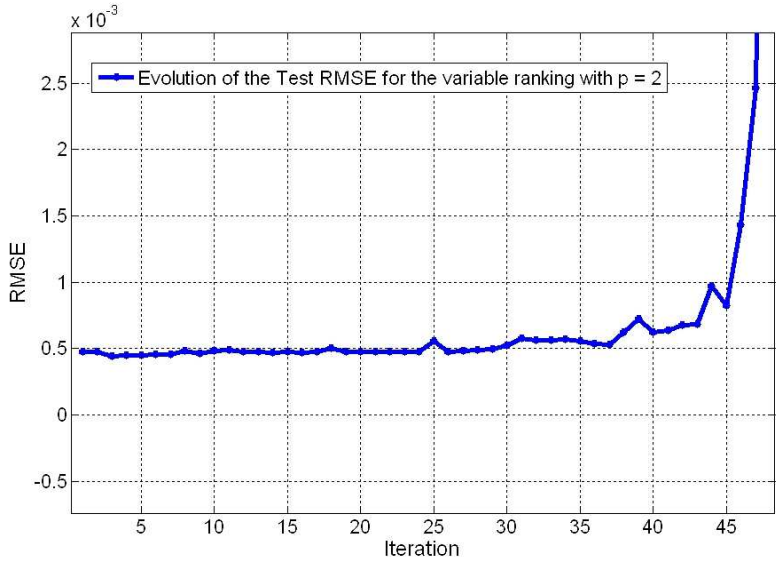


Figura 8.8. *Evolution of the RMSE for the ranking variables for* $p = 2$ *for the Mackey-Glass times series prediction problem*

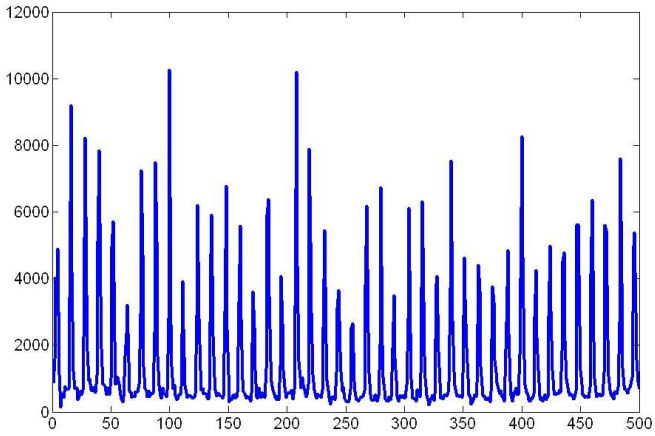


Figura 8.9. *Training data for the monthly river flow series in cubic feet per second of the Snake river, in the period of 1904 until 1994, see [Tim,]*

8.6. Conclusions and Further Work

This chapter has proposed the use of an effective backward variable selection method for function approximation problems, based on the mutual information measure from the Information Theory. It makes use of the concept of Markov blanket of a variable in a set of variables [Koller and Sahami, 1996], [Pearl, 1988] in order to facilitate the backwards

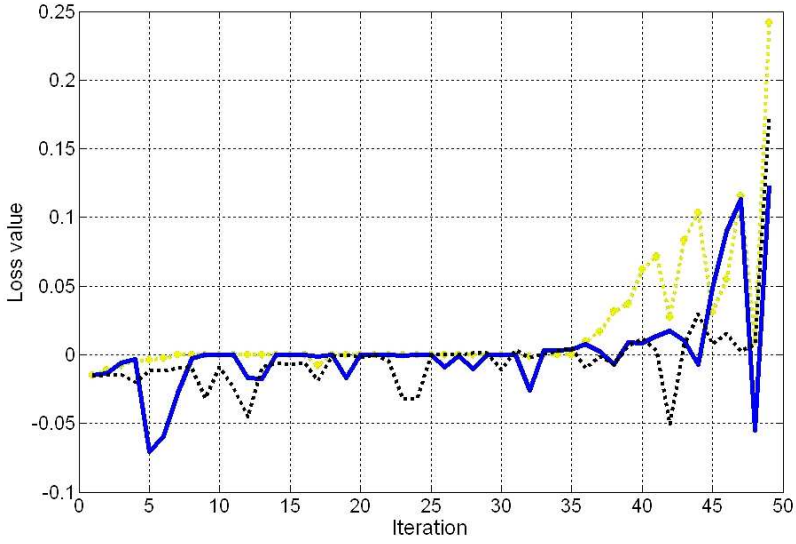


Figura 8.10. *Evolution of the $Loss'$ function in the execution of the algorithm for $p = 1 \ldots 3$ for the River Flow time series prediction problem*

Tabla 8.4. *Performance obtained for different values of p for the River Flow Time series example*

p	Number of Variables	Training RMSE	Test RMSE
Original set X	50	720.63	734.49
1	14	526.90	684.79
2	17	640.24	752.78
3	15	601.55	762.03
4	10	509.65	745.14

process. Its main advantages are that it does not need to perform the estimation of the MI among large groups of variables, thus avoiding the curse of dimensionality problem that MI estimators present.

The stopping criteria for the iterative procedure is performed according to the loss of information that is caused by the elimination of a variable in each step of the algorithm, according to its Markov blanket candidate. For the selection of an appropriate size of Markov blanket, it is proposed to perform a simple wrapper approach for sizes ranging from 1 to 3. The MI estimator used in this work is based on the k-nearest neighbors approach [Kraskov et al., 2004], and both the obtaining of the parameter k, as well as a more robust estimation of the MI, are performed using resampling methods as proposed in previous works [François et al., 2007].

The convenience of the proposed approach has been shown in four significant examples: two spectrometric data problems and two time series prediction problems. It has been

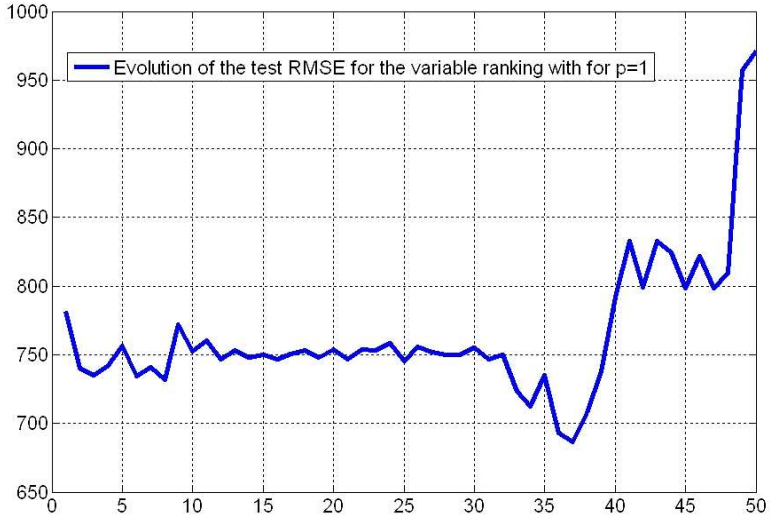


Figura 8.11. *Evolution of the RMSE for the ranking variables for* $p = 1$ *for the River Flow times series prediction problem*

shown in the simulations that for an appropriate value of Markov blanket size, an appropriate small subset of variables is obtained that keeps or improves the performance comparing to the whole set of input variables. As further work, it is intended to test the Markov blanket loss function as a stopping criterion on forward selection, as well as alternatives for mixture backwards-forwards methods based on the MI.

Capítulo 9

Conclusiones y Principales Aportaciones

El trabajo de tesis doctoral presentado en esta memoria es una serie de contribuciones a diversos problemas específicos dentro de la amplísima problemática del modelado de funciones utilizando varias técnicas de modelado continuo. La memoria presenta varios trabajos, cada uno de los cuales pretende resolver una problemática concreta. Tres de estos trabajos tratan el problema de la maldición de la dimensionalidad en los sistemas neuro-difusos para problemas de aproximación funcional, así como el problema de la pérdida de la interpretabilidad y del modelado local en estos sistemas, que ocurre durante el entrenamiento a partir de datos de entrada y salida. Un cuatro trabajo trata el problema de predicción de series temporales utilizando modelos recursivos, utilizando métodos de kernel y sistemas neuro-difusos, aportando una técnica general para mejorar este tipo de predicción. El quinto trabajo trata el problema de la selección de variables en problemas de aproximación funcional, aportando una metodología robusta basada en la medida de la información mutua de la Teoría de la Información.

Las principales aportaciones de cada uno de los trabajos y las conclusiones obtenidas en cada caso se resumen a continuación, junto con un repaso a los capítulos introductorios a la problemática tratada.

- El capítulo 2 ha realizado una revisión de los conceptos más importantes necesarios para una comprensión adecuada del resto de capítulos de la memoria. Se han revisado los modelos para aproximación funcional Takagi-Sugeno-Kang (TSK), redes de Funciones de Base Radial y Máquinas de Vectores Soporte de Mínimos Cuadrados. También se ha revisado la equivalencia entre estos modelos, sin perder sus características y diversidad de funcionamiento, con sus ventajas e inconvenientes, a la hora de tratar un problema de modelado concreto.

- El capítulo 3 ha sentado las motivaciones de los modelos difusos TSK presentados en los siguientes tres capítulos. El aprendizaje automático de sistemas neuro-difusos para problemas de aproximación funcional puede presentar una serie de escollos importantes que surgen a la hora de tratar este tipo de problemas. El capítulo ha introducido brevemente el aprendizaje de los sistemas neuro-difusos TSK y redes RBF, junto con algunas de las opciones existentes en su diseño y los problemas que se presentan en dicho aprendizaje. Vistas estas opciones con sus ventajas e inconvenientes, y los problemas que se presentan a la hora de resolver un problema de modelado continuo utilizando sistemas neuro-difusos, se han introducido las tres soluciones

alternativas concretas que proponen los tres capítulos siguientes.

- El capítulo 4 ha presentado una nueva propuesta de modelo TSK, que denominamos modelo TaSe. Este modelo resuelve el problema de la pérdida de la interpretabilidad y de modelado local que presentan los modelos TSK basados en grid en su entrenamiento. Utilizando la metodología de aprendizaje propuesta, se obtiene un sistema eficaz e interpretable, que hace uso de la expansión en serie de Taylor de una función en torno a un punto, para aproximar la función objetivo usando un número reducido de reglas. Además este trabajo se ha complementado con un análisis estadístico sobre el efecto en el rendimiento de los parámetros más importantes del diseño de un sistema difuso tipo TSK. Dando un relevancia especial al orden de las reglas del modelo (orden polinómico de los consecuentes: cero, uno y dos), los resultados del análisis ANOVA realizado muestran que el orden de las reglas usadas es estadísticamente significativo en el rendimiento del modelo. Así pues, los resultados obtenidos muestran la conveniencia de utilizar el modelo TaSe para una amplia gama de problemas de modelado gracias a sus características de gran capacidad aproximativa y conservación de la interpretabilidad del modelo.

- El capítulo 5 ha presentado un trabajo complementario al modelo TaSe presentado en el capítulo anterior, proporcionando un modelo TaSe-NF con características similares al modelo TaSe, pero que se puede aplicar como modelo basado en clustering, o equivalentemente, a redes RBF. En los modelos tradicionales (basados en clustering, o redes RBF), la optimización de los modelos locales y la extracción de un conjunto de reglas con buena interpretabilidad es en general una tarea nada trivial como viene demostrándose en la literatura. El modelo neuro-difuso basado en clustering propuesto, mantiene las propiedades de optimización de los modelos locales a la vez que entrena la red optimizándola globalmente. Además, gracias a la implementación de un enfoque de compartición de funciones de pertenencia, se mejora la interpretabilidad del conjunto de reglas difusas extraídas del modelo. Además, este trabajo se complementa con un enfoque novedoso para la inicialización de centros, basado en el método de equidistribución de errores de sistemas basados en grid, que obtiene resultados superiores a otros métodos de clustering presentes en la literatura para esta tarea.

- El capítulo 6 ha presentado una alternativa para el tratamiento de problemas de dimensionalidad media utilizando sistemas difusos basados en grid. Este tipo de sistemas en general no se pueden aplicar a problemas complejos debido a la maldición de la dimensionalidad en complejidad que padecen. La arquitectura de los sistemas Multigrid propuesta permite reducir exponencialmente el número de reglas, mediante el descarte de relaciones irrelevantes entre las variables. En este trabajo se ha propuesto un algoritmo de aprendizaje completo para sistemas difusos Multigrid a partir de un conjunto de datos de entrada/salida, que incorpora un proceso de selección de variables. En este capítulo además se ha mostrado la conveniencia de la metodología Multigrid propuesta, con la aplicación al problema complejo de predicción de series temporales CATS.

- El capítulo 7 ha mostrado la aptitud de dos metodologías diferentes, el modelo TaSe y las Máquinas de Vectores Soporte de Mínimos Cuadrados, para resolver el problema de predicción de series temporales a largo plazo utilizando predicción recursiva. Se ha introducido una técnica heurística que incrementa el rendimiento de este tipo de modelos avanzados de predicción a un paso (y en general de cualquier modelo de

predicción a un paso), cuando se utilizan recursivamente para predicción de series temporales a largo plazo. La técnica propuesta ha quedado avalada por los resultados obtenidos utilizando como ejemplos tanto series artificiales como naturales.

- El capítulo 8 ha aportado una metodología novedosa de selección de variables para problemas de aproximación funcional. Se trata de una metodología con estrategia hacia atrás, que utiliza la información mutua como criterio, y que evita los problemas de los estimadores de la IM cuando se trabaja con conjuntos grandes de variables, gracias al concepto de marco de Markov. La conveniencia de la metodología propuesta se ha demostrado en cuatro ejemplos significativos, dos de datos espectrales y dos de predicción de series temporales, ambos tipos de problemas muy importantes y de mucha aplicación en el área de la aproximación funcional.

Esta memoria como se dijo en la introducción de la misma, no pretendía establecer criterios o comparar diferentes metodologías para problemas de aproximación funcional. Sin embargo, en una memoria tan heterogénea, no podemos evitar resumir algunas conclusiones que se sonsacan de estos trabajos aportados.

En primer lugar destaca la necesidad de disponer de buenos métodos de selección de variables para aproximación funcional. Es esencial para evitar la maldición de la dimensionalidad, especialmente en el caso de modelos con complejidad dependiente del número de dimensiones, y para optimizar la capacidad de generalización del modelo. El método propuesto en esta memoria, es un método efectivo que muestra seleccionar un buen conjunto de variables con un buen rendimiento para los problemas concretos propuestos. Sin embargo puede presentar ciertos inconvenientes en el caso de inter-dependencia heterogénea entre las variables de entrada, y empezaremos a investigar sobre nuevas vías de selección y de criterios de parada en algoritmos iterativos.

Respecto de la maldición de la dimensionalidad, en algunos ejemplos en los trabajos aquí propuestos, y una muy numerosa bibliografía demuestra la necesidad de disponer de metodologías que mitiguen este problema. Desde este punto de vista, las máquinas de vectores soporte (SVM) y, para problemas de aproximación funcional, más concretamente las máquinas de vectores soporte de mínimos cuadrados demuestran ser una metodología muy conveniente desde el punto de vista de la precisión obtenida en un problema de modelado. Los métodos de kernel en general presentan una muy buena aptitud para minimizar el problema de tratar con problemas de dimensionalidad muy alta. Sin embargo estas metodologías presentan dos inconvenientes principales: por un lado la maldición de la dimensionalidad en la complejidad del entrenamiento en el número de datos de entrenamiento, y por otro lado la carencia de interpretabilidad.

La optimización de estos modelos de kernel requiere hallar el valor óptimo de tan solo dos hiper-parámetros. Sin embargo es un problema de optimización no lineal que requiere una búsqueda exhaustiva, y por tanto un gran esfuerzo computacional. Además, estos modelos carecen completamente de interpretabilidad y entendimiento por parte del experto ni del diseñador. Se asemejan por tanto altamente a las redes neuronales tradicionales en las que el funcionamiento interno era desconocido. Esto por tanto limita mucho la utilidad de estos modelos, para problemas en los que se requiera una justificación de resultados o una comprobación o comprensión de los mismos.

Los sistemas neuro-difusos y los sistemas difusos proporcionan en ese sentido soluciones interpretables, y útiles en muchos casos para el diseñador o experto. En concreto los sistemas TSK proporcionan buenos resultados de aproximación gracias a la utilización de reglas con consecuentes polinomiales. Sin embargo en general el entrenamiento de estos sistemas puede provocar una pérdida de la interpretabilidad de las reglas en varios factores.

Además, en el caso de utilización de particionamiento del espacio de entrada basado en grid, la complejidad del modelo obtenido puede hacer inviable su utilización.

En esta memoria se han presentado tres alternativas que resuelven al menos parcialmente algunos de los problemas que presentan los modelos neuro-difusos en su aplicación a este tipo de problemas. Cabe resaltar la obtención de modelos con una alta interpretabilidad gracias a la utilización de la serie de Taylor, al utilizar las metodologías TaSe y TaSe-NF aportados. Así, podemos afirmar, que en el caso en el que la interpretabilidad sea un requerimiento, o en casos con una baja o media complejidad dimensional del problema, las alternativas propuestas de sistemas neuro-difusos, son preferibles gracias a las ventajas de interpretabilidad y buen rendimiento que proporcionan.

En cuanto al trabajo futuro, se esperan refinar metodologías de selección de variables utilizando la información mutua con técnicas adicionales para estimar de manera óptima la eficacia esperada de los conjuntos seleccionados. Además, la problemática de la selección de variables junto con métodos efectivos de aproximación funcional es una temática esencial donde numerosos problemas reales pueden ser abordados adecuadamente, y donde aún no se habían obtenido resultados satisfactorios.

Por último, la principal via de investigación que se seguirá a partir de los trabajos realizados engloba la aplicación y adaptación de estos métodos y algoritmos diseñados a plataformas de altas prestaciones, como medio efectivo de abordar problemas reales complejos que requieren soluciones efectivas y eficientes, a veces en tiempo real.

Conclusions and Main Contributions

The work presented in this thesis is composed by a set of contributions to different specific problems inside the large area of function modeling using continuous modeling techniques. Three of these works deal with the curse of dimensionality problem in neuro-fuzzy systems for function approximation problems, as well as the problem of the loss of interpretability and of the local modeling in those systems, that occurs during the training of the models from input/output data. A fourth work deals with the problem of time series prediction using recursive models; it utilizes kernel methods and neuro-fuzzy systems, providing a general technique to improve this type of prediction. The fifth work deals with the problem of variable selection in function approximation problems, providing a robust methodology based on the mutual information measure of the Information Theory.

The main contributions of each of the works and the conclusions obtained in each case are summarized now, together with a review of the introductory chapters

- Chapter 2 has performed a review of the most important concepts needed for an appropriate comprehension of the rest of the chapters of the thesis. The Takagi-Sugeno-Kang models, the Radial Basis Function networks and the Least-Squares Support Vector Machines were introduced in this chapter. Also the equivalence among those models has been revised, remarking too their characteristics and diversity in their operation, with their advantages and drawbacks, when dealing with a specific modeling problem.

- Chapter 3 has stated the motivations for the neuro-fuzzy TSK models presented in the next three chapters. The automatic learning process in neuro-fuzzy systems for function approximation problems can present a set of important handicaps, that occur when dealing with this type of problems. The chapter has briefly introduced the learning process in TSK neuro-fuzzy systems and RBF networks, together with some of the existing alternatives in their design, and the problems that they can present during their learning process. Once these options are reviewed, with their advantages and disadvantages, and the problems that they present when dealing with a continuous modeling problem, the three specific solutions presented in the following three chapters were introduced.

- Chapter 4 has presented a novel proposal of TSK model, that we call TaSe model. This model solves the problem of the loss of interpretability and of local modeling that traditional TSK models present in their learning process. Using the proposed learning methodology, an efficient and interpretable system is obtained, that makes use of the Taylor series expansion around a point, in order to approximate the ob-

jective function using a reduced set of rules. Furthermore, this work has been complemented with a statistical analysis on the effect of the performance of the most important parameters in the design of a TSK fuzzy system. A special relevance was given to the order of the rules of the model (polynomial order of the consequents), and the results provided by the ANOVA analysis have shown that the order of the rules is statistically significant in the performance of the model. Therefore, the results obtained show the convenience of using the TaSe model for a wide range of modeling problems, thanks to their characteristics of high approximative capability and maintenance of the interpretability of the model.

- Chapter 5 has presented a complementary work to the TaSe model presented in the previous chapter, providing a novel model (TaSe-NF model) with similar characteristics to the TaSe model, but that can be applied as a clustering-based fuzzy model, or equivalently as a RBF network. In the traditional models based on clustering (or similarly RBF networks), the optimization of the local models and the extraction of a set of interpretable rules is in general a difficult task as has been seen in the literature. The clustering-based neuro-fuzzy model proposed in this work, keeps the properties of optimization of the local models while training the network and optimizing it globally. Furthermore, thanks to the implementation of a MF sharing approach, the interpretability of the extracted set of rules is improved. Moreover, this work is complemented with a novel approach for centers initialization for clustering-based fuzzy systems, that obtains better performance than other methods proposed in the literature with this objective.

- Chapter 6 has presented an alternative for the treatment of medium dimensionality problems using grid-based fuzzy systems. This type of systems in general can not be applied to complex problems due to the curse of dimensionality in complexity that they suffer. The proposed Multigrid architecture allows to reduce exponentially the number of rules, through ruling out irrelevant variables interrelations. This work has proposed a complete learning algorithm for Multigrid fuzzy systems from a set of input/output data, that incorporates a variable selection process. The convenience of the proposed Multigrid methodology, has been shown through the application to the complex CATS time series prediction problem.

- Chapter 7 has showed the appropriateness of two different methodologies, the TaSe model and Least Squares Support Vector Machines, to solve long term time series prediction problems using recursive prediction. A novel heuristic technique has been introduced that increments the performance of this type of one-step-ahead advanced models (and in general of any one-step-ahead model), when they are recursively used for long term time series prediction. The proposed technique has been corroborated by the results shown using both artificial and natural time series.

- Chapter 8 has provided a novel methodology for variable selection for function approximation problems. This methodology implements a backwards strategy that uses mutual information as criterion, and avoids the problems with the MI estimators when dealing with large groups of variables, thanks to the concept of Markov blanket. The convenience of the proposed methodology has been shown in four significant examples, two spectrometric data problems and two time series prediction problems, both types of problems very important in the function approximation area.

This thesis, as it was mentioned in the introduction, does not intend to establish criteria or compare different methodologies for function approximation problems. However,

in such an heterogeneous work, it is not possible to avoid summarizing some conclusions that can be drawn from the different contributions provided.

First, it is important to note the need to have at disposal good variable selection methods for function approximation problems. It is essential to avoid the curse of dimensionality, specially in the case of models presenting dimensionality-dependent complexity, and in general in order to increase the generalization capability of the model. The proposed methodology for variable selection, is an effective method that shows to select an appropriate subset of variables with a good performance for the specific proposed examples. However, it can present a number of drawbacks in the case of heterogeneous inter-dependency among the input variables; therefore, we're starting new research on ways of selection and stopping criterion in iterative variable selection mechanisms.

In relation to the curse of dimensionality, some examples performed in the works proposed here, and a large literature show the need of having at disposal methodologies that alleviate this problem. From this point of view, SVMs and, more specifically for function approximation problems, LS-SVMs show to be a very convenient methodology from the point of view of the performance. Kernel methods in general present a very good aptitude in minimizing the problem of dealing with very high dimensional data. However, this methodology presents two drawbacks: on the one hand the curse of dimensionality in complexity in the number of training data points, and on the other hand, the lack of interpretability.

The optimization of these kernel methods require finding the optimal value for just 2 hyper-parameters. However, it is a nonlinear optimization problem that requires an exhaustive search, and therefore a large computational effort. Moreover, these models completely lack of interpretability and understanding from the expert or designer. They highly resemble therefore traditional neural networks in which the internal operation is unknown. This therefore highly limits the utility of these models for problems that require the justification of the results or their verification or comprehension.

Neuro-fuzzy systems and fuzzy systems provide in this sense interpretable and useful solutions for the designer or expert. Specifically, TSK systems provide good approximation results thanks to the use of rules with polynomial consequents. However, in general the training of those systems can lead to a loss of interpretability of the rules in a some factors. Moreover, in the case of using grid-based input space partitioning, the final complexity of the model can make its usage unfeasible.

This thesis has presented three different alternatives that at least partially solve some of the problems that neuro-fuzzy models present when dealing with this type of problems. It must be highlighted the obtaining of models with large interpretability thanks to the use of the Taylor series, when using the TaSe or TaSe-NF provided methodologies. Thus, we can claim that in the case that the interpretability is a requirement, or in cases with a low or medium dimensionality complexity of the problem, the proposed alternatives of neuro-fuzzy systems are preferable thanks to the interpretability advantages and good performance that they present.

In relation to the further work, it is expected to refine variable selection methodologies using mutual information, using additional techniques in order to estimate optimally the expected efficiency of the selected subsets. Moreover, the variable selection problem, together with effective function approximation methods, is an essential subject with which a number of real problems could be properly tackled, and in which there still hasn't been found satisfactory results.

Finally, the main research line that will be followed from the works performed, includes the application and adaptation of those methods and designed algorithms to high-performance computing systems, as an effective way of tackling real complex problems that require effective and efficient solutions, sometimes in real time.

Apéndices

Apéndice A

Funciones de Pertenencia

En este apéndice se presentan las configuraciones de funciones de pertenencia utilizadas a lo largo de esta memoria. Común para todas estas configuraciones es la definición de la función escalón $U(x; a, b)$ dada por:

$$U(x; a, b) = \begin{cases} 1 \text{sia} \leq x < b \\ 0 \text{enotrocaso} \end{cases} \tag{A.1}$$

A.1. Funciones Triangulares

En este apartado se presentan las expresiones matemáticas de las funciones de pertenencia triangulares. Se definirá el concepto de partición triangular, en donde los extremos de las funciones triangulares coinciden con los centros de las funciones vecinas, que es el tipo de particionamiento utilizado en este trabajo para funciones de pertenencia triangulares.

Las características de una partición triangular son:

- Presentan un grado de solapamiento dos. Generalmente, cuando se utilizan controladores difusos se suele exigir que un dato no active más de dos funciones de pertenencia simultáneamente ya que el cerebro humano no suele actuar de esa forma y el significado de las reglas obtenidas sería complejo. Además, esto permite que las reglas puedan ser fácilmente comprendidas y completadas en aquellas regiones en las que no existen vectores de E/S para determinarlas.

- Para todo valor del rango que cubren se cumple que $\sum_{s=1}^{n} \mu_{X^s}(x) = 1$. De esta forma, si un dato activa una función de pertenencia con valor $\mu_{X^{s'}}(x)$ debe haber otra función de pertenencia que active con un valor $1 - \mu_{X^{s'}}(x)$. Esta condición, junto con la anterior hace que la distribución quede completamente definida por la posición de los centros de cada una de las funciones triangulares. Los otros dos parámetros (el lateral izquierdo y el derecho) vienen definidos implícitamente por los centros de las dos funciones vecinas. En la siguiente figura se representa la distribución dada por los centros situados en los puntos 0.0, 0.20, 0.70, 0.90, 1.0 para la variable v:

Además, es muy fácil obtener la función $\mu_{X^{s'}}(x)$ para un x dado. Realizando operaciones sencillas es fácil demostrar que:

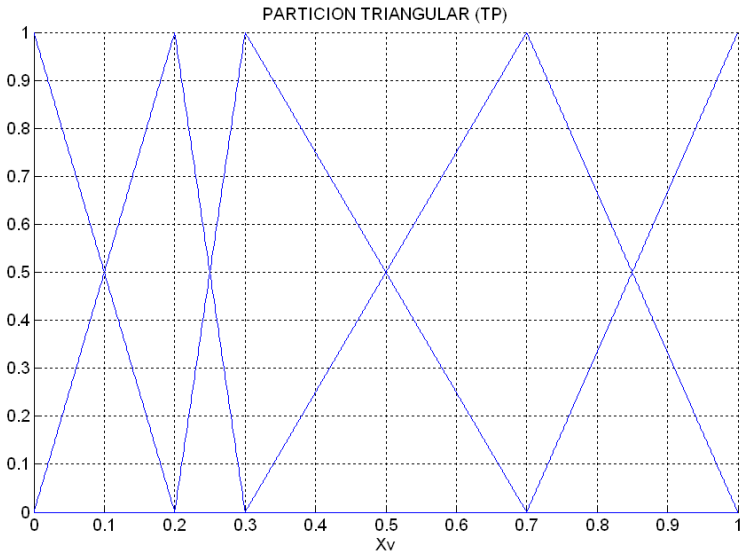


Figura A.1. *Partición triangular (TP)*

$$\mu_{X_v^i}(x) = \begin{cases} U(x;-\infty,c_v^i) + \frac{c_v^{i+1}-x}{c_v^{i+1}-c_v^i} U(x;c_v^i,c_v^{i+1}) \text{ si i} = 1 \\ \frac{x-c_v^{i-1}}{c_v^i-c_v^{i-1}} U(x;c_v^{i-1},c_v^i) + \frac{c_v^{i+1}-x}{c_v^{i+1}-c_v^i} U(x;c_v^i,c_v^{i+1}) \text{ si i} \neq 1 \text{ei} \neq \text{n} \\ \frac{x-c_v^{i-1}}{c_v^i-c_v^{i-1}} U(x;c_v^{i-1},c_v^i) + U(x;c_v^i,\infty) \text{ si i} = \text{n} \end{cases} \quad \text{(A.2)}$$

donde se ha considerado que las funciones de los extremos presentan un grado de pertenencia 1 para los puntos que están más allá del rango de definición de la variable.

Las derivadas de las funciones de pertenencia así definidas con respecto a sus parámetros (que son sólo los centros) vendrán dadas por:

$$\frac{\partial \mu_{X_v^i}(x)}{\partial c_v^j} = \begin{cases} \frac{-\left(c_v^i - x\right)}{\left(c_v^i - c_v^{i-1}\right)^2} U(x;c_v^{i-1},c_v^i) \text{ si j} = \text{i} - 1 \\ \frac{-\left(x - c_v^{i-1}\right)}{\left(c_v^i - c_v^{i-1}\right)^2} U(x;c_v^{i-1},c_v^i) + \frac{c_v^{i+1}-x}{\left(c_v^{i+1}-c_v^i\right)^2} U(x;c_v^i,c_v^{i+1}) \text{ si j} = \text{i} \\ \frac{x-c_v^i}{\left(c_v^{i+1}-c_v^i\right)^2} U(x;c_v^i,c_v^{i+1}) \text{ si j} = \text{i} + 1 \end{cases} \quad \text{(A.3)}$$

siendo la derivada nula en cualquier otro caso.

A.2. Funciones Gaussianas

En este apartado se presentan las expresiones matemáticas de las funciones de pertenencia gaussianas usadas en este trabajo. En el apartado A.2.1 se definirá la configuración estándar. En la sección siguiente (A.2.2), una configuración equivalente a la partición triangular, donde las funciones gaussianas son limitadas fijando la intersección entre ellas a un

cierto valor L. Finalmente, se presentan las gaussianas generalizadas o pseudo-gaussianas asimétricas, donde las funciones de pertenencia están representadas por gaussianas con distinta desviación (σ) a la izquierda y a la derecha de modo que la función de pertenencia deja de ser simétrica.

A.2.1. Funciones de pertenencia gaussianas (G)

Es, junto con la partición triangular, una de las configuraciones más ampliamente utilizadas en la bibliografía. En este caso, cada función de pertenencia tiene la forma de una gaussiana fijada por su centro y su desviación, valiendo la unidad en el punto central y tendiendo a cero conforme nos alejamos de él, sin llegar nunca a dicho valor. En la siguiente figura vemos un ejemplo de dicha configuración.

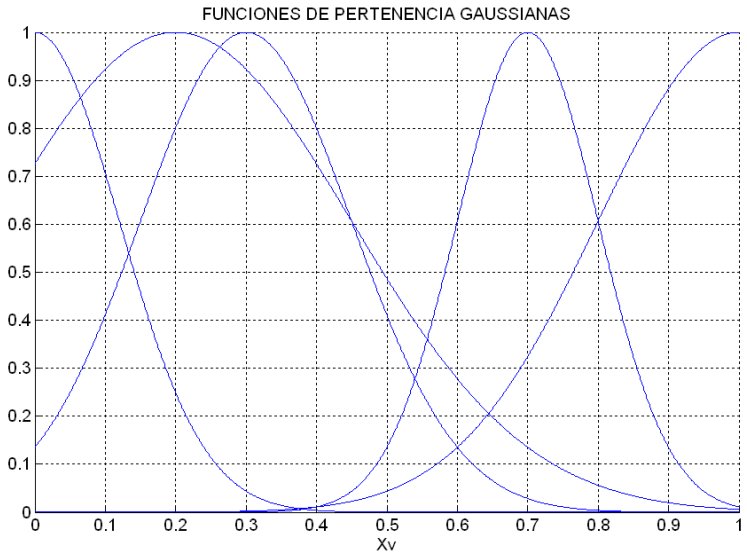


Figura A.2. *Funciones de pertenencia gaussianas.*

El grado de pertenencia de la entrada x a la función de pertenencia X_v^i vendrá dado por:

$$\mu_{X_v^i}(x) = e^{-\frac{\left(x-c_v^i\right)^2}{\sigma_v^i}} \tag{A.4}$$

Las derivadas de las funciones de pertenencia así definidas con respecto a sus parámetros serían, en este caso:

$$\frac{\partial \mu_{X_{\mathrm{v}}^i}(x)}{\partial c_{\mathrm{v}}^i} = 2\frac{x-c_v^i}{\sigma_v^i}\mu_{X_{\mathrm{v}}^i}(x) \tag{A.5}$$

$$\frac{\partial \mu_{X_{\mathrm{v}}^i}(x)}{\partial \sigma_{\mathrm{v}}^i} = 2\left(\frac{x-c_v^i}{\sigma_v^i}\right)^2 \mu_{X_{\mathrm{v}}^i}(x) \tag{A.6}$$

A.2.2. Partición pseudo-gaussiana (PPG)

En este apartado presentaremos un tipo novedoso de configuración de funciones de pertenencia que también depende exclusivamente de la localización de los centros igual que pasaba con la partición triangular del apartado A.2.1. Para poder obtener configuraciones parecidas a las obtenidas con las funciones triangulares vamos a usar pseudo-gaussianas asimétricas, es decir, vamos a usar distintos valores de las desviaciones típicas según la rama que se considere. Denotando por un signo '+' el valor de la desviación en la zona derecha y por un signo '-' la de la zona izquierda, tendremos que para la función de pertenencia X_v^i su centro vendrá dado por c_v^i y sus desviaciones izquierda y derecha por $\sigma_v^{i,-}$ y $\sigma_v^{i,+}$ respectivamente. Fijando el valor L que queremos que tenga la función justo en los centros contiguos tenemos que:

$$\mu_{X_v^i}(x = c_v^{i+1}) = e^{-\frac{\left(c_v^{i+1} - c_v^i\right)^2}{\sigma_v^{i,+}}} = L \tag{A.7}$$

$$\mu_{X_v^i}(x = c_v^{i-1}) = e^{-\frac{\left(c_v^{i-1} - c_v^i\right)^2}{\sigma_v^{i,-}}} = L \tag{A.8}$$

Despejando las desviaciones izquierda y derecha en cada una de las expresiones anteriores obtenemos que:

$$\sigma_v^{i,+} = \frac{\left(c_v^{i+1} - c_v^i\right)^2}{\ln\left(1/L\right)} \equiv k\left(c_v^{i+1} - c_v^i\right)^2 \tag{A.9}$$

$$\sigma_v^{i,-} = \frac{\left(c_v^{i} - c_v^{i-1}\right)^2}{\ln\left(1/L\right)} \equiv k\left(c_v^{i} - c_v^{i-1}\right)^2 \tag{A.10}$$

de forma que las desviaciones vendrán unívocamente determinadas por la localización de los centros, tal y como sucedía en el caso de la partición triangular. De las ecuaciones A.9 y A.10 además tenemos $\sigma_v^{i,+} = \sigma_v^{i+1,-}$.

Si se quiere poner como referencia no el valor del grado de pertenencia en el centro vecino sino en el punto medio (que puede parecer lo más apropiado) lo que tendremos será:

$$\mu_{X_v^i}(x = c_v^i + \frac{c_v^{i+1} - c_v^i}{2}) = e^{-\frac{\left(c_v^{i+1} - c_v^i\right)^2}{4\sigma_v^{i,+}}} = M \tag{A.11}$$

resultando finalmente que $L = M^4$.

En la siguiente figura se representa la distribución definida por los centros situados en los puntos $0{,}0, 0{,}20, 0{,}70, 0{,}90, 1{,}0$, con $L = 0{,}01$:

El grado de pertenencia de la entrada x a la función de pertenencia X_v^i vendrá, en una partición pseudo gaussiana, dada por:

$$\mu_{X_v^i}(x) = \begin{cases} U(x; -\infty, c_v^i) + e^{-\frac{\left(x - c_v^i\right)^2}{\sigma_v^{i,+}}} U(x; c_v^i, \infty) \text{ si i} = 1 \\ e^{-\frac{\left(x - c_v^i\right)^2}{\sigma_v^{i,-}}} U(x; -\infty, c_v^i) + e^{-\frac{\left(x - c_v^i\right)^2}{\sigma_v^{i,+}}} U(x; c_v^i, \infty) \text{ si i} \neq 1 \text{ei} \neq \text{n} \\ e^{-\frac{\left(x - c_v^i\right)^2}{\sigma_v^{i,-}}} U(x; -\infty, c_v^i) + U(x; c_v^i, \infty) \text{ si i} = \text{n} \end{cases} \tag{A.12}$$

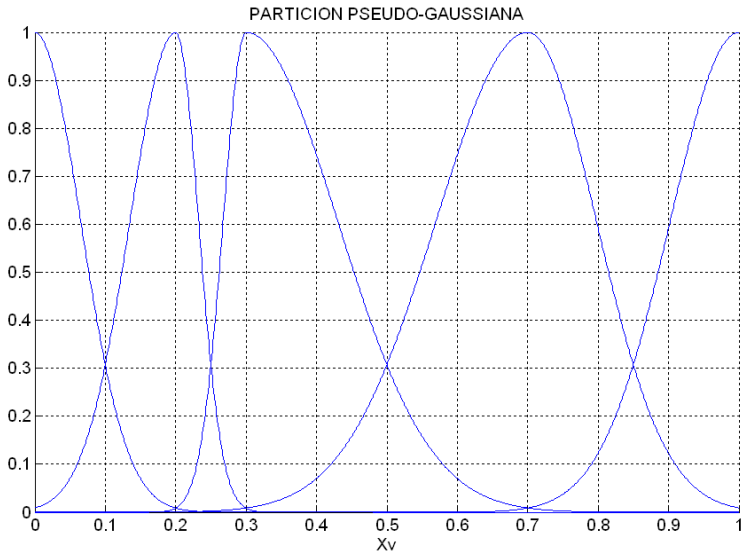


Figura A.3. *Partición pseudo-gaussiana.*

donde, al igual que en el caso de la partición triangular, las funciones de los extremos presentan un grado de pertenencia 1 para los puntos que están más allá del rango de definición de la variable.

Finalmente, a partir de la expresión anterior y de la dependencia de las desviaciones con los centros, tendremos que:

$$\frac{\partial \mu_{X_{\mathrm{v}}^{\mathrm{i}}}(x)}{\partial c_{\mathrm{v}}^{j}} = \begin{cases} -2k\left(c_{\mathrm{v}}^{\mathrm{i}} - c_{\mathrm{v}}^{\mathrm{i}-1}\right)\left(\frac{x-c_{\mathrm{v}}^{\mathrm{i}}}{\sigma_{\mathrm{v}}^{\mathrm{i},-}}\right)^2 e^{-\frac{\left(x-c_{\mathrm{v}}^{\mathrm{i}}\right)^2}{\sigma_{\mathrm{v}}^{\mathrm{i},-}}} U(x;-\infty,c_{\mathrm{v}}^{\mathrm{i}}) \text{ si j} = \text{i} - 1 \\ 2\frac{x-c_{\mathrm{v}}^{\mathrm{i}}}{\sigma_{\mathrm{v}}^{\mathrm{i},-}} e^{-\frac{\left(x-c_{\mathrm{v}}^{\mathrm{i}}\right)^2}{\sigma_{\mathrm{v}}^{\mathrm{i},-}}}\left(1 + k\frac{x-c_{\mathrm{v}}^{\mathrm{i}}}{\sigma_{\mathrm{v}}^{\mathrm{i},-}}\left(c_{\mathrm{v}}^{\mathrm{i}} - c_{\mathrm{v}}^{\mathrm{i}-1}\right)\right) U(x;-\infty,c_{\mathrm{v}}^{\mathrm{i}}) + \\ 2\frac{x-c_{\mathrm{v}}^{\mathrm{i}}}{\sigma_{\mathrm{v}}^{\mathrm{i},+}} e^{-\frac{\left(x-c_{\mathrm{v}}^{\mathrm{i}}\right)^2}{\sigma_{\mathrm{v}}^{\mathrm{i},+}}}\left(1 - k\frac{x-c_{\mathrm{v}}^{\mathrm{i}}}{\sigma_{\mathrm{v}}^{\mathrm{i},+}}\left(c_{\mathrm{v}}^{\mathrm{i}+1} - c_{\mathrm{v}}^{\mathrm{i}}\right)\right) U(x;c_{\mathrm{v}}^{\mathrm{i}},\infty) \text{ si i} = \text{j} \\ 2k\left(c_{\mathrm{v}}^{\mathrm{i}+1} - c_{\mathrm{v}}^{\mathrm{i}}\right)\left(\frac{x-c_{\mathrm{v}}^{\mathrm{i}}}{\sigma_{\mathrm{v}}^{\mathrm{i},+}}\right)^2 e^{-\frac{\left(x-c_{\mathrm{v}}^{\mathrm{i}}\right)^2}{\sigma_{\mathrm{v}}^{\mathrm{i},+}}} U(x;c_{\mathrm{v}}^{\mathrm{i}},\infty) \text{ si j} = \text{i} + 1 \end{cases} \tag{A.13}$$

A.2.3. Funciones pseudo-gaussianas libres (PGL)

Esta es una configuración equivalente a las funciones triangulares libres. Ahora los parámetros de las funciones pseudo-gaussianas del apartado anterior se dejan libres dotando a las funciones de pertenencia de un número mayor de grados de libertad. Al igual que en el caso anterior, la función de pertenencia X_v^i de la variable v vendrá dada por tres parámetros: su centro c_v^i y sus desviaciones izquierda y derecha $\sigma_v^{i,-}$ y $\sigma_v^{i,+}$. En la siguiente figura se representa un ejemplo de dichas funciones de pertenencia:

El grado de pertenencia de la entrada x a la función de pertenencia X_v^i vendrá dado por:

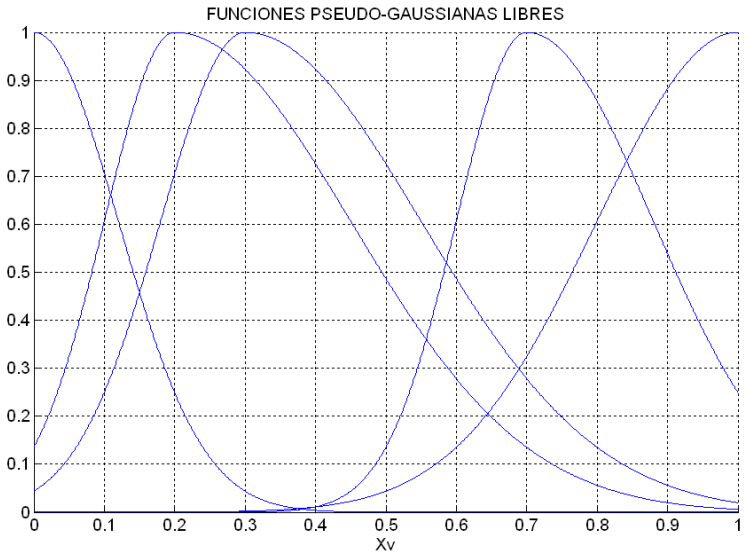


Figura A.4. *Funciones pseudo-gaussianas libres.*

$$\mu_{X_v^i}(x) = e^{-\frac{\left(x-c_v^i\right)^2}{\sigma_v^{i,-}}} U(x; -\infty, c_v^i) + e^{-\frac{\left(x-c_v^i\right)^2}{\sigma_v^{i,+}}} U(x; c_v^i, \infty) \tag{A.14}$$

Las derivadas de las funciones de pertenencia así definidas con respecto a sus parámetros serían:

$$\frac{\partial \mu_{X_v^i}(x)}{\partial c_v^i} = 2\frac{x-c_v^i}{\sigma_v^{i,-}} e^{-\frac{\left(x-c_v^i\right)^2}{\sigma_v^{i,-}}} U(x; -\infty, c_v^i) + 2\frac{x-c_v^i}{\sigma_v^{i,+}} e^{-\frac{\left(x-c_v^i\right)^2}{\sigma_v^{i,+}}} U(x; c_v^i, \infty) \tag{A.15}$$

$$\frac{\partial \mu_{X_v^i}(x)}{\partial \sigma_v^{i,-}} = 2\left(\frac{x-c_v^i}{\sigma_v^{i,-}}\right)^2 e^{-\frac{\left(x-c_v^i\right)^2}{\sigma_v^{i,-}}} U(x; -\infty, c_v^i) \tag{A.16}$$

$$\frac{\partial \mu_{X_v^i}(x)}{\partial \sigma_v^{i,+}} = \left(\frac{x-c_v^i}{\sigma_v^{i,+}}\right)^2 e^{-\frac{\left(x-c_v^i\right)^2}{\sigma_v^{i,+}}} U(x; c_v^i, \infty) \tag{A.17}$$

A.3. Bases OLMF o Particiones OLMF

En el capítulo 4, en la sección 4.3.2 se presentaban los requisitos que deben cumplir las bases OLMF para cumplir con las condiciones de interpretabilidad y modelado local del modelo TaSe. Aquí se repasa la formulación de este tipo de funciones de pertenencia, mostrándose la forma que presentan las particiones de funciones de este tipo.

Dados los parámetros $[a, b, c]$ de una función de pertenencia local, donde a y c son los extremos de la FP y b es el centro de la FP, para $p = 2$, esto es, bases de orden dos, la formulación de la función de pertenencia tipo OLMF $\mu_X(x, [a, b, c])$ se obtiene utilizando

la Interpolación de Hermite. Definimos la función $OLMF(x, a, b)$, que define la activación de la función para un punto entre el extremo "izquierdo" y el centro, que viene dada por

$$OLMF(x,a,b) = \frac{1}{(b-a)^3}(x-a)^3\left[1 - \frac{3(x-b)}{(b-a)} + \frac{6(x-b)^2}{(b-a)^2}\right] si\ x\ \in [a,b] \tag{A.18}$$

La función de pertenencia $\mu_{X^{s'}}(x)$ para un x dado viene por tanto dada por

$$\mu_{X_v^i}(x) = \begin{cases} U(x;-\infty,c_v^i) + OLMF(x,c_v^i,c_v^{i+1})U(x;c_v^i,c_v^{i+1}) \text{ si i} = 1 \\ OLMF(x,c_v^{i-1},c_v^i)U(x;c_v^{i-1},c_v^i) + OLMF(x,c_v^i,c_v^{i+1})U(x;c_v^i,c_v^{i+1}) \text{ si i} \neq 1 \text{ei} \neq \text{n} \\ OLMF(x,c_v^{i-1},c_v^i)U(x;c_v^{i-1},c_v^i) + U(x;c_v^i,\infty) \text{ si i} = \text{n} \end{cases} \tag{A.19}$$

La figura A.5 muestra un ejemplo gráfico de este tipo de FP. Puede observarse que aparte de la continuidad y derivabilidad de la función, el gradiente de la FP se anula en el centro y en los extremos del intervalo donde está definido.

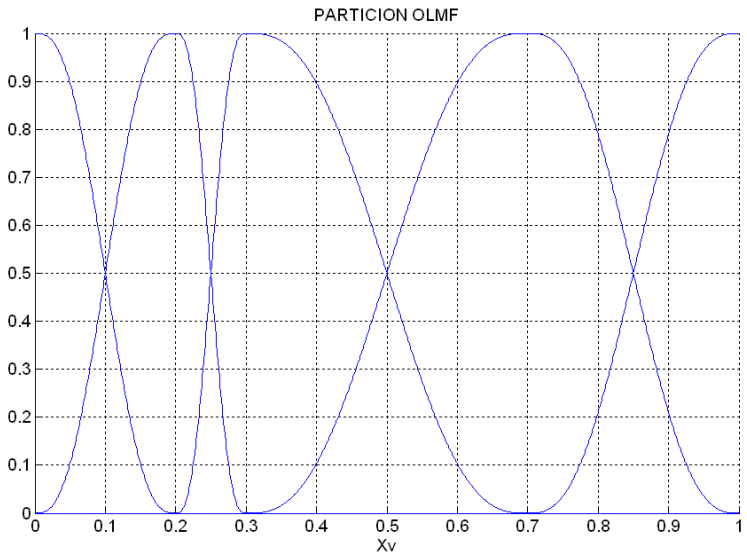


Figura A.5. *Partición OLMF*

Apéndice B

Cálculo de Derivadas Parciales para el Cálculo de los Coeficientes Óptimos de los Consecuentes Usando Mínimos Cuadrados

Se presentan ahora las derivadas parciales de la función de error

$$J = \sum_{m \in D} \left(y^m - F\left(\vec{x}^m\right)\right)^2, \tag{B.1}$$

con respecto a cada coeficiente de los consecuentes de las reglas. Utilizando las expresiones para todas los coeficientes, se obtiene un conjunto de ecuaciones lineales que proporcionan los valores óptimos de dichos coeficientes para un conjunto de datos dado D. La expresión genérica para cualquier coeficiente de orden 0 (w_0), orden 1 ($w_{v1}, \ldots, w_{vn}$) o de orden 2 ($w_{v11}, \ldots, w_{vnn}$) es

$$\begin{array}{c} \frac{\partial J}{\partial w_s^j} = 2 \sum\limits_{m \in D} \left(y^m - \frac{\sum\limits_{k=1}^{K} \mu_k(\vec{x}^m)\left(w_0^k + \vec{w}_1^k \cdot \vec{x}^m + \frac{1}{2}(\vec{x}^m)^T W_2^k \vec{x}^m\right)}{\sum\limits_{k=1}^{K} \mu_k(\vec{x}^m)} \right) \\ \frac{\mu_j(\vec{x}^m) \cdot f_{w_s^j}(\vec{x}^m)}{\sum\limits_{k=1}^{K} \mu_k(\vec{x}^m)}; \quad s = \{0, v_1, ...v_n, v_{11}, ..., v_{nn}\}, \ k = 1..K \end{array} \tag{B.2}$$

Siendo $f_{w_s^j}(\vec{x}^m)$ igual a 1 para $s = 0$, $x_{v_l}^m$ para $s = v_l$ y $x_{v_l}^m \cdot x_{v_h}^m$ para $s = v_{lh}$. Así, las derivadas parciales llevan a los tres siguientes tipos de ecuaciones

$$\begin{array}{c} for s = 0, \ \frac{\partial J}{\partial w_s^j} = 0 \ \Rightarrow \\ \sum\limits_{k=1}^{K} \sum\limits_i w_i^k \sum\limits_{m \in D} \frac{\mu_k(\vec{x}^m) \cdot f_{w_i^k}(\vec{x}^m) \cdot \mu_j(\vec{x}^m)}{\sum\limits_{j=1}^{K} \mu_j(\vec{x}^m)} = \sum\limits_{m \in D} \frac{y^m \cdot \mu_j(\vec{x}^m)}{\sum\limits_{k=1}^{K} \mu_k(\vec{x}^m)} \end{array} \tag{B.3}$$

$$\begin{array}{c} for s = v_l, \ \frac{\partial J}{\partial w_s^j} = 0 \ \Rightarrow \\ \sum\limits_{k=1}^{K} \sum\limits_i w_i^k \sum\limits_{m \in D} \frac{\mu_k(\vec{x}^m) \cdot f_{w_i^k}(\vec{x}^m) \cdot \mu_j(\vec{x}^m) \cdot x_{v_l}^m}{\sum\limits_{j=1}^{K} \mu_j(\vec{x}^m)} = \sum\limits_{m \in D} \frac{y^m \cdot \mu_j(\vec{x}^m) \cdot x_{v_l}^m}{\sum\limits_{k=1}^{K} \mu_k(\vec{x}^m)} \end{array} \tag{B.4}$$

$$\begin{array}{c} for s = v_{lh}, \ \frac{\partial J}{\partial w_s^j} = 0 \ \Rightarrow \\ \sum\limits_{k=1}^{K} \sum\limits_i w_i^k \sum\limits_{m \in D} \frac{\mu_k(\vec{x}^m) \cdot f_{w_i^k}(\vec{x}^m) \cdot \mu_j(\vec{x}^m) \cdot x_{v_l}^m \cdot x_{v_h}^m}{\sum\limits_{j=1}^{K} \mu_j(\vec{x}^m)} = \sum\limits_{m \in D} \frac{y^m \cdot \mu_j(\vec{x}^m) \cdot x_{v_l}^m \cdot x_{v_h}^m}{\sum\limits_{k=1}^{K} \mu_k(\vec{x}^m)} \end{array} \tag{B.5}$$

Para la optimización de las posiciones de los centros de las FPs, se pueden aplicar métodos de descenso en gradiente para obtener un óptimo local a partir de una configuración inicial de os centros de las FPs. Así pues, es necesario calcular la variación de la función de error J, con respecto a la posición de cada centro de FP. Con este propósito deben encontrarse

$$\frac{\partial J}{\partial c_p^{i_p}} = -2 \sum_{m \in D} \left[\left(y^m - F\left(\vec{x}^m\right) \right) \cdot \left(\frac{\partial F\left(\vec{x}^m\right)}{\partial c_p^{i_p}} \right) \right] \tag{B.6}$$

A partir de la función de salida del sistema difuso se tiene que

$$\begin{gathered} \frac{\partial F(\vec{x}^m)}{\partial c_p^{i_p}} = \frac{\partial}{\partial c_p^{i_p}} \left(\frac{\sum_{k=1}^{K} \mu_k(\vec{x}^m) Y_k(\vec{x}^m)}{\sum_{k=1}^{K} \mu_k(\vec{x}^m)} \right) = \\ \frac{\partial}{\partial c_p^{i_p}} \left(\sum_{k=1}^{K} \mu_k\left(\vec{x}^m\right) Y_k\left(\vec{x}^m\right) \right) = \sum_{k=1}^{K} Y_k\left(\vec{x}^m\right) \left(\frac{\partial \mu_k(\vec{x}^m)}{\partial c_p^{i_p}} \right) \end{gathered} \tag{B.7}$$

En el caso de utilizar funciones de pertenencia organizadas como partición, la derivada parcial de la función de activación de cada regla, puesto que las funciones de pertenencia de la variable p dependen en el centro $c_p^{i_p}$, queda como

$$\frac{\partial \mu_k\left(\vec{x}^m\right)}{\partial c_p^{i_p}} = \frac{\partial \prod_{j=1}^{n} \mu_k^j\left(x_j^m\right)}{\partial c_p^{i_p}} = \frac{\partial \mu_k^p\left(x_p^m\right)}{\partial c_p^{i_p}} \prod_{\substack{j=1 \\ j \neq p}}^{n} \mu_k^j\left(x_j^m\right) \tag{B.8}$$

En el caso de utilizar las funciones de pertenencia definidas en el sistema TaSe según las ecuaciones 4.14 y 4.15, se puede comprobar que

$$\frac{\partial \mu_k^p\left(x_p^m\right)}{\partial c_p^{i_p}} = \begin{cases} \frac{30 \cdot (x-a)^2 \cdot (x-b)^3}{(b-a)^6} & si\ x_p^m \in [a,b] \ \ y\ c_p^{i_p} = a \\ & in\ MF_p^{i_p+1}\left(x, [a,b,c]\right) \\ -\frac{30 \cdot (x-a)^3 \cdot (x-b)^2}{(b-a)^6} & si\ x_p^m \in [a,b] \ \ y\ c_p^{i_p} = b \\ & in\ MF_p^{i_p}\left(x, [a,b,c]\right) \\ \frac{30 \cdot (x-b)^3 \cdot (x-c)^2}{(c-b)^6} & si\ x_p^m \in [b,c] \ \ y\ c_p^{i_p} = b \\ & in\ MF_p^{i_p}\left(x, [a,b,c]\right) \\ -\frac{30 \cdot (x-b)^2 \cdot (x-c)^3}{(c-b)^6} & si\ x_p^m \in [b,c] \ \ y\ c_p^{i_p} = c \\ & in\ MF_p^{i_p-1}\left(x, [a,b,c]\right) \end{cases} \tag{B.9}$$

Debe notarse finalmente que los centros que se localizan en los extremos del dominio de cada variable, en los sistemas basados en grid, se consideran fijos, al depender sus posiciones exclusivamente de los valores máximo y mínimo del rango de cada variable.

Bibliografía

[CAT,] Cats benchmark. http://www.cis.hut.fi/ lendasse/competition/competition.html.

[EUN,] Eunite competition 2003: Prediction of product quality in glass manufacturing, oulu, finland. http://www.eunite.org.

[LS-,] Ls-svmlab: a matlab/c toolbox for least squares support vector machines. http://www.esat.kuleuven.ac.be/sista/lssvmlab.

[MIL,] Mutual information least dependent component analysis. http://www.klab.caltech.edu/ kraskov/MILCA/.

[Tim,] Time series data library. http://iis.edu.tama.ac.jp/Delphi/Sample/hydrology.html.

[Alsina et al., 1983] Alsina, C., Trillas, E., and Valverde, L. (1983). On some logical connectives for fuzzy sets theory. *Journal of mathematical analysis and applications*, 93:15–26.

[Angelov and Filev, 2004] Angelov, P. and Filev, D. (2004). An approach to online identification of takagi-sugeno fuzzy models. *IEEE Transactions on Systems, Man, and Cybernetics.*, 34(1):484– 498.

[Awad, 2005] Awad, M. (2005). *Modelado de sistemas complejos mediante estructuras jerárquicas de redes de funciones de base radial*. PhD thesis, Departamento ATC, Universidad de Granada.

[Barbounis and Theocharis, 2006] Barbounis, T. and Theocharis, J. (2006). Locally recurrent neural networks for long-term wind speed and power prediction. *Neurocomputing*, 69:446–496.

[Battiti, 1994] Battiti, R. (1994). Using mutual information for selecting features in supervised neural net learning. *IEEE Trans. Neural Net.*, 5(4):537–550.

[Bellman, 1961] Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University.

[Benoudjit et al., 2004] Benoudjit, N., François, D., Meurens, M., and Verleysen, M. (2004). Spectrophotometric variable selection by mutual information. *Chem. and Int. Lab. Syst.*, 74:243–251.

[Bikdash, 1999] Bikdash, M. (1999). A highly interpretable form of sugeno inference systems. *IEEE Trans. Fuz. Syst.*, 7(6):686–696.

[Bonnlander and Weigend, 1994] Bonnlander, B. and Weigend, A. (1994). Selecting input variables using mutual information and nonparametric density estimation. In *Proc. of the ISANN, Taiwan*, pages 42–50.

[Box, 1989] Box, G. (1989). *Estadística para investigadores. Introducción al diseño de experimentos, análisis de datos y construcción de modelos.* Editorial Reverté.

[Box et al., 1978] Box, G., Hunter, W., and Hunter, J. (1978). *Statistics for Experimenters. An Introduction to Design, Data Analysis and Model Building.* Wiley, New York.

[Brockman and Huwendiek, 2000] Brockman, W. and Huwendiek, O. (2000). Neuro-fuzzy approach for modeling complex functional mappings. *Proc. 2000 IEEE Int. Conf. on Systems Man and Cybernetics - SMC2000*, pages 3734–3739.

[Broomhead and Lowe, 1988] Broomhead, D. and Lowe, D. (1988). Multivariate functional interpolation and adaptive networks. *Complex Systems*, 2:321 – 355.

[Buckley, 1992] Buckley, J. (1992). Universal fuzzy controllers. *Automatica*, 28(6):1245–1248.

[Buckley, 1993] Buckley, J. (1993). Sugeno type controllers are universal controllers. *Automatica*, 53:299–304.

[Buckley and Hayashi, 1994] Buckley, J. and Hayashi, Y. (1994). Can fuzzy neural networks approximate continuous fuzzy functions? *Fuzzy Sets and Systems*, 61:43–52.

[Cao and Kandel, 1992] Cao, Z. and Kandel, A. (1992). Investigations on the applicability of fuzzy inference. *Fuzzy Sets and Systems*, 49:151–169.

[Cárdenas et al., 1993] Cárdenas, E., Castillo, J., Cordón, O., Herrera, F., and Peregrín, A. (1993). Influence of fuzzy implication functions and defuzzification methods in fuzzy control. *BUSEFAL*, 57:69–79.

[Casillas et al.,] Casillas, J., Cordon, O., del Jesus, M. J., and Herrera, F. Tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction. *IEEE Trans. Fuzzy Syst.*, 13(1).

[Casillas et al., 2003] Casillas, J., Cordon, O., Herrera, F., and Magdalena, L. (2003). *Interpretability Issues in Fuzzy Modeling*. Springer-Heidelberg, Germany.

[Castro, 1995] Castro, J. (1995). Fuzzy logic controllers are universal approximators. *IEEE Trans. Syst. Man and Cyber.*, 25(4):629–635.

[Castro and Delgado, 1996] Castro, J. and Delgado, M. (1996). Fuzzy systems with defuzzification are universal approximators. *IEEE Trans. Syst. Man and Cyber.*, 26(1):149–152.

[Chang et al., 2001] Chang, M., Chen, B., and Lin, C. (2001). Eunite network competition: Electricity load forecasting.

[Cherkassky et al., 1996] Cherkassky, V., Gehring, D., and Mulier, F. (1996). Comparison of adaptive methods for function estimation from samples. *IEEE Trans. on Neural Networks*, 7(4):969–984.

[Cho and Wang, 1995] Cho, K. and Wang, B. (1995). Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction. *Fuzzy Sets and Systems*, 83:325–339.

[Chow and Huang, 2005] Chow, T. W. S. and Huang, D. (2005). Estimating optimal feature subsets using efficient estimation of high-dimensional mutual information. *IEEE Trans. Neural Net.*, 16(1):213–224.

[Chung, 2000] Chung, F. (2000). On multistage fuzzy neural network modeling. *IEEE Trans. Fuzz. Syst.*, 8(2):125–142.

[Cordon et al., 2004] Cordon, O., Herrera, F., Gomide, F., Hoffmann, F., and Magdalena, L. (2004). Ten years of genetic-fuzzy systems: a current framework and new trends. *Fuzzy Sets and Systems*, 141(1):5–31.

[Cordon et al., 2001] Cordon, O., Herrera, F., Hoffmann, F., and Magdalena, L. (2001). *Genetic Fuzzy Systems. Evolutionary tuning and learning of fuzzy knowledge bases*. Advances in Fuzzy Systems: Applications and Theory. World Scientific.

[Cordón et al., 1997] Cordón, O., Herrera, F., and Peregrin, A. (1997). Applicability of the fuzzy operators in the design of fuzzy logic controllers. *Fuzzy Sets and Systems*, (86):15–41.

[Cover and Thomas, 1991] Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. Wiley, New York.

[de Souza et al., 2002] de Souza, F., Vellasco, M., and Pacheco, M. (2002). Hierarchical neuro-fuzzy quadtree models. *Fuzzy Sets & Systems*, 130(2):189–205.

[Decoste and Schoelkopf, 2002] Decoste, D. and Schoelkopf, B. (2002). Training invariant support vector machines. *Machine Learning*, 46:161–190.

[Degroot, 1988] Degroot, M. (1988). *Probabilidad y estadística*. Addison-Wesley Iberoamericana, Segunda edición.

[Dijck and Hulle, 2006] Dijck, G. and Hulle, M. (2006). Speeding up the wrapper feature subset selection in regression by mutual information relevance and redundancy analysis. *Lecture Notes in Computer Science*, 4131:31–40.

[Driankov et al., 1993] Driankov, D., Hellendoorn, H., and Reinfrank, M. (1993). *An introduction to fuzzy control*. Springer-Verlag.

[Duda and Hart, 1973] Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. New York: Wiley.

[et al., 1992] et al., B. B. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 5:144–152.

[Filev and Yager, 1993] Filev, D. and Yager, R. (1993). An adaptive approach to defuzzification based on level sets. *Fuzzy Sets and Systems*, 54:355–360.

[Fisher, 1936] Fisher, R. (1936). The comparison of samples with possibly unequal variances. *Annals of Eugenics*, 9:174–180.

[Fisher, 1950] Fisher, R. (1950). *Contribution to Mathematical Statistics*. New York, Wiley.

[François et al., 2007] François, D., Rossi, F., Wertz, V., and Verleysen, M. (2007). Resampling methods for parameter-free and robust feature selection with mutual information. *Neurocomputing*, 70:1276.1288.

[François et al., 2006] François, D., Wertz, V., and Verleysen, M. (2006). The permutation test for feature selection by mutual information. *ESANN'2006 proceedings - European Symposium on Artificial Neural Networks*, pages 239–244.

[Golub and Loan, 1961] Golub, G. and Loan, C. (1961). *Matrix Computations.* The Johns Hopkins University Press.

[Gonzalez et al., 2002] Gonzalez, J., Rojas, I., Pomares, H., Ortega, J., and Prieto, A. (2002). A new clustering technique for function approximation. *IEEE Trans. Neu. Net.*, 13(1):132–142.

[Good, 1994] Good, P. (1994). *Permutation Tests*. Springer, New York.

[Guillaume, 2001] Guillaume, S. (2001). Designing fuzzy inference systems from data: an interpretability-oriented review. *IEEE Trans. Fuz. Syst.*, 9:426–443.

[Guillén et al., 2007] Guillén, A., Rojas, I., González, J., Pomares, H., and Herrera, L. (2007). Output value-based initialization for radial basis function neural networks. *Neural Processing Letters*, Aceptado.

[Gupta and Qi, 1991] Gupta, M. and Qi, J. (1991). Theory of t-norms and fuzzy inference methods. *Fuzzy Sets and Systems*, 40:431–450.

[Harald et al., 2004] Harald, S., Alexander, K., Sergey, A., and Peter, G. (2004). Least dependent component analysis based on mutual information. *Phys.Rev.*, E 70:066123.

[Haykin, 1998] Haykin, S. (1998). *Neural Networks*. Prentice Hall.

[Hellendoorn and Thomas, 1993] Hellendoorn, H. and Thomas, C. (1993). The -quality defuzzification method. *Fifth IFSA World Congress*, pages 1159–1162.

[Herrera et al., 2004a] Herrera, L., Pomares, H., Rojas, I., Gonzalez, J., Awad, M., and Herrera, A. (2004a). Multigrid-based fuzzy systems for time series prediction: Cats competition. *Proceedings of the IEEE International Joint Conference on Neural Network*, pages 1603–1608.

[Herrera et al., 2004b] Herrera, L., Pomares, H., Rojas, I., González, J., Valenzuela, O., and Awad, M. (2004b). Function approximation through fuzzy systems using taylor series expansion-based rules, interpretability and parameter tuning. *Lect. Notes Comput. Sci.*, 2972:508–516.

[Herrera et al., 2005a] Herrera, L., Pomares, H., Rojas, I., and Guillén, A. (2005a). New clustering technique for inicialization of centres in tsk clustering-based fuzzy systems. *Lecture Notes in Computer Science*, 3571:980–991.

[Herrera et al., 2005b] Herrera, L., Pomares, H., Rojas, I., Guillén, A., Awad, M., and González, J. (2005b). Interpretable rule extraction and function approximation from numerical input/output data using the modified fuzzy tsk model: Tase model. *Lecture Notes in Computer Science*, 3641:402–411.

[Herrera et al., 2005c] Herrera, L., Pomares, H., Rojas, I., Guillén, A., Awad, M., and Valenzuela, O. (2005c). Analysis of the tase-ii tsk-type fuzzy system for function approximation. *Lecture Notes in Computer Science*, 3571:613–624.

[Herrera et al., 2005d] Herrera, L., Pomares, H., Rojas, I., Guillén, A., González, J., and Awad, M. (2005d). Clustering-based tsk neuro-fuzzy model for function approximation with interpretable sub-models. *Lecture Notes in Computer Science*, 3512:399–406.

[Herrera et al., 2007a] Herrera, L., Pomares, H., Rojas, I., Guillén, A., González, J., Awad, M., and Herrera, A. (2007a). Multigrid-based fuzzy systems for time series prediction: Cats competition. *Neurocomputing*, 70:2410.2425.

[Herrera et al., 2007b] Herrera, L., Pomares, H., Rojas, I., Guillén, A., Prieto, A., and Valenzuela, O. (2007b). Recursive prediction for long term time series forecasting using advanced models. *Neurocomputing*, page ACCEPTED.

[Herrera et al., 2005e] Herrera, L., Pomares, H., Rojas, I., Guillén, A., Valenzuela, O., and Prieto, A. (2005e). Tase model for long term time series forecasting. *Lecture Notes in Computer Science*, 3512:1027–1034.

[Herrera et al., 2004c] Herrera, L., Pomares, H., Rojas, I., Valenzuela, O., and Awad, M. (2004c). Multigrid-based fuzzy systems for function approximation. *Lect. Notes Comput. Sci.*, 2972:252–261.

[Herrera et al., 2004d] Herrera, L., Pomares, H., Rojas, I., Valenzuela, O., Gonzalez, J., and Awad, M. (2004d). Multigrid-based fuzzy systems for time series forecasting: Overcoming the curse of dimensionality. *Proceedings of the European Symposium on Neural Networks*, pages 127–132.

[Herrera et al., 2005f] Herrera, L., Pomares, H., Rojas, I., Valenzuela, O., and Prieto, A. (2005f). Tase, a taylor series based fuzzy system model that combines interpretability and accuracy. *Fuzzy Sets and Systems*, 153:403–427.

[Herrera et al., 2006] Herrera, L., Pomares, H., Rojas, I., Verleysen, M., and Guillén, A. (2006). Effective input variable selection for function approximation. *Lecture Notes in Computer Science*, 4131:41–40.

[Hornick et al., 1989] Hornick, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

[Jager, 1995] Jager, R. (1995). *Fuzzy Logic in Control*. Tesis Doctoral, Amsterdam.

[Jang et al., 1997] Jang, J., Sun, C., and Mizutani, E. (1997). *Neuro-Fuzzy and Soft Computing*. Prentice-Hall, Englewood cliffs, NJ.

[Jang and Sun, 1993] Jang, J. S. R. and Sun, C.-T. (1993). Functional equivalence between radial basis function networks and fuzzy inference systems. *IEEE Transactions on Neural Networks*, 4(1):156–159.

[Jin and Sendhoff, 2003] Jin, Y. and Sendhoff, B. (2003). Extracting interpretable fuzzy rules from rbf networks. *Neural Processing Letters*, 17:149.164.

[Johansen and Babuska, 2003] Johansen, T. and Babuska, R. (2003). Multiobjective identification of takagi-sugeno fuzzy models. *IEEE Trans. Fuz. Syst.*, 11(6):847–860.

[Jung et al., 2005] Jung, T., Herrera, L., and Schoelkopf, B. (2005). Long term prediction of product quality in a glass manufacturing process using a kernel based approach. *Lecture Notes in Computer Science*, 3512:960–967.

[Kim and Kim, 1997] Kim, D. and Kim, C. (1997). Forecasting time series with genetic fuzzy predictor ensemble. *IEEE Trans. Fuzz. Syst.*, 5(4):523–535.

[Kim et al.,] Kim, M. S., Kim, C. H., and Lee, J. J. Interpretable takagi sugeno fuzzy models with a new encoding scheme. *IEEE Trans. Syst., Man, Cybern.*, 36(5).

[Koller and Sahami, 1996] Koller, D. and Sahami, M. (1996). Toward optimal feature selection. In *Proc. Int. Conf. on Machine Learning*, pages 284–292.

[Kosko, 1992a] Kosko, B. (1992a). *Neural Networks and Fuzzy Systems*. Englewood Cliffs, NJ, Prentice Hall.

[Kosko, 1992b] Kosko, B. (1992b). *Neural Networks and Fuzzy Systems*. Englewood Cliffs, NJ, Prentice Hall.

[Kosko, 1994] Kosko, B. (1994). Fuzzy systems as universal approximators. *IEEE Trans. Computers*, 43(1):1329–1333.

[Kovalerchuk and Taliansky, 1992] Kovalerchuk, B. and Taliansky, V. (1992). Comparison of empirical and computed values of fuzzy conjunction. *Fuzzy Sets and Systems*, 46:49–53.

[Kraskov et al., 2004] Kraskov, A., Stogbauer, H., and Grassberger, P. (2004). Estimating mutual information. *Phys.Rev.*, E 69:066138.

[Kruse et al., 1994] Kruse, R., Gebhardt, J., and Klawonn, F. (1994). *Foundations of fuzzy systems*. John Wiley & Sons.

[Lee, 1990] Lee, C. (1990). Fuzzy logic in control systems: fuzzy logic controller - part i,ii. *IEEE Trans. Syst. Man and Cyber.*, 20(2):404–435.

[Lendasse et al., 2005] Lendasse, A., Ji, Y., Reyhani, N., and Verleysen, M. (2005). Ls-svm hyperparameter selection with a nonparametric noise estimator. *Lecture Notes in Computer Science*, 3697:625–630.

[Lendasse et al., 2004] Lendasse, A., Oja, E., Simula, O., and Verleysen, M. (2004). Time series prediction competition: The cats benchmark. *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 1615–1620.

[Mabuchi, 1993] Mabuchi, S. (1993). A proposal for a defuzzification strategy by the concept of sensitivity analysis. *Fuzzy Sets and Systems*, 55:1–14.

[Mackey and Glass, 1977] Mackey, M. and Glass, L. (1977). Oscillation and chaos in physcological controlsystems. *Science*, 197:287–289.

[Mamdani, 1974] Mamdani, E. (1974). Application on fuzzy algorithms for the control of a dynamic plant. *Proc. IEEE*, 121(12):1585–1588.

[Mao, 2005] Mao, K. (2005). Identifying critical variables of principal components for unsupervised feature selection. *IEEE Trans. Syst., Man, Cybern. B*, 35(2):339–344.

[McNames et al., 1999] McNames, J., Suykens, J., and Vandewalle, J. (1999). Winning entry of the k.u. leuven time-series prediction competition. *International Journal of Bifurcation and Chaos*, 9(8):1485–1500.

[Mendel, 1995] Mendel, J. (1995). Fuzzy logic systems for engineering. *IEEE Proceeding*, 83(3):345–377.

[Mizumoto, 1989] Mizumoto, M. (1989). Pictorial representations of fuzzy connectives, part i : cases of t-norms, t-conorms and averaging operators. *Fuzzy Sets and Systems*, 31:217–242.

[Montgomery, 1991] Montgomery, D. (1991). *Diseño y análisis de experimentos*. Grupo Editorial Iberoamérica.

[Moody and Darken, 1988] Moody, J. and Darken, C. (1988). Learning with localized receptive fields. *Touretzsky,D., Hinton,G.,& Sejnowsk, T.(Eds.) Proceedings*, pages 133–143.

[Moody and Darken, 1989] Moody, J. and Darken, C. (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294.

[More, 1978] More, J. (1978). The levenberg-marquardt algorithm implementation and theory. *Lecture Notes in Mathemattcs*, (630):105–116.

[Nie and Lee, 1996] Nie, J. and Lee, T. (1996). Rule-based modelling: Fast construction and optimal manipulation. *IEEE Trans. on Syst. Man and Cyber Part A*, 26(6):728–738.

[Ning et al., 2003] Ning, Z., Ong, Y. S., Wong, K. W., and Seow, K. T. (2003). Parameter identification using memetic algorithms for fuzzy systems. *Fourth International Conference on Intelligent Technologies*.

[Paiva and Dourado, 2001] Paiva, R. and Dourado, A. (2001). Structure and parameter learning of neuro-fuzzy systems: A methodology and a comparative study. *Journal of Intelligent and Fuzzy Systems*, 11:147–161.

[Paiva and Dourado, 2004] Paiva, R. and Dourado, A. (2004). Interpretability and learning in neuro-fuzzy systems. *Fuzzy Sets and Systems*, 147:17–38.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, CA.

[Piskunov, 1992] Piskunov, A. (1992). Fuzzy implication in fuzzy systems control. *Fuzzy Sets and Systems*, 45:25–35.

[Pomares, 2000] Pomares, H. (2000). *Nueva metodología para el diseño automático de sistemas difusos*. PhD thesis, Departamento ATC, Universidad de Granada.

[Pomares et al., 2002] Pomares, H., Rojas, I., González, J., and Prieto, A. (2002). Structure identification in complete rule-based fuzzy systems. *IEEE Trans. Fuzz. Syst.*, 10(3):349–359.

[Pomares et al., 2000] Pomares, H., Rojas, I., Ortega, J., Gonzalez, J., and Prieto, A. (2000). A systematic approach to a self-generating fuzzy rule-table for function approximation. *IEEE Trans Syst., Man and Cyber.*, 30(3):431–447.

[Powell, 1987] Powell, M. (1987). Radial basis functions for multivariate interpolation: a review. *Neural Computation*, 3:213–225.

[Reddy and Babu, 1992] Reddy, P. and Babu, M. (1992). Some methods of reasoning for fuzzy conditional propositions. *Fuzzy Sets and Systems*, 52.

[Reyhani et al., 2005] Reyhani, M.Ñ., Hao, J., Ji, Y., and Lendasse, A. (2005). Mutual information and gamma test for input selection. *European Symposium on Artificial Neural Networks 2005*, pages 503–504.

[Rojas et al., 2002] Rojas, I., Pomares, H., Bernier, J., Ortega, J., Pino, B., Pelayo, F., and Prieto, A. (2002). Time series analysis using normalized pg-rbf network with regression weights. *NeuroComputing*, 42:267–285.

[Rojas et al., 2000] Rojas, I., Pomares, H., Ortega, J., and Prieto, A. (2000). Self-organized fuzzy system generation from training examples. *IEEE Trans. Fuzz. Syst.*, 8(1):23–36.

[Rojas et al., 1998] Rojas, I., Valenzuela, O., Anguita, M., and Prieto, A. (1998). Analysis of the operators involved in the definition of the implication functions and in the fuzzy inference process. *International Journal of Approximate Reasoning*, (19):367–389.

[Rossi et al., 2006] Rossi, F., Lendasse, A., François, D., Wertz, V., and Verleysen, M. (2006). Mutual information for the selection of relevant variables in spectrometric nonlinear modelling. *Chemometrics and Intelligent Laboratory Systems*, 80:215–226.

[Rovatti and Guerrieri, 1996] Rovatti, R. and Guerrieri, R. (1996). Fuzzy sets of rules for system identification. *IEEE Trans. Fuzzy Syst.*, 4(2):89–102.

[Ruspini, 1969] Ruspini, E. (1969). A new approach to clustering. *Info Control*, (15):22–32.

[Russo and Patane, 1999] Russo, M. and Patane, G. (1999). Improving the lbg algorithm. *Lecture Notes in Computer Science*, 1606:621–630.

[Schoelkopf and Smola, 2002] Schoelkopf, B. and Smola, A. (2002). *Learning with Kernels*. Cambridge, MA: MIT Press.

[Setnes et al., 1998] Setnes, M., Babuska, R., Kaymak, U., and van Nauta Lemke, H. (1998). Similarity measures in fuzzy rule base simplification. *IEEE Trans. on Syss, Man, and Cyber*, 28(3):376–386.

[Sorjamaa et al., 2005] Sorjamaa, A., Hao, J., and Lendasse, A. (2005). Mutual information and k-nearest neighbors approxi-mator for time series prediction. *LNCS*, 3697:553–558. ICANN'05.

[Sudkamp and Hammell, 1994] Sudkamp, T. and Hammell, R. (1994). Interpolation, completion and learning fuzzy rules. *IEEE Trans. Syst. Man and Cyber.*, 24(2):332–342.

[Sugeno, 1985a] Sugeno, M. (1985a). *Industrial applications of fuzzy control*. Amsterdam, North-Holland.

[Sugeno, 1985b] Sugeno, M. (1985b). An introductory survey of fuzzy control. *Information Sciences*, 36:59–83.

[Sugeno and Kang, 1988] Sugeno, M. and Kang, G. (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems*, 28:15–33.

[Suykens et al., 2002] Suykens, J., Gestel, T. V., Brabanter, J. D., Moor, J. D., and Vandewalle, B. (2002). *Least Squares Support Vector Machines*. World Scientific, Singapore.

[Takagi and Sugeno, 1985] Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its applications to modelling and control. *IEEE Trans. Syst. Man and Cyber.*, 15:116–132.

[Uykan et al., 2000] Uykan, Z., Gzelis, C., Celebei, M. E., and Koivo, H.Ñ. (2000). Analysis of input-output clustering for determining centers of rbfn. *IEEE Transactions on Neural Networks*, 11(4):851–858.

[Valenzuela et al., 2004] Valenzuela, O., Marquez, L., Pasadas, M., Rojas, I., Rodriguez, M., and Rojas, F. (2004). Analysis of the functional block and operator involved in fuzzy system design. *Lecture Notes in Computer Science*, (3040):157–166.

[Vapnik, 1999] Vapnik, V. (1999). *The Nature of Statistical Learning Theory*. Springer, 2nd edition edition.

[Wang, 1994] Wang, L. (1994). *Adaptive fuzzy systems and control. Design and stability analysis*. Englewood Cliffs, N.J. Prentice Hall,.

[Wang and Mendel, 1992a] Wang, L. and Mendel, J. (1992a). Fuzzy basis functions, universal approximation, and orthogonal least squares learning. *IEEE Trans. Neural Networks*, 3:807–813.

[Wang and Mendel, 1992b] Wang, L. and Mendel, J. (1992b). Generating fuzzy rules by learning from examples. *IEEE Trans. Syst. Man and Cyber.*, 22(6):1414–1427.

[Weigend and Gershenfeld, 1993] Weigend, A. and Gershenfeld, N. (1993). *Time Series Prediction: Forecasting the Future and Understanding the Past*. Addison-Wesley.

[Wu and Yu, 2000] Wu, B. and Yu, X. (2000). Fuzzy modelling and identification with genetic algorithm based learning. *Fuzzy Sets and Systems*, 113(3):351–365.

[Yen et al., 1998] Yen, J., Wang, L., and Gillespie, C. (1998). Improving the interpretability of tsk fuzzy models by combining global learning and local learning. *IEEE Trans. Fuzz. Syst.*, 6(4):530–537.

[Ying, 1998] Ying, H. (1998). General takagi-sugeno fuzzy systems are universal approximators. *IEEE World Congress on Computational Intelligence*, 1:819–823.

[Yongnan et al., 2005] Yongnan, J., Hao, J., Reyhani, N., and Lendasse, A. (2005). Direct and recursive prediction of time series using mutual information selection. *Lect. Notes Comput. Sci.*, 3512:1010–1017.

[Yu, 1999] Yu, W. (1999). Passive equivalence of chaos in lorenz system. *IEEE Trans. Circ. Syst. I: Fundamental Theory and Applications*, 46(7):876 – 878.

[Zadeh, 1965] Zadeh, L. (1965). Fuzzy sets. *Information and Control*, 8:338–353.

[Zadeh, 1973] Zadeh, L. (1973). Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Trans. Syst., Man and Cyber.*, 3:28–44.

[Zhao and Govind, 1991] Zhao, R. and Govind, R. (1991). Defuzzification of fuzzy intervals. *Fuzzy Sets and Systems*, 43:45–55.

[Zhou and Gan, 2004] Zhou, S. and Gan, J. (2004). Improving the interpretability of takagi-sugeno fuzzy model by using linguistic modifiers and a multiple objective learning scheme. *Int. Joint Conf. on Neural Networks IJCNN2004*, pages 2385–2390.

[Zhou et al., 2004] Zhou, X., Wang, X., Dougherty, E., and Suh, D. (2004). Gene clustering based on clusterwide mutual information. *J. Comput. Biol.*, 11(1):147–161.